JOINT LOSSLESS POINT CLOUD GEOMETRY AND ATTRIBUTE COMPRESSION VIA CAUSAL MULTISCALE CODING

Anonymous authors
Paper under double-blind review

ABSTRACT

Geometry and attributes in colored point clouds are correlated, yet learned lossless codecs primarily exploit this dependency from geometry to attributes, leaving decoded attribute information unused for geometry probability estimation. A seemingly natural joint variant is to share geometry–attribute features throughout the encoder and decoder; however, this naive feature sharing does not necessarily improve overall lossless compression efficiency. We present a causal multiscale framework for joint lossless geometry and attribute compression. At each scale transition, decoded coarse-scale attributes condition finer-scale occupancy prediction; the finer-scale decoded geometry then provides support and context for attribute probability modeling. A shared joint processing unit (JPU) implements this interleaved cross-modal conditioning, with parameters reused across scales and modality-specific refinements for the two entropy models. On static point clouds from 8iVFB and Owlii, our method reduces geometry and total bitrate on every sequence relative to the separate Unicorn geometry and attribute codecs. On average, geometry, attribute, and total bitrates decrease by 5.0%, 0.7%, and 1.0%, respectively, while using 16.8% fewer trainable parameters. Under the same hardware and test settings, our method achieves $1.74\times$ faster encoding and $1.69\times$ faster decoding than the combined Unicorn codecs. Compared with the conventional codec G-PCC, the proposed method reduces the average total bitrate by 12.12%.

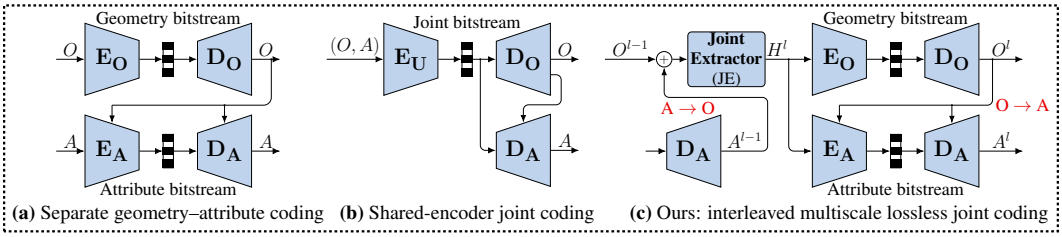
1 INTRODUCTION

Point clouds represent three-dimensional objects and scenes as unordered sets of points, and have become an important data modality for applications such as immersive communication, digital twins, and autonomous systems (Schwarz et al., 2019). A point cloud $\mathcal{P} = (O, A)$ consists of geometry O, which specifies the coordinates of its points, and attributes A, such as color or reflectance. Their large data volume makes efficient point cloud compression (PCC) essential for storage and transmission (Quach et al., 2022).

Learned lossless compression estimates the conditional probability of each symbol from its available context. Given a symbol x_i and context $\mathcal{C}_i$, the entropy model predicts $q_\theta(x_i \mid \mathcal{C}_i)$, corresponding to an ideal coding cost of approximately $-\log_2 q_\theta(x_i \mid \mathcal{C}_i)$ bits. Geometry coding typically predicts binary occupancy symbols, whereas attribute coding requires probability estimation over a much larger value range. More accurate probability estimates therefore lead to lower bitrate. Thus, the key to lossless compression is to construct informative and causally available contexts for accurate entropy modeling.

Geometry and attributes are structurally coupled because geometry defines the spatial support on which attributes are sampled. Existing learned lossless methods commonly encode geometry first, then code attributes conditioned on the decoded geometry, following the factorization $p(O, A) = p(O)p(A \mid O)$. As illustrated in Figure 1(a), Unicorn (Wang et al., 2025b;a) uses separate geometry and attribute codecs, limiting representation sharing and duplicating feature extraction across the two coding pipelines. CNeT (Nguyen & Kaup, 2023) instead performs geometry-first, component-wise conditional modeling: it codes occupancy followed by the three color components, conditioning

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.



(a) Separate geometry–attribute coding **(b)** Shared-encoder joint coding **(c)** Ours: interleaved multiscale lossless joint coding

Figure 1: Comparison of geometry–attribute compression paradigms. (a) Separate geometry and attribute coding. (b) Joint representation learning with a shared encoder and modality-specific decoders. (c) Our interleaved multiscale lossless joint geometry–attribute framework. O, A, $l-1$, l, $\mathbf{E}$, $\mathbf{D}$, H^l and $+$ denote geometry, attributes, the coarser scale, the finer scale, encoder, decoder, shared joint extractor (JE) features and concatenation, respectively.

each color component on previously decoded components. More recent joint codecs (Hsieh et al., 2026; Zhang et al., 2026b; Rudolph et al., 2026; 2024) explicitly model the correlation between geometry and attributes by encoding them into a shared latent representation, followed by separate decoders for occupancy and attributes, as shown in Figure 1(b). These approaches primarily address lossy compression and typically reconstruct geometry before attributes. While their shared latent can carry information from both components, geometry-first order prevents reconstructed attributes from explicitly conditioning the geometry entropy model, leaving the attribute-to-geometry dependency underexploited.

A natural joint approach is to use a shared encoder and decoder for both geometry and attributes. In this way, both components can benefit from a unified representation. However, our ablation shows that such naive feature sharing does not improve overall lossless coding efficiency: gains in geometry coding are offset by degraded attribute modeling, reflecting that the two components benefit from exchanging context while requiring distinct representations for probability estimation. This leads to the central question of this work: *How can a joint lossless codec exploit geometry–attribute dependencies without compromising either entropy model?*

Recently, multiscale sparse representations have shown strong performance for both lossless geometry and attribute coding by efficiently capturing spatial dependencies across and within scales while avoiding computation on empty space (Wang et al., 2023a; 2025a;b). However, despite their similar multiscale coding architectures, geometry and attributes are still modeled by separate networks. This not only leaves their cross-modal dependencies largely unexploited, but also introduces duplicated model parameters and computation.

We observe that the multiscale hierarchy enables causal cross-modal conditioning: reconstructed coarse-scale attributes A^{l-1} inform fine-scale geometry O^l, which, once decoded, provides support and context for current-scale attributes A^l. We exploit this dependency, $A^{l-1} \rightarrow O^l \rightarrow A^l$, in a unified lossless codec. To realize this dependency, we retain Unicorn's multiscale representation and groupwise coding order (Wang et al., 2025b;a) and introduce a joint processing unit (JPU) shared across scales. The JPU uses decoded coarse-scale geometry and attributes to predict fine-scale occupancy, then reuses the shared features on decoded geometry for attribute coding. Modality-specific refinements adapt these features to the two entropy models.

Our main contributions are as follows:

- We formulate a causal multiscale dependency, $A^{l-1} \rightarrow O^l \rightarrow A^l$, that enables decoded coarse-scale attributes to assist fine-scale geometry prediction, while the reconstructed geometry in turn provides context for same-scale attribute coding.
- We develop a unified lossless geometry–attribute codec with a shared joint processing unit (JPU) and modality-specific refinement modules, reducing redundant parameters and computation.
- Compared with separate Unicorn codecs, our method reduces geometry and total bitrate by 5.0% and 1.0% on average, respectively, while using 16.8% fewer parameters and achieving $1.74\times$ faster encoding and $1.69\times$ faster decoding. It also achieves a 12.12% lower average total bitrate than G-PCC.

