Supplementary Material for: TopoMaskV3: 3D Mask Head with Dense Offset and Height Predictions for Road Topology Understanding

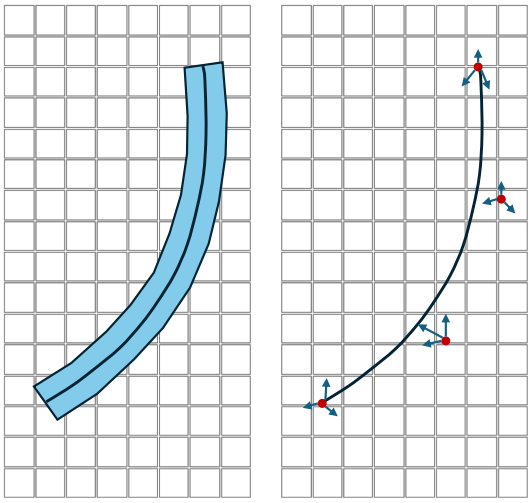

Figure S1. **Attention Mechanisms.** (a) Masked Attention restricts attention to foreground regions using a binary mask. (b) Bezier Deformable Attention (BDA) attends around predicted Bezier control points, enabling flexible and structure-aware feature aggregation.

S. Supplementary Materials

S.1. Masked and Bezier Deformable Attention (Recap)

Masked Attention. To focus attention on relevant regions, Masked Attention introduces a mask term $\boldsymbol{\mathcal{M}}_{l-1}$ into the standard attention formulation:

$$\mathbf{X}_l = \text{softmax}(\boldsymbol{\mathcal{M}}_{l-1} + \mathbf{Q}_l\mathbf{K}_l^{\text{T}})\mathbf{V}_l + \mathbf{X}_{l-1}$$

Here, $\boldsymbol{\mathcal{M}}_{l-1}(x, y) = 0$ for foreground pixels and $-\infty$ otherwise, effectively suppressing background attention. Masks are obtained by thresholding predicted probabilities at 0.5. For detailed information, please refer to the original Mask2Former study [4].

Bezier Deformable Attention. BDA extends deformable attention by replacing single reference points with Bezier control points $\mathbf{C} = \{\mathbf{c}_0, \ldots, \mathbf{c}_N\}$:

$$\text{BDA}(\mathbf{q}, \mathbf{V}, \mathbf{C}) = \sum_{n=0}^{N}\sum_{k=1}^{K} A_{n,k}\mathbf{W}_n\mathbf{V}(\mathbf{c}_n + \Delta\mathbf{p}_{n,k})$$

Each control point acts as an attention head, allowing the model to capture curve geometry without converting control points into dense polylines, reducing overhead and improving structural awareness. For detailed formulations and ablation studies, please refer to the original TopoBDA study [10].

S.2. Mask-L1 Mix Matcher (Recap)

The bipartite matching cost combines regression, mask, and classification terms:

$$\mathcal{L}_{\text{l}} = \lambda_{\text{reg}}\mathcal{L}_{\text{reg}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}} + \lambda_{\text{cls}}\mathcal{L}_{\text{cls}}.$$

When $\lambda_{\text{reg}} = 0$, the matcher relies on mask similarity; when $\lambda_{\text{mask}} = 0$, it becomes an L1-based matcher. Activating both terms yields the *Mask-L1 Mix Matcher*, which follows the principle of MaskDINO [13] by jointly leveraging regression and mask losses in the matching cost.

S.3. Detailed Curve Reconstruction Pipeline

The set of refined 3D points $\{\tilde{\mathbf{p}}_{grid}\}$ generated by the offset proposal (Section 3.3) and height-estimation (Section 3.4) steps is initially in the BEV grid's pixel coordinate system. These points must be transformed and regularized into a smooth, ordered, continuous curve in real-world coordinates. This multi-step process is guided by the **quad-direction label** predicted by the classification head.

S.3.1. Coordinate Transformation

Each refined grid point $\tilde{\mathbf{p}}_{grid} = (i, j, h)$ (where (i, j) is the refined 2D location and h is the estimated normalized height value) is first mapped to its real-world 3D coordinate $\tilde{\mathbf{p}}_{world} = (x, y, z)$ via a pre-defined grid-to-world transformation matrix $\mathbf{V}^{-1}$. This step yields a set of unordered 3D points $\{\tilde{\mathbf{p}}_{world}\}$ in the real-world space, where x represents the vertical (forward) axis and y represents the horizontal (lateral) axis.

