

4DGC: Rate-Aware 4D Gaussian Compression for Efficient Streamable Free-Viewpoint Video

Supplementary Material

1. More Implementation Details

1.1. Entropy Model

Here, we provide a detailed introduction to our implicit entropy model. This model estimates entropy by learning a cumulative distribution function (CDF) that represents the probability distribution of the input data. The model comprises multiple layers, each parameterized by weight matrices, biases, and scaling factors. These parameters transform the input tensor through a series of operations, such as matrix multiplication and bias addition, combined with non-linear activations like softplus. Each layer progressively refines the input values to approximate a CDF, capturing the cumulative probability distribution of the data.

To maintain precision, we avoid assuming any predefined distribution for the data. Instead, we construct a novel distribution within the entropy model to closely approximate the actual data distribution. Specifically, the entropy model computes cumulative logits for values just below and above the actual data point. This approach enables the model to capture the probability interval that contains the data point, thereby improving estimation accuracy. The likelihood, which represents the probability of the input within this interval, is calculated as the difference between the sigmoid activations of these cumulative logits. This likelihood is then incorporated into the overall loss function during training.

When the training process is complete, we apply quantization and a range coder for entropy coding to further compress the data volume and generate the bitstream. The entropy model itself occupies about 100 KB per frame, which is relatively large compared to the compressed motion grid \mathbf{M}_t and the compensated Gaussians $\Delta\mathbf{G}_t$. Therefore, we analyze the actual distribution of the data prior ω_t before entropy coding and transmit ω_t instead of the entire entropy model. This approach further reduces the size of each frame.

The process of entropy encoding can be represented as follows:

$$\begin{aligned} \mathbf{Q}(x) &= \lfloor q \cdot x + 0.5 \rfloor, \\ B_t &= \mathbf{E}(\mathbf{Q}(x) - \mathbf{Q}(\min(x)); \omega_t). \end{aligned} \quad (1)$$

Here, x represents the data to be compressed, which includes \mathbf{M}_t and $\Delta\mathbf{G}_t$. The bitstream after entropy encoding, denoted as B_t , consists of $B_t^{\mathbf{M}}$ and $B_t^{\Delta\mathbf{G}}$, corresponding to \mathbf{M}_t and $\Delta\mathbf{G}_t$, respectively. The range encoder is represented by \mathbf{E} .

Table 1. The average size of each component of inter-frames on the N3DV dataset.

	$B_t^{\mathbf{M}}$	$B_t^{\Delta\mathbf{G}}$	$\omega_t^{\mathbf{M}}$	$\omega_t^{\Delta\mathbf{G}}$	Total
Size (KB)	164.72	65.88	0.25	0.17	231.02

To enable compression into the int8 format, we convert the compressed data into non-negative values and subsequently restore it to its original range during decompression. The variable q denotes the quantization parameter. During quantization, the data is multiplied by q , which effectively expands its range and subtly enhances the precision of the quantization process. On the decoding side, the entropy decoding process can be expressed as follows:

$$\hat{x} = \frac{\mathbf{D}(B_t; \omega_t) + \mathbf{Q}(\min(x))}{q}, \quad (2)$$

where \mathbf{D} is the range decoder. Therefore, for each inter-frame, the data to be transmitted includes the bitstream $B_t^{\mathbf{M}}$ and $B_t^{\Delta\mathbf{G}}$, and the data distributions $\omega_t = \{\omega_t^{\mathbf{M}}, \omega_t^{\Delta\mathbf{G}}\}$. The size of each of these components is detailed in Tab. 1.

1.2. Hyperparameters Settings

In this section, we provide a more detailed explanation of the hyperparameter settings for the two aspects.

Model Parameter Settings: We use two shared global lightweight MLPs Φ_μ and Φ_R , both with an input dimension of 20, a hidden layer size of 64, and output dimensions of 3 and 4, respectively. For the multi-resolution motion grid \mathbf{M}_t , we use feature grid channels of 4, 4, and 2, with dimensions 32^3 , 64^3 , and 128^3 .

Gaussian Compensation Parameter Settings: For Gaussian compensation, we choose $\tau_g = 0.0001$, $\tau_\mu = 0.08$, and $\tau_R = \frac{\pi}{4}$. We only apply the second type of Gaussian compensation to the Gaussians whose scale $s > -0.01$. (The actual scale is activated using the exponential function.) To prevent an excessive increase in the number of Gaussians, we filter out Gaussians with opacity $\alpha < 0.01$ after stage 2.

2. More Results

2.1. Quantitative Results

We provide a quantitative comparison of image quality, measured by PSNR, and model size across all scenes in the N3DV dataset in Tab. 2. To further demonstrate the variable bitrate characteristic and superior RD performance of

