SoDPCC: Shared-Octree Coding for Dynamic LiDAR Point Cloud Compression

Anonymous authors
Paper under double-blind review

Abstract

Dynamic LiDAR compression aims to compactly encode continuously evolving 3D scenes, while existing codecs mainly exploit temporal redundancy by compressing consecutive frames in a frame-wise manner. However, this paradigm leaves temporal correlation only implicitly reflected in the statistical similarity between adjacent encoded frames, which limits the effective exploitation of temporal information. To address this limitation, we propose SoDPCC, a learned LiDAR stream compression framework comprising a shared octree and inter-frame presence states, which respectively reduce spatial symbols through hierarchical geometry sharing and temporal uncertainty through compact state modeling. Specifically, SoDPCC constructs a shared octree for two consecutive frames to remove duplicated geometry across hierarchy levels, encodes each leaf's inter-frame presence as a compact state, and models the resulting occupancy and state streams with dedicated Transformer-based entropy models. For temporal-state coding, we design Progressive Grouped State Factorization to progressively expose denser decoded state cues for subsequent prediction. We further propose Full-Spatial State-Geometry Coupling to integrate these cues with spatially aligned geometry features and propagate the resulting state-aware context over the complete leaf sequence. The experimental results show that our SoDPCC outperforms state-of-the-art methods, achieving average BD-rate reductions of 41.78% and 49.43% on SemanticKITTI and Ford datasets, respectively. Moreover, SoDPCC preserves competitive vehicle-detection performance on reconstructed LiDAR point clouds, demonstrating its practical utility for downstream perception.

1 Introduction

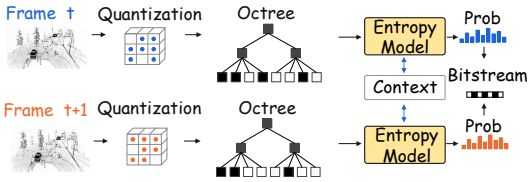
LiDAR systems serve as a fundamental sensing source for three-dimensional perception, supplying precise geometric observations to autonomous vehicles, robots, and large-scale mapping applications (Lin et al., 2025; Giacomini et al., 2025; Ikeda et al., 2026). However, modern LiDAR sensors continuously generate large-scale sparse point cloud streams, which impose substantial storage and transmission costs (Cui et al., 2024). Efficient LiDAR point cloud compression (LPC) is therefore essential. Unlike static LPC, which mainly removes spatial redundancy within individual frame, dynamic LiDAR compression usually exploits temporal correlations across consecutive sweeps. Successive sweeps preserve much of the same scene geometry, while ego-motion, occlusion, and object dynamics introduce localized changes. How to represent and compress such temporal redundancy remains a key challenge.

Existing dynamic LPC methods mainly exploit temporal redundancy through motion-guided prediction and reference-conditioned entropy modeling. Motion-aware methods such as RTST-LPCC (Feng et al., 2020) use IMU-derived transformations. However, sparse sampling, occlusion, and independently moving objects make accurate alignment difficult in outdoor LiDAR scenes (Luo et al., 2025). Recent methods including MUSCLE (Biswas et al., 2020), STAEM (Song et al., 2023), and DAPCC (Cui et al., 2025) instead aggregate historical occupancy, hierarchical octree context, or reference-frame features to estimate current-frame symbol probabilities. As shown in Fig. 1a, these methods still encode each frame with an individual geometry representation, leaving shared octree nodes and ancestor paths repeatedly coded. More importantly, frame-wise conditioning entangles inter-frame correspondence and temporal evolution with occupancy prediction, forcing the entropy

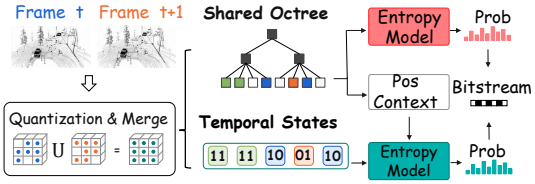
Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping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

model to explain persistence, emergence, and disappearance within the same symbol distribution, which increases probability-modeling ambiguity.

To overcome these limitations, we propose SoDPCC, a shared-octree and temporal-state framework that improves LiDAR stream compression at both the representation and probability-modeling levels. Specifically, SoDPCC first builds a shared octree for two consecutive frames to remove duplicated hierarchical geometry and assigns each leaf a compact state, turning temporal evolution into an explicit and conditionable signal on the shared spatial support. Then, GeoContextNet and TemporalNet are developed to encode the shared-geometry and temporal-state streams, respectively. To exploit the local dependencies among these states, we design a Progressive Grouped State Factorization (PGSF) that progressively provides denser decoded state cues across stages. We further introduce Full-Spatial State-Geometry Coupling (FSGC) in TemporalNet, which fuses these cues with spatially aligned geometry features and propagates the resulting state-aware representation over the complete leaf sequence for more accurate probability estimation. As shown in Fig. 1b, SoDPCC replaces two separate geometry streams with one shared octree and a compact state stream, simultaneously reducing the number of spatial symbols and the uncertainty of temporal symbols.



(a) Frame-wise temporal coding.



(b) The proposed SoDPCC coding paradigm.

Figure 1: Comparison between frame-wise temporal coding and the proposed SoDPCC paradigm for LiDAR point clouds. Frame-wise coding represents consecutive frames with separate geometries, while SoDPCC merges two frames into a unified geometry with temporal presence states and compresses geometry and temporal states.