S.3.2. Direction-Aware Regularization and Ordering

This set of noisy, real-world points $\{\tilde{\mathbf{p}}_{world}\}$ is processed to form the final smooth curve. This regularization, which involves joint polynomial fitting and arc-length interpolation, is crucial for further smoothing discretization artifacts. The benefits of this approach are experimentally evaluated in Section S.11.

The process involves three main operations:

Polynomial Functions Fit First, polynomial functions are fit to the unordered 3D points $\{\tilde{\mathbf{p}}_{world}\}$. This finds

the best-fit curves that represent the underlying shape of the noisy points. Two separate least-squares fits are performed:

- **2D Path Fit:** A direction-aware polynomial is fit to the (x, y) coordinates based on the quad-direction label such that $y = f(x)$ for 'up'/'down' directions and $x = f(y)$ for 'left'/'right' directions.
- **3D Height Fit:** A 2D polynomial surface, $z = g(x, y)$, is fit to the 3D points (x, y, z) by solving for the coefficients $\mathbf{C}$ of a 3rd-order polynomial:

$$z \approx g(x, y) = C_0 + \sum_{i=1}^{3} (C_{2i-1}x^i + C_{2i}y^i). \quad \text{(S1)}$$

3D Curve Generation and Resampling Second, the 3D curve is generated and resampled. A preliminary 3D polyline, $\mathcal{P}_{poly}$, is generated by sampling the fitted 2D path f at linearly-spaced intervals (e.g., using 'linspace') to get new (x_i, y_i) points. The 3D height surface g is then evaluated at these locations to get the corresponding z_i coordinates. This $\mathcal{P}_{poly}$ is smooth but its points are not equidistant. This 3D polyline is then resampled using **arc-length interpolation** ('interp_arc'), which takes $\mathcal{P}_{poly}$ as input and produces the final, uniformly sampled set of N equidistant 3D points, $\{\tilde{\mathbf{p}}'\} = \{(x'_i, y'_i, z'_i)\}_{i=1}^{N}$.

Final Sorting Finally, this resulting set $\{\tilde{\mathbf{p}}'\}$ is **explicitly sorted** to guarantee the correct directional flow, as defined by the quad-direction label. For instance, points for an 'up' centerline are sorted by increasing x'-coordinates, while 'down' centerlines are sorted by decreasing x'-coordinates. This step ensures the final point sequence is correctly ordered from start to end.

S.4. LiDAR Fusion Pipeline

The sensor fusion pipeline integrates camera and LiDAR data at the voxel level before projecting to a unified BEV representation. This early fusion in 3D space preserves fine-grained spatial information. The process, adapted from [10], is as follows:

S.4.1. Camera Feature Voxelization

A set of N perspective-view images $\{\mathbf{I}_i\}_{i=1}^{N}$ is first processed by a backbone f_{PV} to extract features $\{\mathbf{F}_{PV_i}\}_{i=1}^{N}$. These 2D features are then lifted into a 3D voxel representation $\mathbf{F}_{\text{voxelCam}} \in \mathbb{R}^{H_{\text{bev}} \times W_{\text{bev}} \times Z \times C_{\text{camera}}}$ using a voxelization module f_{voxel} (e.g., Lift-Splat [31]):

$$\mathbf{F}_{\text{voxelCam}} = f_{voxel}(\{\mathbf{F}_{PV_i}\}_{i=1}^{N}). \quad \text{(S2)}$$

S.4.2. LiDAR Feature Voxelization

Concurrently, the raw LiDAR point cloud $\{\mathbf{p}^j_{\text{lidar}}\}_{j=1}^{N_{\text{lidar}}}$ is processed by a dedicated LiDAR encoder f_{lidar} (e.g., SECOND [37]). This module converts the sparse point cloud into a dense, feature-rich voxel grid $\mathbf{F}_{\text{voxelLidar}} \in \mathbb{R}^{H_{\text{bev}} \times W_{\text{bev}} \times Z \times C_{\text{lidar}}}$ that shares the same spatial dimensions:

$$\mathbf{F}_{\text{voxelLidar}} = f_{\text{lidar}}(\{\mathbf{p}^j_{\text{lidar}}\}_{j=1}^{N_{\text{lidar}}}). \quad \text{(S3)}$$

S.4.3. Voxel-Space Fusion

The two feature-rich voxel grids are concatenated along the channel dimension to create a unified fused tensor, $\mathbf{F}_{\text{fused}} \in \mathbb{R}^{H_{\text{bev}} \times W_{\text{bev}} \times Z \times (C_{\text{camera}} + C_{\text{lidar}})}$:

$$\mathbf{F}_{\text{fused}} = \text{concat}(\mathbf{F}_{\text{voxelCam}}, \mathbf{F}_{\text{voxelLidar}}). \quad \text{(S4)}$$

