This CVPR Workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Inferring Driving Maps by Deep Learning-based Trail Map Extraction

Michael Hubbertz[1,2] Pascal Colling[2] Qi Han[2] Tobias Meisen[1]

[1]*University of Wuppertal*, <lastname>@uni-wuppertal.de
[2]*Aptiv Services Deutschland GmbH*, <firstname>.<lastname>@aptiv.com

Abstract

High-definition (HD) maps offer extensive and accurate environmental information about the driving scene, making them a crucial and essential element for planning within autonomous driving systems. To avoid extensive efforts from manual labeling, methods for automating the map creation have emerged. Recent trends have moved from offline mapping to online mapping, ensuring availability and actuality of the utilized maps. While the performance has increased in recent years, online mapping still faces challenges regarding temporal consistency, sensor occlusion, runtime, and generalization. We propose a novel offline mapping approach that integrates trails — informal routes used by drivers — into the map creation process. Our method aggregates trail data from the ego vehicle and other traffic participants to construct a comprehensive global map using transformer-based deep learning models. Unlike traditional offline mapping, our approach enables continuous updates while remaining sensor-agnostic, facilitating efficient data transfer. Our method demonstrates superior performance compared to state-of-the-art online mapping approaches, achieving improved generalization to previously unseen environments and sensor configurations. We validate our approach on two benchmark datasets, highlighting its robustness and applicability in autonomous driving systems.

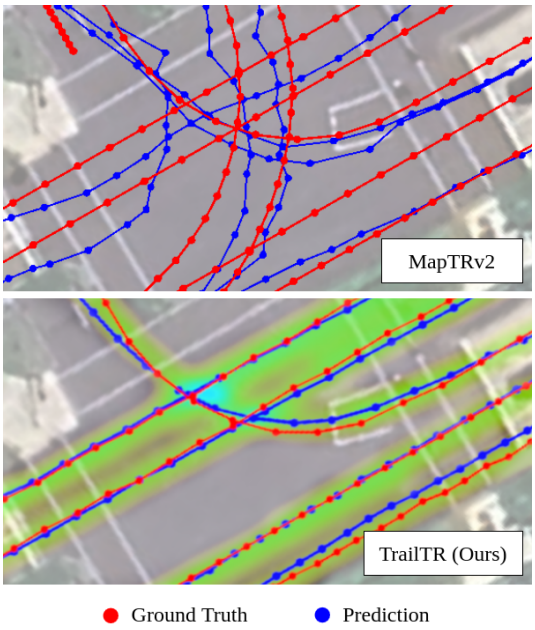


Figure 1. Visualization of the ground truth lane centerlines and predictions of a popular online mapping model (MapTRv2) and our approach (TrailTR) for a nuScenes validation sample. Both models were trained on the geographically disjoint near extrapolation split [23]. Centerlines with no corresponding trail data were removed from the ground truth in our approach.

1. Introduction

Maps are fundamental tools for navigation, providing essential spatial information that enables drivers to traverse complex road networks safely and efficiently. While standard definition (SD) maps are sufficient for human drivers, offering a general overview of road layouts, high-definition (HD) maps are becoming indispensable for autonomous vehicles. These maps provide fine-grained details, such as lane geometries and road surface attributes, which complement onboard sensor perception of the dynamic environment and ensure precise localization and safe navigation [1].

Traditionally, HD maps have been created through labor-intensive processes, including manual labeling and correction. However, these methods lack scalability, making automated HD map generation a critical research focus [14]. Thereby, two primary approaches have emerged: offline mapping and online mapping [1].

Offline mapping relies on pre-recorded data, such as aerial imagery or prior sensor observations, to construct a globally referenced, highly detailed map. While these maps offer high accuracy, they require continuous updates to ac-

count for road modifications, such as construction zones [31]. Furthermore, offline HD maps are limited to predefined regions, preventing autonomous vehicles that rely on map data from operating in unmapped areas.

To address these limitations, online mapping has gained increasing attention. Online approaches generate local, real-time maps by leveraging onboard sensor data, including camera and Lidar inputs [17, 20, 21, 25].

However, the created map based on current online sensor data can suffer from occlusion in certain scenarios, resulting in maps that are not completely covering the roads of the traffic situation. Recent research has also shown limitations regarding generalization to new areas of online mapping models [23]. Furthermore, online mapping approaches, which are often transformer-based, require high computational power for real-time execution [41], which is challenging due to the limited processing capacity of production vehicles. Additionally, high-resolution Lidar-based solutions, while providing detailed 3D representations, remain too costly for widespread adoption and are mainly used in research.

A promising approach to improving map creation is the integration of driving trails — frequently traveled routes derived from real-world traffic patterns. These trails capture informal driving behaviors that may not be easily inferred from static lane markings, providing valuable insights into human-like navigation strategies. By incorporating this information, autonomous vehicles can follow natural, behaviorally aligned paths, enhancing adaptability in diverse environments. This integration enables a more flexible and context-aware driving strategy, bridging the gap between conventional lane-based navigation and real-world driving behavior. [10]

In this paper, we propose an offline mapping approach that overcomes the classical limitations regarding coverage and required constant updates by incorporating trails paired with state-of-the-art transformer neural network architectures from the online mapping domain. Our method aggregates trajectory data from the ego vehicle and other traffic participants to construct a global HD map, leveraging deep learning to extract meaningful features from the collected trails. A key advantage of our approach is its sensor-agnostic nature — it does not rely on specific sensor configurations and can utilize data from any vehicle equipped with 3D object detection and global localization systems. This enables scalability and facilitates deployment across large fleets of vehicles. Furthermore, downstream applications such as motion prediction and path planning benefit from the enriched lane structure information [9, 36]. Unlike conventional mapping techniques that focus solely on static lane features, our approach captures human-like driving patterns, allowing autonomous systems to better align their trajectories with real-world driving behaviors.

