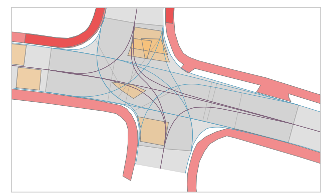
Lanelet2 for nuScenes: Enabling Spatial Semantic Relationships and Diverse Map-based Anchor Paths

Alexander Naumann*[1], Felix Hertlein*[1], Daniel Grimm[1], Maximilian Zipfl[1], Steffen Thoma[1], Achim Rettinger[2], Lavdim Halilaj[3], Juergen Luettin[3], Stefan Schmid[3] and Holger Caesar[4]

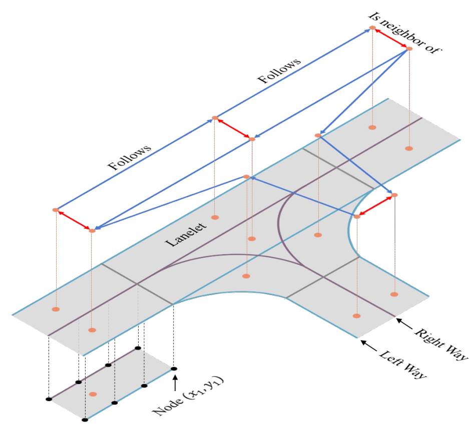
Abstract

Motion prediction and planning are key components to enable autonomous driving. Although high definition (HD) maps provide important contextual information that constrains the action space of traffic participants, most approaches are not able to fully exploit this heterogeneous information. In this work, we enrich the existing road geometry of the popular nuScenes dataset and convert it into the open-source map framework Lanelet2. This allows easy access to the road topology and thus, enables the usage of (1) spatial semantic information, such as agents driving on intersecting roads and (2) map-generated anchor paths for target vehicles that can help to improve trajectory prediction performance. Further, we present DMAP*, a simple, yet effective approach for **d**iverse **m**ap-based **a**nchor **p**ath generation and filtering. We show that combining DMAP with ground truth velocity profile information yields high-quality motion prediction results on nuScenes ($MinADE_5$=1.09, $MissRate_{5,2}$=0.18, Offroad rate=0.00). While it is obviously unfair to compare us against the state-of-the-art, it shows that our HD map accurately depicts the road geometry and topology. Future approaches can leverage this by focusing on data-driven sampling of map-based anchor paths and estimating velocity profiles. Moreover, our HD map can be used for map construction tasks and supplement perception. Code and data are made publicly available at https://felixhertlein.github.io/lanelet4nuscenes.*

(a) Original nuScenes Map



(b) Our Lanelet2 Map



(c) Hierarchical Road Topology

Figure 1. We convert the nuScenes map 1(a) into the open-source map format Lanelet2 1(b) to enable easy access to the full road geometry and topology in a hierarchical structure 1(c).

1. Introduction

Enabling fully autonomous driving has been a long-standing goal in research [1]. The main drivers fuelling this strong interest are the expected benefits: autonomous vehicles could at the same time increase efficiency, safety, and convenience. In order to attain this objective, a number of challenging tasks need to be solved reliably, such as perception, prediction, planning, and control. High definition (HD) maps are highly beneficial for tackling numerous subtasks, as they contain detailed information about the surroundings of a vehicle. This does not only include information on the road geometry and the road topology but also on traffic rules and adjacent areas, such as walkways and bicycle lanes.

While the performance of *perception* modules has increased rapidly in recent years [2], [3], inaccurate sensor measurements, occlusion and limited sensor range are still common challenges. Accurate and reliable HD maps can

[1]FZI Research Center for Information Technology, Karlsruhe, Germany {anaumann, hertlein}@fzi.de
[2]Trier University, Trier, Germany
[3]Bosch Corporate Research, Renningen, Germany
[4]Delft University of Technology, Delft, Netherlands
*Authors contributed equally.

help to correctly interpret observed senor data and serve as a backup by providing an additional layer of redundancy. Thus, they can improve the robustness of perception modules, which is beneficial for subsequent downstream tasks. In addition to that, the easily accessible road topology of HD maps provides important information for *prediction* and *planning* tasks. By mapping traffic participants onto lanes within an HD map, it is possible to infer semantic relationships between traffic participants. Such semantic relationships include *driving on neighboring lanes* or *intersecting roads*, which are highly relevant for prediction and planning [4]. Moreover, the road topology enables the convenient computation of all valid and drivable paths for a given target vehicle. We refer to this set of possible paths within the map as *anchor paths*. Anchor paths can help to diversify prediction hypotheses [5] and to avoid *mode collapse*, where predictions only follow a sub-set of the possible drivable paths. Furthermore, anchor paths contain information on the full road geometry and hence, can be used to constrain predictions to lie on the road by clipping them onto the road boundaries. Finally, existing HD maps are also crucial to enable research for HD map construction.

In this work, we present an approach to enhance the HD road maps of the popular nuScenes dataset [6]. While the dataset already provides detailed information on the geometry of the roads and their surroundings, it is not straightforward to infer the full road topology, since intersections are not modelled explicitly. We bridge this gap by providing an approach to automatically convert the nuScenes map to the popular automated driving format Lanelet2 [7], which enables convenient access to the full road topology. Lanelet2 is an open-source map format and extends the original Lanelet format [8], which is more lightweight compared to other formats such as OpenDRIVE, yet powerful enough for all major needs in autonomous driving [9]. Unlike other existing open-source map formats, Lanelet2 comes with a C++ toolkit that features numerous auxiliary functionalities. Thus, Lanelet2 has recently been adopted as the default map format for several new datasets [10]–[12] and is supported by many tools [13]–[17].

