FROM FRAGMENTS TO GLOBAL MAPS: LEARNING VECTORIZED MAP AGGREGATION WITH LARGE LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large-scale vectorized HD maps provide structured road information that is essential for perception, localization, and planning in autonomous driving. Constructing such maps requires aggregating noisy, fragmented, and overlapping local predictions collected along a vehicle trajectory into a coherent global map. Existing aggregation methods typically rely on hand-crafted rules for fragment association and refinement. However, a fixed set of thresholds cannot effectively handle variations in road structures and prediction errors, often requiring detector-specific tuning or manual adjustment. To address this limitation, we propose MapMergeLLM, a data-driven framework that formulates vectorized map aggregation as conditional sequence generation with a large language model. Given serialized local vectorized maps, our model directly predicts the aggregated global map polylines. To reduce dependence on any particular upstream detector, we train the model on synthetic local maps generated from clean vector maps using corruptions that simulate representative prediction errors. We further introduce a coordinate tokenizer with geometry-aware pretraining to precisely represent map coordinates. In addition, we propose a line-level association loss that explicitly supervises correspondences between local observations of the same map element. Experiments on Argoverse2 and nuScenes using multiple recent upstream detectors demonstrate that MapMergeLLM substantially outperforms heuristic and optimization-based aggregation baselines without detector-specific retraining.

1 INTRODUCTION

High-definition (HD) maps provide detailed information about road structures to support autonomous driving systems (Li et al., 2022; Liao et al., 2024). In vectorized HD maps, line-shaped elements such as lane dividers and road boundaries are represented as polylines, with each polyline represented by an ordered sequence of points. Recent advances in vectorized map construction enable models to predict these map elements directly from sensor observations (Liu et al., 2023; Qiao et al., 2023a; Yuan et al., 2024; Ding et al., 2023). As onboard sensors only capture a limited region around the ego vehicle, a large-scale HD map cannot be constructed from a single observation, and instead, it must be built from local vectorized predictions collected along a driving sequence.

However, simply accumulating these predictions results in redundant, fragmented, and geometrically inconsistent map elements, as the same road structures are repeatedly observed across overlapping local regions. We refer to the process of associating, merging, and refining these local predictions into a coherent global map as *vectorized map aggregation*. This task is challenging because errors in local map predictions, together with overlapping observations, make it difficult to correctly associate fragments of the same map structure and accurately recover its underlying geometry.

Despite its importance, only a few studies have addressed this problem. Existing methods typically rely on hand-crafted geometric rules with fixed parameters, such as distance thresholds, overlap criteria, or sampling intervals (Chen et al., 2023; Qiao et al., 2023b), which may not generalize well across diverse road structures or prediction errors from different upstream detectors. For example, in scenes with dense parallel lanes, noisy observations of one lane may be difficult to distinguish from observations of neighboring lanes. Consider a distance-based association threshold τ, where

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

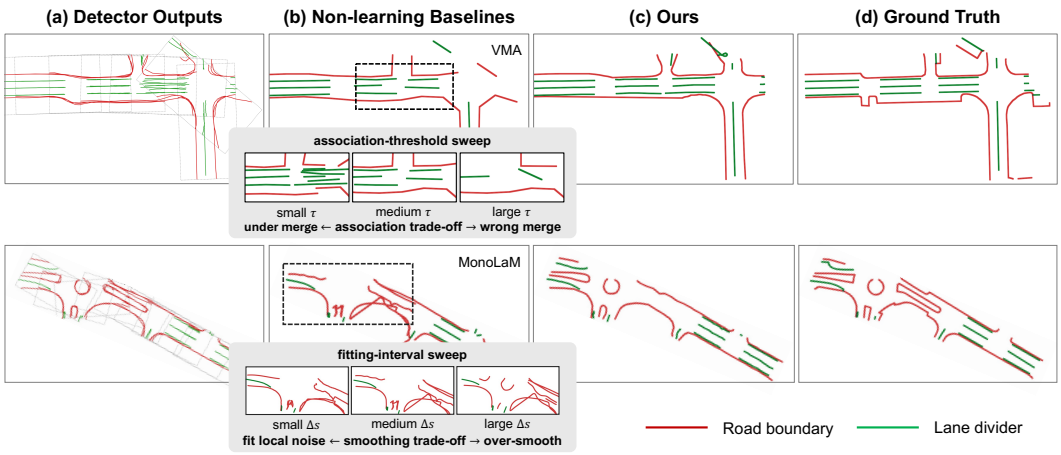


Figure 1: **Non-learning-based aggregation presents association and smoothing trade-offs.** (a) StreamMapNet (Yuan et al., 2024) predictions accumulated across frames. (b) For the heuristic-based method, a small association threshold (τ) leaves related fragments unmerged, whereas a large threshold incorrectly merges neighboring lanes. For the geometry-based method, a small fitting interval (Δs) fits local prediction noise, whereas a large interval over-smooths line structures. (c) MapMergeLLM learns aggregation from data and produces maps close to the (d) ground truth without hand-crafted parameter tuning.

two fragments are considered for merging if their geometric discrepancy falls below τ. As shown in the top zoomed-in views of Fig. 1(b), a larger τ may incorrectly merge neighboring lanes, whereas a smaller threshold better preserves their distinction but leaves related fragments of the same lane unmerged. Geometric refinement presents a similar trade-off. As shown in the bottom zoomed-in views of Fig. 1(b), geometry-based methods (Cheng et al., 2021; Qiao et al., 2023b) often rely on a fixed fitting interval Δs that controls how closely the refined line follows local observations. A smaller Δs allows the line to capture finer geometric variations but also makes it more sensitive to prediction noise, whereas a larger Δs provides stronger smoothing but may drop important geometric details. In the illustrated example, a larger Δs better recovers the circular curb but over-smooths other curb structures. These examples show that no single parameter setting is suitable for all road geometry and detection error patterns. The appropriate association threshold or fitting interval depends on the local context, such as lane spacing, curvature, and detector noise patterns, and may vary even within the same scene. This motivates us to learn the aggregation process from data, enabling fragment association and geometric refinement to adapt to the observations rather than being controlled by globally fixed hand-crafted rules.

Therefore, we propose **MapMergeLLM**, which formulates vectorized map aggregation as a conditional sequence generation problem and uses a large language model (LLM) (Minaee et al., 2024; Qwen Team, 2026) as the sequence-modeling backbone. Given local vectorized maps from multiple frames, the model directly generates the aggregated global polylines from serialized local fragments. However, training such a model presents two key challenges. First, the model should generalize across different upstream detectors rather than overfit to the error distribution of a particular one. To achieve this, we construct synthetic training data by applying structured corruptions to clean vector maps, simulating common map detection errors such as duplication, fragmentation, and geometric perturbations. Second, as LLMs were originally designed for text generation, adapting them to map generation requires a precise representation of map coordinates. Unlike vanilla tokenization, which may split a numerical value into multiple subword units, we encode each coordinate value as a single token, allowing the model to predict a complete value in a single step. To improve map generation, we propose geometry-aware pretraining that teaches the model numerical and spatial relationships among the new coordinate tokens. We further introduce a line-level association loss that explicitly supervises correspondences between local fragments, helping the model associate observations of the same map element and reduce duplication. Together, these components enable accurate and reliable global map aggregation across different upstream detectors.

