Point Cloud Compression with Range Image-based Entropy Model for Autonomous Driving

Sukai Wang[1,4] and Ming Liu[1,2,3]

[1] The Hong Kong University of Science and Technology, Hong Kong SAR, China
[2] The Hong Kong University of Science and Technology (Guangzhou), Nansha, Guangzhou, 511400, Guangdong, China
[3] HKUST Shenzhen-Hong Kong Collaborative Innovation Research Institute, Futian, Shenzhen
[4] Clear Water Bay Institute of Autonomous Driving, Hong Kong SAR, China
{swangcy, eelium}@ust.hk

Abstract. For autonomous driving systems, the storage cost and transmission speed of the large-scale point clouds become an important bottleneck because of their large volume. In this paper, we propose a range image-based three-stage framework to compress the scanning LiDAR's point clouds using the entropy model. In our three-stage framework, we refine the coarser range image by converting the regression problem into the limited classification problem to improve the performance of generating accurate point clouds. And in the feature extraction part, we propose a novel attention Conv layer to fuse the voxel-based 3D features in the 2D range image. Compared with the Octree-based compression methods, the range image compression with the entropy model performs better in the autonomous driving scene. Experiments on LiDARs with different lines and in different scenarios show that our proposed compression scheme outperforms the state-of-the-art approaches in reconstruction quality and downstream tasks by a wide margin.

Keywords: Point Cloud Compression, Entropy Encoding, Autonomous Driving

1 Introduction

Point clouds from scanning LiDARs are to be used in the downstream tasks in the autonomous systems, such as localization[26], detection[42], global mapping[46], etc. In autonomous vehicles, point clouds are required to be transmitted to the server for data recording or backup, or be stored for mapping. The large amount of precision point cloud data from high-frequency scanning LiDAR may cause storage and transmission problems, especially when the network is unstable. Thus, point cloud compression has attracted many people's research attention [3,24,13].

Large-scale outdoor LiDAR point clouds have the characteristics of large-area coverage, unstructured organization, and huge volume in Cartesian space.

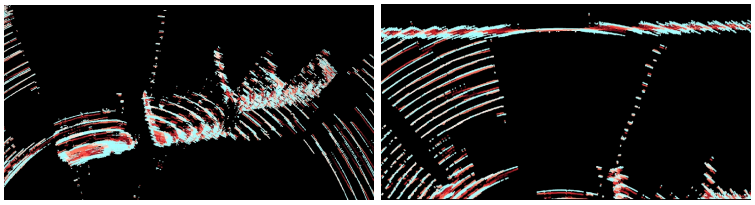

Fig. 1. The comparison of our reconstructed point cloud in red points with the baseline reconstructed point cloud in cyan points. On the left are boxes and to the right is a wall.

Thus, the compression performance is not satisfactory when using the ordinary methods of compressing files to compress the XYZ data. There are two common representations, octree and range image, to make the point cloud more structured. The octree, which divides the three-dimensional space into eight parts recursively, has been widely used to progressively compress point cloud data[27,20,15]. However, octree focuses on structural characteristics and does not eliminate redundancy. It is also inefficient to use the octree method for encoding when the high-precision requirement must be satisfied in the LiDAR point cloud data for autonomous mobile robots. For the range image, the point cloud can be projected into a 2D arrangement, and the shape of the 2D range image is fixed when the point cloud is collected from the scanning LiDAR.

Researchers have focused on using existing image and video encoders to compress the range image[16,36,39,25]. However, these methods are limited in several ways. Traditional image or video encoding algorithms, designed for encoding integer pixel values, will cause significant distortion when encoding floating-point LiDAR data. Additionally, the range image is characterized by sharp edges and homogeneous regions with nearly constant values because of the object geometry. Encoding the range image with traditional techniques, for instance, the block-based discrete cosine transform (DCT)[48] followed by coarse quantization, will result in significant encoding errors at the sharp edges. Moreover, image-based compression methods do not make use of the 3D characteristics of point clouds, while it is inefficient to use existing image-based prediction techniques to remove the redundancy in LiDAR data. Another problem of the traditional compression methods based on the quantization of point clouds is that the reconstructed point cloud will show a wave-like shape from the bird's-eye-view. For example, in Fig. 1, the cyan points are the reconstructed point cloud from the baseline range image-based method[43] using a quantizer. The wavy appearance is extremely obvious in a plane. To solve this issue, we are motivated to refine the quantized point cloud to improve the reconstruction quality.

Wang *et al.* [43] introduced several data encoding algorithms, such as BZip2, LZ4, arithmetic coding, etc. Among all these algorithms, the arithmetic coding is the most popular choice in learning-based compression methods [40,21,24,13], because its differentiable version is open-source and implemented with pytorch[19]. Besides, the probability model in the algorithm can be easily obtained by a neu-

ral network. Tree-based methods [24,13] used voxelized point clouds by octree and predicted the occupancy code of each voxel in the tree. Inspired by that, we propose a three-stage coarse-to-fine framework. By transferring the regression problem to the classification problem, we can use a neural network to predict the probability of the quantized occupancy code for compression.

In this paper, we propose a three-stage framework to compress single-frame large-scale dense point clouds from a mechanical-scanning LiDAR. The first stage projects the point cloud into the 2D range image, and then segments the whole point cloud into ground points and non-ground points. The points are quantized to be the coarse points with a large-error quantization module. The second stage is to refine the coarse points to finer points, by enhancing the accuracy of the non-ground points as a classification problem, and then apply arithmetic coding to encode the probability of each point in the entropy model. The last stage refines the finer point cloud to the accurate points, and makes the reconstructed point cloud more similar to the original point cloud.

To the best of our knowledge, this is the first method that explores the idea of using an end-to-end range image-based entropy network for intra-frame compression. With the 3D feature extracted from sparse voxels and the 2D attention-based fusion module, the reconstructed point cloud can obtain much higher quality within less volume compared with the state-of-the-art methods. The major contributions of the paper are summarized as follows.

