C-LaV: Conditional Latent Velocity Field Denoising for Weather-Robust LiDAR Place Recognition

Supplementary Material

In this supplementary material, we expand on the main paper by first providing complete architectural and training details for all components of C-LaV, then documenting the construction of our KITTI [9], NCLT [7], and Boreas [5] benchmarks together with representative cross-weather retrieval visualizations, and finally presenting additional ablations. Sec. 7 presents the full implementation details of our C-LaV, including the network and descriptor architecture (Sec. 7.1), the flow-matching and ODE configuration (Sec. 7.2), and the training setup for KITTI, NCLT, and Boreas (Sec. 7.3). Sec. 8 details the construction of our unified adverse-weather benchmark, including the weather simulation for KITTI and NCLT (Sec. 8.1), the BEV generation pipeline (Sec. 8.2), spatial train/test splits (Sec. 8.3), and per-weather BEV statistics and visual comparisons (Sec. 8.4). Sec. 9 presents additional ablations on the key components of C-LaV. Unless otherwise stated, we follow the notation and experimental settings of the main paper. To build intuition before diving into the implementation details, Fig. 7 shows a representative successful cross-weather retrieval on KITTI under heavy rain, where all top-5 retrieved clear-weather BEVs are correct within 10 m. In contrast, Fig. 8 presents a typical failure case under dense fog, illustrating that visually similar but geographically mismatched trajectories can still lead to occasional false positives.

7. Implementation Details and Training Setup

7.1. Network and Descriptor Architecture

BEV encoder (Stage 1). Unless otherwise stated, all experiments use the same BEV encoder. For each single-sweep LiDAR scan, we crop the points to a fixed bird's-eye-view (BEV) region around the sensor and voxelize them on a regular 2D grid, obtaining 3-channel BEV images of size 448×448 encoding height, reflectance intensity, and point density (see Sec. 3.3 of the main paper and Sec. 8.2 for the exact construction). After computing these three channels, we apply a single normalization scheme across KITTI, NCLT, and Boreas: each channel is standardized using dataset-wide statistics computed on the training split, and the standardized values are clipped to a robust range. Empty cells (no points) are encoded as zeros before normalization. We do not feed any explicit BEV occupancy masks to the network; instead, we account for input sparsity through a spatial weighting mask in the conditional Flow Matching (diffusion) loss described in Sec. 7.2 ("Spatial weighting and stabilization"), which down-weights empty background cells when computing the loss. The normalized BEVs are then fed to a frozen DINOv2-B/14 transformer-based BEV encoder: patch embedding with 14×14 patches produces a 32×32 token grid, and the encoder outputs a latent tensor

$$\mathbf{z} \in \mathbb{R}^{B \times 768 \times 32 \times 32}.$$

This latent tensor provides a compact 2D encoding of the 3D scene, is shared across all datasets, weather conditions, and model variants (baselines, ablations, and full C-LaV), and is produced by an encoder that remains frozen during Stage 2 (denoising) and Stage 3 (descriptor training).

Latent flow-matching denoiser (Stage 2). Following the probability path z_t defined in Eq. (8) of the main paper, C-LaV learns a conditional velocity field in the latent space of the BEV encoder. The denoiser operates on the 32×32 latent grid $\mathbf{z}_t \in \mathbb{R}^{B \times 768 \times 32 \times 32}$ and is parameterized by a DiT-style transformer. We first apply a 2D patch embedding that reshapes $\mathbf{z}_t$ into $32 \times 32 = 1024$ latent tokens and linearly projects each 768-D channel vector to a 384-D token embedding. These tokens are processed by a 12-block 2D DiT with hidden size 384, 6 attention heads, MLP ratio 4, RoPE 2D positional encodings, RMSNorm, and SwiGLU activations. A final linear projection maps the tokens back to 768 dimensions, which are then reshaped to the original $B \times 768 \times 32 \times 32$ layout. The scalar time $t \in (0, 1)$ is encoded using Gaussian Fourier features and injected into all blocks via AdaLN-style modulation. For conditioning, we globally pool the noisy latent $\mathbf{Z}_{\text{noisy}}$ into a vector, project it to a condition embedding, concatenate it with the time embedding, and broadcast the result to all blocks. The network outputs a velocity field $F_\theta(\mathbf{z}_t, t, \mathbf{Z}_{\text{noisy}})$ along the path, which is used both for the conditional Flow Matching loss in Eq. (11) and for test-time ODE integration in Eq. (12) of the main paper.

SALAD descriptor head (Stage 3). Given an (optionally denoised) latent tensor $\mathbf{z} \in \mathbb{R}^{B \times 768 \times 32 \times 32}$, the SALAD head produces a global descriptor as follows. First, a 1×1 convolution followed by global pooling yields a 256-D global token $\mathbf{t} \in \mathbb{R}^{B \times 256}$. Two additional 1×1 convolutions produce

$$\mathbf{f} \in \mathbb{R}^{B \times 128 \times 32 \times 32}, \quad \mathbf{p} \in \mathbb{R}^{B \times 64 \times 32 \times 32},$$

where $\mathbf{f}$ are local features and $\mathbf{p}$ parameterizes the assignment scores of $K = 64$ soft clusters over $N = 32 \times 32 = 1024$