2 Related Work

Learned lossless geometry compression. Learned lossless geometry codecs estimate occupancy or octree-symbol probabilities from previously decoded geometry. SparseVoxelDNN (Nguyen & Kaup, 2022) uses local autoregressive voxel context, while OctAttention (Fu et al., 2022), EHEM (Song et al., 2023), and TopNet (Wang et al., 2025c) progressively enrich octree context with sibling, ancestor, hierarchical, long-range, and latent features. Multiscale methods such as SparsePCGC (Wang et al., 2023a), Unicorn Part I (Wang et al., 2025b), and UniPCGC (Wang & Gao, 2025) condition fine-scale occupancy on reconstructed coarse geometry and previously decoded same-scale groups, whereas NBP (Liu et al., 2026) predicts geometry bit by bit from decoded geometric context. Despite these advances, existing methods still construct geometry context exclusively from geometry itself.

Learned lossless attribute compression. Lossless attribute codecs typically operate on decoded geometry, which defines the attribute support and provides spatial context for probability estimation. MNeT (Nguyen et al., 2023) constructs multiscale attribute latents and conditions finer-scale attribute prediction on decoded coarser-scale representations. Wang et al. (2023b) further enriches the context with decoded lower-scale attributes, preceding spatial groups, and previously coded color channels. Unicorn Part II (Wang et al., 2025a) instead models cross-scale attribute residuals using spatial lower-scale priors. More recent methods use geometry to organize attribute contexts suited to different point distributions and coding granularities. CEML-PCAC (Xu et al., 2025) constructs coding-free latent contexts across scales, groups, and color channels. ConPCAC (Zhang et al., 2025) partitions points according to their geometric distribution and conditions each group on spatially initialized attributes and previously decoded groups. LOD-PCAC (Zhao et al., 2025) and DALD-PCAC (Fu et al., 2026) derive multilevel reference neighborhoods from geometry and combine their decoded attributes with residual- or channel-level context. Collectively, these methods improve attribute prediction using geometry and decoded attribute context. They do not explicitly use those decoded attributes to condition subsequent geometry entropy models.

Joint geometry–attribute compression. CNeT (Nguyen & Kaup, 2023) codes occupancy and then the three color components in a geometry-first order, with each component conditioned on previously decoded ones and modeled by a separately parameterized network. This order limits parameter sharing and prevents decoded attributes from conditioning geometry coding. More recent joint approaches primarily target lossy compression through shared representation learning. Rudolph et al. (2024) embed geometry and attributes into a unified latent representation, while MEGA-PCC (Hsieh et al., 2026) adopts a shared encoder with modality-specific decoders. DeepJGAC (Zhang et al., 2026b) uses attribute information to enhance geometry features at the encoder, and RR-DPCC (Zhang et al., 2026a) learns a one-stream representation with adaptive geometry–attribute bit allocation. These methods exploit geometry–attribute correlations through shared latent representations, joint bit allocation, or encoder-side feature fusion. However, they are optimized for lossy rate–distortion performance or exploit these correlations only at the encoder. In contrast, our method targets lossless entropy modeling and explicitly makes decoded cross-modal information available to both the encoder and decoder through the causal dependency.

3 Proposed Method

In this section, we first introduce the causal multiscale coding order, then describe the geometry and attribute entropy models, and finally present the joint objective and lossless coding procedure.

3.1 Overview and Causal Multiscale Coding Order

Figure 2 illustrates the construction and reconstruction of our multiscale representation. We represent the input point cloud at the finest scale as a sparse tensor $X^L = (O^L, A^L)$. More generally, $X^l = (O^l, A^l)$ denotes the representation at scale l, where O^l specifies the occupied voxel locations and A^l contains their associated color attributes. Each coarser scale is constructed by applying average pooling (AveP) over non-overlapping $2 \times 2 \times 2$ local cubes. A nonempty cube containing $k \in \{1, \ldots, 8\}$ occupied child voxels is represented by a single occupied parent voxel, while empty cubes are omitted from the sparse representation. The attribute of each parent voxel is obtained by

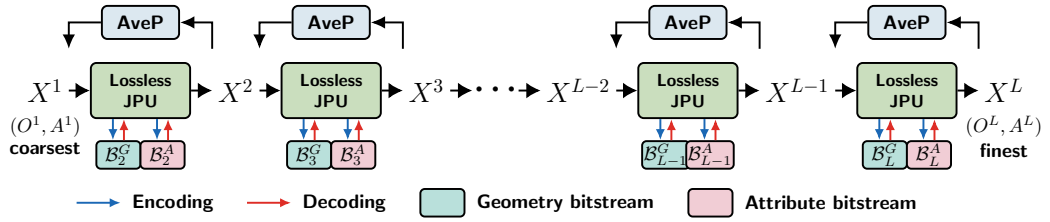


Figure 2: Multiscale hierarchy for lossless joint geometry–attribute coding. Average pooling (AveP) constructs coarser scales; the shared joint processing unit (JPU) reconstructs finer scales. Blue and red arrows denote encoding and decoding.

averaging the attributes of its k occupied children. Repeated application of AveP therefore forms a multiscale hierarchy in which each coarse-scale voxel is associated with the aggregated attributes of its occupied children. This process yields the hierarchy $X^1, \ldots, X^L$, where X^1 denotes the coarsest scale. We continue constructing coarser representations until X^1 contains fewer than 32 points. Given its small size, X^1 is transmitted directly, after which the finer-scale representations are reconstructed losslessly in a coarse-to-fine manner from scale 1 to scale L.

During coarse-to-fine reconstruction, each occupied voxel at scale $l-1$ is subdivided into a $2\times2\times2$ block, yielding eight candidate child voxels at scale l. Following the eight-stage sparse CNN-based occupancy probability approximation (SOPA) scheme of SparsePCGC (Wang et al., 2023a), we predict the binary occupancy of these candidates in a causal group order. For each group, the geometry model estimates occupancy probabilities from the reconstructed $X^{l-1} = (O^{l-1}, A^{l-1})$ and the previously decoded groups. The arithmetic coder uses these probabilities to encode or decode the occupancy symbols. Once all eight candidate positions have been processed, their decoded occupancy states form O^l. At this point, the fine-scale geometry is known, whereas its attributes remain to be reconstructed. Attribute coding operates only on the occupied voxels in O^l. For each such child voxel, we first propagate the attribute of its parent at scale $l-1$ to obtain an initial attribute prediction. Following the conditional coding strategy of Unicorn Part II (Wang et al., 2025a), we code attributes in spatial groups using coarse-scale attributes, decoded geometry, and previously decoded same-scale attributes as context. We represent attributes in YCoCg space and decode luma Y^l before chroma $C^l = (C_o^l, C_g^l)$, allowing reconstructed luma to condition chroma prediction. The decoded attribute information, together with the initial prediction, recovers A^l exactly. The resulting $X^l = (O^l, A^l)$ is then available to both encoder and decoder as context for scale $l+1$.