To demonstrate the effectiveness of MapMergeLLM, we evaluate it on local maps produced by recent vectorized map detectors on the Argoverse2 (Wilson et al., 2021) and nuScenes (Caesar et al.,

2020) datasets. The experimental results show that MapMergeLLM consistently outperforms heuristic and optimization-based aggregation baselines, particularly on road boundaries, where it achieves mean relative F1 improvements of 23.1% on Argoverse2 and 21.9% on nuScenes over the strongest baseline. In summary, our main contributions include:

- We identify key limitations of existing heuristic and geometry-based map aggregation methods, showing that fixed hand-crafted merging rules struggle to generalize across diverse road structures and detector noise patterns.
- We propose MapMergeLLM, an LLM-based, data-driven framework for vectorized map aggregation, and introduce coordinate-token pretraining and line-level association loss to improve map generation, supported by detector-agnostic synthetic training data.
- We conduct comprehensive experiments on Argoverse2 and nuScenes, demonstrating that MapMergeLLM substantially improves over existing aggregation baselines across multiple recent upstream map detectors.

2 Related Work

Vectorized HD Map Construction and Aggregation. Online vectorized HD map construction aims to recover structured map elements, such as lane dividers and road boundaries, within a local region surrounding the ego vehicle from onboard sensor observations (Li et al., 2022; Liu et al., 2023). Early approaches, such as HDMapNet (Li et al., 2022), first predict map elements in the BEV space and then vectorize them, while more recent methods directly predict vectorized maps (Liu et al., 2023; Liao et al., 2023; 2024). Building on this direction, temporal methods further leverage historical observations (Yuan et al., 2024; Chen et al., 2024) or a continuously updated global map prior (Shi et al., 2024) to improve prediction consistency. Beyond local map prediction, several works focus on aggregating local vectorized predictions into global maps. VMA (Chen et al., 2023) associates and merges map elements using predefined geometric rules, while MonoLaM (Qiao et al., 2023b) combines geometric association with graph optimization. GNMap (Fan et al., 2024) introduces a learning-based fusion framework, but it rasterizes vectorized map tiles before fusion and outputs a semantic map rather than the global polylines required in our task. These aggregation approaches either rely on hand-crafted geometric rules or involve rasterization. In contrast, we propose a data-driven framework that learns to aggregate fragmented observations directly in the vector domain through conditional sequence generation.

Language Models for Map Generation. The sequence modeling capabilities of LLMs have recently been explored for structured geometric prediction, including lane and map generation. Zhang et al. (2026) employs an MLLM with lane-specific tokens to predict lane geometry from visual observations. Yuan et al. (2026) formulates vectorized map construction as autoregressive generation, progressively producing map elements from multi-modal inputs and previously constructed map states. Liang et al. (2026) further extends this paradigm to persistent mapping by jointly modeling lane geometry, traffic rules, and temporal observations. Another line of work explores the use of vision-language reasoning for map refinement. Xia et al. (2026) uses such reasoning to identify and correct errors in predicted lane-level maps. While these studies demonstrate the potential of language models for structured map generation and refinement, their application to aggregating multiple vectorized map observations remains largely unexplored.

3 Method

We propose MapMergeLLM, a data-driven framework that formulates vectorized map aggregation as a sequence-to-sequence generation task (Section 3.1). To avoid depending on detector-specific error patterns, we construct synthetic training pairs by applying structured corruptions to clean vector maps (Section 3.2). We further introduce coordinate tokenization and a two-stage training scheme (Section 3.3), including geometry-aware pretraining followed by map aggregation fine-tuning. A line-level association loss complements token-level supervision by encouraging association of related line fragments and reducing duplicate map predictions. At inference time, the model takes serialized local predictions in a common coordinate frame as input and generates the global map polylines. An overview of our LLM-based map aggregation framework is shown in Fig. 2.

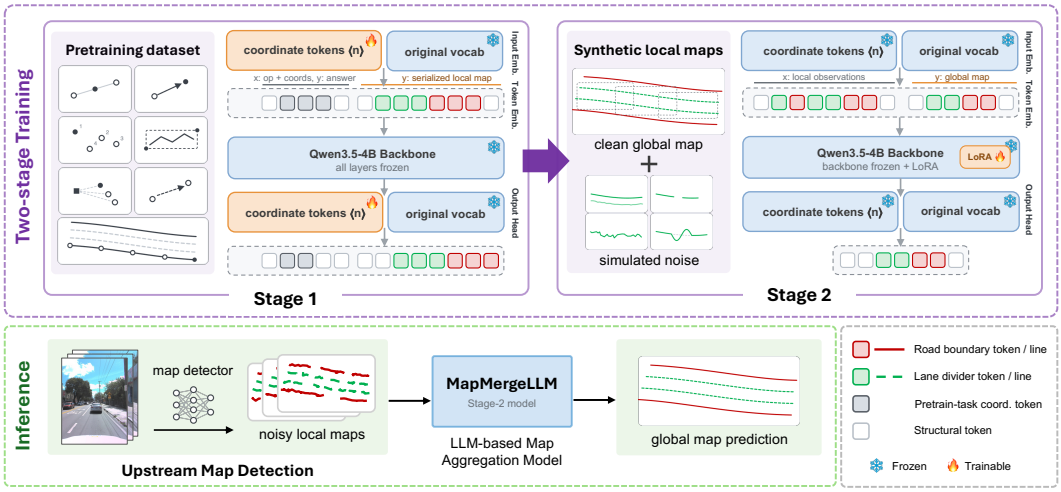


Figure 2: **Overview of MapMergeLLM**. Training consists of two stages: Stage 1 performs coordinate-token pretraining to learn representations for the newly introduced coordinate tokens, and Stage 2 performs LoRA-based fine-tuning for map aggregation using synthetic noisy local maps. At inference time, the trained model aggregates local predictions from an upstream detector into a global vectorized map.

3.1 Problem Formulation

Vectorized Map Definition. We represent each map element by a category c_k and an ordered polyline $\mathbf{P}_k = (\mathbf{p}_{k,1}, \ldots, \mathbf{p}_{k,N_k})$, where $\mathbf{p}_{k,i} = (x_{k,i}, y_{k,i}) \in \mathbb{R}^2$. A vectorized map with K elements is represented as $\mathcal{M} = [(\mathbf{P}_1, c_1), \ldots, (\mathbf{P}_K, c_K)]$, where both K and the number of sampled points N_k may vary across maps.

Global Map Aggregation. Let $\mathcal{M}^*$ denote the clean global map over the region covered by a sequence of local observations, and let $\mathcal{C}_t(\mathcal{M}^*)$ denote the local submap captured at time t. We assume that all local predictions can be accurately aligned using ego poses and focus on element-level prediction errors rather than pose-induced misalignment. Given L local vectorized observations $\widetilde{\mathcal{M}}^{(1)}, \ldots, \widetilde{\mathcal{M}}^{(L)}$, we formulate map aggregation as

$$\widehat{\mathcal{M}} = F_\theta \left(\widetilde{\mathcal{M}}^{(1)}, \ldots, \widetilde{\mathcal{M}}^{(L)} \right), \tag{1}$$

where F_θ aggregates redundant and complementary geometric evidence across overlapping observations to recover the global clean map $\mathcal{M}^*$.

Different upstream detectors may produce different error patterns due to differences in model architecture and sensing modality. Rather than training F_θ on predictions from a particular detector, we synthesize local observations from a detector-agnostic corruption distribution,

$$\widetilde{\mathcal{M}}^{(t)} \sim q_\phi \left(\cdot \mid \mathcal{C}_t(\mathcal{M}^*) \right), \tag{2}$$

where q_ϕ captures representative prediction errors shared across detectors. At inference time, F_θ is applied directly to local predictions from different upstream detectors. The construction of q_ϕ is described in the following section.

