Supplementary Material for CRISP: Cylindrical Rendering for In-Stream Point Clouds

Hyungwoo Kang, Seonyoung Jang, Yeojun Yoon, Byung Tae Oh
Korea Aerospace University
Goyang, Republic of Korea
zmeb987@kau.kr, sy001@kau.kr, dbsdu555@kau.kr, byungoh@kau.ac.kr

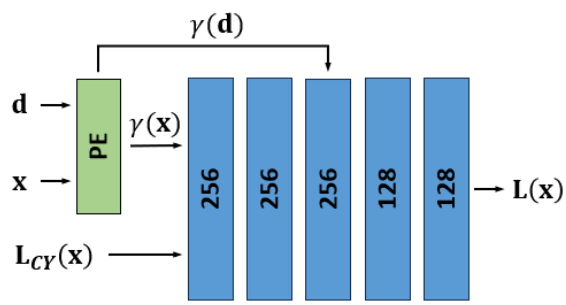

Figure 1. The network structure of MLP F. "FC" means fully connected layer, "PE" means the position encoding.

1. Network Details

This section provides a detailed explanation of the proposed cylindrical feature extraction networks $\Theta = \{\Theta_x, \Theta_y, \Theta_z\}$ and the MLP F [1]. The cylindrical feature extraction networks consist of three independent networks Θ_x, Θ_y, and Θ_z, all of which share the same architecture as illustrated in Figure 2. The MLP F, shown in Figure 1, is used for point-wise feature prediction with position encoding (PE) applied to the input coordinates.

2. Experiment Results

This section presents the evaluation of performance of in-stream point clouds. We conduct experiments on the UVG-VPC dataset [2] under different point densities. The experimental results are summarized as follows:

- Table 3: Results for 700K points/frame.
- Table 4: Results for 300K points/frame.
- Table 5: Results for 100K points/frame.
- Table 6: Results for 50K points/frame.

Table 1. Quantitative results on training conditions for the object Blue with 300K points per frame. The best results are highlighted in **bold**.

Training Condition	**PSNR** (dB)	**SSIM**
Same object	**32.64**	**0.989**
Multi-object, incl. target	27.16	0.964
Multi-object, excl. target	21.94	0.938

Table 2. Quantitative results on projection method for the object Blue with 300K points per frame. The best results are highlighted in **bold**.

Projection Method	**PSNR** (dB)	**SSIM**
Cylindrical projection	**32.64**	**0.989**
Spherical projection	31.98	0.976

2.1. Ablation Study

2.1.1. Training Condition

To analyze the impact of training condition on rendering quality, we compare three training approaches:

- Training on the same object with different motions (Motion Generalization).
- Training on multiple objects with different motions, including the target object.
- Training on multiple objects with different motions, excluding the target object (Scene Generalization).

The results, summarized in Table 1, indicate that when the model is trained with multiple objects, it struggles to accurately capture color information, leading to high SSIM values but a significant drop in PSNR scores. This suggests that while the model maintains structural consistency across different training setups, the ability to preserve accurate colors is compromised when trained with multiple objects. The qualitative comparison in Figure 4 further shows these differences.

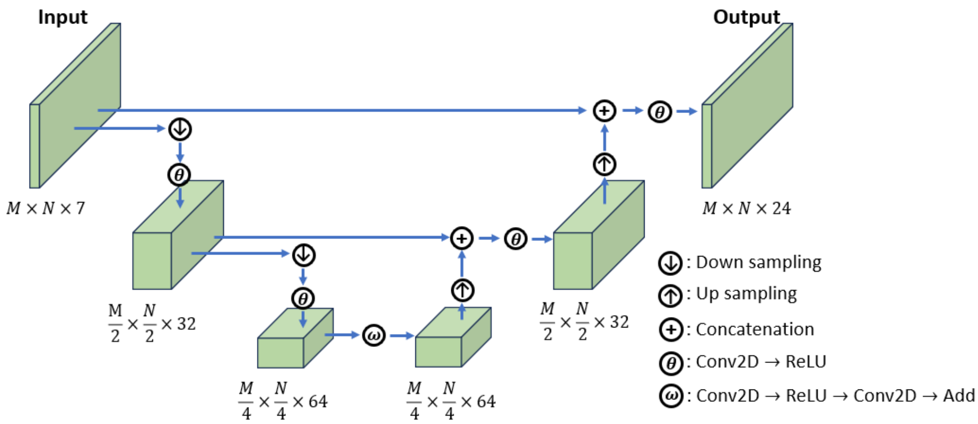

Figure 2. The network structure of cylindrical feature extraction network Θ_x. "Conv2D" means 2D convolutions. The down sampling is completed by 2D max pooling, and the up sampling is completed by bilinear interpolation.

