# MapExpert: Online HD Map Construction with Simple and Efficient Sparse Map Element Expert

**Dapeng Zhang[1], Dayu Chen[2], Peng Zhi[1*], Yinda Chen[3], Zhenlong Yuan[4], Chenyang Li[1], Sunjing[1], Rui Zhou[1], Qingguo Zhou[1*]**

[1]School of Information Science & Engineering, Lanzhou University, China
[2]Smart (Shanghai) Robotics Technology Co., Ltd., China
[3]School of Information Science and Technology, University of Science and Technology of China, China
[4]Institute of Computing Technology, Chinese Academy of Sciences, China
{zhangdp22, zhip13, lchengyang2024, sjing2023, zr, zhouqg}@lzu.edu.cn, cassidy.chen@smart.com, cyd0806@mail.ustc.edu.cn, yuanzhenlong21b@ict.ac.cn

## Abstract

Constructing online High-Definition (HD) maps is crucial for the static environment perception of autonomous driving systems (ADS). Existing solutions typically attempt to detect vectorized HD map elements with unified models; however, these methods often overlook the distinct characteristics of different non-cubic map elements, making accurate distinction challenging. To address these issues, we introduce an expert-based online HD map method, termed MapExpert. MapExpert utilizes sparse experts, distributed by our routers, to describe various non-cubic map elements accurately. Additionally, we propose an auxiliary balance loss function to distribute the load evenly across experts. Furthermore, we theoretically analyze the limitations of prevalent bird's-eye view (BEV) feature temporal fusion methods and introduce an efficient temporal fusion module called Learnable Weighted Moving Descent. This module effectively integrates relevant historical information into the final BEV features. Combined with an enhanced slice head branch, the proposed MapExpert achieves state-of-the-art performance and maintains good efficiency on both nuScenes and Argoverse2 datasets.

## Introduction

High-definition (HD) maps are essential for autonomous driving, conventionally constructed offline with SLAM-based methods (Zhang and Singh 2014; Shan et al. 2020), along with manual annotation. However, these methods are limited by scalability issues and high maintaining costs. In recent years, bird's-eye-view (BEV) feature extractors have introduced a novel thought (Philion and Fidler 2020; Li et al. 2022c; Zhang et al. 2023a), enabling the online construction of HD Maps from BEV features. These online approaches reduce offline human efforts by predicting HD Map elements in real-time, leading to cost savings and the ability to update changes in the environment promptly.

Early researchers leveraged segmentation tasks to obtain rasterized maps based on the BEV feature maps. These methods presented each rasterized pixel as a key point and then extracted map elements and their occupancy presentation (Zhou and Krähenbühl 2022; Hu et al. 2021). With the widespread use of transformers in vision tasks, HDMapNet (Li et al. 2022b) emerged, utilizing queries to predict HD map elements. Inspired by HDMapNet, (Liu et al. 2023, 2024b; Qiao et al. 2023b; Xu, Wong, and Zhao 2024; Zhou et al. 2024; Zhang et al. 2024b; Li 2024; Xiong et al. 2024; Shin et al. 2023) extracted structured map information and constructed vectorized maps by sampling elements as point sets, many of these works have improved performance by designing more reasonable content queries or embedding specified positional information into the queries. Recently, some scientists introduced novel tracking-based methods to enhance the HD map prediction performance, they associate HD map elements between frames via attention queries that evolve a set of track predictions (Yuan et al. 2023; Chen et al. 2024).

However, these DETR-like methods overlook the fact that online HD map construction elements are different from traditional detection objects. Traditional detection objects, such as cars and pedestrians, are typically cube-like and relatively uniform in shape, with centralized offsets. In contrast, HD map elements are vastly different: lane dividers are normally smooth Bézier curves, road boundaries are erratic slender lines that can be jagged or closed, and pedestrian crossings are closed rectangle shapes. Fitting these diverse non-cubic map elements with a single DETR-like design is challenging. Additionally, these methods stack previous BEV features to enhance the BEV feature expression, but this can lead to current BEV features being dominated by outdated data from historical features. These factors constrain prior methods from achieving optimal performance.

In this paper, we theoretically analyze these issues and propose a novel online map construction method based on map element experts, named **MapExpert**, the architecture is illustrated in Figure 1. Instead of using the unified modeling methods introduced by (Liao et al. 2023a; Yuan et al. 2023; Chen et al. 2024), which geometrically abstract different map elements into a unified representation, we employ distinct expert layers to accurately fit various map elements, such as lane dividers, pedestrian crossings, and road boundaries. We also mathematically analyze the drawbacks of stacking BEV features, and design MapExpert to not only strengthen current feature expression but also sieve and ex-

*Corresponding author.

