MapTracker: Tracking with Strided Memory Fusion for Consistent Vector HD Mapping

Jiacheng Chen[1*], Yuefan Wu[1*], Jiaqi Tan[1*], Hang Ma[1], and Yasutaka Furukawa[1,2]

[1]Simon Fraser University [2]Wayve

Abstract. This paper presents a vector HD-mapping algorithm that formulates the mapping as a tracking task and uses a history of memory latents to ensure consistent reconstructions over time. Our method, *MapTracker*, accumulates a sensor stream into memory buffers of two latent representations: 1) Raster latents in the bird's-eye-view (BEV) space and 2) Vector latents over the road elements (i.e., pedestrian-crossings, lane-dividers, and road-boundaries). The approach borrows the query propagation paradigm from the tracking literature that explicitly associates tracked road elements from the previous frame to the current, while fusing a subset of memory latents selected with distance strides to further enhance temporal consistency. A vector latent is decoded to reconstruct the geometry of a road element. The paper further makes benchmark contributions by 1) Improving processing code for existing datasets to produce consistent ground truth with temporal alignments and 2) Augmenting existing mAP metrics with consistency checks. MapTracker significantly outperforms existing methods on both nuScenes and Agroverse2 datasets by over 8% and 19% on the conventional and the new consistency-aware metrics, respectively. The code and models are available on our project page: `https://map-tracker.github.io`.

1 Introduction

Humans forget, so do neural networks. A robust memory is crucial for online systems to produce consistent outputs. Vector HD mapping, a task of reconstructing vector road geometries from vehicle sensor data, has made dramatic progress. A consistent vector HD mapping system, capable of reconstructing a consistent HD map of a city from a single drive-through (See Figure 1 for examples), would have a tremendous impact on our society, reducing the cost of HD map creation for tens of thousands of cities in the world and enhancing the safety and stability of self-driving cars.

Existing vector HD mapping methods [16, 18, 19, 24, 41] focus on per-frame reconstruction via detection-style transformer networks [4]. They detect road elements anew in every frame without consistency enforcement, potentially guided by reconstructions from the previous frame. Furthermore, a standard recurrent

*Equal contribution.

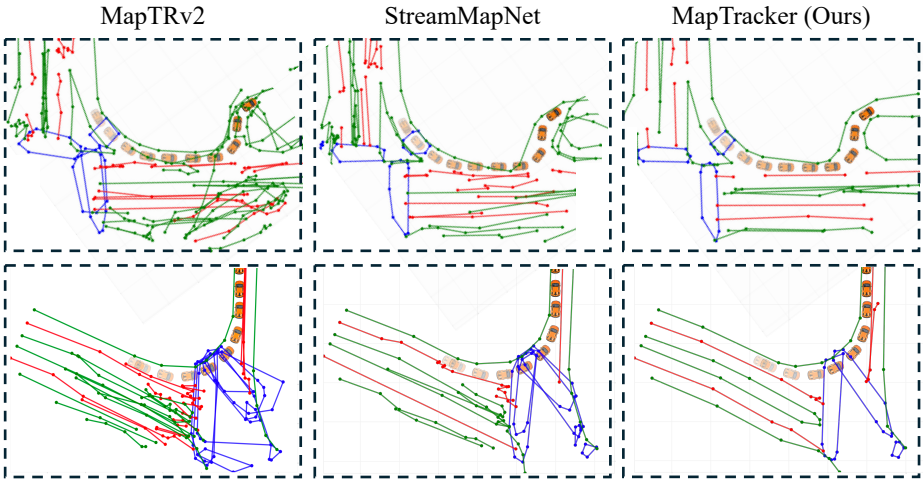

Fig. 1: MapTracker produces high-quality and temporally consistent vector HD maps, which are progressively merged into a global vector HD map by a simple online algorithm. The current state-of-the-art methods, MapTRv2 [19] and StreamMapNet [41], fail to produce consistent reconstructions, leading to very noisy global maps. The figure shows two challenging scenarios (cars are turning) from the nuScenes [2] dataset.

latent embedding is often the choice for memory mechanism [41], where accumulating the entire history in a single latent memory proves challenging, especially for cluttered environments with numerous vehicles obscuring road structures.

Towards ultimate temporal consistency, this paper presents MapTracker with two key design elements. First, tracking instead of detection becomes the formulation, specifically borrowing the query propagation paradigm from the tracking literature that explicitly associates tracked road elements across frames. Second, a sequence of memory latents from past frames serves as the memory mechanism. Concretely, we retain memory buffers for two latent representations from the past frames: 1) Raster latents in the bird's-eye-view (BEV) space and 2) Vector latents over the tracked road elements, while using a subset of memory latents based on distance strides for effective information fusion. A vector latent reconstructs a road element geometry.

To prepare the tracking labels and measure the consistency of HD map reconstructions, this paper introduces a new benchmark based on nuScenes [2] and Agroverse2 [37] datasets. Specifically, we improve the processing code of the two datasets to produce consistent ground truth data with temporal alignments, then propose a consistency-aware mean average precision (mAP) metric.

We have made extensive comparative evaluations based on the traditional and the new mAP metrics. MapTracker significantly outperforms the competing methods by over 8% on the conventional distance-based mAP, reaching 76.1 mAP on nuScenes and 76.9 mAP on Argoverse2. With the new consistency-aware metrics, MapTracker demonstrates superior temporal consistency and improves the StreamMapNet baseline by over 19%.

To summarize, this paper makes three contributions: 1) A novel vector HD mapping algorithm that formulates HD mapping as tracking and leverages the history of memory latents in two representations to achieve temporal consistency; 2) An improved vector HD mapping benchmark with temporally consistent ground truth and a consistency-aware mAP metric; and 3) SOTA performance with significant improvements over the current best methods on traditional and new metrics. The code and the new benchmark data will be available.

2 Related Work

This paper tackles consistent vector HD mapping by 1) borrowing an idea from the visual object tracking literature and 2) devising a new memory mechanism. The section first reviews recent trends in visual object tracking with transformers and memory designs in vision-based autonomous driving. Lastly, we discuss competing vector HD mapping methods.

Visual object tracking with transformers. Visual object tracking [40] has a long history, where end-to-end transformer [35] methods become a recent trend due to the simplicity. TrackFormer [27], TransTrack [34], and MOTR [42, 45] leverage the attention mechanism with *track queries* to explicitly associate instances across frames. MeMOT [3] and MeMOTR [8] further extend the tracking transformers with memory mechanisms for better long-term consistency. This paper formulates vector HD mapping as a tracking task by incorporating track queries with a more robust memory mechanism.

