A Hierarchical Plane-Quadric Surface Fitting Based Framework for Real-Time LiDAR Point Cloud Compression

Mingyue Cui[1], Jiakang Zhang[1], Mingjian Feng[1], Yuyang Zhong[1], Yanwei Lu[1], Chunjie Shu[1], Yehui Li[2,1(✉)], and Weibing Li[1]

[1] School of Computer Science and Engineering, Sun Yat-Sen University, Guangzhou 510006, China
cuimy5@mail.sysu.edu.cn

[2] Engineering Research Center of Software/Hardware Co-design Technology and Application, Ministry of Education (East China Normal University), China
liyh699@mail.sysu.edu.cn

Abstract. Real-time point cloud compression is critical to time-sensitive applications of robotics such as localization, detection, and segmentation. However, previous methods usually model LiDAR point clouds (LPCs) as voxel structures and apply deep neural networks to classify the voxel occupancy and compress the voxel values, which brings unacceptable time consumption. To address this problem, we propose a hierarchical plane-quadric surface fitting-based framework for real-time LPC compression, which flexibly represents varying geometric complexities. Specifically, we first project the raw LPCs into small and compact 2D range images (RIs) by utilizing the inherent physical properties of LiDAR, and then extract the key frame of sequences. To enable adaptive modeling for different geometric regions of the intra frame, we employ the coefficient of determination as a goodness-of-fit measure, simultaneously guiding the choice between planar approximation and quadratic surface fitting. Besides, we further design a multi-frame temporal fitting strategy to exploit spatiotemporal coherence that iteratively fits the surface of point clouds with a similar range value across consecutive frames, reducing redundant computations and enhancing overall encoding efficiency. Experimental results show that our method outperforms the previous state-of-the-art works on the KITTI benchmark, achieving up to 20.61% bit-rate savings compared with PCL baselines while keeping real-time encoding performance.

Keywords: Point Cloud Compression · Surface Fitting · Range Image · Real-Time · LiDAR

* This work was supported in part by the Guangxi Key R & D Program under Grant GuikeAB24010324, in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2025A1515011485, and in part by the Open Project Program for the Engineering Research Center of Software/Hardware Co-design Technology and Application, Ministry of Education (East China Normal University) under Grant 67000-42990016.

1 Introduction

Light Detection and Ranging (LiDAR) sensors [1, 2] serve as a foundational technology for capturing high-resolution, three-dimensional representations of the surrounding environment, which have been widely used in detection [3], segmentation [4], planning [5], and other fields of intelligent robots [6, 7]. However, the large-volume LPCs present a serious challenge in storage and transmission. Taking the Velodyne HDL-64E LiDAR sensor as an example: it generates over 100,000 points per sweep, resulting in approximately 2.88 million points per second [8]. Therefore, this necessitates high-quality point cloud codecs to achieve efficient LPC compression.

Researchers attempt to exploit spatial redundancy by partitioning the point set into hierarchical structures, such as octrees, and encoding the tree topology along with attributes of points in terminal nodes. The Moving Picture Experts Group (MPEG) proposes a standard point cloud compression method GPCC [9], adapting a hand-crafted context-adaptive arithmetic encoder for bit allocation. Huang *et al.* [10] introduce a tree-structured deep conditional entropy model by leveraging context formation about conditions on ancestor nodes to predict the probabilities of the node occupancies. Cui *et al.* [11] further propose OctFormer, which uses non-overlapped context windows to construct sequences and share the results of the multi-head self-attention operation to reduce time overhead. However, these methods focus more on reconstruction quality, which requires high-resolution voxel to organize the point cloud, bringing unacceptable time consumption.

Achieving high-quality real-time LPC compression is not easy. On the one hand, the characteristic of disorder and sparsity of point clouds makes it difficult to effectively extract spatial features from LPCs. On the other hand, real scenes usually contain various dynamic objects with complex surface geometries, which requires a high level of complexity for the model. Recently, transforming 3D LPCs into 2D range images (RIs) and using surface modeling for real-time compression have become a new paradigm. Feng *et al.* [12] propose a real-time spatio-temporal system (RTST) for LPC compression, which exploits range-image representation by projecting LiDAR scans into spherical views and combines plane-fitting and transformation compensation to encode point clouds. R-PCC [13] segments the original point cloud into compact regions on the RI via FPS-based clustering, and then applies quantization to each region to control per-point reconstruction error. However, these methods still remain a massive quantity of fitting errors, which provides an opportunity for enhancing the reconstruction quality of LPCs.

