This CVPR paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

Generalized Gaussian Entropy Model for Point Cloud Attribute Compression with Dynamic Likelihood Intervals

Changhao Peng
Peking University
changhaopeng@foxmail.com

Abstract

Gaussian and Laplacian entropy models are proved effective in learned point cloud attribute compression, as they assist in arithmetic coding of latents. However, we demonstrate through experiments that there is still unutilized information in entropy parameters estimated by neural networks in current methods, which can be used for more accurate probability estimation. Thus we introduce generalized Gaussian entropy model, which controls the tail shape through shape parameter to more accurately estimate the probability of latents. Meanwhile, to the best of our knowledge, existing methods use fixed likelihood intervals for each integer during arithmetic coding, which limits model performance. We propose Mean Error Discriminator (MED) to determine whether the entropy parameter estimation is accurate and then dynamically adjust likelihood intervals. Experiments show that our method significantly improves rate-distortion (RD) performance on three VAE-based models for point cloud attribute compression, and our method can be applied to other compression tasks, such as image and video compression.

1. Introduction

In recent years, point clouds have become an important type of 3D visual data, with widespread applications in virtual reality [29], autonomous driving [30], and various other fields. With the massive generation of point cloud, there is an urgent need for efficient compression methods. Point clouds consist of two main aspects: geometry information and attribute information. Geometry information refers to the 3D coordinates of the points, while attribute information includes various details such as color and reflectance. Current geometry compression methods, such as the traditional coding algorithms used in G-PCC [14] and V-PCC [15], as well as deep learning-based methods like SparsePCGC [26], OctAttention [7] and EHEM [25], have achieved efficient compression of point cloud geometry. However, for attribute compression, the achievable compression ratios with current methods are still far below those for geometry compression. Therefore, our focus is on point cloud attribute compression based on lossless geometry.

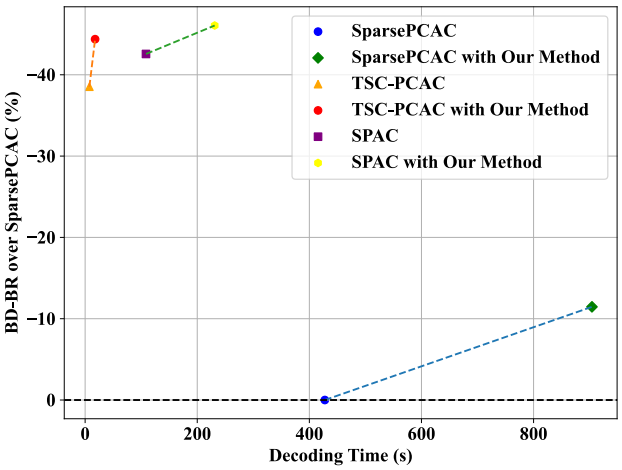

Figure 1. Rate-speed comparison on 8iVFBv2 [1] and MVUB [18]. Left-top is better.

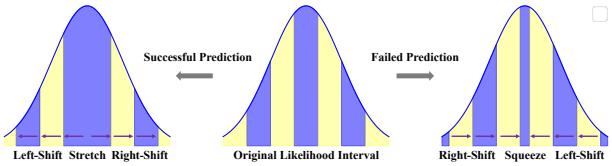

Figure 2. Dynamic likelihood intervals. When the prediction of mean is inaccurate, squeeze the likelihood interval of $\lfloor \mu \rceil$ so that the remaining integer likelihood intervals can shift towards the center of the entropy model, thereby increasing the likelihood. On the contrary, when the prediction is accurate, the likelihood interval of $\lfloor \mu \rceil$ is stretched.

Variational autoencoders (VAEs) [13] have played an important role in deep learning-based compression frameworks. Based on VAE, Ballé et al. [2] propose using hyperpriors to generate multivariate Gaussian distributions

for arithmetic coding of latents. Later, in point cloud attribute compression, Gaussian or Laplacian distributions determined by the mean and scale, have been used as entropy models for latents [9, 19, 20]. In fact, we demonstrate through experiments that these entropy model parameters can be utilized to further improve the performance of the model.

On the other hand, we need to transform continuous entropy models into discrete probability tables for arithmetic coding [22], where each integer corresponds to a probability. Previous work [2] directly uses the half neighborhood of integers as likelihood intervals to calculate discrete probability tables. Our experiment shows that this fixed likelihood interval limits the performance of the model.

Based on the above discussion, we propose:

- Using generalized Gaussian distribution which is more expressive as the entropy model to provide more accurate probability estimation of the latents.
- Determining the accuracy of the current entropy model based on mean, scale and decoded latents, then dynamically adjusting the likelihood intervals for more efficient arithmetic coding.
- Developing a two-step training strategy, as our method does not make significant changes to the original structure of the model except adding a few modules, allowing it to be well applied to current VAE-based models.

2. Related Work

2.1. Learned Image Compression

Learned image compression is a highly developed field, and its methods are of significant reference value for the compression of point cloud attributes. Ballé et al. [2] are the first to propose an image compression method based on hyperpriors, which has become the foundational framework for subsequent image compression techniques. Building on the hyperprior model, Minnen et al. [20] introduce a context module that allows for more accurate estimation of entropy parameters. However, the serial computation required by the context module substantially increases encoding and decoding times. He et al. [9] propose a checkerboard model that utilizes context from anchor regions to assist in the encoding of non-anchor regions, reducing the encoding and decoding time with almost no performance loss. Cheng et al. [5] suggest replacing Gaussian distributions with mixture Gaussian distributions as the entropy model. Minnen et al. [19] propose leveraging context between channels to assist in the encoding process. Recent work [10–12] has focused on maximizing the use of contextual information while maintaining the parallelism of the context modules.

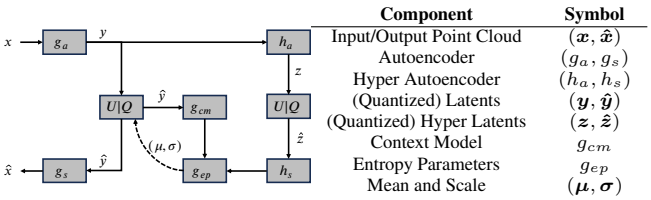

Figure 3. Operational diagrams: Variational compression model with hyperprior and ontext.