Our contributions are summarized as follows:

- We propose SoDPCC, a learned LiDAR stream compression framework, which represents two consecutive frames with a shared octree and compact states to eliminate duplicated hierarchical geometry. Based on this representation, GeoContextNet models hierarchical dependencies for shared occupancy coding, while TemporalNet estimates the conditional distributions of temporal states.
- To improve temporal-state probability estimation, we introduce Full-Spatial State-Geometry Coupling to integrate reconstructed states with spatially aligned geometry features, and Progressive Grouped State Factorization to progressively enrich the full-spatial context for subsequent state prediction.
- Extensive evaluations on SemanticKITTI and Ford demonstrate state-of-the-art rate-distortion performance, with average BD-rate reductions of 41.78% and 49.43% over GPCC, while maintaining competitive vehicle-detection accuracy on reconstructed KITTI point clouds.

2 Related Work

2.1 Single-Frame LiDAR Point Cloud Compression

Single-Frame LiDAR point cloud compression primarily targets geometric redundancy within a single scan. MPEG GPCC (Graziosi & Tabatabai, 2020) organizes occupied space with an octree and codes node occupancies using manually designed contexts. Learned methods replace these handcrafted probability estimators with neural entropy models to capture more complex dependencies. Sparse-tensor codecs represent geometry on voxel grids and extract multiscale contexts with sparse convolutions. SparsePCGC (Wang et al., 2023) establishes a learned sparse-tensor framework for geometry coding, while Unicorn V2 (Wang et al., 2025a) and RENO (You et al., 2025) further improve feature extraction and entropy estimation across scales. AnyPCC (Wang et al., 2026) enhances cross-density generalization through universal context modeling and instance-adaptive fine-tuning,

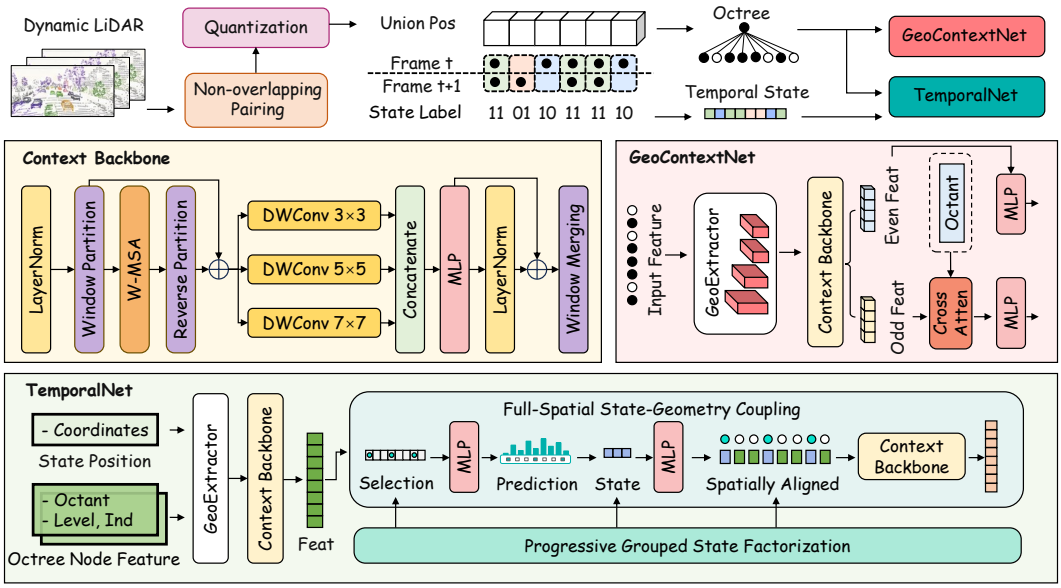


Figure 2: Overview of SoDPCC. Two consecutive LiDAR sweeps are quantized and merged into a shared octree over union coordinates. The shared occupancy is compressed by GeoContextNet, and compact frame-membership states are encoded by TemporalNet with progressive grouped state factorization and full-spatial state-geometry coupling.

whereas ELiC (Kim et al., 2026) improves hierarchical coding efficiency via cross-bit-depth feature reuse and adaptive encoder selection. Such methods benefit from efficient sparse computation, but fixed voxel structures and convolutional neighborhoods may be less adaptive to the strongly varying density of outdoor LiDAR scans. Octree-based methods employ hierarchical partitioning to reduce spatial redundancy effectively. OctSqueeze (Huang et al., 2020) models ancestor and sibling contexts with an MLP-based entropy model. OctAttention (Fu et al., 2022) subsequently introduces masked attention over serialized octree nodes to capture broader dependencies. Building on this attention paradigm, OctFormer (Cui et al., 2023) improves efficiency through non-overlapping windows, while DuOctree (Cui et al., 2024) and TopNet (Wang et al., 2025b) further enhance cross-tree and multi-scale context modeling for LiDAR geometry. Despite these advances, their representation and models are designed mainly for individual scans, leaving correlations between consecutive LiDAR frames unexplored.

2.2 Dynamic Point Cloud Compression

Dynamic point cloud compression extends single-frame coding by exploiting temporal dependencies across successive scans. Motion-compensated residual coding is widely used for dense object-level sequences (Xia et al., 2023; Pan et al., 2024; Akhtar et al., 2024; Zhang & Gao, 2025), but the sparse and nonuniform sampling of LiDAR makes direct inter-frame XOR residuals highly fragmented and difficult to compress efficiently. For sparse LiDAR streams, RTST-LPCC (Feng et al., 2020) uses ego-motion cues from an IMU to align consecutive frames. However, nonuniform sampling and independently moving objects make fine-grained correspondence unreliable in LiDAR scenes. To reduce dependence on explicit alignment for LiDAR point clouds, recent neural codecs use previously reconstructed frames as side information for entropy prediction. STAEM (Song et al., 2023) exploits large-scale spatio-temporal octree contexts through geometry-aware graph feature extraction and attention-based dependency modeling. Along this line, MDLPCC (Luo et al., 2025) incorporates proxy motion descriptions for inter-frame prediction. DAPCC (Cui et al., 2025) establishes temporal correspondences via aggregation embedding and further captures local–global dependencies using diverse attention and U-MLP enhancement. Despite improved entropy prediction, these methods remain frame-wise codecs, which repeatedly represent shared geometry and mix temporal changes with occupancy modeling, increasing the ambiguity of the predicted symbol distribution. SoDPCC differs by changing the coded representation rather than only enriching the temporal context. Its

shared octree reuses common structures, which directly shortens the geometry symbol sequence before entropy coding. SoDPCC further assigns each shared leaf a frame-membership state, converting inter-frame evolution into an explicit temporal field. Since neighboring leaves exhibit strong local dependencies, decoded states can condition subsequent predictions, producing more concentrated probability distributions and reducing the uncertainty of the temporal stream.