Our map conversion process includes several steps to ensure a highly accurate road geometry and road topology. Minimal manual effort is required to handle rare edge cases and the maps can easily be extended, e.g. by adding regulatory elements, such as speed limits. Since we preserve correspondence to the nuScenes IDs, additional information from the original dataset can easily be queried and integrated. Finally, by exploiting the information of the implicitly defined lane connectivity graph of the Lanelet2 map, we present a simple, yet effective algorithm called DMAP to generate a set of diverse drivable anchor paths for a target vehicle, which can be readily used for trajectory prediction and planning tasks. We show, that when combined with the ground truth velocity profile, our map-based anchor path generation approach yields high-quality results for motion prediction on nuScenes. This suggests that our map accurately depicts the road geometry and topology and further, that future approaches can leverage our work to focus on data-driven sampling of map-based anchor paths and estimating velocity profiles.

The main contributions of our work can be summarised as follows:

- We enrich the existing road geometry information of the nuScenes dataset to enable easy access to the full road geometry and topology,
- We present an approach for the generation and the diversity-based filtering of map-based anchor paths called DMAP, and
- We evaluate DMAP by combining it with the ground truth velocity profiles to validate the quality of our map and anchor paths.

This paper is structured as follows. We present related work in Sec. 2. Subsequently, we introduce the map conversion approach in Sec. 3 and the anchor path generation in Sec. 4. The quantitative and qualitative evaluation is presented in Sec. 5 and the paper concludes with Sec. 6.

2. Related Work

We first review the literature on existing datasets. Afterwards, we present related work on map formats, relevant applications and semantic information in the context of autonomous driving.

Datasets There are many datasets focusing on perception for autonomous driving [2], such as KITTI [18], [19], ApolloScape [20], and Waymo Open [21]. These datasets seek to advance research in areas such as 2D and 3D object detection. Other datasets such as INTERACTION [10], Lyft prediction dataset [22], inD [11] and SIND [12] are top-view datasets, which focus on highly interactive traffic scenarios and do not contain raw sensor data. They provide detailed information on the trajectories of traffic participants and seek to advance research for trajectory prediction and planning. Notably, all but the Lyft prediction dataset use the Lanelet2 [7] map format. The nuScenes [6] and the Argoverse [23], [24] datasets provide a multiplicity of annotations and enable research in both fields, perception and motion planning. For further details on existing datasets and their applications, we refer to Chen *et al.* [25].

HD Map Formats Currently, there is no unique standard format for HD road maps. While there are proprietary solutions available, we focus on three popular open formats for HD maps [26]: OpenDRIVE[1], Apollo Maps and Lanelet2 [7]. OpenDRIVE was developed by the Association for Standardization of Automation and Measuring Systems (ASAM). The format was originally designed for the usage in driving simulators. Roads are represented by reference paths, which can be specified by five different types of functions. Lanes within the road are defined by a lateral offset from the reference path. Some concepts, such as pedestrian islands, are cumbersome to model in OpenDRIVE [9]. Furthermore, it pursues a top-down approach and thus, as requirements increase, the representation of the map information becomes increasingly complex.[2] Apollo Maps builds on top of OpenDRIVE and adjusts it for the usage with Apollo[3]. Main differences compared to OpenDRIVE include the description of geometries, which are represented as a set of points in the Apollo format. Other differences include the definition of junctions and the number of supported object types. We refer to Gran [27] for a detailed analysis of the differences. Finally, Lanelet2 [7] extends the earlier Lanelet [8] map format. Limitations of the first Lanelet version, such as only allowing lane changes at fixed points, have been resolved. It is a lightweight format that covers all major needs for autonomous driving and is easier to use than OpenDRIVE [9]. Moreover, it is tailored to map the real world and it is the only map format that provides a full framework which is available as open-source. The framework is implemented in C++ with wrappers for ROS and Python. It provides utilities for common tasks, such as generating a lane connectivity graph, matching vehicles onto the correct lanes even in difficult scenarios and converting global to local coordinates. For more details on the concept and utility of the Lanelet2 format, we refer to Poggenhans *et al.* [7] and for the comparison with other formats to Althoff *et al.* [9] and Bao *et al.* [26].

Many popular datasets, such as nuScenes [6] and Lyft prediction [22] provide their own map format together with utility tools to improve usability. We argue, that a comprehensive and flexible map format such as Lanelet2 is well-suited for applications in prediction and planning. Furthermore, using a unified open-source map format eases data processing and could accelerate research by the community.

Applications High-definition road maps are commonly used for prediction and trajectory planning in the context of autonomous vehicles. Generally, we can distinguish two types of approaches for processing and representing map data. Raster-based methods, such as [5], [28]–[30], analyze raster-based representations of a scene, often using CNNs. However, learning important features from such a high-dimensional space and achieving scale-, rotation- and road-layout invariance can be challenging. Moreover, they have a finite resolution, large memory consumption when used for long distances, and finally, memory consumption grows linearly in the number of channels that are used.

Vector-based methods, on the other hand, exploit the graph structure of the underlying problem by representing the entities as nodes and their relation by edges. Examples for such approaches include VectorNet [31], LaneGCN [32], P2T [33], Autobot [34], and THOMAS [35]. However, these approaches often only use centerlines to represent lanes and are not able to directly exploit details of the road topology, e.g. by considering all valid map-based driving options. Deo *et al.* [36] predict possible anchor trajectories using graph-based policies and condition the final prediction on one such policy. The recently proposed FRM [4] focuses explicitly on reasoning over future relationships of traffic participants.