This paper proposes a novel range image-based framework based on hierarchical plane-quadric surface fitting, which can compress LPCs efficiently. Specifically, the raw LPCs are first sampled via voxel grid filtering to preserve structural features and reduce redundancy. Then, they are transformed into compact 2D RIs in spherical coordinates to facilitate efficient representation, followed by the extraction of key frame (K-frame) from the sequences. For intra-prediction, we fit and encode the K-frame to approximate the surface of discrete point sets by

a hierarchical fitting model combining planar and quadric approximations, and employ the coefficient of determination [14] as a goodness-of-fit metric guiding model selection. For inter-prediction, we design a multi-frame temporal fitting strategy by reusing the parameters to exploit spatiotemporal coherence. Our method combines row-wise parameters sharing for intra-prediction and parameters reuse across frames for inter-prediction, which reduces redundant computations and enhances encoding efficiency. We compare our method with the previous state-of-the-art methods such as PCL [15], Draco [16], RTST [12], and RCPCC [17] on the KITTI large-scale dataset [18]. The experiments show that our method outperforms these methods, which are also only designed for the specific category of point clouds with real-time performance. To summarize, our contributions are highlighted as follows:

- We propose a novel range image-based framework for real-time LiDAR point cloud compression by combining surface modeling, which flexibly represents varying geometric complexities.
- We propose a hierarchical plane–quadric surface fitting method, which enables adaptive surface modeling according to the local geometric complexity of point cloud regions.
- To reduce redundant computations from similar regions, we design a multi-frame temporal fitting strategy that leverages spatio-temporal consistency across consecutive frames, enabling the reuse of fitting parameters.

2 Related Work

2.1 Voxel-based Methods

Voxel-based methods [15] usually convert point clouds into 3D voxel grids and apply transform coding techniques to compress the voxel values. Huang *et al.* [10] propose OctSqueeze, which organizes LPCs into an octree structure and designs a tree-structured conditional entropy model to predict the probabilities of the node occupancies. Que *et al.* [19] further incorporate voxel context into a tree-based deep learning framework, which employs 3D convolution on the generated local voxel context to encode the neighboring spatial information for each node in the constructed octree. Cui *et al.* [20] propose a du-octree based LPC compression model, which develops the cross-attention transformer to fuse prior knowledge features of the traversal format from siblings and capture the hierarchical geometry features between two octrees. However, these methods mainly focus on reconstruction quality, which requires high-resolution voxel to organize the point cloud, which brings a limitation to the ratio of bit-to-reconstruction quality and unacceptable time consumption.

2.2 Point-based Methods

Different from voxel-based methods, point-based methods [21] take the original point cloud as input directly instead of data transformations beforehand.

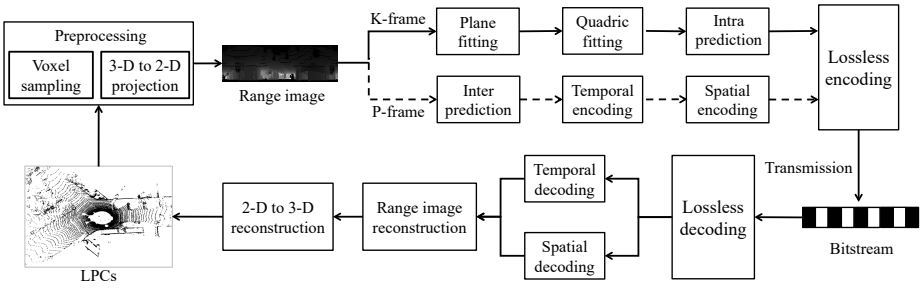

Fig. 1: The framework of our proposed method. The raw LPCs are first converted to RIs after voxel sampling, and we further extract the K-frame of sequences. The K-frame is fitted by the plane or quadric model, and then the fitting parameters are reused for spatial and temporal prediction. Finally, all decompression-required data is serialized and fed into the lossless compression scheme.

Qi *et al.* [22] first propose PointNet, which applies per-point multilayer perceptrons (MLPs) followed by a global max-pooling to extract unified features of 3D data for classification and segmentation. Building on this, Qi *et al.* [23] introduce PointNet++, employing hierarchical grouping and sampling to further capture local geometric structures via farthest point sampling (FPS) and K-nearest neighbor (KNN) search. Gao *et al.* [24] develop the end-to-end analysis/synthesis-based variational autoencoder, which applies multiscale neural graph sampling to characterize neighboring structures as latent features for LPC geometry compression. However, point-based methods often struggle to preserve the fine-grained details of point clouds due to the focus on global feature extraction, which limits the reconstruction precision of LPCs.