This decoding order gives the following factorization for each scale transition:

$$q_\theta(X^l \mid X^{l-1}) = q_\theta(O^l \mid X^{l-1})\, q_\theta(Y^l \mid O^l, X^{l-1})\, q_\theta(C^l \mid Y^l, O^l, X^{l-1}). \tag{1}$$

Each factor depends only on information already reconstructed when its symbols are decoded. Thus, coarse-scale attributes can inform current-scale geometry prediction, and the resulting geometry can in turn inform current-scale attribute prediction. The following sections describe how the shared JPU constructs the contexts for these geometry and attribute probability models.

3.2 Attribute-Conditioned Geometry Coding

Figure 3 shows the detailed architecture of the JPU. At the beginning of scale l, the coarse-scale representation $X^{l-1} = (O^{l-1}, A^{l-1})$ has already been reconstructed. We feed this representation into a joint extractor (JE) to combine geometric and attribute information without introducing a separate prediction network. The JE applies sparse convolutions and upsampling to the occupied coarse-scale voxels and their reconstructed attributes. The upsampling expands each occupied parent voxel into its eight candidate child voxels at scale l, and the JE outputs one feature vector for each candidate:

$$H^l = \mathrm{JE}(O^{l-1}, A^{l-1}). \tag{2}$$

Since O^{l-1} and A^{l-1} have already been reconstructed, both the encoder and decoder can compute H^l and use it to predict fine-scale occupancy.

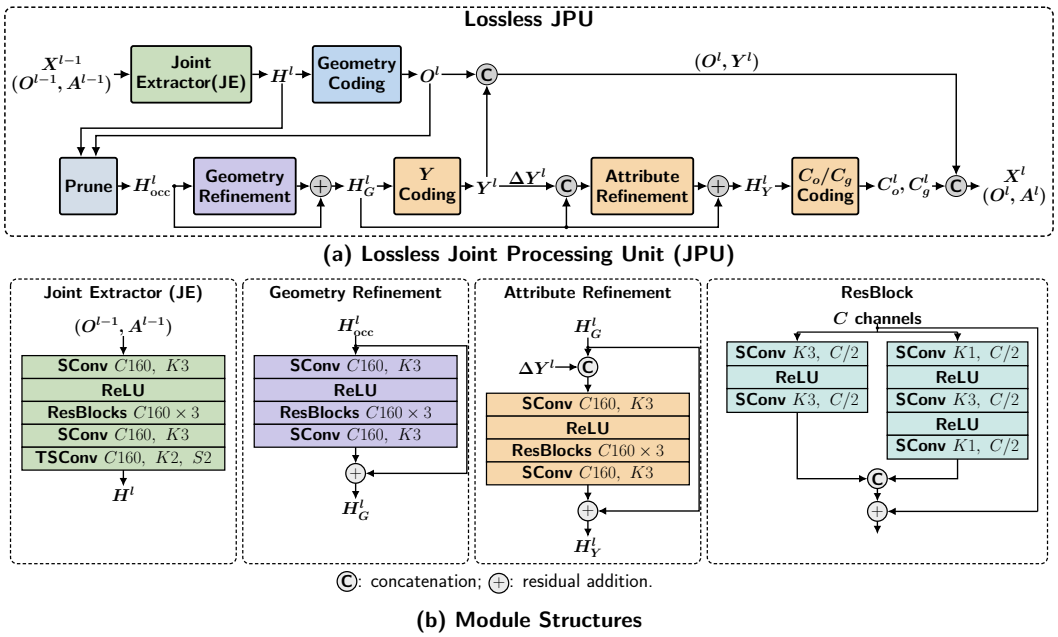


Figure 3: Architecture of the lossless joint processing unit (JPU). (a) Causal coding flow for geometry, luma, and chroma. (b) Structures of the JE, geometry refinement, attribute refinement, and ResBlock modules. The two refinement modules use separate parameters.

We code geometry before attributes at scale l. Specifically, the candidate child locations are partitioned into eight predefined groups $\mathcal{V}_1^l, \ldots, \mathcal{V}_8^l$. The groups are processed sequentially, while the candidates within each group are predicted in parallel. For group g, the geometry model f_{G} combines the candidate features H^l with occupancy states decoded in preceding groups, denoted by $O^l_{<g}$. It predicts a Bernoulli parameter p_v^l for each candidate $v \in \mathcal{V}_g^l$:

$$p_v^l = f_{\mathrm{G}}(H^l, O^l_{<g})_v, \qquad v \in \mathcal{V}_g^l. \tag{3}$$

Thus, current-scale occupancy prediction can use both reconstructed coarser-scale attributes and previously decoded current-scale geometry.

At the encoder, the predicted probabilities are used to arithmetic-code the true occupancy symbols. The decoder computes the same probabilities to recover those symbols from the geometry bitstream $\mathcal{B}_l^G$. After each group is decoded, its occupancy states become available as context for subsequent groups. Once all eight groups have been processed, the fine-scale geometry O^l is reconstructed exactly and defines the spatial support for attribute coding.

3.3 Geometry-Guided Attribute Coding

We represent attributes in the YCoCg color space and code luma Y^l before chroma $C^l = (C_o^l, C_g^l)$. Once the geometry bitstream $\mathcal{B}_l^G$ has been decoded, O^l specifies exactly which candidate child voxels carry attributes. The joint extractor, however, produces H^l at all candidate locations, including those that are unoccupied. Before attribute coding, we prune H^l to the decoded occupied support and refine the retained features:

$$\begin{aligned} H^l_{\mathrm{occ}} &= \mathrm{Prune}(H^l, O^l), \\ H^l_G &= H^l_{\mathrm{occ}} + R_G(H^l_{\mathrm{occ}}). \end{aligned} \tag{4}$$

Here, R_G is a residual geometry-refinement module based on sparse convolutions. Because it operates only at reconstructed occupied locations, its features aggregate information from the actual fine-scale geometric neighborhood. Thus, H^l_G retains the attribute-related information extracted by the JE while incorporating the current-scale occupancy structure and local geometric context needed for attribute prediction.

Once O^l has been decoded, we know which child voxels of each parent are occupied and therefore the number of occupied children. Since the parent attribute was obtained by average pooling over its occupied children during multiscale construction, the exact sum of the children's attribute values can be recovered from the reconstructed parent attribute and the number of occupied children. We denote this sum by S^l. We initialize each undecoded child with the reconstructed attribute value of its parent. As child attributes are decoded, these estimates are replaced by their reconstructed values. Let D^l denote the sum of the child values already decoded, and let r^l denote the number of occupied children that remain undecoded. The estimate for each remaining child is updated as

$$b^l = \frac{S^l - D^l}{r^l}. \tag{5}$$

The attribute model combines this updated context with H_G^l to predict the attribute probabilities of occupied voxels in the current group. After each voxel group is decoded, D^l, r^l, and the remaining estimates are updated accordingly. When only one occupied child remains, its value is uniquely determined by $S^l - D^l$ and therefore requires no entropy-coded symbol. The same applies when a parent has only one occupied child.