2.2. Point Cloud Attribute Compression

In traditional point cloud attribute compression methods, the main research focus is on how to optimize prediction and transformation. Song et al. [24] propose using different prediction methods for point clouds with different features, Shao et al. [23] propose using graph transform instead of discrete cosine transform (DCT), and Peng et al. [21] propose learning an optimal Laplacian matrix through convex optimization to achieve better graph transform.

In recent years, the performance of deep learning-based methods has gradually surpassed traditional methods. Wang et al. [26] propose the first learned point cloud attribute compression framework based on context and hyperpriors. Recent work [27] aims at how to more efficiently utilize context. Guo et al. [8] propose using context between channels to increase the parallelism of context modules and combining sparse convolution and transformer to better capture local and global information. Mao et al. [17] propose layering point clouds to more fully utilize context achieves SOTA RD performance. Zhang et al. [6, 31] also propose using neural networks to enhance traditional methods, which compensates for the shortcomings of traditional methods' insufficient predictive ability.

3. Preliminary

3.1. Variational Compression Model with Hyperprior and Context

The framework of the variational compression model with hyperprior and context is shown in Figure 3. $\boldsymbol{y}$ is the latent representation after encoding, while $\boldsymbol{z}$ is the hyper latent for estimating the entropy model of $\boldsymbol{y}$. During training, quantization is replaced by adding uniform noise from $\mathcal{U}(-\frac{1}{2}, \frac{1}{2})$ to $\boldsymbol{y}$ or $\boldsymbol{z}$. The entropy model for the decoding part of $\hat{\boldsymbol{y}}$ is jointly estimated by $\hat{\boldsymbol{z}}$ and the context obtained from the decoded part, while $\hat{\boldsymbol{z}}$ is compressed by a factorized entropy model. The model uses rate-distortion cost as the loss function for end-to-end training:

$$R + \lambda \cdot D = E_{\boldsymbol{x}\sim p_{\boldsymbol{x}}}\left[-\log_2 p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) - \log_2 p_{\hat{\boldsymbol{z}}}(\hat{\boldsymbol{z}})\right] + \lambda \cdot E_{\boldsymbol{x}\sim p_{\boldsymbol{x}}}[d(\boldsymbol{x}, \hat{\boldsymbol{x}})] \tag{1}$$

where entropy is used to estimate the bitrate of $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{z}}$, and $d(\boldsymbol{x}, \hat{\boldsymbol{x}})$ is the distortion of the reconstructed point cloud, usually represented by MSE or MS-SSIM [28].

In fact, $\hat{\boldsymbol{y}}$ occupies the vast majority of the total bitrate (usually more than 95%), of which the probability is modeled by Gaussian likelihoods:

$$p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) = \prod_i \left[\mathcal{N}(\mu_i, \sigma_i^2) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)\right](\hat{y}_i). \quad (2)$$

The Gaussian distribution in Equation 2 could be replaced with mixture Gaussian distribution or Laplacian distribution.

3.2. Generalized Gaussian Distribution

If the random variable $X \sim \mathcal{G}(\mu, \sigma, \beta)$,then

$$P(X = x) = \frac{\beta}{2\sigma\Gamma\left(\frac{1}{\beta}\right)} e^{-\frac{|x-\mu|^\beta}{\sigma^\beta}} \quad (3)$$

where

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt. \quad (4)$$

Compared to the Gaussian distribution, generalized Gaussian distribution [3] has an additional shape parameter β to control whether the distribution has a heavy tail or a light tail, as shown in Figure 4a.

It is worth noting that the generalized Gaussian distribution degenerates into a Laplacian distribution when $\beta = 1$, or a Gaussian distribution when $\beta = 2$.

4. Explore a More Accurate Likelihood

4.1. The Relationship Between Latent and Entropy Parameters

Obviously, the closer $\boldsymbol{\mu}$ is to $\hat{\boldsymbol{y}}$ in Equation 2, the greater the likelihood and the lower the bitrate, which can be written as:

$$\arg\max_{\boldsymbol{\mu}} p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) = \hat{\boldsymbol{y}}, \quad (5)$$

thus we can consider $\boldsymbol{\mu}$ as the predicted value of $\hat{\boldsymbol{y}}$. Next, we analyze the optimal value of the scale parameter $\boldsymbol{\sigma}$ by fixing $\boldsymbol{\mu}$ and $\hat{\boldsymbol{y}}$:

$$\arg\max_{\boldsymbol{\sigma}} p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}). \quad (6)$$

We can conclude that the i-th element of $\boldsymbol{\sigma}$ satisfies:

$$\sigma_i = \arg\max_{\sigma} \left[\mathcal{N}(\mu_i, \sigma^2) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)\right](\hat{y}_i). \quad (7)$$

Since

$$\mathcal{N}(\mu_i, \sigma^2)(\hat{y}_i) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\hat{y}_i-\mu_i)^2}{2\sigma^2}}, \quad (8)$$

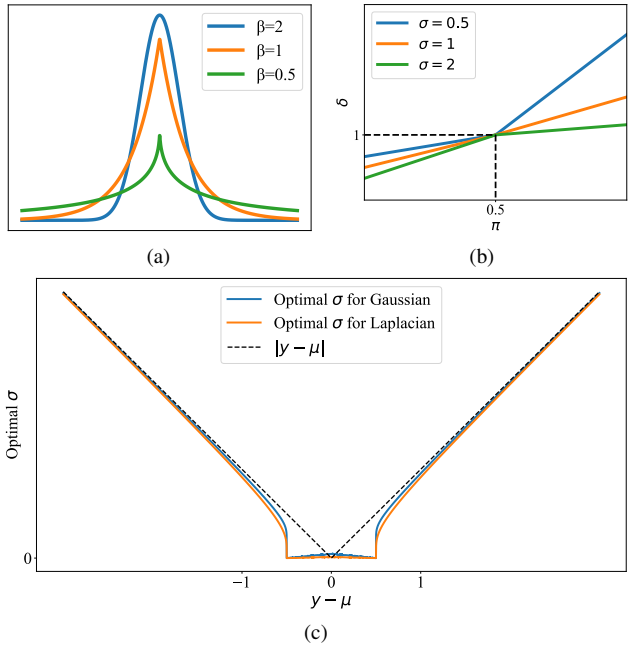

Figure 4. Function diagram: (a) Generalized Gaussian distribution with different shape parameter β while $\mu = 0$. (b) The relationship between likelihood interval radius δ of $\lfloor\mu\rceil$ and $\pi = P(\hat{y} = \lfloor\mu\rceil)$,which is detailed discussed in Section 5.2. (c) Optimal scale σ for $y - \mu$ using Gaussian distribution and Laplacian distribution as the entropy model respectively.

