Producing and Leveraging Online Map Uncertainty in Trajectory Prediction

Xunjiang Gu\textsuperscript{1} Guanyu Song\textsuperscript{1} Igor Gilitschenski\textsuperscript{1,2} Marco Pavone\textsuperscript{3,4} Boris Ivanovic\textsuperscript{3}
\textsuperscript{1}University of Toronto \textsuperscript{2}Vector Institute \textsuperscript{3}NVIDIA Research \textsuperscript{4}Stanford University
\{alfred.gu, guanyu.song\}@mail.utoronto.ca, gilitschenski@cs.toronto.edu,
\{mpavone, bivanovic\}@nvidia.com, pavone@stanford.edu

Abstract

High-definition (HD) maps have played an integral role in the development of modern autonomous vehicle (AV) stacks, albeit with high associated labeling and maintenance costs. As a result, many recent works have proposed methods for estimating HD maps online from sensor data, enabling AVs to operate outside of previously-mapped regions. However, current online map estimation approaches are developed in isolation of their downstream tasks, complicating their integration in AV stacks. In particular, they do not produce uncertainty or confidence estimates. In this work, we extend multiple state-of-the-art online map estimation methods to additionally estimate uncertainty and show how this enables more tightly integrating online mapping with trajectory forecasting\textsuperscript{1}. In doing so, we find that incorporating uncertainty yields up to 50\% faster training convergence and up to 15\% better prediction performance on the real-world nuScenes driving dataset.

1. Introduction

A critical component of autonomous driving is understanding the static environment, e.g., road layout and connectivity, surrounding the autonomous vehicle (AV). Accordingly, high-definition (HD) maps have been developed to capture and provide such information, containing semantics like road boundaries, lane dividers, and road markings at the centimeter level. In recent years, HD maps have proven to be indispensable for AV development and deployment, seeing widespread use today [35]. However, HD maps are costly to label and maintain over time, and they can only be used in geofenced areas, limiting AV scalability.

To address these issues, many recent works turn to estimating HD maps online from sensor data. Broadly, they aim to predict the locations and classes of map elements, typically as polygons or polylines, all from camera images and LiDAR scans. However, current online map estimation methods do not produce any associated uncertainty or confidence information. This is problematic as it causes

\textsuperscript{1}Code: https://github.com/alfredgu001324/MapUncertaintyPrediction

Figure 1. Producing uncertainty from online HD map estimation methods and incorporating it in downstream modules yields a variety of benefits. \textbf{Left}: Ground truth HD map and agent positions. \textbf{Middle}: HiVT [41] predictions using the map output by MapTR [22]. \textbf{Right}: HiVT [41] predictions using the map output by MapTR [22] augmented with point uncertainties (which are large as the left road boundary is occluded by parked vehicles).
2. Related Work

2.1. Online Map Estimation

The goal of online map estimation is to predict a representation of the static world elements surrounding an autonomous vehicle from sensor data. Initial works focused on producing 2D birds-eye-view (BEV) rasterized semantic segmentations as world representations by unprojecting to 3D and collapsing along the Z-axis [26, 28] or by leveraging cross-attention in geometry-aware Transformer [34] models [2, 20].

Recently, vectorized map estimation approaches have emerged, extending BEV rasterization approaches with decoders that regress and classify polyline and polygon map elements (among other curve representations [29]). Initial works such as HDMapNet [19] and SuperFusion [7] propose to fuse LiDAR point clouds and RGB images into a common BEV feature frame followed by a hand-crafted post-processing step to produce polyline map elements. To remove the reliance on hand-crafted post-processing, VectorMapNet [25] and InstaGraM [32] introduce end-to-end models for vectorized HD map learning. Further improvements to avoid information loss from key-point sampling along polylines are proposed by PivotNet [6].

The MapTR line of work [22, 23] and extensions [37] formulate vectorized HD map estimation as a point set prediction task, yielding significant improvements in mapping performance. Most recently, StreamMapNet [38] focuses on incorporating temporal data from past frames, enabling HD map estimation from streaming data online. In each of these methods, however, there is no uncertainty or confidence information provided to downstream consumers, making it difficult to distinguish between accurate and errant map elements.