Our key contributions can be summarized as follows:

- We introduce a novel crowdsourced offline mapping framework that extracts high-definition directed driving maps from recorded trail data using a state-of-the-art transformer-based deep learning model.
- We convert discrete trail data into a neural network input representation by applying preprocessing and data augmentation strategies. Our method enhances feature extraction and improves the model's robustness and generalization across diverse environments.
- Our approach serves as an alternative to existing online mapping approaches by offering improved accuracy and generalization and allows tile-wise map updates.

2. Related Work

Traditional offline map generation approaches: In the early stages of offline mapping, approaches primarily relied on aggregating pre-recorded data such as point clouds, aerial imagery, and GPS trajectories [1]. The constructed maps depicted the high-level structure of the road network. In this context, centerline representations primarily focus on the general road topology, rather than lane-level structures. Consequently, these early maps lacked critical precision and contextual information required for high-accuracy autonomous navigation.

Trail and crowdsourced offline mapping: Trail-based offline mapping approaches utilize recordings of vehicle trajectories for map creation. In general, most early approaches aggregated GPS trails of multiple recording vehicles and extracted the road network structure based on rule-based methods like trace-merging, k-means clustering, or kernel density estimation [2].

Building on these foundational techniques, more advanced trail-based methods have been developed for lane mapping and road geometry extraction. One approach involved generating BEV orthographic images by fusing GPS, INS, and visual odometry data, then deriving lane graphs using a rule-based method incorporating the Random Sample Consensus algorithm [12]. Another method inferred lane geometries from Radar and camera probe data by measuring distances to stationary objects and applying k-means clustering to identify lane markings and widths [27].

A specialized highway mapping approach used an iterative Graph SLAM method, aligning data from odometry, GNSS, and traffic sign measurements, followed by road boundary measurements for refinement [11]. A key challenge in offline mapping is maintaining up-to-date maps, which Kim et al. addressed by incorporating uncertainty estimation to identify reliable landmark observations [15]. Another method introduced a three-stage pipeline for detecting, updating, and integrating map changes [31].

Integrating 3D object tracking modules allows for the inclusion of additional trails from observed traffic participants, expanding the available trail data. One mapping approach involved dividing aggregated trails into tiles and constructing a density map to visualize detection frequencies per pixel [8]. The method further separated density maps by driving direction and used a rule-based algorithm to extract lane-level centerlines.

A specialized approach for intersections estimated lane geometries using detected road user trajectories. It employed a two-stage Markov chain Monte Carlo sampling method [28], later refined with a non-linear least squares formulation for greater accuracy [29]. The system could process data from cameras, Radar, or Lidar but was primarily evaluated using simulated vehicle data. Similarly, FlowMap [10] introduced a framework for deriving human-like driving paths in unmarked road environments by clustering entry and exit points of recorded trajectories. An extended version incorporated traffic flow data to generate HD map patches for intersections [33].

A more recent approach automated lane model generation by leveraging sparse vehicle observations and a deep learning-based architecture to predict lane pairs and their connectivity [30]. While the model demonstrated promising results, its evaluation was limited to internal datasets, restricting generalizability.

Among the aforementioned methods, only one specifically addresses centerline extraction from trail data using state-of-the-art deep learning models. In our approach, we integrate recent advancements from online mapping into an offline mapping pipeline. Therefore, we adopt a BEV tile-based structure, capturing information from aggregated pre-recorded data, and modify the model architecture accordingly. Once trained, our deep learning-based model enables tile-wise map updates for recently collected data, increasing flexibility and scalability compared to existing methods.

Online HD map learning: Online HD mapping approaches construct HD maps in real-time by processing onboard sensor data. Modern methods utilize a combination of sensor modalities, typically including single front-view camera images, camera surround views consisting of multiple images and a sensor point cloud from Radar or Lidar sensors. Due to the complexity of these multimodal inputs, recent research predominantly relies on machine learning-based techniques, as rule-based algorithms are insufficient for capturing the intricate spatial and semantic relationships required for HD map construction [17, 20, 21, 25].

The first online mapping approach capable of generating vectorized map elements directly from sensor data is HDMapNet [17]. Its architecture employs a combination of encoder-decoder networks to predict segmentation maps, instance embeddings, and directional cues, which are subsequently fused to construct a vectorized HD map.