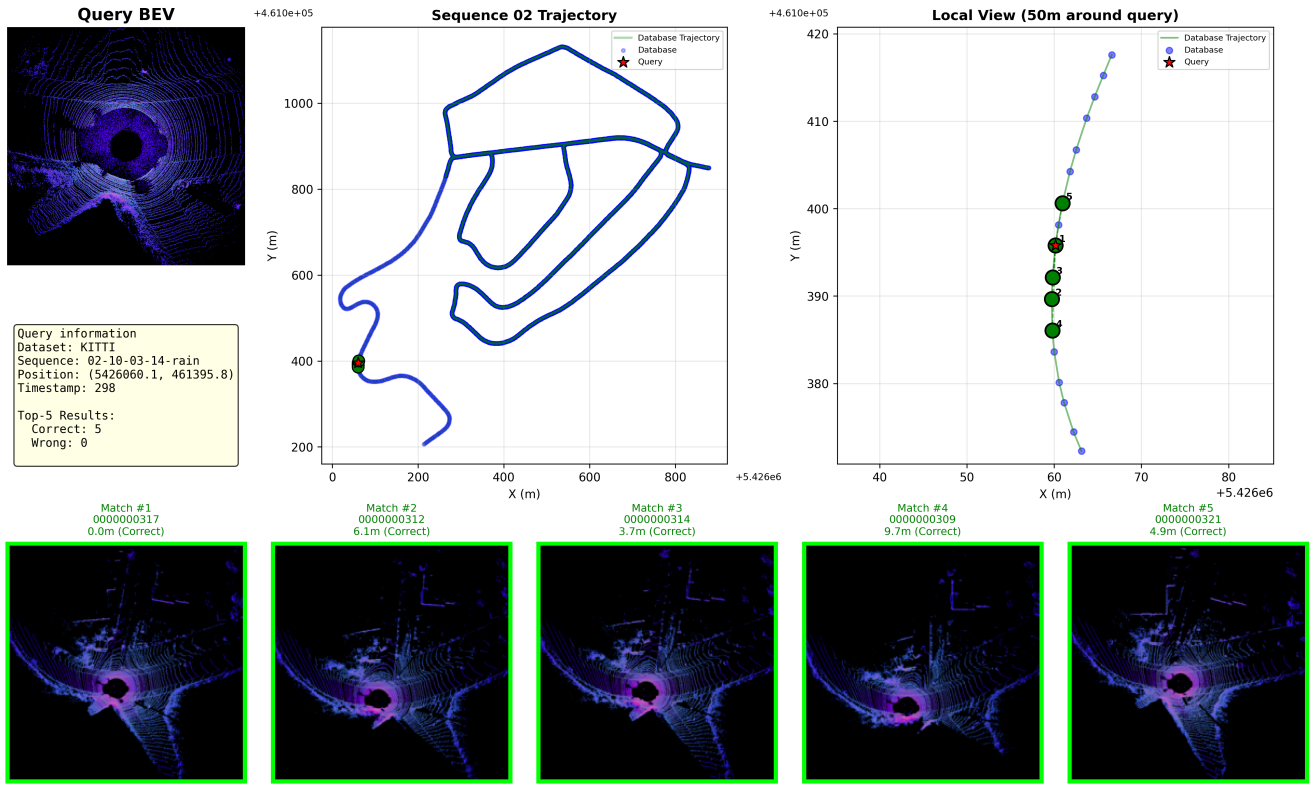

Figure 7. Qualitative cross-weather retrieval example on the KITTI dataset under rainy conditions. The top-left panel shows the query BEV image and its metadata, the center panel shows the full database trajectory with the query position marked, and the top-right panel zooms into a 50 m neighborhood around the query. The bottom row visualizes the top-5 retrieved database BEVs (clear-weather reference); the green borders indicate correct matches, and the captions report their frame IDs and metric position errors. All top-5 results are correct within 10 m, demonstrating that our BEV descriptor remains geometrically consistent even in heavy rain.

spatial locations. After flattening to $\mathbf{f}_{\text{flat}} \in \mathbb{R}^{B\times 128\times 1024}$ and $\mathbf{p}_{\text{flat}} \in \mathbb{R}^{B\times 64\times 1024}$, we follow SALAD [15] and interpret, for each sample, $\mathbf{p}_{\text{flat}}$ as a score matrix $\mathbf{S} \in \mathbb{R}^{N\times(K+1)}$ after adding a dustbin column with a learnable parameter. Let $\boldsymbol{\mu} = \mathbf{1}_N$ and $\boldsymbol{\kappa} = [\mathbf{1}_K^\top, N-K]^\top$ denote the feature and (cluster+dustbin) mass, respectively. We then apply the Sinkhorn algorithm on $\exp(\bar{\mathbf{S}})$ to obtain an optimal-transport assignment $\bar{\mathbf{P}} \in \mathbb{R}^{N\times(K+1)}$ satisfying the same marginal constraints as in SALAD:

$$\bar{\mathbf{P}}\,\mathbf{1}_{K+1} = \boldsymbol{\mu}, \qquad \bar{\mathbf{P}}^\top \mathbf{1}_N = \boldsymbol{\kappa}, \tag{23}$$

and finally drop the dustbin column to get $\mathbf{P} \in \mathbb{R}^{N\times K}$. Soft-cluster aggregation then yields K cluster descriptors in $\mathbb{R}^{128}$,

$$\mathbf{v}_k \;=\; \sum_{i=1}^{N} \mathbf{P}_{i,k}\,\mathbf{f}_i \;\in\; \mathbb{R}^{128}, \quad k = 1,\dots,K, \tag{24}$$

which are concatenated with the 256-D global token and ℓ_2-normalized, resulting in an 8448-D descriptor:

$$D_{\text{desc}} = 256 + 128 \times 64 = 8448.$$

All retrieval experiments use cosine similarity on these descriptors.

7.2. Flow-matching and ODE configuration

Probability path and target velocity. As in the conditional Flow Matching formulation of the main paper (Eqs. (8)–(9)), we define, for each training pair, an affine interpolation path between a Gaussian reference latent and the clean BEV latent. Given a clean latent $\mathbf{z}_1 = \mathbf{Z}_{\text{clean}}$ and a reference sample $\mathbf{z}_0 \sim \mathcal{N}(0, I)$, the path is

$$\mathbf{z}_t = \big(1 - (1-\sigma_{\min})t\big)\,\mathbf{z}_0 + t\,\mathbf{z}_1, \qquad t \in [0,1], \tag{25}$$

where $\sigma_{\min} \in [0,1)$ controls the residual noise at the end of the path. For t=1 this yields a slightly perturbed clean latent $\mathbf{z}_1 + \sigma_{\min}\mathbf{z}_0$. The associated ground-truth velocity (Eq. (9) in the main paper) is the time derivative

$$v_t(\mathbf{z}_t \mid \mathbf{z}_1) = \frac{\partial \mathbf{z}_t}{\partial t} = \mathbf{z}_1 - (1-\sigma_{\min})\,\mathbf{z}_0, \tag{26}$$

which is constant in t for fixed $(\mathbf{z}_0, \mathbf{z}_1)$. This straight-line interpolation is the 2-Wasserstein optimal-transport geodesic between $\mathbf{z}_0$ and $\mathbf{z}_1$, so v_t can be viewed as an optimal-transport conditional velocity field.