3 METHOD

We propose SoDPCC to compress dynamic LiDAR frames using a shared octree and an inter-frame presence-state field. As illustrated in Fig. 2, the quantized coordinates of two consecutive frames are unified to construct a shared octree, eliminating duplicated hierarchical geometry between frames. Its occupancy symbols are modeled and entropy-coded by GeoContextNet to exploit hierarchical spatial dependencies. The inter-frame presence of each leaf is then represented by a compact temporal state, and the resulting state stream is modeled by TemporalNet. Within TemporalNet, we propose Progressive Grouped State Factorization to progressively expose neighboring state cues while limiting serial decoding latency. We further introduce Full-Spatial State-Geometry Coupling to integrate decoded state cues with spatially aligned geometry features for more accurate probability estimation. Finally, the two frames are deterministically reconstructed from the decoded shared octree and temporal states.

3.1 SHARED OCTREE AND TEMPORAL REPRESENTATION

We propose a shared-octree and temporal representation, which removes repeated ancestor nodes across hierarchy levels to reduce the number of symbols, and represents temporal variation as a compact state field to reduce conditional uncertainty through local state correlation. As illustrated in Fig. 2, the continuous LiDAR stream is partitioned into non-overlapping windows of two consecutive frames for joint coding. For a two-frame window, we apply a shared quantization operator $\mathcal{Q}(\mathbf{p})$ as follows:

$$\mathcal{Q}(\mathbf{p}) = \left\lfloor \frac{\mathbf{p}}{q_s} + 0.5 \right\rfloor, \tag{1}$$

where $\mathbf{p}$ denotes the coordinate of a point in either frame, and $q_s = S/(2^D - 1)$ is the quantization step determined by the spatial range S and target octree depth D. The two frames are converted into discrete voxel sets $\mathcal{V}_0, \mathcal{V}_1 \subset \mathbb{Z}^3$. No ego-pose registration or motion compensation is applied before constructing the shared octree. We construct their union $\mathcal{V}^m$ as follows:

$$\mathcal{V}^m = \mathcal{V}_0 \cup \mathcal{V}_1 = \left\{ \mathbf{v} \in \mathbb{Z}^3 \mid \mathbf{v} \in \mathcal{V}_0 \vee \mathbf{v} \in \mathcal{V}_1 \right\}, \tag{2}$$

and build a shared octree $\mathcal{T}^m = \mathcal{T}(\mathcal{V}^m)$ over the union set, where $\mathcal{T}(\cdot)$ denotes the octree construction operator. Consequently, coincident leaves, intermediate nodes, and shared ancestor paths are represented only once.

The shared geometry specifies whether a spatial location appears in at least one frame, but does not retain its frame membership. We therefore define a three-valued temporal state $\mathbf{a}(\mathbf{v})$ for each shared leaf $\mathbf{v}$:

$$\mathbf{a}(\mathbf{v}) = [\mathbb{I}(\mathbf{v} \in \mathcal{V}_0), \mathbb{I}(\mathbf{v} \in \mathcal{V}_1)] \in \{(10), (01), (11)\}, \tag{3}$$

where 10, 01, and 11 denote first-frame-only, second-frame-only, and shared occupancy, respectively. The all-zero state 00 is excluded because every leaf of $\mathcal{T}^m$ belongs to at least one frame. Let $\mathbf{X}^m = \{\mathbf{x}_i\}_{i=1}^{N_m}$ denote the shared-octree occupancy sequence of N_m nodes, where each $\mathbf{x}_i \in \{00000000, \ldots, 11111111\}_2$ is an 8-bit child-occupancy code. Let $\mathbf{A} = \{\mathbf{a}_j\}_{j=1}^{N_a}$ denote the temporal-state sequence defined on the N_a reconstructed leaves. According to Shannon's source coding theorem (Shannon, 1948), the ideal code length of SoDPCC is given by:

$$\begin{aligned} R_{\text{SoDPCC}} &= -\log_2 \left[p_\theta(\mathbf{X}^m) p_\phi \left(\mathbf{A} \mid \widehat{\mathcal{T}}^m \right) \right] \\ &= -\log_2 p_\theta(\mathbf{X}^m) - \log_2 p_\phi \left(\mathbf{A} \mid \widehat{\mathcal{T}}^m \right) \\ &= R_{\text{geo}} + R_{\text{state}}, \end{aligned} \tag{4}$$

where $\widehat{\mathcal{T}}^m$ denotes the shared octree reconstructed from the occupancy stream. The distributions p_θ and p_ϕ model the shared-octree occupancy sequence and the temporal state sequence. R_{geo} and

R_{state} denote the ideal geometry and state code length. After entropy decoding, the two frames are recovered deterministically:

$$\widehat{\mathcal{V}}_0 = \left\{ \mathbf{v} \in \widehat{\mathcal{V}}^m \mid \hat{\mathbf{a}}(\mathbf{v}) \in \{(10), (11)\} \right\},$$
$$\widehat{\mathcal{V}}_1 = \left\{ \mathbf{v} \in \widehat{\mathcal{V}}^m \mid \hat{\mathbf{a}}(\mathbf{v}) \in \{(01), (11)\} \right\}. \tag{5}$$

where $\widehat{\mathcal{V}}^m$ is the decoded shared voxel set, $\hat{\mathbf{a}}(\mathbf{v})$ is the decoded state of voxel $\mathbf{v}$, and $\widehat{\mathcal{V}}_0$ and $\widehat{\mathcal{V}}_1$ are the reconstructed voxel sets of the two frames.