S.4.4. BEV Feature Map Generation

Finally, the height (Z) and channel dimensions of the fused voxels are flattened and passed through a 2D convolutional layer f_{conv2} to produce the final, unified BEV feature map $\mathbf{F}_{\text{bev}} \in \mathbb{R}^{H_{\text{bev}} \times W_{\text{bev}} \times C_{\text{BEV}}}$:

$$\mathbf{F}_{\text{bev}} = f_{\text{conv2}}(\mathbf{F}_{\text{fused}}). \quad \text{(S5)}$$

This BEV map serves as the input to the downstream decoder heads.

S.5. Topology Metric

The TOP_{ll} metric is an Average Precision (AP)-based measure for evaluating directed lane-to-lane topology in graph-structured predictions. A predicted adjacency matrix P_θ is constructed and aligned with the full ground-truth (GT) vertex set V by matching predicted and GT vertices using Fréchet distance thresholds $\Theta = \{1\,\text{m}, 2\,\text{m}, 3\,\text{m}\}$.

For each GT edge:

- If both vertices are matched, the corresponding entry in P_θ is set to the model's predicted edge confidence.
- If either vertex is unmatched, the entry is set to 0 for GT-positive edges and to $0.5 + \varepsilon$ for GT-negative edges. This is intended to penalize missing detections as low-confidence false positives.

For each GT vertex $v \in V$, AP is computed independently for outgoing ($d = \text{out}$) and incoming ($d = \text{in}$) edges. The AP for a given vertex, threshold θ, and direction d is:

$$\text{AP}(v, \theta, d) = \frac{1}{|N_d(v)|} \sum_{\hat{n}' \in \hat{N}'_{\theta,d}(v)} P_{\theta,d}(\hat{n}') \mathbb{I}(\hat{n}' \in N_d(v)) \quad \text{(S6)}$$

where $N_d(v)$ is the set of ground-truth neighbors, $\hat{N}'_{\theta,d}(v)$ is the ranked predicted neighbor list, $P_{\theta,d}(\hat{n}')$ is the precision at that rank, and $\mathbb{I}$ is the indicator function.

The final TOP_{ll} score is the mean of this AP over all thresholds, all vertices, and both directions:

$$\text{TOP}_{\text{ll}} = \frac{1}{2|\Theta||V|} \sum_{\theta \in \Theta} \sum_{v \in V} \sum_{d \in \{\text{in, out}\}} \text{AP}(v, \theta, d) \quad \text{(S7)}$$

Critical Analysis of the 0.5 Confidence Threshold. A significant flaw in the V1.1 metric implementation is that

only predicted edges with **confidence greater than 0.5** are ranked for the AP calculation. This is a non-standard practice for an AP-based metric, which should evaluate the entire precision-recall curve by ranking all predictions. This 0.5 thresholding effectively ignores all predictions in the lower half of the confidence range.

As demonstrated in prior work [11], this flaw can be exploited to artificially inflate scores. Simply remapping low-confidence predictions (e.g., > 0.05) to a value just above the 0.5 threshold can substantially boost performance (e.g., by +10.6 TOP_{ll} for some models [11]) without any change to the model itself. This proves the metric, in its current form, is not a robust measure of performance. The quantitative impact of this remapping is detailed in our Supplementary Section S.10, which provides a direct comparison (Table S3) between the "flawed" V1.1 metric and the "healthy" remapped scores used in our main paper's Table 7.

A proper AP evaluation should rank all predictions. A simple and effective fix would be to lower this ranking threshold from 0.5 to a near-zero value (e.g., **0.01**). Correspondingly, the penalty for unmatched GT-negative edges should be set to $0.01 + \varepsilon$ to ensure they are correctly penalized as low-confidence false positives.

S.6. Loss Functions

The overall training objective is a multi-component loss function. The total loss $\mathcal{L}_{\text{total}}$ is a weighted sum of the primary centerline loss $\mathcal{L}_{\text{l}}$, a traffic element loss $\mathcal{L}_{\text{t}}$ (from DAB-DETR [25]), and topology losses $\mathcal{L}_{\text{ll}}$ and $\mathcal{L}_{\text{lt}}$ (from TopoNet [16]).