Our contributions can enrich vector-based methods by providing spatial semantic information as well as diverse map-based anchor paths. In contrast to Deo *et al.* [36], anchor trajectories can be provided purely map-based without the necessity of training data. Moreover, our lane connectivity graph can facilitate reasoning over future relationships as in [4], since semantic relationships of traffic participants can be directly inferred from the HD map.

Semantic Information Rather than using only the agent's state information such as position and velocity, a number of approaches try to exploit semantic information about the agents, their interactions and the environment in which they operate at a given time. Such information could be of various types: (1) driving maneuvers such as *turning, passing, parking*; (2) participant relations such as *front, behind, left, right*; (3) environment relations such as *isOnLane, partOfRoad* and so forth. The goal of these approaches is to improve the accuracy and robustness of motion prediction by considering the dynamics of the environment and incorporating information of road networks and the behavior of the objects within it. Spaccapietra *et al.* [37] present an early work to conceptually model and enrich trajectories with fine-grained semantic annotations by decomposing it into a series of moves and stops such as *moveAt, stopAt, nextTo, orientation*. This is later represented into a formal ontology [38] with the aim of allowing structuring and querying of the trajectory data. Following this idea, Hu *et al.* [39] developed a geo-ontology design pattern for representing semantic trajectories which can be utilized in different domains related to navigation and monitoring. An approach which uses prior knowledge of driving scenarios modelled in an ontology to improve trajectory prediction is presented in [40]. A survey on knowledge graph based

[1] See https://www.asam.net/standards/detail/opendrive/

[2] For details we refer to the OpenDRIVE manual and the analysis of Poggenhans *et al.* [7].

[3] See https://github.com/ApolloAuto/apollo.

methods for automated driving can be found in [41]. Finally, a comprehensive approach for integrating heterogeneous information such as agent interactions and road topology from various autonomous driving datasets is proposed in [42].

3. Map conversion

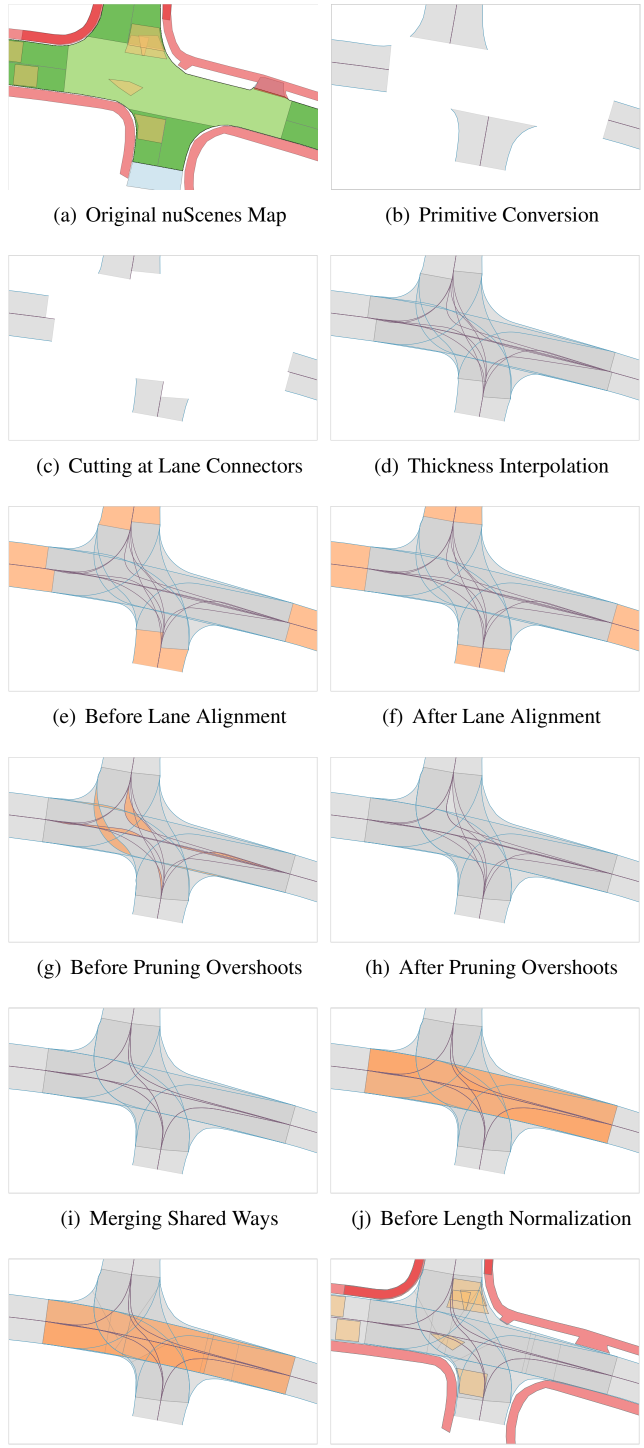
We convert the nuScenes map [6] to the Lanelet2 [7] format in multiple steps, while preserving correspondence to the original nuScenes IDs. The conversion process is summarized in Fig. 2 and the details are presented below.

3.1. Primitive Conversion

Lanelet2 uses primitives different from the ones defined by nuScenes. Thus, we convert all relevant nuScenes primitives into the new format. We start by converting all available nodes in the nuScenes dataset to *Node* entities in the OSM format. Secondly, we parse all lanes to lanelets, which are defined by two line-strings, so-called *Ways*, that represent the left and right border of the given road segment, and a *Relation* entity, which combines the ways to a lanelet. We split the lane boundary that is provided as a linear ring for each nuScenes lane at the given start and stop lines. To assign the attributes *left* and *right* reliably, we generate the directed centerline between start and stop line and verify whether a boundary element is connected with the centerline in a clockwise or anticlockwise loop. The nuScenes input map section is visualized Fig. 2(a) and the resulting map is visualized in Fig. 2(b).