2.2. Map-Informed Trajectory Prediction

Early trajectory prediction works predominantly leveraged rasterized maps to represent and encode scene context [30]. Typically, a Convolutional Neural Network (CNN) encodes the BEV map tensor into a vector which is concatenated with other scene context (e.g., agent state histories) and passed through the rest of the model [10, 16, 27, 31, 39].

Recently, trajectory prediction works have increasingly turned to encoding raw polyline information from vectorized HD maps, achieving significant performance improvements. Initial approaches [9, 11, 12, 21, 40] applied Graph Neural Networks (GNNs) to encode lane polylines and their influence on agent motion. Extending this idea, most current approaches adopt Transformer [34] architectures with map-agent cross-attention [5, 13, 24, 41] to achieve state-of-the-art performance.

In contrast to rasterized approaches, directly encoding vectorized HD maps removes information bottlenecks (discretization in rasterization loses fine geometric details) and enables a direct focus on map elements that are most relevant to agents (as opposed to encoding an entire BEV map of the scene). One core drawback, however, is a lack of uncertainty representations. While uncertainty can be naturally encoded in a BEV format (e.g., as a probability heatmap), how to best represent it in vectorized HD maps remains an open question. Our work addresses this problem by proposing a simple yet general methodology to estimate vectorized HD map uncertainty and represent it.

2.3. End-to-End Driving Architectures

End-to-end AV architectures are a promising approach to developing integrated stacks which account for mapping uncertainty. Recently, UniAD [14], VAD [18], and OccNet [33] have shown how to incorporate both rasterized and vectorized HD map estimation within end-to-end training. UniAD [14] and OccNet [33], for instance, approach online mapping as a dense prediction task, predicting the per-pixel or per-voxel semantics of map elements, whereas VAD produces vectorized HD map representations. In each architecture, mapping serves as both an auxiliary training task and an internal static world representation that informs downstream components. While these methods yield the most integrated stacks, they only implicitly account for uncertainty. Accordingly, our work can be incorporated within end-to-end stacks to provide an explicit model of uncertainty and improve overall system performance.

3. Producing Online Map Uncertainty

As mentioned in Sec. 2, there are many potential architectures for vectorized HD map estimation and accordingly many potential sources of uncertainty (e.g., perspective-to-BEV transformations, multi-sensor fusion, polyline vertex locations, element connectivity). However, as depicted in Fig. 2, nearly all architectures utilize a point regression and classification head to predict the locations of polyline vertices and identify what kind of map element they are (e.g., lane line, stop line, road boundary), respectively. Thus, to ensure the general applicability of our proposed uncertainty formulation to a variety of map estimation approaches, we focus on extending this common output structure to additionally produce location and class uncertainty.

**Regression Uncertainty.** Vectorized HD map estimation models typically employ a simple MLP architecture for their regression heads. For each map element, the regression head produces a 2D vector representing normalized BEV \((x, y)\) coordinates. To transform this into a probabilistic model, we replace the regression head with one that additionally outputs uncertainty parameters associated with the predicted points. While a common choice is to assume Gaussian uncertainty, we found that this yields instabilities during training. Instead, we model each map element vertex \(v = (v_1, v_2)\) with two univariate Laplace distributions.
Accordingly, the joint probability density for a map element $M$ with $V$ vertices, denoted $M = \{v^{(i)}\}_{i=1}^{V}$, is

$$f(M \mid \mu, b) = \prod_{i=1}^{V} \prod_{j=1}^{2} \frac{1}{b^{(i)}_{j}} \exp \left(-\frac{|\mu^{(i)}_{j} - \mu_{j}|}{b^{(i)}_{j}}\right), \quad (1)$$

where $\mu^{(i)}_{j} \in \mathbb{R}$ and $b^{(i)}_{j} \in \mathbb{R}$ are the location and scale parameters of the Laplace distribution for the $j^{th}$ dimension of the $i^{th}$ vertex of the map element.