- We propose a novel three-stage entropy model-based compression framework, to apply the differentiable arithmetic coding in the range image-based method, by transferring the regression problem to the classification problem.
- We introduce a geometry-aware attention layer to replace the 2D convolutional layer in the 3D-2D feature fusion part, to improve the performance of high-resolution processing in the range image.
- The experiment results show that our compression framework yields better performance than the state-of-the-art methods in terms of compression ratio, reconstruction quality, and the performance of downstream tasks.

2 RELATED WORKS

2.1 Point Cloud Compression Frameworks

According to the representation types of point cloud, compression algorithms can be roughly divided into tree-based and range image-based compression.

Tree-based compression: For tree-based compression, Wang *et al.* [41] introduced the voxel representation for sparse point cloud to utilize the geometry information. From the highest root depth level to the lowest leaf depth level, the size of the voxel gradually decreases. The accuracy of the voxel-based point cloud is determined by the size of the leaf voxel[24]. For the traditional algorithm, the MPEG group developed a geometry-based point cloud compression, G-PCC[28,11,10], as a standard compression library. For the learning-based method, VoxelContext-Net[24] and OctSqueeze [13] are two state-of-the-art octree-based methods, but without open-source code. The proposed networks

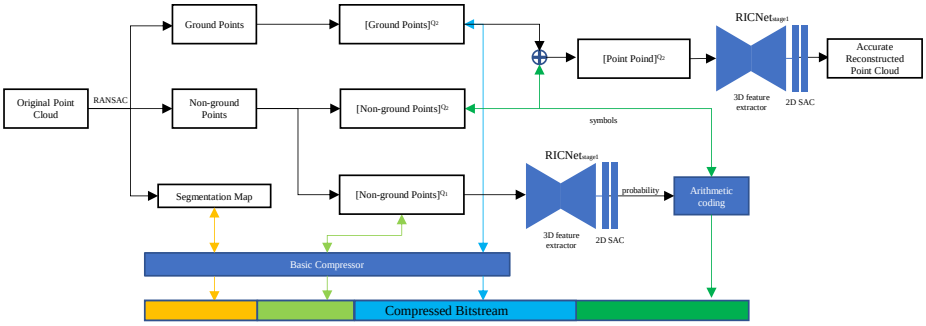

Fig. 2. The overall architecture of our proposed three-stage compression framework. Stage 0 consists of the ground extraction and quantization pre-processing, the stage 1 is an entropy model RICNet$_{stage1}$ for occupancy probability prediction, and the stage 2 is a refinement module RICNet$_{stage2}$ which is used during decompression. The encoding and decoding of the bitstream are totally lossless.

predict the occupancy probability of each voxel in the octree, and apply arithmetic coding to encode the probability with the ground truth symbol.

Range image-based compression: To transform the 3D point cloud into 2D image, Houshiar *et al.* [12] project 3D points onto three panorama images, and Ahn *et al.* [1] projected the point cloud into a range image with the geometry information. Then they compressed the 2D images using an image compression method[39]. Clustering is another common method used in range image compression. Sun *et al.* [32,31] first clustered the point cloud into several segments, and then used traditional compression algorithms, such as BZiP2[29], to encode the residual of the ground truth points with the clustering centers. Wang *et al.*[43] proposed an open-source baseline method for range image-based compression framework, which can choose uniform or non-uniform compression after obtaining the clustering result. However, all of these range image-based compression methods are based on hand-crafted techniques and thus cannot be optimized in an end-to-end network with a large amount of unsupervised point cloud data. Thus, in this paper, we propose an unsupervised end-to-end framework, to encode and refine the point cloud with the entropy model.

2.2 3D and 2D Feature Extractors

PointNet[22] and PointNet++[23] are two widely used point-wise feature extraction backbones for 3D point clouds. PointNet concatenates all points and learns global features, and PointNet++ can extract the local features by grouping the neighbors of each point. The 3D sparse convolution (SPConv)[9] and the MinkowskiEngine[4] are the two latest 3D feature extraction backbones with 3D convolution, pooling, unpooling, and broadcasting operations for sparse voxel tensors. In this work, we choose MinkowskiEngine as our 3D backbones after comparing the performance of different network backbones.

SqueezeSeg[45] and PointSeg [44] use the FireConv and FireDeconv modules to extract the 2D features from a range image and output pixel-wise segmentation results for autonomous driving. Attention networks [37] and graph attention networks [38] are used widely in context-related tasks. In our 2D feature extraction module, a geometry-aware scan-attentive convolutional block is used to fuse the 3D features to smooth the final results.

3 OUR APPROACH

3.1 System Overview

In this paper, we propose a three-stage entropy model-based point cloud compression framework, which is shown in Fig. 2. The first stage is for basic coarse point cloud creation and storage, the second and third stages use the neural network iteratively, RICNet$_{stage1}$ and RICNet$_{stage2}$, for point cloud refinement. The output of RICNet$_{stage1}$ helps to generate the compressed bitstream from arithmetic coding. There are two quantization modules, Q_1 and Q_2, which have different quantization accuracies, q_1 and q_2 respectively, with q_1 larger than q_2. The non-ground points of the stage 0 output have accuracy q_1, and the whole point cloud of the stage 1 output has an accuracy of q_2.

The input of our framework is the range image collected from scanning LiDAR, and each point in the point cloud can be converted from the row and column indexes with the depth of each pixel. If we collect the disordered point cloud at the beginning, we can project the point cloud into the range image, and then use the range image to create an ordered point cloud.