3.2 Detector-Agnostic Corruption Process

We design the detector-agnostic corruption distribution q_ϕ to model *recoverable* prediction errors, *i.e.*, errors that can be corrected by aggregating vectorized local observations without access to the original sensor data. Although individual local predictions may be fragmented, duplicated, or geometrically distorted, the overlapping observations should collectively retain sufficient evidence to recover the underlying map element. Therefore, we focus on four representative failure modes: instance duplication, endpoint corruption and fragmentation, unstructured point-wise perturbation, and structured local geometric perturbation.

Let T_{dup}, T_{frag}, T_{jit}, and T_{str} denote the four polyline-level corruption operators. Each operator acts on a single polyline and may retain, replace, split, or duplicate the corresponding instance. We lift each operator T to a map-level operator $\mathcal{T}$ by independently applying it to each polyline with probability π_T, while leaving unselected polylines unchanged. If an operator produces multiple instances, all resulting polylines are retained in the map collection. We then apply the four map-level operators sequentially to generate a synthetic noisy local map as:

$$\widetilde{\mathcal{M}}^{(t)} = \mathcal{T}_{\mathrm{str}} \circ \mathcal{T}_{\mathrm{jit}} \circ \mathcal{T}_{\mathrm{frag}} \circ \mathcal{T}_{\mathrm{dup}} \left(\mathcal{C}_t(\mathcal{M}^*) \right). \tag{3}$$

The operators are applied sequentially, allowing multiple failure modes to coexist within a single local map. The parameters of each operator are sampled from predefined ranges, and the resulting composition defines the corruption distribution q_ϕ.

Instance Duplication. Duplicate detections occur when multiple predicted instances correspond to the same underlying map element. For a selected polyline $\mathbf{P}$, we retain the original instance and generate an additional copy $\mathbf{P}^{\mathrm{dup}}$. The duplicate is resampled along arc length with a randomly varied sampling density, and a subset of its points is perturbed with small zero-mean Gaussian noise to simulate slight geometric differences between duplicate predictions. The duplication operator is defined as

$$T_{\mathrm{dup}}(\mathbf{P}) = [\mathbf{P}, \mathbf{P}^{\mathrm{dup}}]. \tag{4}$$

Endpoint Corruption and Fragmentation. Local map predictions may contain truncated endpoints or fragmented map elements. For a selected polyline $\mathbf{P}$, we first perform arc-length resampling and randomly choose an endpoint $e \in \{\mathrm{head}, \mathrm{tail}\}$ and a corruption mode $m \in \{\mathrm{trunc}, \mathrm{split}\}$. The corruption operator is defined as

$$T_{\mathrm{frag}}(\mathbf{P}; e, m) = \mathcal{O}_m(\mathbf{P}, e), \tag{5}$$

where $\mathcal{O}_{\mathrm{trunc}}$ removes a short portion near the selected endpoint, while $\mathcal{O}_{\mathrm{split}}$ partitions the polyline near that endpoint into two fragments. Both operations preserve most of the underlying geometry while simulating incomplete and fragmented predictions.

Unstructured Point-wise Perturbation. Even when a map element is correctly detected, its predicted geometry may contain sparse local deviations due to sensor noise or prediction uncertainty. For a selected polyline $\mathbf{P}$, we randomly select a subset of sampled points and independently perturb their coordinates with zero-mean Gaussian noise:

$$\mathbf{p}_i^{\mathrm{jit}} = \mathbf{p}_i + \boldsymbol{\epsilon}_i, \qquad \boldsymbol{\epsilon}_i \sim \mathcal{N}\left(\mathbf{0}, \sigma_{\mathrm{jit}}^2 \mathbf{I}\right), \tag{6}$$

while leaving all other points unchanged. We denote the resulting perturbed polyline as $\mathbf{P}^{\mathrm{jit}} = T_{\mathrm{jit}}(\mathbf{P})$. This operator simulates spatially uncorrelated point-wise errors without introducing coherent deformation of the polyline.

Structured Local Geometric Perturbation. In addition to independent point-wise errors, local map predictions may present smooth, spatially correlated geometric deviations. For example, in regions with closely spaced or complex road markings, a detector may be influenced by nearby lane markings or road boundaries, producing smooth local deviations in the predicted geometry. To simulate such errors, we select a local interval along a polyline and apply a sinusoidal displacement along its local normal direction:

$$\mathbf{p}_i^{\mathrm{str}} = \mathbf{p}_i + \mathbb{I}[s_i \in [s_a, s_b]] A \sin\left(\frac{2\pi(s_i - s_a)}{\lambda}\right) \mathbf{n}_i, \tag{7}$$

where s_i denotes the cumulative arc-length position, $\mathbf{n}_i$ is the local unit normal, A controls the displacement amplitude, and λ denotes the wavelength. The corrupted interval $[s_a, s_b]$ is centered along the polyline, with length $s_b - s_a = n_h \lambda / 2$, where n_h is an integer specifying the number of sinusoidal half-periods. This construction ensures zero displacement at both interval boundaries. We denote the resulting polyline as $\mathbf{P}^{\mathrm{str}} = T_{\mathrm{str}}(\mathbf{P})$.

3.3 LLM-BASED MAP AGGREGATION

Model Architecture. Given the local vectorized observations defined above, we serialize each map as a sequence of map elements, where each element is represented by its category and the ordered coordinates of its polyline. Standard subword tokenizers, such as byte-pair encoding (BPE) (Sennrich et al., 2016), typically construct tokens based on how frequently character sequences appear in text, without explicitly considering their numerical meaning. As a result, a single coordinate value may be split into multiple subword units (Kreitner et al., 2026; Singh & Strouse, 2024). For map aggregation, the model must compare coordinates across observations and generate corrected coordinates for polyline merging and refinement. Such fragmentation makes precise coordinate generation more difficult, as every token of a coordinate value must be predicted correctly. An error in any token can substantially change the resulting map position.

Therefore, we use *coordinate tokenization*, which encodes each discretized coordinate value as a single new token. Given a point $p_i = (x_i, y_i)$ and a spatial resolution δ in meters, we quantize each coordinate to an integer:

$$\tilde{z}_i = \text{round}\left(\frac{z_i}{\delta}\right), \qquad z_i \in \{x_i, y_i\}. \tag{8}$$

For example, a coordinate value of 12.34 m is mapped to 123 at $\delta = 0.1$ m. We then extend the vocabulary with a new token $\langle n \rangle$ for each integer value within the supported coordinate range:

$$\mathcal{V}_{\text{coord}} = \{\langle n \rangle \mid n \in \mathbb{Z},\ \tilde{z}_{\min} \le n \le \tilde{z}_{\max}\}. \tag{9}$$

During decoding, each predicted coordinate token $\langle n \rangle$ is converted back to meters as $z = n\delta$.

MapMergeLLM employs Qwen3.5-4B (Qwen Team, 2026) as its backbone. We add the coordinate tokens to its vocabulary and expand the input embedding matrix and language modeling (LM) head with corresponding token-specific rows. For parameter-efficient adaptation, we apply LoRA (Hu et al., 2022) to the selected linear layers of the backbone. The training procedure consists of two stages, as described below.

Stage 1: Coordinate-Token Pretraining. Although each new coordinate token represents a complete coordinate value, the model still needs to learn its numerical meaning and how to use it in map sequences. Map aggregation requires comparing coordinates across observations and adjusting them to recover the underlying geometry. Therefore, we propose a *coordinate-token pretraining* scheme that uses geometric and arithmetic tasks, such as 2D coordinate translation, midpoint prediction, and point sorting, to help the model learn how to compute and compare coordinates before the model is trained for map aggregation. We also include next-token prediction on serialized local maps to help the model learn how to use the new tokens within polyline sequences. Details of the pretraining tasks are provided in Appendix A.2.