With its sharper peak and heavier tails, the Laplace distribution is particularly adept at handling outliers compared to the Gaussian distribution. Further, many online map estimation methods are trained using the Manhattan ($\ell_1$) distance as their regression loss [22, 23, 38], making the Laplace distribution a natural choice. As we will show in Sec. 5.2, such a direct uncertainty formulation can already model interesting sources of uncertainty, such as occlusions.

**Classification Uncertainty.** The classification head predicts class confidence scores for each regressed vertex. Since this head is already probabilistic (it is a Categorical distribution), we simply expose the semantic class logits to downstream consumers.

**Training Loss.** To train an uncertainty-producing map estimation model, only the regression loss $L_{\mathcal{R}}$ needs to be changed to a Negative Log-Likelihood (NLL) loss,

$$L_{\mathcal{R}}(M \mid \mu, b) = \sum_{i=1}^{V} \sum_{j=1}^{2} \log \left(2b_{j}^{(i)}\right) + \frac{\mu_{j}^{(i)} - \mu_{j}}{b_{j}^{(i)}}. \quad (2)$$

**Models.** In this work, we extend the MapTR [22], MapTRv2 [23], and StreamMapNet [38] online HD map estimation models to demonstrate the benefits of producing map uncertainty. We choose these approaches as they are all very recent works that achieve state-of-the-art online HD mapping performance. At a high level, MapTR [22] and MapTRv2 [23] are Transformer-based models which adopt an encoder-decoder architecture. As depicted in Fig. 2, they first encode RGB images to a common BEV feature $B \in \mathbb{R}^{H \times W \times C}$ (using the LSS [28]-based BEVPoolv2 [15]). Their map decoders consist of map queries and several decoder layers. Each decoder layer utilizes self-attention and cross-attention to update the map queries before finally decoding them with a (non-probabilistic) regression and classification head. Note that, while MapTRv2 [23] optionally supports LiDAR, we do not use it.

On the other hand, StreamMapNet [38] focuses on operating from streaming data, containing an additional memory buffer that stores prior queries and BEV features which are ego-pose-corrected and combined with queries and BEV features in the current timestep to incorporate temporal information.

Each model produces three types of map elements: road boundary, pedestrian crosswalk, and lane divider. MapTRv2 [23] can additionally output lane centerlines, which have been shown to be critical for trajectory forecasting [4]. Each of these four models predict vectorized map elements within a perception range of 60m longitudinally by 30m laterally (centered on the AV).

### 4. Incorporating Map Uncertainty in Trajectory Prediction

The vast majority of trajectory prediction models employ an encoder-decoder architecture [30], where the encoder encodes scene context (e.g., vectorized map information and agent trajectories) and the decoder leverages such information to predict the future motion of surrounding agents. In the encoder, as depicted in Fig. 2, map element vertices are most commonly encoded as nodes in a GNN (e.g., in DenseTNT [13]) or tokens in Transformer cross-attention (e.g., in HiVT [41]). In either of these models, vertex coordinates are first encoded by an MLP $\phi_v$ before being incorporated in message passing or attention layers. Formally, the $i^{th}$ vertex of a map element $M$ is encoded as $e^{(i)}_{v} = \phi_v(v^{(i)})$.

To incorporate upstream uncertainty information in prediction models, we instead encode the Laplace distribution location $\mu$ and scale $b$ parameters, as well as the class probabilities $c \in \Delta^{C-1}$, yielding

$$e^{(i)}_{\text{unc}} = \phi_v\left(\mu^{(i)}; b^{(i)}; c^{(i)}\right), \quad (3)$$

where $[\cdot; \cdot]$ represents concatenation and $\Delta^{C-1}$ denotes the probability simplex with $C$ classes.

**Models.** We augment the DenseTNT [13] and HiVT [41]
trajectory prediction models to incorporate upstream map uncertainty, choosing these models as they implement the two dominant paradigms of encoding map information: GNNs and Transformers, respectively.