In the stage 0, the segmentation map M, ground points P_g and non-ground points P_{ng} are extracted from the original point cloud by the traditional RANSAC algorithm[43]. Note that a small difference in the segmentation module will only have a very limited impact on the compression rate, which can be ignored. The ground points are quantized with Q_2, $[P_g]^{Q_2} = \lfloor P_g/q_2 \rceil * q_2$, where $\lfloor \rceil$ represents a rounding operation, $[P]^Q$ means the original complete point cloud P is quantized with the quantization module Q, and the non-ground points P_{ng} are quantized with Q_1 and Q_2, and named $[P_{ng}]^{Q_1}$ and $[P_{ng}]^{Q_2}$ respectively. The ground points are easier for compression compared with the non-ground points because they are denser and well-organized. Thus, in the stage 1, RICNet$_{stage1}$ only predicts the probability distribution of the non-ground points $[P_{ng}]^{Q_2}$. The differentiable arithmetic coding[19] takes as input the probability with the occupancy symbol, and outputs the encoded bitstream. In the decompression process, the stage 1 will recover the point cloud with accuracy q_2 losslessly. Stage 2 is trained with the ground truth point cloud and only works in the decompression process. In the stage 2, the point cloud with quantized module Q_2, $[P]^{Q_2}$, is fed into RICNet$_{stage2}$ to get the final accurate reconstructed point cloud.

During compression, the M, $[P_g]^{Q_2}$, and $[P_{ng}]^{Q_1}$ are encoded by a basic compressor, and the distribution occupancy symbols are encoded by the entropy coding. All of these coding and decoding processes are losslessly. Based on the comparative results given in [43], we choose BZip2 as our basic compressor.

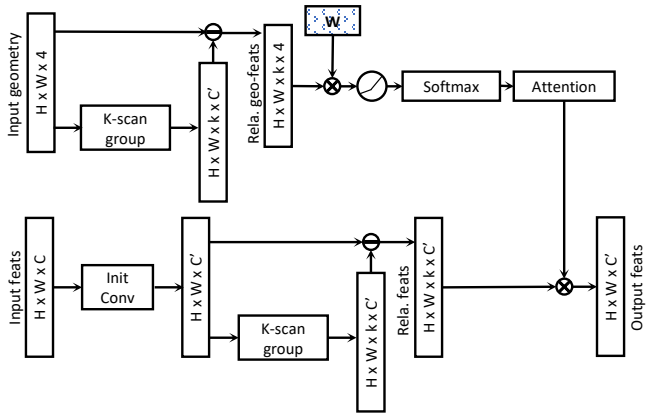

Fig. 3. Our proposed SAC block. The shadow **W** block is the added weights for attention calculation. The K-scan group block groups the neighbors of each point in the same LiDAR scan.

3.2 Network Architecture

RICNet$_{stage1}$ and RICNet$_{stage2}$ are two similar 3D-2D feature fusion networks with the same network architecture, for point cloud refinement. These two networks take as input the coarse points and output the probability of the occupancy code of the refined points. In the stage 1, the output is fed into the arithmetic coding for entropy encoding; in the stage 2, the output can output the final accurate reconstructed point cloud as a refinement module. The 3D feature extractor and 2D attention block in RICNet$_{stage1}$ and RICNet$_{stage2}$ are the same, but the weights are not shared.

3D Feature Extractor: We implement the Minkowski convolutional UNet backbone[4] as our 3D feature extraction module. It is an open-source auto-differentiation library for high-dimensional sparse tensors. The encoder-decoder 3D UNet architecture is similar to the well-known 2D UNet for point-wise prediction, including four convolutional layers and four transposed convolutional layers. The encoder can reduce the spatial dimensions and increase the feature channels, and the skip connection can directly fast-forward the high-resolution features from the encoder to the decoder. The input features are the concatenated point depths with the Cartesian XYZ coordinates, and the output features are the point-wise features.

2D Scan-Attentive Feature Extractor: Owing to the features of different points in the same voxel obtained from the 3D extractor being the same, the 3D features can be seen as the global features. Inspired by GCN[38] and scan-based geometry features in range images used in SLAM [47], we devise the **S**can-**A**ttentive **C**onv (SAC) block to integrate the geometry information of the neighbors of each pixel. Our proposed 2D feature extraction module can fuse the 3D features in the 2D range image, which consists of two SAC blocks after the 3D feature extractor.

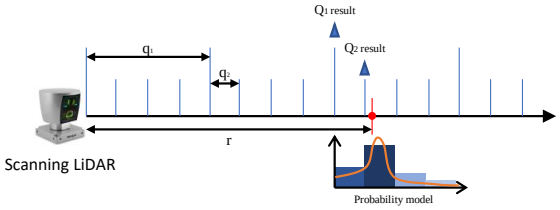

Fig. 4. Toy example of the two quantizers' combination relationship. The red point is the target point, with depth r. The quantized points after Q_1 and Q_2 are labeled. The residual is calculated by quantized depth $-r$. The probability distribution predicted from the RICNet$_{stage1}$ is fed into the arithmetic encoder to encode the ground truth occupancy label ([0, 1, 0, 0] in this example).

The details of the SAC block are shown in Fig. 3. After obtaining the input features, let F_{in} be the input of a SAC block, $F_{in} \in \mathbb{R}^{H \times W \times C}$, where H and W are the height and width of the range image respectively, and C is the number of channels. After the quantization, the points in single scan have obvious geometric characteristics. Thus, for each pixel p_i, we only group $2k$ neighbors (adjacent pixels) in the scan s, $\{q_j, |i-j| \leq k\}$, where $\{s, i\}, \{s, j\}$ are the pixel coordinates and k is the kernel size ($k = 3$ in experiments). Because range images have sharp edges, the values of adjacent pixels may vary greatly. We would like to pay more attention to the features of the adjacent pixels that are not too far away, and ignore the points in different objects. We first obtain the relative geometry features $\Delta G = G_p - G_q$ and relative input features $\Delta F = Conv_{init}(F_p) - Conv_{init}(F_q)$, where $\Delta G = \{(r, x, y, z)\} \in \mathbb{R}^4$, r is the depth of the points, $Conv_{init}$ is an initial convolutional layer with kernel size 3 and out channels C', and $\Delta F \in \mathbb{R}^{C'}$. To calculate the important coefficients between the grouped pixels and the object point, a weight-shared linear transformation with weight matrix $\mathbf{W} \in \mathbb{R}^{4 \times C'}$ is applied to every pixel. The attention coefficient between pixel p_i and its neighbors can be calculated:

$$\alpha = \text{softmax}(\textbf{LeakyReLU}(\mathbf{W}\Delta G)). \tag{1}$$

Then, the relative features of the neighbors ΔF are multiplied with the attention, and the sum of the neighbors' geometry-attentive features is the output of our SAC block.