Memory designs in autonomous driving. Single-frame self-driving systems have difficulty in handling occlusion, sensor failure, or complex environments. Temporal modeling with memories offers promising complements [9, 10, 12, 13, 22, 23, 36, 39, 41]. Many memory designs exist for the raster BEV features [17, 29], which form the foundation of most autonomous-driving tasks [15, 26]. BEVDet4D [12] and BEVFormerv2 [39] stack features of multiple past frames as a memory, but the computation scales linearly with history length, struggling to capture long-term information. VideoBEV [10] propagates BEV raster queries across frames to accumulate information recurrently. In the vector domain, Sparse4Dv2 [22] employs a similar RNN-style memory for object queries, while Sparse4Dv3 [23] further uses temporal denoising for robust temporal learning. These ideas have been partially incorporated by vector HD mapping approaches [36, 41]. This paper proposes a new memory design for both the raster BEV latents and the vector latents of road elements.

Vector HD mapping. Traditionally HD maps are reconstructed offline with SLAM-based methods [32, 33, 44], followed by human curation, requiring high maintenance costs. Online vector HD mapping algorithms are gaining more interest over their offline counterparts as their accuracy and efficiency improve, which would simplify the production pipeline and handle map changes. HDMapNet [16] turns raster map segmentation into vector map instances via post-processing and has established the first Vector HD mapping benchmark. VectorMapNet [24]

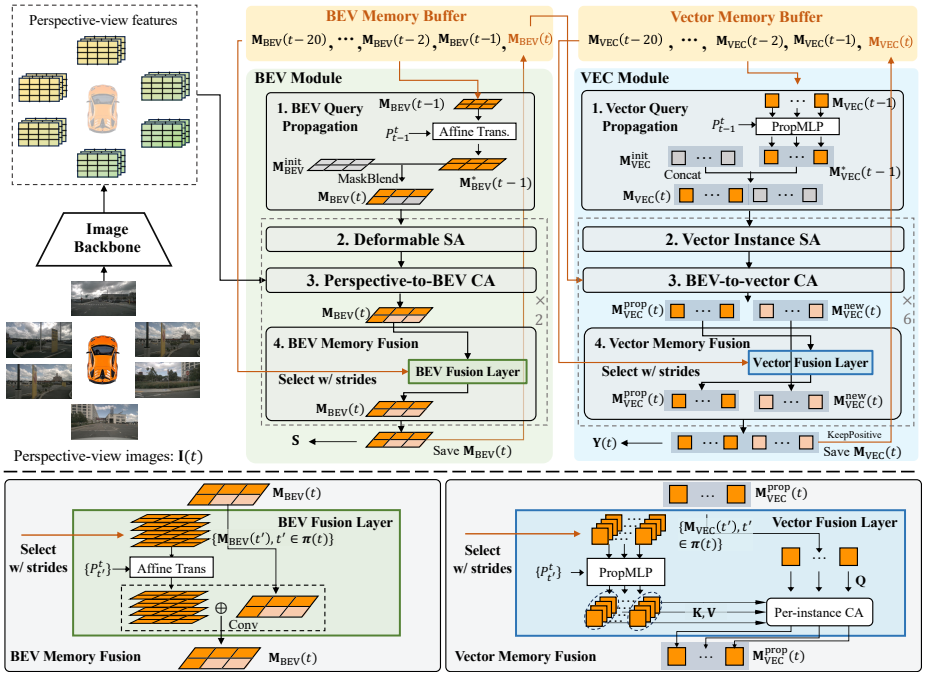

Fig. 2: (Top) The overall architecture of MapTracker. **(Bottom)** The close-up views of the BEV and the Vector fusion layers.

and MapTR [18] both leverage DETR-based [4] transformers for end-to-end prediction. The former predicts the vertices of each detected curve autoregressively, while the latter uses hierarchical queries and matching loss to predict all the vertices simultaneously. MapTRv2 [19] further complements MapTR with auxiliary tasks and network modifications. Curve representation [7, 30, 46], network design [38], and training paradigm [5, 43] are the focus of other works. StreamMapNet [41] steps towards consistent mapping by borrowing the streaming idea from BEV perception. The idea accumulates the past information into memory latents and passes as a condition (i.e., a conditional detection framework). SQD-MapNet [36] proposes temporal curve denoising to facilitate temporal learning, mimicking DN-DETR [14].

3 MapTracker

A robust memory mechanism is the core of MapTracker, accumulating a sensor stream into latent memories of two representations: 1) Bird's-eye-view (BEV) memory of a region around a vehicle in the top-down BEV coordinate frame as a latent image; and 2) Vector (VEC) memory of road elements (i.e., pedestrian-crossings, lane-dividers, and road-boundaries) as a set of latent vectors.

Two simple ideas with the memory mechanism achieve consistent mapping. The first idea is to use a buffer of memories from the past instead of a single

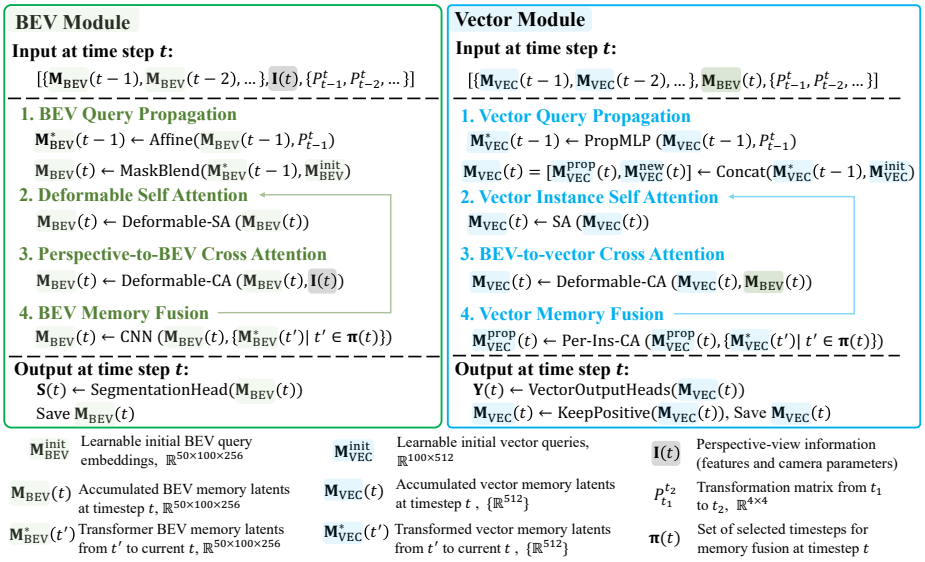

Fig. 3: The architecture details of the BEV and the Vector modules. The BEV-related representations are in green, while the vector-related representations are in cyan. Details of the attention layers are described in §3.

memory at the current frame [10, 17, 41]. A single memory should hold the entire past information but is susceptible to memory loss, especially in cluttered environments with numerous vehicles obscuring road structures. Concretely, we select a subset of the past latent memories for fusion at each frame based on the vehicle motions for efficiency and coverage. The second idea is to formulate the online HD mapping as a tracking task. The VEC memory mechanism maintains a sequence of memory latents with each road element and makes this formulation straightforward by borrowing a query propagation paradigm from the tracking literature. The rest of the section explains our neural architectures (See Figure 2 and Figure 3), consisting of the BEV and VEC memory buffers and their corresponding network modules, and then presents the training details.