2.3 Projection-based Methods

Projection-based methods [25] convert 3D LiDAR point clouds to structured 2D grid representations by mapping points via a single or multi-view projection, and then combine surface models for compression, which gives it more advantages benefiting from its higher coding efficiency. Video-based point cloud compression (V-PCC) [26] developed by the MPEG divides the point cloud into connected regions, which are projected independently into 2-D patches and performs compression with established video codecs. Heo *et al.* [27] proposes a lightweight LPC compression method, which is based on lossy RIs as an intermediate representation and uses a lossless existing dictionary coder to encode. Recently, Cao *et al.* [17] propose a real-time LPC compression and transmission framework RCPCC, which exploits structured range-image projections by fitting surface models with a similar range value, and applies shape adaptive discrete cosine transform to unfit points, with an adaptive bitrate control strategy based on quality of experience.

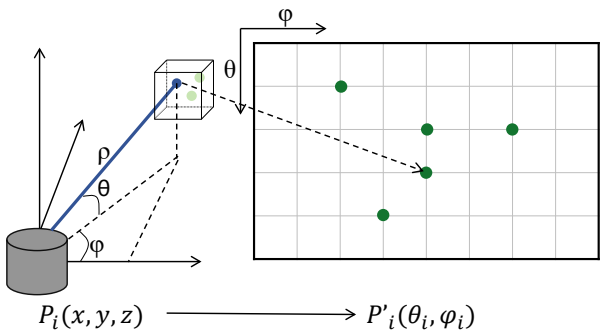

Fig. 2: The transformation of the point in the same voxel from the Cartesian coordinates to the specific voxelized point in the RI.

3 Method

As shown in Figure 1, we propose a real-time LPC compression framework based on hierarchical plane-quadric surface fitting. First, the raw point clouds are transformed into compact 2D RIs through a sequential process of voxel-based sampling and spherical projection. Subsequently, we extract the K-frame from the RI sequence, and the uniform blocks are iteratively fitted and encoded using a hybrid model consisting of the plane and the quadric surface regression function. On one hand, the surface parameters are shared across blocks within the same row of the RI, enabling efficient prediction of unfitted regions for intra-frame prediction. On the other hand, we develop a multi-frame temporal module to exploit spatiotemporal coherence for inter-encoding of predicted frames (P-frames). Specifically, the parameters from the K-frame are reused for the blocks at the same spatial location in P-frames instead of introducing redundant variables, which improves computational efficiency.

3.1 Preprocessing

To reduce redundancy and facilitate efficient representation, the raw point clouds are converted into the RIs through two preprocessing stages, voxel sampling stage and 3D to 2D projection, as shown in Figure 2. In the first stage, the 3D space is discretized into a uniform voxel grid at fixed resolution for spatial occupancy representation. We downsample the point clusters to one point in each valid voxel for spatial redundancy reduction while preserving the global geometric structure. In the second stage, the voxelized 3D point cloud is projected onto a 2D RI plane based on the spherical coordinates of points. Specifically, given a point $P_i(x, y, z)$, its corresponding depth ρ_i, the discretized elevation index θ_i, and azimuth index ψ_i can be computed as:

$$\rho_i = \sqrt{x^2 + y^2 + z^2}, \quad \theta_i = \frac{\arctan\left(\frac{z}{\rho_i}\right)}{\Delta\theta}, \quad \psi_i = \frac{\arccos\left(\frac{x}{y}\right)}{\Delta\psi} \tag{1}$$

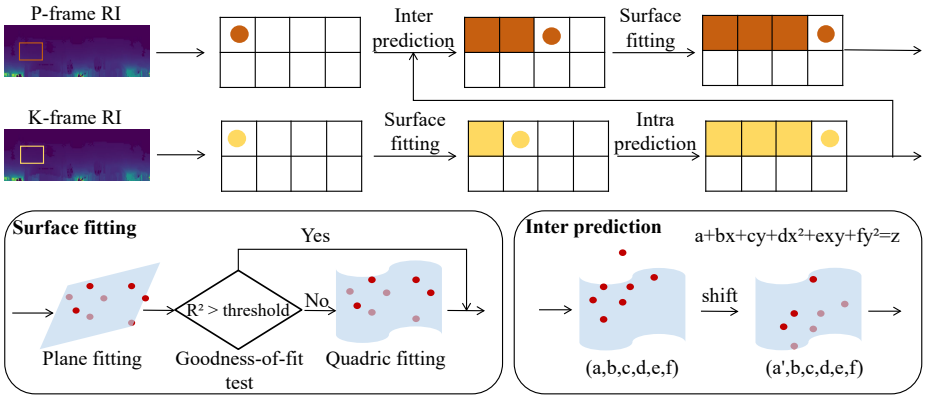

Fig. 3: Architecture of the encoder in the proposed framework. It block-wise fits plane and quadric surfaces on the K-frame to enable intra-prediction, and reuses shifted K-frame parameters for P-frame inter-prediction.

where $\Delta\theta$ and $\Delta\psi$ denote the resolutions of LiDAR's elevation and azimuth angle, respectively. Finally, we obtain the corresponding RI representation $P_i'(\theta_i, \psi_i)$ with its pixel value ρ_i of the point.