Once the luma Y^l is decoded, we subtract each child's parent luma value from its decoded luma value to obtain ΔY^l:

$$\Delta Y^l = Y^l - Y^{l-1}, \tag{6}$$

where Y^{l-1} is the luma value of the parent voxel at scale $l-1$. We then concatenate ΔY^l with the geometry-aware features and apply a residual Attribute Refinement R_A module:

$$H_Y^l = H_G^l + R_A\big([H_G^l, \Delta Y^l]\big). \tag{7}$$

Sparse convolutions in R_A propagate the decoded luma information over the reconstructed geometry. Hence, H_Y^l provides spatially structured cross-channel context rather than only a pointwise luma value. The two chroma components $C^l = (C_o^l, C_g^l)$ are coded together using H_Y^l as the prior, following the same coding procedure as luma Y^l. The two refinement modules have the same residual design but independent parameters. Once luma Y^l and chroma $C^l = (C_o^l, C_g^l)$ have been decoded, $X^l = (O^l, A^l)$ is reconstructed and serves as input to the next scale.

3.4 Joint Objective and Lossless Coding Procedure

We train the model by minimizing the estimated rates of the geometry and attribute symbols. Since reconstruction is lossless, the objective contains no distortion term. At scale l, the geometry model predicts whether each candidate voxel in the eight groups is occupied. Its estimated rate is the sum of the corresponding binary cross-entropies:

$$R_{\text{geo}}^{(l)} = -\sum_{g=1}^{8} \sum_{v \in \mathcal{V}_g^l} \left[o_v^l \log_2 p_v^l + (1 - o_v^l) \log_2 (1 - p_v^l) \right], \tag{8}$$

where $\mathcal{V}_g^l$ is the set of candidate voxels in group g, $o_v^l \in \{0, 1\}$ is the true occupancy, and p_v^l is its predicted probability of being occupied. Assigning a higher probability to the true occupancy state therefore reduces the estimated rate.

For attributes, the model predicts a discrete Laplace distribution for each symbol that must be entropy-coded. Summing the negative log-probabilities of the observed YCoCg symbols gives

$$R_{\text{attr}}^{(l)} = \sum_{c \in \{Y, C_o, C_g\}} \sum_{g=1}^{8} \sum_{i \in \mathcal{I}_{g,c}^l} -\log_2 q_i^{l,g,c}, \tag{9}$$

where $\mathcal{I}_{g,c}^l$ indexes the coded symbols of component c in group g, and $q_i^{l,g,c}$ is the probability mass assigned to the observed symbol. Symbols determined exactly from the parent attribute sum and previously decoded child values, including the last remaining child, are excluded from $\mathcal{I}_{g,c}^l$.

The training loss combines both rates over all modeled scales and normalizes them by the number M_L of points at the finest scale:

$$\mathcal{L} = \frac{1}{M_L} \sum_{l=2}^{L} \left(R_{\text{geo}}^{(l)} + R_{\text{attr}}^{(l)} \right). \tag{10}$$

The sum starts at $l = 2$ because the coarsest representation X^1 is transmitted directly.

Table 1: Lossless geometry, attribute, and total bitrates in bits per point (bpp) on 8iVFB and Owlii. Δ is the percentage change in total bitrate relative to Unicorn, and CR Gain is the percentage change relative to G-PCC; negative values indicate savings. Best results are in bold.

	G-PCC v23			Unicorn			**Causal JPU (Ours)**			
Point Cloud	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Δ (%)↓
8iVFB										
Longdress	0.746	11.554	12.300	0.629	10.057	10.687	**0.605**	**9.891**	**10.495**	**-1.80**
Loot	0.708	6.061	6.769	0.600	**5.384**	5.983	**0.569**	5.390	**5.959**	**-0.40**
Red&Black	0.829	9.204	10.033	0.701	8.119	8.821	**0.672**	**8.089**	**8.761**	**-0.68**
Soldier	0.743	6.931	7.674	0.637	5.853	6.490	**0.592**	**5.849**	**6.441**	**-0.76**
Owlii										
Basketball Player	0.599	8.033	8.633	0.460	7.406	7.866	**0.436**	**7.339**	**7.775**	**-1.16**
Dancer	0.601	8.100	8.701	0.461	7.357	7.817	**0.435**	**7.312**	**7.747**	**-0.90**
Exercise	0.578	8.607	9.186	0.447	8.160	8.607	**0.428**	**8.136**	**8.564**	**-0.50**
Model	0.563	8.288	8.851	0.461	7.307	7.768	**0.441**	**7.219**	**7.660**	**-1.39**
Average	0.671	8.347	9.018	0.550	7.455	8.005	**0.522**	**7.403**	**7.925**	**-1.00**
CR Gain (%)	–	–	–	-18.09	-10.68	-11.24	**-22.15**	**-11.31**	**-12.12**	–

4 Experiments and Results

4.1 Experimental Setup

Datasets and evaluation protocol. For a fair comparison, we follow the training and test split used by Unicorn's attribute codec. Training uses the Real-World Textured Things (RWTT) dataset (Maggiordomo et al., 2020), which contains 568 textured meshes densely sampled as colored point clouds using the same sampling procedure as Unicorn. For solid static evaluation, we use four frames from 8iVFB (d'Eon et al., 2017): *Longdress* (1300), *Loot* (1200), *Red&Black* (1550), and *Soldier* (0690); and four from Owlii (Xu et al., 2017): *Basketball Player* (0200), *Dancer* (0001), *Model* (0001), and *Exercise* (0001). Both datasets are represented at 10-bit precision. We choose these test point clouds because they are used by both the geometry and attribute codecs of Unicorn, enabling a direct and consistent comparison with our joint codec under the same evaluation data. In addition, they are widely used benchmarks for point cloud compression. For the G-PCC baseline, we use the TMC13 v23.0-rc2 reference software (MPEG, 2026) under the MPEG common test conditions (MPEG 3D Graphics Coding Group, 2021).

Implementation details. We implement our model in PyTorch and use MinkowskiEngine (Choy et al., 2019) for sparse convolutions. Training is performed on a single NVIDIA RTX 6000 Ada Generation GPU, and all runtimes are measured on a single NVIDIA RTX 3090 GPU. During training, only 4 successive scale transitions are unrolled, and a single set of JPU parameters is shared across all of them. We train for 26 epochs with a batch size of 4 using Adam with a weight decay of 10^{-4}. The learning rate is 4×10^{-5} for the first 20 epochs, 10^{-5} for the next three epochs, and 5×10^{-6} for the final three epochs. Unicorn uses 32 channels for geometry coding and 128 channels for attribute coding. To maintain a comparable overall feature capacity, we use 160 channels in the JE and in both refinement modules. For occupancy modeling, we adopt the eight-stage conditional prediction architecture of Unicorn Part I (Wang et al., 2025b), increasing its channel width from 32 to 40. For attribute modeling, we adopt the conditional prediction architectures of Unicorn Part II (Wang et al., 2025a), reducing the width from 128 to 40 channels for luma and to 80 channels for chroma. The narrower attribute prediction networks retain task-specific modeling while contributing to the lower overall parameter count. Detailed configurations are provided in Appendix A.

4.2 Lossless Compression Performance

Table 1 compares our joint lossless codec with G-PCC and the separate Unicorn geometry and attribute codecs. We use Unicorn as our primary learned baseline because it adopts a similar multiscale lossless coding framework, while modeling geometry and attributes with separate networks.