At a high-level, DenseTNT [13] leverages VectorNet [9] to extract features from lanes and agents. It employs a hierarchical GNN consisting of two stages: local information from individual polylines is first aggregated and encoded, followed by global interactions between the resulting polyline node features. DenseTNT [13] then employs a dense goal probability estimation technique to predict the endpoints of trajectories and generates complete trajectories based on the best goal candidates. To augment DenseTNT to incorporate map uncertainty, we integrate map element vertex uncertainty into the lane feature encoding, alongside the vertex coordinates (as in Eq. (3)). These uncertainty-enhanced vectors are then encoded with VectorNet [9].

HiVT [13] similarly encodes scene context in two hierarchical stages: first encoding local context (relative to each agent), followed by global interaction modeling between the local neighborhoods to capture long-range dependencies and scene-level dynamics. The resulting agent embeddings are then decoded with an MLP to produce the parameters of a multimodal trajectory distribution.

We augment HiVT [13] to incorporate map uncertainty by inputting the estimated map as a point set, instead of a vector set as in the original model, enabling the direct incorporation of vertex uncertainty in the encoder. Specifically, the uncertainty (scale parameter $b$) of each point is directly concatenated with the mean values of the point set, which are then encoded by the local neighborhood encoder together with agent trajectory information.

As we will show in Sec. 5.3, incorporating polyline uncertainty directly in this manner enables prediction models to understand when map element estimations may be unreliable and adjust their outputs accordingly, yielding significant accuracy improvements.

5. Experiments

5.1. Experiment Setup

Dataset. We evaluate our probabilistic map estimation and prediction framework on the large-scale nuScenes dataset [1], which provides ground truth (GT) HD maps, sensor data (RGB images), as well as agent trajectories. It consists of 1000 driving scenes with each scene sampled at 2 Hz, and is split into training, validation, and test sets containing 500, 200, and 150 scenes, respectively.

We leverage trajdata [17] to provide a unified interface between vectorized map estimation models and downstream prediction models. To ensure compatibility across prediction models, we upsample nuScenes’ data frequency to 10 Hz (from its original 2 Hz) using trajdata’s time interpolation utilities [17]. This modification provides a denser dataset, thereby facilitating finer-grained analyses and aligning our data more closely with the real-time execution rates of onboard prediction models. Finally, we task each prediction model to predict motion 3 seconds into the future from 2 seconds of history.

Metrics. The Chamfer distance $D_{\text{Ch}}$ is employed to measure the distance between two maps (represented as point sets $S_1$ and $S_2$). Formally,

$$D_{\text{Ch}} = \sum_{x \in S_1} \min_{y \in S_2} \frac{\|x - y\|_2}{|S_1|} + \sum_{y \in S_2} \min_{x \in S_1} \frac{\|y - x\|_2}{|S_2|}.$$  (4)

In line with prior works [22, 23, 38], we adopt Average Precision (AP) as the evaluation metric for our probabilistic map construction of four map elements: road boundary, pedestrian crossing, lane divider, and lane centerlines. Mean AP (mAP) is further calculated as the mean AP under three distinct $D_{\text{Ch}}$ thresholds: 0.5 m, 1.0 m, and 1.5 m.

For trajectory prediction, we evaluate our model on standard metrics adopted by numerous recent prediction challenges [3, 8, 36], specifically minimum Average Displacement Error (minADE), minimum Final Displacement Error (minFDE), and Miss Rate (MR) [3]. For each agent, 6 potential trajectories are output for evaluation. The minADE metric computes the average Euclidean ($\ell_2$) distance in meters across all future time steps between the most accurately predicted trajectory and the ground truth trajectory. Similarly, minFDE calculates the error of only the final predicted time step. The most accurately predicted trajectory is identified based on having the smallest FDE. MR quantifies the proportion of scenarios where the endpoint of the best-predicted trajectory deviates from the ground truth trajectory’s endpoint by more than 2.0 meters.