More recent architectures in online mapping follow a similar structure, largely inspired by the Detection Transformer (DETR) framework [5]. In these models, a feature extraction backbone processes the input and projects it into the bird's-eye view (BEV) space. These features are then attended to by learnable object queries within a transformer decoder. The output of the decoder is then passed to a prediction head consisting of a classification and regression head. The former predicts the class of the map object, while the latter predicts its shape represented by discrete points on the map. Examples of this type of architecture can be seen in MapTR [20], VectorMapNet [25] and MapTRv2 [21], with the last two being able to also predict lane centerlines. Further approaches since then have optimized the architecture, e.g. through mask guiding the prediction of map objects [24], improving the query design of the transformer decoder [26], integrating prior and additional information [13, 16, 18, 34, 35, 37, 39] or performing temporal fusion of the frames to overcome inconsistency limitations due to occlusion and complexity [6, 19, 32, 40].

In our approach, we leverage a similar DETR-based architecture, but we modify the input features to work with collected and processed detection data. This mitigates problems regarding occlusion and temporal inconsistency, while unifying the input representation and thereby enabling great generalization properties.

3. Methodology

In the following section, the overall method of our trail-based mapping approach is outlined. Firstly, the input and ground truth representation are described, followed by the data augmentation for training, our model architecture and the output representation.

3.1. Input and ground truth representation

The first step of our proposed method is trail aggregation (cf. Fig. 2a). Vehicles equipped with self-localization and 3D perception sensors record driving data, which is subsequently processed to integrate localization information with an object detection and tracking algorithm. This results in two distinct types of trails: Ego trails representing the trajectory of the ego vehicle during the recording period and object trails depicting the perceived motion paths of other vehicles in traffic. A trail t is defined by a series of cuboid center points c_i, $i \in [1, p]$, forming a linearized path $t = \{c_1, c_2, ..., c_p\}$. The trail width w_t corresponds to the width of the respective vehicle w_{veh}.

The trails are then mapped onto a global coordinate system. To serve as input for a neural network, the trails are converted into a grid-based map representation. This process involves dividing the covered area into discrete tiles (cf. Fig. 2b). For a single tile, the real world area

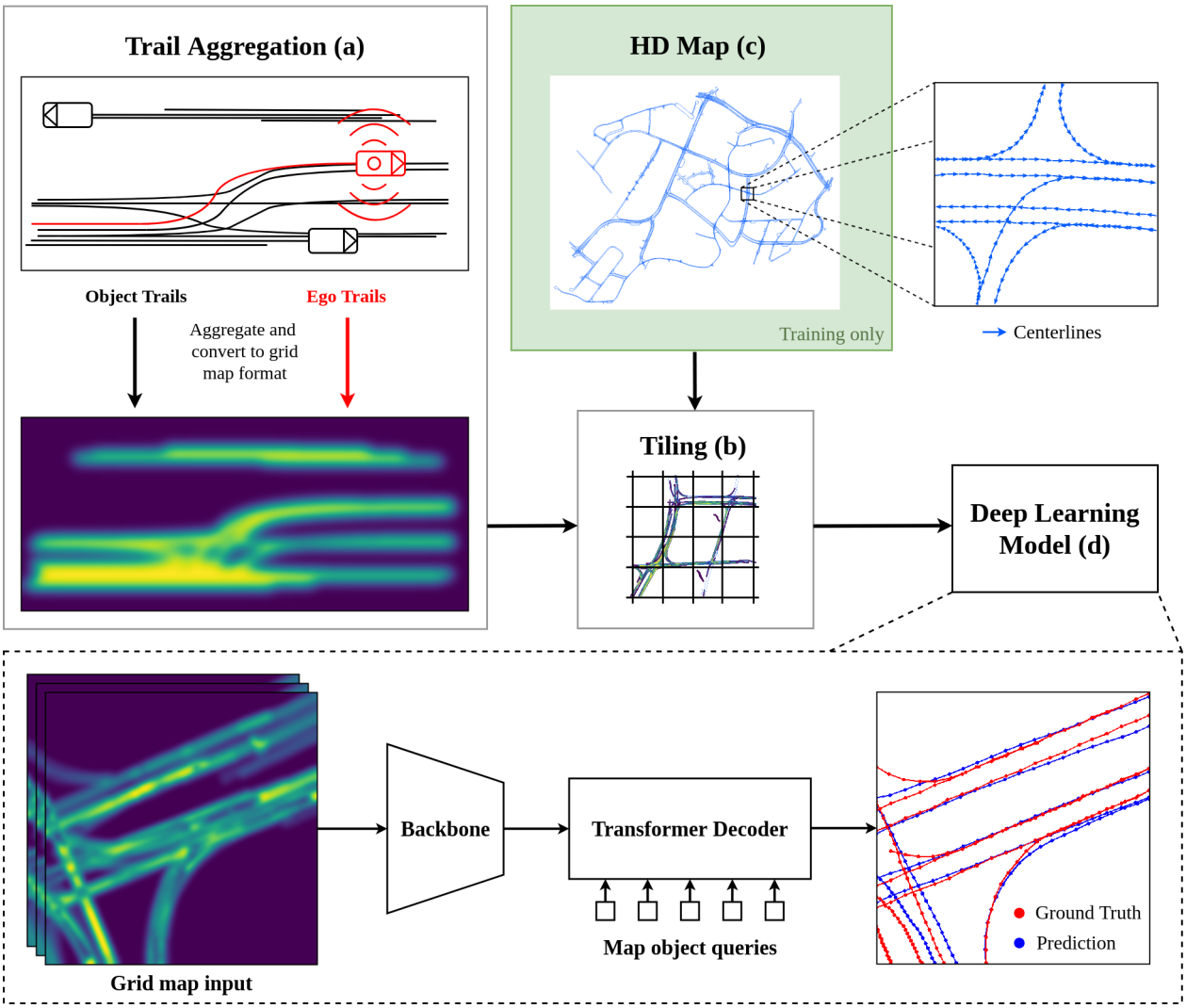


Figure 2. The overall architecture of the trail-based mapping approach. Starting with the trail aggregation (a), object and ego trails are converted into a grid map format. The covered area is divided into tiles (b) and for training the ground truth is obtained from a centerline HD map (c). The resulting samples are then fed into a deep learning model (d) that, once trained, infers tile-wise centerline predictions for the input trail grid maps.