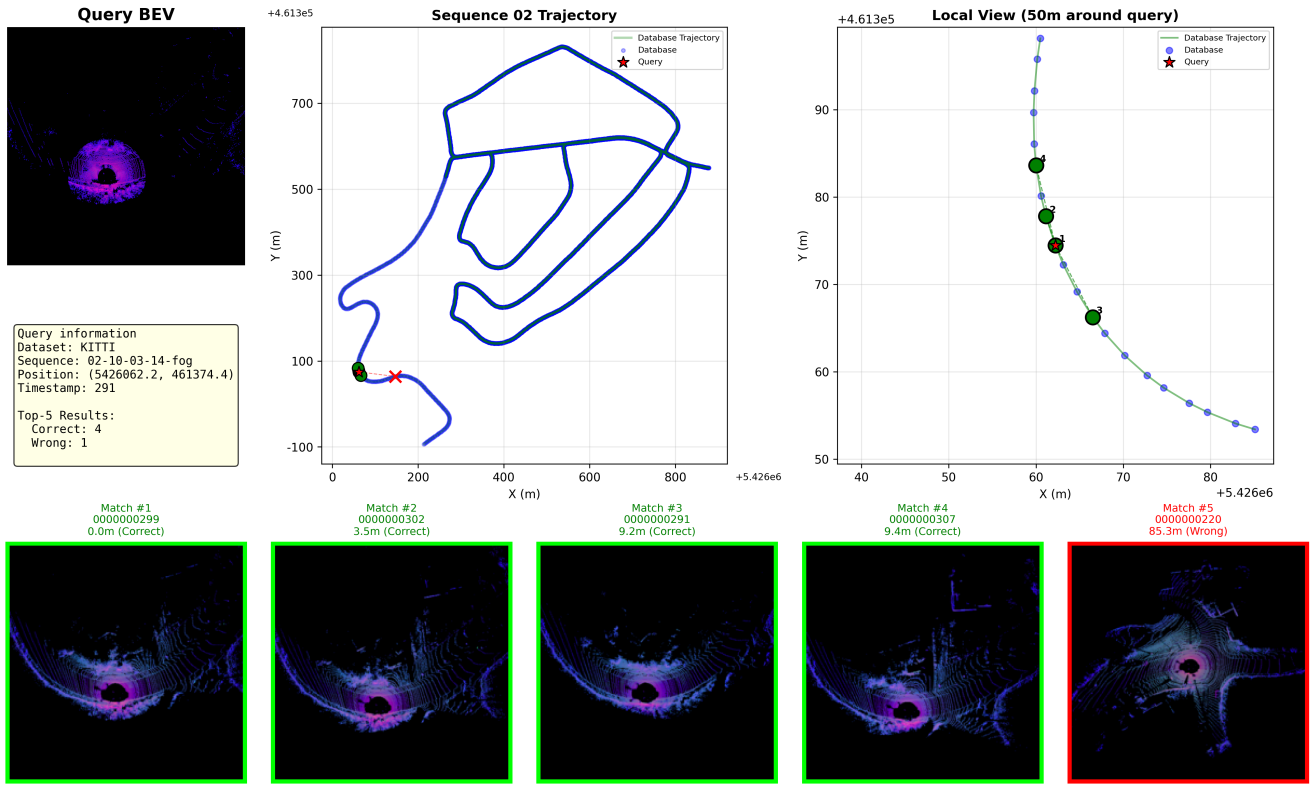

Figure 8. Failure case of cross-weather retrieval on the KITTI dataset under foggy conditions. This example illustrates that, although most top-ranked results are geometrically consistent, challenging views in dense fog can still lead to occasional false positives.

Conditional formulation and Flow Matching loss. For every BEV pair $(X_{\text{noisy}}, X_{\text{clean}})$, the frozen encoder E produces latents $\mathbf{Z}_{\text{noisy}} = E(X_{\text{noisy}})$ and $\mathbf{Z}_{\text{clean}} = E(X_{\text{clean}})$. During training we sample $\mathbf{z}_0 \sim \mathcal{N}(0, I)$, set $\mathbf{z}_1 = \mathbf{Z}_{\text{clean}}$, draw $t \sim \mathcal{U}(0, 1)$, and evaluate $\mathbf{z}_t$ on the path above. The DiT denoiser implements a *conditional* velocity field

$$\hat{v}_t = F_\theta(\mathbf{z}_t, t, \mathbf{Z}_{\text{noisy}}), \tag{27}$$

where the noisy latent $\mathbf{Z}_{\text{noisy}}$ is used as an external conditioning signal. We train F_θ with a conditional Flow Matching objective,

$$\mathcal{L}_{\text{CFM}} = \mathbb{E}\Big[\big\| F_\theta(\mathbf{z}_t, t, \mathbf{Z}_{\text{noisy}}) - v_t(\mathbf{z}_t \mid \mathbf{z}_1{=}\mathbf{Z}_{\text{clean}}) \big\|_2^2\Big], \tag{28}$$

where the expectation is taken over paired latents $(\mathbf{Z}_{\text{noisy}}, \mathbf{Z}_{\text{clean}})$, Gaussian samples $\mathbf{z}_0$, and times t. Here the analytical target velocity $v_t(\mathbf{z}_t \mid \mathbf{z}_1)$ depends on the clean latent through the interpolation path $(\mathbf{z}_0, \mathbf{z}_1)$, while the conditioning BEV latent $\mathbf{Z}_{\text{noisy}}$ enters only through the parametrized field F_θ. This follows the standard conditional FM design: the clean latent defines the target distribution, and the network learns a velocity field that uses $\mathbf{Z}_{\text{noisy}}$ to transport Gaussian samples towards the corresponding scene. At test time the same conditioning latent $\mathbf{Z}_{\text{noisy}}$ is used when integrating the probability-flow ODE, so the denoised latent is a deterministic function of the input BEV.

Time sampling strategy. Unless otherwise stated, we sample t uniformly from $(0, 1)$. We also implement a logit-normal time sampling option in our codebase, but in practice we observed no consistent benefit over uniform sampling, so all reported results use the uniform scheme.

ODE solver and number of steps. At inference, we follow the probability-flow ODE in Eq. (12) of the main paper and interpret F_θ as the right-hand side of

$$\frac{\mathrm{d}\mathbf{z}_t}{\mathrm{d}t} = F_\theta(\mathbf{z}_t, t, \mathbf{Z}_{\text{noisy}}), \qquad t \in [0, 1], \tag{29}$$

with initial condition $\mathbf{z}_0 \sim \mathcal{N}(0, I)$ and fixed conditioning $\mathbf{Z}_{\text{noisy}}$. We discretize this ODE with an explicit Euler sampler using a fixed number of steps T:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{1}{T} F_\theta(\mathbf{x}_k, t_k, \mathbf{Z}_{\text{noisy}}), \qquad t_k = \frac{k}{T}, \tag{30}$$

initialized at $\mathbf{x}_0 = \mathbf{z}_0$ and iterated until $t_T = 1$. Unless otherwise stated, all main results use $T{=}50$ steps.

Spatial weighting and stabilization. Because BEV inputs are highly sparse, we incorporate a spatial weighting mask into the conditional Flow Matching (diffusion) loss to down-weight empty background cells while preserving full weight on road and structure regions. The weighting mask is pre-computed from the BEV occupancy: background cells are assigned a weight of 0.1 in all datasets, and this weighting is applied only to the diffusion / Flow Matching loss term. We additionally apply gradient clipping (global norm 1.0) and use mixed-precision evaluation during ODE integration to avoid numerical instability on extreme latents.

7.3. Training setup for KITTI, NCLT, and Boreas

We adopt a nearly unified optimization setup across KITTI, NCLT, and Boreas, and expose all dataset-specific options via YAML configuration files that will be released with the code.

Stage 2. In this phase, we train the latent flow-matching denoiser on paired BEVs $(X_{\text{noisy}}, X_{\text{clean}})$ for each dataset.

- **Epochs and batch size.** For KITTI, NCLT, and Boreas we train the denoiser for 100 epochs using a per-GPU batch size of 16 and gradient accumulation of 2, yielding an effective batch size of 32. The number of iterations per epoch differs only due to dataset size.
- **Optimizer and scheduler.** We use AdamW with a learning rate of 1.0×10^{-4} and weight decay 0.01. The learning rate follows a cosine-annealing learning-rate schedule with warm restarts (T_0=20, T_{mult}=2, $\eta_{\min}$=1.0×10^{-7}) and 5 warm-up epochs.
- **EMA and regularization.** We maintain an exponential moving average of the denoiser weights with decay 0.9999 and use it for evaluation. Gradients are clipped to a global norm of 1.0, and mixed-precision training is enabled for stability.

