Neural Compression for 3D Geometry Sets (Supplementary Material)

In this supplementary material, we provide more details about the regular geometry representation of our NeCGS in Sec. 1, the auto-decoder neural network used in NeCGS in Sec. 2, and additional experimental settings and results in Sec. 3.

1. Details of Regular Geometry Representation

1.1. Tensor Quantization

Denoted by $\mathbf{x}$ a tensor. We quantize it in a fixed interval, $[a, b]$, at $(2^N + 1)$ levels[1] by

$$\mathcal{Q}(\mathbf{x}) = \texttt{Round}\left(\frac{\texttt{Clamp}(\mathbf{x}, a, b) - a}{s}\right) \times s + a, \tag{4}$$

where $s = (b - a)/2^N$. In our experiment, we set $a = -1$ and $b = 1$.

1.2. Optimization of TSDF-Def Tensor

We set a series of camera pose, $\mathcal{T} = \{\mathbf{T}_i\}_{i=1}^{E}$, around the meshes. Let $\mathbf{I}_1^{\mathrm{D}}(\mathbf{T}_i)$ and $\mathbf{I}_2^{\mathrm{D}}(\mathbf{T}_i)$ represent the depth images obtained from the reconstructed mesh $\texttt{DMC}(\mathbf{V})$ and the given mesh $\mathbf{S}$ at the pose $\mathbf{T}_i$ respectively. Similarly, let $\mathbf{I}_1^{\mathrm{M}}(\mathbf{T}_i)$ and $\mathbf{I}_2^{\mathrm{M}}(\mathbf{T}_i)$ denote their respective silhouette images at pose $\mathbf{T}_i$. The reconstruction error produced by silhouette and depth images at all pose are

$$\mathcal{E}_{\mathrm{M}}(\texttt{DMC}(\mathbf{V}), \mathbf{S}) = \sum_{\mathcal{T}_i \in \mathcal{T}} \|\mathbf{I}_1^{\mathrm{M}}(\mathbf{T}_i) - \mathbf{I}_2^{\mathrm{M}}(\mathbf{T}_i)\|_1 \tag{5}$$

and

$$\mathcal{E}_{\mathrm{D}}(\texttt{DMC}(\mathbf{V}), \mathbf{S}) = \sum_{\mathcal{T}_i \in \mathcal{T}} \| \left(\mathbf{I}_1^{\mathrm{D}}(\mathbf{T}_i) - \mathbf{I}_2^{\mathrm{D}}(\mathbf{T}_i)\right) * \mathbf{I}_2^{\mathrm{M}}(\mathbf{T}_i)\|_1. \tag{6}$$

Then the reconstruction error is defined as

$$\mathcal{E}_{\mathrm{Rec}}(\texttt{DMC}(\mathbf{V}), \mathbf{S}) = \mathcal{E}_{\mathrm{M}}(\texttt{DMC}(\mathbf{V}), \mathbf{S}) + \lambda_{\mathrm{rec}} \mathcal{E}_{\mathrm{D}}(\texttt{DMC}(\mathbf{V}), \mathbf{S}), \tag{7}$$

where $E = 4$ and $\lambda_{\mathrm{rec}} = 10$ in our experiment.

[1] We partition the interval $[a, b]$ into $(2^N + 1)$ levels, rather than 2^N levels, to ensure the inclusion of the value 0.

2. Auto-decoder-based Neural Compression

2.1. Upsampling Module

In each upsampling module, we utilize a PixelShuffle layer [2] between the convolution and activation layers to upscale the input, as shown in Fig. 13. The input feature tensor has dimensions $(N_{\text{in}}, N_{\text{in}}, N_{\text{in}}, C_{\text{in}})$, with an upsampling scale of s and an output channel count of C_{out}.