S.6.1. Total Loss

The total loss is defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{l}} + \mathcal{L}_{\text{t}} + \mathcal{L}_{\text{ll}} + \mathcal{L}_{\text{lt}}.$$

Our primary contributions are captured within the centerline loss $\mathcal{L}_{\text{l}}$, which is a composite of five distinct terms:

$$\mathcal{L}_{\text{l}} = \lambda_{\text{cls}}\mathcal{L}_{\text{cls}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}} + \lambda_{\text{mask}}\mathcal{L}_{\text{mask}} + \lambda_{\text{offset}}\mathcal{L}_{\text{offset}} + \lambda_{\text{height}}\mathcal{L}_{\text{height}}.$$

S.6.2. Core Centerline Losses

Three components of the centerline loss are adapted from prior work.

Classification Loss ($\mathcal{L}_{\text{cls}}$) A standard cross-entropy loss is applied to each query to predict its class. This includes the four quad-direction labels (as explained in Section 3) and a "no-object" class for empty queries. Due to the dominance of empty queries, the "no-object" class is down-weighted by a factor of 0.1 following [4].

Bezier Regression Loss ($\mathcal{L}_{\text{reg}}$) For the optional Bezier head, a standard L1 regression loss is applied between the N predicted normalized control points $\mathbf{c}_i$ and the ground-truth points $\hat{\mathbf{c}}_i$:

$$\mathcal{L}_{\text{reg}} = \frac{1}{L}\sum_{j=1}^{L}\sum_{i=0}^{N} \|\mathbf{c}_{i,j} - \hat{\mathbf{c}}_{i,j}\|_1,$$

where L is the number of matched ground-truth centerlines.

Instance Mask Loss ($\mathcal{L}_{\text{mask}}$) Following [4], the instance mask loss is a combination of a Binary Cross Entropy (BCE) loss and a Dice loss. This is computed efficiently by sampling K points $\{\mathbf{a}_k\}$ from the predicted probability map $\mathbf{M}_{\text{prob}}$. The total mask loss is the sum of $\mathcal{L}_{\text{BCE}}$ and $\mathcal{L}_{\text{Dice}}$:

$$\mathcal{L}_{\text{BCE}} = \frac{1}{K}\sum_{k=1}^{K} \text{BCE}(\mathbf{M}_{\text{prob}}(\mathbf{a}_k), \mathbf{G}_{\text{map}}(\mathbf{a}_k))$$

$$\mathcal{L}_{\text{Dice}} = 1 - \frac{2\sum_{k=1}^{K}\mathbf{M}_{\text{prob}}(\mathbf{a}_k)\mathbf{G}_{\text{map}}(\mathbf{a}_k)}{\sum_{k=1}^{K}\mathbf{M}_{\text{prob}}(\mathbf{a}_k) + \sum_{k=1}^{K}\mathbf{G}_{\text{map}}(\mathbf{a}_k)}$$

S.6.3. Dense Offset and Height Losses (Ours)

To supervise the new architectural components introduced in the main paper (Sections 3.3 and 3.4), two novel L1-based losses are introduced.

A key aspect of this supervision is the definition of the foreground mask, $\mathbf{M}_{\text{fg}}$. For each pixel (i, j), the L2 norm (Euclidean distance) of its target offset vector $\hat{\mathbf{O}}(i, j)$ is computed. The pixel is included in the foreground mask only if this norm is less than a centerline thickness threshold, which is set to 4 pixels.

$$\mathbf{M}_{\text{fg}}(i, j) = \mathbb{I}\big[\|\hat{\mathbf{O}}(i, j)\|_2 < 4\big]$$

This defines a supervision band, capturing all grid pixels whose closest point on the continuous centerline is up to 4 pixels away.

Offset Loss ($\mathcal{L}_{\text{offset}}$) The dense offset head predicts an offset map $\mathbf{O} \in \mathbb{R}^{H\times W\times 2}$. This is supervised by a masked L1 loss against the target offset map $\hat{\mathbf{O}}$, using the dynamic foreground mask $\mathbf{M}_{\text{fg}}$. The loss is normalized by the number of active foreground pixels:

$$\mathcal{L}_{\text{offset}} = \frac{1}{\sum \mathbf{M}_{\text{fg}}}\sum_{i,j} \|\mathbf{O}(i, j) - \hat{\mathbf{O}}(i, j)\|_1 \cdot \mathbf{M}_{\text{fg}}(i, j).$$

Height Loss ($\mathcal{L}_{\text{height}}$) Similarly, the dense height head predicts a height map $\mathbf{H} \in \mathbb{R}^{H\times W}$. This is supervised by a masked L1 loss against the target height map $\hat{\mathbf{H}}$, using the exact same foreground mask $\mathbf{M}_{\text{fg}}$:

$$\mathcal{L}_{\text{height}} = \frac{1}{\sum \mathbf{M}_{\text{fg}}}\sum_{i,j} |\mathbf{H}(i, j) - \hat{\mathbf{H}}(i, j)| \cdot \mathbf{M}_{\text{fg}}(i, j).$$

S.7. Implementation Details

S.7.1. Architecture Overview

The model architecture is built upon the TopoMaskV2 [11] and TopoBDA [10] frameworks, utilizing distinct, non-weight-sharing backbones for the traffic element and centerline branches. The traffic element branch is based on DAB-DETR [25] and integrates concepts from DN-DETR [12] and DINO [43]. For all experiments in this study, both branches use the ResNet50 [30] architecture.