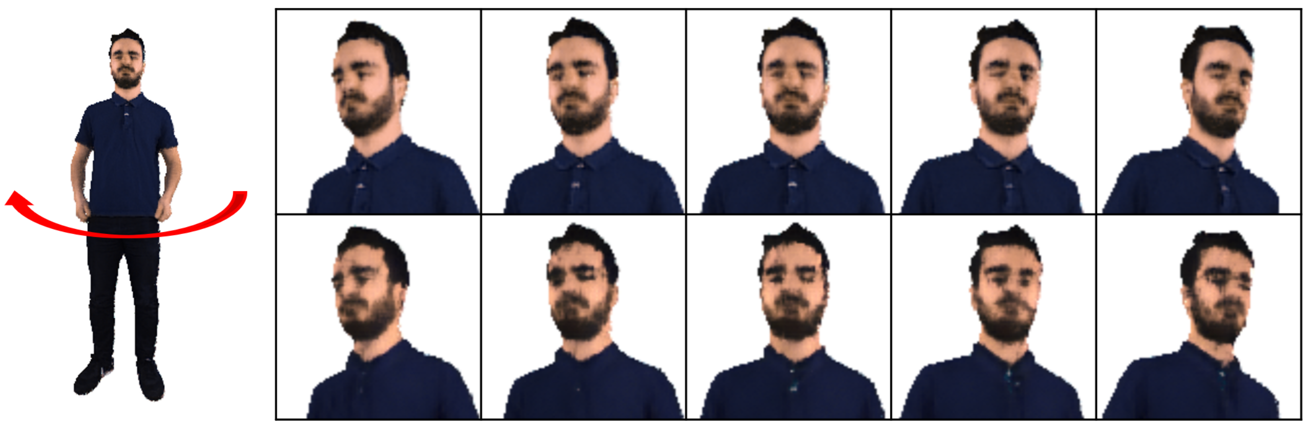

Figure 3. Visual comparison of view consistency. The top row shows results with cylindrical projection, and the bottom row shows results with spherical projection.

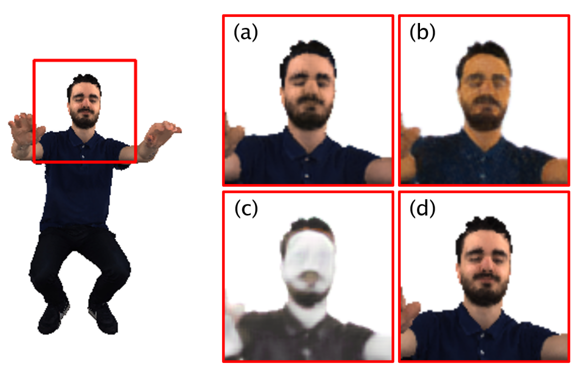

Figure 4. Visual comparison of training conditions for the object Blue with 300K points per frame. (a) Same object, (b) Multi-object, incl. target, (c) Multi-object, excl. target, (d) Ground truth.

In particular, the PSNR decreases sharply when the target object is excluded from the training data, as shown in Table 1, highlighting a loss in precise color reconstruction. This confirms that motion generalization is best achieved when the model is trained on the same object performing different motions, as it ensures both structural and color accuracy in novel motion scenarios.

2.1.2. Projection Method

To evaluate the impact of projection method on rendering quality, we conducted rendering tests at 360°/25° intervals, comparing the following two approaches:

- Cylindrical projection [1]: 3-axis cylindrical UV mapping with 2D CNN + CSW.
- Spherical projection [3]: 3-axis spherical UV mapping with 2D CNN.

Table **??** and Figure 3 show that cylindrical projection with CSW not only provides better view consistency and fewer artifacts than spherical projection but also achieves superior quantitative performance.