for situations with accurate predictions (μ_i and $\hat{y}_i$ are close), the bitrate is relatively low. Therefore, we consider the case of inaccurate predictions where μ_i and $\hat{y}_i$ have a noticeable difference. To simplify the calculations in this situation, we make the following approximations:

$$\begin{aligned} f(\hat{y}_i, \mu_i, \sigma) :&= \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\hat{y}_i-\mu_i)^2}{2\sigma^2}} \\ &\approx \frac{1}{\sqrt{2\pi}\sigma} \int_{\hat{y}_i-\mu_i-\frac{1}{2}}^{\hat{y}_i-\mu_i+\frac{1}{2}} e^{-\frac{t^2}{2\sigma^2}}\, dt \\ &= \int_{\hat{y}_i-\frac{1}{2}}^{\hat{y}_i+\frac{1}{2}} \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(t-\mu_i)^2}{2\sigma^2}}\, dt \\ &= \left[\mathcal{N}(\mu_i, \sigma^2) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)\right](\hat{y}_i). \end{aligned} \quad (9)$$

Then find σ^{opt} by the partial derivative:

$$\begin{aligned} \frac{\partial f}{\partial \sigma^{opt}} &= \frac{1}{\sqrt{2\pi}} e^{-\frac{(\hat{y}_i-\mu_i)^2}{2(\sigma^{opt})^2}} \left[\frac{(\hat{y}_i-\mu_i)^2}{(\sigma^{opt})^4} - \frac{1}{(\sigma^{opt})^2}\right] \\ &= 0, \end{aligned} \quad (10)$$

so the optimal σ_i for $(\hat{y}_i, \mu_i)$ can be approximated by $\sigma^{opt} = |\hat{y}_i - \mu_i|$. Moreover, we can get the same result if the Gaussian distribution in Equation 2 is replaced with

a mixture Gaussian distribution (fix the mixture weights $\boldsymbol{w}$) or Laplacian distribution. To visually show the relationship between $\sigma^{opt}(y,\mu)$ and $|y-\mu|$, we numerically compute $\sigma^{opt}(y,\mu)$ and present the results in Figure 4c.

In general, the optimal $\boldsymbol{\sigma}$ can be approximated as:

$$\arg\max_{\boldsymbol{\sigma}} p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) \approx |\hat{\boldsymbol{y}}-\boldsymbol{\mu}| \tag{11}$$

From the above deduction, the scales $\boldsymbol{\sigma}$ could be considered as the prediction of the absolute residual between the latent and means.

4.2. Preliminary Improvements for Failed Predictions

As discussed above, we can determine whether the predicted mean μ_i for $\hat{y}_i$ is accurate by σ_i: μ_i and $\hat{y}_i$ are more likely to have a noticeable difference if σ_i is relatively large, which usually needs more bits for arithmetic encoding. The integration intervals for calculating likelihoods are typically at the "tail" of the entropy model for these "failed predictions", thus we can improve the entropy model:

- Adopting a more "heavy-tailed" entropy model, such as generalized Gaussian distribution with shape parameter $\beta < 1$.
- Sacrificing the likelihood interval of integers at the central for the likelihood interval of integers at the tail.

Based on the above discussion, we make preliminary improvements to SparsePCAC, which is a classical variational compression model for point cloud attribute compression with hyperprior and autoregressive context. As SparsePCAC uses Laplacian distribution as its entropy model, we first roughly assume that for i with $\sigma_i > 2$, the predicted mean μ_i for $\hat{y}_i$ is inaccurate, then use more heavy-tailed distributions and likelihood intervals closer to the center for these i without retraining.

Specifically, we directly use the trained model to obtain the latent and parameters of entropy models, but modify the likelihood calculation during encoding and decoding: for $0 < \sigma_i \leq 2$, let

$$p_{\hat{y}_i|\hat{\boldsymbol{z}}}(\hat{y}_i|\hat{\boldsymbol{z}}) = \left[\mathcal{L}(\mu_i,\sigma_i) * \mathcal{U}\left(-\frac{1}{2},\frac{1}{2}\right)\right](\hat{y}_i). \tag{12}$$

For $2 < \sigma_i$ and $|\hat{y}_i - \mu_i| \leq \frac{1}{2}$, let

$$p_{\hat{y}_i|\hat{\boldsymbol{z}}}(\hat{y}_i|\hat{\boldsymbol{z}}) = \left[\mathcal{G}(\mu_i,\sigma_i,0.5) * \mathcal{U}\left(-\frac{1}{4},\frac{1}{4}\right)\right](\hat{y}_i). \tag{13}$$

For $2 < \sigma_i$ and $|\hat{y}_i - \mu_i| > \frac{1}{2}$, let

$$p_{\hat{y}_i|\hat{\boldsymbol{z}}}(\hat{y}_i|\hat{\boldsymbol{z}}) = \left[\mathcal{G}(\mu_i,\sigma_i,0.5) * \mathcal{U}\left(-\frac{1}{2},\frac{1}{2}\right)\right](\hat{y}_i+\Delta) \tag{14}$$

where the offset Δ is:

$$\Delta = \frac{1}{4}\operatorname{sgn}(\mu_i - \hat{y}_i). \tag{15}$$

We use generalized Gaussian distribution with $\beta = 0.5$ for i satisfying $\sigma_i > 2$ in Equation 13 and 14, and narrow the likelihood interval of $\lfloor \mu_i \rceil$, making the likelihood intervals of the remaining integers closer to the center.

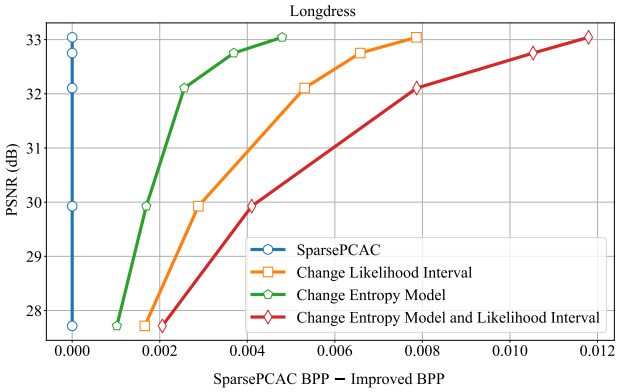

Figure 5. Results of preliminary experiment. To highlight the differences between different methods, the horizontal axis is calculated by subtracting improved bpp from the bpp of SparsePCAC.