3.1 Memory Buffers

A BEV memory, $\mathbf{M}_{\text{BEV}}(t) \in \mathbb{R}^{50\times100\times256}$, is a 2D latent image in the BEV coordinate frame centered and oriented with the vehicle at frame t. The spatial dimension (i.e., 50×100) covers a rectangular area, 15m left/right and 30m front/back. Each memory latent accumulates the entire past information, while the buffer holds such memory latents for the last 20 frames, making the memory mechamism redundant but robust.

A VEC memory, $\mathbf{M}_{\text{VEC}}(t) \in \{\mathbb{R}^{512}\}$, is a set of vector latents, each of which accumulates information of an active road element up to frame t. The number of active elements varies per frame. The buffer holds the latent vectors of the past

20 frames and their correspondences across frames (i.e., a sequence of vector latents corresponding to the same road element).

3.2 BEV Module

Inputs are 1) CNN features of the onboard perspective images processed by the image backbone (the official ResNet50 model [11] pretrained on ImageNet [6]) and their camera parameters $\mathbf{I}(t)$; 2) the BEV memory buffer $\{\mathbf{M}_{\text{BEV}}(t-1), \mathbf{M}_{\text{BEV}}(t-2), ...\}$; and 3) the vehicle motions $\left\{P^t_{t-1}, P^t_{t-2}, ...\right\}$, where $P^{t_2}_{t_1} \in \mathbb{R}^{4\times4}$ is the affine transformation of the vehicle coordinate frame from frame t_1 to t_2. The following explains the four components of the BEV module architecture and its outputs.

[1. BEV Query Propagation]. A BEV memory is a 2D latent image in a vehicle coordinate frame. An affine transformation P^t_{t-1} and a bilinear interpolation initialize the current BEV memory $\mathbf{M}_{\text{BEV}}(t)$ with the previous one $\mathbf{M}_{\text{BEV}}(t-1)$. For pixels that fall outside the latent image after the transformation, per-pixel learnable embedding vectors $\mathbf{M}^{\text{init}}_{\text{BEV}} \in \mathbb{R}^{50\times100\times256}$ are the initialization instead, whose operation is denoted as "MaskBlend" in Figure 3.

[2. Deformable Self-Attention]. A deformable self-attention layer [47] enriches the BEV memory $\mathbf{M}_{\text{BEV}}(t)$.

[3. Perspective-to-BEV Cross-Attention]. Similar to StreamMapNet [41], a spatial deformable cross-attention layer from BEVFormer [17] injects the perspective-view information $\mathbf{I}(t)$ into $\mathbf{M}_{\text{BEV}}(t)$, followed by a standard feed-forward network (FFN) layer [35].

[4. BEV Memory Fusion]. The memory latents in the buffer are fused to enrich $\mathbf{M}_{\text{BEV}}(t)$. Using all the memories is computationally expensive and redundant. We use a strided selection of four memories without repetition, whose vehicle positions are the closest to (1m/5m/10m/15m) from the current position. An affine transformation and a bilinear interpolation align the coordinate frames of the selected memories to the current: $\{\mathbf{M}^*_{\text{BEV}}(t'), t' \in \boldsymbol{\pi}(t)\}$, where $\boldsymbol{\pi}(t)$ denotes the selected times. We concatenate $\mathbf{M}_{\text{BEV}}(t)$ with the aligned memories and use a lightweight residual block with two convolution layers to udpate $\mathbf{M}_{\text{BEV}}(t)$. The last three components of the BEV module repeat twice without weight sharing.

Outputs are 1) the final memory $\mathbf{M}_{\text{BEV}}(t)$ saved to the buffer and passed to the VEC module; and 2) the rasterized road element geometries $\mathbf{S}(t)$ which is inferred by a segmentation head and used for a loss calculation (See §3.4). The segmentation head is a linear projection module that projects each pixel in the memory latent to a 2×2 segmentation mask, thus producing a 100×200 mask.

3.3 VEC Module

Inputs are 1) the BEV memory $\mathbf{M}_{\text{BEV}}(t)$; 2) the vector memory buffer $\{\mathbf{M}_{\text{VEC}}(t-1), \mathbf{M}_{\text{VEC}}(t-2), ...\}$; and 3) the vehicle motions $\left\{P^t_{t-1}, P^t_{t-2}, ...\right\}$.

[1. Vector Query Propagation]. A vector memory is a set of latent vectors of the active road elements. Borrowing the query propagation paradigm from transformer-based tracking approaches [27,34,42], we initialize the vector memory as $\mathbf{M}_{\text{VEC}}(t) = [\mathbf{M}^{\text{prop}}_{\text{VEC}}(t), \mathbf{M}^{\text{new}}_{\text{VEC}}(t)]$. $\mathbf{M}^{\text{new}}_{\text{VEC}}(t)$ denotes 100 latent vectors for 100 new road element candidates, which are initialized with 100 learnable embeddings $\mathbf{M}^{\text{init}}_{\text{VEC}}$. $\mathbf{M}^{\text{prop}}_{\text{VEC}}(t)$ denotes latent vectors for the currently tracked road elements, which are initialized with the corresponding latent vectors in the previous memory $\mathbf{M}_{\text{VEC}}(t-1)$ after using a two-layer MLP to align the coordinate frame. Concretely, we turn P^t_{t-1} into a 4D vector of rotation quaternion and 3D vector of translation parameters, represent with their positional encodings [35], concatenate them with each vector latent in $\mathbf{M}_{\text{VEC}}(t-1)$, and apply an MLP. We call it PropMLP, which handles the temporal propagation.

[2. Vector Instance Self Attention]. Similar to StreamMapNet, a standard self-attention layer enriches the vector latents in the memory $\mathbf{M}_{\text{VEC}}(t)$.

[3. BEV-to-Vector Cross Attention]. The Multi-Point Attention from StreamMapNet, which is an extension of the vanilla deformable cross-attention [47], injects the BEV information from $\mathbf{M}_{\text{BEV}}(t)$ into $\mathbf{M}_{\text{VEC}}(t)$.