All Unicorn bitrate and runtime results are obtained using the officially released code and checkpoints under the same hardware and evaluation settings as our method. Compared with Unicorn, our method achieves a lower geometry bitrate on all eight test point clouds, with an average reduction of 5.0%. This consistent improvement suggests that decoded coarse-scale attributes provide useful additional context for occupancy prediction. The average attribute bitrate also decreases by 0.7%, indicating that joint modeling does not compromise attribute coding efficiency overall. Together, these gains reduce total bitrate on every test sequence, by 1.0% on average. Relative to G-PCC, our method reduces average total bitrate by 12.12%. These results show that geometry–attribute dependencies can be exploited within a causal lossless decoding order while preserving attribute-coding efficiency on average. The following ablations examine the contribution of each design choice to the observed gains. A comparison with CNeT (Nguyen & Kaup, 2023) is given in Appendix C.

4.3 Complexity Analysis

Table 2 compares the runtime and model complexity of our method with those of Unicorn. Since Unicorn's attribute codec (Wang et al., 2025a) also provides a channel-autoregressive configuration that codes luma before chroma, consistent with our $Y \rightarrow (C_o, C_g)$ coding order, we use the corresponding checkpoint for evaluation. We sum the geometry and attribute coding times for Unicorn and average the results over the eight test point clouds. Unicorn requires 22.663 s/frame for encoding and 21.360 s/frame for decoding, compared with 13.004 and 12.638 s/frame for our joint codec. This corresponds to $1.74\times$ faster encoding and $1.69\times$ faster decoding. The joint design also reduces model size. Unicorn's geometry and attribute models contain 1.25M and 52.54M trainable parameters, respectively, for a combined total of 53.79M. Our model contains 44.73M trainable parameters, fewer than Unicorn's attribute model alone and 16.8% fewer than its two models combined. These results show that sharing feature extraction, computation, and parameters across geometry and attribute coding reduces both runtime and model size within a unified joint codec.

Table 2: Runtime and model complexity comparison with Unicorn.

	Unicorn			Causal JPU (Ours)	
Metric	Geo.	Attr.	Total	Joint	Speedup↑
Enc. (s)↓	2.854	19.810	22.663	**13.004**	**1.74**×
Dec. (s)↓	2.430	18.930	21.360	**12.638**	**1.69**×
Params (M)↓	1.25	52.54	53.79	**44.73**	–
Size (MB)↓	4.77	200.42	205.19	**170.63**	–

4.4 Ablation Study

Shared encoder and decoder. We construct a Shared En/Decoder variant to examine whether a shared prediction network is sufficient for efficient joint lossless coding. After the joint extractor (JE), both geometry and attributes use Unicorn's eight-stage SOPA geometry prediction network (Wang et al., 2025b), with task-specific output heads for occupancy probabilities and attribute distribution parameters. The shared network uses 160 channels, matching the sum of the 32 geometry and 128 attribute channels in Unicorn. As shown in Table 3, compared with the separate Unicorn codecs, this variant reduces the average geometry bitrate from 0.550 to 0.460 bpp, but increases the attribute bitrate from 7.455 to 7.902 bpp and the total bitrate from 8.005 to 8.362 bpp. Thus, feature sharing benefits geometry prediction but does not improve overall coding efficiency. Because the variant also changes the geometry context available for attribute coding and the recovery of the final child attribute, the attribute-rate increase cannot be attributed solely to the lack of attribute-specific refinement. The full JPU achieves a lower total bitrate of 7.925 bpp, supporting the combined benefit of structured conditioning and modality-specific processing.

Attribute channel conditioning. Table 4 evaluates two components of the Causal JPU: attribute-channel coding order and modality-specific refinement. For channel coding, we compare three settings: parallel coding of Y, C_o, and C_g, denoted (Y, C_o, C_g); coding Y first, followed by C_o and C_g jointly, denoted $Y \rightarrow (C_o, C_g)$; and fully sequential coding in the order $Y \rightarrow C_o \rightarrow C_g$. Relative to parallel coding, $Y \rightarrow (C_o, C_g)$ reduces average attribute bitrate from 7.660 to 7.403 bpp and total bitrate from 8.207 to 7.925 bpp. The fully sequential order further reduces these rates to 7.324 and 7.848 bpp, respectively, at the cost of an additional sequential channel-coding stage. These results suggest that attribute entropy modeling benefits strongly from richer causal context. Conditioning

Table 3: Ablation on the joint coding architecture. Average bitrates (bpp); CR Gain is relative to G-PCC.

	G-PCC v23			Unicorn			**Shared En/Dec. (Ours)**			**Causal JPU (Ours)**		
Dataset	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓
8iVFB	0.757	8.438	9.194	0.642	7.353	7.995	**0.530**	7.871	8.401	0.609	**7.305**	**7.914**
Owlii	0.585	8.257	8.843	0.457	7.558	8.015	**0.390**	7.933	8.323	0.435	**7.502**	**7.937**
Overall average	0.671	8.347	9.018	0.550	7.455	8.005	**0.460**	7.902	8.362	0.522	**7.403**	**7.925**
CR Gain (%)	–	–	–	-18.09	-10.68	-11.24	**-31.41**	-5.33	-7.28	-22.15	**-11.31**	**-12.12**

Table 4: Ablation of attribute channel coding order and refinement in Causal JPU.

	(Y, C_o, C_g)			$Y \to (C_o, C_g)$			$Y \to C_o \to C_g$			w/o Refinement		
Dataset	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓
8iVFB	0.630	7.560	8.189	**0.609**	7.305	7.914	0.610	**7.265**	**7.874**	0.613	7.498	8.111
Owlii	0.463	7.761	8.224	**0.435**	7.502	7.937	0.440	**7.383**	**7.822**	0.442	7.696	8.138
Overall average	0.546	7.660	8.207	**0.522**	7.403	7.925	0.525	**7.324**	**7.848**	0.528	7.597	8.125

chroma on decoded luma improves probability estimation, and introducing additional cross-channel context further reduces the attribute bitrate.

Refinement module. The effect of the refinement modules is reported in the *w/o Refinement* column of Table 4. Keeping the $Y \to (C_o, C_g)$ coding order fixed, we remove both the geometry refinement module R_G and the attribute refinement module R_A. Without these modules, the average total bitrate increases from 7.925 to 8.125 bpp, including increases of 0.006 bpp in geometry bitrate and 0.194 bpp in attribute bitrate. This indicates that the two refinement modules jointly help adapt the shared representation to the distinct prediction requirements of geometry and attributes.

5 Discussion and Limitations

Our results highlight the importance of when cross-modal information becomes available, rather than feature sharing alone. The proposed factorization does not change the joint entropy of geometry and attributes; instead, its coarse-to-fine order exposes decoded cross-modal context to both sides of the codec, enabling more accurate probability estimation in a finite-capacity model. The naive joint variant achieves a lower geometry bitrate but a higher total bitrate, underscoring the need to evaluate geometry and attributes together.

Our codec requires coarse-scale attributes before finer-scale geometry can be decoded. It therefore does not support independent decoding of the complete geometry bitstream. Moreover, the current method is primarily designed for dense point clouds; its generalization to sparse or dynamic point clouds and other attribute types remains future work.