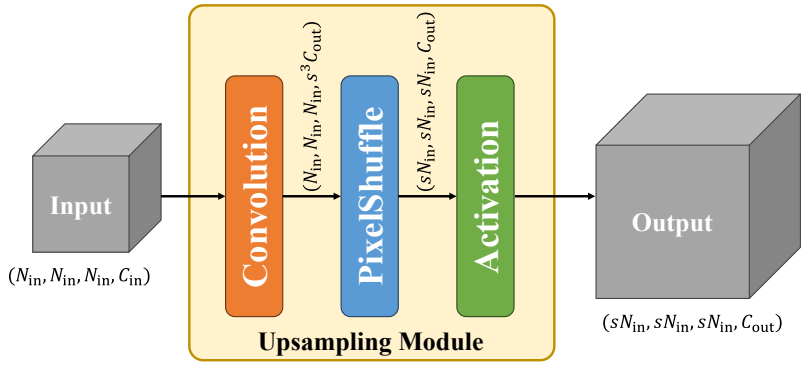


Figure 13. Upsampling Module.

3. Experiment

3.1. Evaluation Metric

Let $\mathbf{S}_{\text{Rec}}$ and $\mathbf{S}_{\text{GT}}$ denote the reconstructed and ground-truth 3D shapes, respectively. We then randomly sample $N_{\text{eval}} = 10^5$ points on them, obtaining two point clouds, $\mathbf{P}_{\text{Rec}}$ and $\mathbf{P}_{\text{GT}}$. For each point of $\mathbf{P}_{\text{Rec}}$ and $\mathbf{P}_{\text{GT}}$, the normal of the triangle face where it is sampled is considered to be its normal vector, and the normal sets of $\mathbf{P}_{\text{Rec}}$ and $\mathbf{P}_{\text{GT}}$ are denoted as $\mathbf{N}_{\text{Rec}}$ and $\mathbf{N}_{\text{GT}}$, respectively. Let $\texttt{NN\_Point}(\mathbf{x}, \mathbf{P})$ be the operator that returns the nearest point of $\mathbf{x}$ in the point cloud $\mathbf{P}$. The CD between them is defined as

$$\begin{aligned}\texttt{CD}(\mathbf{S}_{\text{Rec}}, \mathbf{S}_{\text{GT}}) =& \frac{1}{2N_{\text{eval}}} \sum_{\mathbf{x}\in\mathbf{P}_{\text{Rec}}} \|\mathbf{x} - \texttt{NN\_Point}(\mathbf{x}, \mathbf{P}_{\text{GT}})\|_2 \\ &+ \frac{1}{2N_{\text{eval}}} \sum_{\mathbf{x}\in\mathbf{P}_{\text{GT}}} \|\mathbf{x} - \texttt{NN\_Point}(\mathbf{x}, \mathbf{P}_{\text{Rec}})\|_2.\end{aligned} \tag{8}$$

Let $\texttt{NN\_Normal}(\mathbf{x}, \mathbf{P})$ be the operator that returns the normal vector of the point $\mathbf{x}$'s nearest point in the point cloud $\mathbf{P}$. The NC is defined as

$$\begin{aligned}\texttt{NC}(\mathbf{S}_{\text{Rec}}, \mathbf{S}_{\text{GT}}) =& \frac{1}{2N_{\text{eval}}} \sum_{\mathbf{x}\in\mathbf{P}_{\text{Rec}}} |\mathbf{N}_{\text{Rec}}(\mathbf{x}) \cdot \texttt{NN\_Normal}(\mathbf{x}, \mathbf{P}_{\text{GT}})| \\ &+ \frac{1}{2N_{\text{eval}}} \sum_{\mathbf{x}\in\mathbf{P}_{\text{GT}}} |\mathbf{N}_{\text{GT}}(\mathbf{x}) \cdot \texttt{NN\_Normal}(\mathbf{x}, \mathbf{P}_{\text{Rec}})|.\end{aligned} \tag{9}$$