[4. Vector Memory Fusion]. For each latent vector in the current memory $\mathbf{M}_{\text{VEC}}(t)$, latent vectors in the buffer associated with the same road element are fused to enrich its representation. The same strided frame-selection chooses four latent vectors, where the selected frames $\boldsymbol{\pi}(t)$ would be different and fewer for some road elements with a short tracking history. For example, an element that has been tracked for two frames has only two latents in the buffer. A standard cross-attention followed by an FFN layer injects the selected latents after aligning their coordinate frames by the same PropMLP module $\{\mathbf{M}^*_{\text{VEC}}(t'), t' \in \boldsymbol{\pi}(t)\}$. To be precise, a query is a latent in $\mathbf{M}^{\text{prop}}_{\text{VEC}}(t)$, where a key/value is a latent in $\{\mathbf{M}^*_{\text{VEC}}(t'), t' \in \boldsymbol{\pi}(t)\}$. In Figure 3, we omit the element index for the "Per-Ins-CA" operation, which acts on each element independently. The last three components of the VEC module repeat six times without weight sharing.

Outputs are 1) the final memory $\mathbf{M}_{\text{VEC}}(t)$ for "positive" road elements that pass the classification test by a single fully connected layer from $\mathbf{M}_{\text{VEC}}(t)$; and 2) vector road geometries of the positive road elements, regressed by the 3-layer MLP from $\mathbf{M}_{\text{VEC}}(t)$. The threshold of the classification test is 0.4 for the first frame, and 0.5/0.6 for the propagated/new road elements for subsequent frames. $\mathbf{Y}(t) = \{(V_i, p_i)\}$ denotes the outputs. Following the prior convention [18], each element geometry $V_i = [(x_1, y_1), \ldots, (x_{20}, y_{20})]$ is a polygonal curve with 20 points in the BEV coordinate frame. p_i is the class probability score.

3.4 Training

The ground-truth road element geometries are denoted as $\hat{\mathbf{Y}}(t) = \{\hat{Y}_i\}$. $\hat{Y}_i = (\hat{V}_i, \hat{c}_i)$, where V_i has 20 points interpolated from the raw ground-truth vector. $\hat{c}_i$ is the class label. Standard OpenCV and PIL libraries rasterize $\hat{\mathbf{Y}}(t)$ on an empty BEV canvas to obtain the ground-truth segmentation image $\hat{\mathbf{S}}(t)$

BEV loss. We employ per-pixel Focal loss [21] and per-class Dice loss [28] on the BEV outputs $\mathbf{S}(t)$, which are common auxiliary losses in vector HD mapping approaches [19, 31]. The loss is defined by

$$\mathcal{L}_{\text{BEV}} = \lambda_1 \mathcal{L}_{\text{focal}}(\mathbf{S}(t), \hat{\mathbf{S}}(t)) + \lambda_2 \mathcal{L}_{\text{dice}}(\mathbf{S}(t), \hat{\mathbf{S}}(t)) \tag{1}$$

VEC loss. Inspired by MOTR [42], an end-to-end transformer for multi-object tracking, we extend the matching-based loss [18,41] to explicitly consider ground-truth tracks (See §4 for ground-truth processing). For each frame t, $\hat{\mathbf{Y}}(t)$ consists of two disjoint subsets: new elements $\hat{\mathbf{Y}}_{\text{new}}(t)$ and tracked elements $\hat{\mathbf{Y}}_{\text{track}}(t)$. For the vector outputs, we denote the results from the propagated latents $\mathbf{M}^{\text{prop}}_{\text{VEC}}(t)$ as $\mathbf{Y}_{\text{track}}(t)$, and results from the new latents $\mathbf{M}^{\text{new}}_{\text{VEC}}(t)$ as $\mathbf{Y}_{\text{new}}(t)$. Note that to make the VEC module robust to potential errors in pose estimation, we randomly perturb the transformation matrix P^t_{t-1} by adding a Gaussian noise during training. We train the module with a tracking loss that explicitly considers the temporal alignments. §4 explains the ground-truth preparation. The optimal instance-level label assignment for new elements is defined as:

$$\boldsymbol{\omega}_{\text{new}}(t) = \underset{\boldsymbol{\omega}_{\text{new}}(t) \in \boldsymbol{\Omega}(t)}{\arg\min} \mathcal{L}_{\text{match}}(\hat{\mathbf{Y}}_{\text{new}}(t)|_{\boldsymbol{\omega}_{\text{new}}(t)}, \mathbf{Y}_{\text{new}}(t)). \tag{2}$$

$\boldsymbol{\Omega}(t)$ is the space of all bipartite matches. $\mathcal{L}_{\text{match}}$ is the hierarchical matching cost similar to the one proposed in MapTR [18], consisting of a focal loss $\mathcal{L}_{\text{focal}}(\{\hat{c}_i\}|_{\boldsymbol{\omega}_{\text{new}}(t)}, \{p_i\})$ and a permutation-invariant line coordinate loss $\mathcal{L}_{\text{line}}(\{\hat{V}_i\}|_{\boldsymbol{\omega}_{\text{new}}(t)}, \{V_i\})$. The label assignments $\boldsymbol{\omega}(t)$ between all outputs and ground truth is then defined inductively:

$$\boldsymbol{\omega}(t) = \boldsymbol{\omega}_{\text{track}}(t) \cup \boldsymbol{\omega}_{\text{new}}(t);\ \boldsymbol{\omega}_{\text{track}}(t) = \begin{cases} \varnothing, & \text{if } t = 0 \\ \boldsymbol{\omega}(t-1), & \text{if } t > 0 \end{cases}. \tag{3}$$

$\boldsymbol{\omega}_{\text{track}}(t)$ is the label assignments between $\mathbf{Y}_{\text{track}}(t)$ and $\hat{\mathbf{Y}}_{\text{track}}(t)$. The tracking-style loss for the vector outputs is:

$$\mathcal{L}_{\text{track}} = \lambda_3 \mathcal{L}_{\text{focal}}(\{\hat{c}_i\}|_{\boldsymbol{\omega}(t)}, \{p_i\}) + \lambda_4 \mathcal{L}_{\text{line}}(\{\hat{V}_i\}|_{\boldsymbol{\omega}(t)}, \{V_i\}). \tag{4}$$

Transformation loss. We borrow the transformation loss L_{trans} from StreamMapNet [41] to train the PropMLP, which enforces that the query transformation in the latent space maintains the vector geometry and class type. Full details are provided in the Appendix. The final training loss is

$$\mathcal{L} = \mathcal{L}_{\text{BEV}} + \mathcal{L}_{\text{track}} + \lambda_5 \mathcal{L}_{\text{trans}}. \tag{5}$$