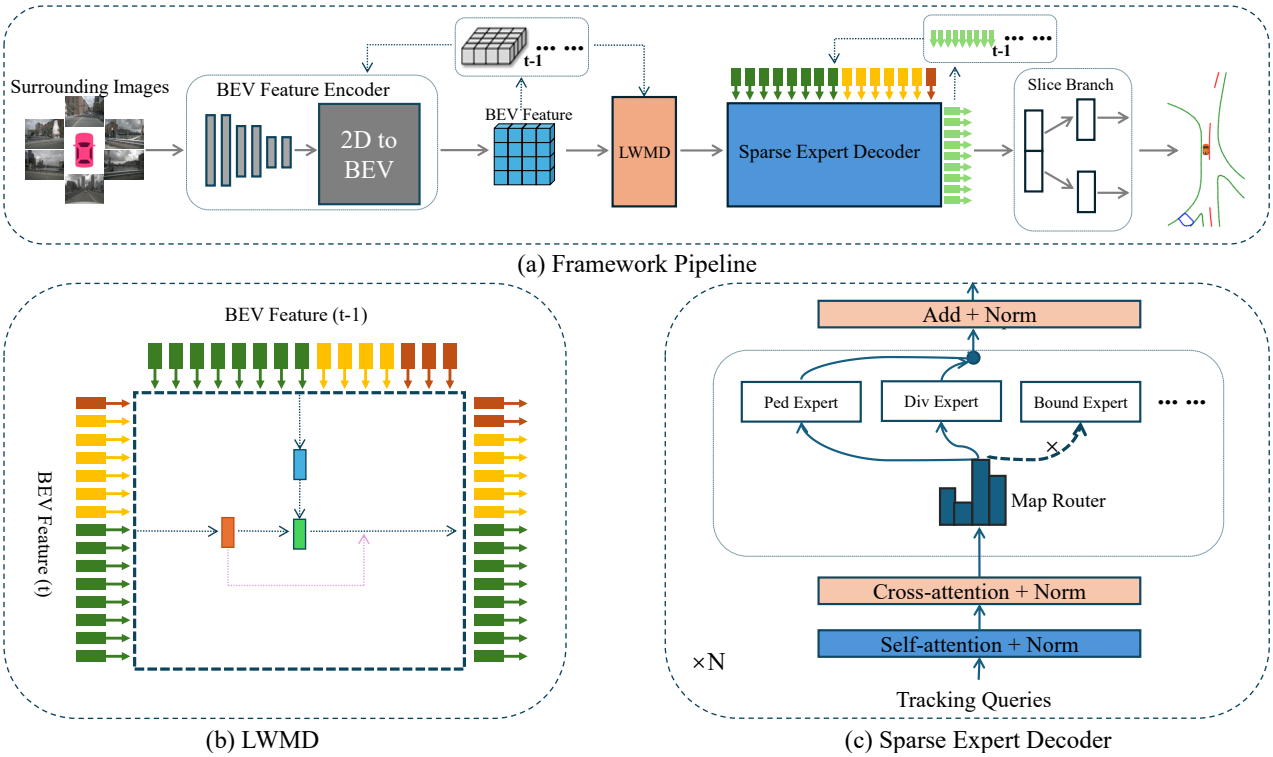


Figure 1: Overview of our newly introduced MapExpert: (a) The pipeline of MapExpert, consisting of our BEV feature encoder, learnable Weighted Moving Descent (LWMD), and sparse expert decoder. This pipeline processes surrounding images as input and generates vectorized map elements in an end-to-end module. (b) The detailed process of the Learnable Weighted Moving Descent, which extracts critical information from previous BEV features and enhances the representation of BEV HD map elements. (c) The structure of our unique sparse expert transformer layer is designed to effectively extract features of various map elements, such as lane dividers, pedestrian crossings, and road boundaries.

tract useful information for our decoder. To enhance the geographical position of predictions, we introduce a refined slice head branch that independently regresses map element positions and dimensions from slice tensors.

Our experiments demonstrate that MapExpert achieves state-of-the-art performance on the public nuScenes (Caesar et al. 2020) and Argoverse2 (Wilson et al. 2023) datasets. Concretely, with the same backbone and training epochs, MapExpert reaches 76.5 mAP on nuScenes, surpassing the existing best method by 1.8 mAP. Additionally, on Argoverse2, MapExpert outperforms the existing best method by 1.4 mAP for local HD map construction, using the same backbone and training epochs. Our ablation studies illustrate that MapExpert achieves significant improvements across various and complex HD map construction scenarios.

To summarize, the contributions of our paper are as follows:

- We propose an online HD map construction method based on a novel sparse map element expert. Our approach utilizes map element routers and sparse experts specifically designed to handle map elements of varying shapes.
- Building on this novel modeling approach, we theoretically analyze the limitations of prevalent BEV feature temporal fusion methods and introduce an efficient temporal fusion module called Learnable Weighted Moving Descent (LWMD). This module not only enhances the representation of current features but also filters and extracts useful information for our final BEV features.
- Experiments conducted on the nuScenes and Argoverse2 datasets demonstrate that our method achieves state-of-the-art performance, showing significant improvements over existing methods.

## Related Works

### Rasterization-based Methods

Rasterization-based approaches construct online HD maps by extracting a rasterized Bird's-Eye-View (BEV) representation from surrounding cameras (Philion and Fidler 2020; Li et al. 2022c; Zhang et al. 2023a) and then segmenting individual rasterized instances. Early methods are similar to 2D segmentation methods (Tsai et al. 2020; Xu et al. 2019; Zhang et al. 2023a), which usually predict the traversable area of BEV (Pan et al. 2020; Can et al. 2022). BEVLaneDet (Wang et al. 2023) and Persformer (Chen et al. 2022) treat 3D lane detection as a segmentation task based on rasterized BEV feature maps, they present each raster-

ized pixel as a key point to extract lane occupancy presentation. Similarly, (Hu et al. 2021) use simplified BEV raster representations of the surrounding scene for segmentation tasks. Furthermore, (Zhou and Krähenbühl 2022) discards positional embeddings derived from calibrated camera intrinsics and extrinsics, learning a camera-calibration-dependent mapping to predict a binary semantic segmentation mask. HDMapNet (Li et al. 2022b) also predicts semantic segmentation results on BEV features; however, unlike other rasterization-based methods, it employs complex post-processing to generate vectorized HD maps.

### Vectorization-based Methods

Despite the use of rasterized maps, distinguishing HD map elements remains challenging. VectorMapNet (Liu et al. 2023) was the first to introduce a two-stage network for predicting sequential sampling points of HD Map elements. Unlike VectorMapNet (Liu et al. 2023), MapTR (Liao et al. 2023a) introduces an end-to-end transformer structure that samples elements as point sets using a group of fixed permutations, this method uses hierarchical queries to extract structured map information and construct vectorized maps. Subsequently, several studies have improved performance by designing novel hierarchical queries. These methods use scattered instance queries that share content information within the same map elements to avoid inconsistencies in the content of sampling points (Liu et al. 2024b; Qiao et al. 2023b; Xu, Wong, and Zhao 2024; Zhou et al. 2024; Zhang et al. 2024b; Li 2024; Xiong et al. 2024; Shin et al. 2023). In addition, BeMapNet (Qiao et al. 2023a) delves a piecewise Bézier network with control point coordinates to manipulate curve shapes. Some papers introduce anti-disturbance methods to optimize jittery or jagged outputs (Hu et al. 2024). Furthermore, (Zhang et al. 2024a) enhances instance queries by adding specified positional information embedded from reference points. To alleviate the difficulty in element localization and relevant feature extraction due to the sparse and irregular detection targets, MGMap (Liu et al. 2024a) incorporates the guidance of learned map masks with instance and point queries. Unlike other vectorization-based methods, MapVR (Zhang et al. 2023b) introduces a combined solution. It transforms vectorized map elements into an HD Map and then adds segmentation supervision on the rasterized HD map, experiment results present a significant improvement. Most previous approaches use a fixed number of points, which may elide essential details. PivotNet (Ding et al. 2023) proposes a novel Point-to-Line mask structure to encode both subordinate and geometrical point-line priors, experiment shows a remarkably superior to others. Besides, P-MapNet (Jiang et al. 2024) and MapVision (Yang et al. 2024) incorporate standard-definition (SD) maps and sensors to improve performance, although their application is limited due to misalignment between SD map skeletons and BEV features. Additionally, some researchers exploit the HD map performance in long-range scenarios. They propose a hierarchical sparse map construction to obtain superior performance (Yu et al. 2024). Notably, recent research introduces generative methods that combine vecterized HD map models with learned generative models for semantic map layouts, these methods obtain a better accuracy, realism, and uncertainty awareness (Zhu et al. 2023; Chen, Deng, and Furukawa 2023).