$w_{glob} \times h_{glob}$ with a resolution of r results in a representative grid map G of size $w_{grid} \times h_{grid} := \frac{w_{glob}}{r} \times \frac{h_{glob}}{r}$.

G is separated into n different bins based on the current yaw angle φ of each trail segment, which enables clear separation of directions at intersections and distinction of opposing lanes (cf. Fig. 3b). The yaw angle of a trail segment is defined as $\varphi \in (-\pi, \pi]$ in the global coordinate system, resulting in an interval of I_j per bin:

$$I_j = \left(-\pi + (j-1)\frac{2\pi}{n}, -\pi + j\frac{2\pi}{n}\right], \quad j \in [1, n]. \quad (1)$$

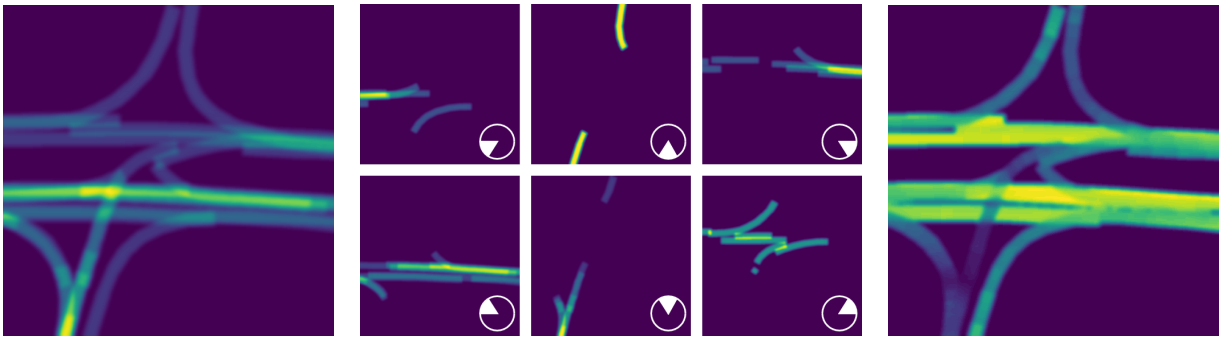
Each bin corresponds to an individual channel $C_{dir,j} \in \mathbb{R}^{w \times h}$ of G. For each trail segment, the values of the individual direction channel $C_{dir,j}$ are incremented by 1 for each containing grid cell.

Additionally, the average speed for each grid cell is computed and incorporated as an additional input channel $C_{speed} \in \mathbb{R}^{w \times h}$ (cf. Fig. 3c).

The resulting definition of G is given by:

$$G \in \mathbb{R}^{w \times h \times (n+1)}. \quad (2)$$

To ease the detection of sensible features by the neural network, Gaussian smoothing is applied to each input matrix channel. The kernel for the smoothing operation is characterized by the standard deviation σ of the normal distribution and determines the degree of smoothing applied to



(a) Visualization of an example trail grid map sample without yaw angle separation.

(b) Visualization the trail grid map sample from Fig. 3a separated into $n = 6$ different bins by yaw angle.

(c) Visualization of the average speed profile for the example sample from Fig. 3a.

Figure 3. Visualization of an example sample from the nuScenes dataset.

the input data.

To train our model, ground truth information is required in addition to the trail-based input. In our approach, we derive the ground truth from an HD centerline map, ensuring alignment with the areas covered by the recorded trails (cf. Fig. 2c). Since the neural network is expected to predict centerline directions, the HD map must provide directional information for each centerline. To generate the training data, we extract centerlines from the HD map that correspond to regions with nonzero trail density while explicitly filtering out centerlines that lack associated trails. This filtering process is performed by matching the yaw angles of the centerlines and trails to ensure consistency. From the remaining centerline graph, we derive all drivable paths following the approach of [22]. We represent each path by a fixed number of ordered points m.

3.2. Data augmentation

In addition to the grid-based tiling shown in Fig. 2b, we apply a sample augmentation method to enhance training diversity. Instead of partitioning the regions containing trails and ground truth data into fixed grid tiles, we randomly select center points within these areas to define augmented tiles. To introduce further variation, we assign the rotation to each tile randomly and we only select a subset of trails within the tile area, ensuring exposure to different trail distributions during training.

