Supplementary Material – 3DAC: Learning Attribute Compression for Point Clouds

Guangchi Fang[1,2], Qingyong Hu[3], Hanyun Wang[4], Yiling Xu[5], Yulan Guo[1,2,6*]
[1]Sun Yat-sen University, [2]The Shenzhen Campus of Sun Yat-sen University, [3]University of Oxford
[4]Information Engineering University, [5]Shanghai Jiaotong University
[6]National University of Defense Technology

In this document, we describe details of Region Adaptive Hierarchical Transform (RAHT) and then describe network architecture details of our algorithm. We provide additional experiments on MPEG/JPEG point cloud compression database to validate the effectiveness of our approach, compare our method with learning-based baselines and also benchmark the runtime of our method to show its potential for real-world applications. Moreover, we show additional qualitative results on ScanNet and SemanticKITTI. Finally, we discuss the limitations and broader impact of our approach.

1. Additional RAHT Details

RAHT is a variation of Haar wavelet transform tailored for 3D point clouds. It first voxelizes point clouds and transforms point cloud attributes to low- and high-frequency coefficients along three dimensions repeatedly (*e.g.*, along the x axis first, then the y axis and the z axis) until all points are merged to the entire 3D space. Here, we show the details about transforming two low-frequency coefficients to low and high-frequency coefficients.

Two neighboring points are merged during encoding and low- and high-frequency coefficients are generated from the corresponding low-frequency coefficients with the following transform:

$$\begin{bmatrix} l_{d+1,x,y,z} \\ h_{d+1,x,y,z} \end{bmatrix} = \mathbf{T}_{w_1,w_2} \begin{bmatrix} l_{d,2x,y,z} \\ l_{d,2x+1,y,z} \end{bmatrix}, \qquad (1)$$

where $l_{d,2x,y,z}$ and $l_{d,2x+1,y,z}$ are low-frequency coefficients of two neighboring points along the x dimension, and $l_{d+1,x,y,z}$ and $h_{d+1,x,y,z}$ are the decomposed low-frequency and high-frequency coefficients. For the first depth level, point cloud attributes are regarded as low-frequency coefficients. Here, $\mathbf{T}_{w_1}$ is defined as

$$\mathbf{T}_{w_1,w_2} = \frac{1}{\sqrt{w_1+w_2}} \begin{bmatrix} \sqrt{w_1} & \sqrt{w_2} \\ -\sqrt{w_2} & \sqrt{w_1} \end{bmatrix}, \qquad (2)$$

where w_1 and w_2 are the weights (i.e., the number of leaf nodes) of $l_{d,2x,y,z}$ and $l_{d,2x+1,y,z}$, respectively. Low-frequency coefficients are directly passed to the next level if the point does not have a neighbor.

2. Additional Architecture Details

RAHT tree node context. For a given RAHT tree node, the context information contains the depth level, (*i.e.*, depth of the node in the RAHT tree), the weight w, (*i.e.*, the number of child nodes), the reconstructed low-frequency coefficients l (*i.e.*, the accessible low-frequency coefficients during decoding) and the reconstructed attributes a, (*i.e.*, mean attributes of all points in the corresponding subspace). Note that the reconstructed attributes can be obtained by $a = \frac{l}{\sqrt{w}}$.

Architecture Details. For context feature extraction from high-frequency nodes and inter-channel coefficient correlation, we use a 3-layer MLP (8, 16 and 8 dimensional hidden features) with context of high-frequency nodes and previous encoded coefficients as input, respectively. For context feature extraction from low-frequency nodes and inter-channel spatial correlation, we adopt torchsparse [6] to construct 4 3D sparse convolution layers (3, 3, 6 and 8 dimensional hidden features, and convolution stride as 2) and use trilinear interpretation to obtain latent features from output feature volume of each convolution layer. Besides, latent feature aggregation is realized by a 3-layer MLP (8, 16 and 8 dimensional hidden features) for both initial coding context module and inter-channel correlation module.

Implementation Details. Our network is implemented in PyTorch and trained with one NVIDIA 1080TI GPU. We train our model over 20 epochs using the Adam optimizer with an initial learning rate of 0.01.