Figure 5 shows the results of the above improvements made to the SparsePCAC. From Figure 5, it can be seen that although the model structure has not been modified or retrained, there is still a noticeable improvement in encoding performance.

5. Proposed Method

The overall framework of the proposed method is illustrated in Figure 6 based on the above deduction and experimental results. We utilize the generalized Gaussian entropy model and dynamic likelihood interval to encode latents more efficiently, which will be detailed discussed next.

5.1. Generalized Gaussian Entropy Model

For the input point cloud $\boldsymbol{x}$, we can first obtain quantized latent $\hat{\boldsymbol{y}}$ and hyper latent $\hat{\boldsymbol{z}}$ through autoencoders (g_a, g_s) and (h_a, h_s):

$$\begin{aligned}\hat{\boldsymbol{y}} &= Q(\boldsymbol{y}) = Q(g_a(\boldsymbol{x}))\\ \hat{\boldsymbol{z}} &= Q(\boldsymbol{z}) = Q(h_a(\boldsymbol{y})),\end{aligned} \tag{16}$$

then use the generalized Gaussian distribution as the entropy model for $\hat{\boldsymbol{y}}$:

$$p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) \sim \mathcal{G}(\boldsymbol{\mu},\boldsymbol{\sigma},\boldsymbol{\beta}). \tag{17}$$

We estimate the mean and scale of the generalized Gaussian entropy model by combining hyperprior and context ,

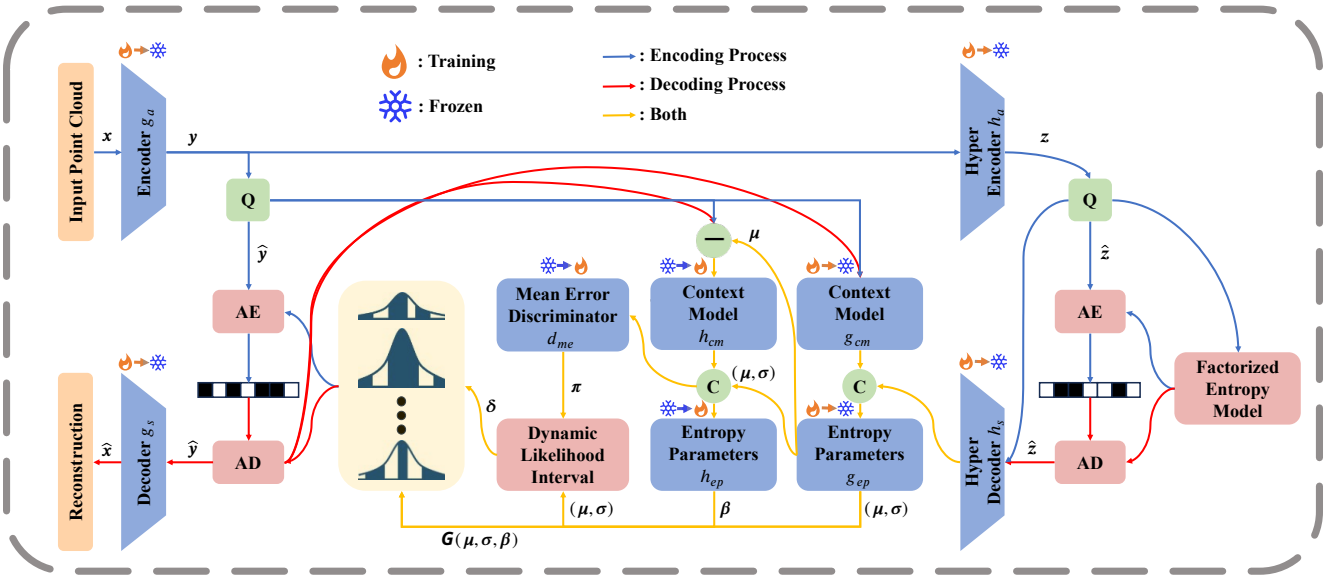

Figure 6. Overall framework. We propose using the generalized Gaussian distribution as the entropy model to better estimate the probability of latents, and dynamically adjust likelihood intervals based on predicted entropy parameters. The entire network is trained by proposed two-step training strategy. **Q** represents quantization and **C** represents concatenation. Blue and red lines represent the encoding and decoding steps respectively, while yellow lines represent the steps that need to be performed in both encoding and decoding.

which is similar to SparsePCAC:

$$(\mu_i, \sigma_i) = g_{ep}(g_{cm}(\hat{\boldsymbol{y}}_{<i}), h_s(\hat{\boldsymbol{z}})). \tag{18}$$

As the likelihood is determined by the probability density function in Equation 3 which depends on $\boldsymbol{x}-\boldsymbol{\mu}$ (i.e., $\hat{\boldsymbol{y}}-\boldsymbol{\mu}$), $\boldsymbol{\sigma}$ and $\boldsymbol{\beta}$, the optimal $\boldsymbol{\beta}$ can be determined given $\hat{\boldsymbol{y}}-\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ in order to maximize the likelihood. Different from μ_i and σ_i, $\hat{y}_i$ is unknown when determining β_i, so we propose another context model h_{cm} to extract context from the absolute difference between predicted $\boldsymbol{\mu}$ and decoded $\hat{\boldsymbol{y}}$:

$$\mathcal{C} = h_{cm}(|\hat{\boldsymbol{y}} - \boldsymbol{\mu}|_{<i}). \tag{19}$$

Therefore, we combine $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ and context $\mathcal{C}$ to estimate $\boldsymbol{\beta}$:

$$\begin{aligned} \beta_i &= h_{ep}(\mathcal{C}, \boldsymbol{\mu}_{\leq i}, \boldsymbol{\sigma}_{\leq i}) \\ &= h_{ep}(h_{cm}(|\hat{\boldsymbol{y}} - \boldsymbol{\mu}|_{<i}), \boldsymbol{\mu}_{\leq i}, \boldsymbol{\sigma}_{\leq i}). \end{aligned} \tag{20}$$

5.2. Dynamic Likelihood Interval

In current methods, when performing arithmetic encoding on $\hat{y}_i$ using a continuous entropy model $\mathcal{G}(\mu_i, \sigma_i, \beta_i)$, the probability of integer n is determined based on fix likelihood intervals:

$$\begin{aligned} P(\hat{y}_i = n) &= \left[\mathcal{G}(\mu_i, \sigma_i, \beta_i) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right)\right](n) \\ &= \int_{n-\frac{1}{2}}^{n+\frac{1}{2}} \mathcal{G}(y|\mu_i, \sigma_i, \beta_i)\, dy. \end{aligned} \tag{21}$$

Results of preliminary experiment indicate that a fixed likelihood interval does not fully utilize the effective information contained in the parameters of the entropy model. In fact, we can determine whether the prediction for $\hat{y}_i$ is accurate based on the entropy model parameters and decoded latent. If the prediction is inaccurate, indicated by a large $|\hat{y}_i - \mu_i|$, we can obtain a more precise discrete probability distribution for arithmetic encoding by shortening the likelihood interval for the central region integers, which causes bigger likelihoods of the integers in the marginal regions as the likelihood intervals of these integers converge toward the center. Conversely, if the prediction is accurate, we can extend the likelihood interval for the central region integers by sacrificing the likelihood interval of the marginal region integers.