During this stage, we optimize only the newly added coordinate-token rows in the input embedding matrix and LM head, while keeping all original pretrained parameters frozen. Let θ_0 denote all frozen pretrained model parameters and $\mathbf{E}_c$ denote the newly added coordinate-token rows in the input embedding matrix and LM head. We optimize $\mathbf{E}_c$ using the following autoregressive cross-entropy loss:

$$\mathcal{L}_{\text{pre}}(\mathbf{E}_c) = -\,\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_{\text{pre}}}\left[\frac{1}{|\mathbf{y}|}\sum_{j=1}^{|\mathbf{y}|}\log p_{\theta_0,\mathbf{E}_c}\left(y_j \mid \mathbf{y}_{<j}, \mathbf{x}\right)\right]. \tag{10}$$

Here, $\mathcal{D}_{\text{pre}}$ represents the pretraining dataset. For geometric and arithmetic tasks, $\mathbf{x}$ specifies the operation and input coordinates, and $\mathbf{y}$ is the corresponding answer, with the loss applied only to the answer tokens. For map sequence prediction, we train the model to predict each token in a serialized local map given the previous tokens, and the loss is computed over all tokens.

Stage 2: Map Aggregation Training. In Stage 2, we adapt the model for map aggregation by only optimizing the LoRA parameters, while keeping the pretrained model parameters θ_0 and the coordinate-token parameters $\mathbf{E}_c$ learned in Stage 1 frozen, as shown in Fig. 2. Each training pair consists of the synthetic noisy local observations $\mathbf{x}$, generated by the corruption process introduced in Section 3.2, and the clean global map $\mathbf{y}$ as the target. We supervise autoregressive map generation

with a token-level cross-entropy loss over the target sequence:

$$\mathcal{L}_{\text{agg}}(\Delta) = -\,\mathbb{E}_{(\mathbf{x},\mathbf{y})\sim\mathcal{D}_{\text{agg}}}\left[\frac{1}{|\mathbf{y}|}\sum_{j=1}^{|\mathbf{y}|}\log p_{\theta_0,\Delta,\mathbf{E}_c}\left(y_j \mid \mathbf{y}_{<j}, \mathbf{x}\right)\right], \tag{11}$$

where $\mathcal{D}_{\text{agg}}$ denotes the map aggregation training dataset, and Δ represents the trainable LoRA parameters. Under teacher forcing, each target token is predicted conditioned on the local observations and the ground-truth prefix $\mathbf{y}_{<j}$.

Line-Level Association Loss. When combining two local maps, a natural first step is to determine which line fragments belong to the same underlying map element and should be merged together. Token-level cross-entropy supervises the generated map sequence but does not explicitly supervise these fragment-level correspondences. Therefore, we introduce a *line-level association loss* to encourage fragments of the same map element to share similar representations while separating the unrelated ones, facilitating both fragment association and deduplication. We formulate this objective as a contrastive loss. For each input polyline, we construct a line embedding by mean-pooling the final-layer hidden states of its coordinate tokens:

$$\mathbf{h}_i = \frac{1}{|\mathcal{J}_i|}\sum_{j\in\mathcal{J}_i}\mathbf{h}_j^{\text{token}}, \tag{12}$$

where $\mathcal{J}_i$ denotes the set of coordinate-token positions belonging to the i-th local polyline, excluding structural tokens such as commas and parentheses.

Each polyline i is associated with a category c_i, a ground-truth global element ID g_i, and a frame index t_i. We define positive samples as observations of the same global element from adjacent local frames: $\mathcal{P}(i) = \{k \neq i \mid (c_k, g_k) = (c_i, g_i),\ |t_k - t_i| = 1\}$, and negative samples as observations of different global elements: $\mathcal{N}(i) = \{k \mid (c_k, g_k) \neq (c_i, g_i)\}$. The line-level contrastive objective is defined as:

$$\mathcal{L}_{\text{line}} = -\frac{1}{|\mathcal{A}|}\sum_{i\in\mathcal{A}}\frac{1}{|\mathcal{P}(i)|}\sum_{p\in\mathcal{P}(i)}\log\frac{\exp(\mathbf{z}_i^\top\mathbf{z}_p/\tau)}{\sum_{k\in\mathcal{P}(i)\cup\mathcal{N}(i)}\exp(\mathbf{z}_i^\top\mathbf{z}_k/\tau)}, \tag{13}$$

where $\mathcal{A} = \{i \mid \mathcal{P}(i) \neq \emptyset\}$ is the index set of input polylines with at least one positive sample, $\mathbf{z}_i = \mathbf{h}_i/\|\mathbf{h}_i\|_2$, and τ is the temperature parameter. The training objective of Stage 2 combines both the token-level generation loss and the line-level association loss:

$$\mathcal{L} = \mathcal{L}_{\text{agg}} + \lambda_{\text{line}}\mathcal{L}_{\text{line}}, \tag{14}$$

where λ_{line} is a weighting coefficient to balance the two objectives.

Inference. At inference time, local map predictions from an upstream detector are transformed into a common coordinate frame and serialized using the proposed coordinate tokenization. Conditioned on the serialized observations, the model autoregressively generates the global map sequence. The generated sequence is then parsed into polylines, with coordinate tokens converted back to coordinate values at resolution δ.

4 Experiments

4.1 Experimental Setup

Datasets. We evaluate MapMergeLLM on the Argoverse2 (Wilson et al., 2021) and nuScenes (Caesar et al., 2020) datasets, focusing on lane dividers and road boundaries. To construct training pairs, we first crop local maps from the ground-truth vectorized maps at intervals of 15m along the ego trajectory, with each crop covering a $60\text{m} \times 30\text{m}$ region centered at the corresponding ego position. We then apply the synthetic corruption process described in Section 3.2 to these local maps to generate the noisy input observations. The corresponding target global map is defined over the union of these local regions. At inference time, we obtain local map predictions using the official checkpoints of MapTracker (Chen et al., 2024) and StreamMapNet (Yuan et al., 2024). We first transform these predictions from their respective ego coordinate frames into a common global coordinate frame and then use the aligned predictions as input to our model for aggregation.

Baselines. We compare MapMergeLLM with two representative map aggregation baselines: (1) VMA (Chen et al., 2023) incrementally merges local vectorized maps using predefined association and merging rules, such as checking whether two polylines sufficiently overlap or whether the endpoints of each polyline are close to the other. (2) MonoLaM (Qiao et al., 2023b) represents line elements using Catmull-Rom splines. It associates local observations with existing map elements based on geometric criteria and incrementally updates the map through graph optimization. We use the default settings of MonoLaM and adjust the parameters of VMA to better preserve the geometry of the predicted lines.

Evaluation Metrics. We adopt the line-level matching criterion from MonoLaM to evaluate aggregated global maps over complete driving scenes. Following MonoLaM, a predicted polyline is considered a match to a ground-truth polyline if more than 75% of the points from the ground-truth polyline are within a distance threshold (d) of the prediction. We use greedy matching, where each predicted polyline can be counted as a true positive at most once. We report precision, recall, and F1 scores separately for lane dividers and road boundaries. For each distance threshold, these metrics are computed from counts accumulated over all test scenes. The final results are averaged across $d \in \{0.5\text{m}, 1.0\text{m}, 1.5\text{m}\}$.

Implementation and training details are provided in Appendix A.1.