Training details. For each training sample, we randomly choose 4 out of the previous 10 frames to compose a training clip with a length of 5. We freeze the image backbone for the first four training frames to reduce the memory cost for the clip-based training. The training of the system has three stages: 1) Pre-train the image backbone and BEV encoder with only $\mathcal{L}_{\text{BEV}}$; 2) Warm up the vector

decoder while freezing all other parameters with $\mathcal{L}$, where the vector memory is turned on after 500 warmup iterations; 3) Jointly train all parameters with $\mathcal{L}$. The second stage warms up the vector module with a large batch size to facilitate initial convergence, as we cannot afford in the joint training. The loss weights are $\lambda_1 = 10.0$, $\lambda_2 = 1.0$, $\lambda_3 = 5.0$, $\lambda_4 = 50.0$, $\lambda_5 = 0.1$. We use an AdamW [25] optimizer with an initial learning rate 5e-4 and the weight decay is set to 0.01. A cosine learning rate scheduler is used with a final learning rate of 1.5e-6.

4 Consistent Vector HD Mapping Benchmarks

The section makes existing HD mapping benchmarks consistency-aware by 1) Improving pre-processing to generate temporally consistent ground truth with "track" labels (§4.1); and 2) Augmenting the standard mAP metric with consistency checks (§4.2).

4.1 Consistent ground truth

MapTR [18,19] created vector HD mapping benchmark from nuScenes and Agroverse2 datasets, adopted by many follow-ups [5,19,30,43,46]. However, pedestrian crossings are merged naively and inconsistent across frames. Divider lines are also inconsistent (for Argoverse2) with the failures of its graph tracing process.

StreamMapNet [41] inherited code from VectorMapNet [24] and created a benchmark with better ground truth, which has been used in the workshop challenge [1]. However, there are still issues. For Argoverse2, divider lines are sometimes split into shorter segments. For nuScenes, large pedestrian crossings sometimes split out small loops, whose inconsistencies arise randomly per frame, leading to temporarily inconsistent representations. We provide visualizations for the issues of existing benchmarks in the Appendix.

We improve processing code from existing benchmarks to (1) enhance per-frame ground-truth geometries, then (2) compute their correspondences across frames, forming ground-truth "tracks".

(1) Enhancing per-frame geometries. We inherit and improve the MapTR codebase, which has been popular in the community, while making two changes: Replace the pedestrian-zone processing with the one in StreamMapNet and further improve the quality by more geometric constraints; and Enforce temporal consistency in the divider processing by augmenting the graph tracing algorithm to handle noises of raw annotations (only for Argoverse2).

(2) Forming tracks. Given per-frame road element geometries, we solve an optimal bipartite matching problem between every pair of adjacent frames to establish correspondences of road elements. Pairwise correspondences are chained to form tracks of road elements. The matching score between a pair of road elements is defined as follows. A road-element geometry is either a polygonal curve or a loop. We transform an element geometry in an older frame to the newer

one based on the vehicle motion, then rasterize both curves/loops with a certain thickness into instance masks. Their intersection over union is the matching score. Please refer to Appendix for the full algorithmic details.

4.2 Consistency-aware mAP metric

The standard mean average precision (mAP) metric does not penalize temporarily inconsistent reconstructions. We match reconstructed road elements and the ground truth in each frame independently with Chamfer distance, as in the standard mAP process, then remove temporarily inconsistent matches with the following check. First, for baseline methods that do not predict tracking information, we form tracks of reconstructed road elements using the same algorithm we used to get ground-truth temporal correspondences (we also extend the algorithm to re-identify a lost element by trading off the speed; see Appendix for details). Next, let an "ancestor" be a road element that belongs to the same track in a prior frame. From the beginning of the sequence, we remove a per-frame match (of reconstructed and ground-truth elements) as temporarily inconsistent if any of their ancestors was not a match. The standard mAP is then calculated with the remaining temporarily consistent matches. See Appendix for complete algorithmic details.

5 Experiments

We build our system based on the StreamMapNet codebase, while using 8 NVIDIA RTX A5000 GPUs to train our model for 72 epochs on nuScenes (18, 6, and 48 epochs for the three stages) and 35 epochs on Argoverse2 (12, 3, and 20 epochs for the three stages). The batch sizes for the three training stages are 16, 48, and 16, respectively. The training takes roughly three days, while the inference speed is roughly 10 FPS. After explaining the datasets, the metrics, and the baseline methods, the section provides the experimental results.

Datasets. We use the nuScenes [2] and Argoverse2 [37] datasets. nuScenes dataset is annotated with 2Hz with 6 synchronized surrounding cameras. Input perspective images are of size 480×800. Argoverse2 dataset is annotated with 10 Hz, using 7 surrounding cameras. Input perspective images are of size 608×608. We follow MapTRv2 [19] and use an interval of 4 to subsample the sequences of Argoverse2. We evaluate the methods with the official dataset splits as well as the geographically non-overlapping splits proposed in StreamMapNet [41].

Metrics. We follow prior works [18,19,24,41] and use Average Precision (AP) as the main evaluation metric, where Chamfer distance is the matching criterion. The AP is averaged across three distance thresholds $\{0.5m, 1.0m, 1.5m\}$. The final mean AP (mAP) is computed by averaging the results over the three road element types: pedestrian crossing, lane-divider, and road-boundary. We provide both the original scores and the new consistency-aware augmented scores (§4.2).

Baselines. MapTRv2 [19] and StreamMapNet [41] are the main baselines due to their popularity and superior performance. We run their official codebase and

Table 1: Results on nuScenes [2]. The first column shows three different ground truth used for training and testing. "Consistent" is our temporarily consistent ground truth. The standard AP scores are reported for pedestrian crossing, lane-divider, road-boundary, and their average. C-mAP is our consistency-aware metric, which requires tracking information in the ground truth and is reported only for Consistent. $^{+}$: Numbers are from the original papers. †: Epochs for our multi-frame training.

G.T. data	Method	Backbone	Epoch	AP_p	AP_d	AP_b	mAP	C-mAP
MapTR	MapTR^{+} [18]	R50	110	56.2	59.8	60.1	58.7	-
	PivotNet^{+} [7]	SwinT	110	62.6	68.0	69.7	66.8	
	MapTRv2^{+} [19]	R50	110	68.1	68.3	69.7	68.7	
	GeMap^{+} [46]	R50	110	67.1	69.8	71.4	69.4	
StmMapNet	StreamMapNet [41]	R50	110	68.0	71.2	68.0	69.1	-
	SQD-MapNet^{+} [36]	R50	24	63.6	66.6	64.8	65.0	
	MapTracker (Ours)	R50	72†	77.3	72.4	74.2	74.7	
Consistent	MapTRv2 [19]	R50	110	69.6	68.5	70.3	69.5	50.5
	StreamMapNet [41]	R50	110	70.0	72.9	68.3	70.4	56.4
	MapTracker (Ours)	R50	72†	80.0	74.1	74.1	76.1	69.1

train the models until complete convergence. The results of recent competing methods [7,36,46] are also included for reference by copying numbers from their corresponding papers.