F-Score is defined as the harmonic mean between the precision and the recall of points that lie within a certain distance threshold ϵ between $\mathbf{S}_{\text{Rec}}$ and $\mathbf{S}_{\text{GT}}$,

$$\texttt{F} - \texttt{Score}(\mathbf{S}_{\text{Rec}}, \mathbf{S}_{\text{GT}}, \epsilon) = \frac{2 \cdot \texttt{Recall} \cdot \texttt{Precision}}{\texttt{Recall} + \texttt{Precision}}, \tag{10}$$

where

$$\begin{aligned}
\texttt{Recall}(\mathbf{S}_{\mathrm{Rec}}, \mathbf{S}_{\mathrm{GT}}, \epsilon) &= \left| \left\{ \mathbf{x}_1 \in \mathbf{P}_{\mathrm{Rec}}, \text{s.t.} \min_{\mathbf{x}_2 \in \mathbf{P}_{\mathrm{GT}}} \|\mathbf{x}_1 - \mathbf{x}_2\|_2 < \epsilon \right\} \right|, \\
\texttt{Precision}(\mathbf{S}_{\mathrm{Rec}}, \mathbf{S}_{\mathrm{GT}}, \epsilon) &= \left| \left\{ \mathbf{x}_2 \in \mathbf{P}_{\mathrm{GT}}, \text{s.t.} \min_{\mathbf{x}_1 \in \mathbf{P}_{\mathrm{Rec}}} \|\mathbf{x}_1 - \mathbf{x}_2\|_2 < \epsilon \right\} \right|.
\end{aligned} \tag{11}$$

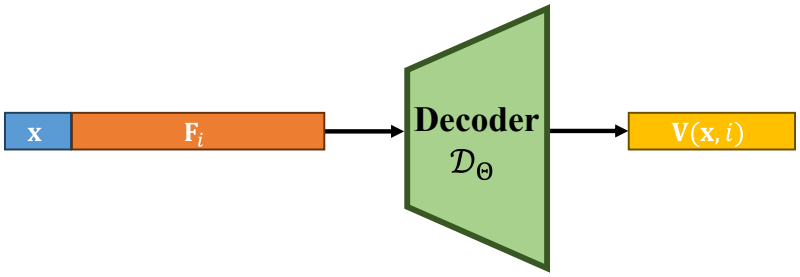


Figure 14. Pipeline of QuantDeepSDF.

3.2. QuantDeepSDF

Compared to DeepSDF [1], our QuantDeepSDF incorporates the following two modifications:

- The decoder parameters are quantized to enhance compression efficiency.
- To maintain consistency with our NeCGS, the points sampled during training are drawn from TSDF-Def tensors.