6 Conclusion

Existing learned lossless codecs use decoded geometry to predict attributes but not the reverse. We showed that a causal multiscale order, $A^{l-1} \to O^l \to A^l$, closes this gap without breaking decodability, letting decoded attributes improve geometry prediction and reconstructed geometry support attribute coding. Our experiments further show that naive encoder–decoder sharing improves geometry coding but degrades attribute coding and total bitrate, whereas causal conditioning with a shared joint processing unit and modality-specific refinements outperforms separate Unicorn codecs and G-PCC with fewer parameters and less runtime. Effective joint lossless compression thus depends on when cross-modal information becomes available, not merely on shared representations. Future work will extend this strategy to other point cloud types and lossy joint compression.

AI USE STATEMENT

In this work, we used generative AI tools to improve the clarity, grammar, and consistency of the manuscript; to search for and summarize relevant literature; and to assist with the layout and formatting of figures and tables. We have not used generative AI tools to generate experimental data or measurements. The proposed method and all experiments were carried out by the authors; other research tasks requiring disclosure were either performed without AI assistance or were not applicable to this work. We have reviewed the AI-assisted text, literature summaries, and figure and table formatting, and take full responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

We describe the multiscale representation and causal coding order in Section 3.1, the geometry and attribute entropy models in Sections 3.2 and 3.3, and the network structures, including channel widths, in Figure 3 and Appendix A. The training objective is given in Section 3.4. Section 4.1 specifies the training and test data, the test frames, and all optimization settings. For the baselines, we use the G-PCC reference software TMC13 v23.0-rc2 under the MPEG common test conditions and the official Unicorn code and pretrained checkpoints, with the configuration and timing protocol described in Section 4.3. We will release the source code and trained models upon publication.

REFERENCES

Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3070–3079, June 2019. doi: 10.1109/CVPR.2019.00319. URL https://ieeexplore.ieee.org/document/8953494.

Eugene d'Eon, Bob Harrison, Taos Myers, and Philip A. Chou. 8i Voxelized Full Bodies – A Voxelized Point Cloud Dataset. Input document m40059/M74006, ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG), Geneva, January 2017.

Chunyang Fu, Ge Li, Rui Song, Wei Gao, and Shan Liu. OctAttention: Octree-Based Large-Scale Contexts Model for Point Cloud Compression. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 36, pp. 625–633, June 2022. doi: 10.1609/aaai.v36i1.19942. URL https://doi.org/10.1609/aaai.v36i1.19942.

Chunyang Fu, Ge Li, Wei Gao, Shiqi Wang, Zhu Li, and Shan Liu. DALD-PCAC: Density-Adaptive Learning Descriptor for Point Cloud Lossless Attribute Compression. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 22(4):104:1–104:23, April 2026. doi: 10.1145/3785465. URL https://dl.acm.org/doi/10.1145/3785465.

Kai-Hsiang Hsieh, Monyneath Yim, Wen-Hsiao Peng, and Jui-Chiu Chiang. MEGA-PCC: A Mamba-based Efficient Approach for Joint Geometry and Attribute Point Cloud Compression. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 1820–1830, March 2026. doi: 10.1109/WACV61042.2026.00182. URL https://openaccess.thecvf.com/content/WACV2026/html/Hsieh_MEGA-PCC_A_Mamba-based_Efficient_Approach_for_Joint_Geometry_and_Attribute_WACV_2026_paper.html.

Bojun Liu, Yangzhi Ma, Li Li, Dong Liu, Zhu Li, and Houqiang Li. Next Bit Prediction: A Unified Lossless and Lossy Point Cloud Geometry Compression Framework. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(5):5371–5387, May 2026. doi: 10.1109/TPAMI.2025.3650590. URL https://ieeexplore.ieee.org/document/11328862.

Andrea Maggiordomo, Federico Ponchio, Paolo Cignoni, and Marco Tarini. Real-World Textured Things: A Repository of Textured Models Generated with Modern Photo-Reconstruction Tools. *Computer Aided Geometric Design*, 83:101943, November 2020. doi: 10.1016/j.cagd.2020.101943. URL https://doi.org/10.1016/j.cagd.2020.101943.

MPEG. G-PCC Reference Software TMC13, Version 23.0-rc2. GitHub repository, 2026. URL https://github.com/MPEGGroup/mpeg-pcc-tmc13/tree/release-v23.0-rc2. Accessed September 2026.

MPEG 3D Graphics Coding Group. Common Test Conditions for G-PCC. Output document N00106, ISO/IEC JTC1/SC29/WG7, April 2021.

Dat Thanh Nguyen and André Kaup. Learning-Based Lossless Point Cloud Geometry Coding Using Sparse Tensors. In *IEEE International Conference on Image Processing (ICIP)*, pp. 2341–2345, October 2022. doi: 10.1109/ICIP46576.2022.9897827. URL https://ieeexplore.ieee.org/document/9897827.

Dat Thanh Nguyen and André Kaup. Lossless Point Cloud Geometry and Attribute Compression Using a Learned Conditional Probability Model. *IEEE Transactions on Circuits and Systems for Video Technology*, 33(8):4337–4348, August 2023. doi: 10.1109/TCSVT.2023.3239321. URL https://ieeexplore.ieee.org/document/10024999.

Dat Thanh Nguyen, Kamal Gopikrishnan Nambiar, and André Kaup. Deep Probabilistic Model for Lossless Scalable Point Cloud Attribute Compression. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1–5, June 2023. doi: 10.1109/ICASSP49357.2023.10095385. URL https://ieeexplore.ieee.org/document/10095385.

Maurice Quach, Jiahao Pang, Dong Tian, Giuseppe Valenzise, and Frédéric Dufaux. Survey on Deep Learning-Based Point Cloud Compression. *Frontiers in Signal Processing*, 2:846972, February 2022. doi: 10.3389/frsip.2022.846972. URL https://doi.org/10.3389/frsip.2022.846972.

Michael Rudolph, Aron Riemenschneider, and Amr Rizk. Learned Compression of Point Cloud Geometry and Attributes in a Single Model through Multimodal Rate-Control. arXiv preprint arXiv:2408.00599, August 2024. URL https://arxiv.org/abs/2408.00599.

Michael Rudolph, Aron Riemenschneider, and Amr Rizk. Unified Compression of Point Cloud Geometry and Attributes through Variable-Rate Conditioning. In *ACM Multimedia Systems Conference (MMSys)*, pp. 1–12, April 2026. doi: 10.1145/3793853.3795742. URL https://dl.acm.org/doi/10.1145/3793853.3795742.

Sebastian Schwarz, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip A. Chou, Robert A. Cohen, Maja Krivokuća, Sébastien Lasserre, Zhu Li, Joan Llach, Khaled Mammou, Rufael Mekuria, Ohji Nakagami, Ernestasia Siahaan, Ali Tabatabai, Alexis M. Tourapis, and Vladyslav Zakharchenko. Emerging MPEG Standards for Point Cloud Compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(1):133–148, March 2019. doi: 10.1109/JETCAS.2018.2885981. URL https://ieeexplore.ieee.org/document/8571288.