5.1 Quantitative evaluations

One of our contributions is the temporarily consistent ground truth (GT) over the two existing counterparts (i.e., MapTR [18, 19] and StreamMapNet [41]). Table 1 and Table 2 show the results where a system is trained and tested on one of the three GTs (shown in the first column). Since our codebase is based on StreamMapNet, we evaluate our system on the StreamMapNet GT and our temporarily consistent GT.

nuScenes results. Table 1 shows that both MapTRv2 and StreamMapNet achieve better mAP with our GT, which is expected as we fixed the inconsistencies in their original GT (explained in §4.1). StreamMapNet's improvement is slightly higher since it has temporal modeling (whereas MapTR does not) and exploits temporal consistency in the data. MapTracker significantly outperforms the competing methods, especially with our consistent GT by more than 8% and 22% in the original and the consistency-aware mAP scores. Note that MapTracker is the only system to produce explicit tracking information (i.e., correspondences of reconstructed elements across frames), which is required for the consistency-area mAP. A simple matching algorithm creates tracks for the baseline methods (See Appendix for details).

Argoverse2 results. Table 2 shows that both MapTRv2 and StreamMapNet achieve better mAP scores with our consistent GT, which has higher quality GT (for pedestrian crossings and dividers) besides being temporarily consis-

Table 2: Results on Argoverse2 [37]. $^{+}$: Numbers are from the original papers. †: Epochs for our multi-frame training.

G.T. data	Method	Backbone	Epoch	AP_p	AP_d	AP_b	mAP	C-mAP
MapTR	MapTRv2^{+} [19]	R50	6*4	62.9	72.1	67.1	67.4	-
	GeMap^{+} [46]	R50	24*4	69.2	75.7	70.5	71.8	
StmMapNet	StreamMapNet^{+} [41]	R50	30	62.0	59.5	63.0	61.5	-
	StreamMapNet [41]	R50	72	65.0	62.2	64.9	64.0	
	SQD-MapNet^{+} [36]	R50	30	64.9	60.2	64.9	63.3	
	MapTracker (Ours)	R50	35†	74.5	66.4	73.4	71.4	
Consistent	MapTRv2 [19]	R50	24*4	68.3	75.6	68.9	70.9	56.1
	StreamMapNet [41]	R50	72	70.5	74.2	66.1	70.3	57.5
	MapTracker (Ours)	R50	35†	77.0	80.0	73.7	76.9	68.3

Table 3: Results with geographically non-overlapping data proposed in StreamMapNet [41]. Our consistent ground truth is used. †: Epochs for our multi-frame training.

Dataset	Method	Backbone	Epoch	AP_p	AP_d	AP_b	mAP	C-mAP
nuScenes [2]	StreamMapNet [41]	R50	110	31.6	28.1	40.7	33.5	22.2
	MapTracker (Ours)	R50	72†	45.9	30.0	45.1	40.3	32.5
Argoverse2 [37]	StreamMapNet [41]	R50	72	61.8	68.2	63.2	64.4	54.4
	MapTracker (Ours)	R50	35†	70.0	75.1	68.9	71.3	63.2

tent, benefiting all the methods. MapTracker outperforms all the other baselines by significant margins (*i.e.*, 11% or 8%, respectively) in all settings. The consistency-aware score (C-mAP) further demonstrates our superior consistency, showing an improvement of more than 18% over StreamMapNet.

5.2 Results with geographically non-overlapping data

Official training/testing splits of nuScenes and Agroverse2 datasets have geographical overlaps (i.e., the same roads appear in both training/testing), which allows overfitting [20]. Table 3 compares the best baseline method StreamMapNet and MapTracker, based on geographically non-overlapping splits, proposed by StreamMapNet. MapTracker consistently outperforms with significant margins, demonstrating robust cross-scene generalization. Note that the performance for nuScenes datasets degrades for both methods. Upon careful inspection, the detection of road elements is successful but the regressed coordinates have large errors, leading to low performance.

5.3 Ablation studies

Ablation studies in Table 4 demonstrate the contributions of key design elements in MapTracker. The first "baseline" entry is StreamMapNet without its temporal reasoning capabilities (i.e., without its BEV and vector streaming memories

Table 4: Ablation studies on the key design elements of MapTracker, evaluated on the nuScenes dataset with our consistent ground truth.

Method	Task	Memory			Metrics				
		Embed.	+Fusion	+Stride	AP_p	AP_d	AP_b	mAP	C-mAP
Baseline [41]	Detection	-	-	-	69.5	71.7	68.5	69.9	56.1
StmMapNet [41]	Cond. detect.	✓	-	-	70.0	72.9	68.3	70.4	56.4
MapTracker	Tracking	✓	-	-	73.8	69.2	69.4	70.8	62.4
		✓	✓	-	78.6	73.3	72.8	74.9	68.1
		✓	✓	✓	80.0	74.1	74.1	76.1	69.1

and modules). The second entry is StreamMapNet. Both methods are trained for 110 epochs till full convergence. The last three entries are the variants of MapTracker with or without the key design elements. The first variant drops the memory fusion components in the BEV/VEC modules. This variant utilizes the tracking formulation but relies on a single BEV/VEC memory to hold the past information, like the GRU embedding of StreamMapNet. The second variant adds the memory buffers and the memory fusion components but without the striding strategy, that is, using the latest 4 frames for the fusion. This variant boosts performance, demonstrating the effectiveness of our memory mechanism. The last variant adds memory striding, which makes more effective use of the memory mechanism and improves performance.

5.4 Qualitative evaluations

Figure 4 presents qualitative comparisons of MapTracker and the baseline methods on both nuScenes and Argoverse2 datasets. For better visualization, we use a simple algorithm to merge per-frame vector HD maps into a global vector HD map. Please refer to Appendix for the details of the merging algorithm and the visualization of per-frame reconstructions. MapTracker produces much more accurate and cleaner results, demonstrating superior overall quality and temporal consistency. For scenarios where the vehicle is turning or not trivially moving forward (including the two examples in Figure 1), StreamMapNet and MapTRv2 can produce unstable results, thus leading to broken and noisy merged results. This is mainly because the detection-based formulation has difficulties maintaining temporally coherent reconstructions under complex vehicle motions.