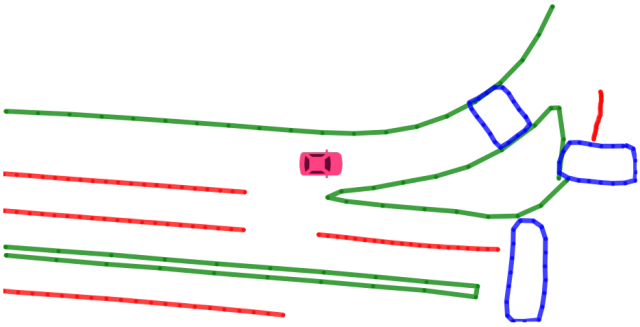

Figure 2: Topology of map elements (red: lane dividers, blue: pedestrian crossings, green: road boundaries). Differ from detection objects, which are typically cube-shaped, map elements are non-cubic and can take various shapes.

### Tracking-based Methods

Recently, researchers use transformer attention mechanism with track queries to associate tracking instances across frames (Meinhardt et al. 2022; Sun et al. 2021; Cai et al. 2022; Gao and Wang 2024). These methods achieve significant performance improvements in tracking tasks. Inspired by these methods, StreamMapNet (Yuan et al. 2023) selects the former foregoing top k (where k is less than the maximum number of queries) queries based on confidence score as potential tracking queries, and then concatenates them with initialized queries. Experiment results demonstrate that it outperforms previous methods. Similarly, SQD-MAP (Wang et al. 2024) builds upon StreamMapNet (Yuan et al. 2023) by incorporating stream query denoising (Li et al. 2022a). It feeds noised ground-truth map elements as noised queries (such as shifting, angular rotation, and scale transformation) along with learnable queries into the decoder to effectively decrease instability. MapTracker (Chen et al. 2024) further extends the tracking queries with memory mechanisms for better fusion. It explicitly associates tracked HD map elements from historical frames to further enhance temporal consistency. Researchers also use tracking-based HD maps as an online mapping component of end-to-end autonomous driving systems to achieve superior performance among all tasks (Sun et al. 2024).

## Problem Statement

### Inconsistent of Map Elements

Recently, most online HD map construction methods have been approached as detection tasks, involving the learning of anchors and relative offsets (Liu et al. 2023; Liao et al. 2023a; Chen et al. 2024). While this design is undoubtedly straightforward and efficient, it overlooks a critical distinction: online HD map elements differ significantly from traditional object detection targets. Traditional detection targets typically involve cube-like objects such as vehicles, pedestrians and animals with fixed physical dimensions. These

objects, which vary relatively little in shape, allow for a centralized representation of offsets across categories, simplifying the model design. It is easier to represent the features of different categories of objects using a unified model design. In contrast, as illustrated in Figure 2, HD map elements exhibit a wide range of geometric structures. Lane dividers are often smooth Bézier curves, road boundaries may appear as erratic, slender lines that can be jagged or closed, and pedestrian crossings are typically closed rectangles. This variability poses a significant challenge for DETR-like decoders, which are commonly used in HD map methods, as they may struggle to simultaneously capture the features of non-cubic map elements, such as Bézier curves, erratic lines, and rectangles. In addition, the diverse geometric characteristics of these map elements complicate the development of a unified geometric representation. Therefore, our analysis suggests the necessity of designing a novel approach that accurately represents various map elements without substantially increasing resource consumption.

## Dominance of Outdated Historical Data in the Current Frame

HD map elements are static instances, which differentiates them from objects in detection tasks; theoretically, they should be easy to fuse. To achieve superior performance, researchers often incorporate historical frames to enhance the BEV feature representation. A common and effective approach is to stack the aligned historical BEV features (Liao et al. 2023a; Yuan et al. 2023; Chen et al. 2024). For example, Maptracker (Chen et al. 2024) stacks four historical features. While this technique can improve performance, it also leads to an overreliance on historical frames, significantly reducing the contribution of the current frame. Additionally, it introduces substantial noise from distant historical frames. In this subsection, we will explore the nature of this issue in more details.

According to our analysis, the BEV feature stack principle reveals that each historical BEV feature contributes differently to the current stacked BEV feature used in the decoding process, as shown in Eq. 1 below:

$$C_t = \frac{1}{S}F_t + \frac{1}{S}\sum_{n=1}^{S-1} C_{t-n} \tag{1}$$

where $S(S >= 2)$ denotes the number of stacked frames. $t$ indicates the frame index of $t$-th, and $F_t$ is the BEV feature of the $t$-th frame. $C_t$ is the stacked BEV feature of $t$-th frame, which is composed of the current BEV feature extracted from the current frame and $S-1$ historical stacked BEV features. Similarly, $C_{t-n}$ is the stacked BEV feature of $(t-n)$-th frame, where $n$ ranges from 1 to $S-1$. We use a solved formula to represent the stacked BEV feature $C_t$, as shown in Eq. 2. Furthermore, approximation methods, such as the one in Eq. 3, are used to derive an explicit approximation expression, allowing for a clearer representation of the composition of the current BEV stacked features. The forms of Eq. 2 and Eq. 3 are as follows:

$$C_t = \frac{1}{S}F_t + \frac{1}{S}\sum_{n=1}^{t-1} W_{t-n}F_{t-n} \tag{2}$$

$$C_t \approx \begin{cases} \frac{1}{S}F_t + \frac{1}{S^2}F_{t-1} + \frac{1}{S^3}F_{t-2} + \hat{R}_1, & S = 2 \\ \frac{1}{S}F_t + \frac{1}{S^2}F_{t-1} + \frac{S+1}{S^3}F_{t-2} + \hat{R}_2. & S > 2 \end{cases} \tag{3}$$

where $W_{t-n}$ is the weight of $(t-n)$-th feature, which varies as $n$ increasing from 1 to $t-1$, $\hat{R}_1$ and $\hat{R}_2$ are remainders of formula. As indicated by Eq. 3, when the stacked number $S$ increases, the composition of the current BEV feature extracted from the current frame will decrease, which is $\frac{1}{S}F_t$. In other words, the stacked BEV feature is more likely to be dominated by historical BEV features. This is contrary to the expectation that the proportion of important information from the current BEV feature should be more reliable and substantial, which means that the current BEV feature should contribute more to current stacked BEV feature $C_t$. Furthermore, our analysis reveals that the proportion of HD map elements in a rasterized HD map is less than 5%. This implies that adding more historical BEV features introduces more irrelevant information, which can even transmit noise that negatively affects HD map reconstruction. We will propose a method to address this issue effectively.

# Methods

## Architecture Overview

The overall model architecture is illustrated in Figure 1, and is streamlined into three components as follows:

**BEV Encoder:** Given surrounding images $P \in \mathbb{R}^{N\times 3\times H\times W}$ from multi-camera setups, we extract multi-scale image features using ResNet (He et al. 2015) and FPN (Lin et al. 2017). These multi-scale features are then fed into a 2D-to-BEV transformer encoder (Li et al. 2022c; Chen et al. 2024). Our BEV queries are initialized with the previously aligned BEV feature, to obtain the BEV feature map.

**Learnable Weighted Moving Descent:** Most existing methods concatenate the previously aligned hidden states of BEV features with the current BEV feature to enhance the expression of BEV HD map elements. However, as mentioned earlier, stacked BEV features are more likely to be dominated by historical BEV features, despite the current BEV feature's potential for greater contribution. In this method, we mathematically analyze this issue and propose a novel strategy named Learnable Weighted Moving Descent (LWMD) to fuse the BEV features reasonably.

**Sparse Expert Decoder:** We utilize a hierarchical tracking query scheme to explicitly fuse historical tracking features and extract map elements with our sparse expert transformer layer. Specifically, we initialize the tracking query as $TQ(t) = [TQ_{element}(t-1), TQ_{init}]$. $TQ(t)$ is the final current tracking query that will be fed into the sparse expert transformer layer. $TQ_{element}(t-1)$ denotes the aligned previous decoder output, $TQ_{init}$ presents the initialized empty query, which can be used to pad the $TQ(t)$ to the designed

| Methods | Backbone | Epoch | $AP_{ped}$ | $AP_{div}$ | $AP_{bound}$ | mAP | $AP_{ped}$ | $AP_{div}$ | $AP_{bound}$ | mAP |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Hard: {0.2, 0.5, 1.0}m | | | | Easy: {0.5, 1.0, 1.5}m | | | |
| HDMapNet(Li et al. 2022b) | EB0 | 120 | – | – | – | – | 14.4 | 21.7 | 33.0 | 23.0 |
| VectorMapNet(Liu et al. 2023) | R50 | 110 | 20.6 | 32.4 | 24.3 | 25.7 | 42.5 | 51.4 | 44.1 | 46.0 |
| MapTR(Liao et al. 2023a) | R50 | 110 | 31.4 | 40.5 | 35.5 | 35.8 | 56.2 | 59.8 | 60.1 | 58.7 |
| MapTRv2(Liao et al. 2023b) | R50 | 110 | 43.6 | 49 | 43.7 | 45.4 | 68.1 | 68.3 | 69.7 | 68.7 |
| MapQR(Liu et al. 2024b) | R50 | 110 | 46.2 | 57.3 | 48.1 | 50.5 | 67.1 | 70.4 | 71.2 | 69.6 |
| StreamMapNet(Yuan et al. 2023) | R50 | 110 | 44.4 | 60.5 | 48.6 | 51.2 | 68 | 71.2 | 69.2 | 69.5 |
| MapTracker(Chen et al. 2024) | R50 | 100 | 52.3 | 62.5 | 59.6 | 58.13 | 77.3 | 72.4 | 74.2 | 74.7 |
| Proposed | R50 | 100 | **55.7** | **64.2** | **61.3** | **60.4** | **79.4** | **73.9** | **76.2** | **76.5** |

Table 1: Performance comparison of various methods on the original nuScenes split at both 50m and 30m perception ranges. Specifically, results are evaluated on {0.2 m, 0.5 m, 1.0 m} and {0.5 m, 1.0 m, 1.5 m} thresholds. MapExpert notably outperforms other methods across all categories. The best results for same settings (i.e., backbone and epoch) are highlighted in bold.

| Methods | $AP_{ped}$ | $AP_{div}$ | $AP_{bound}$ | mAP |
|---|---|---|---|---|
| StreamMapNet(Yuan et al. 2023) | 31.6 | 28.1 | 40.7 | 33.5 |
| MapTracker(Chen et al. 2024) | 45.9 | 30.0 | 45.1 | 40.3 |
| Proposed | **46.7** | **34.1** | 45.1 | **42.0** |

Table 2: Comparison with SOTA Methods on the new nuScenes split validation set. All the experiments are based on ResNet50 backbone. Results are evaluated on {0.5 m, 1.0 m, 1.5 m} thresholds.

size. This tracking strategy is borrowed from (Gao and Wang 2024; Chen et al. 2024). Subsequently, we send the $TQ(t)$ and final BEV feature into our sparse expert transformer layer, where the map element experts of our sparse expert transformer layer will extract excellent map element characteristics. Details of the sparse expert transformer layer will be provided later.