Occupancy Head and Refinement Head: We propose an occupancy head for RICNet$_{stage1}$ and a refinement head for RICNet$_{stage2}$. The output of the occupancy head is the probability of each occupancy code for entropy encoding, and the output of the refinement head is the residual of the Q_2 quantized point cloud for accurate point cloud reconstruction.

Fig. 4 shows a toy example of a point in red and its ground truth occupancy label in the stage 0 and stage 1. Q_1 result corresponds to the coarse point, and Q_2 result corresponds to the finer point, in Fig. 2. For point $p_{\{i,j\}}$ in the range image with depth r, the residual between the quantized point from the

two quantizers and the ground truth points:

$$res_{\{i,j\}}^{Q_1} = [p]^{Q_1} - r = \lfloor \frac{r}{q_1} \rceil * q_1 - r \in (-\frac{q_1}{2}, \frac{-q_1}{2}], \tag{2}$$

$$res_{\{i,j\}}^{Q_2} = [p]^{Q_2} - r = \lfloor \frac{r}{q_2} \rceil * q_2 - r \in (-\frac{q_2}{2}, \frac{-q_2}{2}], \tag{3}$$

and the length of the occupancy label in the stage 1 for arithmetic coding can be calculated as

$$\text{len}(O) = (\lceil \frac{q_1}{2} / q_2 \rceil) * 2 + 1 \geq \lceil res_{\{i,j\}}^{Q_1} / res_{\{i,j\}}^{Q_2} \rceil. \tag{4}$$

Eq. 4 helps to ensure the occupancy label contains all possible conditions when q_1 cannot be divided by q_2. Thus, in the stage 1, the occupancy head outputs the probability prediction

$$occ_{pred} = \text{softmax}(Conv(F)), \tag{5}$$

where $Conv$ is the 1D convolutional layer with out channels $\text{len}(O)$.

In the stage 2, since the absolute ground truth residual of every point is less than $q_2/2$, the refinement head predicts a sigmoid residual with q_2 gain:

$$res_{pred} = \text{sigmoid}(Conv(F)) * q_2 - q_2/2, \tag{6}$$

where $Conv$ has one out channel as the limited residual. In this way, we can ensure the maximum error of the reconstructed point cloud does not exceed $q_2/2$.

3.3 Network Learning

The training of our network is unsupervised, with the real-world point cloud only. RICNet_{stage1} is an entropy model with the classification output. The loss function in this stage consists of three parts: an l2-regression loss for the residual between the predicted range image and the original range image, a cross entropy loss for classification in the occupancy label, and an entropy loss for end-to-end encoding the bitrate from the differentiable arithmetic coding algorithm. And in RICNet_{stage2}, only the mean square error loss of the residual is counted. Thus, the total loss is

$$\mathcal{L} = \mathcal{L}^{S_1} + \mathcal{L}^{S_2} = \mathcal{L}_{MSE}^{S_1} + \mathcal{L}_{CE} + \mathcal{L}_{BPP} + \mathcal{L}_{MSE}^{S_2}.$$

More specifically, to calculate the predicted point cloud in the stage 1, the classification probability can be converted to the regression residual by accumulating each occupancy location with its probability.

3.4 Compression and Decompression

During compression and decompression, our method can keep the number of points constant, and the only lossy part in our framework is the second quantizer Q_2. The first quantizer is restricted by the ground truth occupancy label, and the maximum error of each pixel in the range image will be less than $q_2/2$. All encoding and decoding processes can be fully lossless in compression and decompression. RICNet$_{stage1}$ takes as input the probability model with the ground truth label of the occupancy label and outputs the compressed bitstream during data encoding. It decodes the ground truth occupancy losslessly using the predicted probability model and the encoded bitstream. Meanwhile, RICNet$_{stage2}$ only works during the decompression as a refinement module.

4 EXPERIMENTS

4.1 Datasets

We evaluate our proposed compression framework on three real-world point cloud datasets, KITTI[7], Oxford[17], and Campus16[43]. The KITTI dataset is collected from a Velodyne-HDL64 LiDAR with 64 scans. The city scene in the KITTI raw-data dataset is evaluated in reconstruction quality experiments, and the KITTI detection dataset is evaluated in the detection downstream task experiments. The Oxford dataset is collected from the Velodyne-HDL32 LiDAR with 32 LiDAR scans. The point clouds collected from the left-hand LiDAR are used for training and testing. Meanwhile, the Campus16 dataset is shared by Wang *et al.*[43], who collected from a Velodyne-VLP16 LiDAR with 16 scans. All three datasets are split into training (2,000 frames), validation (1,000 frames) and testing (1000 frames) sets. The shapes of the range images on three datasets are $[64, 2000]$, $[32, 2250]$, and $[16, 1800]$, respectively. All of the experimental results are evaluated on the testing dataset.

4.2 Evaluation Metrics

To evaluate the degree of compression, we apply bit-per-point (BPP) and compression ratio (CR) as two evaluation metrics. This is because we only consider the geometric compression of the point cloud, and the original points have float32 x, y, and z, CR equals $(32 * 3)/\text{BPP}$.