6 Conclusion

This paper introduces MapTracker, which formulates vector HD mapping as a tracking task and leverages a history of raster and vector latents to maintain temporal consistency. We employ a query propagation mechanism to associate tracked road elements across frames, and fuse a subset of memory entries selected with distance strides to enhance consistency. We also improve existing

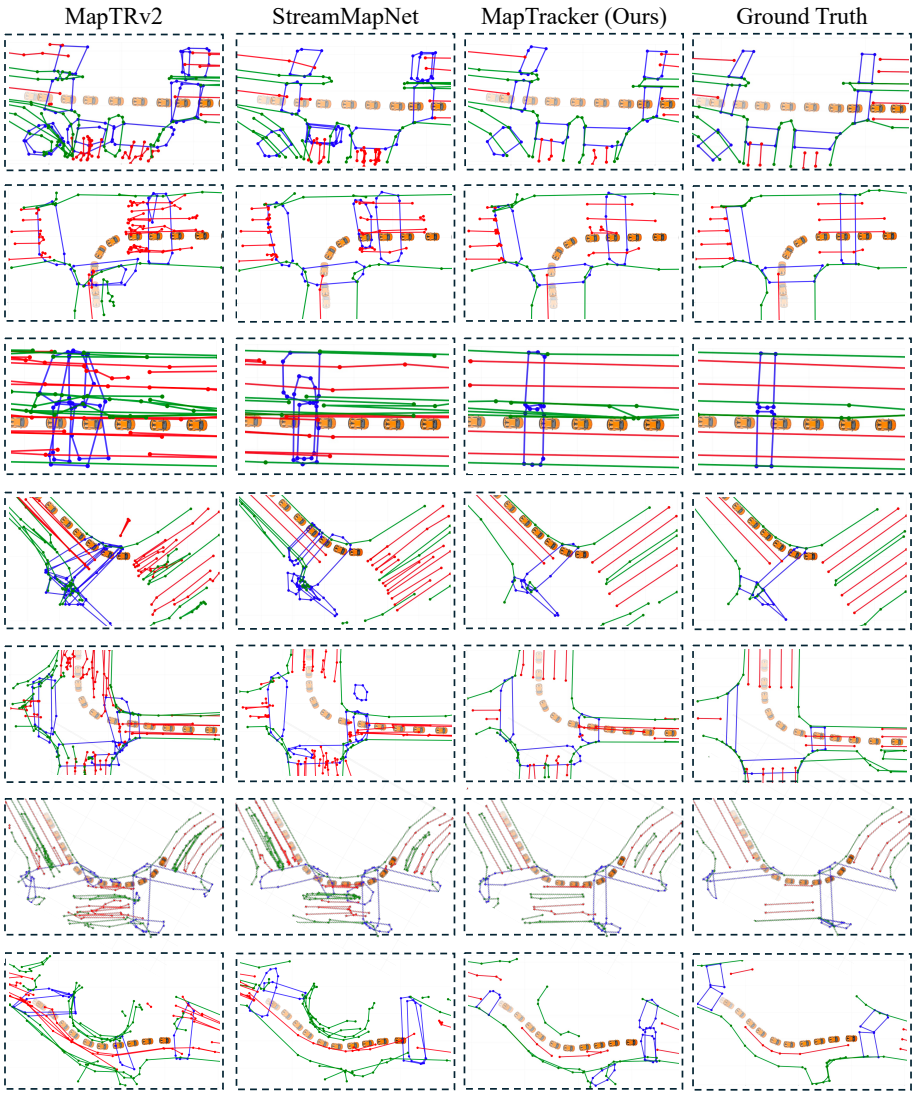

Fig. 4: Qualitative comparisons of the two representative baselines, MapTracker (Ours), and the ground truth. A simple online algorithm merges per-frame vector HD map reconstructions across a single drive-through into a global vector HD map. The top five examples are from nuScenes, while the bottom two are from Argoverse2.

benchmarks by generating consistent ground truth with tracking labels and augmenting the original mAP metric with temporal consistency checks. MapTracker significantly outperforms existing methods on nuScenes and Agroverse2 datasets when evaluated with the traditional metrics and demonstrates superior temporal consistency when evaluated with our consistency-aware metrics.

Acknowledgements. This research is partially supported by NSERC Discovery Grants, NSERC Alliance Grants, and John R. Evans Leaders Fund (JELF). We thank the Digital Research Alliance of Canada and BC DRI Group for providing computational resources.

References

1. Online hd map construction challenge for autonomous driving on cvpr 2023 workshop on end-to-end autonomous driving. `https://github.com/Tsinghua-MARS-Lab/Online-HD-Map-Construction-CVPR2023` (2023)
2. Caesar, H., Bankiti, V., Lang, A.H., Vora, S., Liong, V.E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., Beijbom, O.: nuscenes: A multimodal dataset for autonomous driving. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 11621–11631 (2020)
3. Cai, J., Xu, M., Li, W., Xiong, Y., Xia, W., Tu, Z., Soatto, S.: Memot: Multi-object tracking with memory. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8090–8100 (2022)
4. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European conference on computer vision. pp. 213–229. Springer (2020)
5. Chen, J., Deng, R., Furukawa, Y.: Polydiffuse: Polygonal shape reconstruction via guided set diffusion models. arXiv preprint arXiv:2306.01461 (2023)
6. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
7. Ding, W., Qiao, L., Qiu, X., Zhang, C.: Pivotnet: Vectorized pivot learning for end-to-end hd map construction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 3672–3682 (2023)
8. Gao, R., Wang, L.: Memotr: Long-term memory-augmented transformer for multi-object tracking. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9901–9910 (2023)
9. Gu, J., Hu, C., Zhang, T., Chen, X., Wang, Y., Wang, Y., Zhao, H.: Vip3d: End-to-end visual trajectory prediction via 3d agent queries. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 5496–5506 (2023)
10. Han, C., Sun, J., Ge, Z., Yang, J., Dong, R., Zhou, H., Mao, W., Peng, Y., Zhang, X.: Exploring recurrent long-term temporal fusion for multi-view 3d perception. arXiv preprint arXiv:2303.05970 (2023)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
12. Huang, J., Huang, G.: Bevdet4d: Exploit temporal cues in multi-camera 3d object detection. arXiv preprint arXiv:2203.17054 (2022)
13. Li, E., Casas, S., Urtasun, R.: Memoryseg: Online lidar semantic segmentation with a latent memory. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2023)
14. Li, F., Zhang, H., Liu, S., Guo, J., Ni, L.M., Zhang, L.: Dn-detr: Accelerate detr training by introducing query denoising. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13619–13627 (2022)