3.2 GeoContextNet and TemporalNet

As illustrated in Fig. 2, SoDPCC employs GeoContextNet and TemporalNet for shared octree coding and temporal-state modeling, respectively. GeoContextNet parameterizes the shared-occupancy distribution $p_\theta(\mathbf{X}^m)$, while TemporalNet models the conditional state distribution $p_\phi(\mathbf{A} \mid \widehat{\mathcal{T}}^m)$.

An octree is constructed from the merged voxel set $\mathcal{V}^m$ and serialized into an occupancy sequence $\mathbf{X}^m = \{x_i\}_{i=1}^{N_m}$. Since octree occupancy symbols exhibit strong local and hierarchical dependencies, GeoContextNet uses GeoExtractor to embed their causal structural cues, including ancestor occupancies, octree levels, octant indices, spatial extents, and node coordinates. Internally, GeoExtractor embeds these individual geometric features and employs K-Nearest Neighbors to aggregate local contexts from spatially adjacent nodes. The resulting node features are then processed by the context backbone. The backbone combines window-based self-attention with parallel depthwise convolutions of different kernel sizes, thereby capturing hierarchical dependencies and multi-scale local structures.

After the shared occupancy stream is decoded, $\widehat{\mathcal{T}}^m$ provides a shared spatial support for temporal-state coding. TemporalNet extracts a structural feature for every reconstructed leaf. Specifically, it embeds the leaf coordinates, ancestor occupancies, octree levels, and octant indices using its GeoExtractor, and processes them with an independently parameterized context backbone:

$$\mathbf{H}^{(0)} = F_{\text{ctx}}\left(F_{\text{geo}}\left(\widehat{\mathcal{T}}^m\right)\right), \tag{6}$$

where F_{geo} and F_{ctx} denote the GeoExtractor and context backbone, and $\mathbf{H}^{(0)} = \{\mathbf{h}_j^{(0)}\}_{j=1}^{N_a}$ is the complete structural representation of the N_a leaves.

3.3 Progressive Grouped State Factorization

The initial leaf-wise representation $\mathbf{H}^{(0)}$ extracted by TemporalNet contains complete geometric context but lacks decoded state cues from neighboring leaves. To exploit such neighborhood dependencies, autoregressive coding provides fine-grained causal context but incurs excessive serial latency. For LiDAR streams, localized temporal changes yield locally coherent state patterns, favoring a progressive schedule that expands neighborhood context across stages. We therefore propose Progressive Grouped State Factorization (PGSF), which progressively densifies the available state context through stage-wise grouping, improving neighborhood modeling while limiting decoding latency. Specifically, the full state sequence is partitioned into non-overlapping 16-position blocks, and each block is decoded in four groups:

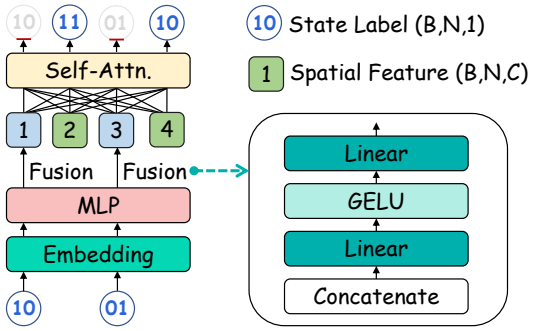


Figure 3: Full-Spatial State-Geometry Coupling with aligned fusion and full-sequence propagation.

$$G_1 = \{8\}, G_2 = \{4, 12, 16\}, G_3 = \{2, 6, 10, 14\},$$
$$G_4 = \{1, 3, 5, 7, 9, 11, 13, 15\}. \tag{7}$$

Accordingly, the conditional distribution is factorized as

$$p_\phi\left(\mathbf{A} \mid \widehat{\mathcal{T}}^m\right) = \prod_{r=1}^{4} \prod_{j \in G_r} p_\phi\left(\mathbf{a}_j \mid \mathbf{c}_j^{(r-1)}, \mathbf{h}_j^{(r-1)}\right). \tag{8}$$

where $\mathbf{A}$ denotes the complete state sequence, G_r is the r-th decoding group, and $\mathbf{c}_j^{(r-1)}$ and $\mathbf{h}_j^{(r-1)}$ denote the context and carrier features available before stage r, respectively.

3.4 Full-Spatial State-Geometry Coupling

PGSF specifies which reconstructed state symbols are available as context at each decoding stage. To exploit these symbols and their neighborhood dependencies, we propose Full-Spatial State-Geometry Coupling (FSGC), which incorporates them into the complete spatial carrier while preserving leaf-wise state-geometry correspondence. GeoContextNet employs subset-based spatial cross-attention to capture hierarchical dependencies from reconstructed occupancy symbols, but applying this strategy to temporal states disrupts their intrinsic correspondence and limits context aggregation because each two-bit state carries limited information. FSGC couples each decoded state with its aligned geometry feature and propagates the resulting state-aware representation over the full leaf sequence, retaining valid geometry carriers and accumulating temporal evidence to enrich subsequent probability estimation.