3.2. Centerline Interpolation

After converting all primitives to the Lanelet2 format, information at intersections is still missing, since the nuScenes dataset does not contain lane data for this case. Instead, the dataset provides lane connectors which encode the centerline as a parametric curve to represent the connectivity at intersections. We create additional lanelets from the nuScenes lane connectors to cover the intersections. Transforming (i.e. scaling, translating and rotating) the shape of the centerline that is given by the lane connectors to match the start and end points of their incoming and outgoing lanes is problematic for the following reasons: (1) lane connectors are not aligned properly with their respective incoming and outgoing lanes and (2) the lane width frequently varies between incoming and outgoing lanes. To resolve the alignment issue, we cut the incoming lane at the start point and the outgoing lane at the end point of their respective lane connector orthogonal to the centerline. This reduces the length of the lanes and temporarily removes map information, however, the proper alignment enables a better interpolations in the subsequent step. In the case of multiple outgoing or incoming lanes, we cut off the union of all redundant areas. The resulting map is visualized in Fig. 2(c).



(a) Original nuScenes Map (b) Primitive Conversion (c) Cutting at Lane Connectors (d) Thickness Interpolation (e) Before Lane Alignment (f) After Lane Alignment (g) Before Pruning Overshoots (h) After Pruning Overshoots (i) Merging Shared Ways (j) Before Length Normalization (k) After Length Normalization (l) Final Map

Figure 2. Overview of the map conversion process: Step-by-step improvement of the map quality.

To ensure realistic lane geometries at intersections, we must also cope with the fact that lane thickness can vary between the incoming and outgoing lane. This is done by generating two road segments along the centerline of the lane connector: one with the thickness of the incoming lane and one with the thickness of the outgoing lane. By interpolating both generated road segment boundaries, we arrive at a smooth thickness interpolation as visualized in Fig. 2(d).

3.3. Lane Alignment

A complete representation of the road topology is necessary to infer the set of anchor paths at every given position in the map. The Lanelet2 format implicitly defines the road topology via shared primitives. The *following relation* of lanes is defined by matching start and end points of the respective lanes and a *lateral relation* is represented by a shared *Way* between two lanelets. Such proper alignment is frequently missing in the nuScenes map. To remedy this and thus, make the full road topology easily accessible, we implement an iterative cutting method. To enable the iterative cutting method, we first need to discover which lanes are parallel. This information is inferred from four different sources: (1) lane dividers, (2) road dividers, (3) geometric proximity and (4) at intersections from the parallelism of predecessors and successors. In the first two cases, the information of the nuScenes dataset is directly mapped to create lateral associations between lanes. Moreover, we analyze the geometric similarity and proximity of lanes to detect parallelism. Finally, at intersections, parallelism is inferred by checking if both predecessors and successors are parallel. If that is the case, we also create a lateral association between the two intersection elements.

The iterative cutting method is then applied to groups of laterally associated, i.e. parallel, lanes. We iteratively consider pairs of lanes within such a group and cut the longer one to the length of the shorter one. This procedure is repeated, until all lanes within the group have equal lengths (cf Fig. 3). Afterwards, the lane partitions that were removed during cutting are added to their longitudinally connected lanelets. The comparison between the lane alignment steps is visualized in Fig. 2(e) and 2(f).

3.4. Pruning of Overshooting Lanes

Especially at intersections, lanes for different driving options can overshoot as visualized in Fig. 2(g). We remove these overshoots since they alter the road geometry, i.e. create non-existent lanelet overlaps. The overshoot removal considers each lanelet in our map independently and we denote the considered lanelet in the following as *base lanelet*. For each of the successors of the base lanelet, we compute its approximate change in direction relative to the base lanelet by calculating the cosine similarity of the respective centerlines. Afterwards, we order the list of successors w.r.t. this change in direction in order to retain straight and prune curved lanelets. We consider all pairs of successors within

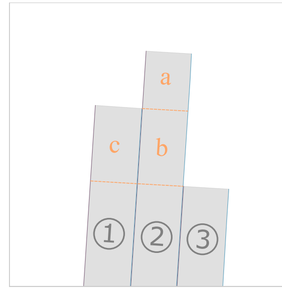


Figure 3. Example for the iterative cutting of lanes. Starting with lane (1) and (2), we first cut the longer (2) to the length of the shorter lane (1) by cutting off area (a). This procedure is repeated until all lanes within the considered group have the same length, i.e. $[(2), (3)] \rightarrow$ cut off (b) and $[(1), (2)] \rightarrow$ cut off (c).

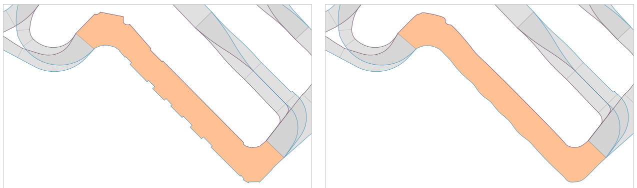
this list without repeated elements in sorted order. For each such pair, the successor with less change in direction is used as reference, and the other successor is labelled for pruning. The pruning step considers each way separately. We determine all points of intersection between the two successors. If there are multiple intersection points, we choose the one that is at the largest distance from the base lanelet. The way of the overshooting lane is projected for each dimension x and y onto the reference way up to the intersection point. Finally, since this projection can lead to sharp edges, we apply one-dimensional Gaussian filtering in the vicinity of the intersection point. Vicinity is defined as $d = \min(4\text{m}, 0.25l)$, where l is the length of the centerline. Since the Gaussian filter can create another marginal overshoot, we apply the pruning again without filtering. The same procedure can be applied for predecessors. Moreover, we consider parallel lanes for both cases (successors and predecessors) by exploiting the neighborhood relation of the road topology. An example map after pruning is visualized in Fig. 2(h).