To evaluate the reconstruction quality, we calculate the Chamfer distance (CD)[14,6], F_1 score, point-to-point PSNR, and point-to-plane peak signal-to-noise ratio (PSNR)[34,18], where the voxel size of the F_1 score is set as 0.02m, and the peak constant values of the two PSNR metrics are set as 59.70m[2]. The definition and the other settings are the same as the corresponding cited papers. The chamfer distance and the PSNR are all symmetric for the original point cloud and the reconstructed point cloud. We then choose the average of these two bi-directional results as the final results.

3D	2D	Attention	Bpp ↓	CD ↓	F1 score ↑	PSNR1 ↑	PSNR2 ↑
✗	SqueezeSeg	✗	2.65	0.0367	0.285	67.48	72.53
PointNet++	✗	✗	2.46	0.0283	0.421	68.41	75.19
Minkowski	✗	✗	2.38	0.0265	0.467	69.74	75.36
Minkowski	2D Conv	✗	2.26	0.027	0.466	69.54	75.28
Minkowski	2D Attentive Conv (SAC)		**2.25**	**0.0255**	**0.508**	**69.86**	**75.39**

Table 1. Compression ratio and reconstruction quality vs. different network architectures. The bold font highlights the best results. PSNR1 is the point-to-point PSNR, and PSNR2 is the point-to-plane PSNR.

BPP	0.2	0.3	0.4	0.5	0.6
w/o Stage 0 & 1	2.27	2.27	2.27	2.27	2.27
w/o Stage 0	2.04	1.97	2.1	2.14	2.18
Ours	2.04	**1.92**	2.08	2.09	2.13

Table 2. BPP ↓ results of different architectures and quantizer combination ratios. The bold font highlights the best results.

To utilize our compression and decompression framework in an autonomous driving system, the performance degradation after using the reconstructed point cloud is also important. In this paper, we evaluate the bounding box average precision (AP) [5,30] for 3D object detection, and the absolute trajectory error (ATE) and the relative pose error (RPE) [49] for simultaneous localization and mapping (SLAM) using the reconstructed point clouds.

4.3 Ablation Study

In this section, we perform ablation studies on compression frameworks and network architectures. The reconstruction quality and compression ratio are evaluated over different 3D and 2D network modules, and only the compression ratio is evaluated over different quantizer combinations.

Three-stage Architecture. In this section, we evaluate the BPP when the first quantizer uses different quantization accuracy q_1, and proves the significance of stage 0 and stage 1 in our proposed three-stage framework. In the experiments, we first guarantee that the accuracy of the second quantizer is always 0.1, and change the accuracy of the first quantizer to 0.2, 0.3, 0.4, 0.5, and 0.6. When the first quantizer changes from finer to coarser, the bitstream length of the non-ground points encoded by the basic compressor changes from long to short, but the bitstream length of the arithmetic coding will grow because the probability results become worse simultaneously. The BPP results in Tab. 2 show that the compression performance of our stage 1 entropy model is best when $q1/q2 = 3$. When we remove stage 0 and stage 1 from our three-stage architecture, RICNet_{stage1} is replaced with the basic compressor BZiP2, and the point cloud will be quantized with $q_2 = 0.1$. In addition, when we remove stage

0 only, the input of RICNet$_{stage1}$ changes to the whole point cloud rather than the non-ground points. The experimental results in Tab. 2 show that our hierarchical range image entropy model is better than the traditional encoding, and only using the non-ground points can remove the interference of ground points, especially in the low-precision quantization situation.

Network Architectures. In this section, we test and compare different well-known 3D and 2D feature extractors for point-wise tasks from the point cloud or pixel-wise tasks from the range image:

- SqueezeSeg [45,44]: We implement the SqueezeSeg twice to replace RICNet, and the heads of the two networks are changed to be the same as the occupancy head and refinement head in RICNet$_{stage1}$ and RICNet$_{stage2}$, respectively.
- PointNet++ [22,23]: Consists of four down-sampling layers (set abstraction modules), and four up-sampling layers (feature propogation modules) with the skip connections.
- Our Minkowski UNet architecture [4]: The Minkowski UNet14 is implemented as the 3D feature extractor. The network architecture performs quickly and well on the point-wise segmentation tasks.
- 2D Conv: We replace the two SAC blocks in the 2D module with two 2D Conv layers with kernel size 3.

Tab. 1 shows the comparative results of different network architectures. The bottom row is the setting in our proposed RICNet. From the first three rows, we can find that the Minkowski encoder-decoder architecture performs best when the network only has a single 2D or 3D feature extractor. The fourth row shows that using the single 2D feature fusion with 2D convolutional layers, the network performs better, with a 0.12 BPP improvement in the stage 1. However, the performance of stage 2 remains unchanged. And with our proposed 2D attentive convolutional layer, the BPP metric shows a further 0.01 improvement, and our refinement model in the stage 2 can predict the point cloud with better reconstruction quality.

4.4 Comparative Results

In this section, we compare our RICNet with the baseline point cloud compression frameworks: Google Draco[8], G-PCC[28,11], JPEG Range[39,25] (using FPEG2000 for range image compression), and R-PCC [43]. In R-PCC implementation, we only evaluate and compare the uniform compression framework for equal comparison.