As shown in Fig. 3, reconstructed state is embedded and fused with the geometry feature at the same leaf position through a lightweight MLP. The fused features replace their original geometry carriers, while the remaining positions retain their geometry-only features. At stage r, the current full carrier is processed to estimate the states in group G_r:

$$\begin{aligned} \mathbf{c}_j^{(r-1)} &= F_{\text{ctx}}\left(\mathbf{h}_j^{(r-1)}\right), \\ p_\phi\left(\mathbf{A}_{G_r} \mid \widehat{\mathcal{T}}^m, \widehat{\mathbf{A}}_{G_{<r}}\right) &= \prod_{j \in G_r} p_\phi\left(\mathbf{a}_j \mid \mathbf{c}_j^{(r-1)}, \mathbf{h}_j^{(r-1)}\right). \end{aligned} \tag{9}$$

where $\widehat{\mathbf{A}}_{G_{<r}}$ denotes the reconstructed states from all preceding groups, with $G_{<r} = \bigcup_{k=1}^{r-1} G_k$. After the current states are entropy-decoded, the full carrier is updated as

$$\mathbf{h}_j^{(r)} = \begin{cases} F_{\text{couple}}\left(\mathbf{h}_j^{(r-1)}, E_{\text{state}}(\widehat{\mathbf{a}}_j)\right), & j \in G_r, \\ \mathbf{h}_j^{(r-1)}, & j \notin G_r. \end{cases} \tag{10}$$

where E_{state} maps the reconstructed state $\widehat{\mathbf{a}}_j$ to a learnable embedding, and F_{couple} denotes a lightweight fusion module. During training, each group is incorporated only after its probabilities have been evaluated, maintaining consistency with causal entropy decoding.

4 Experimental Results

4.1 Experimental Setup

Dataset.

Experiments are conducted on SemanticKITTI (Behley et al., 2019) and Ford (Pandey et al., 2011), two widely used LiDAR datasets with distinct sensing configurations and point-density distributions. For SemanticKITTI, sequences 00–10 are adopted for training, while sequences 11–21 are used for testing. For Ford, the model is trained on Sequence 01 and evaluated on Sequences 02–03. If the last sweep is unpaired, it is encoded independently as a single frame.

Baselines. We evaluate SoDPCC against GPCC TMC13 v27 and learned LPC methods, including Unicorn V2 (Wang et al., 2025a), RENO (You et al., 2025), AnyPCC (Wang et al., 2026), ELiC (Kim et al., 2026), OctFormer (Cui et al., 2023), DuOctree (Cui et al., 2024), TopNet (Wang et al., 2025b), STAEM (Song et al., 2023), DAPCC (Cui et al., 2025), and MDLPCC (Luo et al., 2025). For GPCC, we use octree geometry coding under the MPEG common test conditions (CTC) (WG 07 MPEG 3D Graphics Coding and Haptics Coding, 2024). We additionally evaluate its inter-frame configuration

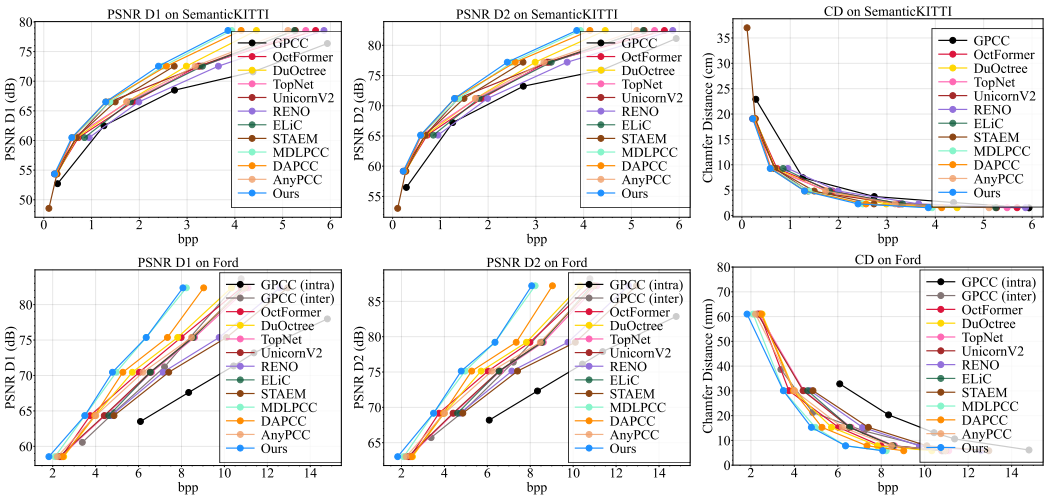


Figure 4: Rate-distortion curves on SemanticKITTI (upper) and Ford (bottom).

Table 1: BD-rate (%) relative to GPCC, measured using D1 PSNR, D2 PSNR, and CD. The symbols ∘ and ⋆ denote single-frame and dynamic compression methods, respectively.

Method	Ref.	SemanticKITTI			Ford		
		D1	D2	CD	D1	D2	CD
OctFormer∘	AAAI'23	-23.78%	-25.05%	-25.97%	-39.75%	-38.75%	-39.32%
STAEM⋆	ICME'23	-30.66%	-32.88%	-27.74%	-23.43%	-21.11%	-24.11%
DuOctree∘	ICRA'24	-28.05%	-29.41%	-28.32%	-40.54%	-39.33%	-39.82%
Unicorn V2∘	TPAMI'25	-32.07%	-33.28%	-31.83%	-33.66%	-32.67%	-32.49%
RENO∘	CVPR'25	-13.85%	-14.32%	-4.76%	-26.08%	-23.83%	-26.61%
TopNet∘	CVPR'25	-28.47%	-29.67%	-33.40%	-32.95%	-31.96%	-31.82%
MDLPCC⋆	JVCI'25	-40.95%	-41.29%	-35.57%	-49.40%	-48.65%	-48.03%
DAPCC⋆	TGRS'25	-39.80%	-40.91%	-37.34%	-43.77%	-42.66%	-42.78%
AnyPCC∘	CVPR'26	-31.17%	-32.61%	-23.45%	-36.67%	-35.49%	-36.50%
ELiC∘	CVPR'26	-21.70%	-22.14%	-13.54%	-33.57%	-31.85%	-31.20%
Ours⋆	-	**-42.13%**	**-43.19%**	**-40.03%**	**-50.18%**	**-49.28%**	**-48.82%**

on Ford. As the CTC provides no inter-frame configuration for SemanticKITTI, GPCC inter-frame results are reported only on Ford. All methods are evaluated on the same test sequences using consistent rate and distortion metrics.