Data Preprocessing and Training. We standardize all agent and lane features by transforming their coordinates to be relative to ego-vehicle’s position, as well as rotating the scene to make the AV’s heading point up. As a consequence, we also transform the map uncertainty with

$$\sigma_{x'} = \sqrt{\sigma_x^2 \cos^2(\theta) + \sigma_b^2 \sin^2(\theta)},$$

$$\sigma_{y'} = \sqrt{\sigma_y^2 \sin^2(\theta) + \sigma_b^2 \cos^2(\theta)},$$  (5)

where $\theta$ is the rotated angle and $\sigma = \sqrt{2 \cdot b}$ is the Laplace distribution’s standard deviation (derived from its scale parameter $b$). All models are trained using a single NVIDIA GeForce RTX 4090 GPU. For full model hyperparameter settings and training details, please refer to Appendix A.

5.2. Producing Map Uncertainty

Augmenting MapTR [22], MapTRv2 [23] (and its centerpiece-producing version), and StreamMapNet [38] to produce uncertainty does not substantially affect their original mapping performance. We are able to reproduce most models’ published performance within 2% mAP, with some uncertainty-augmented versions even outperforming
the original models. In the following, we analyze the uncertainty output by these map estimation models and identify various sources of uncertainty that our approach captures.

**Uncertainty from Occlusion.** Our proposed uncertainty formulation is able to capture uncertainty stemming from occlusions between the AV’s cameras and the surrounding map elements. As can be seen in Fig. 3, the top right portion of the map (forward and to the right of the AV) is occluded by a red callbox and a grey parked car. Importantly, even though our work only modifies the final output heads, Fig. 3 shows that it is still able to identify when certain map elements are occluded in the input RGB images.

We also observe in Fig. 3e the benefits of StreamMapNet’s memory module: It outputs less uncertainty in the same top-right portion of the map, owing to its incorporation of temporal information from past frames (when map elements were visible). Conversely, MapTR and MapTRv2 are single-frame models and cannot enjoy such benefits.

Similarly, Fig. 4 visualizes a scenario where all models struggle to delineate between driveable road and parking spots in a parking lot (region at the bottom of the map, behind the ego-vehicle). Additionally, in easily-observed parts of the map (the region at the top of the map, in front of the ego-vehicle), all models produce confident predictions with very little uncertainties.

**Uncertainty from Sensor Range.** Another important source of uncertainty in map estimation is the distance from the onboard cameras to the map elements, stemming from the 2D-to-BEV transformation in many mapping models. As can be seen in Fig. 5, map uncertainty generally increases with increasing distance between the vehicle and the corresponding map elements. We can also see that MapTRv2 generally yields lower uncertainties than its predecessor MapTR, matching the fact that MapTRv2 is generally more accurate than MapTR [23]. StreamMapNet’s uncertainty remains relatively constant compared to the other per-frame models, again highlighting the benefits of aggregating temporal information.

Further, note the increase in pedestrian crosswalk uncertainty when MapTRv2 is tasked with estimating lane centerlines. One hypothesis is that lane centerlines frequently pass through pedestrian crosswalks, causing confusion in the model about which polyline to optimize during training.

**Uncertainty from Lighting and Weather.** Fig. 6 and Fig. 12 in Appendix B show the effect of different lighting and weather conditions, respectively, on map estimation un-
Figure 5. Our uncertainty formulation captures the fact that uncertainty generally increases with the distance between the predicted map elements and the AV, owing to the difficulty of resolving the details of faraway objects in images. Error bars show 95% confidence intervals.

Figure 6. Different time-of-day lighting can significantly affect the confidence with which map estimation models predict certain elements, such as pedestrian crossings. Error bars show 95% confidence intervals.

certainty. In Fig. 6, we can see that map elements which are typically lit by street lamps, vehicle headlights, and/or contain reflective surfaces (i.e., lane boundaries and dividers) retain the same level of uncertainty in day and nighttime. Conversely, models predict pedestrian crossings with significantly more uncertainty at night, potentially indicating that they may not have the same consistent lighting at night compared to other elements. Fig. 12 in Appendix B additionally shows that StreamMapNet produces more uncertainty in rainy conditions, indicating potential difficulties in aggregating temporal information due to rain.