Reconstruction Quality in Different Datasets. Fig. 5 shows the quantitative results of our proposed method with the baseline methods on the KITTI, Oxford, and Campus datasets. The results show that our RICNet shows a large improvement in the low-BPP compression, which means it can generate a better refined point cloud from the low-precision point cloud from stage 1. At the same time, the bitrate and reconstruction quality of the range image-based methods are better than the tree-based methods, which means that the range image

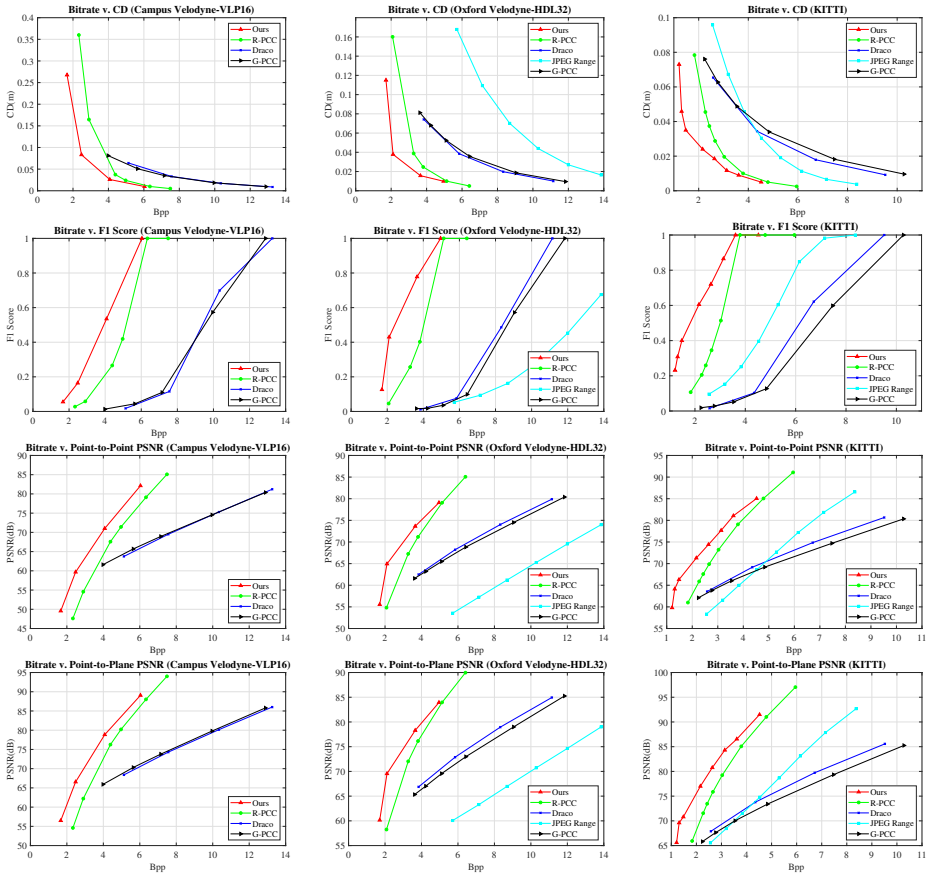

Fig. 5. Quantitative results on KITTI city dataset. Bit-per-point *vs.* symmetric Chamfer distance (↓), F1 score (with $\tau_{geo} = 0.02$m) (↑), and point-to-plane PSNR (with $r = 59.70$) (↑) are shown from top to bottom. The left column is the campus point clouds collected from a Velodyne-VLP16, the middle column is the Oxford dataset collected from a Velodyne-HDL32, and the right column is the KITTI dataset collected from a Velodyne-HDL64.

presentation is efficient enough for compression. In high-accuracy compression situations, our method becomes closer to R-PCC. This is because the error from LiDAR collection and the uneven surfaces of the objects will infer the geometric feature learning when the quantization accuracy of the first quantizer is high. It is also hard to use the irregular learned features to recover the original point cloud.

Downstream Tasks. In this section, we evaluate the performance of the downstream tasks (3D object detection and SLAM) using the reconstructed point cloud. In Fig. 6, the first row shows the evaluation results of the 3D object detection, and the second row shows the comparative SLAM results with

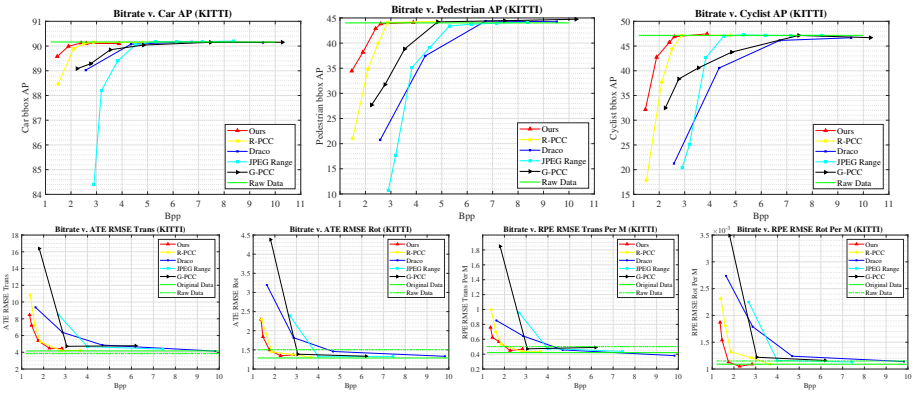

Fig. 6. Quantitative results of the 3D object detection (first row) and SLAM (second row) using reconstructed point cloud. The 3D object detection task is using pre-trained PointPillar from OpenPCDet[33] on the KITTI detection dataset. Car, pedestrian, and cyclist bounding box AP are evaluated from left to right in the first row. The SLAM is using A-LOAM[35] on the KITTI odometry dataset (seq 00).

the baseline methods. Similar to the experimental results of reconstruction quality, the performance in the low-BPP compression situation shows obvious improvements in the downstream tasks too. The lossless threshold of our proposed method is lower than that of the other baseline methods. Since our learning-based method can learn more features from the structured point regions and reconstruct them better, our method has advantages in the downstream tasks for autonomous driving system implementation.

4.5 Qualitative Results

From the quantitative results, we illustrate the advantages of our method in terms of reconstruction quality and downstream tasks. And in Fig. 7, we show the comparative bird's-eye-view image of the predicted point cloud with the ground-truth point cloud and the baseline quantized point cloud in the stage 1. It shows the outstanding prediction and refinement ability of our proposed network, whether using a high-precision quantized point cloud or a low-precision point cloud. And it is also robust for point clouds of different densities, which are collected from different LiDARs. In structured locations and environments especially, such as walls, the predicted points are almost the same as the real points.