Stage 3. Stage 3 trains the SALAD descriptor head with the Truncated Smooth-AP loss under the unified retrieval protocol.

- **Epochs and batch size.** We train for 50 epochs on all datasets with an effective batch size of 64: on KITTI and NCLT we directly use batch size 64, whereas on Boreas we use batch size 64 with gradient accumulation of 2 to match the same effective batch size under GPU memory constraints.
- **Frozen modules.** During Stage 3, the BEV encoder and the Stage 2 denoiser are kept frozen, and only the descriptor head is updated.
- **Optimizer and scheduler.** We again use AdamW with a learning rate of 1.0×10^{-4} and weight decay 0.01, together with the same cosine-annealing learning-rate schedule with warm restarts as in Stage 2.
- **Loss function.** We optimize the descriptor head with the Truncated Smooth-AP loss using temperature τ_1=0.01, `positives_per_query`= 4, a positive distance threshold of 10 m, and a negative distance threshold of 50 m.

Hardware. All experiments were conducted on a workstation with an AMD® EPYC™ 7K62 CPU, eight NVIDIA® GeForce RTX™ 5090 GPUs (32,GB VRAM each), and 252,GB RAM, using a PyTorch implementation of our C-LaV pipeline. Unless otherwise specified, all models are trained and evaluated on these eight GPUs with mixed-precision.

Efficiency Analysis. We additionally report the accuracy–runtime trade-off under different Euler ODE integration steps for the latent denoiser. As shown in Tab. 4, increasing the number of ODE steps improves retrieval performance but also increases inference time. We choose 50 steps as the default, which achieves a good balance between accuracy and computational efficiency.

Table 4. Accuracy–runtime trade-off under different ODE steps.

Metric	0 (w/o)	10	20	30	50	100
KITTI Avg. R@1	29.32%	41.39%	50.77%	57.47%	**62.83**%	63.13%
Time	189 ms	203 ms	217 ms	237 ms	**246** ms	337 ms

8. Unified Adverse-Weather Benchmark Protocol

In this section we formalize our unified adverse-weather benchmark protocol, describe how we construct the BEV-based datasets, and provide per-weather statistics and visualizations for KITTI, NCLT, and Boreas. Unless otherwise noted, all experiments in the main paper follow this protocol.

8.1. Datasets and Weather Simulation

We consider three LiDAR benchmarks covering both synthetic and real adverse weather.

KITTI and NCLT (synthetic weather). For KITTI [9] and NCLT [7], we start from the original clear-weather LiDAR sequences and generate corresponding adverse-weather variants using physically based simulation models [10, 11]. Concretely, for each raw scan we use the published FogSim and SnowSim pipelines (and their rain extension) to corrupt the clear point cloud with attenuation, scattering, and spurious returns that mimic real fog, snow, and rain. The simulation is applied in the LiDAR sensor frame using the original range/intensity values and calibrated poses. This produces aligned clear/adverse pairs

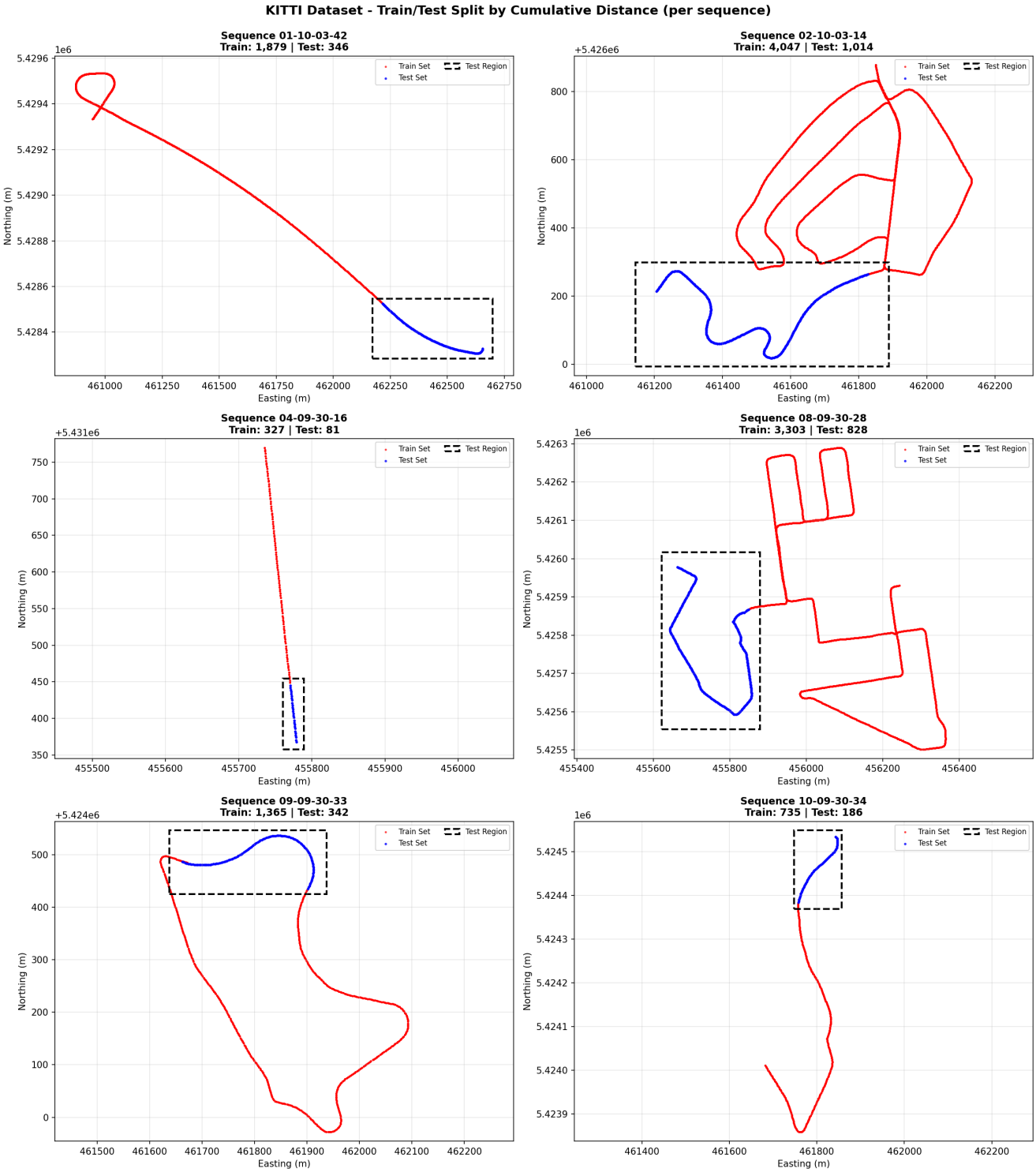

Figure 9. Spatial distribution of the KITTI sequences used in our benchmark.

along identical trajectories, which we later project to BEVs (Sec. 8.2). These pairs form the input to the Stage 2 denoiser on the training regions and the query–database tuples for retrieval on the disjoint test regions. All simulation parameters (e.g., meteorological visibility, snow density, and rain rate) are stored in configuration files and will be released with our code to enable exact reproduction.