Metrics and Implementation. The coding rate is measured in bits per input point (bpp), with geometry and state bits normalized by the total points of both frames. Reconstruction quality is assessed using point-to-point PSNR (D1), point-to-plane PSNR (D2), and Chamfer Distance (CD). We compute BD-rate (Bjøntegaard, 2001) relative to the GPCC intra-frame anchor, with negative values indicating bit-rate reductions. We implement SoDPCC in PyTorch Lightning and run it on a computer with an Intel Xeon Gold 6234 CPU at 3.30 GHz and an NVIDIA RTX 4090.

4.2 Main Results

Table 1 and Fig. 4 demonstrate that SoDPCC delivers the strongest rate–distortion performance on both benchmarks. Relative to GPCC, SoDPCC achieves BD-rate reductions of 42.13/43.19/40.03% on SemanticKITTI and 50.18/49.28/48.82% on Ford for D1/D2/CD. With dataset-specific training, SoDPCC consistently improves rate-distortion performance on both benchmarks with different LiDAR sensing configurations and point-density distributions. As reported in Table 2, the octree and state streams account for approximately 89.2% and 10.8% of the total bitrate on average, confirming the compactness of the temporal-state representation. The gains mainly stem from the shared representation, which removes duplicated hierarchical geometry before entropy coding and represents

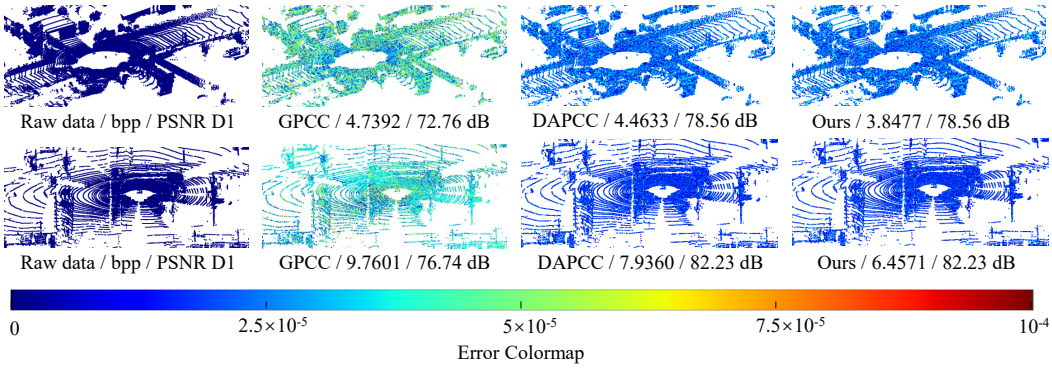


Figure 5: Visualization of SoDPCC and other methods under different bpp on SemanticKITTI (upper) and Ford (bottom) datasets, with blue and red representing low and high quantization errors, respectively.

Table 2: Octree and temporal-state statistics on SemanticKITTI at depths $D = 12/13/14$, with inference times in seconds per input frame.

Metric	Oct.	State
bpp	1.10 / 2.13 / 3.56	0.18 / 0.26 / 0.29
Enc. (s)	0.41 / 0.72 / 1.22	1.96 / 3.10 / 4.11
Dec. (s)	0.45 / 0.74 / 1.21	2.06 / 3.11 / 3.86

Table 3: Encoding and decoding inference times of different methods on SemanticKITTI at depths $D = 12/13/14$, in seconds per input frame.

Method	Enc. (s)	Dec. (s)
STAEM	1.59 / 3.26 / 5.69	1.61 / 3.24 / 5.82
DAPCC	0.52 / 0.64 / 0.99	0.77 / 0.83 / 1.09
Ours	2.37 / 3.82 / 5.33	2.51 / 3.85 / 5.07

the remaining inter-frame variation with a compact temporal-state stream. Table 3 further compares the computational complexity of SoDPCC with representative temporal codecs. We also visualize the per-point geometric deviations, as shown in Fig. 5. From the figure, we can observe that our method remains closer to the original point clouds and achieves lower bitrates.

4.3 Ablation Study

Analysis of Different Modules. Table 4 evaluates the components of SoDPCC at depth 14. XOR and TS denote inter-frame XOR coding and temporally shuffled point-cloud streams. SO denotes shared octree, while PGSF, FSGC, and SSCA denote Progressive Grouped State Factorization, Full-Spatial State-Geometry Coupling, and its subset-spatial baseline, respectively. The shared octree reduces the geometry rate from 4.69 to 3.56 bpp, since shared ancestors are encoded only once instead of being repeated. With FSGC fixed, introducing PGSF reduces the state bitrate from 0.66 to 0.29 bpp by progressively exposing denser decoded state cues. With PGSF fixed, FSGC reduces it from 0.45 to 0.29 bpp through aligned state-geometry fusion and full-sequence context propagation. XOR increases the geometry rate to 7.22 bpp, as sparse and nonuniform LiDAR sampling produces dispersed residual occupancies with limited ancestor sharing. TS lowers the state rate but raises the total rate because reduced inter-frame overlap enlarges the shared octree while making shared states less frequent.

Table 4: Ablation on SemanticKITTI. The second row is single-frame coding, and the gray row is the complete SoDPCC model.