3.3. Model architecture and output

The model architecture builds upon MapTRv2 [21], utilizing the same DETR-inspired query-based transformer decoder. However, we introduce modifications to the backbone to better accommodate our input representation. We replace the original multi-view camera and Lidar processing pipeline with a single ResNet50 backbone for feature extraction. This change is motivated by the fact that our input consists of a multi-channel matrix. Several auxiliary task heads, including depth prediction, BEV segmentation, and perspective-view segmentation, are removed, as they are not applicable to our setting. The classification head is also omitted, as our model focuses exclusively on centerline prediction.

The maximum number of predicted centerlines in the output of the model is a tunable hyperparameter that correlates with the amount of queries in the transformer decoder. Each prediction consists of the same fixed number of points m compared to the ground truth centerlines. An example of a prediction output and corresponding ground truth is shown in the bottom right of Fig. 2. The order of the coordinates in the prediction indicates the direction of the centerline.

The loss of the model is based on instance and point-level matching as described in [21], with only a single possible permutation per centerline since its direction is fixed. We employ the same one-to-one and one-to-many set prediction loss, but omit the focal loss since there is no classification head.

4. Experiments

In the following section, we present experiments of our trail-based mapping approach. We first describe the datasets. Next, we compare our method's performance against state-of-the-art online mapping approaches. Finally, we conduct ablation studies to assess the impact of input variations on performance.

4.1. Datasets

To ensure reproducibility and comparability with existing approaches, we primarily rely on public datasets for training and evaluation. These datasets must contain both trail information and ground truth centerlines to be suitable for our approach. Trail information can be given by recorded GPS poses of the ego vehicle, combined with dynamic ob-

Table 1. Validation split performance of MapTRv2 in comparison to our approach (TrailTR) on the nuScenes original (orig.) and near extrapolation geographical (geo.) splits [23] as well as the original nuPlan split.

Model	Training Data	Validation Data	Validation Split AP
MapTRv2	nuScenes (orig.)	nuScenes (orig.)	53.3
TrailTR	nuScenes (orig.)	nuScenes (orig.)	**56.4**
MapTRv2	nuScenes (geo.)	nuScenes (geo.)	30.5
		+ trail area masking	27.2
TrailTR	nuScenes (geo.)	nuScenes (geo.)	**44.6**
TrailTR	nuPlan (orig.)	nuPlan (orig.)	**70.3**
TrailTR	nuScenes (geo.)	nuPlan (orig.)	34.0
TrailTR	nuPlan (orig.)	nuScenes (geo.)	35.0
TrailTR	nuScenes (geo.) + nuPlan (orig.)	nuPlan (orig.)	69.7
TrailTR	nuScenes (geo.) + nuPlan (orig.)	nuScenes (geo.)	44.3

Table 2. Validation split performance of MapTRv2 in comparison to our approach (TrailTR) on the nuScenes original (orig.) and near extrapolation geographical (geo.) splits [23] with different input data foundation modalities (*: object tracking data). The performance is compared between trails extracted from ground truth object tracking and Camera + Lidar-based object tracking through FocalFormer3D [7] (72.4 mAP, 74.5 NDS, 73.9 AMOTA). Further evaluation is performed on a closed Las Vegas dataset based on Radar object tracking.

Model	Input Data Foundation	Training Data	Validation Data	Validation Split AP
MapTRv2	Camera	nuScenes (geo.)	nuScenes (geo.)	30.5
TrailTR	Ground truth*	nuScenes (geo.)	nuScenes (geo.)	44.6
TrailTR	Camera + Lidar*	nuScenes (geo.)	nuScenes (geo.)	43.7
TrailTR	Ground truth* (training), Radar* (validation)	nuScenes (geo.) + nuPlan (orig.)	closed Las Vegas dataset	30.0

ject data that includes tracking over time. Ground truth centerlines must be embedded within a global HD map, covering the entire trail region. Before a dataset can be used for training, the sample generation process (cf. Fig. 2a-c) must be applied to extract the necessary input representations.

In online mapping research, the most widely used datasets are nuScenes [3] and Argoverse2 [38]. Despite containing only 15 hours of driving data, nuScenes provides a detailed HD map for each included city [3]. In Argoverse2, the HD map is divided into local, scenario-specific tiles, which renders it incompatible with our proposed tiling and augmentation pipeline. In contrast, the nuPlan dataset [4] provides city-wide HD maps and significantly larger amounts of trail data, with 1,500 hours of recorded driving data.

For comparability with online mapping approaches, we train and evaluate our method on the nuScenes dataset. To assess transferability and generalization, we further validate our approach on the nuPlan dataset as well as additional closed data sources. Since the selected datasets do not provide trail data for every centerline element in the HD map, each generated tile is refined according to the procedure described in Sec. 3.1.

4.2. Comparison to online HD mapping approaches

Metric: For evaluation, we employ the same metric as in most online mapping approaches. We calculate the average precision (AP) for all samples by utilizing the Chamfer distance $D_{Chamfer}$ as a threshold to determine the positive matches between prediction and ground truth. The AP is calculated for each of the three thresholds $\tau \in T, T = \{0.5, 1.0, 1.5\}$ individually and then averaged across all to determine the final AP metric used for comparison:

$$AP = \frac{1}{|T|} \sum_{\tau \in T} AP_{\tau}. \tag{3}$$

Generalization to new areas: A recent study highlighted limitations in the original training, validation, and testing splits of the nuScenes dataset, particularly regarding geographical overlaps between these splits [23]. The authors show, that most state-of-the-art online mapping mod-



Figure 4. Visualization of ground truth (red), model predictions (blue) and its input grid map on an aerial image. The samples are from the nuScenes geograhical validation split (a-b) and nuPlan validation split (c-f). The samples from the nuScenes dataset lack sufficient amounts of trails to depict the entirety of lanes in the map tile, so the ground truth was refined as described in Sec. 3.1.

els have been trained and evaluated on the original splits and show limited generalization when applied to geographically disjoint splits. To assess whether this trend also affects centerline detection in online mapping, we trained MapTRv2 [21] using both the original nuScenes split and geographically disjoint near extrapolation nuScenes split [23]. The results of this evaluation are presented in Tab. 1.