5 Conclusion

Our proposed unsupervised end-to-end three-stage compression framework with RICNet outperforms the state-of-the-art methods not only in terms of the reconstruction quality but also in downstream tasks. The experimental results show

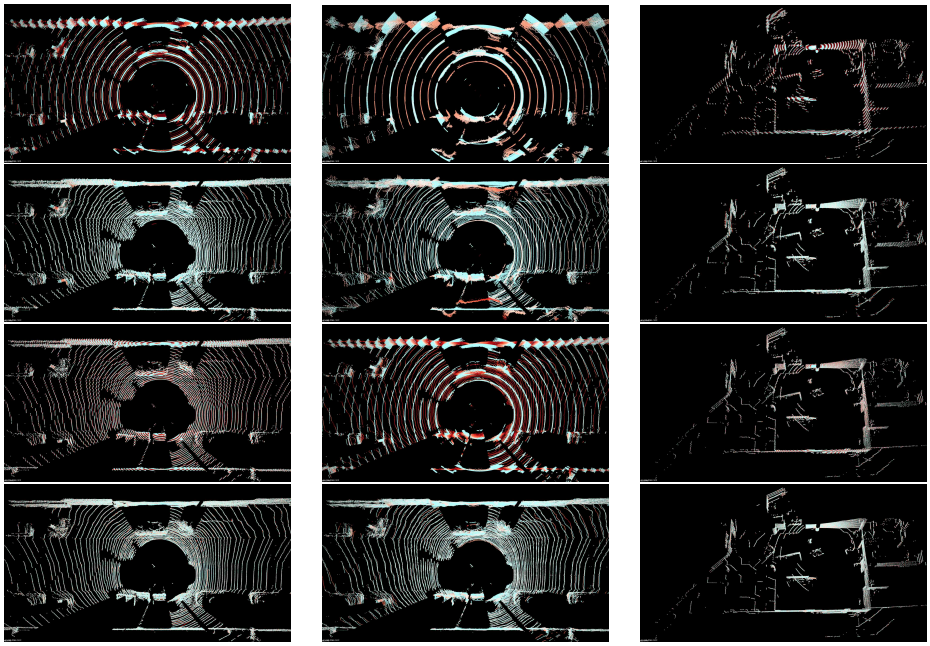

Fig. 7. The qualitative results of our predicted point cloud with the baseline quantized point cloud and ground truth point cloud. From top to bottom, the red points are the predicted point clouds of RICNet$_{stage1}$, RICNet$_{stage1}$, RICNet$_{stage2}$, and RICNet$_{stage2}$, respectively; the cyan points are the baseline quantized point cloud using the first quantizer Q_1, ground truth point cloud, the quantized point cloud using Q_2, and ground truth point cloud, respectively. The left column is the KITTI dataset with the two quantizers $q_1 = 0.3m, q_2 = 0.1m$, while in the middle column $q_1 = 1.5m, q_2 = 0.5m$. The right column is the Campus dataset with $q_1 = 0.3m, q_2 = 0.1m$.

that our compression framework can bring great improvement in low-precision quantization situations, and the network can learn and reconstruct the structured point regions better. The drawback of our framework is that our method can only compose point clouds collected from a scanning LiDAR. The bottleneck of all range image-based point cloud compression frameworks is the error caused by point cloud projection.

Acknowledgement

This work was supported by Zhongshan Science and Technology Bureau Fund, under project 2020AG002, Foshan-HKUST Project no. FSUST20-SHCIRI06C, and the Project of Hetao Shenzhen-Hong Kong Science and Technology Innovation Cooperation Zone(HZQB-KCZYB-2020083), awarded to Prof. Ming Liu. (Corresponding author: Ming Liu)

References

1. Ahn, J.K., Lee, K.Y., Sim, J.Y., Kim, C.S.: Large-scale 3d point cloud compression using adaptive radial distance prediction in hybrid coordinate domains. IEEE Journal of Selected Topics in Signal Processing **9**(3), 422–434 (2014)
2. Biswas, S., Liu, J., Wong, K., Wang, S., Urtasun, R.: Muscle: Multi sweep compression of lidar using deep entropy models. arXiv preprint arXiv:2011.07590 (2020)
3. Cao, C., Preda, M., Zaharia, T.: 3d point cloud compression: A survey. In: The 24th International Conference on 3D Web Technology. pp. 1–9 (2019)
4. Choy, C., Gwak, J., Savarese, S.: 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3075–3084 (2019)
5. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2), 303–338 (2010)
6. Fan, H., Su, H., Guibas, L.J.: A point set generation network for 3d object reconstruction from a single image. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 605–613 (2017)
7. Geiger, A., Lenz, P., Stiller, C., Urtasun, R.: Vision meets robotics: The kitti dataset. The International Journal of Robotics Research **32**(11), 1231–1237 (2013)
8. Google: Draco: 3D Data Compression. https://github.com/google/draco (2018)
9. Graham, B., Engelcke, M., van der Maaten, L.: 3d semantic segmentation with submanifold sparse convolutional networks. CVPR (2018)
10. Graziosi, D., Nakagami, O., Kuma, S., Zaghetto, A., Suzuki, T., Tabatabai, A.: An overview of ongoing point cloud compression standardization activities: Video-based (v-pcc) and geometry-based (g-pcc). APSIPA Transactions on Signal and Information Processing **9** (2020)
11. Group, M.: MPEG G-PCC TMC13. https://github.com/MPEGGroup/mpeg-pcc-tmc13 (2020)
12. Houshiar, H., Nuchter, A.: 3d point cloud compression using conventional image compression for efficient data transmission. In: XXV International Conference on Information (2015)
13. Huang, L., Wang, S., Wong, K., Liu, J., Urtasun, R.: Octsqueeze: Octree-structured entropy model for lidar compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1313–1323 (2020)
14. Huang, T., Liu, Y.: 3d point cloud geometry compression on deep learning. In: Proceedings of the 27th ACM International Conference on Multimedia. pp. 890–898 (2019)
15. Koh, N., Jayaraman, P.K., Zheng, J.: Parallel point cloud compression using truncated octree. In: 2020 International Conference on Cyberworlds (CW). pp. 1–8. IEEE (2020)
16. Korshunov, P., Ebrahimi, T.: Context-dependent jpeg backward-compatible high-dynamic range image compression. Optical Engineering **52**(10), 102006 (2013)
17. Maddern, W., Pascoe, G., Linegar, C., Newman, P.: 1 Year, 1000km: The Oxford RobotCar Dataset. The International Journal of Robotics Research (IJRR) **36**(1), 3–15 (2017). https://doi.org/10.1177/0278364916679498
18. Mekuria, R., Laserre, S., Tulvan, C.: Performance assessment of point cloud compression. In: 2017 IEEE Visual Communications and Image Processing (VCIP). pp. 1–4. IEEE (2017)