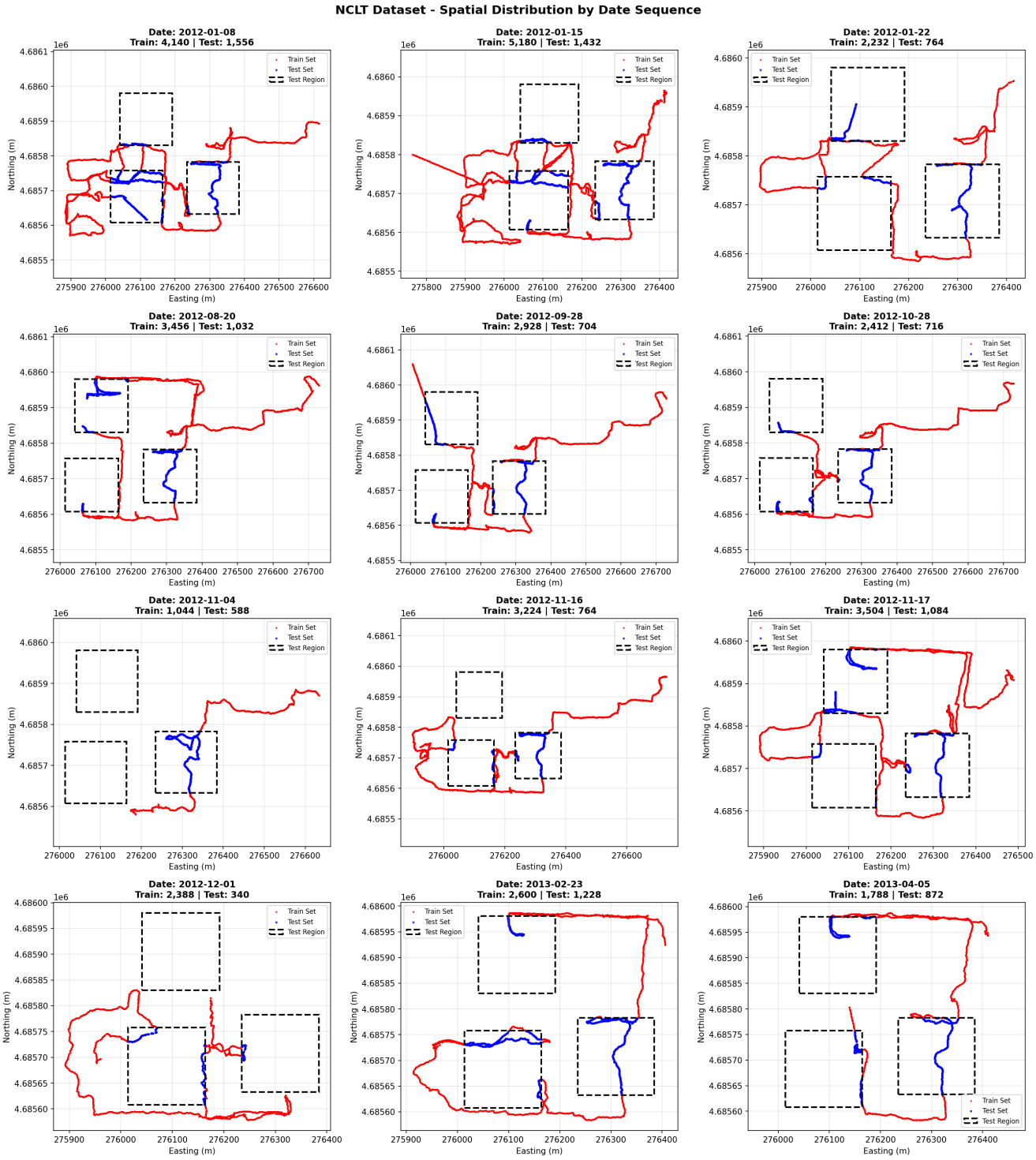

Figure 10. Spatial distribution of the NCLT sequences used in our benchmark.

8.1.1. Weather Simulation Details

Our degraded point cloud simulation is built upon well-established physical models. For instance, rain simulation follows the Marshall-Palmer raindrop distribution, snow uses geometric optics for occlusion and reflection, and fog is modeled via Mie scattering for extinction and backscattering. All exact simulator knob values are summarized in Tab. 5. Unless otherwise noted, we use the *Heavy* setting.