However, it is challenging to accurately determine whether the likelihood interval of each integer should be extended or shortened. Therefore, we consider scaling the likelihood interval of $\lfloor \mu_i \rceil$ while translating the likelihood intervals of the remaining integers to obtain a more accurate discrete probability distribution for the sake of model simplicity and effectiveness, which is illustrated in Figure 2.

As discussed above, we should first determine the accuracy of the prediction μ_i for $\hat{y}_i$. Therefore, we estimate the probability of $\hat{y}_i = \lfloor \mu_i \rceil$ using Mean Error Discriminator

Encoder g_a*	Decoder g_s*	Hyper Encoder h_a*	Hyper Decoder h_s*	Context Prediction g_{cm}*, h_{cm}	Entropy Parameters g_{ep}*	Entropy Parameters h_{ep}	Mean Error Discriminator d_{me}
SConv 3^3 c64	TSConv 3^3 c128 s2↑	SConv 3^3 c128	TSConv 3^3 c128 s2↑	Masked 5^3 c256	SConv 1^3 c384	SConv 1^3 c384	SConv 1^3 c384
SConv 3^3 c128 s2↓	SConv 3^3 c128	Leaky ReLU	SConv 3^3 c128		Leaky ReLU	Leaky ReLU	Leaky ReLU
Leaky ReLU	Leaky ReLU	SConv 3^3 c128	Leaky ReLU		SConv 1^3 c256	SConv 1^3 c256	SConv 1^3 c256
SConv 3^3 c128	TSConv 3^3 c128 s2↑	SConv 3^3 c128 s2↓	TSConv 3^3 c256 s2↑		Leaky ReLU	Leaky ReLU	Leaky ReLU
SConv 3^3 c128 s2↓	SConv 3^3 c128	Leaky ReLU	SConv 3^3 c256		SConv 1^3 c256	SConv 1^3 c128	SConv 1^3 c128
Leaky ReLU	Leaky ReLU	SConv 3^3 c128	Leaky ReLU				Leaky ReLU
SConv 3^3 c128	TSConv 3^3 c64 s2↑	SConv 3^3 c128 s2↓	SConv 3^3 c256				Sigmoid
SConv 3^3 c128 s2↓	SConv 3^3 c3						

Table 1. Network Architecture. The modules marked with * refer to SparsePCAC and will be replaced with corresponding modules in subsequent experiments while h_{cm} we propose is consistent with g_{cm}. "SConv" is sparse convolution and "TSConv" is transposed sparse convolution.

d_{me} based on $(\boldsymbol{\mu}, \boldsymbol{\sigma})$ and context $\mathcal{C}$:

$$\begin{aligned}\pi_i :&= P(\hat{y}_i = \lfloor \mu_i \rceil) \\ &= d_{me}(\mathcal{C}, \boldsymbol{\mu}_{\leq i}, \boldsymbol{\sigma}_{\leq i}) \\ &= d_{me}(h_{cm}(|\hat{\boldsymbol{y}} - \boldsymbol{\mu}|_{<i}), \boldsymbol{\mu}_{\leq i}, \boldsymbol{\sigma}_{\leq i}).\end{aligned} \tag{22}$$

Next, We determine the likelihood intervals of $\lfloor \mu_i \rceil$ and the remaining integers separately. In order to determine the likelihood interval of $\lfloor \mu_i \rceil$:

$$LI(\lfloor \mu_i \rceil | \mu_i, \sigma_i, \pi_i) = \mathcal{U}\left(-\frac{\delta_i}{2}, \frac{\delta_i}{2}\right)(\lfloor \mu_i \rceil), \tag{23}$$

we set the range for δ_i as follow:

$$\frac{1}{1+\sigma_i} \leq \delta_i \leq \frac{1}{1-e^{-\sigma_i}}. \tag{24}$$

Equation 24 uses σ to determine the upper bound of stretching ($\frac{1}{1-e^{-\sigma_i}}$, with a range of $(1, +\infty)$) and the lower bound of squeezing ($\frac{1}{1+\sigma_i}$,with a range of $(0, 1)$) of the scaling ratio δ_i. Since larger σ_i means inaccurate μ_i, it is better to reduce the likelihood interval of μ_i at this time, so Equation 24 ensures that both upper bound and lower bound will decrease when σ_i increases. By combining the bounds of δ_i and the probability π_i, the final δ_i is:

$$\delta_i = \begin{cases} 1 + (2\pi_i - 1)(\frac{1}{1-e^{-\sigma_i}} - 1) & \text{if } \pi_i \geq \frac{1}{2} \\ 1 - (1 - 2\pi_i)(1 - \frac{1}{1+\sigma_i}) & \text{if } \pi_i < \frac{1}{2} \end{cases}. \tag{25}$$

Equation 25 uses the relationship between π_i and 0.5 to determine whether to stretch or squeeze, and uses $|2\pi_i - 1|$ (to make the whole function continuous) and σ_i to determine the final δ_i. As shown in Figure 4b, for $\pi_i \geq \frac{1}{2}$, we consider that $\hat{y}_i$ is more likely to be in the tail of the entropy model, which leads us to shorten the likelihood interval of $\lfloor \mu_i \rceil$. Conversely, we expand the likelihood interval of $\lfloor \mu_i \rceil$ for $\pi_i < \frac{1}{2}$.