Rui Song, Chunyang Fu, Shan Liu, and Ge Li. Efficient Hierarchical Entropy Model for Learned Point Cloud Compression. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14368–14377, June 2023. doi: 10.1109/CVPR52729.2023.01381. URL https://ieeexplore.ieee.org/document/10205051.

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, July 2023a. doi: 10.1109/TPAMI.2022.3225816. URL https://ieeexplore.ieee.org/document/9968173.

Jianqiang Wang, Dandan Ding, and Zhan Ma. Lossless Point Cloud Attribute Compression Using Cross-Scale, Cross-Group, and Cross-Color Prediction. In *Data Compression Conference (DCC)*, pp. 228–237, March 2023b. doi: 10.1109/DCC55655.2023.00031. URL https://ieeexplore.ieee.org/document/10125514.

Jianqiang Wang, Ruixiang Xue, Jiaxin Li, Dandan Ding, Yi Lin, and Zhan Ma. A Versatile Point Cloud Compressor Using Universal Multiscale Conditional Coding – Part II: Attribute. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(1):252–268, January 2025a. doi: 10.1109/TPAMI.2024.3462945. URL https://ieeexplore.ieee.org/document/10682566.

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, January 2025b. doi: 10.1109/TPAMI.2024.3462938. URL https://ieeexplore.ieee.org/document/10682571.

Kangli Wang and Wei Gao. UniPCGC: Towards Practical Point Cloud Geometry Compression via an Efficient Unified Approach. In *AAAI Conference on Artificial Intelligence (AAAI)*, volume 39, pp. 12721–12729, April 2025. doi: 10.1609/aaai.v39i12.33387. URL https://doi.org/10.1609/aaai.v39i12.33387.

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 *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 27305–27314, June 2025c. doi: 10.1109/CVPR52734.2025.02543. URL https://ieeexplore.ieee.org/document/11093721.

Qiang Xu, Lixuan Meng, Guangjie Zhang, Wei Gao, and Ge Li. Coding-Free Multiscale Latent Variables for Lossless Point Cloud Attribute Compression. In *IEEE International Conference on Multimedia and Expo (ICME)*, pp. 1–6, June 2025. doi: 10.1109/ICME59968.2025.11209985. URL https://ieeexplore.ieee.org/document/11209985.

Yi Xu, Yao Lu, and Ziyu Wen. Owlii Dynamic Human Mesh Sequence Dataset. Input document m41658, ISO/IEC JTC1/SC29/WG11 (MPEG), Macau, October 2017.

Junzhe Zhang, Tong Chen, Kang You, Dandan Ding, and Zhan Ma. ConPCAC: Conditional Lossless Point Cloud Attribute Compression via Spatial Decomposition. *IEEE Transactions on Circuits and Systems for Video Technology*, 35(7):7210–7221, July 2025. doi: 10.1109/TCSVT.2025.3540931. URL https://ieeexplore.ieee.org/document/10879568.

Yun Zhang, Lewen Fan, Zixi Guo, Xu Wang, Xiaoxia Huang, and Sam Kwong. Rate-Reconfigurable Deep Point Cloud Compression With Perceptual Bit Allocation Optimization. *IEEE Transactions on Image Processing*, 35:3451–3465, 2026a. doi: 10.1109/TIP.2026.3676604. URL https://ieeexplore.ieee.org/document/11457367.

Yun Zhang, Zhiwei Guo, Zixi Guo, Linwei Zhu, and C.-C. Jay Kuo. Deep-JGAC: End-to-End Deep Joint Geometry and Attribute Compression for Dense Colored Point Clouds. *IEEE Transactions on Circuits and Systems for Video Technology*, 36(8):11416–11430, August 2026b. doi: 10.1109/TCSVT.2026.3676115. URL https://ieeexplore.ieee.org/document/11449279.

Wenbo Zhao, Wei Gao, Dingquan Li, Jing Wang, and Guoqing Liu. LOD-PCAC: Level-of-Detail-Based Deep Lossless Point Cloud Attribute Compression. *IEEE Transactions on Image Processing*, 34:3918–3929, 2025. doi: 10.1109/TIP.2025.3578760. URL https://ieeexplore.ieee.org/document/11039156.

A Network Configuration

Table 5 lists the number of channels in each module of the JPU and in the corresponding modules of the baselines. Unicorn codes geometry and attributes with separate networks of 32 and 128 channels. We keep this total width of 160 channels in the shared JE and the two refinement modules, and split it among the occupancy, luma, and chroma prediction networks as $40+40+80 = 160$. Compared with Unicorn, the luma and chroma prediction networks therefore use fewer channels than its 128-channel attribute coder, and the complete model contains 44.73M instead of 53.79M trainable parameters.

Table 5: Channel configuration of the JPU and the corresponding modules of the baselines.

Module	Ours	Baseline
Joint extractor (JE)	160	–
Geometry refinement R_G	160	–
Attribute refinement R_A	160	–
Occupancy prediction (eight-stage SOPA)	40	32 (SparsePCGC, Unicorn)
Luma (Y) prediction	40	128 (Unicorn)
Chroma (C_o, C_g) prediction	80	128 (Unicorn)
Total prediction width	$40 + 40 + 80 = 160$	$32 + 128 = 160$
Trainable parameters (M)	44.73	53.79 (Unicorn)

B Per-Sequence Ablation Results

Tables 6 and 7 report the per-sequence bitrates of the ablation variants in Section 4.4; their averages match Tables 3 and 4.

Table 6 compares different attribute channel coding orders with both refinement modules enabled. Relative to parallel coding of Y, C_o, and C_g, our default $Y \rightarrow (C_o, C_g)$ order reduces both attribute and total bitrate on all eight test point clouds. The fully sequential order $Y \rightarrow C_o \rightarrow C_g$ further reduces both rates on every sequence. These results indicate that decoded luma provides useful cross-channel context for chroma probability modeling, while finer channel dependencies can further improve attribute compression at the cost of an additional sequential coding stage.

Table 7 further examines the effect of different architectural choices. The Shared En/Decoder variant achieves the lowest geometry bitrate on every point cloud, yet its total bitrate is higher than that of the full model, the *w/o Refinement* variant, and all channel-order variants in Table 6 on every point cloud, and higher on average than all variants. This suggests that fully shared feature representations can benefit geometry prediction, but do not by themselves ensure efficient joint coding, as attribute modeling still requires features adapted to its own statistics.

The *w/o Refinement* variant keeps the $Y \rightarrow (C_o, C_g)$ coding order unchanged while removing both the geometry refinement module R_G and the attribute refinement module R_A. Its average total bitrate increases from 7.925 to 8.125 bpp, with degradation on all eight test point clouds. This shows that, after feature sharing, modality-specific refinement remains important for adapting the shared representation to the distinct prediction requirements of geometry and attributes.

The final variant removes both the refinement modules and the luma-to-chroma conditioning by coding Y, C_o, and C_g in parallel. Its average total bitrate further increases to 8.336 bpp, representing the largest degradation among these ablations. In comparison, removing only luma-to-chroma conditioning gives 8.207 bpp, while removing only the refinement modules gives 8.125 bpp. These results suggest that cross-channel conditioning and modality-specific refinement provide complementary benefits: the former improves probability modeling across attribute channels, whereas the latter helps adapt the shared features to the different geometry and attribute prediction tasks.