3. Additional Experiments

MPEG/JPEG database. We also conduct experiments on MPEG/JPEG point cloud compression datasets, including MVUB [3], Owlii [7] and 8iVFB [2]. All these datasets

[*] Corresponding author: Yulan Guo (guoyulan@sysu.edu.cn).

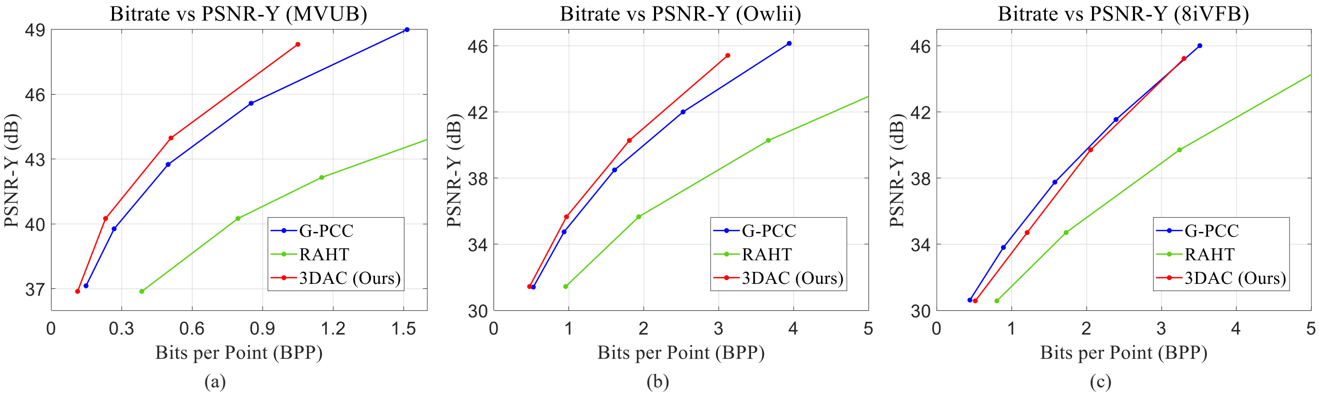


Figure 1. Additional experiments on MPEG/JPEG database. Quantitative results of different attribute compression approaches on the MVUB (a), Owlii (b) and 8iVFB (c) datasets.

contain four to five dynamic human point cloud sequences. We voxelize point clouds of these dataset with a 9-level octree. For MVUB, we use subjects, Andrew, David, Phil and Ricardo for training and Sara for testing. For Owlii, we use basketball player, dancer, exercise for training and model for testing. For 8iVFB, we use longdress, loot, redandblack for training and soldier for testing.

The quantitative attribute compression results on MPEG/JPEG database are shown in Fig. 1. Our 3DAC achieves better compression performance with other baselines on MVUB and Owlii. Due to the huge diversity of attributes in 8iVFB, our method has a small performance gap compared with the standard point cloud compression software, G-PCC, while it is still much better than RAHT. The results on these point cloud compression datasets further illustrate the effectiveness of our method.

Comparison with learning-based methods. In order to demonstrate the superiority of our method, we additionally compare our 3DAC with other complex learning-based baselines. In particular, we additionally include a concurrent work, DeepPCAC [5], on ScanNet. As shown in Figure 2, our 3DAC significantly outperforms all other learning-based baselines.

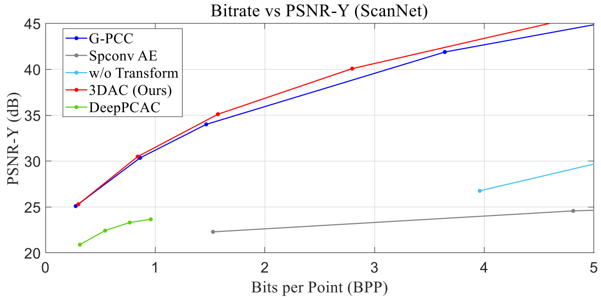


Figure 2. Quantitative results of different learning-based attribute compression approaches on the ScanNet dataset.