Uncertainty from Motion. Finally, Fig. 13 in Appendix B shows that current models do not have any particular lack of confidence across different AV driving speeds. However, nuScenes [1] does not contain much high-speed driving (shown in Figure 9 of [17]), leaving high-speed analyses (e.g., about rolling shutter effects) to future work.

5.3. Incorporating Map Uncertainty in Prediction

To evaluate the effect of incorporating map uncertainty in downstream autonomy stack components, we train DenseTNT [13] and HiVT [41] on the outputs of the aforementioned mapping models with and without our output uncertainty formulation, yielding 16 total combinations.

Prediction Accuracy Improvements. As shown in Tab. 1, for virtually all mapping/prediction model combinations, incorporating uncertainty yields better prediction performance. In general, the improvements in MR are the greatest, indicating that, by incorporating map uncertainty, prediction models can effectively adjust their behaviors to more closely match the ground truth future, especially at the endpoints. Endpoint accuracy is particularly important for trajectory prediction as many methods adopt a two-stage pipeline where the first stage predicts possible endpoints.

Further, although MapTRv2 significantly outperforms MapTR in map estimation [23], there is little resulting difference in prediction performance (in fact, MapTR yields better prediction performance than MapTRv2, see the first two sets of rows in Tab. 1). This indicates that accuracy improvements in upstream map estimation models may not directly improve downstream prediction accuracy.

The best prediction performance across all metrics (by far, in some cases) is achieved when leveraging the lane centerlines output by MapTRv2-Centerline. This confirms the superiority of using centerlines to guide trajectory prediction [4] and indicates where integrated systems can see the most improvement from future map estimation research.

Most interestingly, the performance of DenseTNT trained on maps from MapTRv2-Centerline exceeds the performance of DenseTNT trained on GT lane centerlines (Tab. 3). The reason for this stems from MapTRv2-Centerline sometimes producing multiple centerlines for one lane. For a target-based model such as DenseTNT, multiple centerlines in the same lane provides a richer set of options for endpoint selection, focusing more closely the resulting endpoints within lanes and yielding better predic-
## Table 1. Quantitative prediction results for all 16 mapping/prediction model combinations on the nuScenes [1] dataset. In general, incorporating upstream map uncertainty improves the performance of prediction models, especially for endpoint prediction accuracy.

<table>
<thead>
<tr>
<th>Prediction Method</th>
<th>HiVT [41]</th>
<th>DenseTNT [13]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Online HD Map Method</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MapTR [22]</td>
<td>0.4015 0.8418</td>
<td>0.0981 1.091 2.058 0.3543</td>
</tr>
<tr>
<td>MapTR [22] + Ours</td>
<td>0.3854 (−4%) 0.7909 (−6%)</td>
<td>0.0834 (−15%) 1.089 (0%) 2.006 (−3%) 0.3499 (−1%)</td>
</tr>
<tr>
<td>MapTRv2 [23]</td>
<td>0.4057 0.8499</td>
<td>0.0992 1.214 2.312 0.4138</td>
</tr>
<tr>
<td>MapTRv2 [23] + Ours</td>
<td>0.3930 (−3%) 0.8127 (−4%)</td>
<td>0.0857 (−14%) 1.262 (+4%) 2.340 (+1%) 0.3912 (−5%)</td>
</tr>
<tr>
<td>StreamMapNet [38]</td>
<td>0.3972 0.8186</td>
<td>0.0926 0.9492 1.740 0.2569</td>
</tr>
<tr>
<td>StreamMapNet [38] + Ours</td>
<td>0.3848 (−3%) 0.7954 (−3%)</td>
<td>0.0861 (−7%) 0.9036 (−5%) 1.645 (−5%) 0.2359 (−8%)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>HiVT [41]</th>
<th>DenseTNT [13]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epochs to Convergence</td>
<td></td>
</tr>
<tr>
<td>Map Model</td>
<td>Without Unc.</td>
</tr>
<tr>
<td>MapTR [22]</td>
<td>8</td>
</tr>
<tr>
<td>MapTRv2 [23]</td>
<td>7</td>
</tr>
<tr>
<td>MapTRv2-Centerline [23]</td>
<td>9</td>
</tr>
<tr>
<td>StreamMapNet [38]</td>
<td>6</td>
</tr>
</tbody>
</table>

Table 2. When trained with map uncertainty, DenseTNT [13] consistently converges faster, arriving at equal or better validation performance, irrespective of the upstream mapping model.