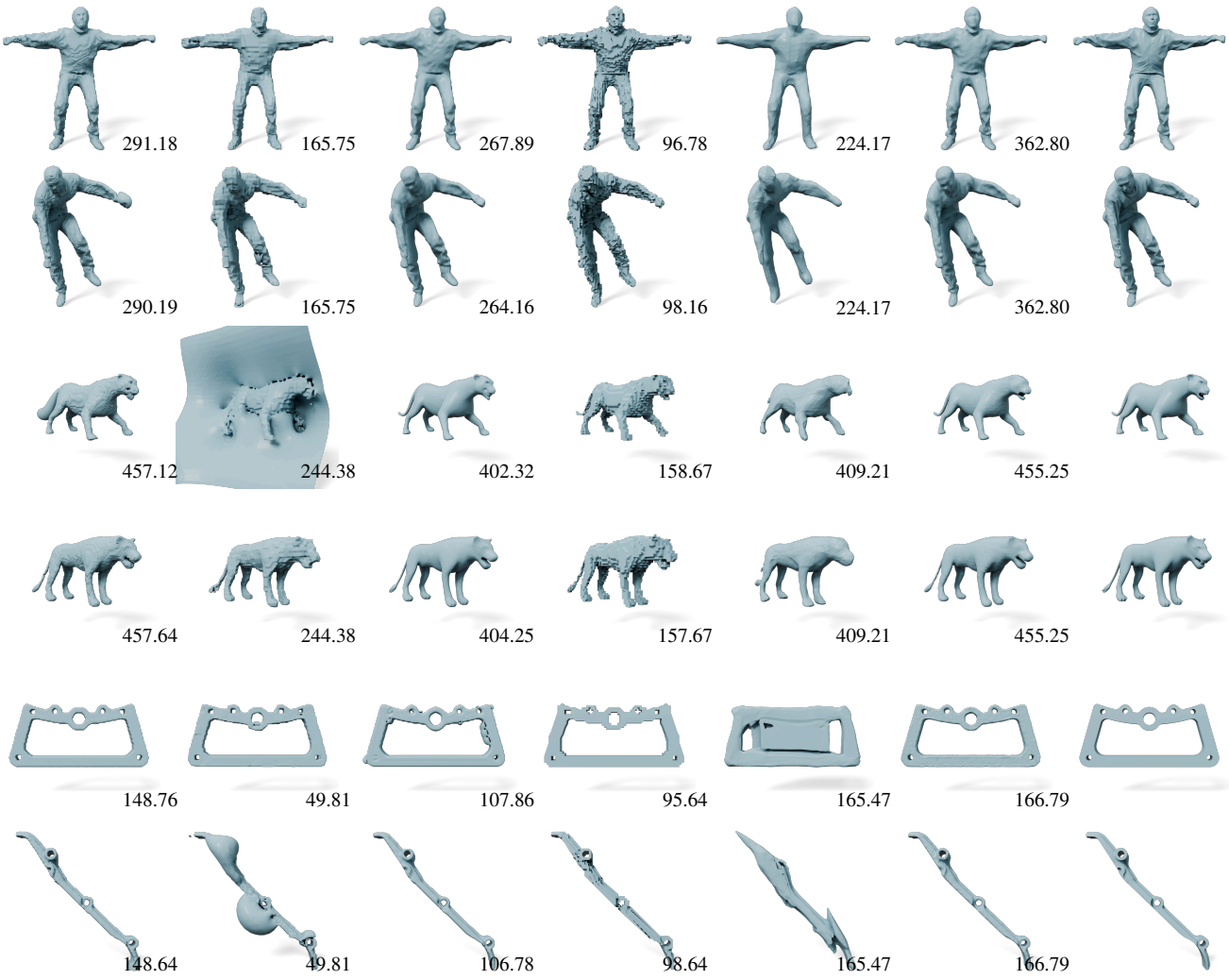
The pipeline of QuantDeepSDF is shown in Fig. 14. Specifically, the decoder is an MLP, where the input is the concatenated vector of coordinate $\mathbf{x} \in \mathbb{R}^3$ and the i-th embedded feature vector $\mathbf{F}_i \in \mathbb{R}^C$, and the output is the corresponding TSDF-Def value. In our experiment, the decoder consists of 8 layers, and the compression ratio is controled by changing the width of each layer.

3.3. Auto-Encoder in Ablation Study

Different from the auto-encoder used in our framework, where the embed features are directly optimized, auto-encoder utilizes an encoder to produce the embedded features, where the inputs are the TSDF-Def tensors. And the decoder is kept the same as our framework. During the optimization, the parameters of encoder and decoder are optimized. Once optimized, the embedded features produced by the encoder and decoder parameters are compressed into bitstreams.

3.4. More Visual Results

Fig. 15 shows more visual comparison between different methods on the AMA dataset, DT4D dataset, and Thingi10K dataset. Obviously, the decompressed models through our method have less distortion than baseline methods. Fig. 16 showcases the decompressed models with detailed structures obtained using our method, demonstrating that the detailed structures are well-preserved after decompression. Figs. 17, 18 and 19 depicts the visual results of the decompressed models from these datasets under various compression ratios. With the compression ratio increasing, the decompressed models still preserve the detailed structures, without large distortion.