For BEV feature generation, the multi-height bin Lift-Splat [11, 31] and efficient Voxel Pooling [8] implementations were used. The topology heads project the query embeddings from both branches into a shared space using MLPs, concatenate them, and process the result with a final MLP.

S.7.2. Optimization Parameters

The model was trained with a batch size of 8 using the AdamW optimizer. The base learning rate was set to 3×10^{-4} with a weight decay of 1×10^{-2}. The learning rates for both the PV and BEV backbones were scaled by 0.1. A polynomial learning rate decay schedule (factor 0.9) with 1000 warm-up iterations was used. Gradient norm clipping was set to 35. As detailed in Section S.6, the loss coefficients for the bipartite matcher were: $\lambda_{\text{cls}} = 2$, $\lambda_{\text{reg}} = 5$, $\lambda_{\text{mask}_{BCE}} = 5$, $\lambda_{\text{mask}_{Dice}} = 5$, $\lambda_{\text{offset}} = 20$, and $\lambda_{\text{height}} = 50$.

S.7.3. Architecture Hyperparameters

The BEV grid dimensions ($H_{\text{bev}}, W_{\text{bev}}$) were set to 200 and 104, respectively, at a 0.5m x 0.5m resolution. The vertical dimension (Z) was set to 20 bins, spanning $[-10, 10]$ meters. For ground-truth generation, the centerline mask width was set to 4 pixels, following [10, 11]. When the Bezier branch was active, the number of control points was set to 4 [10, 35]. The number of queries was 200, and the transformer hidden channel dimension was 256. The transformer encoder and decoder were set to 6 and 10 layers, respectively. Multi-scale features were used from PV scales $\frac{1}{8}$, $\frac{1}{16}$, and $\frac{1}{32}$, and BEV scales 1, $\frac{1}{2}$, and $\frac{1}{4}$.

S.7.4. Dataset Preprocessing

Standard preprocessing, including a uniform 0.5× scaling, was applied to all camera inputs. For Subset-A, images were resized to 1024×736 (width $\times$ height) with top crops of 178 pixels (front) or 19 pixels (others) after scaling by $0.5\times$.

S.7.5. Benchmark Release

To support reproducibility of the Argoverse2-derived benchmark, we release the processed annotations, split definitions, prebuilt pkl files, and setup instructions through our OpenLane-V2 benchmark fork[3]. This release covers the **Original**, **Near**, **FarA**, **FarB**, and **FarC** protocols and their corresponding ± 100 m variants. The current public release is centered on these benchmark artifacts and usage instructions rather than the full post-processing code.

S.8. Comparative Ablation of Architectural Enhancements Across Head Types

Table S1. Comparative ablation of architectural enhancements across different head types, with results reported using OLS_l.

Enhancement	Bezier	Mask	Fusion
Base	38.3	40.2	40.6
+ ML1M (from MM)	<u>39.0</u>	<u>41.1</u>	<u>41.5</u>
+ BDA (from MA)	**43.8**	**41.5**	**42.8**

To assess the impact of enhancements introduced in the TopoBDA study [10] when adapted to our framework, we perform a comparative ablation across three head types: Bezier, Mask, and Fusion. Table S1 reports OLS_l scores for the base configuration and two adapted mechanisms—Mask L1 Mix Matcher (ML1M) and Bezier Deformable Attention (BDA).

Although ML1M and BDA are not proposed in this study, they are adapted from the TopoBDA framework to evaluate their compatibility and effectiveness within our mask head design. ML1M yields consistent improvements across all head types, suggesting its general utility. In contrast, BDA shows a particularly strong effect on the Bezier head, likely due to the direct alignment between Bezier regression outputs and the Bezier deformable attention mechanism. This synergy may facilitate more effective gradient flow and feature refinement. While BDA also benefits the Mask and Fusion heads, its most pronounced impact is observed in the Bezier head.