References

[1] Joung, S., et al. ”Cylindrical convolutional networks for joint object detection and viewpoint estimation.” In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 1, 2

[2] Gautier, G., et al. ”UVG-VPC: Voxelized point cloud dataset for visual volumetric video-based coding.” In *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*, IEEE, 2023. 1, 4

[3] You, Yang, et al. ”Pointwise rotation-invariant network with adaptive sampling and 3D spherical voxel convolution.” *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020. 2

Table 3. UVG-VPC [2], 700K points/frame (PSNR and SSIM)

UVG-VPC, 700K points/frame					
Method	**Blue**	**Flower**	**Casual**	**Elegant**	**Avg.**
			PSNR ↑		
Pytorch3D	-	-	-	-	-
Poisson	-	-	-	-	-
BPCR	27.3	24.8	24.1	28.4	26.1
3DGS	29.7	26.2	25.8	32.0	28.4
Pointersect	34.2	30.3	29.1	36.5	32.5
TriVol	33.5	29.7	28.5	35.6	31.8
Proposed	**34.8**	**30.5**	**29.6**	**36.7**	**32.6**
			SSIM ↑		
Pytorch3D	-	-	-	-	-
Poisson	-	-	-	-	-
BPCR	0.918	0.910	0.904	0.924	0.914
3DGS	0.926	0.920	0.918	0.928	0.923
Pointersect	0.983	0.979	0.977	0.985	0.981
TriVol	0.978	0.972	0.970	0.982	0.976
Proposed	**0.988**	**0.982**	**0.980**	**0.990**	**0.985**

Table 4. UVG-VPC [2], 300K points/frame (PSNR and SSIM)

UVG-VPC, 300K points/frame					
Method	**Blue**	**Flower**	**Casual**	**Elegant**	**Avg.**
			PSNR ↑		
Pytorch3D	30.5	28.1	27.0	33.3	29.7
Poisson	31.5	29.1	27.9	34.1	30.4
BPCR	26.4	24.3	23.1	27.8	25.1
3DGS	28.7	26.5	25.2	30.1	27.6
Pointersect	31.9	29.7	28.5	33.9	30.7
TriVol	32.4	30.2	29.1	34.4	31.0
Proposed	**32.6**	**32.2**	**28.8**	**35.2**	**32.2**
			SSIM ↑		
Pytorch3D	0.958	0.953	0.948	0.962	0.955
TriVol	0.957	0.952	0.946	0.961	0.954
Pointersect	0.926	0.918	0.912	0.928	0.922
Poisson	0.918	0.912	0.906	0.920	0.914
BPCR	0.968	0.962	0.958	0.970	0.965
3DGS	0.972	0.966	0.962	0.974	0.977
Proposed	**0.989**	**0.980**	**0.979**	**0.990**	**0.984**

Table 5. UVG-VPC [2], 100K points/frame (PSNR and SSIM)

UVG-VPC, 100K points/frame					
Method	**Blue**	**Flower**	**Casual**	**Elegant**	**Avg.**
			PSNR ↑		
Pytorch3D	28.5	26.1	25.0	30.0	27.4
Poisson	29.1	26.8	25.7	30.8	27.9
BPCR	23.8	21.0	20.2	23.5	22.1
3DGS	25.5	22.9	21.8	25.8	23.8
Pointersect	30.8	28.5	27.2	31.9	29.5
TriVol	31.6	29.3	28.0	32.7	30.4
Proposed	**33.2**	**30.5**	**29.3**	**34.9**	**31.9**
			SSIM ↑		
Pytorch3D	0.937	0.932	0.928	0.940	0.934
Poisson	0.949	0.944	0.940	0.951	0.946
BPCR	0.926	0.918	0.915	0.930	0.922
3DGS	0.932	0.924	0.920	0.936	0.928
Pointersect	0.952	0.946	0.943	0.955	0.949
TriVol	0.968	0.962	0.958	0.970	0.964
Proposed	**0.976**	**0.970**	**0.968**	**0.974**	**0.972**

Table 6. UVG-VPC [2], 50K points/frame (PSNR and SSIM)

UVG-VPC, 50K points/frame					
			PSNR ↑		
Method	**Blue**	**Flower**	**Casual**	**Elegant**	**Avg.**
Pytorch3D	27.5	25.3	24.0	28.3	26.2
Poisson	27.8	25.6	24.4	28.9	26.4
BPCR	23.1	20.9	19.8	22.9	21.4
3DGS	24.5	22.3	21.0	24.5	22.5
Pointersect	28.5	26.5	25.3	29.8	27.2
TriVol	31.3	29.0	27.7	32.6	30.1
Proposed	**33.0**	**30.2**	**29.1**	**34.1**	**31.1**
			SSIM ↑		
Pytorch3D	0.926	0.920	0.918	0.924	0.922
Poisson	0.936	0.930	0.928	0.938	0.932
BPCR	0.916	0.910	0.906	0.916	0.912
3DGS	0.920	0.914	0.910	0.920	0.916
Pointersect	0.940	0.934	0.930	0.942	0.937
TroVol	0.962	0.956	0.952	0.964	0.959
Proposed	**0.966**	**0.960**	**0.958**	**0.964**	**0.962**