3.2 Hierarchical Plane-Quadric Surface Fitting

As illustrated in Figure 3, the encoder in the proposed framework is composed of a hierarchical plane-quadric surface fitting module for intra-frame point approximation and geometry representation, and a temporal prediction module for inter-frame region estimation and encoding. In the plane-quadric surface fitting, each RI is uniformly partitioned into blocks, which are the fundamental units in the subsequent surface fitting process. Given an unfitted block in RI, the point coordinates in this block are represented as (θ_i, ψ_i), while its true pixel value is ρ_i as previously stated. For this block, we first attempt the plane model fitting $\hat{\rho}_i = a + b\theta_i + c\psi_i$ to approximate the points, and use the determination coefficient [14] to evaluate the goodness-of-fit, denoted as R^2_{plane}. If R^2_{plane} exceeds an experimentally predefined threshold τ, the plane fitting is accepted and the parameters are retained. Otherwise, we continue processing with quadric fitting using the second-order model $\hat{\rho}_i = a + b\theta_i + c\psi_i + d\theta_i^2 + e\theta_i \cdot \psi_i + f\psi_i^2$ and detect another determination of coefficient R^2_{quadric} to identify whether the block can be represented by a quadratic function. For the blocks that are unable to be fitted by either of the surfaces, it is directly encoded using its raw pixel value to decrease the offset. This conditional fitting model can be formally summarized as:

$$\hat{\rho}_i = \begin{cases} a + b\theta_i + c\psi_i, & \tau < R^2_{\text{plane}}, \\ a + b\theta_i + c\psi_i + d\theta_i^2 + e\theta_i \cdot \psi_i + f\psi_i^2, & R^2_{\text{plane}} \le \tau < R^2_{\text{quadric}}, \\ \rho_i, & \text{otherwise.} \end{cases} \tag{2}$$

Besides, by adjusting τ, we can balance these complementary strengths between geometric fidelity-limited plane fitting and increased cost quadric fitting.

Since plane fitting can be regarded as a special case of quadric fitting, we adopt a unified representation for both models. Each point (θ_i, ψ_i) in the block with pixel value ρ_i is associated with a feature vector:

$$\mathbf{X}_i = [1, \theta_i, \psi_i, \theta_i^2, \theta_i \cdot \psi_i, \psi_i^2] \tag{3}$$

The fitting task is then formulated as an optimization problem to estimate the parameter vector $\boldsymbol{u}$ by minimizing the squared residuals:

$$\min_{\boldsymbol{u}} \sum_i \left(\mathbf{X}_i \boldsymbol{u} - \rho_i\right)^2 \tag{4}$$

For unified and consistent storage and processing of all fitted models, we define the parameter vector of plane fitting as $\boldsymbol{u} = [a,\ b,\ c,\ 0,\ 0,\ 0]^T$, while quadric fitting uses $\boldsymbol{u} = [a,\ b,\ c,\ d,\ e,\ f]^T$. For plane fitting, we compute the closed-form solution by solving a linear least-squares problem derived from the normal equations. For quadric fitting, we adopt the singular value decomposition (SVD) method [28], which reformulates the quadric surface fitting problem into a linear system. The SVD formulation estimates the surface coefficients by minimizing the algebraic fitting residual, and due to its numerical robustness, it remains stable across a wide range of input conditions.

After fitting a block, we reuse the surface parameters to initiate intra prediction by propagating to horizontally adjacent blocks through a row-wise region-diffused process. When processing the next block, we test whether the propagated surface could encode all the points under the same threshold τ. If the test fails, the current block returns to the surface fitting module, and the entire process is repeated until all blocks in the RI have been processed.

3.3 Multi-frame Temporal Fitting

Considering that consecutive LiDAR frames are sequential temporal samples of the same physical scene, inherent spatiotemporal coherence exists between contiguous sweeps. It may result in redundant computations and inefficient compression when encoding each point cloud individually using the hybrid intra-frame surface fitting model. Therefore, we design a multi-frame temporal fitting strategy that performs inter-frame prediction by reusing the surface fitting parameters from the K-frame, as shown in Figure 3.