Overall, the ablation results show that efficient joint lossless coding depends not simply on sharing more features, but on both when cross-modal and cross-channel information becomes available and how the shared representation is adapted to the different prediction tasks.

Table 6: Per-sequence bitrates (bpp) for different attribute channel coding orders, with both refinement modules enabled. The middle group is our default configuration.

	(Y, C_o, C_g)			$Y \to (C_o, C_g)$			$Y \to C_o \to C_g$		
Point Cloud	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓
8iVFB									
Longdress	0.627	10.350	10.977	**0.605**	9.891	10.495	0.606	**9.831**	**10.437**
Loot	0.592	5.494	6.086	**0.569**	5.390	5.959	0.572	**5.380**	**5.952**
Red&Black	0.682	8.365	9.048	0.672	8.089	8.761	**0.668**	**8.010**	**8.678**
Soldier	0.618	6.030	6.648	**0.592**	5.849	6.441	0.593	**5.838**	**6.431**
Average	0.630	7.560	8.189	**0.609**	7.305	7.914	0.610	**7.265**	**7.874**
Owlii									
Basketball Player	0.468	7.595	8.063	**0.436**	7.339	7.775	0.440	**7.187**	**7.627**
Dancer	0.464	7.543	8.007	**0.435**	7.312	7.747	0.440	**7.194**	**7.634**
Exercise	0.453	8.338	8.791	**0.428**	8.136	8.564	0.432	**8.015**	**8.447**
Model	0.465	7.568	8.034	**0.441**	7.219	7.660	0.446	**7.135**	**7.581**
Average	0.463	7.761	8.224	**0.435**	7.502	7.937	0.440	**7.383**	**7.822**
Overall average	0.546	7.660	8.207	**0.522**	7.403	7.925	0.525	**7.324**	**7.848**

Table 7: Per-sequence bitrates (bpp) for the architecture ablations. *w/o Refinement* removes R_G and R_A under the $Y \to (C_o, C_g)$ order; the last group additionally codes Y, C_o, and C_g in parallel.

	Shared En/Dec.			w/o Refinement			w/o Ref., (Y, C_o, C_g)		
Point Cloud	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓
8iVFB									
Longdress	**0.504**	10.913	11.417	0.609	**10.276**	**10.886**	0.605	10.620	11.226
Loot	**0.518**	5.593	6.111	0.574	**5.469**	**6.043**	0.567	5.650	6.217
Red&Black	**0.581**	8.746	9.327	0.673	**8.265**	**8.938**	0.661	8.540	9.200
Soldier	**0.516**	6.234	6.751	0.597	**5.980**	**6.577**	0.588	6.216	6.804
Average	**0.530**	7.871	8.401	0.613	**7.498**	**8.111**	0.605	7.756	8.362
Owlii									
Basketball Player	**0.390**	7.748	8.137	0.444	**7.525**	**7.968**	0.433	7.691	8.123
Dancer	**0.388**	7.731	8.118	0.443	**7.479**	**7.922**	0.433	7.675	8.108
Exercise	**0.383**	8.546	8.928	0.434	**8.291**	**8.725**	0.424	8.453	8.877
Model	**0.402**	7.707	8.108	0.448	**7.490**	**7.938**	0.442	7.688	8.130
Average	**0.390**	7.933	8.323	0.442	**7.696**	**8.138**	0.433	7.877	8.310
Overall average	**0.460**	7.902	8.362	0.528	**7.597**	**8.125**	0.519	7.817	8.336

C Comparison with CNeT

CNeT (Nguyen & Kaup, 2023) is a prior learned codec for joint lossless geometry and attribute compression using geometry-first, component-wise conditional modeling. It codes occupancy first, followed by the three color components, conditioning each color component on previously decoded components. It trains four separately parameterized probability networks, one for occupancy and one for each color component. The public CNeT repository provides only the attribute networks and their encoder, without the geometry network, the decoder, or trained checkpoints, so we cannot reproduce CNeT under our test setting. Following the evaluation protocol of Unicorn (Wang et al., 2025a), we therefore compare with the results reported for CNeT on the two 8iVFB sequences, *Red&Black* and *Loot*, and evaluate our method on the same sequences.

Table 8 reports the bitrates. Our method achieves an average of 7.360 bpp and a 12.39% rate reduction relative to G-PCC, compared with 7.402 bpp and 11.89% for Unicorn; our geometry bitrate of 0.620 bpp is also lower than that of Unicorn (0.651 bpp). CNeT achieves the lowest geometry, at-

tribute, and total bitrates (0.36, 5.93, and 6.29 bpp) by coding occupancy and each color component voxel by voxel.

Table 9 compares the coding times on the same two sequences. Unicorn and our codec are measured on the same hardware; on average, our codec requires 12.870 s/frame for encoding and 12.552 s/frame for decoding, compared with 22.423 and 21.231 s/frame for Unicorn. CNeT reports only average times over its eleven test point clouds, measured on an NVIDIA RTX 3090 GPU (Nguyen & Kaup, 2023): its voxel-wise autoregressive design requires 25 s/frame for encoding and 590 s/frame for decoding. Unicorn (Wang et al., 2025a) further reports a CNeT model size of 240 MB. Although our runtimes are also measured on an RTX 3090 GPU, the CNeT figures were obtained on a different system and averaged over different test point clouds, so they are not directly comparable with our measurements; they nevertheless indicate that our method trades a higher bitrate than CNeT for decoding that is more than an order of magnitude faster.

Table 8: Lossless geometry, attribute, and total bitrates (bpp) on the two 8iVFB sequences used to evaluate CNeT. CNeT results are those reported in its original paper (Nguyen & Kaup, 2023). CR Gain is the percentage change relative to our G-PCC results.

	G-PCC v23			Unicorn			CNeT			Causal JPU (Ours)		
Point Cloud	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓	Geo.↓	Attr.↓	Total↓
Red&Black	0.829	9.204	10.033	0.701	8.119	8.821	**0.39**	**7.21**	**7.60**	0.672	8.089	8.761
Loot	0.708	6.061	6.769	0.600	5.384	5.983	**0.33**	**4.65**	**4.98**	0.569	5.390	5.959
Average	0.769	7.633	8.401	0.651	6.752	7.402	**0.36**	**5.93**	**6.29**	0.620	6.740	7.360
CR Gain (%)	–	–	–	-15.35	-11.54	-11.89	**-53.16**	**-22.31**	**-25.13**	-19.27	-11.70	-12.39

Table 9: Average encoding and decoding times per frame and the speedup of our codec over each method. Unicorn and our codec are measured on the same hardware and averaged over *Red&Black* and *Loot*; Unicorn times include both its geometry and attribute codecs. †CNeT reports only average times over its eleven test point clouds (Nguyen & Kaup, 2023), measured with the same GPU model (RTX 3090) on a different system, so its speedups are indicative only.

	Time (s)		Speedup (×)	
Method	Enc.	Dec.	Enc.	Dec.
Unicorn	22.423	21.231	1.74	1.69
CNeT†	25	590	1.94	47.00
Causal JPU (Ours)	**12.870**	**12.552**	–	–