Unveiling the Hidden: Online Vectorized HD Map Construction with Clip-Level Token Interaction and Propagation

Nayeon Kim[*] **Hongje Seong**[*] **Daehyun Ji** **Sujin Jang**[†]
Samsung Advanced Institute of Technology (SAIT)
`{nayeon.kim, hongje.seong, derek.ji, s.steve.jang}@samsung.com`

Abstract

Predicting and constructing road geometric information (*e.g.*, lane lines, road markers) is a crucial task for safe autonomous driving, while such static map elements can be repeatedly occluded by various dynamic objects on the road. Recent studies have shown significantly improved vectorized high-definition (HD) map construction performance, but there has been insufficient investigation of temporal information across adjacent input frames (*i.e.*, clips), which may lead to inconsistent and suboptimal prediction results. To tackle this, we introduce a novel paradigm of clip-level vectorized HD map construction, ***MapUnveiler***, which explicitly unveils the occluded map elements within a clip input by relating dense image representations with efficient clip tokens. Additionally, MapUnveiler associates inter-clip information through clip token propagation, effectively utilizing long-term temporal map information. MapUnveiler runs efficiently with the proposed clip-level pipeline by avoiding redundant computation with temporal stride while building a global map relationship. Our extensive experiments demonstrate that MapUnveiler achieves state-of-the-art performance on both the nuScenes and Argoverse2 benchmark datasets. We also showcase that MapUnveiler significantly outperforms state-of-the-art approaches in a challenging setting, achieving +10.7% mAP improvement in heavily occluded driving road scenes. The project page can be found at `https://mapunveiler.github.io`.

1 Introduction

Vectorized HD map construction (***VHC***) is a task of predicting instance-wise vectorized representations of map elements (*e.g.*, pedestrian crossings, lane dividers, road boundaries). Such static map elements are crucial information for self-driving vehicles, including applications like lane keeping [1, 6], path planning [29, 24, 15], and trajectory prediction [30, 40, 11]. Prior approaches to constructing dense and high-quality HD maps typically rely on SLAM-based offline methods (*e.g.*, [48, 37, 38]). Such an offline method generally involves a series of steps including feature extraction and selection (*e.g.*, edge, plane), odometry estimation via feature matching, and mapping. However, these processes involve complicated and computationally burdensome tasks, limiting their use to offline applications.

More recently, camera-based multi-view VHC has been actively investigated as a cost-efficient and real-time alternative to existing expensive offline approaches. Current works on camera-based VHC typically aim to extract unified 3D Bird's Eye View (***BEV***) features that cover the surrounding environment of the ego-vehicle [20, 22, 23], relying on various Perspective View (PV) to BEV transformation methods [50, 32, 21, 5]. Subsequently, a task-specific head follows to decode and

[*]Equal contribution.
[†]Corresponding author.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

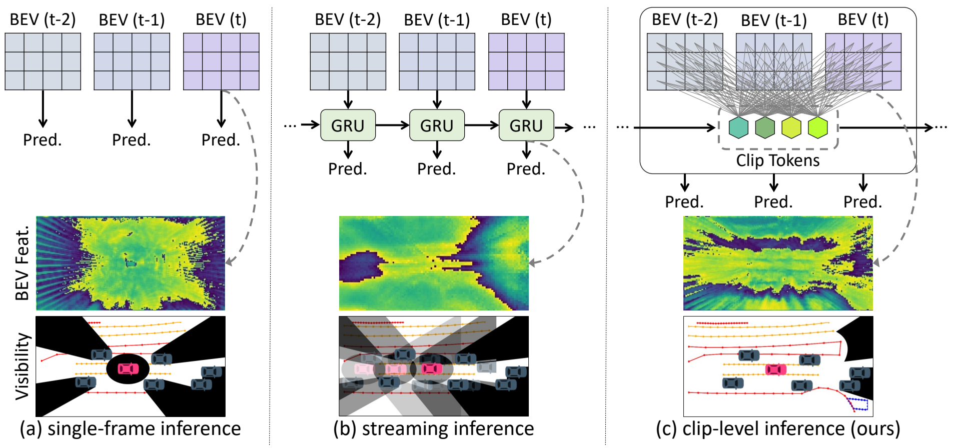

Figure 1: (a) Existing approaches relying on single-frame inference cannot capture the entire map information in the BEV features [26, 22, 23]. (b) Recent alternatives explore temporal information via streaming, but they cannot address the inherent nature of maps and propagate noise from previous timestamps. (c) We directly unveil hidden maps in BEV features by interacting with clip tokens that contain high-level map information. We visualize BEV features by 1D PCA projection. The BEV features are extracted from (a) MapTRv2 [23], (b) StreamMapNet [46], and (c) our MapUnveiler.

predict map elements from the extracted BEV features. Despite significant progress, prior works still suffer from frequently occluded map elements caused by dynamic foreground objects such as vehicles and pedestrians, as described in Fig. 1-(a). Moreover, the prediction performance degrades when applied to a larger perception range. To address such issues, prior works try to leverage temporal information extracted from a stream of preceding feature frames [46, 41]. While such approaches have achieved improved *online* prediction performance, they still do not fully leverage the potential of temporal information across a longer range of frames. In particular, prior works do not consider the cumulative impacts of occluded map elements, leading to noisy BEV feature generation, as shown in Fig. 1-(b) and Tab. 3. Moreover, such cumulative flaws can ultimately lead to degraded performance in longer perception ranges (see Tab. 1), which is crucial for safety-critical autonomous driving. On the other hand, the key idea of conventional SLAM-based offline methods is to stack and associate static features collected over a longer window of frames (*i.e.*, mapping features). This *offline* mapping strategy effectively addresses occlusion issues by leveraging diverse views of the map elements across multiple frames, while it requires manual human annotation and complex pipelines.

Based on these insights, we introduce a novel clip-level construction framework, ***MapUnveiler***, which incorporates the effective offline mapping strategy into state-of-the-art online VHC approaches. In contrast to the direct *global feature mapping* used in offline methods, our MapUnveiler collects *clip-level* temporal map information and learns differentiable associations for efficient online inference. We generate compact clip tokens consisting of temporal map information within a clip input and update BEV features with these tokens to unveil hidden map elements that are visible in certain frames, as shown in Fig. 1-(c). Subsequently, we transfer the inter-clip tokens to the next clip's BEV features to facilitate establishment the long-term intra-clip associations among map elements.

As a result, MapUnveiler outperforms state-of-the-art methods on two widely recognized benchmarks: +1.3% mAP on nuScenes [3] and +0.9% mAP on Argoverse2 [44]. We also observe a marginal increase in computational burden, as we exploit the benefits of the clip-level inference strategy, which allows an efficient inference of multi-frame inputs. In summary, our contributions include:

- We propose MapUnveiler, an online VHC model that incorporates the offline mapping strategy and unveils the hidden maps by interacting multiple dense BEV features with compact tokens.
- We introduce the *clip-level* pipeline to infer MapUnveiler online by mapping within a clip set of BEV features and propagating the map information to subsequent timestamps, thereby building a global map efficiently.
- Our method significantly improves a frame-level model on longer perception range settings (58.6% $\rightarrow$ 68.7%) and heavy occlusion splits (53.1% $\rightarrow$ 63.8%), and achieves state-of-the-art performance on two standard VHC benchmarks with marginal extra computations (15.6 FPS $\rightarrow$ 12.7 FPS).

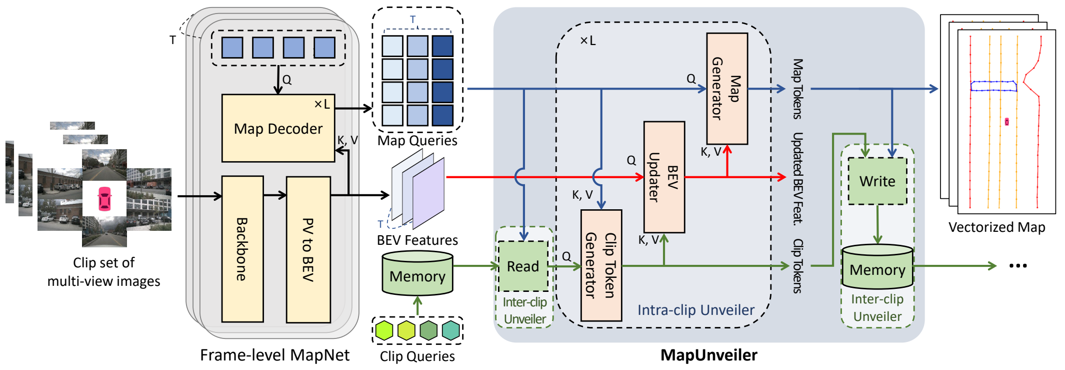

Figure 2: Our framework takes clip-level multi-view images and outputs clip-level vectorized HD maps. All components in the frame-level MapNet (*i.e.*, Backbone, PV to BEV, Map Decoder) are adopted from MapTRv2 [23]. The frame-level MapNet extracts map queries and BEV features independently at each frame. MapUnveiler generates compact clip tokens that contain clip-level temporal map information and directly interact with dense BEV features. With the updated BEV features, we construct high-quality clip-level vectorized maps. The generated map tokens and clip tokens are then written to memory.