Specifically, if a block in the K-frame is fitted with surface parameters $[a, b, c, d, e, f]$, we assume that its corresponding block in the P-frame shares the same shape coefficients $[b, c, d, e, f]$. A new offset a' is then estimated to account for temporal displacement, which can be viewed as vertically shifting the original surface to align with the local geometry in the P-frame. To estimate the offset term a', we compute the mean residual between the observed and predicted range values over all points in the block of the P-frame:

$$a' = \frac{1}{N} \sum_{i=1}^{N} \left(\rho_i - (b\theta_i + c\psi_i + d\theta_i^2 + e\theta_i \cdot \psi_i + f\psi_i^2)\right), \tag{5}$$

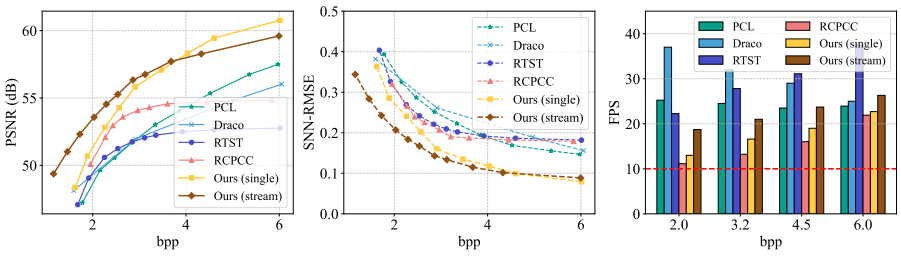

Fig. 4: Quality results and time performance of different compression methods on KITTI dataset at different bitrates, including Ours without multi-frame temporal fitting (single) and Ours with multi-frame temporal fitting (stream).

where N is the number of points in the block, (θ_i, ψ_i) denotes the i^{th} point in the block, ρ_i means its pixel value, and $\rho_i - (b\theta_i + c\psi_i + d\theta_i^2 + e\theta_i \cdot \psi_i + f\psi_i^2)$ represents the offset of the corresponding point with respect to K-frame. The shifted surface is retained and its parameters are stored if they satisfy the goodness-of-fit criterion, after which all associated blocks are marked as encoded and removed from the RI. The remaining blocks in the RI independently undergo single-frame fitting by the hierarchical plane-quadric surface fitting module.

3.4 Parallel Optimization

To further improve time performance, we implement both thread-level and data-level parallelism for RI predicting acceleration. On one hand, considering the inherent independence for intra-frame and inter-frame, we employ thread-level parallelism. Specifically, in intra-frame fitting, individual image rows are processed concurrently by separate threads, which leverages their row-wise independence to reduce latency. In inter-frame fitting, we allocate a thread to each P-frame for exploiting on the calculation independence across different P-frames. On the other hand, we use data-level parallelism within each thread to utilize the point-wise characteristics of key operations. Operations like index lookups and dense matrix solving for surface fitting are vectorized to execute concurrently. By combining these two parallelization strategies, our method efficiently accelerates both the spatial and temporal stages of our LiDAR compression pipeline.

4 Experiment

4.1 Experimental Setup

Dataset Our experiments are conducted on the widely-used KITTI dataset [18] for evaluating the compression ability and quality in the point clouds. The dataset is collected using a Velodyne HDL-64E LiDAR sensor with 64 lasers, including 22 driving sequences, with LiDAR scans captured at 10 Hz and each scan containing approximately 120,000 points.

Table 1: BD-Rate gains over the PCL measure using PSNR for our and other baselines.

Sequence	Draco	RTST	RCPCC	Ours (Single)	Ours (Stream)
Residential	39.15%	-2.72%	-35.54%	-30.33%	-31.75%
Highway	-5.68%	43.04%	-1.50%	-0.18%	-2.47%
Countryside	57.35%	24.98%	-4.21%	-25.64%	-30.30%
City	34.85%	6.46%	1.20%	-16.27%	-17.93%
Avg	31.41%	17.94%	-10.10%	-18.11%	-20.61%

Table 2: BD-Rate gains over the PCL measure using SNN-RMSE for our and other baselines.

Sequence	Draco	RTST	RCPCC	Ours (Single)	Ours (Stream)
Residential	36.61%	-12.70%	-20.45%	-34.59%	-35.95%
Highway	6.00%	25.14%	0.71%	-1.59%	-5.96%
Countryside	33.78%	10.36%	-14.74%	-30.10%	-35.46%
City	37.74%	8.81%	1.20%	-22.08%	-24.56%
Avg	28.53%	15.03%	-14.14%	-22.09%	-25.48%

Baseline Methods We compare our method with the previous state-of-the-art methods, such as the octree-based method point cloud library (PCL) [15], the kdtree-based method Draco [16], and the image-based methods RTST [12] and RCPCC [17]. They are also only designed for a specific category of point clouds with real-time performance.