## Sparse Expert Transformer Layer

As analyzed in the "Inconsistent of Map Elements" subsection, most online HD Map construction methods are detection tasks, which involve learning anchors and relative offsets. This design is undoubtedly simple and efficient. However, traditional detection objects, such as cube-like vehicles and animals with fixed physical dimensions, are significantly different in shape from map elements like bézier curves, erratic slender lines, and rectangles. Therefore, using a detection-based design to fit these map elements is inappropriate.

Inspired by Mixture of Experts (Shazeer et al. 2017), which selects different parameters for each incoming example, we have designed a novel decoder layer named the sparse expert transformer layer. This layer mainly consists of self-attention, cross-attention, and sparse map element experts. A brief overview of this layer is illustrated in Figure 1. First, self-attention takes tracking queries as inputs to obtain a representation, then extracts map features with cross-attention. The output from cross-attention is fed into our sparse map element expert, which is composed by rooters and map element experts. Each expert is responsible for specific map elements (lane dividers, pedestrian crossings, and road boundaries). Concretely, our map expert router primarily routes the representation from the previous module to the best-determined top-K experts selected from a set $\{E_i(x)\}_{i=0}^{N-1}$, where $N$ is the total number of experts, and $E_i$ is the $i$-th expert. Briefly, we normally do not compute the outputs of experts whose routes are zero, this sparse route could limit the computation costs. Our routers are implemented by normalizing via a softmax distribution over the top-K logits. As illustrated below:

$$R(x) = SoftMax(TopK(x \cdot W_r)) \tag{4}$$

where $x$ is the map element feature, $R(x)$ denotes the output of the map expert router, $R(x) = ri$ if $ri$ is among the top-K coordinates of logits, and $R(x) = -\infty$ otherwise. The router variable $W_r$ produces logits $x \cdot W_r$. The value K of top-K is a hyper-parameter that modulates the amounts of experts used to process map elements. This design has a notable success in computational efficiency, which means that even if we increase $N$ while keeping $K$ fixed, the model's parameter count increases while the computational cost remains constant. This motivates a distinction between the model's total parameter count and the number of parameters used for processing an individual active parameter count, also known as sparse expression. Our map element feature $x$ meant to be processed by specific experts, is routed to the corresponding expert for processing. The expert's output is then returned to the original query position. As shown in Figure 1, we design three types of experts: the lane divider experts, the pedestrian crossing experts, and the road boundary experts. These experts are intended to extract different types of map element features, such as bézier curves, erratic slender lines, and rectangles. We use the expert router to select top-K experts from a set $\{E_i(x)\}_{i=0}^{N-1}$ expert networks, the simplified expression of the sparse map element expert is given by:

$$y = \sum_{i=0}^{N-1} R_i(x) E_i(x) \tag{5}$$

here, $y$ is the output, $x$ is the map element feature, $N$ is the total experts count, $R_i(x)$ is the $i$-th router output, and $E_i(x)$ is the $i$-th sparse map element expert. Concretely, we use the SwiGLU as the expert $E_i(x)$, which means that each map element feature $x$ is routed to K SwiGLU blocks with

| Methods | Backbone | Epoch | $AP_{ped}$ | $AP_{div}$ | $AP_{bound}$ | mAP |
|---|---|---|---|---|---|---|
| HDMapNet(Li et al. 2022b) | EB0 | 120 | 13.1 | 5.7 | 37.6 | 18.8 |
| VectorMapNet(Liu et al. 2023) | R50 | 110 | 36.5 | 35.0 | 36.2 | 35.8 |
| MapTR(Liao et al. 2023a) | R50 | 110 | 55.4 | 58.7 | 59.1 | 57.8 |
| MapTRv2(Liao et al. 2023b) | R50 | 110 | 60.7 | 68.9 | 64.5 | 64.7 |
| MapQR(Liu et al. 2024b) | R50 | 110 | 71.2 | 60.1 | 66.2 | 65.9 |
| StreamMapNet(Yuan et al. 2023) | R50 | 110 | 64.9 | 60.2 | 64.9 | 63.3 |
| MapTracker(Chen et al. 2024) | R50 | 100 | 74.5 | 66.4 | 73.4 | 71.4 |
| Proposed | R50 | 100 | **76.4** | **66.9** | **75.1** | **72.8** |

Table 3: Performance comparison with baseline methods on the original Argoverse2 split at 30 m perception ranges. MapExpert outperforms existing state-of-the-art methods. Results are evaluated using thresholds of {0.5 m, 1.0 m, 1.5 m}.

| Methods | $AP_{ped}$ | $AP_{div}$ | $AP_{bound}$ | mAP |
|---|---|---|---|---|
| StreamMapNet(Yuan et al. 2023) | 61.8 | 68.2 | 63.2 | 64.4 |
| MapTracker(Chen et al. 2024) | 70.0 | 75.1 | 68.9 | 71.3 |
| Proposed | **71.2** | **75.6** | 68.7 | **71.8** |

Table 4: Comparison with state-of-the-art method on new Argoverse2 split, following the evaluation criteria used in Table 3. Results are evaluated on {0.5 m, 1.0 m, 1.5 m} thresholds.

different sets of weights. The output $y$ for an input token $x$ is represented as:

$$y = \sum_{i=0}^{N-1} SoftMax(TopK(x \cdot W_r))_i \cdot SwinGLU_i(x) \quad (6)$$

Note that this final formulation introduces challenges in load balancing, which will be analyzed in the auxiliary loss subsection.

## Learnable Weighted Moving Descent

As illustrated in Eq.3, existing methods usually concatenate historical BEV features, which can lead to the issue analyzed in the second part of the problem statement: an over-reliance on historical BEV features can disproportionately influence the current BEV feature. Therefore, this paper proposes a novel module called Learnable Weighted Moving Descent (LWMD), which integrates historical BEV features into the current BEV features without increasing device memory. Our LWMD uses a single previous BEV feature to achieve superior performance compared to stacking multiple BEV frames. Our approach is a learnable method that automatically adjusts the fusion between features. The formula for LWMD is as follows:

$$C_t = \beta F_t + f_t(F_t, C_{t-1}) \quad (7)$$

here, the final fused result $C_t$ consists of two components: the current BEV feature $F_t$ and the map element information extracted from the current BEV feature and the last fused BEV feature via $f_t$, $f_t$ is a neural network received the current BEV feature $F_t$ and the previous fused BEV feature $C_{t-1}$ as inputs. $\beta$ is a learnable parameter. In our formula, the proportion of current BEV features is learnable, thereby mitigating the issue of historical data dominance as highlighted in the problem statements. To sum up, our approach, which fuses two frames of features, achieves performance comparable to or better than that of stacking multiple BEV frames.