<table>
<thead>
<tr>
<th>DenseTNT [13] + Map Model</th>
<th>minADE ↓</th>
<th>minFDE ↓</th>
<th>MR ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>GT Map</td>
<td>0.8809  1.489</td>
<td>0.1903</td>
<td></td>
</tr>
<tr>
<td>MapTRv2-Centerline [23]</td>
<td>0.8466 (−4%) 1.345 (−10%) 0.1520 (−20%)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MapTRv2-Centerline [23] + Ours</td>
<td>0.8135 (−6%) 1.311 (−12%) 0.1593 (−16%)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3. DenseTNT [13] is able to achieve better prediction performance with MapTRv2-Centerline [23] compared to the GT map.

### 6. Conclusion

In this work, we propose a general vectorized map uncertainty formulation and extend multiple state-of-the-art online map estimation methods MapTR [22], MapTRv2 [23], and StreamMapNet [38] to additionally output uncertainty. We systematically analyze the resulting uncertainties and find that our approach captures many sources of uncertainty (occlusion, distance to camera, time of day, and weather). Finally, we combine these online map estimation models with state-of-the-art trajectory prediction approaches (DenseTNT [13] and HiVT [41]) and show that incorporating online mapping uncertainty significantly improves the performance and training characteristics of prediction models, by up to 50% and 15%, respectively. An exciting future research direction is leveraging these uncertainty outputs to measure the calibration of map models (similar to [16]). However, this is complicated by the need for fuzzy point set matching, a challenging problem itself.
Figure 7. A complicated intersection with many map elements. By leveraging uncertainty information, both combinations of map estimation and prediction models show enhancements in prediction, correctly predicting that the center vehicle will stay in its current lane.

Figure 8. The parking lot of a bus terminal, with many occlusions from stationary vehicles. By leveraging uncertainty information, both combinations reduce overshoot, minimizing endpoint error, and tightly cluster the predicted trajectories around the GT future.

Figure 9. A tunnel-like entrance to a building, with significant occlusions from trucks behind the center agent. By leveraging this uncertainty information, both HiVT and DenseTNT are able to produce sensible, on-road predictions, even with significant map uncertainty.
References


[29] Limeng Qiao, Wenjie Ding, Xi Qiu, and Chi Zhang. End-to-end vectorized HD-map construction with piecewise bezier curve. In *IEEE Conf. on Computer Vision and Pattern Recognition*, 2023. 2


[33] Wenwen Tong, Chonghao Sima, Tai Wang, Li Chen, Silei Wu, Hanming Deng, Yi Gu, Lewei Lu, Ping Luo, Dahua Lin, and Hongyang Li. Scene as occupancy. In *IEEE Int. Conf. on Computer Vision*, 2023. 2


[38] Tianyuan Yuan, Yicheng Liu, Yue Wang, Yilun Wang, and Hang Zhao. StreamMapNet: Streaming mapping network for vectorized online HD map construction. In *IEEE Winter Conf. on Applications of Computer Vision*, 2024. 2, 3, 4, 7, 1


Producing and Leveraging Online Map Uncertainty in Trajectory Prediction

Supplementary Material

A. Training Details

To account for possible differences in the rates of convergence for mapping and prediction models trained with and without uncertainty, each model’s hyperparameters are tuned separately to optimize performance. The best outcomes from these individually-tuned models are compared to show the effect of integrating uncertainty. Tab. 4 summarizes the core method hyperparameters.