2 Related Work

Multi-View HD Map Construction. SLAM (Simultaneous Localization and Mapping) [10] has been a central technique for constructing accurate HD maps [48, 37, 38]. However, these methods require memory-intensive, complex pipelines for global mapping of geometric features, and are therefore typically executed offline. Recently, deep learning approaches have emerged as an appealing alternative to those expensive offline methods, enabling online HD map construction using cost-efficient multi-view camera sensors. The perspective-view (PV) to bird's-eye-view (BEV) transformation methods [50, 32, 21, 5] enable the generation of 3D features from the surrounding environment of the ego-vehicles using camera sensors, even in the absence of precise spatial cues. BEVFormer [21] utilizes the deformable attention mechanism [52] to extract BEV features and predict rasterized semantic maps. However, it cannot generate instance-wise representation of map elements. To address this, HDMapNet [20] introduces a heuristic method to group pixel-level semantic maps into a vectorized representation. Similarly, VectorMapNet [26] proposes an end-to-end learning approach to predicting vectorized map representations. Although such methods have demonstrated notable prediction performance in single-frame inference, they do not consider the temporal information from multi-frame inputs. More recently, StreamMapNet [46] and SQD-MapNet [41] have proposed a streaming feature paradigm [42], which aims to leverage temporal information for improved temporal consistency across predictions. However, these methods propagate dense BEV features directly, incorporating map information from previous frames that may have been occluded and undetected, resulting in the accumulation of noise, as shown in Fig. 1-(b). To address this issue, we propose an end-to-end clip token learning approach that combines offline mapping techniques with online strategies, aiming for high performance and computational efficiency.

Temporal Token Learning. With the rapid development of transformers [39], there has been significant interest in efficient token learning alongside dense features, *e.g.*, CNN representations. In particular, temporal token learning has emerged as an attractive alternative to memory-intensive spatio-temporal dense CNN representations [45, 31]. VisTR [43] extends DETR [4] into the 3D domain to extract spatio-temporal instance tokens that can be directly used for instance segmentation. IFC [17] proposes an efficient spatio-temporal token communication method, which replaces the heavy interactions within dense CNN features. VITA [13] learns efficient video tokens from frame-level instance tokens without dense CNN features. Cutie [7] updates CNN representations with tokens to avoid spatio-temporal dense matching. TTM [36] introduces an efficient long-term memory mechanism by summarizing tokens into memory rather than stacking [2, 18, 34] or recurrence [14, 8]. While all the aforementioned approaches were designed to handle foreground instances, we discover the potential of token learning to construct background maps. By learning compact tokens and interacting with dense BEV features, we impose traditional mapping into online VHC model and enable online running.

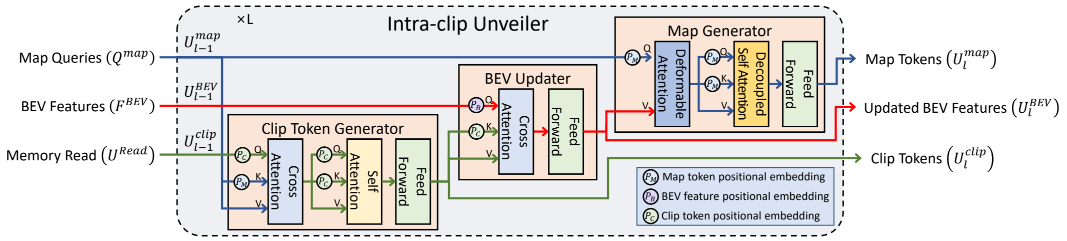

Figure 3: A detailed implementation of Intra-clip Unveiler. We use blue, red, and green arrows to indicate the flows of map tokens, BEV features, and clip tokens, respectively. In each attention and feed forward layer, standard layer normalization, dropout, and residual connections are followed.

3 Method

3.1 Overview

We present the overall architecture of MapUnveiler in Fig. 2. Given a set of synchronized multi-view images (*i.e.*, clip inputs), our model sequentially construct clip-level vectorized HD maps. We first extract frame-level BEV features and map queries, which are then used as inputs to MapUnveiler module. We employ memory tokens, which are written from the previous clip and facilitate the establishment of long-term temporal relationships. From the memory tokens and map queries, we generate clip tokens that embed temporal map element cues in a compact feature space. This is the first step to understand clip-level map information. We then update BEV features with clip tokens, which is the core step of unveiling hidden maps. Using the updated (unveiled) BEV features, we extract map tokens and construct clip-level vectorized HD maps. After a clip-level inference, we generate new memory tokens using clip tokens, map tokens, and the current memory tokens. The new memory tokens are used for providing temporal cues for the subsequent clip-level inference. Since we opt for a clip-level pipeline, MapUnveiler efficiently infers with a temporal stride S, performing clip-level inference only N_T/S times for a sequence of N_T frames. In the following subsections, we detail each module in the proposed framework.

3.2 Frame-level MapNet

We adopt MapTRv2 [23] as our frame-level MapNet architecture to extract a clip set of BEV features and map queries from synchronized multi-view images. We extract perspective view (PV) image features using a backbone network, then these PV image features are transformed into BEV features through a PV-to-BEV module. Following the setup of MapTRv2, we adopt ResNet50 [12] and Lift, splat, shoot (LSS) [32]-based BEV feature pooling [16] for our backbone and PV-to-BEV module, respectively. These BEV features are utilized for querying maps in the map decoder. With their BEV features, the map decoder outputs frame-level map queries which can be directly used for constructing vectorized HD maps. Finally, the frame-level MapNet outputs BEV features and map queries, which are the results from the PV-to-BEV module and the map decoder, respectively. BEV features represent rasterized map features, whereas map queries embed vectorized map information; thus, we can directly construct vectorized HD maps using the map queries.

3.3 MapUnveiler

MapUnveiler is a novel framework designed to unveil invisible map information that cannot be captured by frame-level BEV features alone. To avoid heavy computations, we adopted a clip-level inference scheme with temporal window (clip length) T and stride S. A detailed explanation of the inference scheme with the temporal window T and stride S is provided in Appendix (see Sec. A.1). Our MapUnveiler consists of two main components: (1) Intra-clip Unveiler and (2) Inter-clip Unveiler. For each clip-level pipeline, our Intra-clip Unveiler generates vectorized maps for T frames. The Inter-clip Unveiler then writes memory tokens with the tokens generated in Intra-clip Unveiler to build global relationships.

3.3.1 Intra-clip Unveiler

Intra-clip Unveiler is composed of a sequence of L layers. It initially takes a clip set of frame-level map queries Q^{map}, BEV features F^{BEV}, and memory read U^{Read} (read at Inter-clip Unveiler, detailed in Sec. 3.3.2). In the first step, compact clip tokens are created by the clip token generator. The BEV updater then unveils hidden maps in the BEV features with the clip tokens. Finally, map generator outputs clip-level map tokens from the updated BEV features. The map tokens are directly used for constructing vectorized HD maps with perception heads. We illustrate the Intra-clip Unveiler in Fig. 3. In the followings, we describe the detailed implementation of each module.

Clip Token Generator. Clip token generator yields clip tokens $U_l^{clip} \in \mathbb{R}^{N_c \times C}$ from frame-level map queries $Q^{map} \in \mathbb{R}^{T \times N_i \times N_p \times C}$, where N_c, N_i, and N_p denote the clip token size, number of predicted map element, and number of points per map element, respectively. To globally gather intra-clip map features, we opt for a naive cross-attention [39]. Through this step, we obtain compact clip-level map representations, enabling efficient intra-clip communication with small-scale features.

BEV Updater. The second step is the BEV Updater, which updates bev features $F^{BEV} \in \mathbb{R}^{T \times H \times W \times C}$ with the clip tokens U_l^{clip} to unveil the hidden map element information. In cross-attention, query is derived from the bev features F^{BEV}, and the key and value are derived from the clip token U_l^{clip}. The output of this step is robustly updated bev features $U_l^{BEV} \in \mathbb{R}^{T \times H \times W \times C}$ enhanced via clip tokens for hidden areas relative to the original bev features. The main idea of BEV Updater is to avoid heavy computation in spatio-temporal cross attention. To achieve this, we do not directly communicate intra-clip BEV features, but instead decouple the spatial BEV features and the temporal clip tokens. We then update the spatial BEV features with compact temporal clip tokens, effectively communicating spatio-temporal information with reasonable computational costs. The updated bev features U_l^{BEV} are used as value features in the next step.

Map Generator. The last step is the Map Generator, which generates map tokens $U_l^{map} \in \mathbb{R}^{T \times N_i \times N_p \times C}$ using the updated BEV features U_l^{BEV} created in the previous step. The objective of this step is to generate a refined version of frame-level map queries. As illustrated in Fig. 3, the map generator uses deformable attention [52] and decoupled self-attention [22] mechanisms, following [22]. In deformable attention, query is derived from the map queries Q^{map}, and the value is derived from the updated BEV features U_l^{BEV}. Since the updated BEV features are spatio-temporally communicated, we directly extract map tokens. Each map token represents a vectorized map element through a 2-layer Multi-Layer Perceptron (MLP). The map tokens U_l^{map} are written to the memory of the Inter-clip Unveiler, and when the map tokens U_l^{map} of the L-th layer pass through the prediction head, vectorized maps are generated.

3.3.2 Inter-clip Unveiler

Inter-clip Unveiler propagates the tokens from previous clip input to the next one, thereby preserving the dense temporal information from the prior frames. As shown in Fig. 2, Inter-clip Unveiler writes map tokens U_l^{map} and clip tokens U_l^{clip} from the Intra-clip Unveiler to the memory. Here, we adopt token turning machine (TTM) [36] to efficiently manage the long-term map information. In the followings, we describe the detailed implementation of read and write.