## Auxiliary Expert Balance Loss

Without a balance strategy, experts may encounter an inhomogeneous situation. For example, one expert might be trained to handle all three map elements, while others may be skipped forever. To encourage a balanced load across different experts, we add an auxiliary loss named the auxiliary expert balance loss for the sparse expert transformer layer. This additional loss encourages each expert to be of equal importance. Differing from (Shazeer et al. 2017; Lepikhin et al. 2020), we distribute the workload evenly with a simplified design. Given N experts indexed by i = 0 to $N-1$ and $T$ tokens, the auxiliary expert balance loss is calculated as the scaled dot product between parameters $f$ and $P$.

$$L_{expert-balance} = \alpha \cdot N \cdot \sum_{i=0}^{N-1} f_i \cdot P_i \quad (8)$$

$$f_i = \frac{1}{T} OneHot(TopK(\frac{e^{p_i(x)}}{\sum_{j=0}^{N-1} e^{p_j(x)}})) \quad (9)$$

$$P_i = \frac{1}{T} p_i(x) \quad (10)$$

here, $f_i$ is the percentage of inputs routed to each expert, $\alpha$ is a hyper-parameter, and $P_i$ is the fraction of the router probability allocated to each corresponding expert, $p_i$ is the probability of routing token $x$ to expert $i$.

The auxiliary expert balance loss, as described in Eq.8, ensures uniform routing for three map elements. This loss function is differentiable thanks to the $P_i$. Finally, we add an auxiliary balance loss to the total loss during training.

# Experiments

## Experimental Settings

**Datasets.** We evaluate our MapExpert on the nuScenes (Caesar et al. 2020) and Argoverse2 datasets (Wilson et al. 2023). The nuScenes is a comprehensive, synthetically generated autonomous driving dataset, consisting of 1000 scenes with annotations at 2 Hz. Each frame contains data

| Index | + Expert | + Expert Balance loss | + LWMD | + Slice Branch | $AP_{ped}$ | $AP_{div}$ | $AP_{bound}$ | mAP |
|---|---|---|---|---|---|---|---|---|
| 0 | | | | | 77.6 | 71.2 | 73.0 | 73.9 |
| 1 | ✔ | | | | 78.3 | 72.7 | 73.9 | 75.0 |
| 2 | ✔ | ✔ | | | 79.0 | 73.5 | 76.1 | 76.2 |
| 3 | ✔ | ✔ | ✔ | | 79.5 | 73.4 | 76.4 | 76.4 |
| 4 | ✔ | ✔ | ✔ | ✔ | 79.4 | 73.9 | 76.2 | **76.5** |

Table 5: Ablation studies on the key design elements of MapExpert, evaluated on the origin nuScenes split dataset. Results show that each modification contributes to the performance gain.

from six synchronized surrounding cameras. We use the 2D vectorized map elements provided by nuScenes as the ground truth. Argoverse2 is another large-scale benchmark with approximately 108,000 frames, each offering images from seven surrounding cameras. Differing with nuScenes, Argoverse2 provides 3D vectorized map elements as ground truth.

**Implementation Details.** We follow the majority of the settings from MapTracker (Chen et al. 2024), which uses ResNet50 (He et al. 2015) and BEVFormer (Li et al. 2022c) for BEV feature extraction, we then conduct with our decoder to obtain a refined prediction result. We perform our experiments on six A800 GPUs. MapExpert has notable successes in local HD map construction, however, it suffers from training instabilities. To address this, we apply several training techniques. First, we incorporate an additional segmentation loss to facilitate convergence, as the vectorized loss may cause divergence during training. Second, we use a large batch size to prevent getting trapped in local optima or experiencing complete divergence.

## Comparisons with State-of-the-art Methods

We implemented our method based on the approaches described in (Liao et al. 2023a; Yuan et al. 2023; Chen et al. 2024). We also adopted the dataset split strategy of StreamMapNet, and evaluated our method with both the original and new split strategies. Table 1 and Table 3 follow the original split strategy, Table 2 and Table 4 follow the new split strategy of StreamMapNet. These split strategies only differ in the division between the training set and the validation set. For a fair comparison, we evaluated our method and other methods using distinct thresholds: $\{1.0$ m, 1.5 m, 2.0 m$\}$ for the 50 m range and $\{0.5$ m, 1.0 m, 1.5 m$\}$ for the 30 m range.

**Performance on nuScenes.** We provide a comparison of MapExpert's performance against existing methods to ensure a comprehensive analysis. As illustrated in Table 1, our method achieves a better mAP with the original dataset split ground truth. Concretely, MapExpert significantly outperforms the competing methods by more than 2.4% in mAP scores with the original dataset split. Note that, MapExpert achieves 79.4 AP of pedestrian crossings, which is 2.1 mAP higher than the result of the previous best-performing method. We further compare our method with existing methods using the new NuScenes split dataset. As shown in Table 2, our approach outperforms MapTracker by a notable margin (with improvements of +1.7 mAP overall, +4.1 AP in lane dividers, +0.9 AP in pedestrian crossings).

**Performance on Argoverse2.** We also evaluated our method on Argoverse2 datasets. Table 3 shows the comparison on the original Argoverse2 split. Our method achieves superior performance over previous best-performing methods across all map elements, with 1.4 mAP higher than MapTracker and 9.5 mAP higher than StreamMapNet. Based on geographically non-overlapping splits proposed by StreamMapNet, Table 4 reveals performance on the new Argoverse2 split. The experiments on the new Argoverse2 dataset split demonstrate the superior construction ability of MapExpert in local HD map construction. We achieve 71.8 mAP, which is 0.5 mAP higher than MapTracker.