For training and evaluation of our trail-based approach, we delete ground truth data for areas with no trails. In order to compare the results for MapTRv2 with our approach, we employ masking for the respective validation splits. Thereby, we omit the Chamfer distances $D_{j,Chamfer}$ for areas for which no trails exist during the calculation of the AP. This way, it is ensured that the areas without trails do not decrease the performance of the MapTRv2 results (e.g. because of increased complexity inside the masked area) in comparison to the trail-based approach. The results for the centerline AP with masking are shown in Tab. 1.

Our trail-based approach outperforms MapTRv2 on both the original (56.4 AP, +5.8 %) and geographically disjoint near extrapolation (44.6 AP, +46.2 %) nuScenes splits, demonstrating superior generalization across different data distributions. Additionally, our approach benefits from larger and more diverse training datasets. This is evident from the performance improvement observed when training and evaluating on the original nuPlan training and validation splits, which contain a significantly higher number of trails compared to nuScenes and lead to a performance of 70.3 AP on the validation split. The predictions for exemplary samples from the nuScenes geographical validation split and the nuPlan validation split are visualized in Fig. 4. When cross-evaluating the two datasets, a performance decrease can be observed, which may be caused by the difference in trail amounts per tile between nuScenes and nuPlan. By training on both datasets, this effect can be mitigated and the validation split performance on both datasets reaches similar performance compared to trainings on the individual datasets.

Generalization to different sensor setups: To avoid introducing additional noise during training due to sensor imperfections, we utilized the ground truth object tracking annotations provided by the nuScenes and nuPlan datasets

Table 3. Different smoothing settings and the corresponding performance on the geographical nuScenes validation split.

Smoothing Settings	Validation Split AP
No smoothing	44.1
Gaussian smoothing, σ = 1	44.1
Gaussian smoothing, σ = 2	**44.6**
Gaussian smoothing, σ = 4	41.5
Gaussian smoothing, σ = 8	38.9

Table 4. Different channel combinations and binning strategies with the corresponding performance on the geographical nuScenes validation split.

Channel Settings	Validation Split AP
Single trail grid channel	9.4
+ speed profile	36.9
n = 4 channel bins + speed profile	44.1
n = 6 channel bins + speed profile	**44.6**
n = 12 channel bins + speed profile	43.4
n = 36 channel bins + speed profile	43.2

to extract trail data for our previously listed experiments. To verify the sensor-agnostic properties of our approach, we additionally extracted sensor-based trail data using the state-of-the-art object detection and tracking algorithm FocalFormer3D [7]. The results, presented in Tab. 2, indicate a minimal performance decrease, as the general shape of the grid tiles is preserved through sufficiently accurate object tracking. Furthermore, we applied our approach to a closed Las Vegas dataset, using Radar-based detections as the foundation for trail data. The corresponding results are also listed in Tab. 2.

4.3. Ablation Studies

To further optimize the centerline extraction, we provide ablation studies regarding modifications to the model input.

Gaussian smoothing: To ease the detection of sensible features in the input, we employ Gaussian smoothing on each individual input channel. Table 3 presents the results for various standard deviations σ of the Gaussian smoothing kernel. It appears that the optimal value is $\sigma = 2$, as it effectively unifies visual features across different input samples while preserving the overall shape of trail densities.

Channel variations and speed profile: The network is designed to predict directed centerlines. Introducing directional bins helps distinguish individual directions, providing additional information to the model. Furthermore, incorporating the average speed profile offers insights into lane structure, as speed variations often correlate with specific road layouts. In Tab. 4, we present the performance differences resulting from the inclusion of the speed profile and the use of different binning strategies. The speed profile and direction bin separation increase the input complexity while providing additional information, therefore we expect that an optimal balance exists. We experimentally verified our assumptions and found an optimum at $n = 6$.

Table 5. Training with and without augmented samples from the nuScenes geographical training split and the corresponding performance on the geographical nuScenes validation split.

Training Samples	Validation Split AP
Training without augmented samples	25.2
Training with augmented samples	**44.6**

Augmentation: As described in Sec. 3.2, we applied an augmentation technique to enhance training diversity. To assess its impact on performance, we additionally trained the model without augmented tiles and report the results in Tab. 5. As expected, the additional training samples increased the performance on the validation split (+77.0 %).

5. Conclusion

In this work, we introduced a novel trail-based mapping approach that extracts high-definition driving maps by aggregating trajectories from both ego vehicles and other traffic participants. Our method leverages deep learning to convert trail data into a map representation, enabling tile-wise global map construction. Unlike traditional offline HD maps, our approach is sensor-agnostic and scalable, allowing for continuous map updates without dependence on predefined static datasets.