This strong synergistic effect of BDA on the Bezier head is pivotal. As shown in Table S1, in the **Base** (MA) configuration, the Bezier head (38.3 OLS_l) is weaker than the Mask head (40.2 OLS_l), and Fusion (40.6 OLS_l) provides a clear benefit. With the integration of **BDA**, this dynamic reverses: the Bezier head (43.8 OLS_l) becomes the superior component, and Fusion (42.8 OLS_l) is no longer beneficial. This explains why the enhanced standalone Bezier head outperforms the Mask and Fusion heads on the **Original** dataset split. However, this outcome is shown to vary on different data partitions, as a more detailed analysis across dataset splits is presented in the main paper Section 4.5.

S.9. Sensor Fusion Impact on Different Splits and Ranges

Table S2 presents detection performance across different geographic splits, sensor configurations, and perception

[3] https://github.com/artest08/OpenLane-V2/tree/different_splits

Table S2. Detection performance across geographic splits, perception ranges, and sensor configurations. Metrics include DET_l, $DET_{l,ch}$, TOP_{ll}, and OLS_l, with relative improvements from LiDAR integration reported as OLS%. This analysis is conducted using the TopoBDA benchmark setup.

Split	Range	Sensor	DET_l	$DET_{l,ch}$	TOP_{ll}	OLS_l	OLS%↑
Original	±50 m	RGB	37.6	37.7	28.3	42.9	–
Original	±50 m	RGB + LiDAR	**47.0**	**49.8**	**37.0**	**52.6**	**22.61%**
Original	±100 m	RGB	24.1	28.5	20.4	32.6	–
Original	±100 m	RGB + LiDAR	**43.4**	**48.4**	**34.7**	**50.2**	**53.99%**
Near	±50 m	RGB	19.2	24.5	15.2	27.6	–
Near	±50 m	RGB + LiDAR	**24.2**	**31.6**	**18.0**	**32.7**	**18.48%**
Near	±100 m	RGB	12.1	15.6	6.7	17.9	–
Near	±100 m	RGB + LiDAR	**18.4**	**22.5**	**12.2**	**25.3**	**41.34%**

ranges. In this analysis, the TopoBDA [10] architecture has been utilized. Across all settings, incorporating LiDAR consistently improves detection metrics compared to using RGB alone. Notably, the performance gain from LiDAR becomes more pronounced in the extended range setting (±100 m), where RGB-only setups suffer from significant degradation.

In both range settings, LiDAR integration yields consistently higher relative improvements in the **Original** split compared to the **Near** split. Specifically, in the ±50 m range, LiDAR provides a relative gain of **22.61%** in the **Original** split, while the improvement in the **Near** split is only **18.48%**. This trend becomes even more pronounced in the extended ±100 m range, where the relative gain reaches **53.99%** in the **Original** split versus **41.34%** in the **Near** split. These discrepancies suggest that LiDAR-based models may benefit more from geographic overlap, potentially leveraging memorized spatial priors. In contrast, RGB-only models, while less performant overall, exhibit more stable generalization across unseen regions. These findings highlight the importance of evaluating sensor fusion strategies not only by absolute performance but also by their robustness across diverse geographic domains.

S.10. Analysis of Score Remapping on the Near Split

Table S3. Direct comparison of topology and OLS scores with and without score remapping on the Near Split. Note the significant, non-uniform boost from remapping.

Method	Flawed Metric		Remapped Metric	
	TOP_{ll}	OLS_l	TOP_{ll}	OLS_l
TopoNet	6.2	22.4	12.7 (+6.5)	26.0 (+3.6)
TopoMLP	13.0	24.7	14.5 (+1.5)	25.3 (+0.6)
TopoLogic	15.0	26.1	15.5 (+0.5)	26.3 (+0.2)
TopoMaskV2 (M)	6.1	20.4	10.9 (+4.8)	23.2 (+2.8)
TopoMaskV2 (F)	6.6	22.6	11.7 (+5.1)	25.5 (+2.9)
TopoBDA	10.6	26.1	13.0 (+2.4)	27.3 (+1.2)
TopoMaskV3 (M)	5.8	23.0	13.6 (+7.8)	27.3 (+4.3)
TopoMaskV3 (F)	6.6	24.2	15.1 (+8.5)	28.5 (+4.3)

As detailed in the main paper and Supplementary Section S.5, the standard TOP_{ll} metric's 0.5 thresholding flaw prevents a stable comparison of topological reasoning. The main paper's **Section 4.6** (Table 7) presents the SOTA comparison on the 'Near Split' using the corrected, "healthy" score remapping.