4.2 Results

Comparison with Map Aggregation Baselines. Table 1 compares our method with VMA and MonoLaM using local predictions from MapTracker and StreamMapNet on Argoverse2 (AV2) and nuScenes. MapMergeLLM consistently outperforms both baselines in F1 across all datasets, upstream detectors, and map categories. For instance, using StreamMapNet predictions on AV2, MapMergeLLM improves F1 scores over VMA by 35.0 percentage points for lane dividers and 29.7 percentage points for road boundaries. Compared to the strongest baseline, MonoLaM, the corresponding improvements are 5.8 and 8.6 percentage points, respectively. The improvement is particularly notable for road boundaries, which often have more complex shapes, including sharp curves and circular structures. Their local predictions can also contain fragmented, distorted, or duplicated segments. These variations make it difficult to apply the fixed association and smoothing rules across different cases. By learning from diverse training examples, our approach can better handle these geometric structures and prediction errors.

We further evaluate how the methods handle these complex road structures and prediction errors through qualitative comparisons in Fig. 3. In Fig. 3(a), a circular road boundary lies between several lane dividers. Both VMA and MonoLaM fail to recover the complete circular structure from the overlapping local predictions, while MapMergeLLM reconstructs the complete circular geometry. In Fig. 3(b), the highlighted road boundary is a single continuous line in the ground truth. VMA fails to merge fragments from different local predictions, leaving two disconnected segments with a large gap in between. MonoLaM produces several overlapping lines with distorted shapes. In

Table 1: **Comparison with map aggregation baselines** using per-frame predictions from MapTracker and StreamMapNet on AV2 and nuScenes datasets. Precision, recall, and F1 (%) are reported for lane dividers and road boundaries and averaged over matching thresholds $d \in \{0.5, 1.0, 1.5\}$ meters.

	Argoverse2						nuScenes					
	Lane Divider			Road Boundary			Lane Divider			Road Boundary		
Method	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
MapTracker (Chen et al., 2024)												
VMA (Chen et al., 2023)	49.6	24.7	33.0	15.0	12.5	13.6	82.7	68.6	<u>75.0</u>	24.4	21.8	23.0
MonoLaM (Qiao et al., 2023b)	52.6	48.5	<u>50.5</u>	34.0	33.6	<u>33.8</u>	69.9	80.1	74.7	38.1	35.0	<u>36.5</u>
Ours	54.4	48.7	**51.4**	42.8	41.0	**41.9**	74.7	82.1	**78.2**	48.1	43.3	**45.5**
StreamMapNet (Yuan et al., 2024)												
VMA (Chen et al., 2023)	48.9	30.3	37.4	18.9	16.3	17.5	61.2	57.1	59.1	18.8	18.6	18.7
MonoLaM (Qiao et al., 2023b)	59.0	76.5	<u>66.6</u>	37.7	39.5	<u>38.6</u>	60.7	77.3	<u>68.0</u>	32.8	33.3	<u>33.0</u>
Ours	68.9	76.4	**72.4**	46.8	47.7	**47.2**	64.5	78.7	**70.9**	40.8	37.9	**39.3**

contrast, our method recovers a single continuous boundary that more closely matches the ground truth. Finally, Fig. 3(c) shows an example with substantial geometric noise along the road boundary. VMA removes some noisy fragments but also loses parts of the true boundary, while MonoLaM overfits the prediction noise and produces redundant lines. MapMergeLLM reduces both noise and duplicate predictions while recovering a smoother boundary that is closer to the ground truth.

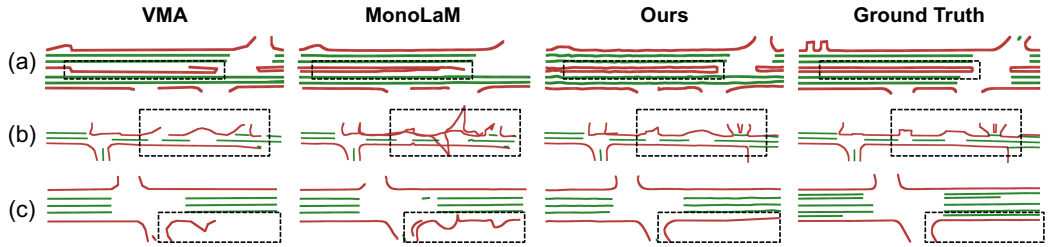


Figure 3: **Qualitative comparison with map aggregation baselines**. (a) StreamMapNet predictions on AV2, (b) StreamMapNet predictions on nuScenes, and (c) MapTracker predictions on AV2. Our MapMergeLLM preserves road structures while reducing duplicate predictions and geometric artifacts.

Ablation Studies. Table 2 studies the effects of coordinate tokenization, geometry-aware pretraining, and the line-level association loss using StreamMapNet predictions. MapMergeLLM, highlighted in blue, achieves the highest F1 on both datasets and categories. We first examine coordinate tokenization and pretraining. Compared with the vanilla tokenizer (Table 2, row 1), coordinate tokenization with pretraining improves F1 across all four settings, particularly for lane dividers of AV2. Removing pretraining results in lower F1 scores in every setting (Table 2, row 2). These results demonstrate the effectiveness of pretraining when introducing new coordinate tokens for map aggregation. Moreover, the line-level association loss further improves F1 across all four settings compared with training only using the token-level generation loss (Table 2, row 3). This suggests that explicit supervision over the line fragments helps the model associate related observations and reduce duplicate predictions.

Table 2: **Ablation study on coordinate tokenization, coordinate-token pretraining, and the line-level contrastive objective**, using StreamMapNet predictions. Our proposed training strategy is highlighted in blue.

	Argoverse2						nuScenes					
	Lane Divider			Road Boundary			Lane Divider			Road Boundary		
Setting	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
w/o Coord. Tok.	54.6	38.7	45.3	41.1	38.6	39.8	62.0	77.2	68.7	35.7	32.3	33.9
w/o Pretrain	63.4	69.3	66.2	41.8	41.2	41.5	60.8	74.9	67.1	38.8	32.8	35.5
w/o Contrastive	67.3	73.5	70.3	46.8	46.5	46.6	64.7	78.0	70.7	40.6	36.2	38.2
Ours (full)	68.9	76.4	**72.4**	46.8	47.7	**47.2**	64.5	78.7	**70.9**	40.8	37.9	**39.3**

5 Conclusion

In this paper, we introduce MapMergeLLM, a data-driven framework for vectorized map aggregation. Instead of relying on predefined association and smoothing rules, our approach formulates map aggregation as a conditional sequence generation problem given the local vectorized map observations. To reduce dependence on specific detectors, we train on synthetic noisy local maps constructed from clean vector maps, where the noise simulates representative detection errors. We further introduce coordinate tokenization with geometry-aware coordinate-token pretraining to better represent map elements, as well as a line-level association loss to improve association of map fragments. Experiments on Argoverse2 and nuScenes demonstrate that MapMergeLLM consistently improves over heuristic and optimization-based aggregation baselines across multiple recent upstream detectors. Overall, our work highlights data-driven map aggregation as a powerful direction for reducing dependence on heuristic merging rules and improving the robustness of large-scale vectorized map construction.

AI USE STATEMENT

The use of Qwen as the backbone of our proposed method is described in the main paper. In preparing this manuscript, generative AI tools were used to improve clarity and refine the writing. They were not used to generate research ideas, develop methodologies, or conduct experiments. We verified all AI-assisted edits and take full responsibility for the content of this work.

REPRODUCIBILITY STATEMENT

To ensure reproducibility of our work, we have included details of the model implementation, training configurations, and evaluation protocol in Section 4.1, Appendix A.1, and Appendix A.2. We will also make our code publicly available.

REFERENCES

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 *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11618–11628. IEEE, 2020.