In the probabilistic map estimation approaches for MapTR [22] and MapTRv2 [23], we alter the loss function from the initial $\ell_1$ loss to the Negative Log-Likelihood (NLL) of the Laplacian distribution. The Laplace output is added to all layers of MapTR’s Transformers. The two core reasons we chose a Laplace distribution are: it produced more accurate maps across models ($\sim$ 3-5% better AP) and was much more numerically stable during training (Fig. 14).

We also adjust the regressor’s loss weight from 5 to 0.03, compensating for the increased gradient norm resulting from the new loss function and difficulty in training. Given the use of a single GPU, we reduce the learning rate to 1.5E-4. Additionally, to avoid gradient explosion, we clip gradients to a maximum norm of 3. All other settings are retained as per the original configurations.

For StreamMapNet [38], we incorporate two distinct dataset splits: the original nuScenes [1] split and a newly proposed split. This new split is designed to address the overlapping scene challenges in the original training and validation splits [38]. To enhance StreamMapNet’s performance, we train using the new split to reduce overfitting risks and assess on the original nuScenes validation set. As shown in Tab. 5, modification impacts model uncertainty. This is admittedly a small improvement, but it comes for free along with the other benefits stated in the main body of the paper.

After modification, most of the map estimation models maintain their original performance. As shown in Tab. 5, MapTR and MapTRv2 produce 1% better AP when producing uncertainty. This is admittedly a small improvement, but it comes for free along with the other benefits stated in the main body of the paper.

For the HiVT [41] prediction model, we have increased the dropout rate to 0.2. All other hyperparameters are unchanged. For DenseTNT [13], the hyperparameters are tuned separately for each combination to yield the optimal results. The hyperparameters used for different methods are shown in Tab. 6.

For HiVT, we double the node dimension to account for uncertainty in both the $x$ and $y$ directions. For DenseTNT, the configurations of layer sizes and structures are maintained without significant alterations. The model utilizes a 128-dimensional vector to represent lane information, including details like vertices, intersection signals, traffic lights, etc. In our adaptation, we merely integrate uncertainty information into this raw feature vector.

**Distant Agents.** For both HiVT [41] and DenseTNT [13], there is no special treatment for agents that are beyond map perception range. This means that some far-away agents do not have agent-lane interactions to incorporate, making the model rely only on the agent’s past history, surrounding agent motion (agent-agent interactions remain unchanged), and any learned priors as a result of training with the absence of far-away map information.

B. Additional Visualizations

Fig. 10 visualizes the predictions of other agents when using multi-agent prediction models such as HiVT.

Fig. 11 shows another qualitative example of how occlusion impacts model uncertainty.

Fig. 12 shows that StreamMapNet [38] produces more...
uncertainty in rainy conditions, indicating potential difficulties in aggregating temporal information due to rain.

Fig. 13 shows that current models do not have any particular lack of confidence across different AV driving speeds.

**Calibration.** As seen in Fig. 14, MapTR’s lane type estimates are well-calibrated. Further, Fig. 14 shows that prediction methods like HiVT are robust to lane type miscalibration (evaluated by linearly interpolating between MapTR’s well-calibrated probabilities and a uniform distribution over type, and predicting trajectories with the resulting probabilities). One possible hypothesis is that HiVT focuses more on the presence of lanes rather than their types.
Figure 10. Multi-agent visualizations. Red indicates the GT and pink shows future agent predictions. In all three scenarios, our approach produces sensible predictions for both ego and non-ego agents.

Figure 11. A normal straight-driving scenario. Note that the parked cars on the rear right induce a larger uncertainty compared to the rear left, showing the effect of occlusions in online mapping uncertainty.

Figure 12. Different weather conditions such as rain can affect the confidence with which map estimation models predict certain elements, such as pedestrian crossings for certain models. Error bars show 95% confidence intervals.
Figure 13. For some scenarios, our uncertainty formulation captures the fact that uncertainty increases as the velocity of the AV increases. Error bars show 95% confidence intervals.

Figure 14. **Left:** MapTR’s lane type estimates are well-calibrated for divider, ped crossing and boundary. **Middle:** HiVT is robust to lane type miscalibration. **Right:** Laplace outputs are much more stable than Gaussian to train with.