Naturally, for integer $n \neq \lfloor \mu_i \rceil$, we set its likelihood interval as:

$$LI(n|\mu_i, \sigma_i, \pi_i) = \begin{cases} \mathcal{U}\left(\frac{\delta_i}{2} - 1, \frac{\delta_i}{2}\right)(n) & \text{if } n > \lfloor \mu_i \rceil \\ \mathcal{U}\left(-\frac{\delta_i}{2}, -\frac{\delta_i}{2} + 1\right)(n) & \text{if } n < \lfloor \mu_i \rceil \end{cases}. \tag{26}$$

5.3. Network Architecture

The structures of different neural network modules in our method are outlined in Table 1. Please note that the modules marked with * are the same as SparsePCAC for example. In subsequent experiments, we will replace the structures of these modules with those of corresponding modules in different existing methods.

5.4. Loss Function

We still use the rate-distortion cost with MSE distortion as the loss function for training:

$$\begin{aligned}\mathcal{L} &= R + \lambda \cdot D \\ &= E_{\boldsymbol{x} \sim p_{\boldsymbol{x}}}\left[-\log_2 p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) - \log_2 p_{\hat{\boldsymbol{z}}}(\hat{\boldsymbol{z}})\right] \\ &+ \lambda \cdot E_{\boldsymbol{x} \sim p_{\boldsymbol{x}}}[d(\boldsymbol{x}, \hat{\boldsymbol{x}})],\end{aligned} \tag{27}$$

but the likelihoods of latent $\hat{\boldsymbol{y}}$ need to be calculated according to generalized gaussian entropy model and dynamic likelihood intervals:

$$p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) = \prod_i \left[\mathcal{G}(\mu_i, \sigma_i, \beta_i) * LI\left(\mu_i, \sigma_i, \pi_i\right)\right](\hat{y}_i). \tag{28}$$

6. Experimental Results

6.1. Datastes

To show the model's transferability, we construct the training and testing datasets as follow:

Training Dataset. We use ShapeNet [4] and COCO [16] to construct the training dataset as in [26]. We randomly sample points from the meshes in ShapeNet, apply random rotation to them, and then quantize the coordinates into 8-bit integers to represent the geometry of point clouds. The number of points in each point cloud ranges from 50,000 to 100,000. Next, we randomly select images from COCO and project them onto the point clouds obtained through the aforementioned steps as the attributes of these point clouds. We generate 12,000 samples for traning by this approach.

Testing Dataset. We selected nine point cloud sequences from 8iVFBv2 [1] and MVUB [18] as the testing dataset. Each point cloud sequence contains between 200 and 300 frames, with each frame consisting of 500,000 to 2,000,000 points, which contains rich texture information.

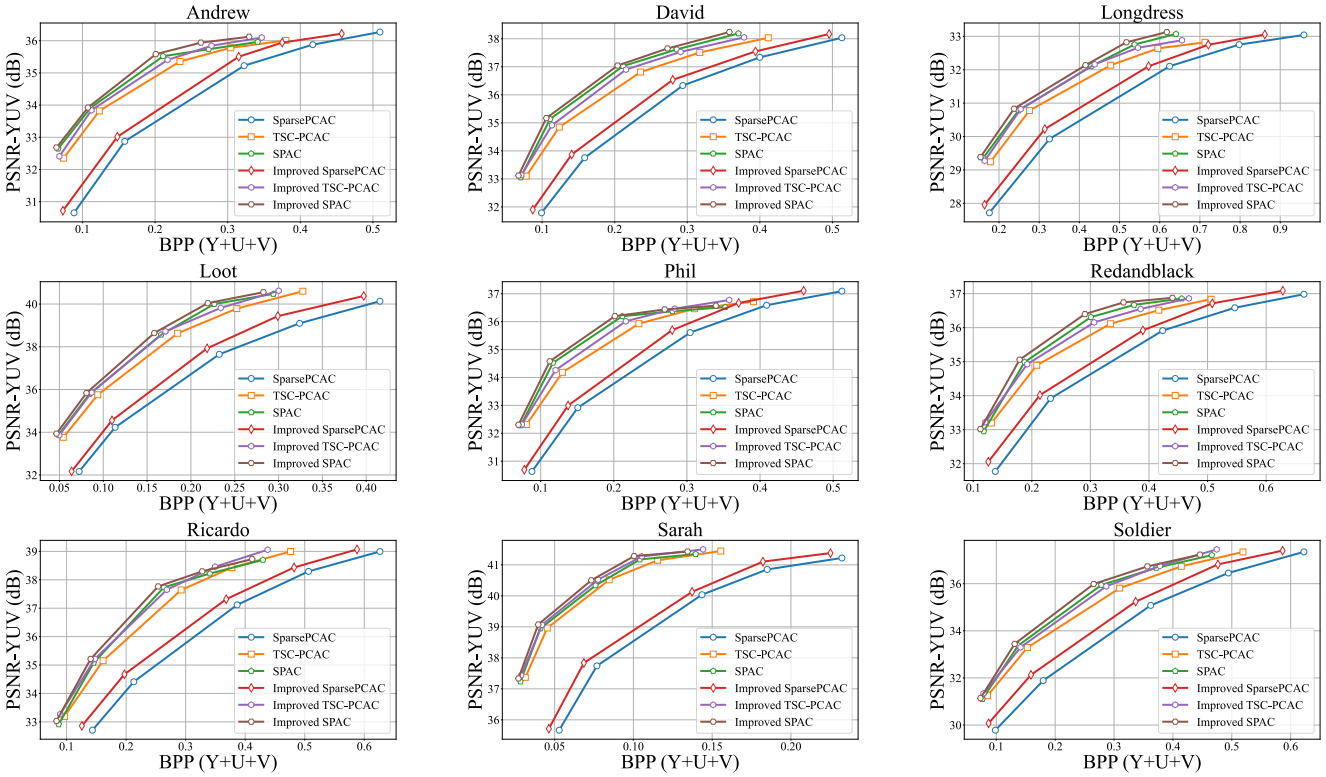

Figure 7. Demonstration of rate-distortion curves for the proposed method applied to three baseline models on nine point cloud sequences.

6.2. Baseline Models

We apply the proposed method to the following VAE-based models to demonstrate the effectiveness of our method.

SparsePCAC: The first variational model with hyperprior and context for point cloud attribute compression, which generates autoregressive context serially using masked sparse convolutions [26].

TSC-PCAC: The TSCM module is introduced based on SparsePCAC, which combines transformer and sparse convolution to better capture both local and global information. Additionally, a channel-based context module is proposed to reduce encoding and decoding time [8].

SPAC: One of the current state-of-the-art frameworks based on VAE. SPAC proposes using Fast Fourier Transform to hierarchically process the point clouds and then compress them by SparsePCAC, where the high-frequency layers could leverage the contex from the low-frequency layers while coding [17].