To illustrate the impact of this correction, **Table S3** in this section provides a direct side-by-side comparison of the OLS_l and TOP_{ll} scores computed with the **flawed** V1.1 metric versus the **remapped** metric.

Table S3 quantifies the impact of applying the score remapping technique ($P(x) \rightarrow P(x) + 1 \times [P(x) > 0.05]$) [11]. As shown, this yields significantly higher and more representative TOP_{ll} scores for most methods (e.g., TopoNet's score increases from 6.2 to 12.7). However, the impact is non-uniform across different methods. For **TopoLogic**, applying the remapping technique was not feasible, as it already implements its own inherent distance-based score manipulation. To avoid the complexity of compounding these two strategies, a comparable "healthy" result was obtained by simply modifying the standard metric's evaluation threshold from 0.5 to 0.05 for its outputs. Other methods, such as TopoMLP, are also less affected, as their original implementations already produce high-confidence predictions that naturally bypass the flawed 0.5 threshold. This non-uniform impact highlights the instability of the original metric and validates the use of score remapping in the main paper (Table 7) for a more stable and fair comparison.

S.11. Impact of Post-Processing Methods on Instance-Level Point Outputs

Table S4. Performance comparison of post-processing methods. Best scores are bolded, second-best scores are underlined.

Method	DET_l	$DET_{l,ch}$	TOP_{ll}	OLS_l
None	28.1	34.0	17.6	34.7
P2	30.6	37.1	21.9	38.2
P3	32.7	37.4	23.4	39.5
Arc	25.7	33.0	14.5	32.2
P3+Arc	<u>33.1</u>	<u>37.8</u>	<u>25.0</u>	<u>40.3</u>
P4+Arc	**33.3**	**38.0**	**25.8**	**40.7**

To evaluate the effect of different post-processing strategies applied to the proposed instance-level point outputs,

several configurations were tested. These include polynomial fitting of varying degrees and arc interpolation. Polynomial fitting was applied in the vertical-horizontal domain using second-, third-, and fourth-order polynomials. Arc interpolation was also considered both independently and in combination with polynomial fitting. Table S4 summarizes the results across four metrics: DET_{l}, $\text{DET}_{\text{l\_chamfer}}$, TOP_{ll}, and OLS_{l}.

As shown in Table S4, applying polynomial fitting significantly improves performance over the raw point outputs. While arc interpolation alone degrades performance, its combination with polynomial fitting—particularly third- and fourth-order—leads to notable gains. The best overall results are achieved with fourth-order polynomial fitting combined with arc interpolation, indicating that higher-order curve modeling and smooth interpolation are complementary in refining centerline predictions. This also highlights a limitation of the current mask-based formulation: unlike the direct Bezier path, it requires additional thresholding, dense refinement, and curve reconstruction after decoder prediction, and therefore introduces extra post-processing overhead.

S.12. Threshold Sensitivity for Mask-Based Point Selection

Table S5. Performance across different mask probability thresholds. Best scores are bolded, second-best scores are underlined.

Threshold	DET_{l}	$\text{DET}_{\text{l\_ch}}$	TOP_{ll}	OLS_{l}
0.01	30.4	36.6	22.4	38.1
0.10	32.8	37.5	23.9	39.7
0.20	32.9	37.6	24.2	39.9
0.30	<u>33.0</u>	37.7	24.3	40.0
0.40	<u>33.0</u>	37.7	24.5	40.1
0.50	**33.1**	<u>37.8</u>	24.6	<u>40.2</u>
0.60	**33.1**	<u>37.8</u>	24.8	<u>40.2</u>
0.70	**33.1**	<u>37.8</u>	<u>24.9</u>	**40.3**
0.80	**33.1**	<u>37.8</u>	**25.0**	**40.3**
0.90	**33.1**	**37.9**	**25.0**	**40.3**
0.95	**33.1**	**37.9**	**25.0**	**40.3**

To analyze the effect of thresholding on the mask probability map $\mathbf{M}_{\text{prob}}^{(l)}$, a range of values was evaluated to select grid points corresponding to each instance query. This threshold τ (Eq. 2) determines which points are retained from the gridized mask map. Table S5 presents the performance across DET_{l}, $\text{DET}_{\text{l\_ch}}$, TOP_{ll}, and OLS_{l} metrics for varying threshold values.

As shown in Table S5, increasing the threshold improves performance across all metrics, with convergence observed around $\tau = 0.95$. This indicates that higher confidence regions in the mask probability map yield more reliable point selections for instance-level queries, enhancing centerline detection quality.