Jiacheng Chen, Yuefan Wu, Jiaqi Tan, Hang Ma, and Yasutaka Furukawa. Maptracker: Tracking with strided memory fusion for consistent vector hd mapping. In *European Conference on Computer Vision*, pp. 90–107. Springer, 2024.

Shaoyu Chen, Yunchi Zhang, Bencheng Liao, Jiafeng Xie, Tianheng Cheng, Wei Sui, Qian Zhang, Chang Huang, Wenyu Liu, and Xinggang Wang. Vma: Divide-and-conquer vectorized map annotation system for large-scale driving scene. In *arXiv preprint arXiv:2304.09807*, 2023.

Wentao Cheng, Sheng Yang, Maomin Zhou, Ziyuan Liu, Yiming Chen, and Mingyang Li. Road mapping and localization using sparse semantic visual features. *IEEE Robotics and Automation Letters*, 6(4):8118–8125, 2021.

Wenjie Ding, Limeng Qiao, Xi Qiu, and Chi Zhang. Pivotnet: Vectorized pivot learning for end-to-end hd map construction. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 3649–3659, 2023. doi: 10.1109/ICCV51070.2023.00340.

Miao Fan, Yi Yao, Jianping Zhang, Xiangbo Song, and Daihui Wu. Neural hd map generation from multiple vectorized tiles locally produced by autonomous vehicles. In *Spatial Data and Intelligence: 5th China Conference, SpatialDI 2024, Nanjing, China, April 25–27, 2024, Proceedings*, pp. 307–318, Berlin, Heidelberg, 2024. Springer-Verlag. ISBN 978-981-97-2965-4. doi: 10.1007/978-981-97-2966-1_22. URL https://doi.org/10.1007/978-981-97-2966-1_22.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=nZeVKeeFYf9.

Linus Kreitner, Paul Hager, Jonathan Mengedoht, Georgios Kaissis, Daniel Rueckert, and Martin J. Menten. Efficient numeracy in language models through single-token number embeddings. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Bh4Ubk80M8.

Qi Li, Yue Wang, Yilun Wang, and Hang Zhao. Hdmapnet: An online hd map construction and evaluation framework. In *2022 International Conference on Robotics and Automation (ICRA)*, pp. 4628–4634, 2022. doi: 10.1109/ICRA46639.2022.9812383.

Shiyi Liang, Xinyuan Chang, Changjie Wu, Huiyuan Yan, Yifan Bai, Xinran Liu, Hang Zhang, Yujian Yuan, Shuang Zeng, Mu Xu, et al. Persistent autoregressive mapping with traffic rules for autonomous driving. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 6862–6870, 2026.

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 *International Conference on Learning Representations*, 2023.

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*, pp. 1–23, 2024.

Yicheng Liu, Tianyuan Yuan, Yue Wang, Yilun Wang, and Hang Zhao. Vectormapnet: End-to-end vectorized hd map learning. In *International Conference on Machine Learning*. PMLR, 2023.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. Large language models: A survey. *arXiv preprint arXiv:2402.06196*, 2024.

Limeng Qiao, Wenjie Ding, Xi Qiu, and Chi Zhang. End-to-end vectorized hd-map construction with piecewise bezier curve. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13218–13228, June 2023a.

Zhijian Qiao, Zehuan Yu, Huan Yin, and Shaojie Shen. Online monocular lane mapping using catmull-rom spline. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7179–7186. IEEE, 2023b.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)*, pp. 1715–1725, 2016.

Anqi Shi, Yuze Cai, Xiangyu Chen, Jian Pu, Zeyu Fu, and Hong Lu. Globalmapnet: An online framework for vectorized global hd map construction. *arXiv preprint arXiv:2409.10063*, 2024.

Aaditya K Singh and DJ Strouse. Tokenization counts: the impact of tokenization on arithmetic in frontier llms. *arXiv preprint arXiv:2402.14903*, 2024.

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 *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks 2021)*, 2021.

Deguo Xia, Zihan Li, Haochen Zhao, Dong Xie, Yuyao Kong, Xiyan Liu, Jizhou Huang, Mengmeng Yang, and Diange Yang. Mapagent: An industrial-grade agentic framework for city-scale lane-level map generation. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, KDD '26, pp. 8289–8300, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 9798400722592. doi: 10.1145/3770855.3818443.

Tianyuan Yuan, Yicheng Liu, Yue Wang, Yilun Wang, and Hang Zhao. Streammapnet: Streaming mapping network for vectorized online hd map construction. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 7356–7365, January 2024.

Yujian Yuan, Changjie Wu, Xinyuan Chang, Sijin Wang, Hang Zhang, Shiyi Liang, Shuang Zeng, and Mu Xu. Unimapgen: A generative framework for large-scale map construction from multimodal data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 12277–12285, 2026.

Yun Zhang, Xin Cheng, Zhou Zhou, Jingmei Zhou, and Tong Yang. Ladel: Lane detection via multimodal large language model with visual instruction tuning. *Journal of Visual Communication and Image Representation*, 116:104704, 2026.

A APPENDIX

Table of Contents

A.1 Implementation and Training Details 12
A.1.1 Training Setup 12
A.1.2 Implementation Details 12
A.2 Pretraining Tasks for Coordinate Tokenization 13
A.3 Training on Detector Outputs 13
A.4 Qualitative Results 13
A.4.1 Comparison with Map Aggregation Baselines 13
A.4.2 Coordinate Tokenization and Pretraining 14
A.5 Further Discussion 14

A.1 IMPLEMENTATION AND TRAINING DETAILS

A.1.1 TRAINING SETUP

All experiments are conducted on a single NVIDIA B200 GPU. We use Qwen3.5-4B as the backbone and fine-tune it for map aggregation with LoRA, as described in Section 3.3. We apply LoRA to the query and value projections in the attention layers, as well as the gate, up-projection, and down-projection layers in the feed-forward blocks. We use a batch size of 1 with gradient accumulation over 8 steps. The learning rate is set to 2×10^{-4} with a cosine learning-rate schedule and 200 warm-up steps. We optimize the model using AdamW with a weight decay of 0.01. The checkpoint with the lowest validation loss is used for evaluation.

A.1.2 IMPLEMENTATION DETAILS

We train separate models for Argoverse2 and nuScenes. For each dataset, a single aggregation model is trained on the synthetic local maps described in Section 3.2 and evaluated on predictions from both MapTracker and StreamMapNet, without detector-specific retraining.

Before map serialization, we first translate all coordinates in the local observations and the target global map so that the ego position of the first frame becomes the origin:

$$\mathbf{p}_i = \mathbf{p}_i^{\text{global}} - \mathbf{o}_{T_1}, \tag{15}$$

where $\mathbf{o}_{T_1} \in \mathbb{R}^2$ denotes the ego position of the first frame in the original map coordinate system. Coordinates are represented at a resolution of $0.1\,\text{m}$ over a range of $[-350, 350]$ meters. The newly added coordinate-token rows in the input embedding matrix and LM head are initialized from the pretrained representations of the corresponding digit strings and rescaled to match the norms of the existing vocabulary representations.

We serialize each map element as its category followed by its ordered point coordinates:

$$(\mathbf{P}, c) = \big[c, (x_1, y_1), \ldots, (x_N, y_N)\big], \tag{16}$$

where (x_i, y_i) denotes the translated coordinates of the i-th point. The coordinates are encoded using the new coordinate tokens. The input sequence contains local polylines from multiple frames, while the target sequence contains the corresponding clean global polylines. The numbers of frames, polylines, and points per polyline vary across different scenes, so the input and target sequences also vary in length.

Since a global vectorized map is an unordered set of polylines, we use a deterministic ordering for the target polylines during training. Specifically, we first canonicalize the direction of each polyline

and then sort the polylines according to their endpoint coordinates. This gives each target map a consistent serialization for autoregressive training. We also randomize the order of local polylines within each input frame during training to reduce sensitivity to input ordering.