15. Li, H., Sima, C., Dai, J., Wang, W., Lu, L., Wang, H., Zeng, J., Li, Z., Yang, J., Deng, H., et al.: Delving into the devils of bird's-eye-view perception: A review, evaluation and recipe. IEEE Transactions on Pattern Analysis and Machine Intelligence (2023)
16. Li, Q., Wang, Y., Wang, Y., Zhao, H.: Hdmapnet: An online hd map construction and evaluation framework. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 4628–4634. IEEE (2022)
17. Li, Z., Wang, W., Li, H., Xie, E., Sima, C., Lu, T., Qiao, Y., Dai, J.: Bevformer: Learning bird's-eye-view representation from multi-camera images via spatiotemporal transformers. In: European conference on computer vision. pp. 1–18. Springer (2022)
18. Liao, B., Chen, S., Wang, X., Cheng, T., Zhang, Q., Liu, W., Huang, C.: Maptr: Structured modeling and learning for online vectorized hd map construction. arXiv preprint arXiv:2208.14437 (2022)
19. Liao, B., Chen, S., Zhang, Y., Jiang, B., Zhang, Q., Liu, W., Huang, C., Wang, X.: Maptrv2: An end-to-end framework for online vectorized hd map construction. arXiv preprint arXiv:2308.05736 (2023)
20. Lilja, A., Fu, J., Stenborg, E., Hammarstrand, L.: Localization is all you evaluate: Data leakage in online mapping datasets and how to fix it. arXiv preprint arXiv:2312.06420 (2023)
21. Lin, T.Y., Goyal, P., Girshick, R., He, K., Dollár, P.: Focal loss for dense object detection. In: Proceedings of the IEEE international conference on computer vision. pp. 2980–2988 (2017)
22. Lin, X., Lin, T., Pei, Z., Huang, L., Su, Z.: Sparse4d v2: Recurrent temporal fusion with sparse model. arXiv preprint arXiv:2305.14018 (2023)
23. Lin, X., Pei, Z., Lin, T., Huang, L., Su, Z.: Sparse4d v3: Advancing end-to-end 3d detection and tracking. arXiv preprint arXiv:2311.11722 (2023)
24. Liu, Y., Yuan, T., Wang, Y., Wang, Y., Zhao, H.: Vectormapnet: End-to-end vectorized hd map learning. In: International Conference on Machine Learning. pp. 22352–22369. PMLR (2023)
25. Loshchilov, I., Hutter, F.: Fixing weight decay regularization in adam. ArXiv **abs/1711.05101** (2017)
26. Ma, Y., Wang, T., Bai, X., Yang, H., Hou, Y., Wang, Y., Qiao, Y., Yang, R., Manocha, D., Zhu, X.: Vision-centric bev perception: A survey. arXiv preprint arXiv:2208.02797 (2022)
27. Meinhardt, T., Kirillov, A., Leal-Taixe, L., Feichtenhofer, C.: Trackformer: Multi-object tracking with transformers. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8844–8854 (2022)
28. Milletari, F., Navab, N., Ahmadi, S.A.: V-net: Fully convolutional neural networks for volumetric medical image segmentation. In: 2016 fourth international conference on 3D vision (3DV). pp. 565–571. Ieee (2016)
29. Philion, J., Fidler, S.: Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3d. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16. pp. 194–210. Springer (2020)
30. Qiao, L., Ding, W., Qiu, X., Zhang, C.: End-to-end vectorized hd-map construction with piecewise bezier curve. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13218–13228 (2023)
31. Qiao, L., Zheng, Y., Zhang, P., Ding, W., Qiu, X., Wei, X., Zhang, C.: Machmap: End-to-end vectorized solution for compact hd-map construction. arXiv preprint arXiv:2306.10301 (2023)

32. Shan, T., Englot, B.: Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 4758–4765. IEEE (2018)
33. Shan, T., Englot, B., Meyers, D., Wang, W., Ratti, C., Rus, D.: Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In: 2020 IEEE/RSJ international conference on intelligent robots and systems (IROS). pp. 5135–5142. IEEE (2020)
34. Sun, P., Cao, J., Jiang, Y., Zhang, R., Xie, E., Yuan, Z., Wang, C., Luo, P.: Transtrack: Multiple object tracking with transformer. arXiv preprint arXiv:2012.15460 (2020)
35. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
36. Wang, S., Jia, F., Liu, Y., Zhao, Y., Chen, Z., Wang, T., Zhang, C., Zhang, X., Zhao, F.: Stream query denoising for vectorized hd map construction. arXiv preprint arXiv:2401.09112 (2024)
37. Wilson, B., Qi, W., Agarwal, T., Lambert, J., Singh, J., Khandelwal, S., Pan, B., Kumar, R., Hartnett, A., Pontes, J.K., et al.: Argoverse 2: Next generation datasets for self-driving perception and forecasting. arXiv preprint arXiv:2301.00493 (2023)
38. Xu, Z., Wong, K.K., Zhao, H.: Insightmapper: A closer look at inner-instance information for vectorized high-definition mapping. arXiv preprint arXiv:2308.08543 (2023)
39. Yang, C., Chen, Y., Tian, H., Tao, C., Zhu, X., Zhang, Z., Huang, G., Li, H., Qiao, Y., Lu, L., et al.: Bevformer v2: Adapting modern image backbones to bird's-eye-view recognition via perspective supervision. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 17830–17839 (2023)
40. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. Acm computing surveys (CSUR) **38**(4), 13–es (2006)
41. Yuan, T., Liu, Y., Wang, Y., Wang, Y., Zhao, H.: Streammapnet: Streaming mapping network for vectorized online hd map construction. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 7356–7365 (2024)
42. Zeng, F., Dong, B., Zhang, Y., Wang, T., Zhang, X., Wei, Y.: Motr: End-to-end multiple-object tracking with transformer. In: European Conference on Computer Vision. pp. 659–675. Springer (2022)
43. Zhang, G., Lin, J., Wu, S., Song, Y., Luo, Z., Xue, Y., Lu, S., Wang, Z.: Online map vectorization for autonomous driving: A rasterization perspective. arXiv preprint arXiv:2306.10502 (2023)
44. Zhang, J., Singh, S.: Loam: Lidar odometry and mapping in real-time. In: Robotics: Science and systems (2014)
45. Zhang, Y., Wang, T., Zhang, X.: Motrv2: Bootstrapping end-to-end multi-object tracking by pretrained object detectors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 22056–22065 (2023)
46. Zhang, Z., Zhang, Y., Ding, X., Jin, F., Yue, X.: Online vectorized hd map construction using geometry. arXiv preprint arXiv:2312.03341 (2023)
47. Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J.: Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159 (2020)