3.5. Merging Neighboring Lanes

Up to now, every lanelet consists of two unique ways. As described in Sec. 3.3, Lanelet2 infers the roads' topology implicitly via shared ways and nodes. Thus, we utilize the previously established lateral neighborhood relations to merge shared ways. In case the two ways do not coincide perfectly, we perform linear interpolation. The resulting map is visualized in Fig. 2(i).

3.6. Lane Length Normalization

We follow Deo *et al.* [36] and normalize the map by restricting the maximum lanelet length to $20\,\text{m}$. To perform realistic cuts across the map, we apply them by iterating over connected components, i.e. laterally associated lanelets. For each such connected component, we first determine the necessary number of cuts by considering the maximum of the way lengths. Subsequently, we compute



(a) Before Smoothing (b) After Smoothing

Figure 4. Since the nuScenes map contains lane geometries with sharp edges (a), we apply Gaussian filtering to smooth the respective shapes (b).

equidistant cutting points. One way is chosen as reference to calculate the orthogonal cutting lines through the cutting points. We approximate the tangent at a given cutting point using a $4\,\mathrm{m}$ vicinity, since the centerline may contain sharp edges. Afterwards, we cut all ways orthogonally w.r.t. the approximated tangent. Finally, we create and update relations where necessary. A visual comparison of the map before and after lane normalization can be found in Fig. 2(j) and 2(k).

3.7. Post-processing

Since the map at this point still contains some lanelets with sharp edges, which do not comply with the Lanelet2 requirements, we apply shape smoothing as visualized in Fig. 4. For this purpose a one-dimensional Gaussian filter with $\sigma = 2$ is used in each dimension. Finally, some remaining edge cases regarding the road geometry and topology are fixed manually. These issues are mostly caused by rare edge cases (e.g. due to uncommon shapes) and inaccuracies in the original nuScenes map.

3.8. Lane Dividers

Ways form the boundary of lanelets and at the same time contain information about the lane divider type, e.g. solid, dashed, double solid, which prescribes the possibility of lane changes. This information is essential for a valid road connectivity graph and we thus, add lane divider types to all ways. First, we transfer all information available in the nuScenes dataset. Next, at all intersections we choose virtual lines, to indicate the prohibition of lane changes. Finally, for all other ways we use a solid line as default. This last group mostly comprises road boundaries.

3.9. Other Non-geometric Layers

Finally, we add the geometric information of stop lines, traffic lights, pedestrian crossings and road surroundings, i.e. walkways and carpark areas, to the map. The final map version is visualized in Fig. 2(l). We export the map to the Lanelet2 format and use the UTM projection to convert coordinates into the latitude, longitude format. The automated conversion process takes less than $5\,\mathrm{min}$ per map on consumer grade hardware.

4. Anchor Path Generation

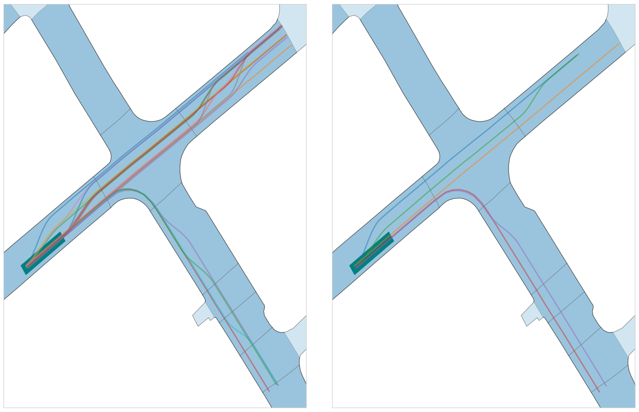
Trajectory prediction frequently leads to mode collapse, where multiple predictions resemble each other closely. In order to diversify the prediction capabilities of existing approaches, we provide a simple, yet effective algorithm for **d**iverse **m**ap-based **a**nchor **p**ath generation and diversity-based sorting, called DMAP.

For an agent at any position on the map, the lane connectivity graph prescribes all its valid driving options along the road. We call this set of valid driving options the set of anchor paths for a given starting point. Every anchor path constitutes an ordered list of lanelet IDs, which can be used for arbitrary interpolations, e.g. smoothly merging an agent onto the centerline or driving with a constant offset from it.

For the task of trajectory prediction, having a diverse set of anchor paths can be very beneficial as it helps to reduce mode collapse and it additionally provides an initial trajectory, from which only offsets need to be estimated. Two parameters need to be freely configurable to use the anchor path generation in practice: the number of anchor paths that should be generated and the desired driving distance that should be covered by an anchor path. The former depends on the number of modes that should be estimated, while the latter depends on the expected velocity profile of the agent.

To generate the relevant set of anchor paths, we recursively query the lane connectivity graph for adjacent lanelets to find all paths with a specified maximum length. During this search, we do not allow lane changes in opposite directions to reduce the search space. More precisely, once an anchor contains a lane change to the left, no adjacent lanelets that would require a lane change to the right are considered and vice versa. Note, that these anchor paths usually exceed the desired length but can be easily cut. Depending on the local road geometry this might yield too many anchor paths. For this case, we present a simple, yet effective approach for downsampling the number of anchors paths to the desired number. We create a similarity matrix for all pairs of anchor paths by calculating the Intersection over Union (IoU) between the two anchor path centerlines, each with an additional $1\,\mathrm{m}$ buffer. We determine an order for all anchors by iteratively removing the least important anchor based on diversity, which is defined as the sum over the IoUs with all the remaining anchor paths. Finally, by choosing the first n anchor paths from the sorted list, we are able to provide diverse anchors. The procedure is visualized in Fig. 5.