Through extensive experiments, we demonstrated that our method outperforms state-of-the-art online mapping models, particularly in terms of generalization to new areas. Our approach remains robust across various sensor setups, with minimal performance degradation when using sensor-based trail extraction. Additionally, we showed that augmenting training data with diverse trail distributions further enhances model performance.

While this work focuses on centerline extraction, the methodology can be extended to extract additional road elements, such as lane boundaries, pedestrian crossings, and traffic signs, by incorporating corresponding sensor data. Furthermore, integrating our method into a comprehensive map generation framework could enable the construction of fully connected lane graphs, further bridging the gap between online and offline mapping approaches.

References

[1] Zhibin Bao, Sabir Hossain, Haoxiang Lang, and Xianke Lin. High-Definition Map Generation Technologies for Autonomous Driving, 2022. arXiv:2206.05400 [cs].

[2] James Biagioni and Jakob Eriksson. Inferring Road Maps from Global Positioning System Traces: Survey and Comparative Evaluation. *Transportation Research Record: Journal of the Transportation Research Board*, 2291(1):61–71, 2012.

[3] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving, 2020. arXiv:1903.11027 [cs].

[4] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. NuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles, 2022. arXiv:2106.11810 [cs].

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers, 2020. arXiv:2005.12872 [cs].

[6] Jiacheng Chen, Yuefan Wu, Jiaqi Tan, Hang Ma, and Yasutaka Furukawa. MapTracker: Tracking with Strided Memory Fusion for Consistent Vector HD Mapping, 2024. arXiv:2403.15951 [cs].

[7] Yilun Chen, Zhiding Yu, Yukang Chen, Shiyi Lan, Animashree Anandkumar, Jiaya Jia, and Jose Alvarez. FocalFormer3D : Focusing on Hard Instance for 3D Object Detection, 2023. arXiv:2308.04556 [cs].

[8] Pascal Colling, Dennis Muller, and Matthias Rottmann. HD Lane Map Generation Based on Trail Map Aggregation. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 600–606, Aachen, Germany, 2022. IEEE.

[9] Daniel Dauner, Marcel Hallgarten, Andreas Geiger, and Kashyap Chitta. Parting with Misconceptions about Learning-based Vehicle Motion Planning, 2023. arXiv:2306.07962 [cs].

[10] Wenchao Ding, Jieru Zhao, Yubin Chu, Haihui Huang, Tong Qin, Chunjing Xu, Yuxiang Guan, and Zhongxue Gan. FlowMap: Path Generation for Automated Vehicles in Open Space Using Traffic Flow. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1616–1622, London, United Kingdom, 2023. IEEE.

[11] Christopher Doer, Michael Henzler, Heiner Messner, and Gert F. Trommer. HD Map Generation from Vehicle Fleet Data for Highly Automated Driving on Highways. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 2014–2020, Las Vegas, NV, USA, 2020. IEEE.

[12] Chunzhao Guo, Kiyosumi Kidono, Junichi Meguro, Yoshiko Kojima, Masaru Ogawa, and Takashi Naito. A Low-Cost Solution for Automatic Lane-Level Map Generation Using Conventional In-Car Sensors. *IEEE Transactions on Intelligent Transportation Systems*, 17(8):2355–2366, 2016.

[13] Zhou Jiang, Zhenxin Zhu, Pengfei Li, Huan-ang Gao, Tianyuan Yuan, Yongliang Shi, Hang Zhao, and Hao Zhao. P-MapNet: Far-seeing Map Generator Enhanced by both SDMap and HDMap Priors, 2024. arXiv:2403.10521 [cs].

[14] Jialin Jiao. Machine Learning Assisted High-Definition Map Creation. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, pages 367–373, Tokyo, Japan, 2018. IEEE.

[15] Kitae Kim, Soohyun Cho, and Woojin Chung. HD Map Update for Autonomous Driving With Crowdsourced Data. *IEEE Robotics and Automation Letters*, 6(2):1895–1901, 2021.

[16] Han Li, Zehao Huang, Zitian Wang, Wenge Rong, Naiyan Wang, and Si Liu. Enhancing 3D Lane Detection and Topology Reasoning with 2D Lane Priors, 2024. arXiv:2406.03105 [cs].

[17] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. HDMapNet: An Online HD Map Construction and Evaluation Framework, 2022. arXiv:2107.06307 [cs].

[18] Siyu Li, Kailun Yang, Hao Shi, Jiaming Zhang, Jiacheng Lin, Zhifeng Teng, and Zhiyong Li. Bi-Mapper: Holistic BEV Semantic Mapping for Autonomous Driving, 2023. arXiv:2305.04205 [cs].

[19] Siyu Li, Jiacheng Lin, Hao Shi, Jiaming Zhang, Song Wang, You Yao, Zhiyong Li, and Kailun Yang. DTCLMapper: Dual Temporal Consistent Learning for Vectorized HD Map Construction, 2024. arXiv:2405.05518 [cs, eess].

[20] Bencheng Liao, Shaoyu Chen, Xinggang Wang, Tianheng Cheng, Qian Zhang, Wenyu Liu, and Chang Huang. MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction, 2023. arXiv:2208.14437 [cs].