Read. We generate compact tokens that contain a global map information by reading from memory tokens and map queries. Following TTM [36], we read with the token summarizer [35] which efficiently selects informative tokens from inputs as follows:

$$U^{read} = Read(U^{memory}_{t-2S:t-S}, Q^{map}) = S_{N_c}([U^{memory}_{t-2S:t-S} || Q^{map}]), \tag{1}$$

where $[U^{memory}_{t-2S:t-S} || Q^{map}]$ denotes the concatenation of two elements, and $U^{memory}_{t-2S:t-S}$ denotes memory tokens for a clip. We employ the location-based memory addressing used in [36] utilizing the positional embedding (detailed in Section 3.3.3). Note that the memory is not available in the first clip-level pipeline. Therefore, we initially write the memory token from learnable clip embeddings.

Write. We employ the write operation with the same token summarizer [35] that is used in [36]. The new memory $U^{memory}_{t-S:t} \in \mathbb{R}^{M \times C}$ is generated by summarizing the clip tokens U_L^{clip}, map tokens U_L^{map}, and old memory $U_{t-2S:t-S}^{memory}$ as follows:

$$U_{t-S:t}^{memory} = Write(U_L^{clip}, U_L^{map}, U_{t-2S:t-S}^{memory}) = S_M([U_L^{clip} || U_L^{map} || U_{t-2S:t-S}^{memory}]), \tag{2}$$

where M denotes the memory token size. The newly generated memory through the write operation is used in the read operation of the first layer of the Intra-clip Unveiler in the subsequential per-clip process. If the tokens within the memory are not re-selected in the subsequent steps, it will be removed from the memory, and the selection mechanism will be determined through the learning. We employ the same memory addressing method used in the read operation. The write operation is applied in the last layer of the Intra-clip Unveiler, generating new memory that preserves the information of the clip tokens and map tokens.

3.3.3 Positional Embedding

While the standard transformer structure is permutation-invariant, we require position information added with temporal information to predict map elements at the clip-level. For the BEV features (P_B), we use a fixed 3D sinusoidal positional embedding, following [43]. For the map tokens (P_M), we use learnable positional embeddings used in frame-level MapNet [23] with newly defined learnable temporal positional embeddings. For the clip tokens (P_C), we define new learnable positional embeddings. Similarly, learnable positional embeddings are defined for the memory tokens that are used in read and write of the Inter-clip Unveiler.

3.3.4 Loss

Since our model is built on top of the frame-level MapNet (MapTRv2 [23]), we basically follow the loss functions used in MapTR [22] and MapTRv2 [23]. Specifically, we employ the overall loss functions as

$$\mathcal{L}_{MapUnveiler} = \lambda_c^M \mathcal{L}_{cls}^M + \lambda_p^M \mathcal{L}_{p2p}^M + \lambda_d^M \mathcal{L}_{dir}^M + \lambda_s^M \mathcal{L}_{PVSeg}^M, \tag{3}$$

$$\mathcal{L}_{Frame\_MapNet} = \mathcal{L}_{one2one} + \mathcal{L}_{one2many} + \mathcal{L}_{dense}, \tag{4}$$

$$\mathcal{L}_{one2one} = \lambda_c^F \mathcal{L}_{cls}^F + \lambda_p^F \mathcal{L}_{p2p}^F + \lambda_d^F \mathcal{L}_{dir}^F, \tag{5}$$

$$\mathcal{L}_{dense} = \lambda_t^F \mathcal{L}_{depth} + \lambda_b^F \mathcal{L}_{BEVSeg} + \lambda_s^F \mathcal{L}_{PVSeg}, \tag{6}$$

where $\mathcal{L}_{MapUnveiler}$ and $\mathcal{L}_{Frame\_MapNet}$ indicate the loss functions for training MapUnveiler and frame-level mapnet, respectively. $\mathcal{L}_{cls}$, $\mathcal{L}_{p2p}$, $\mathcal{L}_{dir}$, and $\mathcal{L}_{PVSeg}$ denote classification loss [22], point-to-point loss [22], edge direction loss [22], and PV segmentation loss [23], respectively. $\mathcal{L}_{one2one}$, $\mathcal{L}_{one2many}$, and $\mathcal{L}_{dense}$ are used for training frame-level MapNet and denote one-to-one loss [22], one-to-many loss [23], and auxiliary dense prediction loss [23], respectively. L_{depth} and $\mathcal{L}_{BEVSeg}$ denote depth prediction loss [23] and BEV segmentation loss [23], respectively. We set hyperparameters to $\lambda_c^M = 2$, $\lambda_p^M = 5$, $\lambda_d^M = 0.005$, $\lambda_s^M = 2$, $\lambda_c^F = 2$, $\lambda_p^F = 5$, $\lambda_d^F = 0.005$, $\lambda_t^F = 3$, $\lambda_b^F = 1$, and $\lambda_s^F = 2$. Note that we did not use one2many loss [23] to save GPU memory during the main training.

4 Experiments

4.1 Dataset and Metric

We construct experiments on two standard VHC benchmarks: nuScenes [3] and Argoverse2 [44] datasets. nuScenes offers synchronized six-view images with high-quality HD maps. It provides 1,000 scenes and each scene consists of about 20 seconds. In this dataset, we follow the official scene split of 700, 150, and 150 for training, validation, and testing, respectively. Similarly Argoverse2 provides synchronized seven-view images with high-quality HD maps. It contains 1,000 scenarios and each scenario consists of about 15 seconds. Argoverse2 dataset provides the official scene split of 700, 150, and 150 for training, validation, and testing, respectively, which we follow.

For each dataset, we construct maps in two perception ranges: a standard range of 60×30m and a longer range of 100×50m. Additionally, we create challenging validation splits to demonstrate the efficacy of our model under heavy occlusions. Specifically, we collect the occluded frames if any dynamic objects exists within 2.5m around the ego vehicle. Thankfully, nuScenes provides 3D cuboid annotations for vehicles and pedestrians, allowing us to automatically create the new challenging split, and the result with this split is given in Sec. 4.4.

Table 1: Comparisons on nuScenes and Argoverse2 `val` sets. AP$_p$, AP$_d$, AP$_b$ indicate the average precision for pedestrian crossing, divider, and boundary, respectively. FPS is measured using a single NVIDIA A100 GPU. * are taken from the corresponding papers and are scaled based on the FPS of MapTRv2 [23] for a fair comparison.

Range	Method	nuScenes						Argoverse2				
		Epoch	AP$_p$	AP$_d$	AP$_b$	mAP	FPS	Epoch	AP$_p$	AP$_d$	AP$_b$	mAP
$60 \times 30\ m$	MapTR[ICLR'23] [22]	24	46.3	51.5	53.1	50.3	16.7*	6	54.7	58.1	56.7	56.5
	MapVR[NeurIPS'23] [47]	24	47.7	54.4	51.4	51.2	16.7*	-	54.6	60.0	58.0	57.5
	GeMap[ECCV'24] [49]	24	49.2	53.6	54.8	52.6	13.2*	-	-	-	-	-
	PivotNet[ICCV'23] [9]	24	56.2	56.5	60.1	57.6	11.1*	-	-	-	-	-
	BeMapNet[CVPR'23] [33]	30	57.7	62.3	59.4	59.8	9.7*	-	-	-	-	-
	MapTRv2[IJCV'24] [23]	24	59.8	62.4	62.4	61.5	15.6	6	62.9	72.1	67.1	67.4
	StreamMapNet[WACV'24] [46]	24	-	-	-	62.9	12.5*	6	62.0	59.5	63.0	61.5
	SQD-MapNet[preprint] [41]	24	63.0	62.5	63.3	63.9	-	6	64.9	60.2	64.9	63.3
	MGMap[CVPR'24] [25]	24	61.8	65.0	67.5	64.8	12.3*	-	-	-	-	-
	MapQR[ECCV'24] [27]	24	63.4	68.0	67.7	66.4	14.2*	6	64.3	72.3	68.1	68.2
	HIMap[CVPR'24] [51]	30	62.6	**68.4**	**69.1**	66.7	9.7*	6	**69.0**	69.5	**70.3**	69.6
	MapUnveiler (ours)	24	**67.6**	67.6	68.8	**68.0**	12.7	6	68.9	**73.7**	68.9	**70.5**
$100 \times 50\ m$	MapTR[ICLR'23] [22]	24	45.5	47.1	43.9	45.5	16.7*	6	-	-	-	47.5
	MapTRv2[IJCV'24] [23]	24	58.1	61.0	56.6	58.6	15.6	6	66.2	61.4	54.1	60.6
	StreamMapNet[WACV'24] [46]	24	62.9	63.1	55.8	60.6	12.5*	6	-	-	-	57.7
	SQD-MapNet[preprint] [41]	24	67.0	65.5	59.5	64.0	-	6	66.9	54.9	56.1	59.3
	MapUnveiler (ours)	24	**68.0**	**70.0**	**68.2**	**68.7**	12.7	6	**69.7**	**67.1**	**59.3**	**65.4**

For fair comparisons with state-of-the-art VHC methods, we follow the standard metric [20, 26] of average precision (AP) under several Chamfer thresholds $\{0.5m, 1.0m, 1.5m\}$. We report the average AP in three Chamfer thresholds for each semantic map categories of pedestrian crossing, divider, and boundary. We average AP at three Chamfer thresholds and report the results for each semantic map category: pedestrian crossing, divider, and boundary. To validate the scalability in map categories, we further examine our model by learning semantic centerline, and the results are given in Appendix (see Sec. A.2).