This anchor generation approach assumes knowing the lanelet on which the considered agent is currently driving. We infer this information by using a probabilistic matching based on the agent position and its angle w.r.t. the center-



(a) Before Reduction (b) After Reduction

Figure 5. Example for the reduction from 18 available anchor paths of length $100\,\mathrm{m}$ (a) to the most diverse five (b). We highlight the start lanelet in green and visualize all anchor paths with random color coding.

line. We allow a maximum distance of the vehicle bounding box of $0.5\,\mathrm{m}$ to consider a lanelet as a candidate and use the Lanelet2 library to calculate the Mahalanobis distance for each candidate [43]. For the calculation, we assume a variance of $0.5\,\mathrm{m}$ in x and y, and a covariance of 5 degrees. Subsequently, we consider all matches that deviate at most $5\,\%$ from the lowest observed Mahalanobis distance to discard unlikely matches. We convert the remaining candidates into a probability distribution by normalizing the Mahalanobis distance. To finally sample anchor trajectories, we generate and sort the anchor paths for each lanelet candidate. The number of anchor paths for each lanelet candidate is then sampled according to the previously calculated probability distribution. Since our anchor paths are sorted by diversity, always the first n anchor paths are selected per lanelet candidate.

5. Evaluation

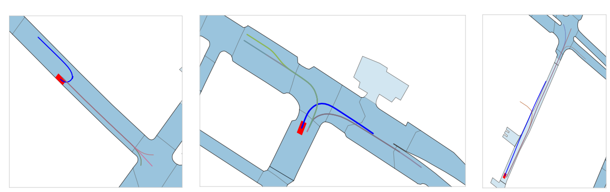
We evaluate the quality of our HD map and the associated anchor path generation approach in the following. We present quantitative evidence that our high-quality anchor paths provide a strong baseline for lateral diversity in Sec. 5.1. Subsequently, we present qualitative examples of our anchor paths in Sec. 5.2 and insights into computational costs in Sec. 5.3.

5.1. Quantitative Results

To provide quantitative evidence of the quality of our HD map and our anchor retrieval approach DMAP, we evaluate it for motion prediction on the nuScenes dataset. We match each agent probabilistically onto the map and retrieve the specified number of anchor paths as described in Sec. 4. Since our approach only covers lateral diversity, i.e. a diverse set of possible driving options, we evaluate it assuming the average past velocity as future velocity (DMAP$_{\bar{v}}$) and by using the ground truth velocity profile information (DMAP*). Tab. 1 shows an unfair comparison with the state-of-the-art, where DMAP* is the only approach having access to the ground truth velocity profiles. The results provide evidence of the high-quality of our HD map and anchor retrieval approach. DMAP* achieves the lowest MinADE$_5$ suggesting that our anchor paths closely resemble the possible driving options for an agent. The particularly low MissRate$_{5,2}$ implies that the overall resemblance between the ground truth trajectories and our anchor paths is large, and thus, that our HD map accurately depicts the road geometry and topology. Performance of MinADE and MissRate$_{x,2}$ does not improve significantly when considering 10 instead of 5 anchor paths. This indicates that our diversity-based anchor sorting successfully identifies the general driving options and that to further improve results fine-grained adjustments to our anchor paths are needed. Moreover, our approach is the only one with an Offroad Rate of 0.00, since our trajectories are directly based on the road geometry. Finally, comparing DMAP* with DMAP$_{\bar{v}}$, it becomes obvious that velocity profile estimation is crucial to achieve high-quality motion prediction results.

5.2. Qualitative Results

To find potential inaccuracies in our map, we manually inspected all scenes where the MinADE across all available anchor paths is larger than 4. We found five such scenes of which three occurred at the same junction. Thus, we visualize only the three relevant scenarios in Fig. 6. The error cases include illegal driving maneuvers and agents driving outside the lane or road. In most cases with a MinADE between 2 and 4, deviations seem to stem from the fact that either agents do not follow the centerline closely or lane changes were performed at locations different from the ones specified by our anchor paths.



(a) Illegal Maneuver (b) Outside Lane (c) Offroad

Figure 6. Example cases where the MinADE across all our anchor paths is above 4. The ground truth trajectory is highlighted in blue, the start position of the agent in red and our anchor paths are visualized with random colors.

Model	Uses Velocity GT	MinADE_5	MinADE_{10}	$\text{MissRate}_{5,2}$	$\text{MissRate}_{10,2}$	Offroad Rate
$\text{DMAP}_{\bar{v}}$ (ours)	×	4.83	4.83	0.91	0.91	**0.00**
P2T [33]	×	1.45	1.16	0.64	0.46	0.03
Autobot [34]	×	1.37	1.03	0.62	0.44	0.02
THOMAS [35]	×	1.33	1.04	0.55	0.42	0.03
PGP [36]	×	1.27	0.94	0.52	0.34	0.03
FRM [4]	×	1.18	**0.88**	0.48	0.30	0.02
DMAP* (ours)	✓	**1.09**	1.07	**0.19**	**0.18**	**0.00**

Table 1. Unfair comparison with the state-of-the-art on the nuScenes dataset. *Note, that our approach is the only one that uses the ground truth (GT) velocity profile and thus, can only be evaluated on the validation split.