XOR	TS	SO	PGSF	SSCA	FSGC	Oct. bpp↓	State bpp↓
✓	✗	✗	✗	✗	✗	7.22	0.0
✗	✗	✗	✗	✗	✗	4.69	0.0
✗	✗	✓	✗	✗	✓	3.56	0.66
✗	✗	✓	✓	✓	✗	3.56	0.45
✗	✓	✓	✓	✗	✓	3.79	0.21
✗	✗	✓	✓	✗	✓	3.56	0.29

Hierarchical Node Reduction. Fig. 6 compares the average non-leaf node counts of separate-tree, shared-tree, and pairwise-XOR representations at different octree depths. Averaged over all evaluated depths on SemanticKITTI and Ford, the shared octree reduces hierarchical nodes by 29.63%, whereas pairwise XOR increases them by 15.03%. This is because nonuniform sampling in LiDAR

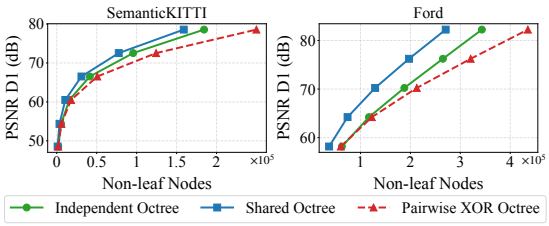


Figure 6: Hierarchical node-count comparison on LPC.

Table 5: Comparison of progressive grouped state factorizations on SemanticKITTI.

Pattern	Scheme	State bpp	Enc. (s)	Dec. (s)
$8/8$	Even-Odd	0.457	1.76	1.83
$1/3/12$	Early-Stop	0.312	3.01	2.97
$1/3/4/8$	Shuffled	0.296	4.22	4.25
2×8	Pairwise	0.307	10.85	10.56
$1/3/4/8$	Progressive	**0.293**	**4.11**	**3.86**

produces fragmented XOR residuals with limited ancestor sharing, while merging directly reuses common parent nodes and hierarchical paths.

Analysis of Progressive Grouped State Factorization. Table 5 compares different grouping schemes under the same FSGC model. Patterns specify the group sizes in decoding order, and state-branch times are averaged per frame. Even-Odd groups states by index parity, Early-Stop defers the remaining positions to the final group, Shuffled changes position assignments with fixed group sizes, Pairwise uses two-state groups, and Progressive adopts our regular refinement schedule. From the table, we can observe that our proposed Progressive scheme achieves the lowest state bitrate of 0.293 bpp with encoding and decoding times of 4.11 s and 3.86 s. The progressive scheme reduces the rate by 0.164 and 0.019 bpp over Even-Odd and Early-Stop, respectively, by exposing increasingly dense decoded states across stages. Pairwise increases decoding time to 10.56 s without improving the rate because frequent FSGC propagation introduces substantial serial computation. The 0.003-bpp gap between Progressive and Shuffled shows that the main gain comes from the four-stage 1–3–4–8 schedule, while regular ordering further improves the rate and runtime.

5 Application

To evaluate the applicability of SoDPCC to downstream perception, we conduct vehicle detection experiments on the KITTI dataset (Geiger et al., 2013) using Part-A^2 Net (Shi et al., 2021). We report Average Precision (AP) at an IoU threshold of 0.7 for the easy, moderate, and hard difficulty levels, and compare SoDPCC with GPCC and DAPCC. As shown in Table 6, SoDPCC achieves a Hard AP of 78.42% at 3.97 bpp, close to the 78.59% obtained from raw point clouds. SoDPCC and DAPCC achieve the same AP because they use the same octree quantization precision. This rate advantage comes from representing shared geometry in a unified octree, reducing redundancy while preserving the vehicle structures needed for detection. These results show that SoDPCC substantially reduces the coding rate while preserving detection performance across difficulty levels.

Table 6: Vehicle detection results. AP is reported at an IoU threshold of 0.7.

Method	bpp ↓	Easy	Mod	Hard
Raw data	–	90.16	80.27	78.59
GPCC	3.53	89.04	79.42	77.78
	5.09	88.86	79.41	78.17
DAPCC	2.65	89.94	80.26	78.27
	4.52	89.97	80.24	78.42
SoDPCC	2.60	89.94	80.26	78.27
	3.97	89.97	80.24	78.42

6 Conclusion

This paper presents SoDPCC, a learned framework for dynamic LiDAR point cloud compression. SoDPCC represents two consecutive frames with a shared octree and compact temporal states, reducing duplicated hierarchical geometry and inter-frame redundancy. GeoContextNet models shared-octree occupancy, while TemporalNet estimates state distributions. Within TemporalNet, PGSF progressively exposes denser decoded state cues to capture neighborhood dependencies with limited serial latency, while FSGC couples these cues with aligned geometry features to preserve state-geometry correspondence and improve probability estimation. Experiments demonstrate that SoDPCC achieves competitive rate-distortion performance on SemanticKITTI and Ford while preserving vehicle-detection accuracy for reconstructed KITTI point clouds.

AI USE STATEMENT

In this work, we used generative AI tools for language refinement and improving manuscript readability. We have not used generative AI tools for other tasks requiring disclosure, and the remaining required disclosure tasks are not applicable to this work. Additionally, we did not use generative AI tools for other tasks with recommended disclosure. We have reviewed all AI-assisted text for accuracy and consistency with the authors' intended meaning. All technical content, analyses, and conclusions were developed and verified by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

REFERENCES

Anique Akhtar, Zhu Li, and Geert Van der Auwera. Inter-frame compression for dynamic point cloud geometry coding. *IEEE Transactions on Image Processing*, 33:584–594, 2024.

J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. Semantickitti: A dataset for semantic scene understanding of lidar sequences. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9297–9307, 2019.

Sourav Biswas, Jerry Liu, Kelvin Wong, Shenlong Wang, and Raquel Urtasun. Muscle: multi sweep compression of lidar using deep entropy models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Gisle Bjøntegaard. Calculation of average psnr differences between rd-curves. In *ITU-T SG 16/Q6, 13th VCEG Meeting*, April 2001.