4.2 Implementation Details

Our model is trained with 8 NVIDIA A100 GPUs using batch size of 16. We train MapUnveiler using AdamW [28] with a learning rate of 6×10^{-4} and a weight decay of 0.01. We follow the standard setup [22, 23] that training models for the total epochs of 24 with an image resolution of 800×450 and the total epochs of 6 with an image resolution of 614×614 on nuScenes and Argoverse2 datasets, respectively. The number of the embedding dimension C, memory tokens M, and clip tokens N_c, map elements N_i, points per map element N_p are set to 256, 96, 50, 50, and 20 respectively. We use temporal window size T and stride S of 3 and 2, respectively. The spatial size of the BEV feature is 100×200. We pre-train the frame-level MapNet on the standard setting and then main train our model to facilitate the exploitation of the frame-level map information. For fair comparisons, MapUnveiler is pre-trained for 12 and 3 epochs, and then main trained for an additional 12 and 3 epochs, resulting in a total of 24 and 6 epochs on nuScenes and Argoverse2 datasets, respectively. During main training, we perform clip-level inference three times and compute losses after each clip-level inference to effectively handle the GPU memory overhead.

4.3 Comparisons

We compare our MapUnveiler against state-of-the-art VHC methods trained with standard settings, which use ResNet50 [12] as a backbone, multi-camera modality, and train for 24 epochs and 6 epochs on nuScenes [3] and Argoverse2 [44] respectively. As shown in Tab. 1, MapUnveiler achieves state-of-the-art performance on all validation sets. Specifically, we surpass the state-of-the-art temporal model (SQD-MapNet [41]) by mAP of 4.1% and 4.7% on two range settings of nuScenes validation set and 7.2% and 6.1% on Argoverse2. We also outperform a heavy VHC model (HiMap [51]) by 1.3% and 0.9% on two benchmark sets. Notably, we boost frame-level model (MapTRv2 [23]) by 6.5% and 10.1% on nuScenes and 3.1% and 4.8% on Argoverse2. The superiority of our approach is particularly evident in the long-range setting: mAP on $60\times30m \rightarrow 100\times50m$ settings are 61.5%$\rightarrow$58.6% (-2.9%)

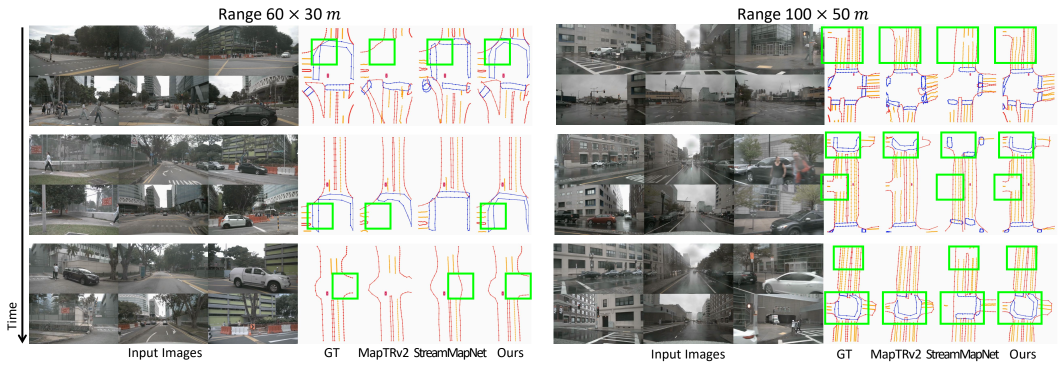

Figure 4: Qualitative comparisons on two range variants of nuScenes `val` set: $60\times30m$ and $100\times50m$. We compare our MapUnveiler with MapTRv2 [23] and StreamMapNet [46]. We marked significant improvements from MapTRv2 and StreamMapNet using green boxes.

Table 2: Experimental results of the long training schedules.

Range	nuScenes					Argoverse2				
	Epoch	AP_p	AP_d	AP_b	mAP	Epoch	AP_p	AP_d	AP_b	mAP
$60 \times 30\ m$	24	67.6	67.6	68.8	68.0	6	68.9	73.7	68.9	70.5
	48	69.5	69.4	70.5	69.8	12	69.0	74.9	69.1	71.0
	110	71.0	69.1	71.8	70.6	30	72.5	74.2	71.9	72.9
$100 \times 50\ m$	24	68.0	70.0	68.2	68.7	6	69.7	67.1	59.3	65.4
	48	68.4	71.2	68.3	69.3	12	70.4	66.8	59.3	65.5
	110	71.2	71.7	72.2	71.7	30	71.7	67.9	62.6	67.4

and 68.0%→68.7% (+0.7%) for MapTRv2 [23] and ours. Although MapUnveiler incorporates temporal modules, we achieve a reasonable inference speed (12.7 FPS) compared to frame-level MapNet (MapTRv2 [23], 15.6 FPS), surpassing both performance and speed compared to the state-of-the-art (HiMap [51], 9.7 FPS).

Qualitative Comparison. Fig. 4 shows qualitative comparisons on two perception range settings on nuScenes benchmark. We compare our MapUnveiler with the state-of-the-art frame-level model (MapTRv2 [23]) and temporal model (StreamMapNet [46]). As shown in the figure, MapTRv2 often mis-detects complex pedestrian crossings and cannot precisely predict dividers occluded by vehicles on the road. StreamMapNet struggles to accurately predict the boundaries of intersections, despite leveraging temporal information. In contrast, MapUnveiler consistently delivers accurate results for all map categories in most cases. We provide more qualitative results on various scenes in Appendix and full-frame results through the supplementary video.

4.4 Analysis Experiments

In this study, we present extensive experimental results on nuScenes benchmark and provide analysis.

Long Training Schedules. Tab. 2 demonstrates the performance improvement achieved through longer training schedules. We obtained (24, 48, 110) and (6, 12, 30) epoch models by pre-training for (12, 24, 24) and (3, 6, 6) epochs, and then main training for (12, 24, 86) and (3, 6, 24) epochs on nuScenes and Argoverse2 datasets, respectively. As given in the table, the performance consistently improves with longer training, although the gains are relatively marginal. We conjecture that our MapUnveiler converges rapidly because we train the temporal modules from a pre-trained frame-level MapNet. Since our model shows no significant gain from the 110 epoch model on nuScenes, we opt for 48 epochs for the remaining analysis experiments.

Robustness to Occlusion. Tab. 3 presents results on our challenging validation splits collected based on the nearest dynamic objects (detailed in Sec. 4.1). For comparisons, we evaluate MapTRv2 and StreamMapNet using the code and pre-trained weight provided in each official repository. As shown in the table, MapUnveiler surpasses MapTRv2 and StreamMapNet in all evaluated metrics. In particular, existing approaches degrade performance significantly compared to the results on the standard

Table 3: Experimental results under heavy occlusions.

Method	AP$_p$	AP$_d$	AP$_b$	mAP
MapTRv2 [23]	24.1	62.2	55.2	47.1
StreamMapNet [46]	32.1	59.4	67.9	53.1
MapUnveiler (ours)	**38.6**	**80.6**	**72.2**	**63.8**

Table 4: Ablation study on the proposed modules.

Method	AP$_p$	AP$_d$	AP$_b$	mAP
MapTRv2 [23]	58.8	61.8	62.8	61.2
+ Intra-clip Unveiler	65.6	67.6	68.0	67.1
+ Inter-clip Unveiler	**69.5**	**69.4**	**70.5**	**69.8**

Table 5: Speed analysis of the proposed modules.

Method	FPS	GPU (MB)	Params (MB)
MapTRv2 [23]	15.6	830.4	76.4
+ Intra-clip Unveiler	13.1	1552.5	144.0
+ Inter-clip Unveiler	12.7	1614.9	213.9

Table 6: Results with freezing frame-level MapNet.

Read	AP$_p$	AP$_d$	AP$_b$	mAP
Freeze	64.9	66.5	68.7	66.7
End-to-End	**69.5**	**69.4**	**70.5**	**69.8**

Table 7: Input variants in read.

Read	AP$_p$	AP$_d$	AP$_b$	mAP
Memory	68.3	68.5	70.0	68.9
Memory + Q^{map}	**69.5**	**69.4**	**70.5**	**69.8**

Table 8: Input variants in write.

Write	AP$_p$	AP$_d$	AP$_b$	mAP
None	65.6	67.6	68.0	67.1
U_L^{clip}	68.7	68.9	69.6	69.1
$U_L^{clip} + U_L^{map}$	**69.5**	**69.4**	**70.5**	**69.8**
$U_L^{clip} + U_L^{map} + U_L^{BEV}$	68.5	68.8	69.5	68.9

Table 9: Temporal window size T and stride S.

T	S	AP$_p$	AP$_d$	AP$_b$	mAP
1	1	63.5	66.0	66.1	65.2
3	1	69.0	69.4	69.4	69.3
3	2	69.5	69.4	**70.5**	69.8
3	3	68.6	68.7	69.6	68.9
5	3	**70.4**	**69.5**	**70.5**	**70.1**

Table 10: Variant memory token sizes.

M	AP$_p$	AP$_d$	AP$_b$	mAP
24	68.2	68.3	69.1	68.5
48	68.4	69.3	69.9	69.2
96	69.5	**69.4**	**70.5**	**69.8**
192	69.0	**69.4**	70.1	69.5
384	**70.0**	69.3	69.9	69.7

Table 11: Variant clip token sizes

N_c	AP$_p$	AP$_d$	AP$_b$	mAP
25	67.8	69.6	70.1	69.2
50	**69.5**	**69.4**	**70.5**	**69.8**
100	67.9	68.1	69.6	68.5
200	69.3	68.8	70.2	69.4

split, *i.e.*, 61.5%→47.1% (-14.4%) in MapTRv2 and 62.9%→53.1% (-9.8%) in StreamMapNet, MapUnveiler also shows a performance degradation of 69.8%→63.8% (-6.0%), but it demonstrates a smaller performance gap compared to previous studies.