(a) GPCC (b) VPCC (c) PCGCv2 (d) Draco (e) QuantDeepSDF (f) Ours (g) Original

Figure 15. Visual comparisons of different compression methods. All numbers in corners represent the compression ratio. Zoom in for details.

References

[1] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174, 2019. 3

[2] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1874–1883, 2016. 2

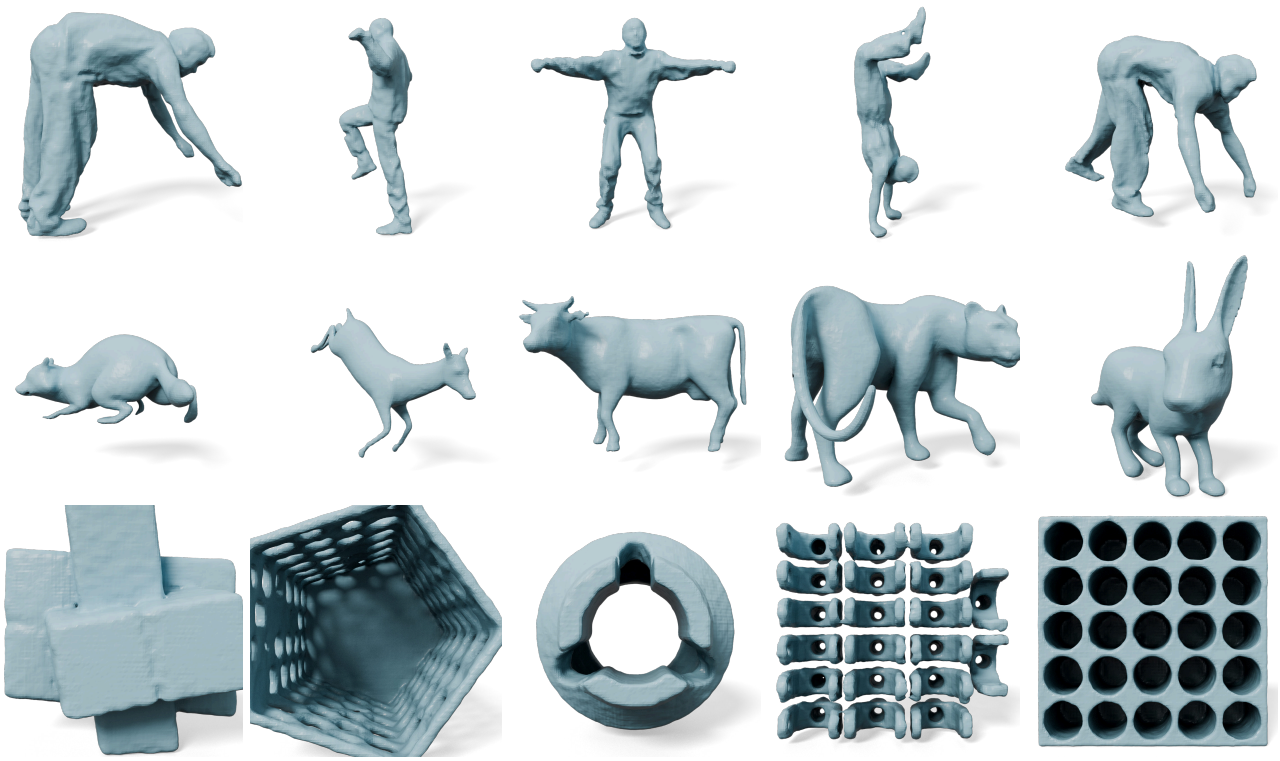
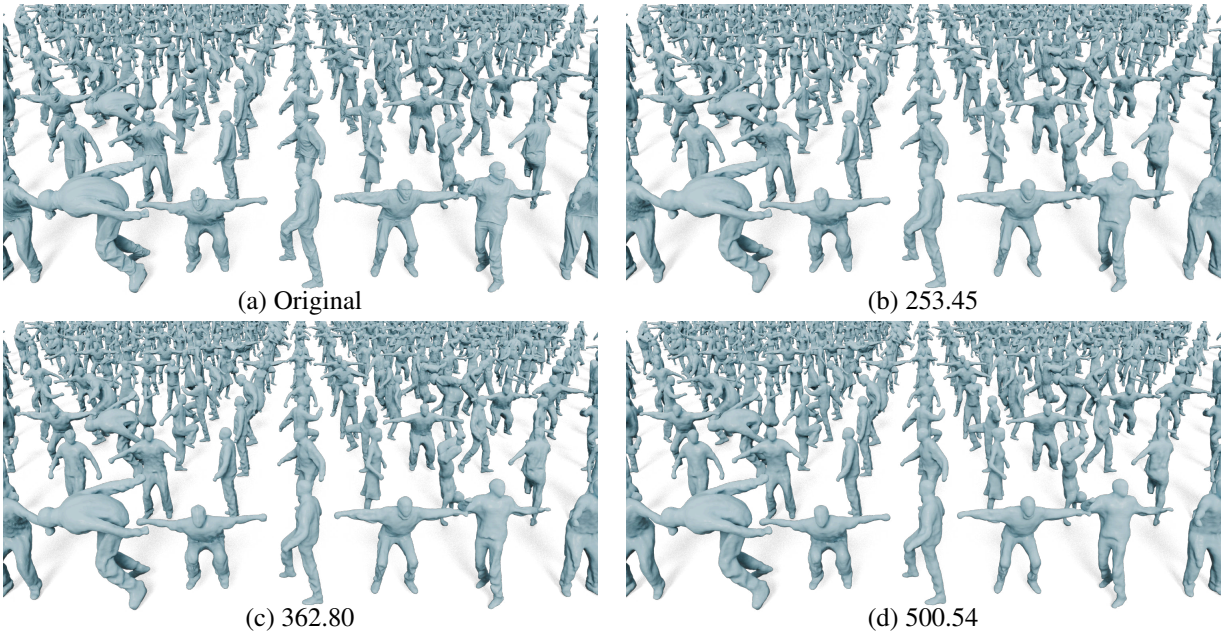