Evaluation Metrics For evaluation, we use bits per point (bpp) [29] for the compression ratio metric, point-to-point PSNR [30] and the symmetric nearest neighbor root mean square error (SNN-RMSE) [31] to describe the point cloud reconstruction quality. Besides, Bjøntegaard Delta Rate (BD-Rate) [32] quantifies the bitrate-distortion trade-off. Then, frames per second (FPS) [33] is used to describe the computational efficiency. In addition, we also report compression rate (CR) [12], to further evaluate the data volume after compression.

4.2 Results

The rate-distortion curves of LiDAR compression are shown in Figure 4, and we also calculate the BD-Rate gains over PCL using PSNR and SNN-RMSE in Table 1 and Table 2. From the figure, we can observe that as expected, our proposed method consistently surpasses baseline methods across the entire

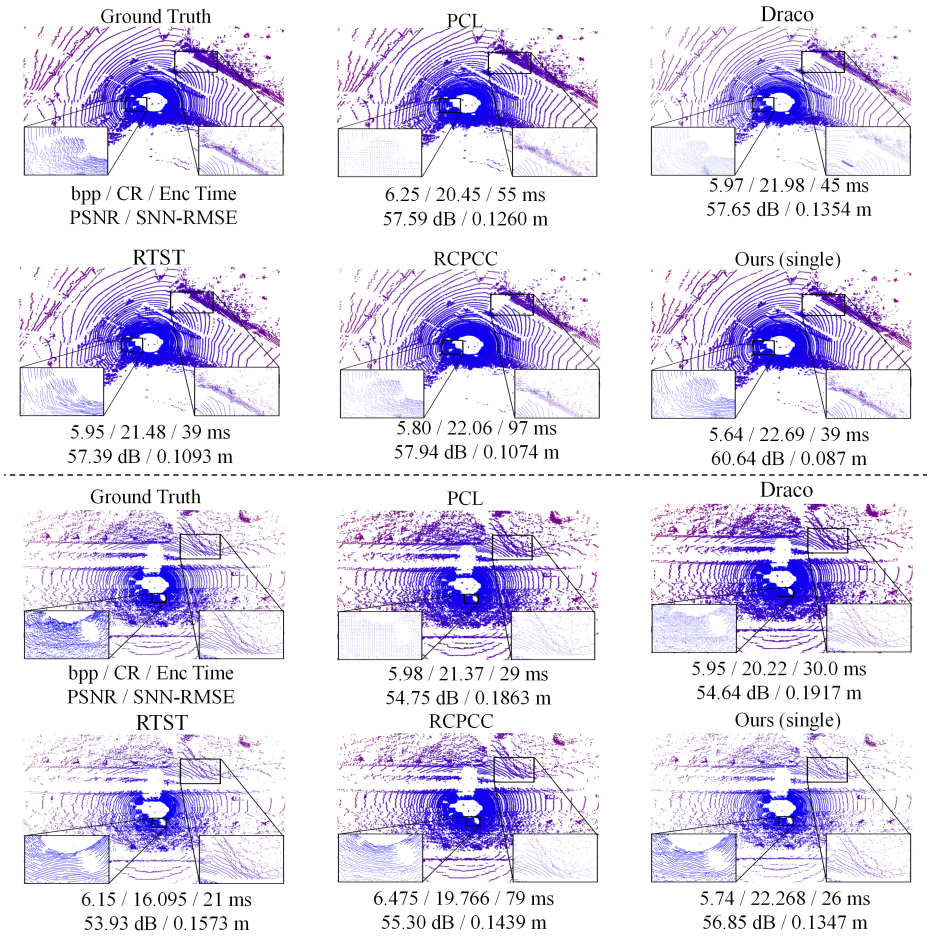

Fig. 5: Visualization of compression results of different methods.