19. Mentzer, F., Agustsson, E., Tschannen, M., Timofte, R., Van Gool, L.: Practical full resolution learned lossless image compression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2019)
20. Morell, V., Orts, S., Cazorla, M., Garcia-Rodriguez, J.: Geometric 3d point cloud compression. Pattern Recognition Letters **50**, 55–62 (2014)
21. Nguyen, D.T., Quach, M., Valenzise, G., Duhamel, P.: Multiscale deep context modeling for lossless point cloud geometry compression. In: 2021 IEEE International Conference on Multimedia & Expo Workshops (ICMEW). pp. 1–6. IEEE (2021)
22. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)
23. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. arXiv preprint arXiv:1706.02413 (2017)
24. Que, Z., Lu, G., Xu, D.: Voxelcontext-net: An octree based framework for point cloud compression. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6042–6051 (2021)
25. Rabbani, M.: Jpeg2000: Image compression fundamentals, standards and practice. Journal of Electronic Imaging **11**(2), 286 (2002)
26. Rozenberszki, D., Majdik, A.L.: Lol: Lidar-only odometry and localization in 3d point cloud maps. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 4379–4385. IEEE (2020)
27. Schnabel, R., Klein, R.: Octree-based point-cloud compression. In: PBG@ SIGGRAPH. pp. 111–120 (2006)
28. Schwarz, S., Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P.A., Cohen, R.A., Krivokuća, M., Lasserre, S., Li, Z., 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)
29. Seward, J.: bzip2 and libbzip2. avaliable at http://www. bzip. org (1996)
30. Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al.: Scalability in perception for autonomous driving: Waymo open dataset. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2446–2454 (2020)
31. Sun, X., Wang, S., Liu, M.: A novel coding architecture for multi-line lidar point clouds based on clustering and convolutional lstm network. IEEE Transactions on Intelligent Transportation Systems (2020)
32. Sun, X., Wang, S., Wang, M., Wang, Z., Liu, M.: A novel coding architecture for lidar point cloud sequence. IEEE Robotics and Automation Letters **5**(4), 5637–5644 (2020)
33. Team, O.D.: Openpcdet: An open-source toolbox for 3d object detection from point clouds. `https://github.com/open-mmlab/OpenPCDet` (2020)
34. Tian, D., Ochimizu, H., Feng, C., Cohen, R., Vetro, A.: Geometric distortion metrics for point cloud compression. In: 2017 IEEE International Conference on Image Processing (ICIP). pp. 3460–3464. IEEE (2017)
35. Tong Qin, S.C.: Advanced implementation of loam. https://github.com/HKUST-Aerial-Robotics/A-LOAM (2019)
36. Tu, C., Takeuchi, E., Miyajima, C., Takeda, K.: Compressing continuous point cloud data using image compression methods. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). pp. 1712–1719. IEEE (2016)

37. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in neural information processing systems. pp. 5998–6008 (2017)
38. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y.: Graph attention networks. arXiv preprint arXiv:1710.10903 (2017)
39. Wallace, G.K.: The jpeg still picture compression standard. IEEE transactions on consumer electronics **38**(1), xviii–xxxiv (1992)
40. Wang, J., Ding, D., Li, Z., Ma, Z.: Multiscale point cloud geometry compression. In: 2021 Data Compression Conference (DCC). pp. 73–82. IEEE (2021)
41. Wang, J., Zhu, H., Ma, Z., Chen, T., Liu, H., Shen, Q.: Learned point cloud geometry compression. arXiv preprint arXiv:1909.12037 (2019)
42. Wang, S., Cai, P., Wang, L., Liu, M.: Ditnet: End-to-end 3d object detection and track id assignment in spatio-temporal world. IEEE Robotics and Automation Letters **6**(2), 3397–3404 (2021)
43. Wang, S., Jiao, J., Cai, P., Liu, M.: R-pcc: A baseline for range image-based point cloud compression. arXiv preprint arXiv:2109.07717 (2021)
44. Wang, Y., Shi, T., Yun, P., Tai, L., Liu, M.: Pointseg: Real-time semantic segmentation based on 3d lidar point cloud. arXiv preprint arXiv:1807.06288 (2018)
45. Wu, B., Wan, A., Yue, X., Keutzer, K.: Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 1887–1893. IEEE (2018)
46. Ye, H., Chen, Y., Liu, M.: Tightly coupled 3d lidar inertial odometry and mapping. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 3144–3150. IEEE (2019)
47. Zhang, J., Singh, S.: Loam: Lidar odometry and mapping in real-time. In: Robotics: Science and Systems. vol. 2 (2014)
48. Zhang, X., Wan, W., An, X.: Clustering and dct based color point cloud compression. Journal of Signal Processing Systems **86**(1), 41–49 (2017)
49. Zhang, Z., Scaramuzza, D.: A tutorial on quantitative trajectory evaluation for visual (-inertial) odometry. In: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 7244–7251. IEEE (2018)