5.3. Performance Evaluation

To analyze the computational cost of DMAP, we present a 2D histogram showing the correlation between the number of $100\,\mathrm{m}$ long anchor paths found for a given start lanelet and the corresponding computation time to find and sort all anchors in Fig. 7. Note, that our implementation is not optimized for speed, but rather should facilitate reusing and adopting our code. The prescribed $100\,\mathrm{m}$ anchor path length corresponds to driving $60\,\mathrm{km/h}$ over a $6\,\mathrm{s}$ horizon. We observe that $57.23\,\%$ of lanelet have less than five anchors, and $79.13\,\%$ less than 10. Regarding the computation time, the 25^{th}, 50^{th} and 75^{th} percentiles are at $47.18\,\mathrm{ms}$, $241.41\,\mathrm{ms}$ and $935.60\,\mathrm{ms}$, respectively. Furthermore, we observe a very high number of anchor paths in scenarios with numerous lane change possibilities in the vicinity of intersections, as visualized in Fig. 8. Note, that these results are based on specified start lanelets while in applications such as motion prediction, matching an agent onto the HD map is necessary. This matching process can yield multiple lanelet candidates, for each of which all anchor paths should be considered.

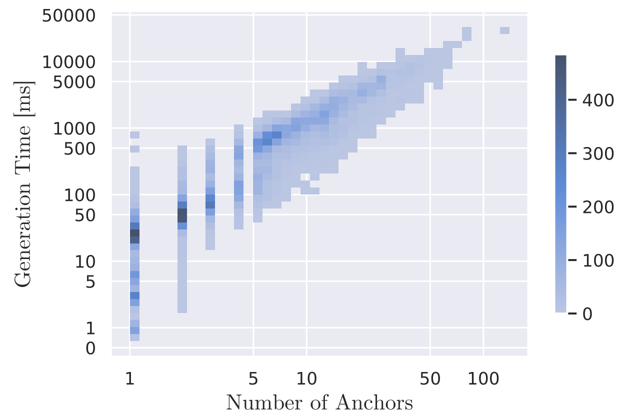


Figure 7. 2D histogram showing the correlation between the number of available anchor paths of length $100\,\mathrm{m}$ and the computational cost by considering all lanelets across all maps of the nuScenes dataset as starting point.

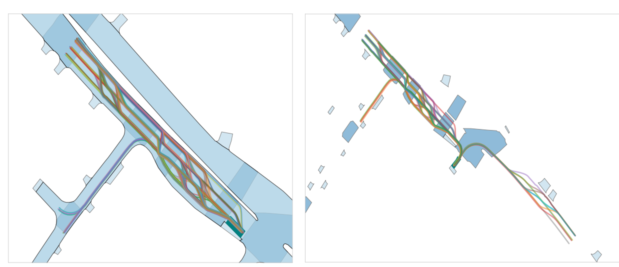


Figure 8. Visualization of edge cases with 140 (left) and 63 (right) possible anchor paths. We added a slight random lateral offset to each anchor path to improve visibility.

6. Conclusion

Motion planning and prediction are key components necessary for the success of fully autonomous driving. Vector representations that interpret agents and lanes as nodes of a graph have recently gained strong momentum. In this work, we presented practical and easy-to-apply enhancements to the existing HD map of the nuScenes dataset [6] which can be readily leveraged for such graph-structured approaches.

We first presented our approach to enrich and convert the existing map into the popular open-source map format Lanelet2 [7]. This enhanced map allows easy access to the full road geometry and topology, which enables the usage of spatial semantic information (e.g. agents driving on intersecting roads) and map-based anchor paths. Moreover, we presented DMAP, a simple, yet effective approach for generating map-based anchor paths and sorting them according to their lateral diversity. Our evaluation shows that combining DMAP with the ground truth velocity profile information yields high-quality results for motion prediction (MinADE_5=1.09, $\text{MissRate}_{5,2}$=0.18, Offroad rate=0.00). This implies that the road geometry and topology are accurately represented by our HD map. Future approaches can leverage the potential of our HD map and DMAP by focusing on learning to select the most suitable anchor paths and estimating velocity profiles.

References

[1] C. Thorpe, M. Hebert, T. Kanade, and S. Shafer, "Vision and navigation for the Carnegie-Mellon Navlab," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1988.

[2] J. Janai, F. Güney, A. Behl, and A. Geiger, "Computer Vision for Autonomous Vehicles: Problems, Datasets and State of the Art," *Foundations and Trends® in Computer Graphics and Vision*, 2020.

[3] Y. Li and J. Ibanez-Guzman, "Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems," *IEEE Signal Processing Magazine*, 2020.

[4] D. Park, H. Ryu, Y. Yang, J. Cho, J. Kim, and K.-J. Yoon, "Leveraging Future Relationship Reasoning for Vehicle Trajectory Prediction," in *International Conference on Learning Representations*, 2023.

[5] Y. Chai, B. Sapp, M. Bansal, and D. Anguelov, "MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction," in *Proceedings of the Conference on Robot Learning*, 2020.

[6] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, *et al.*, "nuScenes: A Multimodal Dataset for Autonomous Driving," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[7] F. Poggenhans, J.-H. Pauls, J. Janosovits, S. Orf, M. Naumann, F. Kuhnt, *et al.*, "Lanelet2: A high-definition map framework for the future of automated driving," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[8] P. Bender, J. Ziegler, and C. Stiller, "Lanelets: Efficient map representation for autonomous driving," in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, 2014.

[9] M. Althoff, S. Urban, and M. Koschi, "Automatic Conversion of Road Networks from OpenDRIVE to Lanelets," in *2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, 2018.

[10] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, *et al.* "INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps." arXiv: `arXiv:1910.03088`. (2019).

[11] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[12] Y. Xu, W. Shao, J. Li, K. Yang, W. Wang, H. Huang, *et al.*, "SIND: A Drone Dataset at Signalized Intersection in China," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022.