## Ablation Studies

**Key Components Design.** We conducted several ablation studies on the original nuScenes split to confirm the necessity of the proposed modules. Initially, we integrated our modules into the baseline of MapTracker, and the influence of each component is presented in Table 5. It is evident that all design elements in our MapExpert contribute to performance improvements, thereby validating their necessity. The index 0 is baseline. The second variant incorporates map expert components into the decoder modules, which induces approximately a 0.4% performance increase over the baseline. The third variant includes both the experts and the auxiliary expert balance loss components, this variant surpasses the baseline by 1.5 mAP, demonstrating the effectiveness of our mechanism. The fourth variant incorporates our LWMD after the BEV encoder. This design enables the model to extract useful information from historical features without the need to stack BEV features, resulting in a 0.2 mAP improvement. Additionally, we introduced a slice branch to predict map elements. Although this modification only results in a 0.1 mAP improvement.

**Extended version** — https://arxiv.org/abs/2412.12704

# Conclusion

MapExpert is a simple and efficient online HD map construction method that introduces a sparse map element expert transformer architecture and Learnable Weighted Moving Descent (LWMD) strategy to model road structure topology based on tracking-based methods. Extensive experiments demonstrate that it significantly outperforms existing methods on nuScenes and Agroverse2 datasets.

## References


Caesar, H.; Bankiti, V.; Lang, A. H.; Vora, S.; Liong, V. E.; Xu, Q.; Krishnan, A.; Pan, Y.; Baldan, G.; and Beijbom, O. 2020. nuScenes: A multimodal dataset for autonomous driving. arXiv:1903.11027.

Cai, J.; Xu, M.; Li, W.; Xiong, Y.; Xia, W.; Tu, Z.; and Soatto, S. 2022. MeMOT: Multi-Object Tracking with Memory. arXiv:2203.16761.

Can, Y. B.; Liniger, A.; Unal, O.; Paudel, D.; and Gool, L. V. 2022. Understanding Bird's-Eye View of Road Semantics using an Onboard Camera. arXiv:2012.03040.

Chen, J.; Deng, R.; and Furukawa, Y. 2023. PolyDiffuse: Polygonal Shape Reconstruction via Guided Set Diffusion Models. arXiv:2306.01461.

Chen, J.; Wu, Y.; Tan, J.; Ma, H.; and Furukawa, Y. 2024. MapTracker: Tracking with Strided Memory Fusion for Consistent Vector HD Mapping. arXiv:2403.15951.

Chen, L.; Sima, C.; Li, Y.; Zheng, Z.; Xu, J.; Geng, X.; Li, H.; He, C.; Shi, J.; Qiao, Y.; and Yan, J. 2022. PersFormer: 3D Lane Detection via Perspective Transformer and the OpenLane Benchmark. arXiv:2203.11089.

Ding, W.; Qiao, L.; Qiu, X.; and Zhang, C. 2023. PivotNet: Vectorized Pivot Learning for End-to-end HD Map Construction. arXiv:2308.16477.

Gao, R.; and Wang, L. 2024. MeMOTR: Long-Term Memory-Augmented Transformer for Multi-Object Tracking. arXiv:2307.15700.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385.

Hu, A.; Murez, Z.; Mohan, N.; Dudas, S.; Hawke, J.; Badrinarayanan, V.; Cipolla, R.; and Kendall, A. 2021. FIERY: Future Instance Prediction in Bird's-Eye View from Surround Monocular Cameras. arXiv:2104.10490.

Hu, H.; Wang, F.; Wang, Y.; Hu, L.; Xu, J.; and Zhang, Z. 2024. ADMap: Anti-disturbance framework for reconstructing online vectorized HD map. arXiv:2401.13172.

Jiang, Z.; Zhu, Z.; Li, P.; ang Gao, H.; Yuan, T.; Shi, Y.; Zhao, H.; and Zhao, H. 2024. P-MapNet: Far-seeing Map Generator Enhanced by both SDMap and HDMap Priors. arXiv:2403.10521.

Lepikhin, D.; Lee, H.; Xu, Y.; Chen, D.; Firat, O.; Huang, Y.; Krikun, M.; Shazeer, N.; and Chen, Z. 2020. GShard: Scaling Giant Models with Conditional Computation and Automatic Sharding. arXiv:2006.16668.

Li, F.; Zhang, H.; Liu, S.; Guo, J.; Ni, L. M.; and Zhang, L. 2022a. DN-DETR: Accelerate DETR Training by Introducing Query DeNoising. arXiv:2203.01305.

Li, Q.; Wang, Y.; Wang, Y.; and Zhao, H. 2022b. HDMapNet: An Online HD Map Construction and Evaluation Framework. arXiv:2107.06307.

Li, T. 2024. MapNeXt: Revisiting Training and Scaling Practices for Online Vectorized HD Map Construction. arXiv:2401.07323.

Li, Z.; Wang, W.; Li, H.; Xie, E.; Sima, C.; Lu, T.; Yu, Q.; and Dai, J. 2022c. BEVFormer: Learning Bird's-Eye-View Representation from Multi-Camera Images via Spatiotemporal Transformers. arXiv:2203.17270.

Liao, B.; Chen, S.; Wang, X.; Cheng, T.; Zhang, Q.; Liu, W.; and Huang, C. 2023a. MapTR: Structured Modeling and Learning for Online Vectorized HD Map Construction. arXiv:2208.14437.

Liao, B.; Chen, S.; Zhang, Y.; Jiang, B.; Zhang, Q.; Liu, W.; Huang, C.; and Wang, X. 2023b. MapTRv2: An End-to-End Framework for Online Vectorized HD Map Construction. arXiv:2308.05736.

Lin, T.-Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; and Belongie, S. 2017. Feature Pyramid Networks for Object Detection. arXiv:1612.03144.

Liu, X.; Wang, S.; Li, W.; Yang, R.; Chen, J.; and Zhu, J. 2024a. MGMap: Mask-Guided Learning for Online Vectorized HD Map Construction. *ArXiv*, abs/2404.00876.

Liu, Y.; Yuan, T.; Wang, Y.; Wang, Y.; and Zhao, H. 2023. VectorMapNet: End-to-end Vectorized HD Map Learning. arXiv:2206.08920.

Liu, Z.; Zhang, X.; Liu, G.; Zhao, J.; and Xu, N. 2024b. Leveraging Enhanced Queries of Point Sets for Vectorized Map Construction. arXiv:2402.17430.

Meinhardt, T.; Kirillov, A.; Leal-Taixe, L.; and Feichtenhofer, C. 2022. TrackFormer: Multi-Object Tracking with Transformers. arXiv:2101.02702.