6.3. Two-Step Training

We apply the generalized Gaussian entropy model and dynamic likelihood intervals proposed in Section 5 to each baseline model in Section 6.2.Then train these improved models and compare the coding results with those of the baseline models. Specifically, we replace the structures of the modules marked with * in Table 1 (i.e., the modules in the classic variational compression model with hyperprior and context) with the corresponding module structures from baseline models. For the sake of simplicity and effectiveness in training, we perform following two-step training to prevent the model from being difficult to converge due to too many parameters:

- First train the modules in the baseline models, including the autoencoder (g_a, g_s), hyper autoencoder (h_a, h_s), context model g_{cm} and the estimating module g_{ep} for mean and scale. Meanwhile, disable h_{cm}, h_{ep} and d_{me} we propose, and set $\boldsymbol{\beta} = 1, \boldsymbol{\pi} = \frac{1}{2}$, thus the generalized Gaussian distribution degenerates into Laplacian distribution and dynamic likelihood intervals degenerates into $\mathcal{U}(-\frac{1}{2}, \frac{1}{2})$:

$$
\begin{aligned}
p_{\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}}(\hat{\boldsymbol{y}}|\hat{\boldsymbol{z}}) &= \prod_i \left[\mathcal{G}(\mu_i, \sigma_i, 1) * LI\left(\mu_i, \sigma_i, \frac{1}{2}\right) \right] (\hat{y}_i) \\
&= \prod_i \left[\mathcal{L}(\mu_i, \sigma_i) * \mathcal{U}\left(-\frac{1}{2}, \frac{1}{2}\right) \right] (\hat{y}_i).
\end{aligned} \tag{29}
$$

- Train h_{cm}, h_{ep} and d_{me} while freezing the modules trained in the first step.

We set $\lambda = 400, 1000, 4000, 8000$ and 16000 in order to get different bitrate points.

Point Clouds	Ours vs SparsePCAC				Ours vs TSC-PCAC				Ours vs SPAC			
	BD-BR(%)		BD-PSNR(dB)		BD-BR(%)		BD-PSNR(dB)		BD-BR(%)		BD-PSNR(dB)	
	Y	YUV	Y	YUV	Y	YUV	Y	YUV	Y	YUV	Y	YUV
Andrew	-10.46	-11.97	0.35	0.38	-8.82	-9.99	0.21	0.24	-6.07	-6.38	0.13	0.14
David	-9.92	-11.93	0.41	0.46	-9.37	-10.09	0.31	0.32	-5.18	-5.35	0.17	0.17
Longdress	-10.39	-10.65	0.34	0.36	-8.87	-9.31	0.24	0.25	-5.29	-5.70	0.15	0.16
Loot	-10.83	-11.15	0.52	0.54	-8.43	-9.67	0.36	0.39	-4.29	-5.76	0.18	0.22
Phil	-9.35	-10.97	0.39	0.42	-7.95	-9.40	0.25	0.29	-5.03	-5.92	0.13	0.15
Redandblack	-10.06	-10.50	0.35	0.36	-8.02	-8.83	0.24	0.25	-5.24	-5.87	0.20	0.20
Ricardo	-9.74	-11.07	0.47	0.49	-8.76	-9.24	0.33	0.36	-5.20	-5.93	0.21	0.22
Sarah	-10.85	-11.00	0.39	0.42	-8.34	-10.11	0.27	0.28	-6.38	-8.13	0.21	0.24
Soldier	-11.96	-13.93	0.59	0.61	-7.99	-8.91	0.29	0.32	-4.77	-5.96	0.18	0.21
Average	**-10.40**	**-11.46**	**0.42**	**0.45**	**-8.51**	**-9.51**	**0.28**	**0.30**	**-5.27**	**-6.11**	**0.17**	**0.19**

Table 2. BD-BR and BD-PSNR comparisons between three baseline models improved by proposed method and original results.

Point Clouds	SparsePCAC with GGEM		SparsePCAC with DLI	
	BD-BR(%)	BD-PSNR(dB)	BD-BR(%)	BD-PSNR(dB)
Andrew	-5.94	0.18	-6.51	0.30
David	-5.03	0.22	-6.60	0.28
Longdress	-5.79	0.16	-6.47	0.29
Loot	-5.19	0.21	-7.70	0.30
Phil	-6.08	0.17	-5.49	0.25
Redandblack	-5.48	0.20	-6.46	0.28
Ricardo	-5.82	0.16	-6.16	0.28
Sarah	-5.11	0.18	-7.94	0.27
Soldier	-5.90	0.19	-8.13	0.31
Average	**-5.59**	**0.19**	**-6.83**	**0.28**

Table 3. Ablation study for GGEM (generalized Gaussian entropy model) and DLI (dynamic likelihood interval) based on SparsePCAC.

6.4. Rate-Distortion Performance

Figure 7 illustrates the rate-distortion performance of different methods in nine point cloud sequences. Table 2 shows the BD-BR and BD-PSNR gains achieved by applying generalized Gaussian entropy model and dynamic likelihood interval to baseline models.

As shown in Figure 7 and Table 2, our method improves performance across all baseline models. The largest improvement is observed for SparsePCAC, which achieves an average bitrate reduction of 11.46%, followed by TSC-PCAC with an average bitrate reduction of 9.51%. The smallest improvement is observed for SPAC, which still results in an average bitrate saving of 6.11%. We be believe this outcome can be attributed to the fact that SparsePCAC has the weakest performance among the baseline models, with less accurate entropy parameter estimation. Therefore, adjustments to the entropy model and likelihood intervals are more effective in this case. In contrast, SPAC already has relatively accurate estimation for mean and scale parameters, so even optimal adjustments lead to a smaller improvement in performance.

6.5. Computational Complexity

Table 4 illustrates the encoding and decoding time for compressing point cloud sequences using different methods.

Our method does not significantly affect the parallelism of baseline models' encoding and decoding, as we set the second context module h_{cm} to be the same as g_{cm}. For TSC-PCAC with the highest parallelism, our method incurs the least additional encoding and decoding time. In contrast, for SparsePCAC, which uses a fully serial context module, our method incurs the most additional time.

Methos	Ours vs SparsePCAC		Ours vs TSC-PCAC		Ours vs SPAC	
	Origin	Ours	Origin	Ours	Origin	Ours
Enc.(s)	74.05	163.74	3.92	8.24	132.60	287.59
Dec.(s)	427.63	904.17	7.60	17.80	108.58	231.14

Table 4. Comparison of complexity. Take the average time of encoding and decoding all point clouds as results.