Effectiveness of The Proposed Modules. Tabs. 4 and 5 show an ablation study and speed analysis of our two proposed modules: Intra-clip Unveiler and Inter-clip Unveiler. We started with a frame-level MapNet, MapTRv2 [23], obtaining the result by reproducing it on our system. As shown in the result, Intra-clip Unveiler significantly boosts the performance from 61.2% to 67.1% (+5.9%) by mapping within a clip, and it rarely requires extra computation (15.6 FPS → 13.1 FPS). Inter-clip Unveiler further improves performance from 67.1% to 69.8% (+2.7%) by building global relationship of the mapping. However, our approach requires approximately two times more peak GPU memory and three times more parameters compared to the frame-level MapTRv2 model during inference. This could be considered a potential limitation of our method, but the amounts are not large.

Frozen Frame-level MapNet. Tab. 6 presents the results of MapUnveiler where frame-level MapNet is frozen and only Intra-clip Unveiler and Inter-clip Unveiler are trained. Despite the frozen model consuming the same amount of computation during inference, it effectively reduces GPU memory consumption and converges quickly during training. As given in the table, we obtained slightly lower performance compared to the end-to-end training setting. Interestingly, however, the frozen model also achieves the state-of-the-art performance on nuScenes benchmark (current SOTA performance is mAP of 66.7%, the concurrent work of HIMap [51]). This indicates that our approach effectively unveils the hidden maps in the frame-level representations and does not rely solely on strong BEV representation learning.

Ablation Study on Read and Write. As given in Tab. 7, reading only from memory is not enough, and feeding map tokens facilitates the understanding of the current map information (68.9%→69.8%).

We also ablate the input in write, and the result is given in Tab. 8. We significantly improve the performance by writing clip tokens to the memory (67.1%→69.1%). This suggests that the propagation of memory tokens facilitates constructing the global map. By additionally writing map tokens, we further boost the performance by explicitly providing the current map information (69.1%→69.8%). Unfortunately, however, we cannot obtain performance gain by writing dense BEV feature additionally. We conjecture that memorizing different types of features, *i.e.*, vectorized representation of tokens and rasterized representation of dense features, may disturb training and tend to converge to a sub-optimal state.

Temporal Window Size T and Stride S. We train MapUnveiler with various temporal window size T and stride S. As shown in Tab. 9, increasing the temporal window leads to performance improvement by interacting with more frames in a clip. However, training with a $T = 5$ setting requires $>$ 40GB of GPU memory, limiting the GPU models. If we train MapUnveiler with $T = 3$, it consumes $<$ 32GB of GPU memory, making it possible to train with various GPU models, and it achieves comparable performance. Additionally, selecting either too short ($S = 1$) or too long ($S = 3$) a temporal stride yields sub-optimal results. Therefore, we opt for $T = 3$ and $S = 2$ as the default setting.

Memory Token Size M. Tab. 10 presents the experimental results with various memory token sizes. Selecting a memory token size of 96 or above does not significantly change the performance, whereas choosing a size smaller than 96 results in considerable performance degradation. This indicates that MapUnveiler does not require a large memory token size to store the 50 clip tokens and 50×20 map tokens, demonstrating that it performs memory-efficiently.

Clip Token Size N_c. Tab. 11 gives the results with various clip token sizes. MapUnveiler achieves state-of-the-art performance of 69.2% using only 25 clip tokens. We further boost the performance by using a larger clip token size of 50. However, employing clip token sizes beyond 50 disrupts learning and leads to a performance degradation.

More Analysis in Appendix. We provide additional experimental results including centerline, 3D map construction, geo-disjoint split, various backbones, and limitation analysis in Appendix.

4.5 Limitation

MapUnveiler takes temporally consecutive frames as input. Therefore, if an intermediate frame is not correctly inputted due to communication errors in real-world scenarios, the unveiling within the clip may not be performed properly. We expect that the performance will recover quickly from the subsequent unveiling pipeline. Additionally, if the model cannot see a clear region across all frames as shown in Fig. 5, MapUnveiler may fail to recognize the occluded map information.

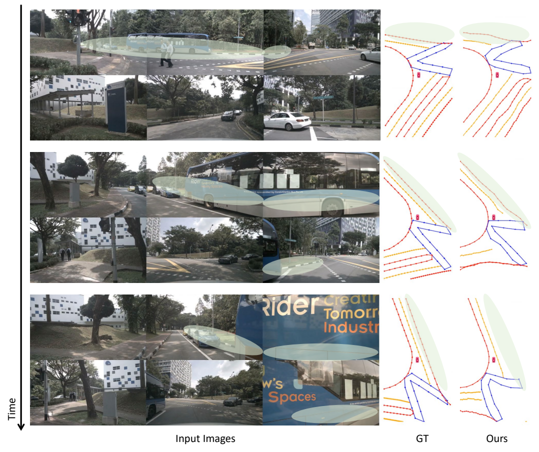

Figure 5: A limitation under heavy occlusions. MapUnveiler initially roughly predicts the boundary where we highlighted in green. However, the invisible regions are continuous, and MapUnveiler eventually predicts the region as having no boundary.

5 Conclusion

In this paper, we present a new VHC paradigm that constructs clip-level maps to efficiently incorporate conventional offline mapping strategies. In contrast to the recent temporal VHC models that do not consider the cumulative impacts of occluded map elements and directly propagate the noisy BEV features, we unveil the hidden map and noise in BEV features by interacting with compact clip tokens. To establish global mapping efficiently, we propagate the clip tokens instead of dense BEV features. With these two advanced modules, we propose MapUnveiler, which significantly outperforms previous works in challenging scenarios such as long-range perception and heavy occlusions. Since we introduce a novel insight into online VHC approaches to incorporate mapping strategy efficiently, we hope that our research motivates follow-up studies to delve deeper into establishing global mapping online and leads to practical VHC solutions.

References

[1] Angelos Amditis, Matthaios Bimpas, George Thomaidis, Manolis Tsogas, Mariana Netto, Saïd Mammar, Achim Beutner, Nikolaus Möhler, Tom Wirthgen, Stephan Zipser, et al. A situation-adaptive lane-keeping support system: Overview of the safelane approach. *IEEE Transactions on Intelligent Transportation Systems*, 11(3):617–629, 2010. 1

[2] Mikhail S Burtsev, Yuri Kuratov, Anton Peganov, and Grigory V Sapunov. Memory transformer. *arXiv preprint arXiv:2006.11527*, 2020. 3

[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. In *CVPR*, pages 11621–11631, 2020. 2, 6, 7

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229. Springer, 2020. 3

[5] Shaoyu Chen, Tianheng Cheng, Xinggang Wang, Wenming Meng, Qian Zhang, and Wenyu Liu. Efficient and robust 2d-to-bev representation learning via geometry-guided kernel transformer. *arXiv preprint arXiv:2206.04584*, 2022. 1, 3

[6] Zhilu Chen and Xinming Huang. End-to-end learning for lane keeping of self-driving cars. In *IV*, pages 1856–1860. IEEE, 2017. 1

[7] Ho Kei Cheng, Seoung Wug Oh, Brian Price, Joon-Young Lee, and Alexander Schwing. Putting the object back into video object segmentation. *arXiv preprint arXiv:2310.12982*, 2023. 3

[8] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. In *NeurIPS Workshops*, 2014. 3

[9] Wenjie Ding, Limeng Qiao, Xi Qiu, and Chi Zhang. Pivotnet: Vectorized pivot learning for end-to-end hd map construction. In *ICCV*, pages 3672–3682, 2023. 7

[10] Hugh Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006. 3

[11] Xunjiang Gu, Guanyu Song, Igor Gilitschenski, Marco Pavone, and Boris Ivanovic. Producing and leveraging online map uncertainty in trajectory prediction. *arXiv preprint arXiv:2403.16439*, 2024. 1

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 4, 7, 14

[13] Miran Heo, Sukjun Hwang, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. Vita: Video instance segmentation via object token association. In *NeurIPS*, volume 35, pages 23109–23120, 2022. 3

[14] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 3

[15] Yihan Hu, Jiazhi Yang, Li Chen, Keyu Li, Chonghao Sima, Xizhou Zhu, Siqi Chai, Senyao Du, Tianwei Lin, Wenhai Wang, et al. Planning-oriented autonomous driving. In *CVPR*, pages 17853–17862, 2023. 1

[16] Junjie Huang and Guan Huang. Bevpoolv2: A cutting-edge implementation of bevdet toward deployment. *arXiv preprint arXiv:2211.17111*, 2022. 4

[17] Sukjun Hwang, Miran Heo, Seoung Wug Oh, and Seon Joo Kim. Video instance segmentation using inter-frame communication transformers. In *NeurIPS*, volume 34, pages 13352–13363, 2021. 3

[18] Hung Le, Truyen Tran, and Svetha Venkatesh. Learning to remember more with less memorization. In *ICLR*, 2019. 3

[19] Youngwan Lee and Jongyoul Park. Centermask: Real-time anchor-free instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13906–13915, 2020. 14

[20] Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: An online hd map construction and evaluation framework. In *ICRA*, pages 4628–4634. IEEE, 2022. 1, 3, 7

[21] Zhiqi Li, Wenhai Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In *ECCV*, pages 1–18. Springer, 2022. 1, 3

[22] 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. In *ICLR*, 2023. 1, 2, 5, 6, 7, 13

[23] 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. *International Journal of Computer Vision*, pages 1–23, 2024. 1, 2, 3, 4, 6, 7, 8, 9, 13, 14

[24] Chang Liu, Seungho Lee, Scott Varnhagen, and H Eric Tseng. Path planning for autonomous vehicles using model predictive control. In *IV*, pages 174–179. IEEE, 2017. 1