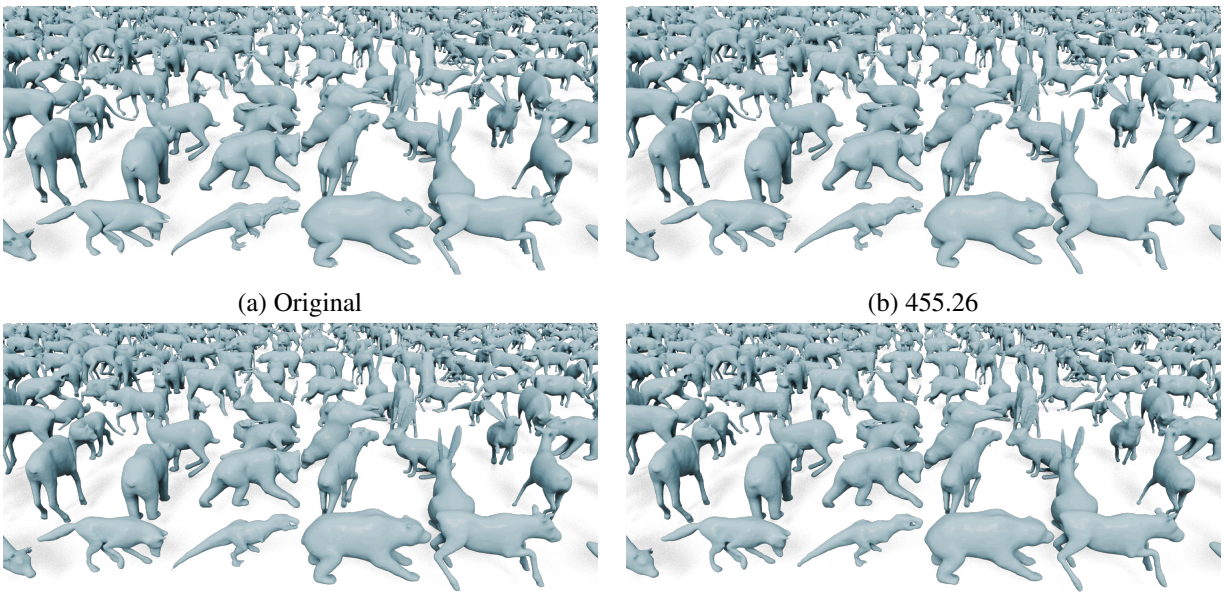
Figure 16. Visualization of the decompressed models with detailed structures. Zoom in for details.