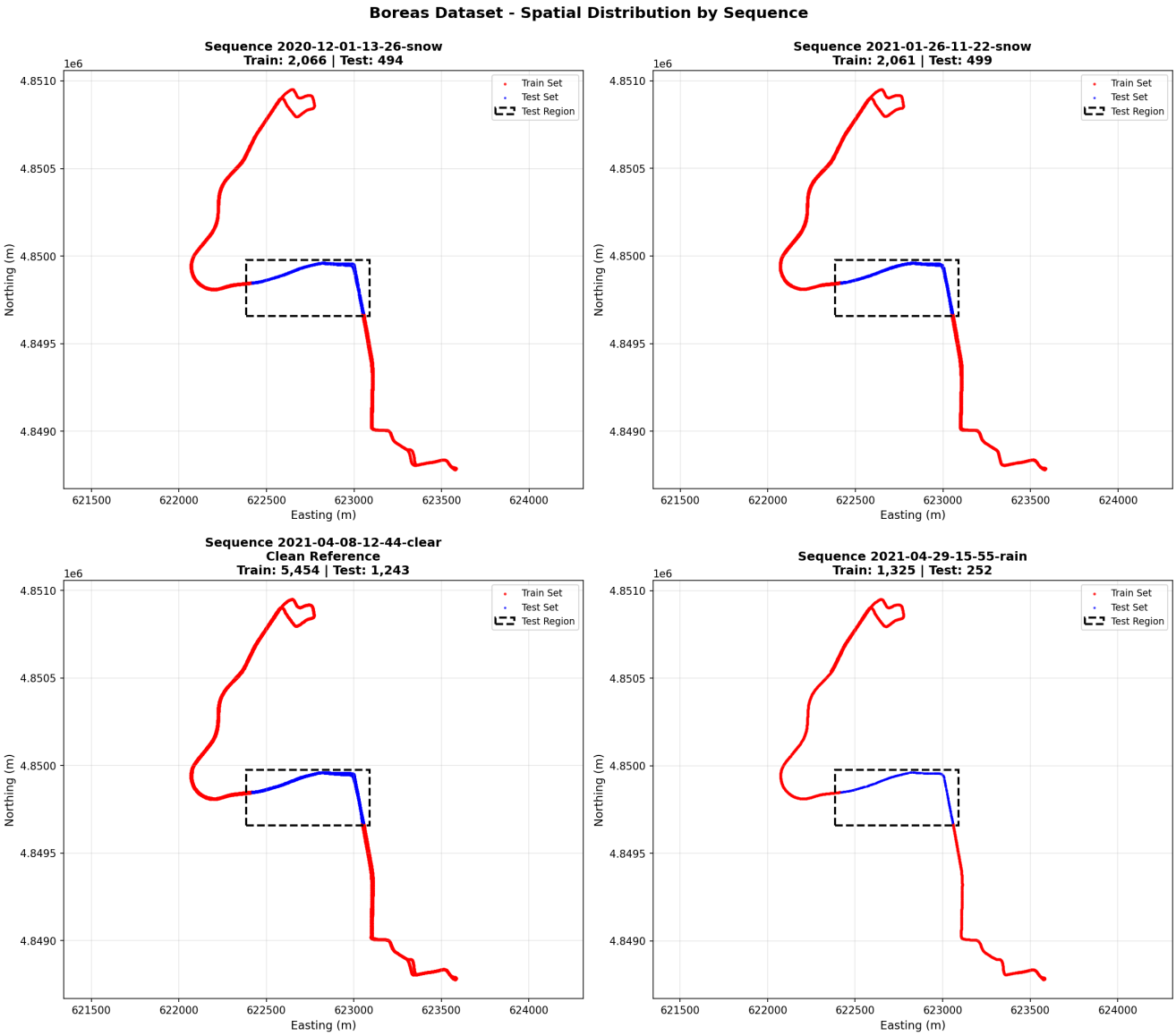

Figure 11. Spatial distribution of the Boreas sequences used in our benchmark.

Table 5. Exact parameter settings used in weather simulation.

Severity	Fog (α m^{-1}; β; MOR m; LT)	Rain (LISA) (rain_rate (N_0); Equiv.Intensity mm/h)	Snow (Gunn–Marshall) (snowfall_rate mm/h; v_t m/s; rain_rate*; occupancy*)
Light	0.01; 0.2; ∼391; LT−1	1000; ∼5	2; 2.0; 17.91; 2.78×10^{-6}
Moderate	0.03; 0.2; ∼100; same table(near α)	4000; ∼20	10; 2.0; 200.21; 1.39×10^{-5}
Heavy (chosen)	0.06; 0.2; ∼50; same table(near α)	8000; ∼50	50; 2.0; 2238.38; 6.94×10^{-5}

* Derived internally by the simulator from snowfall_rate and v_t.

Boreas (real weather). For Boreas [5], we directly use multi-weather sequences (clear, rain, snow) whose trajectories overlap in space. No simulation is applied: adverse-weather artifacts (e.g., heavy snow sparsity, rain streaks) are purely physical. We only perform geometric alignment using the provided poses and constrain the evaluation region to overlapping areas across weather conditions (see Fig. 11).

8.2. BEV Generation Pipeline

As described in Sec. 3.3 of the main paper and Sec. 7.1 of this supplementary material, we represent each LiDAR scan P by a three-channel bird's-eye-view (BEV) raster $I = \phi(P) \in \mathbb{R}^{3\times H\times W}$ encoding height, reflectance intensity, and point density (see Fig. 5 of the main paper). The same BEV pipeline is applied to clear and adverse-weather scans across KITTI, NCLT, and Boreas.

Projection and per-cell statistics. Given a point cloud with points $(x, y, z, \text{intensity})$, we first crop it to a fixed BEV window centered at the sensor. The horizontal plane (x, y) is discretized into a regular grid with cell size on the order of $0.2\,\text{m}$; cells outside the region of interest are discarded. For each grid cell p, we aggregate per-point statistics into three raw channels:

- **Height.** The height channel is computed as the normalized maximum z-value per cell:

$$H_p = \begin{cases} \text{round}\left(255 \cdot \dfrac{\max(z_n) - z_{\min}}{z_{\max} - z_{\min}}\right), & M_p = 1, \\ 0, & M_p = 0, \end{cases} \tag{31}$$

  where $M_p = 1$ indicates that at least one point falls into cell p, and $(z_{\min}, z_{\max})$ denotes the global vertical range.
- **Intensity.** The intensity channel stores the mean reflectance in each occupied cell, rescaled to $[0, 1]$ using dataset-wide statistics computed on the training set; background cells (M_p=0) are set to zero.
- **Density.** The density channel encodes the (log-scaled) number of points per cell. We count the points in each occupied cell, normalize by the global maximum count, and optionally apply a logarithm to compress heavy tails; empty cells are again set to zero.

This construction yields a raw $3 \times H \times W$ BEV raster with H=W=448 in all experiments. For reproducibility, the exact cropping ranges, cell size, and dataset-specific constants will be exposed in our configuration files.

Global standardization and clipping. After constructing the height, intensity, and density channels as above, we further standardize each channel using the dataset-wide mean and standard deviation computed on the training split and clip the standardized values to a robust range, as described in Sec. 7.1. The resulting normalized BEVs are fed to the frozen DINOv2-based BEV encoder and are used consistently across all datasets, weather conditions, and experiments.

8.3. Spatial Train/Test Splits and Trajectory Visualization

To make cross-weather retrieval results comparable, we enforce a consistent spatial splitting strategy and visualize the resulting trajectories.

Frame spacing and tuple construction. For all datasets, we sub-sample the LiDAR streams with a uniform spacing of 3 m along the vehicle trajectory. This is done after BEV generation using the ground-truth poses (converted to UTM coordinates), so that each retained frame corresponds to approximately 3 m of traveled distance along the path. For retrieval, we construct query–database tuples by treating frames within 10 m as positives and frames farther than 50 m as negatives; frames in $[10, 50]$ m are ignored when computing retrieval metrics and Smooth-AP masks.

KITTI and NCLT trajectories. Figs. 9 and 10 show the spatial distribution of the selected KITTI and NCLT sequences in UTM coordinates. For each sequence, we plot the 3 m-sampled vehicle trajectory and color-code segments according to their split (red: train, blue: test). The dashed black rectangle(s) in each subplot indicate the spatial test region: blue segments lying inside the box are reserved for evaluation, whereas red segments outside the box are used for training. These plots are generated directly from the poses associated with the BEV frames, ensuring that what is visualized exactly matches the data used in our experiments.

Boreas shared evaluation region. For Boreas, we additionally enforce a *single* common evaluation region shared across all weather sequences. As illustrated in Fig. 11, we plot two snow sequences (top row), one clear reference sequence (bottom-left), and one rain sequence (bottom-right) in UTM coordinates. The dashed black rectangle marks the common evaluation box; in all four subplots, blue segments inside the box are used only for Stage 3 geo-localization evaluation (queries and database), while red trajectory segments outside the box provide the pool of frames from which we build Stage 2 surrogate denoising pairs. This design yields a shared spatial test area across weather conditions while keeping the denoiser training region strictly outside the evaluation box.