[25] Xiaolu Liu, Song Wang, Wentong Li, Ruizi Yang, Junbo Chen, and Jianke Zhu. Mgmap: Mask-guided learning for online vectorized hd map construction. In *CVPR*, pages 14812–14821, 2024. 7

[26] Yicheng Liu, Tianyuan Yuan, Yue Wang, Yilun Wang, and Hang Zhao. Vectormapnet: End-to-end vectorized hd map learning. In *ICML*, pages 22352–22369. PMLR, 2023. 2, 3, 7, 13

[27] Zihao Liu, Xiaoyu Zhang, Guangwei Liu, Ji Zhao, and Ningyi Xu. Leveraging enhanced queries of point sets for vectorized map construction. In *ECCV*, 2024. 7

[28] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 7

[29] Jean-Arcady Meyer and David Filliat. Map-based navigation in mobile robots:: Ii. a review of map-learning and path-planning strategies. *Cognitive Systems Research*, 4(4):283–317, 2003. 1

[30] Sriram Narayanan, Ramin Moslemi, Francesco Pittaluga, Buyu Liu, and Manmohan Chandraker. Divide-and-conquer for lane-aware diverse trajectory prediction. In *CVPR*, pages 15799–15808, 2021. 1

[31] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, pages 9226–9235, 2019. 3

[32] Jonah Philion and Sanja Fidler. Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In *ECCV*, pages 194–210. Springer, 2020. 1, 3, 4

[33] Limeng Qiao, Wenjie Ding, Xi Qiu, and Chi Zhang. End-to-end vectorized hd-map construction with piecewise bezier curve. In *CVPR*, pages 13218–13228, 2023. 7

[34] Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. Compressive transformers for long-range sequence modelling. In *ICLR*, 2019. 3

[35] Michael Ryoo, AJ Piergiovanni, Anurag Arnab, Mostafa Dehghani, and Anelia Angelova. Tokenlearner: Adaptive space-time tokenization for videos. In *NeurIPS*, volume 34, pages 12786–12797, 2021. 5

[36] Michael S Ryoo, Keerthana Gopalakrishnan, Kumara Kahatapitiya, Ted Xiao, Kanishka Rao, Austin Stone, Yao Lu, Julian Ibarz, and Anurag Arnab. Token turing machines. In *CVPR*, pages 19070–19081, 2023. 3, 5

[37] Tixiao Shan and Brendan Englot. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *IROS*, pages 4758–4765. IEEE, 2018. 1, 3

[38] Tixiao Shan, Brendan Englot, Drew Meyers, Wei Wang, Carlo Ratti, and Daniela Rus. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In *IROS*, pages 5135–5142. IEEE, 2020. 1, 3

[39] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, 2017. 3, 5

[40] Jingke Wang, Tengju Ye, Ziqing Gu, and Junbo Chen. Ltp: Lane-based trajectory prediction for autonomous driving. In *CVPR*, pages 17134–17142, 2022. 1

[41] Shuo Wang, Fan Jia, Yingfei Liu, Yucheng Zhao, Zehui Chen, Tiancai Wang, Chi Zhang, Xiangyu Zhang, and Feng Zhao. Stream query denoising for vectorized hd map construction. *arXiv preprint arXiv:2401.09112*, 2024. 2, 3, 7

[42] Shihao Wang, Yingfei Liu, Tiancai Wang, Ying Li, and Xiangyu Zhang. Exploring object-centric temporal modeling for efficient multi-view 3d object detection. In *ICCV*, pages 3621–3631, 2023. 3

[43] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. In *CVPR*, pages 8741–8750, 2021. 3, 6

[44] 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. In *NeurIPS Datasets and Benchmarks*, 2021. 2, 6, 7

[45] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, pages 5188–5197, 2019. 3

[46] Tianyuan Yuan, Yicheng Liu, Yue Wang, Yilun Wang, and Hang Zhao. Streammapnet: Streaming mapping network for vectorized online hd map construction. In *WACV*, pages 7356–7365, 2024. 2, 3, 7, 8, 9, 13, 14

[47] Gongjie Zhang, Jiahao Lin, Shuang Wu, Zhipeng Luo, Yang Xue, Shijian Lu, Zuoguan Wang, et al. Online map vectorization for autonomous driving: A rasterization perspective. In *NeurIPS*, volume 36, 2023. 7

[48] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and systems*, volume 2, pages 1–9. Berkeley, CA, 2014. 1, 3

[49] Zhixin Zhang, Yiyuan Zhang, Xiaohan Ding, Fusheng Jin, and Xiangyu Yue. Online vectorized hd map construction using geometry. In *ECCV*, 2024. 7

[50] Brady Zhou and Philipp Krähenbühl. Cross-view transformers for real-time map-view semantic segmentation. In *CVPR*, pages 13760–13769, 2022. 1, 3

[51] Yi Zhou, Hui Zhang, Jiaqian Yu, Yifan Yang, Sangil Jung, Seung-In Park, and ByungIn Yoo. Himap: Hybrid representation learning for end-to-end vectorized hd map construction. In *CVPR*, pages 15396–15406, 2024. 7, 8, 9, 13

[52] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *ICLR*, 2021. 3, 5

A Appendix

A.1 Clip-level Inference Scheme

Our MapUnveiler runs in a clip-level. Specifically, the k-th clip of C_k includes

$$C_k = \{f_t\}_{t=kS+1}^{kS+T}. \tag{7}$$

MapUnveiler infers from $k = 0$. f_t represents a frame at time t, and $\{f_t\}_{t=kS+1}^{kS+T}$ denote a sequence of consecutive frames from $kS + 1$ to $kS + T$, where S and T indicate the clip stride and the temporal window, respectively. Therefore, when $T = 3$ and $S = 2$, we obtain clip sets such as $C_0 = \{f_1, f_2, f_3\}$, $C_1 = \{f_3, f_4, f_5\}$, ..., and $C_k = \{f_{2k+1}, f_{2k+2}, f_{2k+3}\}$.

A.2 Predicting with Centerline

In Tab. 12, we present a quantitative comparison on centerline predictions. MapUnveiler outperforms MapTRv2 [23] by significant margins on both nuScenes (54.0%→63.0%, +9.0%) and Argoverse (62.6%→68.0%, +5.4%) benchmarks. This demonstrates the robustness of our approach on various map categories.

Table 12: Centerline predictions. AP$_p$, AP$_d$, AP$_b$, AP$_c$ indicate the average precision for pedestrian crossing, divider, boundary, and centerline, respectively.

Method	nuScenes						Argoverse2					
	Epoch	AP$_p$	AP$_d$	AP$_b$	AP$_c$	mAP	Epoch	AP$_p$	AP$_d$	AP$_b$	AP$_c$	mAP
MapTRv2[IJCV'24] [23]	24	50.1	53.9	58.8	53.1	54.0	6	55.2	67.2	64.8	63.2	62.6
MapUnveiler (ours)	24	**59.0**	**65.7**	**68.0**	**59.4**	**63.0**	6	**62.2**	**73.2**	**70.0**	**66.6**	**68.0**

A.3 Extension to 3D Vectorized Map Construction

Tab. 13, presents a comparison on 3D map construction. In this study, we conduct experiment on Argoverse2 dataset. MapUnveiler achieves state-of-the-art performance in Tab. 13, demonstrating the efficacy of our approach even when extended to 3D VHC.

Table 13: Comparisons on Argoverse2 `val` set with 3D map construction.

Method	Epoch	AP$_p$	AP$_d$	AP$_b$	mAP
MapTRv2[IJCV'24] [23]	6	60.7	68.9	64.5	64.7
HIMap[CVPR'24] [51]	6	**66.7**	68.3	**70.3**	68.4
MapUnveiler	6	66.0	**72.6**	67.6	**68.7**

A.4 A More Comparison on Geo-disjoint Split

Tab. 14 presents a experimental results on a recent geo-disjoint dataset split proposed in StreamMapNet [46]. In this dataset, the performance of MapTRv2 [23] has been significantly drop from 61.5% to 36.6%. We also successfully achieve state-of-the-art performance on geo-disjoint dataset split.

Table 14: Experimental results on geo-disjoint dataset split proposed in StreamMapNet [46]. The results are obtained on nuScenes with 60×30m range.

Method	Epoch	AP$_p$	AP$_d$	AP$_b$	mAP
VectorMapNet[ICML'23] [26]	120	15.8	17.0	21.2	18.0
MapTR[ICLR'23] [22]	24	6.4	20.7	35.5	20.9
StreamMapNet[WACV'24] [46]	24	29.6	**30.1**	41.9	33.9
MapTRv2[IJCV'24] [23]	24	37.2	26.5	46.1	36.6
MapUnveiler (ours)	24	**43.2**	26.5	**48.7**	**39.4**

A.5 Various Backbones

We present experimental results with various backbones: ResNet-18 [12] and V2-99 [19]. As shown in Tab. 15, our method is not limited to MapTRv2 [23] with ResNet50, but can be extended to ResNet18 and V2-99 backbones.

Table 15: Experimental results with various backbones.