(a) Original (b) 253.45

(c) 362.80 (d) 500.54

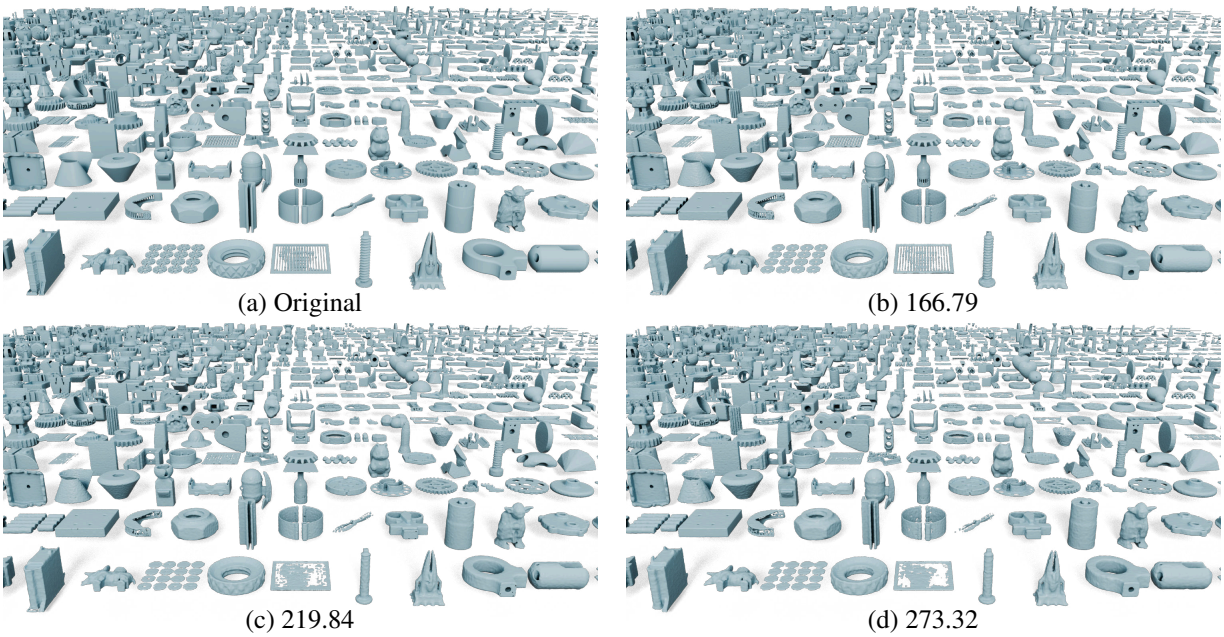
Figure 17. Visual results of the decompressed models from the AMA dataset under different compression ratios.



(a) Original (b) 455.26

(c) 651.85 (d) 899.73

Figure 18. Visual results of the decompressed models from the DT4D dataset under different compression ratios.



(a) Original (b) 166.79

(c) 219.84 (d) 273.32

Figure 19. Visual results of the decompressed models from the Thingi10K dataset under different compression ratios.