For the line-level association loss, we set both the loss weight λ_{line} and the temperature τ to 0.1.

A.2 Pretraining Tasks for Coordinate Tokenization

We construct synthetic geometric and arithmetic tasks using 2D coordinates represented by the new coordinate tokens. The tasks include 2D coordinate translation, point-wise subtraction, midpoint prediction, point sorting, nearest-point selection, and bounding-box corner prediction. Translation adds a displacement to a 2D point, while subtraction predicts the displacement between two points. Midpoint prediction computes the coordinate-wise average of two points, providing supervision for an operation relevant to merging nearby observations. Point sorting orders points first by y and then by x, encouraging the model to learn a consistent spatial ordering. Nearest-point selection determines which of two candidate points is closer to a query point, providing supervision for geometric distance comparison. Bounding-box corner prediction returns the minimum and maximum coordinates of a polyline, capturing its spatial extent. Each task is defined by an operation and its input coordinates, followed by the correct answer. We encode all coordinates using the new tokens and apply the training loss only to the answer tokens.

We also pretrain the model on serialized local maps from the training scenes using next-token prediction. Given the current sequence prefix, the model predicts the next token in the serialized map sequence. This task is important, as it helps the model learn how the coordinate tokens are used within polyline sequences, which complements the numerical and spatial relationships learned from the geometric and arithmetic tasks. Since this pretraining task only involves local map observations, it does not require any ground-truth global maps.

Task	**Example**	**Downstream role**
Translation	`[10,2] + [-25,30] = [-15,32]`	coordinate translation
Subtraction	`[30,40] - [10,15] = [20,25]`	offset between two points
Midpoint	`mid: [10,20] [30,40] -> [20,30]`	merging nearby observations
Sorting	`sorty: [7,5] [2,1] -> [2,1] [7,5]`	spatial ordering
Nearest point	`near [0,0]: [3,4] [10,2] -> [3,4]`	geometric distance comparison
Bounding box	`bbox: [3,9] [-4,2] -> [-4,2] [3,9]`	spatial extent of a polyline

Table 3: Synthetic pretraining tasks over the new coordinate tokens. The training loss is applied only to the answer tokens.

A.3 Training on Detector Outputs

To evaluate whether training on the proposed synthetic dataset is sufficient for learning map aggregation, we compare our training strategy with directly training on real detector outputs. We keep the model architecture and two-stage training procedure unchanged and vary only the source of the noisy local maps used for training. The training set combines predictions from both MapTracker and StreamMapNet, and the resulting model is evaluated on real predictions from each detector.

As shown in Table 4, training with synthetic local maps improves F1, particularly for road boundaries. For lane dividers, synthetic training also provides substantial gains on Argoverse2 and improves MapTracker on nuScenes, while the model trained on real detector outputs achieves slightly higher F1 on nuScenes lane dividers when evaluated on StreamMapNet predictions. Overall, these additional results show that the proposed local map corruption process provides effective training signals and transfers well to different upstream map detectors.

A.4 Qualitative Results

A.4.1 Comparison with Map Aggregation Baselines

Fig. 4 presents additional qualitative comparisons with VMA (Chen et al., 2023) and MonoLaM (Qiao et al., 2023b) on the Argoverse2 and nuScenes datasets using predictions from

Table 4: **Training on synthetic data vs. real detector outputs.** Both rows use our method, and only the training datasets are different. Evaluation is on real MapTracker / StreamMapNet predictions.

	Argoverse2						nuScenes					
	Lane Divider			Road Boundary			Lane Divider			Road Boundary		
Training data	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
MapTracker												
Detection outputs	51.6	37.1	43.1	33.7	31.5	32.6	80.7	73.5	76.9	42.4	31.9	36.4
Synthetic (**Ours**)	54.4	48.7	**51.4**	42.8	41.0	**41.9**	74.7	82.1	**78.2**	48.1	43.3	**45.5**
StreamMapNet												
Detection outputs	57.0	57.1	57.1	34.9	34.9	34.9	72.5	72.4	**72.5**	34.4	22.4	27.1
Synthetic (**Ours**)	68.9	76.4	**72.4**	46.8	47.7	**47.2**	64.5	78.7	70.9	40.8	37.9	**39.3**

StreamMapNet (Yuan et al., 2024) and MapTracker (Chen et al., 2024). These examples cover different road structures and aggregation challenges. VMA often fails to associate road-boundary fragments belonging to the same global map element, particularly near boundary endpoints in Fig. 4(b) and (c) and along circular boundaries in Fig. 4(d) and (f). These association errors result in fragmented or incomplete aggregated road boundaries. MonoLaM struggles to recover the underlying geometry of complex road boundaries, especially around circular structures or closely spaced map elements, as illustrated in Fig. 4(a), (b), (e), and (f). It often introduces geometric distortions and redundant predictions in these regions. In comparison, MapMergeLLM better associates related fragments and reconstructs road boundaries that more closely match the ground truth.

A.4.2 Coordinate Tokenization and Pretraining

Fig. 5 illustrates how coordinate tokenization and pretraining can affect the aggregated maps. The vanilla tokenizer often fails to generate road boundaries present in the ground truth, resulting in incomplete maps and lower recall. In addition, both vanilla tokenization and coordinate tokenization without pretraining can produce polylines that extend far beyond their intended endpoints, as illustrated by the long red lines in Fig. 5(b) and (d). These errors suggest that these models struggle to determine where to stop generating a polyline. In contrast, our full model recovers more complete road structures and better preserves polyline endpoints, producing more complete and geometrically consistent maps.

A.5 Further Discussion

In the main paper, our experimental results demonstrate that vectorized map aggregation can be learned from synthetic local observations, which reduces the dependence on predefined association and merging rules. By combining the proposed coordinate-token pretraining with line-level association supervision, MapMergeLLM improves aggregation across different upstream detectors on Argoverse2 and nuScenes datasets. These results also motivate further study of data-driven approaches for larger maps and other related applications.

Limitations and Future Work. Our experiments demonstrate effective aggregation over individual driving scenes. Extending this framework to longer driving trajectories is a natural direction for future work. However, this could be challenging because the growing number of local observations can exceed the context window of LLMs. Compared with standard subword tokenization, our coordinate tokenization already alleviates this limitation by representing each coordinate value as a single token, which allows more map observations to fit within the same context window. Incremental aggregation over overlapping regions is a promising future direction for handling longer driving trajectories while keeping each input sequence within the model's context window.

The coordinate resolution of $0.1\mathrm{m}$ used in our experiments provides enough spatial detail and well supports the aggregation results reported on Argoverse2 and nuScenes. Adopting a finer resolution or extending the spatial range would further increase the vocabulary size. One possible direction is to explore hierarchical coordinate representations, which could accommodate larger maps or finer resolutions with fewer additional tokens.