Run time. We benchmark runtime of our method on ScanNet with an Intel Core i7-8700 CPU and a Nvidia GeForce GTX 1060 6GB GPU. In our experiments, the arithmetic coder is implemented in C++, and the initial coding method and our entropy model are implemented in python. We set the quantization parameter as 10. The encoding and decoding time of our method are 3.21s and 3.27s, respectively, and those of G-PCC [4], which is implemented in C++, are 0.36s and 0.31s. Although our method is slower than G-PCC, we believe that it is possible to speed up our method with parallel computation and an optimization in data I/O.

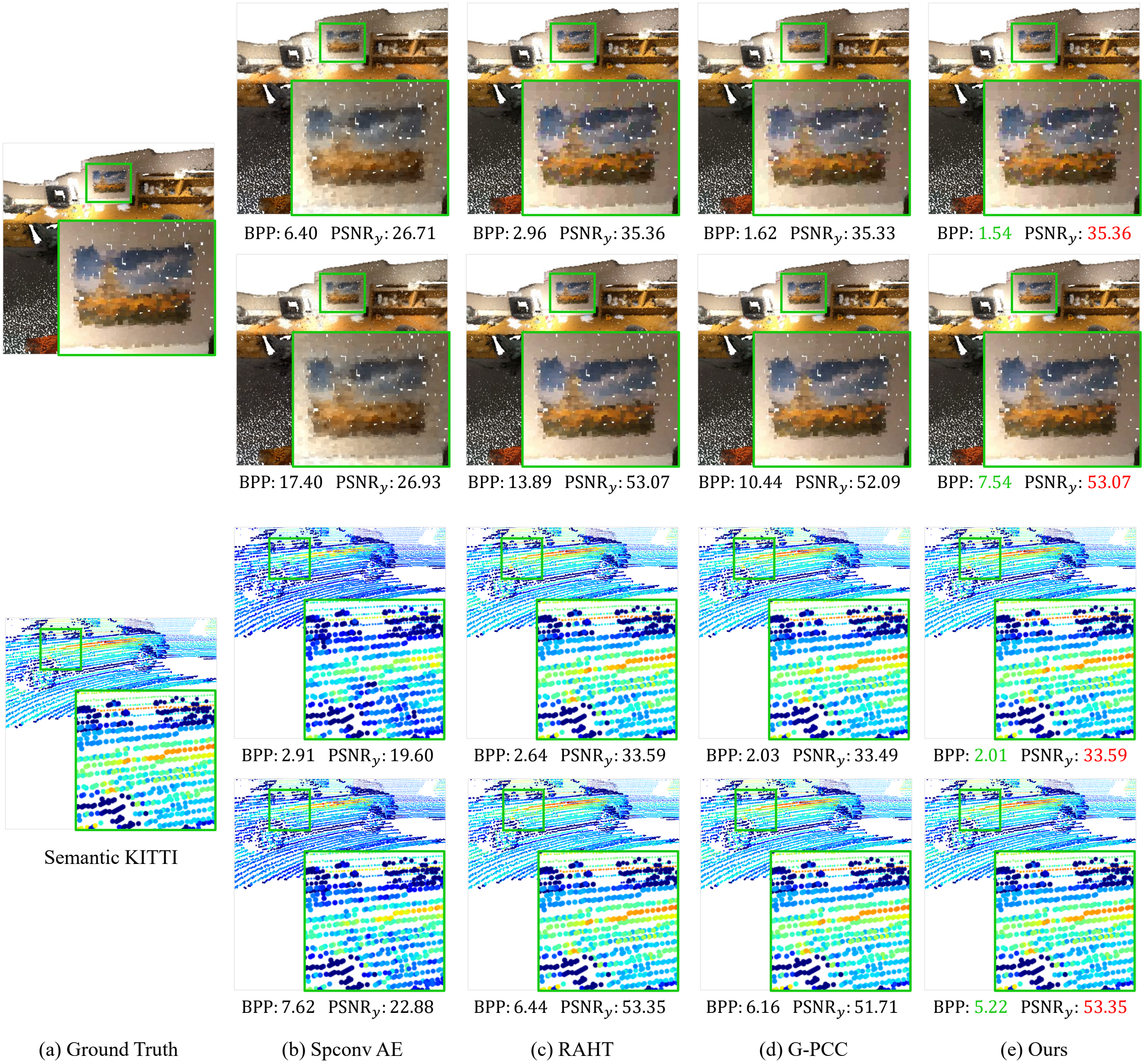
4. Additional Qualitative Results

We show additional qualitative results on ScanNet and SemanticKITTI to show our compression performance in Fig. 3. As shown in the figure, our method can retain better reconstruction quality as well as reducing bitrates.