Mingyue Cui, Junhua Long, Mingjian Feng, Boyang Li, and Kai Huang. Octformer: Efficient octree-based transformer for point cloud compression with local enhancement. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(1):470–478, Jun. 2023.

Mingyue Cui, Mingjian Feng, Junhua Long, Daosong Hu, Shuai Zhao, and Kai Huang. A du-octree based cross-attention model for lidar geometry compression. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3796–3802, 2024.

Mingyue Cui, Yuyang Zhong, Mingjian Feng, Yehua Ling, Junhua Long, Jinhong Xia, and Kai Huang. Dapcc: Diverse attention-based entropy model for dynamic lidar point cloud compression. *IEEE Transactions on Geoscience and Remote Sensing*, 63:1–13, 2025.

Yu Feng, Shaoshan Liu, and Yuhao Zhu. Real-time spatio-temporal lidar point cloud compression. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10766–10773, 2020.

Chunyang Fu, Ge Li, Rui Song, Wei Gao, and Shan Liu. Octattention: Octree-based large-scale contexts model for point cloud compression. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

Emanuele Giacomini, Luca Di Giammarino, Lorenzo De Rebotti, Giorgio Grisetti, and Martin R. Oswald. Splat-loam: Gaussian splatting lidar odometry and mapping. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27630–27639, 2025.

D. Graziosi and A. Tabatabai. An overview of ongoing point cloud compression standardization activities: video-based (v-pcc) and geometry-based (g-pcc). *APSIPA Transactions on Signal and Information Processing*, 9:e13, 2020.

Lila Huang, Shenlong Wang, Kelvin Wong, Jerry Liu, and Raquel Urtasun. Octsqueeze: Octree-structured entropy model for lidar compression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 1313–1323, 2020.

Kazuma Ikeda, Ryosei Hara, Rokuto Nagata, Ozora Sako, Zihao Ding, Takahiro Kado, Ibuki Fujioka, Taro Beppu, Mariko Isogawa, and Kentaro Yoshioka. Ghost-fwl: A large-scale full-waveform lidar dataset for ghost detection and removal. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Junsik Kim, Gun Bang, and Soowoong Kim. Elic: Efficient lidar geometry compression via cross-bit-depth feature propagation and bag-of-encoders. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 39011–39020, June 2026.

R.D. Lin, Pengcheng Weng, Yinqiao Wang, Han Ding, Jinsong Han, and Fei Wang. Hilots: High-low temporal sensitive representation learning for semi-supervised lidar segmentation in autonomous driving. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1429–1438, 2025.

Ao Luo, Linxin Song, Keisuke Nonaka, Jinming Liu, Kyohei Unno, Kohei Matsuzaki, Heming Sun, and Jiro Katto. Mdlpcc: Misalignment-aware dynamic lidar point cloud compression. *J. Vis. Comun. Image Represent.*, 110(C), July 2025. ISSN 1047-3203.

Zirui Pan, Mengbai Xiao, Xu Han, Dongxiao Yu, Guanghui Zhang, and Yao Liu. patchdpcc: A patchwise deep compression framework for dynamic point clouds. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(5):4406–4414, Mar. 2024.

Gaurav Pandey, James R McBride, and Ryan M Eustice. Ford campus vision and lidar data set. *The International Journal of Robotics Research*, 30(13):1543–1552, 2011.

Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948.

Shaoshuai Shi, Zhe Wang, Jianping Shi, Xiaogang Wang, and Hongsheng Li. From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE transactions on pattern analysis and machine intelligence*, 43(8):2647–2664, 2021.

Rui Song, Chunyang Fu, Shan Liu, and Ge Li. Large-scale spatio-temporal attention based entropy model for point cloud compression. In *2023 IEEE International Conference on Multimedia and Expo (ICME)*, pp. 2003–2008, 2023.

Jianqiang Wang, Dandan Ding, Zhu Li, Xiaoxing Feng, Chuntong Cao, and Zhan Ma. Sparse tensor-based multiscale representation for point cloud geometry compression. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(7):9055–9071, 2023.

Jianqiang Wang, Ruixiang Xue, Jiaxin Li, Dandan Ding, Yi Lin, and Zhan Ma. A versatile point cloud compressor using universal multiscale conditional coding – part i: Geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(1):269–287, 2025a.

Kangli Wang, Qianxi Yi, Yuqi Ye, Shihao Li, and Wei Gao. Anypcc: Compressing any point cloud with a single universal model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2972–2982, 2026.

Xinjie Wang, Yifan Zhang, Ting Liu, Xinpu Liu, Ke Xu, Jianwei Wan, Yulan Guo, and Hanyun Wang. Topnet: Transformer-efficient occupancy prediction network for octree-structured point cloud geometry compression. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27305–27314, 2025b.

WG 07 MPEG 3D Graphics Coding and Haptics Coding. Common test conditions for G-PCC. Output document N00944, ISO/IEC JTC1/SC29/WG7, 147th MPEG Meeting, September 2024.

Shuting Xia, Tingyu Fan, Yiling Xu, Jenq-Neng Hwang, and Zhu Li. Learning dynamic point cloud compression via hierarchical inter-frame block matching. In *Proceedings of the 31st ACM International Conference on Multimedia*, MM '23, pp. 7993–8003, New York, NY, USA, 2023. Association for Computing Machinery. ISBN 9798400701085.

Kang You, Tong Chen, Dandan Ding, M. Salman Asif, and Zhan Ma. Reno: Real-time neural compression for 3d lidar point clouds. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22172–22181, 2025.

Chenhao Zhang and Wei Gao. Adadpcc: Adaptive rate control and rate-distortion-complexity optimization for dynamic point cloud compression. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(12):13188–13196, Apr. 2025.