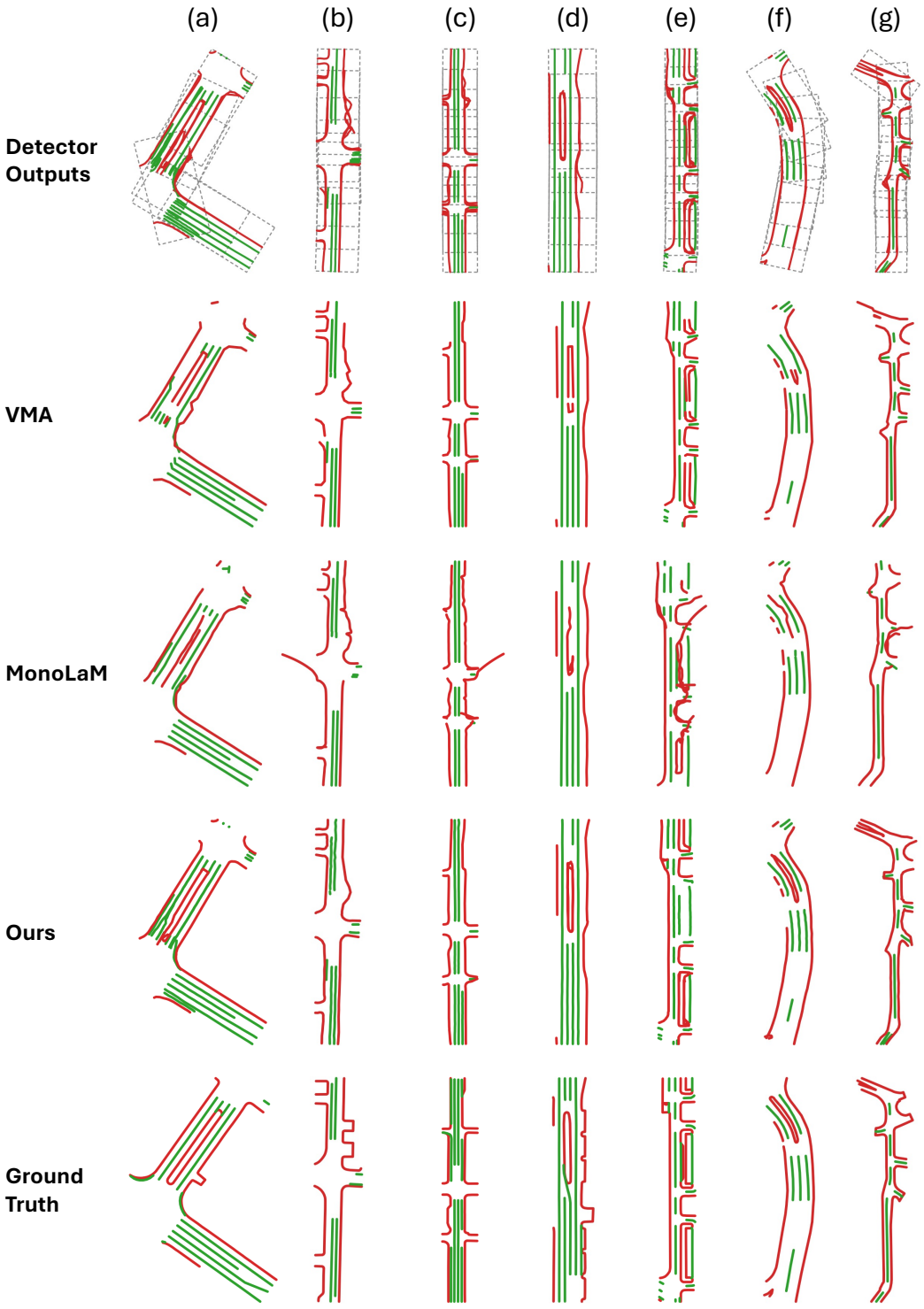


Figure 4: **Additional qualitative comparisons with map aggregation baselines.** (a)-(b) Argoverse2 with StreamMapNet predictions, (c)-(d) Argoverse2 with MapTracker predictions, (e) nuScenes with StreamMapNet predictions, and (f)-(g) nuScenes with MapTracker predictions.

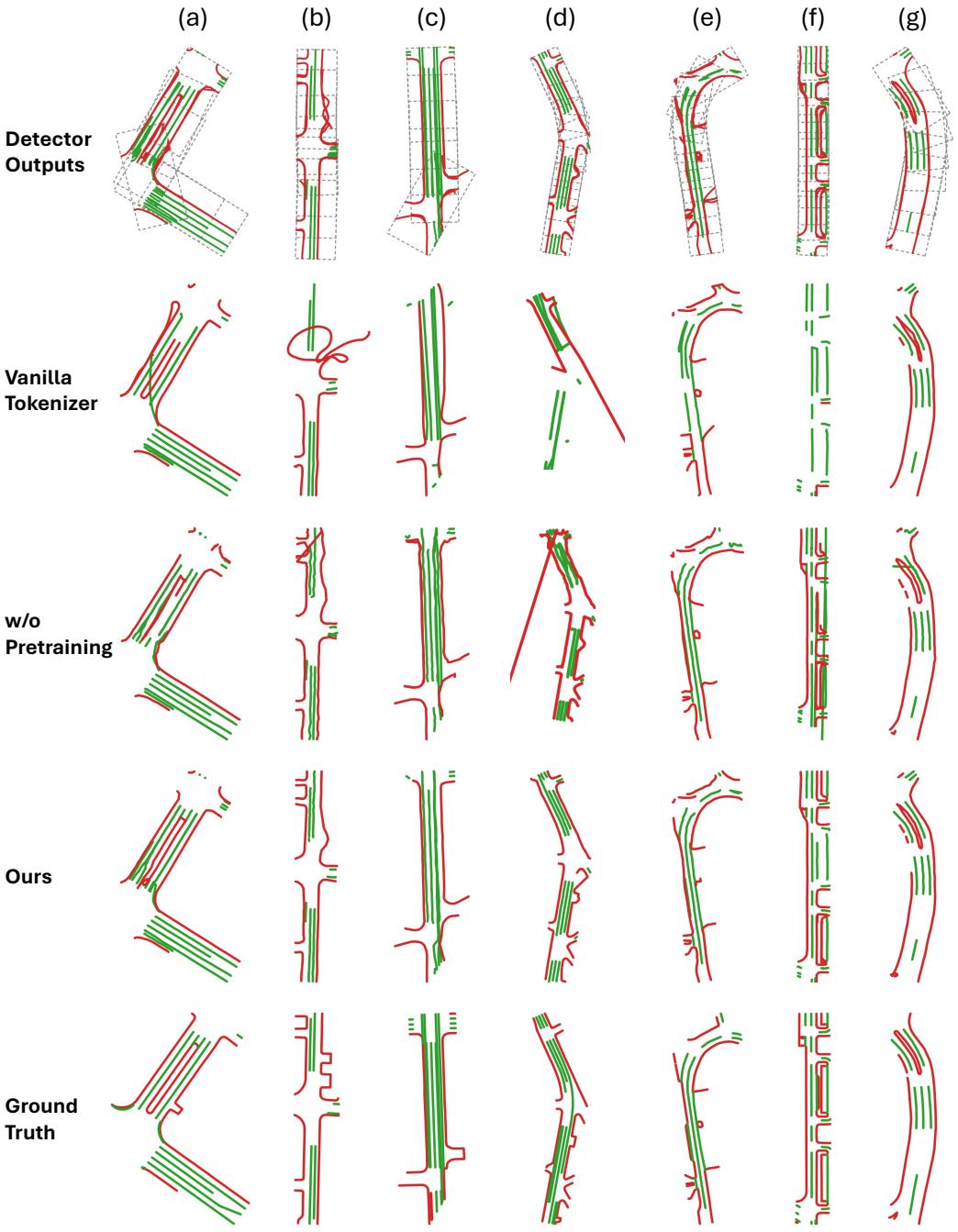


Figure 5: **Qualitative results on ablation study of coordinate tokenization and pretraining.** (a)-(c) Argoverse2 with StreamMapNet predictions, (d) Argoverse2 with MapTracker predictions, (e)-(f) nuScenes with StreamMapNet predictions, and (g) nuScenes with MapTracker predictions.