Range	Method	Backbone	nuScenes					Argoverse2				
			Epoch	AP$_p$	AP$_d$	AP$_b$	mAP	Epoch	AP$_p$	AP$_d$	AP$_b$	mAP
$60 \times 30\ m$	MapTRv2[IJCV'24] [23]	R18	24	53.3	58.5	58.5	56.8	6	58.8	68.5	64	63.8
	MapTRv2[IJCV'24] [23]	R50	24	59.8	62.4	62.4	61.5	6	62.9	72.1	67.1	67.4
	MapTRv2[IJCV'24] [23]	V2-99	24	63.6	67.1	69.2	66.6	6	64.5	72.2	70.1	68.9
	MapUnveiler (ours)	R18	24	62.4	65.2	65.7	64.4	6	63.8	70.1	67.1	67.0
	MapUnveiler (ours)	R50	24	67.6	67.6	68.8	68.0	6	68.9	73.7	68.9	70.5
	MapUnveiler (ours)	V2-99	24	**69.8**	**72.0**	**74.7**	**72.1**	6	**69.6**	**75.1**	**72.8**	**72.5**
$100 \times 50\ m$	MapTRv2[IJCV'24] [23]	R18	24	52.7	57.3	51.5	53.8	6	60.3	57.6	49.6	55.8
	MapTRv2[IJCV'24] [23]	R50	24	58.1	61.0	56.6	58.6	6	66.2	61.4	54.1	60.6
	MapTRv2[IJCV'24] [23]	V2-99	24	62.6	67.8	65.2	65.2	6	68.5	62.1	58.4	63.0
	MapUnveiler (ours)	R18	24	64.1	67.3	65.3	65.6	6	65.4	63.2	54.7	61.1
	MapUnveiler (ours)	R50	24	68.0	70.0	68.2	68.7	6	69.7	**67.1**	59.3	65.4
	MapUnveiler (ours)	V2-99	24	**70.5**	**74.6**	**73.6**	**72.9**	6	**71.1**	66.1	**61.4**	**66.2**

A.6 Randomly Dropped Intermediate Frames

In the limitations section (Sec. 4.5), we discussed that our MapUnveiler relies on temporally consecutive frames, and the performance can degrade when intermediate frames are missing. However, we expected that the performance would recover quickly from the subsequent unveiling pipeline. To validate this, we evaluated two models with randomly dropped intermediate frames. Frames were dropped by converting multi-camera images into black images. The experiment was conducted with drop rates of 20%, 10%, and 5%, and the results are given in Tab. 16. MapUnveiler is affected by dropped frames, but the performance degradation is reasonable compared to MapTRv2 [23].

Table 16: Experimental results with randomly dropped intermediate frames. The results are obtained on nuScenes with 60×30m range

Method	Drop rate	AP$_p$	AP$_d$	AP$_b$	mAP
MapTRv2[IJCV'24] [23]	0%	58.8	61.8	62.8	61.2
	5%	56.6	56.6	58.7	57.7
	10%	53.4	54.8	55.3	54.5
	20%	47.2	48.5	38.9	48.2
MapUnveiler (Ours)	0%	69.5	69.4	70.5	69.8
	5%	58.0	60.6	60.0	66.9
	10%	63.3	64.7	65.0	64.3
	20%	66.2	66.9	67.6	59.6

A.7 More Qualitative Results

We present additional qualitative results in Figs. 6, 7, 8, and 9. For all results, we provide multi-camera input and the results of MapTRv2 [23], StreamMapNet [46], and our MapUnveiler.

A.8 Broader Impacts

While our framework notably enhances the performance of online vectorized HD map construction relying on streaming multi-view camera sensors, it does not assure flawless prediction of all map elements. Therefore, it is crucial to have a backup plan ready for safety-critical real-world applications.

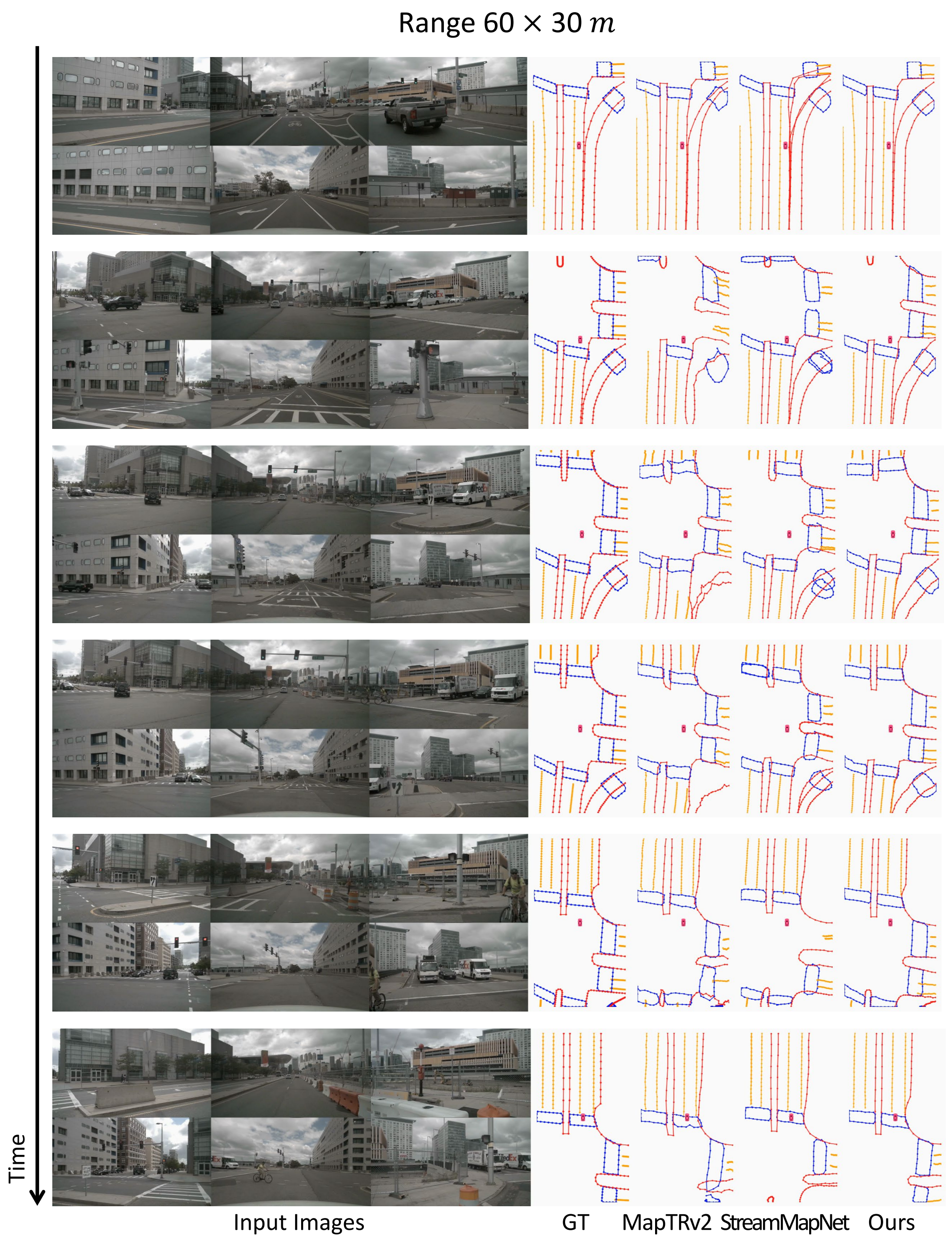

Figure 6: Qualitative comparisons on nuScenes `val` with $60\times30m$ perception range setting.

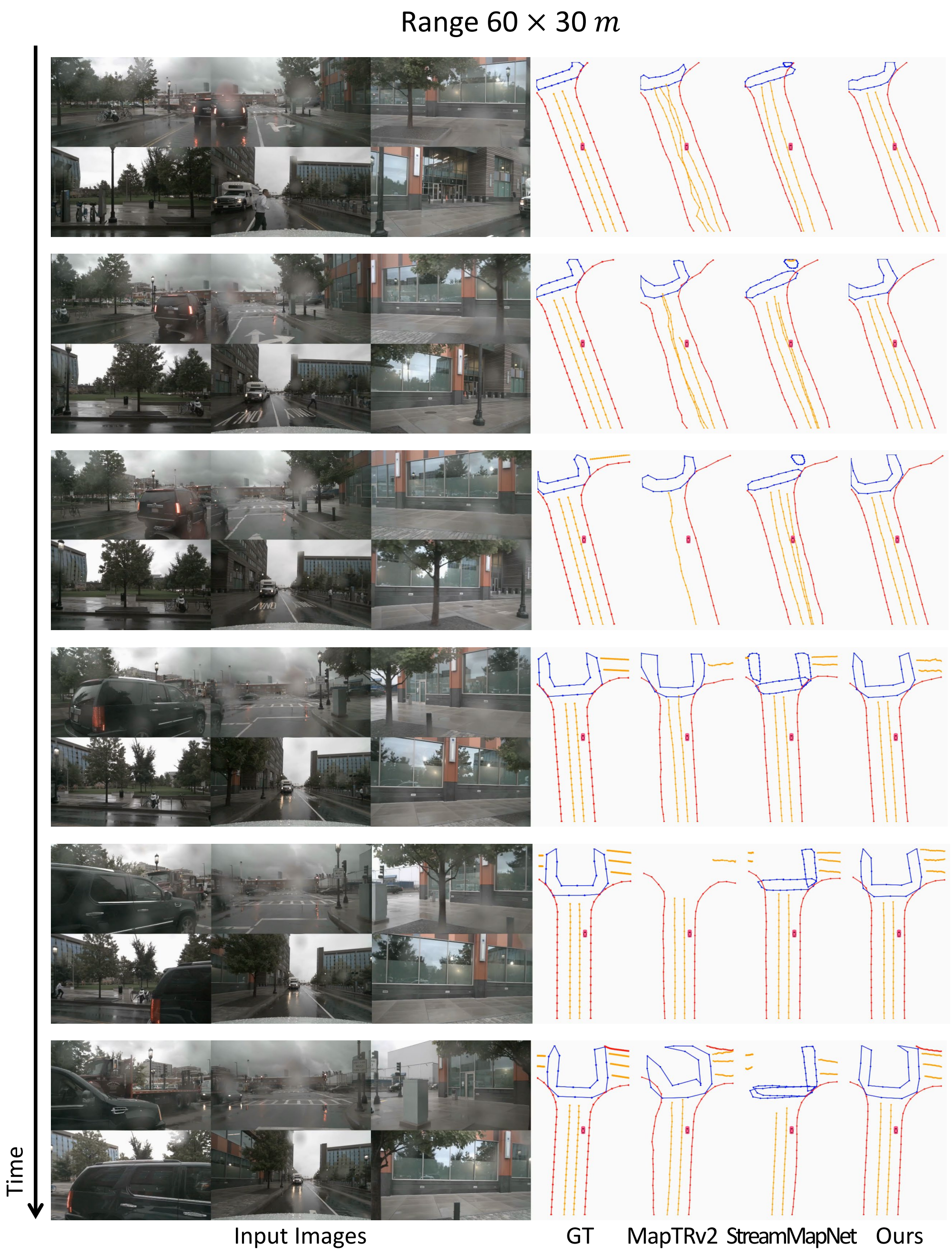

Figure 7: Qualitative comparisons on nuScenes `val` with 60×30m perception range setting.