Unified train/test protocol across datasets and stages. Across all three datasets (KITTI, NCLT, and Boreas) we adopt a unified spatial train/test protocol that is shared by Stage 2 (latent denoiser training) and Stage 3 (descriptor learning and retrieval). Concretely, Stage 2 is always trained *only* on frames that lie in the training regions (red trajectories outside the dashed evaluation boxes in Figs. 9–11), while all retrieval metrics in the main paper are computed exclusively on the disjoint test regions (blue trajectories inside the boxes). For KITTI and NCLT, denoising pairs and place-recognition tuples are constructed from the same 3 m-sampled streams and then partitioned spatially into train and test using the per-sequence boxes. For Boreas, Stage 2 denoising pairs are constructed only from the "outside box" training segments, whereas Stage 3 queries and database frames come solely from the shared evaluation box ("inside box"). This protocol ensures that Stage 2 training and Stage 3 evaluation are strictly disjoint in space across all datasets and weather conditions, avoiding any form of train–test leakage.

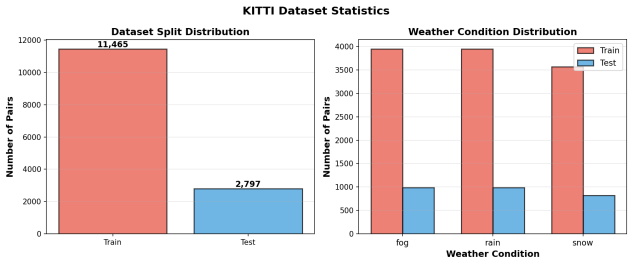

Figure 12. Per-weather BEV statistics on the KITTI dataset.

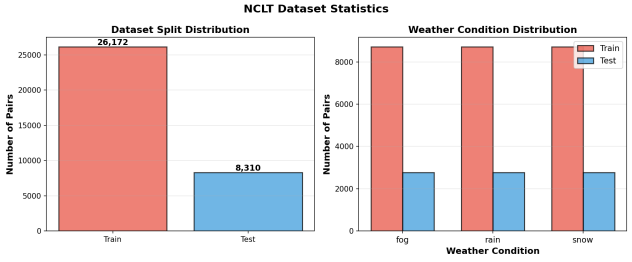

Figure 13. Per-weather BEV statistics on the NCLT dataset.

8.4. Per-Weather BEV Statistics and Visual Comparisons

BEV counts and statistics. Figs. 12, 13, and 14 summarize the per-weather BEV statistics for KITTI, NCLT, and Boreas, respectively. To generate these plots, we count the number of BEV frames after 3 m sub-sampling for each combination of dataset split (train/test), weather condition (clear, rain, fog, snow), and task (place-recognition tuples vs. denoising pairs). These statistics are computed directly from the BEV directories used by our training scripts and are thus guaranteed to be consistent with the experiments reported in the main paper.

Weather-wise BEV comparison. To qualitatively illustrate how weather affects the BEV inputs, Figs. 15, 16, and 17 show BEV snapshots of the same (or closely aligned) locations across different weather conditions. These images are produced by directly visualizing the three-channel BEVs after normalization: height is mapped to intensity, while reflectance and density are combined into pseudo-color channels for better contrast. On KITTI and NCLT, the simulated fog and snow reduce returns at long range and blur lane/structure boundaries, whereas rain introduces additional near-range clutter. On Boreas, real snow causes severe sparsity and missing structure, and rain corrupts the intensity patterns on reflective surfaces. These visualizations highlight the severity and diversity of adverse-weather artifacts and motivate the need for a latent denoising mechanism such as C-LaV: the raw BEV inputs alone are often insufficient for weather-robust place recognition.

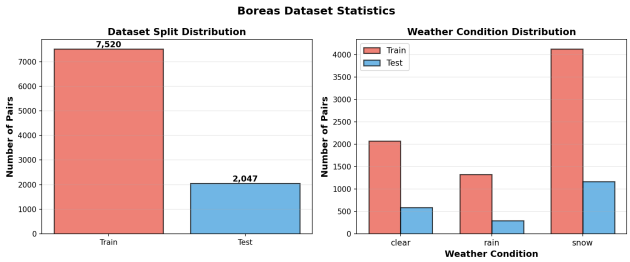

Figure 14. Per-weather BEV statistics on the Boreas dataset.

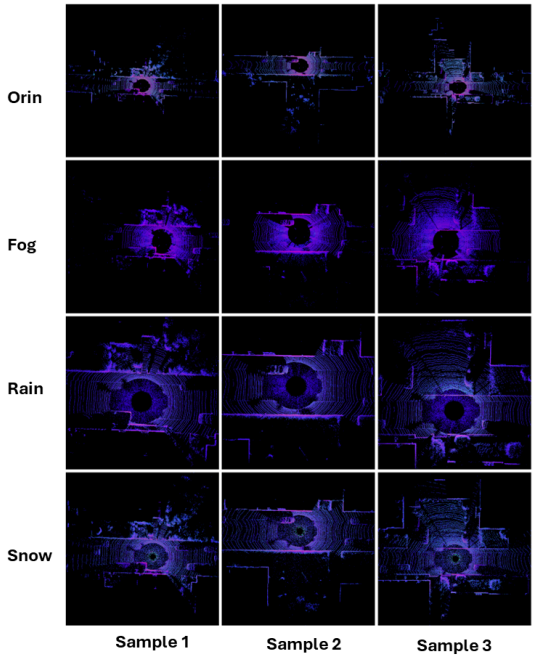

Figure 15. Weather comparison of the KITTI sequences used in our benchmark.

9. Ablations on Denoising Strategies

In this section, we provide additional ablations on the denoising component of C-LaV to complement the ablations in Sec. 4.3 of the main paper. Our goal is to answer two key questions: (i) *Is latent-space denoising necessary compared to directly using noisy BEVs?* (ii) *Does latent flow-matching offer advantages over a standard BEV image-space denoiser?* Unless otherwise noted, we follow the unified evaluation protocol described in Sec. 8.

9.1. Experimental Setup

We conduct a focused ablation on KITTI under adverse weather conditions (fog, rain, snow). All variants use the same BEV encoder architecture (adapted DINOv2-B) and the same SALAD descriptor design; only the denoising module between them differs (no denoiser, BEV U-Net, or C-LaV (Latent Flow-Matching)). For each variant, we train the corresponding denoiser (if present) and SALAD

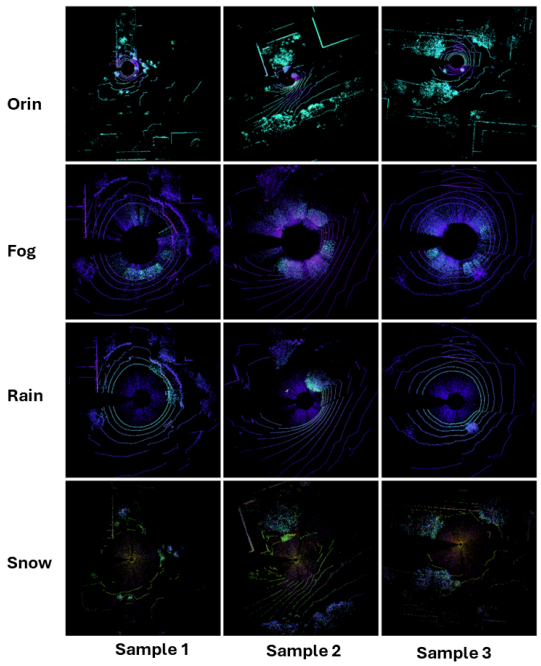

Figure 16. Weather comparison of the NCLT sequences used in our benchmark.