S.13. Comparison of Lane Divider and Centerline Representations

Table S6. Chamfer distance-based mAP scores ($\text{DET}_{\text{l\_ch}}$) of TopoBDA method for Centerline and Lane Divider representations across five splits in the Argoverse2 HDMap dataset. Best scores per split are highlighted in bold.

Representation	orig	near	farA	farB	farC
Centerline	**39.8**	25.0	20.3	19.1	25.3
Lane Divider	38.9	**27.6**	**22.5**	**20.8**	**25.5**

In addition to centerline representation, lane dividers are also extracted from the HDMap to extend the scope of the analysis. Unlike the OpenLane-V2 dataset, which treats lane dividers as part of the lane segment concept, this work considers lane dividers as independent geometric entities as in studies [15, 22, 26]. Since HDMaps are structured around lane segments, converting them into lane dividers introduces duplication, particularly between adjacent lane segments. To address this, a chamfer distance-based elimination strategy is applied to remove redundant lane divider instances and ensure non-duplicate divider instances.

We conduct a comparative analysis of lane dividers and centerline representations across five distinct geographic splits of the Argoverse2 HDMap dataset. In this analysis, the Bezier head is utilized. The evaluation metric is the Chamfer distance-based mean Average Precision ($\text{DET}_{\text{l\_ch}}$), and the results are summarized in Table S6.

In the **Original** split, which exhibits significant geographic overlap between training and evaluation scenes, the centerline representation achieves the highest performance with an mAP of 39.8, slightly outperforming lane dividers at 38.9. However, in all other splits—**Near**, **FarA**, **FarB**, and **FarC**—lane dividers consistently outperform centerlines. For instance, in the **Near** split, lane dividers achieve an mAP of **27.6** compared to 25.0 for centerlines, and in the **FarA** split, lane dividers score **22.5** versus 20.3 for centerlines.

These results suggest that while centerline representations may benefit from memorization in geographically overlapping regions, they exhibit reduced generalization in unseen areas. In contrast, lane divider representations demonstrate more robust performance across diverse geographic domains, indicating stronger generalization capabilities and reduced susceptibility to overfitting.

S.14. Comparison with SOTA on the Standard (Geographically Overlapping) Original Split

The **Original** split of OpenLane-V2 contains geographic overlap between training and validation sets [24, 42], which can inflate reported performance through memorization (see Sec. 4.6 and Sec. 4.5 in the main paper). We report results on this standard benchmark for completeness and compa-

Table S7. Comparative Evaluation of TopoMaskV3 and other Camera-Only Methods on the **Original** split (**geographically overlapping**) of OpenLane-V2 Subset-A using V1.1 Metric Baseline.

Method	DET_l	DET_t	TOP_{ll}	TOP_{lt}	OLS
STSU [2]	12.7	43.0	2.9	19.8	29.3
VectorMapNet [26]	11.1	41.7	2.7	9.2	24.9
MapTR [22]	8.3	43.5	2.3	8.3	24.2
TopoNet [16]	28.6	48.6	10.9	23.9	39.8
TopoMLP [35]	28.5	49.5	21.7	26.9	44.1
Topo2D [14]	29.1	50.6	22.3	26.2	44.4
TopoLogic [5]	29.9	47.2	23.9	25.4	44.1
RoadPainter [29]	30.7	47.7	22.8	27.2	44.6
TopoFormer [28]	34.7	48.2	24.1	29.5	46.3
TopoMaskV2 (M) [11]	29.6	<u>53.8</u>	20.6	31.9	46.3
TopoMaskV2 (F) [11]	34.5	<u>53.8</u>	24.5	35.6	49.4
TopoBDA [10]	**38.9**	**54.3**	**27.6**	**37.3**	**51.7**
TopoMaskV3 (M) (Ours)	34.7	53.4	23.8	35.4	49.1
TopoMaskV3 (F) (Ours)	<u>35.5</u>	53.4	<u>25.9</u>	<u>36.7</u>	<u>50.1</u>

(M): Mask-based, (F): Fusion-based approaches.
Formatting: **bold** = best, <u>underline</u> = second-best.

rability with prior literature. As discussed in the main paper, **TopoMaskV3 (F)** achieves 50.1 OLS — the second-highest score — surpassing all methods except the Bezier-based TopoBDA [10], whose advantage is attributable to Bezier's higher susceptibility to geographic memorization. The **TopoMaskV3 (M)** standalone head achieves 49.1 OLS, substantially outperforming TopoMaskV2 (M) (46.3 OLS), validating the newly introduced offset and height prediction mechanisms. This improvement surpasses most published works, including TopoFormer [28] and RoadPainter [29].