5. Limitations and Broader Impact

In the current experimental setup, our algorithm only achieves lossy point cloud compression. In specific, we voxelize point clouds with an octree (which leads to geometry distortion), and then adopt RAHT and uniform quantization to process the voxlized point clouds (which leads to attribute distortion). Due to these operations, our framework can not realize lossless compression. A possible solution is to upsample compressed attributes form voxelized points to original points through interpolation, and then transmit the residual of attributes. We leave the attribute interpolation and the residual coding as a future study.

Our point cloud attribute compression algorithm directly helps to 3D data compression, storage and transmission. Thus, we do not foresee any direct negative societal impact of our method. However, the compression and transmission of point cloud data, such as human body and human face, may indirectly lead to invasion of privacy. Thus, we need to be aware of some malicious applications of our method.


BPP: 6.40 PSNR$_y$: 26.71
BPP: 2.96 PSNR$_y$: 35.36
BPP: 1.62 PSNR$_y$: 35.33
BPP: 1.54 PSNR$_y$: 35.36
BPP: 17.40 PSNR$_y$: 26.93
BPP: 13.89 PSNR$_y$: 53.07
BPP: 10.44 PSNR$_y$: 52.09
BPP: 7.54 PSNR$_y$: 53.07
BPP: 2.91 PSNR$_y$: 19.60
BPP: 2.64 PSNR$_y$: 33.59
BPP: 2.03 PSNR$_y$: 33.49
BPP: 2.01 PSNR$_y$: 33.59
Semantic KITTI
BPP: 7.62 PSNR$_y$: 22.88
BPP: 6.44 PSNR$_y$: 53.35
BPP: 6.16 PSNR$_y$: 51.71
BPP: 5.22 PSNR$_y$: 53.35

(a) Ground Truth (b) Spconv AE (c) RAHT (d) G-PCC (e) Ours

Figure 3. Additional qualitative results achieved by our method and other baselines including Spconv AE, RAHT [1] and G-PCC [4]. We visualize ScanNet scans with RGB colors and Semantic KITTI with the intensity of reflectance at relatively low and high bitrates. It is clear that our method can achieve the best compression quality (PSNR sores) with the lowest bitrates.

References

[1] Ricardo L De Queiroz and Philip A Chou. Compression of 3d point clouds using a region-adaptive hierarchical transform. IEEE TIP, 25(8):3947–3956, 2016. 3

[2] Eugene d'Eon, Bob Harrison, Taos Myers, and Philip A Chou. 8i voxelized full bodies-a voxelized point cloud dataset. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006, 7:8, 2017. 1

[3] Charles Loop, Qin Cai, S Orts Escolano, and Philip A Chou. Microsoft voxelized upper bodies-a voxelized point cloud dataset. ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document m38673 M, 72012:2016, 2016. 1

[4] Sebastian Schwarz, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo Cesar, Philip A Chou, Robert A Cohen, Maja Krivokuća, Sébastien Lasserre, Zhu Li, et al. Emerging mpeg standards for point cloud compression. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 9(1):133–148, 2018. 2, 3

[5] Xihua Sheng, Li Li, Dong Liu, Zhiwei Xiong, Zhu Li, and Feng Wu. Deep-pcac: An end-to-end deep lossy compression

framework for point cloud attributes. IEEE TMM, 2021. 2

[6] Haotian Tang, Zhijian Liu, Shengyu Zhao, Yujun Lin, Ji Lin, Hanrui Wang, and Song Han. Searching efficient 3d architectures with sparse point-voxel convolution. In ECCV, pages 685–702. Springer, 2020. 1

[7] Yi Xu, Yao Lu, and Ziyu Wen. Owlii dynamic human mesh sequence dataset. In ISO/IEC JTC1/SC29/WG11 m41658, 120th MPEG Meeting, 2017. 1