Pan, B.; Sun, J.; Leung, H. Y. T.; Andonian, A.; and Zhou, B. 2020. Cross-View Semantic Segmentation for Sensing Surroundings. *IEEE Robotics and Automation Letters*, 5(3): 4867–4873.

Philion, J.; and Fidler, S. 2020. Lift, Splat, Shoot: Encoding Images From Arbitrary Camera Rigs by Implicitly Unprojecting to 3D. arXiv:2008.05711.

Qiao, L.; Ding, W.; Qiu, X.; and Zhang, C. 2023a. End-to-End Vectorized HD-map Construction with Piecewise Bezier Curve. arXiv:2306.09700.

Qiao, L.; Zheng, Y.; Zhang, P.; Ding, W.; Qiu, X.; Wei, X.; and Zhang, C. 2023b. MachMap: End-to-End Vectorized Solution for Compact HD-Map Construction. arXiv:2306.10301.

Shan, T.; Englot, B.; Meyers, D.; Wang, W.; Ratti, C.; and Rus, D. 2020. LIO-SAM: Tightly-coupled Lidar Inertial Odometry via Smoothing and Mapping. arXiv:2007.00258.

Shazeer, N.; Mirhoseini, A.; Maziarz, K.; Davis, A.; Le, Q.; Hinton, G.; and Dean, J. 2017. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. arXiv:1701.06538.

Shin, J.; Rameau, F.; Jeong, H.; and Kum, D. 2023. InstaGraM: Instance-level Graph Modeling for Vectorized HD Map Learning. arXiv:2301.04470.

Sun, P.; Cao, J.; Jiang, Y.; Zhang, R.; Xie, E.; Yuan, Z.; Wang, C.; and Luo, P. 2021. TransTrack: Multiple Object Tracking with Transformer. arXiv:2012.15460.

Sun, W.; Lin, X.; Shi, Y.; Zhang, C.; Wu, H.; and Zheng, S. 2024. SparseDrive: End-to-End Autonomous Driving via Sparse Scene Representation. arXiv:2405.19620.

Tsai, Y.-H.; Hung, W.-C.; Schulter, S.; Sohn, K.; Yang, M.-H.; and Chandraker, M. 2020. Learning to Adapt Structured Output Space for Semantic Segmentation. arXiv:1802.10349.

Wang, R.; Qin, J.; Li, K.; Li, Y.; Cao, D.; and Xu, J. 2023. BEV-LaneDet: a Simple and Effective 3D Lane Detection Baseline. arXiv:2210.06006.

Wang, S.; Jia, F.; Liu, Y.; Zhao, Y.; Chen, Z.; Wang, T.; Zhang, C.; Zhang, X.; and Zhao, F. 2024. Stream Query Denoising for Vectorized HD Map Construction. arXiv:2401.09112.

Wilson, B.; Qi, W.; Agarwal, T.; Lambert, J.; Singh, J.; Khandelwal, S.; Pan, B.; Kumar, R.; Hartnett, A.; Pontes, J. K.; Ramanan, D.; Carr, P.; and Hays, J. 2023. Argoverse 2: Next Generation Datasets for Self-Driving Perception and Forecasting. arXiv:2301.00493.

Xiong, H.; Shen, J.; Zhu, T.; and Pan, Y. 2024. EAN-MapNet: Efficient Vectorized HD Map Construction with Anchor Neighborhoods. arXiv:2402.18278.

Xu, W.; Wang, H.; Qi, F.; and Lu, C. 2019. Explicit Shape Encoding for Real-Time Instance Segmentation. arXiv:1908.04067.

Xu, Z.; Wong, K.-Y. K.; and Zhao, H. 2024. InsMapper: Exploring Inner-instance Information for Vectorized HD Mapping. arXiv:2308.08543.

Yang, Z.; Liu, M.; Xie, J.; Zhang, Y.; Shen, C.; Shao, W.; Jiao, J.; Xing, T.; Hu, R.; and Xu, P. 2024. MapVision: CVPR 2024 Autonomous Grand Challenge Mapless Driving Tech Report. arXiv:2406.10125.

Yu, J.; Zhang, Z.; Xia, S.; and Sang, J. 2024. ScalableMap: Scalable Map Learning for Online Long-Range Vectorized HD Map Construction. arXiv:2310.13378.

Yuan, T.; Liu, Y.; Wang, Y.; Wang, Y.; and Zhao, H. 2023. StreamMapNet: Streaming Mapping Network for Vectorized Online HD Map Construction. arXiv:2308.12570.

Zhang, C.; Song, Q.; Li, F.; Chen, Y.; and Huang, R. 2024a. HybriMap: Hybrid Clues Utilization for Effective Vectorized HD Map Construction. arXiv:2404.11155.

Zhang, D.; Zhi, P.; Yong, B.; Wang, J.-Q.; Hou, Y.; Guo, L.; Zhou, Q.; and Zhou, R. 2023a. EHSS: An Efficient Hybrid-supervised Symmetric Stereo Matching Network. *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, 1044–1051.

Zhang, G.; Lin, J.; Wu, S.; Song, Y.; Luo, Z.; Xue, Y.; Lu, S.; and Wang, Z. 2023b. Online Map Vectorization for Autonomous Driving: A Rasterization Perspective. arXiv:2306.10502.

Zhang, J.; and Singh, S. 2014. LOAM: Lidar Odometry and Mapping in Real-time.

Zhang, Z.; Zhang, Y.; Ding, X.; Jin, F.; and Yue, X. 2024b. Online Vectorized HD Map Construction using Geometry. arXiv:2312.03341.

Zhou, B.; and Krähenbühl, P. 2022. Cross-view Transformers for real-time Map-view Semantic Segmentation. arXiv:2205.02833.

Zhou, Y.; Zhang, H.; Yu, J.; Yang, Y.; Jung, S.; Park, S.-I.; and Yoo, B. 2024. HIMap: HybrId Representation Learning for End-to-end Vectorized HD Map Construction. arXiv:2403.08639.

Zhu, X.; Zyrianov, V.; Liu, Z.; and Wang, S. 2023. MapPrior: Bird's-Eye View Map Layout Estimation with Generative Models. arXiv:2308.12963.