6.6. Ablation Study

In order to separately demonstrate the effectiveness of generalized Gaussian entropy model and dynamic likelihood interval, we add only one of these two modules to SparsePCAC then perform two-step training, and the results are shown in Table 3. We can learn from Table 3 that both generalized Gaussian entropy model and dynamic likelihood interval have a positive effect on improving coding performance, which is consistent with the results from the preliminary experiment.

7. Conclusion and Future Work

We conduct simple experiments to demonstrate the limitations of current Gaussian and Laplace entropy models. Then we propose using generalized Gaussian distribution for more accurate probability estimation of latents and leverage MED to dynamically adjust the likelihood intervals of integers. Our method significantly improves the RD performance of three tested VAE-based models. Since our method only adds a few modules to original models without fundamentally changing their architecture, and can be trained by proposed two-step training, it is easily portable to other SOTA VAE-based models.

In future work, we will focus on enhancing the parallelism of proposed modules and testing our method on other compression tasks, such as image and video compression.

References

[1] 8i Labs. 8i voxelized full bodies version 2 – a voxelized point cloud dataset. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document, 2017. 1, 6

[2] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. *arXiv preprint arXiv:1802.01436*, 2018. 1, 2

[3] Ronit Bustin, H. Vincent Poor, and Shlomo Shamai. Analytical properties of generalized gaussian distributions. *Journal of Statistical Distributions and Applications*, 5(1):6, 2018. 3

[4] Angel X. Chang, Thomas Funkhouser, Leonidas J. Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. ShapeNet: An Information-Rich 3D Model Repository. Technical Report Tech Report, Stanford University, 2015. 6

[5] Zhengxue Cheng, Heming Sun, Masaru Takeuchi, and Jiro Katto. Learned image compression with discretized gaussian mixture likelihoods and attention modules. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7939–7948, 2020. 2

[6] Dandan Ding, Junzhe Zhang, Jianqiang Wang, and Zhan Ma. Carnet: Compression artifact reduction for point cloud attribute. *arXiv preprint arXiv:2209.08276*, 2022. 2

[7] 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*, pages 625–633, 2022. 1

[8] Zixi Guo, Yun Zhang, Linwei Zhu, Hanli Wang, and Gangyi Jiang. Tsc-pcac: Voxel transformer and sparse convolution-based point cloud attribute compression for 3d broadcasting. *IEEE Transactions on Broadcasting*, 2024. 2, 7

[9] Dailan He, Yaoyan Zheng, Baocheng Sun, Yan Wang, and Hongwei Qin. Checkerboard context model for efficient learned image compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14771–14780, 2021. 2

[10] Dailan He, Ziming Yang, Weikun Peng, Rui Ma, Hongwei Qin, and Yan Wang. Elic: Efficient learned image compression with unevenly grouped space-channel contextual adaptive coding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5718–5727, 2022. 2

[11] Wei Jiang, Jiayu Yang, Yongqi Zhai, Feng Gao, and Ronggang Wang. Mlic++: Linear complexity multi-reference entropy modeling for learned image compression. *arXiv preprint arXiv:2307.15421*, 2023.

[12] Wei Jiang, Jiayu Yang, Yongqi Zhai, Peirong Ning, Feng Gao, and Ronggang Wang. Mlic: Multi-reference entropy model for learned image compression. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 7618–7627, 2023. 2

[13] Diederik P Kingma. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013. 1

[14] Ge Li, Wei Gao, and Wen Gao. Mpeg geometry-based point cloud compression (g-pcc) standard. In *Point Cloud Compression: Technologies and Standardization*, pages 135–165. Springer, 2024. 1

[15] Ge Li, Wei Gao, and Wen Gao. Mpeg video-based point cloud compression (v-pcc) standard. In *Point Cloud Compression: Technologies and Standardization*, pages 199–218. Springer, 2024. 1

[16] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common Objects in Context. *CoRR*, abs/1405.0312, 2014. 6

[17] Xiaolong Mao, Hui Yuan, Tian Guo, Shiqi Jiang, Raouf Hamzaoui, and Sam Kwong. Spac: Sampling-based progressive attribute compression for dense point clouds. *arXiv preprint arXiv:2409.10293*, 2024. 2, 7

[18] Microsoft. Voxelized Upper Bodies – A Voxelized Point Cloud Dataset. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document, 2016. 1, 6

[19] David Minnen and Saurabh Singh. Channel-wise autoregressive entropy models for learned image compression. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 3339–3343. IEEE, 2020. 2

[20] David Minnen, Johannes Ballé, and George D Toderici. Joint autoregressive and hierarchical priors for learned image compression. *Advances in neural information processing systems*, 31, 2018. 2

[21] Changhao Peng and Wei Gao. Laplacian matrix learning for point cloud attribute compression with ternary search-based adaptive block partition. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 10412–10420, 2024. 2

[22] Amir Said. Introduction to Arithmetic Coding – Theory and Practice. Technical Report cs/2302.00819, arXiv, 1999. 2

[23] Yiting Shao, Zhaobin Zhang, Zhu Li, Kui Fan, and Ge Li. Attribute compression of 3d point clouds using laplacian sparsity optimized graph transform. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE, 2017. 2

[24] Fei Song, Ge Li, Xiaodong Yang, Wei Gao, and Thomas H Li. Fine-grained correlation representation for graph-based point cloud attribute compression. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2022. 2

[25] Rui Song, Chunyang Fu, Shan Liu, and Ge Li. Efficient hierarchical entropy model for learned point cloud compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14368–14377, 2023. 1

[26] 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, 2022. 1, 2, 6, 7

[27] 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*, 2024. 2

[28] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 3

[29] Florian Wirth, Jannik Quehl, Jeffrey Ota, and Christoph Stiller. Pointatme: efficient 3d point cloud labeling in virtual reality. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1693–1698. IEEE, 2019. 1

[30] Xiangyu Yue, Bichen Wu, Sanjit A Seshia, Kurt Keutzer, and Alberto L Sangiovanni-Vincentelli. A lidar point cloud generator: from a virtual world to autonomous driving. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, pages 458–464, 2018. 1

[31] Junteng Zhang, Jianqiang Wang, Dandan Ding, and Zhan Ma. Scalable point cloud attribute compression. *IEEE Transactions on Multimedia*, 2023. 2