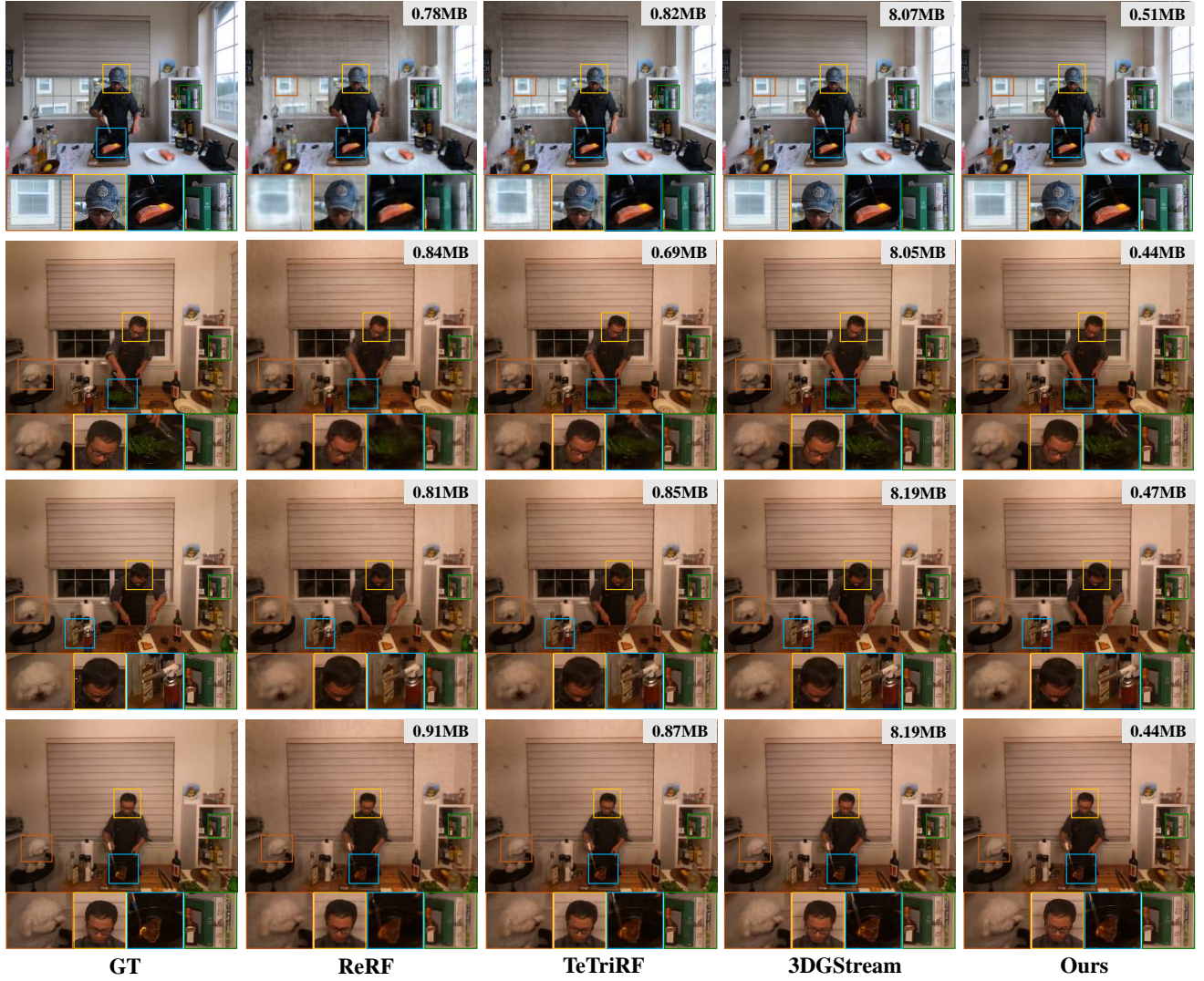


Figure 1. More Qualitative comparison on more sequences of the N3DV dataset against ReRF, TeTriRF, and 3DGStream.

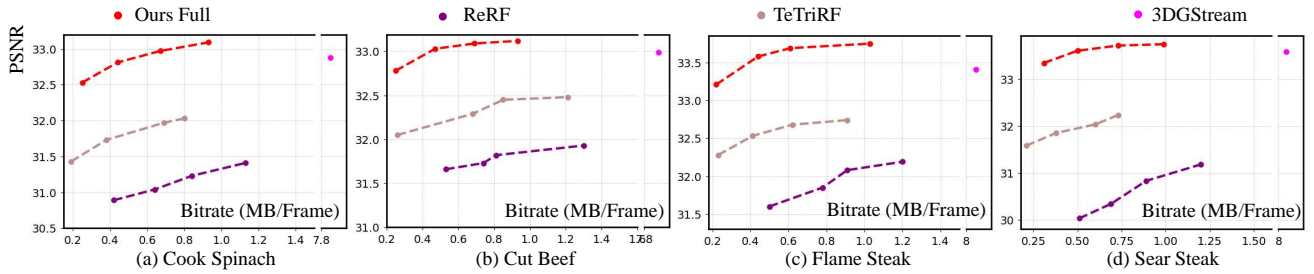


Figure 2. More Rate-distortion curves across different sequences of N3DV datasets.

our method, we present additional RD curves for more sequences in Fig. 2.

2.2. Qualitative Results

We have prepared more qualitative comparison results in the Fig. 1. We also prepared videos to show the free view synthesis results on various scenes from the N3DV dataset

Table 2. Quantitative comparison of average PSNR values(dB) and model size(MB) across all sequences in the N3DV dataset.

Method	Coffee Martini	Cook Spinach	Cut Beef	Flame Salmon	Flame Steak	Sear Steak	Mean
StreamRF	27.77/9.34	31.54/7.48	31.74/7.17	28.19/7.93	32.18/7.02	32.29/6.88	30.61/7.64
ReRF	26.24/0.79	31.23/0.84	31.82/0.81	26.80/0.78	32.08/0.91	30.03/0.51	29.71/0.77
TeTriRF	27.10/0.73	31.97/0.69	32.45/0.85	27.61/0.82	32.74/0.87	32.03/0.60	30.65/0.76
3DGStream	27.96/8.00	32.88 /8.05	32.99/8.19	28.52 /8.07	33.41/8.19	33.58/8.16	31.54/8.11
Ours	27.98/0.58	32.81/ 0.44	33.03/0.47	28.49/ 0.51	33.58/0.44	33.60/0.50	31.58/0.49

in the supplementary materials.