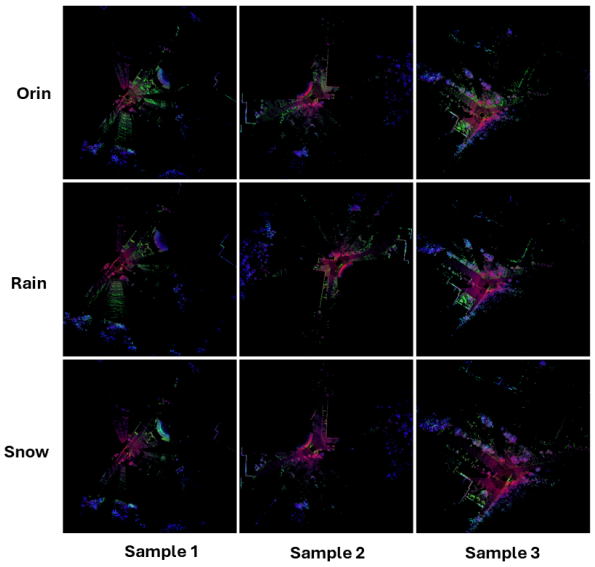

Figure 17. Weather comparison of the Boreas sequences used in our benchmark.

head on KITTI using the unified Stage 3 training protocol of Sec. 7.3, while keeping the BEV encoder frozen, so that the comparison isolates the effect of the denoising strategy.

Task A: No Denoising. The first baseline directly feeds noisy BEVs to the encoder without any denoising. Formally, we compute descriptors as

$$\mathbf{d} = \mathrm{SALAD}(f_{\mathrm{enc}}(X_{\mathrm{noisy}}))\,.$$

This variant corresponds to a strong BEV-based place recognition model with no explicit weather compensation.

Task B: BEV Image-Space U-Net. The second baseline performs denoising in BEV image space via a lightweight U-Net. Given a noisy BEV X_{noisy}, the U-Net predicts a denoised BEV $\hat{X}_{\mathrm{clean}}$, and descriptors are defined as

$$\mathbf{d} = \mathrm{SALAD}\Big(f_{\mathrm{enc}}(\hat{X}_{\mathrm{clean}})\Big)\,.$$

The U-Net is trained with an ℓ_2 loss to regress from adverse BEVs to their clear-weather counterparts; all other components are frozen.

C-LaV (Latent Flow-Matching). Our full model performs denoising in the latent space via a conditional flow-matching velocity field and an ODE-based sampler. Descriptors are obtained by

$$\mathbf{d} = \mathrm{SALAD}(\phi_{\mathrm{den}}(f_{\mathrm{enc}}(X_{\mathrm{noisy}})))\,,$$

Table 6. Ablation on denoising strategies on KITTI under adverse weather (fog/rain/snow). All methods share the same BEV encoder and SALAD head; only the denoising module differs. We report Recall@1/5/10 averaged over all adverse-weather conditions.

Method	R@1	R@5	R@10
Task A: No Denoising	29.32%	53.13%	62.28%
Task B: BEV U-Net	37.00%	62.09%	71.62%
C-LaV (Latent Flow-Matching)	**62.84%**	**83.10%**	**89.25%**
Δ vs Task A	**+33.52**	**+29.97**	**+26.97**
Δ vs Task B	**+25.84**	**+21.01**	**+17.63**

where ϕ_{den} denotes the learned latent flow-matching denoiser with $T{=}50$ Euler steps at inference time. All three methods are evaluated under the same 3 m frame spacing and positive/negative distance thresholds (10 m / 50 m).

9.2. Adverse-Weather Average Performance

Table 6 reports Recall@1/5/10 on KITTI averaged over fog, rain, and snow. Compared to the strong "No Denoising" baseline (Task A), C-LaV improves Recall@1 by more than 33 absolute points. BEV image-space denoising (Task B) also yields noticeable gains, but remains substantially behind latent flow-matching, highlighting the benefit of operating in the encoder latent space.

From Table 6, we observe that: (i) explicit denoising is crucial—Task A fails to provide robust cross-weather retrieval, especially at rank-1; (ii) BEV image-space denoising (Task B) improves Recall@1 by +7.68 points over Task A but is still far from the +33.52 points achieved by

Table 7. **Sinkhorn assignment mass in SALAD.** BG/FG denote background/foreground tokens. $p_{\text{dustbin}} + p_{\text{valid}} = 1$.

Mass	Latent (BG)	Latent (FG)	Signal (BG)	Signal (FG)
p_{dustbin}	0.823	0.186	0.512	0.478
p_{valid}	0.177	0.814	0.488	0.522

Table 8. Per-weather ablation on KITTI: Recall@1/5 for fog, rain, and snow. C-LaV consistently outperforms both the no-denoising baseline and the BEV U-Net denoiser, with particularly strong gains under real heavy degradation (snow and rain).

Weather	No Denoising	BEV U-Net	C-LaV (Ours)
Fog	40.79% / 68.79%	44.44% / 72.22%	**62.12% / 86.16%**
Rain	13.03% / 30.40%	19.19% / 40.00%	**47.07% / 68.89%**
Snow	34.15% / 60.22%	47.37% / 74.05%	**79.31% / 94.25%**

C-LaV; (iii) the gap between BEV-space and latent-space denoising persists at R@5 and R@10, indicating that latent flow-matching consistently enhances the entire retrieval ranking.

9.3. Sinkhorn Assignment Mass Analysis (Domain-gap Metric)

To quantify how weather corruption affects the semantic assignment in SALAD, we report the Sinkhorn assignment mass to the *dustbin* (background) and to *valid* clusters (foreground) for both latent-space and signal-space features.

As shown in Tab. 7, latent features preserve a much cleaner separation: background tokens are mostly assigned to the dustbin while foreground tokens stay mostly valid, whereas signal-space features become ambiguous.

9.4. Per-Weather Breakdown

To understand how different denoising strategies behave under specific weather types, Table 8 reports Recall@1/5 separately for fog, rain, and snow. C-LaV shows large gains in all three conditions, with the largest improvement observed under snow, which is the most challenging setting.

Weather-wise, we see that: under fog, C-LaV improves Recall@1 from 40.79% (no denoising) and 44.44% (BEV U-Net) to 62.12%; under rain, which severely corrupts intensity and creates clutter, C-LaV nearly quadruples Recall@1 from 13.03% to 47.07%; under snow, the improvement is even more pronounced, from 34.15% to 79.31%. These results indicate that latent flow-matching denoising is particularly effective when the input BEVs are heavily degraded, and that operating in latent space is more robust than directly denoising BEV images.

Overall, the ablations in this section support our design choices: (i) latent denoising is essential for adverse-weather LiDAR place recognition, and (ii) latent flow-matching with an ODE sampler substantially outperforms a strong BEV-space U-Net denoiser under a unified encoder and descriptor architecture.