[13] M. Naumann, F. Poggenhans, M. Lauer, and C. Stiller, "CoInCar-Sim: An Open-Source Simulation Framework for Cooperatively Interacting Automobiles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[14] S. Kato, S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, *et al.*, "Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems," in *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, 2018.

[15] R. Queiroz, T. Berger, and K. Czarnecki, "GeoScenario: An Open DSL for Autonomous Driving Scenario Representation," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019.

[16] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, *et al.*, "LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020.

[17] S. Maierhofer, M. Klischat, and M. Althoff, "CommonRoad Scenario Designer: An Open-Source Toolbox for Map Conversion and Scenario Creation for Autonomous Vehicles," in *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, 2021.

[18] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[19] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, 2013.

[20] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, *et al.*, "The ApolloScape Dataset for Autonomous Driving," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2018.

[21] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, *et al.*, "Scalability in Perception for Autonomous Driving: Waymo Open Dataset," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[22] J. Houston, G. Zuidhof, L. Bergamini, Y. Ye, L. Chen, A. Jain, *et al.*, "One thousand and one hours: Self-driving motion prediction dataset," in *4th Conference on Robot Learning, CoRL*, 2020.

[23] M.-F. Chang, J. Lambert, P. Sangkloy, J. Singh, S. Bak, A. Hartnett, *et al.*, "Argoverse: 3D Tracking and Forecasting With Rich Maps," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[24] B. Wilson, W. Qi, T. Agarwal, J. Lambert, J. Singh, S. Khandelwal, *et al.*, "Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting," 2023. arXiv: arXiv:2301.00493.

[25] L. Chen, Y. Li, C. Huang, B. Li, Y. Xing, D. Tian, *et al.*, "Milestones in Autonomous Driving and Intelligent Vehicles: Survey of Surveys," *IEEE Transactions on Intelligent Vehicles*, 2022.

[26] Z. Bao, S. Hossain, H. Lang, and X. Lin. "High-Definition Map Generation Technologies For Autonomous Driving." arXiv: arXiv:2206.05400. (2022), preprint.

[27] C. W. Gran, "HD-Maps in Autonomous Driving," M.S. thesis, Norwegian University of Science and Technology (NTNU), 2019.

[28] S. Casas, W. Luo, and R. Urtasun, "IntentNet: Learning to Predict Intention from Raw Sensor Data," in *Proceedings of The 2nd Conference on Robot Learning*, 2018.

[29] H. Cui, V. Radosavljevic, F.-C. Chou, T.-H. Lin, T. Nguyen, T.-K. Huang, *et al.*, "Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019.

[30] J. Liang, L. Jiang, K. Murphy, T. Yu, and A. Hauptmann, "The Garden of Forking Paths: Towards Multi-Future Trajectory Prediction," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[31] J. Gao, C. Sun, H. Zhao, Y. Shen, D. Anguelov, C. Li, *et al.*, "VectorNet: Encoding HD Maps and Agent Dynamics From Vectorized Representation," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[32] M. Liang, B. Yang, R. Hu, Y. Chen, R. Liao, S. Feng, *et al.*, "Learning Lane Graph Representations for Motion Forecasting," in *Computer Vision – ECCV 2020*, 2020.

[33] N. Deo and M. M. Trivedi. "Trajectory Forecasts in Unknown Environments Conditioned on Grid-Based Plans." arXiv: arXiv:2001.00735. (2021).

[34] R. Girgis, F. Golemo, F. Codevilla, M. Weiss, J. A. D'Souza, S. E. Kahou, *et al.* "Latent Variable Sequential Set Transformers For Joint Multi-Agent Motion Prediction." arXiv: arXiv:2104.00563. (2022), preprint.

[35] T. Gilles, S. Sabatini, D. Tsishkou, B. Stanciulescu, and F. Moutarde, "THOMAS: Trajectory Heatmap Output with learned Multi-Agent Sampling," in *International Conference on Learning Representations*, 2022.

[36] N. Deo, E. Wolff, and O. Beijbom, "Multimodal trajectory prediction conditioned on lane-graph traversals," in *Conference on Robot Learning*, PMLR, 2022.

[37] S. Spaccapietra, C. Parent, M. L. Damiani, J. A. F. de Macêdo, F. Porto, and C. Vangenot, "A conceptual view on trajectories," *Data & Knowledge Engineering*, 2008.

[38] Z. Yan, J. Macedo, C. Parent, and S. Spaccapietra, "Trajectory ontologies and queries," *Transactions in GIS*, 2008.

[39] Y. Hu, K. Janowicz, D. Carral, S. Scheider, W. Kuhn, G. Berg-Cross, *et al.*, "A geo-ontology design pattern for semantic trajectories," in *Spatial Information Theory - 11th International Conference, COSIT. Proceedings*, 2013.

[40] S. Wang, P. Zhao, B. Yu, W. Huang, and H. Liang, "Vehicle trajectory prediction by knowledge-driven LSTM network in urban environments," *Traffic Safety in Intelligent and Connected Environment*, 2020.

[41] J. Luettin, S. Monka, C. Henson, and L. Halilaj, "A Survey on Knowledge Graph-Based Methods for Automated Driving," in *Knowledge Graphs and Semantic Web*, 2022.

[42] L. Halilaj, J. Luettin, C. Henson, and S. Monka, "Knowledge graphs for automated driving," in *5th IEEE International Conference on Artificial Intelligence and Knowledge Engineering, AIKE*, 2022.

[43] D. Petrich, T. Dang, D. Kasper, G. Breuel, and C. Stiller, "Map-based long term motion prediction for vehicles in traffic environments," in *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*, 2013.