range of displayed bitrates, which means our method achieves better compression performance compared with other baselines. Specifically, our method obtains 20.61% and 25.48% BD-Rate gains in PSNR and SNN-RMSE. This is mainly attributed to our hierarchical plane-quadric surface fitting-based framework and multi-frame temporal fitting strategy, which effectively exploits spatio-temporal coherence. Moreover, even our single-based method also achieves up to 18.11% and 22.09% BD-Rate gains, respectively. Another observation is that our single-based method achieves higher PSNR at high bpp, while stream-based method outperforms in PSNR at low bpp. This is because the stream-based method reduces bits via inter-frame parameters reuse. At low resolution, the small geometric differences between frames make the bit reduction more significant at low resolution, while the same parameters lead to increased geometric distortion at

high resolution. As shown in Figure 4, we also report the time performance of different methods. It can be seen that both our single-based and stream-based methods achieve real-time (10 FPS) compression performance. Compared to RTST, which relies solely on plane fitting, our hierarchical plane–quadric framework better preserves geometric fidelity at the expense of increased computational complexity. We can also see that our stream-based method consistently outperforms its single-based method, as it benefits from the multi-channel coding strategy reducing redundant computations by reusing inter-frame parameters. In general, the experimental results show that our method achieves higher bpp savings compared with other baseline methods while keeping real-time encoding performance, which demonstrates the advantages of our method.

As shown in Figure 5, we visualize compression results of different methods at approximately 6 bpp regimes. For instance, in the bottom frame, our proposed method achieves 5.74 bpp and 22.268 CR, which obtains 56.85 dB PSNR and 0.1347 m SNN-RMSE with only 26 ms encoding time, outperforming other baselines. The thumbnails further reveal distinct geometric characteristics in reconstructed LPCs. We can see that for other methods, PCL shows shape distortion, Draco shows noticeable point dispersion, RCPCC exhibits surface noise in curved regions, and RTST has rigid curves compared with Ground Truth. In contrast, our method maintains cleaner surface continuity, more closely resembling the Ground Truth's structure.

5 Conclusion

In this paper, we propose a novel range image-based framework based on hierarchical plane-quadric surface fitting for real-time LPC compression, which flexibly represents varying geometric complexities. Specifically, our method uses the coefficient of determination as a goodness-of-fit measure guiding the choice between planar approximation and quadratic surface fitting in the intra frame and iteratively fits the surface of point clouds with a similar range value across the inter frame. The experimental results verify the effectiveness of the proposed method, achieving satisfactory rate-distortion performance while maintaining real-time capability.

References

1. Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Q. Yu, and J. Dai, "Bevformer: Learning bird's-eye-view representation from lidar-camera via spatiotemporal transformers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 47, no. 3, pp. 2020–2036, 2025.
2. Y. Cai, F. Kong, Y. Ren, F. Zhu, J. Lin, and F. Zhang, "Occupancy grid mapping without ray-casting for high-resolution lidar sensors," *IEEE Transactions on Robotics*, vol. 40, pp. 172–192, 2024.
3. X. Wang, K. Li, and A. Chehri, "Multi-sensor fusion technology for 3d object detection in autonomous driving: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 25, no. 2, pp. 1148–1165, 2024.

4. A. Xiao, D. Guan, X. Zhang, and S. Lu, "Domain adaptive lidar point cloud segmentation with 3d spatial consistency," *IEEE Transactions on Multimedia*, vol. 26, pp. 5536–5547, 2024.
5. T. Tuna, J. Nubert, Y. Nava, S. Khattak, and M. Hutter, "X-icp: Localizability-aware lidar registration for robust localization in extreme environments," *IEEE Transactions on Robotics*, vol. 40, pp. 452–471, 2024.
6. H. Li, C. Sima, J. Dai, W. Wang, L. Lu, H. Wang, J. Zeng, Z. Li, J. Yang, H. Deng, H. Tian, E. Xie, J. Xie, L. Chen, T. Li, Y. Li, Y. Gao, X. Jia, S. Liu, J. Shi, D. Lin, and Y. Qiao, "Delving into the devils of bird's-eye-view perception: A review, evaluation and recipe," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 46, no. 4, pp. 2151–2170, 2024.
7. J. Wang, F. Li, Y. An, X. Zhang, and H. Sun, "Toward robust lidar-camera fusion in bev space via mutual deformable attention and temporal aggregation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 7, pp. 5753–5764, 2024.
8. X. Wang, Y. Zhang, T. Liu, X. Liu, K. Xu, J. Wan, Y. Guo, and H. Wang, "Topnet: Transformer-efficient occupancy prediction network for octree-structured point cloud geometry compression," in *Proceedings of the Computer Vision and Pattern Recognition Conference (CVPR)*, pp. 27305–27314, June 2025.
9. Mammou, "Mpeg 3d graphics coding. g-pcc test model v27.," in *Output document N18189. ISO/IEC MPEG (JTC 1/SC 29/WG 11)*, 145th MPEG meeting, 2024.
10. L. Huang, S. Wang, K. Wong, J. Liu, and R. Urtasun, "Octsqueeze: Octree-structured entropy model for lidar compression," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1310–1320, 2020.
11. M. Cui, J. Long, M. Feng, B. Li, and H. Kai, "Octformer: Efficient octree-based transformer for point cloud compression with local enhancement," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, pp. 470–478, 2023.
12. Y. Feng, S. Liu, and Y. Zhu, "Real-time spatio-temporal lidar point cloud compression," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10766–10773, 2020.
13. S. Wang, J. Jiao, P. Cai, and M. Liu, "R-pcc: A baseline for range image-based point cloud compression," *2022 International Conference on Robotics and Automation (ICRA)*, pp. 10055–10061, 2021.
14. Q. Liu, L. Fu, Q. Chen, G. Wang, P. Luo, and R. Sharma, "Analysis of the spatial differences in canopy height models from uav lidar and photogrammetry," *Remote Sensing*, vol. 12, 09 2020.
15. J. Kammerl, N. Blodow, R. B. Rusu, S. Gedikli, and E. Steinbach, "Real-time compression of point cloud streams," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2012.
16. J. Brettle and F. Galligan, "Introducing Draco: Compression for 3d graphics." Google Open Source Blog, Jan. 2017. `https://opensource.googleblog.com/2017/01/introducing-draco-compression-for-3d.html`.
17. Y. Cao, Y. Wang, and H. Chen, "Real-time lidar point cloud compression and transmission for resource-constrained robots," in *Proceedings of the 2025 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2025.
18. A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.
19. Z. Que, G. Lu, and D. Xu, "Voxelcontext-net: An octree based framework for point cloud compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6042–6051, 2021.