[21] Bencheng Liao, Shaoyu Chen, Yunchi Zhang, Bo Jiang, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. MapTRv2: An End-to-End Framework for Online Vectorized HD Map Construction, 2023. arXiv:2308.05736 [cs].

[22] Bencheng Liao, Shaoyu Chen, Bo Jiang, Tianheng Cheng, Qian Zhang, Wenyu Liu, Chang Huang, and Xinggang Wang. Lane Graph as Path: Continuity-preserving Path-wise Modeling for Online Lane Graph Construction, 2024. arXiv:2303.08815 [cs].

[23] Adam Lilja, Junsheng Fu, Erik Stenborg, and Lars Hammarstrand. Localization is All You Evaluate: Data Leakage in Online Mapping Datasets and How to Fix it. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22150–22159, Seattle, WA, USA, 2024. IEEE.

[24] Xiaolu Liu, Song Wang, Wentong Li, Ruizi Yang, Junbo Chen, and Jianke Zhu. MGMap: Mask-Guided Learning for Online Vectorized HD Map Construction. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14812–14821, Seattle, WA, USA, 2024. IEEE.

[25] Yicheng Liu, Tianyuan Yuan, Yue Wang, Yilun Wang, and Hang Zhao. VectorMapNet: End-to-end Vectorized HD Map Learning, 2023. arXiv:2206.08920 [cs].

[26] Zihao Liu, Xiaoyu Zhang, Guangwei Liu, Ji Zhao, and Ningyi Xu. Leveraging Enhanced Queries of Point Sets for Vectorized Map Construction, 2024. arXiv:2402.17430 [cs].

[27] K. Massow, B. Kwella, N. Pfeifer, F. Häusler, J. Pontow, I. Radusch, J. Hipp, F. Dölitzscher, and M. Haueis. Deriving

HD maps for highly automated driving from vehicular probe data. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1745–1752, Rio de Janeiro, Brazil, 2016. IEEE.

[28] Annika Meyer, Jonas Walter, Martin Lauer, and Christoph Stiller. Anytime Lane-Level Intersection Estimation Based on Trajectories of Other Traffic Participants, 2019. arXiv:1906.02495 [cs].

[29] Annika Meyer, Jonas Walter, and Martin Lauer. Fast Lane-Level Intersection Estimation using Markov Chain Monte Carlo Sampling and B-Spline Refinement, 2020. arXiv:2007.06904 [cs].

[30] Michael Mink, Thomas Monninger, and Steffen Staab. LMT-Net: Lane Model Transformer Network for Automated HD Mapping from Sparse Vehicle Observations, 2024. arXiv:2409.12409 [cs].

[31] David Pannen, Martin Liebner, Wolfgang Hempel, and Wolfram Burgard. How to Keep HD Maps for Automated Driving Up To Date. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2288–2294, Paris, France, 2020. IEEE.

[32] Nan Peng, Xun Zhou, Mingming Wang, Xiaojun Yang, Songming Chen, and Guisong Chen. PrevPredMap: Exploring Temporal Modeling with Previous Predictions for Online Vectorized HD Map Construction, 2024. arXiv:2407.17378 [cs].

[33] Tong Qin, Haihui Huang, Ziqiang Wang, Tongqing Chen, and Wenchao Ding. Traffic Flow-Based Crowdsourced Mapping in Complex Urban Scenario. *IEEE Robotics and Automation Letters*, 8(8):5077–5083, 2023.

[34] Jingyu Song, Xudong Chen, Liupei Lu, Jie Li, and Katherine A. Skinner. MemFusionMap: Working Memory Fusion for Online Vectorized HD Map Construction, 2024. arXiv:2409.18737 [cs].

[35] Rémy Sun, Li Yang, Diane Lingrand, and Frédéric Precioso. Mind the map! Accounting for existing map information when estimating online HDMaps from sensor, 2024. arXiv:2311.10517 [cs].

[36] Simon Ulbrich, Andreas Reschka, Jens Rieken, Susanne Ernst, Gerrit Bagschik, Frank Dierkes, Marcus Nolte, and Markus Maurer. Towards a Functional System Architecture for Automated Vehicles, 2017. arXiv:1703.08557 [cs].

[37] Rongxuan Wang, Xin Lu, Xiaoyang Liu, Xiaoyi Zou, Tongyi Cao, and Ying Li. PriorMapNet: Enhancing Online Vectorized HD Map Construction with Priors, 2024. arXiv:2408.08802 [cs].

[38] Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting, 2023. arXiv:2301.00493 [cs].

[39] Xuan Xiong, Yicheng Liu, Tianyuan Yuan, Yue Wang, Yilun Wang, and Hang Zhao. Neural Map Prior for Autonomous Driving. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17535–17544, Vancouver, BC, Canada, 2023. IEEE.

[40] Tianyuan Yuan, Yicheng Liu, Yue Wang, Yilun Wang, and Hang Zhao. StreamMapNet: Streaming Mapping Network for Vectorized Online HD Map Construction. In *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 7341–7350, Waikoloa, HI, USA, 2024. IEEE.

[41] Juan Zhong, Zheng Liu, and Xi Chen. Transformer-based models and hardware acceleration analysis in autonomous driving: A survey, 2023. arXiv:2304.10891 [cs].