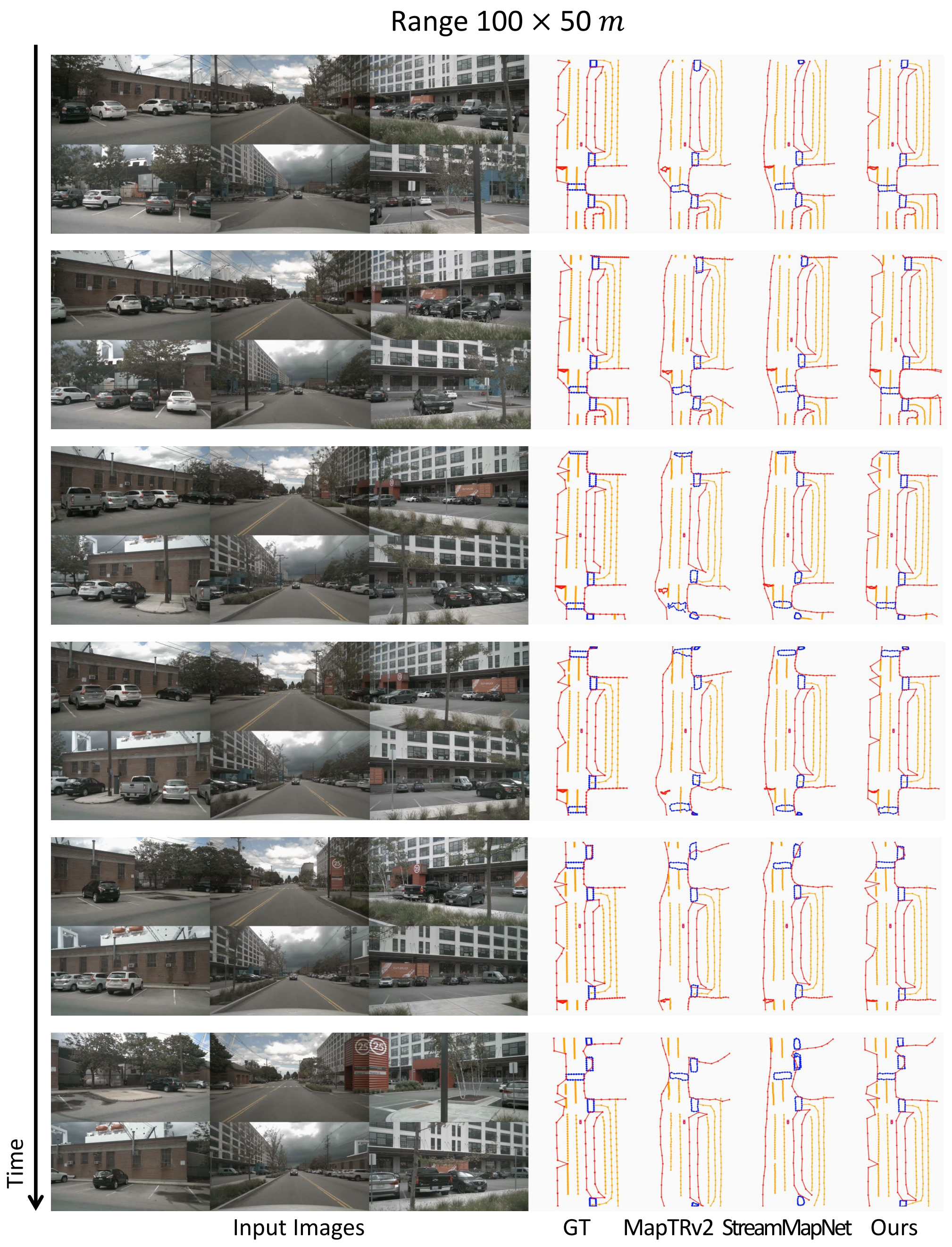

Figure 8: Qualitative comparisons on nuScenes `val` with $100 \times 50m$ perception range setting.

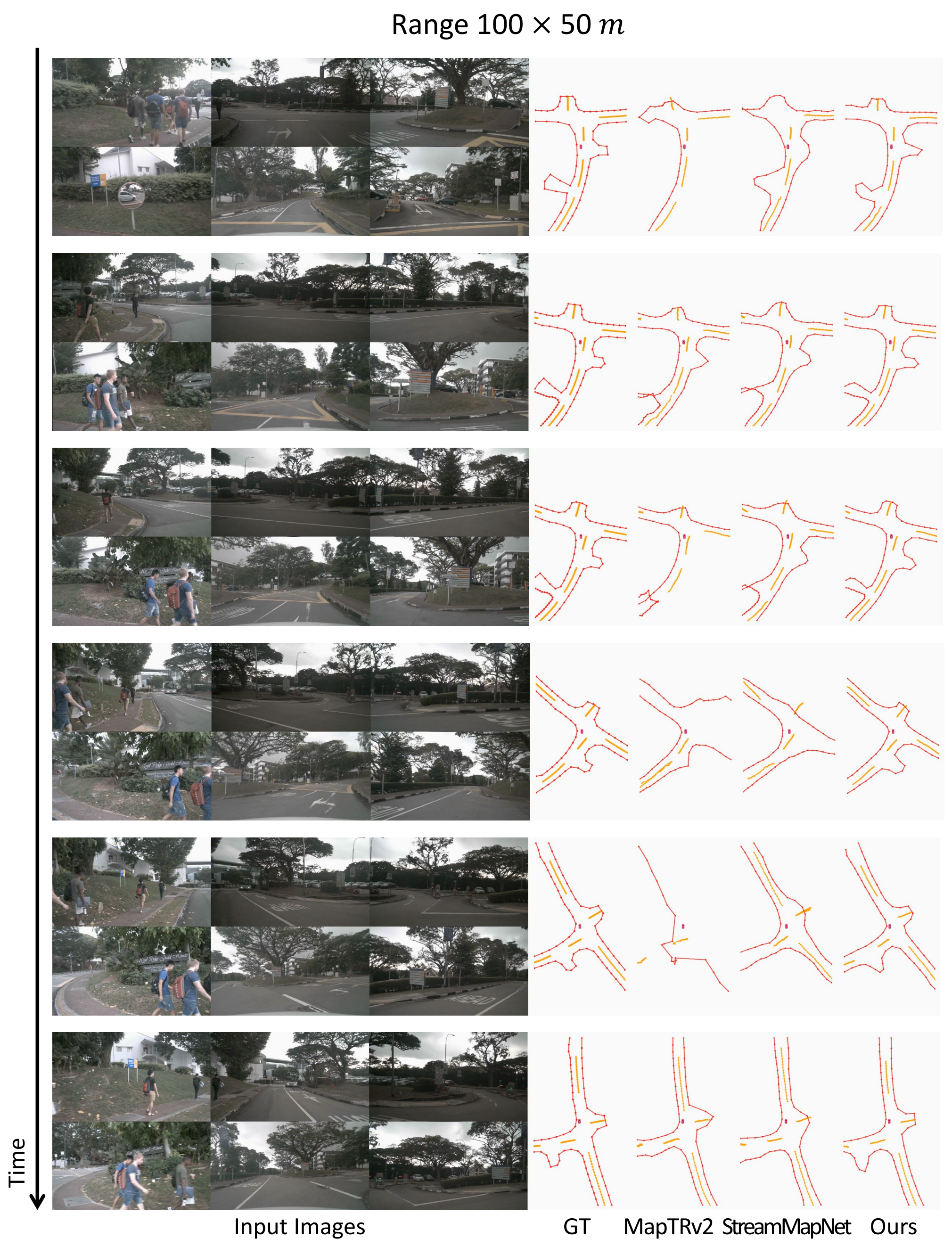

Figure 9: Qualitative comparisons on nuScenes `val` with $100 \times 50m$ perception range setting.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The contributions of our work are summarized at the end of the introduction, aligning with the content of the abstract.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: In Sec. 4.5 (Limitation) we addressed and outlined the constraints of the suggested method.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [NA]

Justification:

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: In Sec. 4.2, we provide implementation details in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We plan to release source codes upon acceptance.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (https://nips.cc/public/guides/CodeSubmissionPolicy) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: In Sec. 4, we provide detailed information on the dataset and experimental setup.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification:

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: In Sec. 4.2, we describe the computing resources used for training in the paper.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We abide by the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: In Appendix (Sec. A.8), we describe the broader impacts of our work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Proper citation has been provided for the data, models, and code used in the paper.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.