20. M. Cui, M. Feng, J. Long, D. Hu, S. Zhao, and K. Huang, "A du-octree based cross-attention model for lidar geometry compression," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3796–3802, IEEE, 2024.
21. Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, 2018.
22. R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 77–85, 2017.
23. C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: deep hierarchical feature learning on point sets in a metric space," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 5105–5114, Curran Associates Inc., 2017.
24. L. Gao, T. Fan, J. Wan, Y. Xu, J. Sun, and Z. Ma, "Point cloud geometry compression via neural graph sampling," in *2021 IEEE International Conference on Image Processing (ICIP)*, pp. 3373–3377, 2021.
25. G. Yang, S. Mentasti, M. Bersani, Y. Wang, F. Braghin, and F. Cheli, "Lidar point-cloud processing based on projection methods: a comparison," in *Proceedings of the 2020 AEIT International Conference of Electrical and Electronic Technologies for Automotive*, (Torino, Italy), pp. 1–6, AEIT, 2020.
26. S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuća, S. Lasserre, Z. Li, J. Llach, K. Mammou, R. Mekuria, O. Nakagami, E. Siahaan, A. Tabatabai, A. M. Tourapis, and V. Zakharchenko, "Emerging mpeg standards for point cloud compression," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 133–148, 2019.
27. J. U. Heo, C. Phillips, and A. Gavrilovska, "FLiCR: A fast and lightweight lidar point cloud compression based on lossy RI," in *Proceedings of the IEEE/ACM 7th Symposium on Edge Computing (SEC)*, (Seattle, WA, USA), pp. 54–67, 2022.
28. YongchangWang and L. Zhu, "Research and implementation of svd in machine learning," in *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*, pp. 471–475, 2017.
29. C. Fu, G. Li, R. Song, W. Gao, and S. Liu, "Octattention: Octree-based large-scale contexts model for point cloud compression," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, p. 625–633, June 2022.
30. S. Schwarz, M. Preda, V. Baroncini, M. Budagavi, P. Cesar, P. A. Chou, R. A. Cohen, M. Krivokuća, S. Lasserre, Z. Li, J. Llach, K. Mammou, R. Mekuria, O. Nakagami, E. Siahaan, A. Tabatabai, A. M. Tourapis, and V. Zakharchenko, "Emerging mpeg standards for point cloud compression," *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, vol. 9, no. 1, pp. 133–148, 2019.
31. C. Tu, E. Takeuchi, A. Carballo, and K. Takeda, "Point cloud compression for 3d lidar sensor using recurrent neural network with residual blocks," in *IEEE International Conference on Robotics and Automation (ICRA2019)*, 2019.
32. G. Bjontegaard, "Calculation of average psnr differences between rd-curves," *ITU SG16 Doc. VCEG-M33*, 2001.
33. L. Zhao, K.-K. Ma, X. Lin, W. Wang, and J. Chen, "Real-time lidar point cloud compression using bi-directional prediction and range-adaptive floating-point coding," *IEEE transactions on broadcasting*, vol. 68, no. 3, pp. 620–635, 2022.