DYNAMIC POINT CLOUD INTERPOLATION

Anique Akhtar[⋆] *Zhu Li*[⋆] *Geert Van der Auwera*[†] *Jianle Chen*[†]

[⋆]University of Missouri-Kansas City [†]Qualcomm Technologies Inc.

ABSTRACT

Dense photorealistic point clouds can depict real-world dynamic objects in high resolution and with a high frame rate. Frame interpolation of such dynamic point clouds would enable the distribution, processing, and compression of such content. In this work, we propose a first point cloud interpolation framework for photorealistic dynamic point clouds. Given two consecutive dynamic point cloud frames, our framework aims to generate intermediate frame(s) between them. The proposed deep learning framework has three major components: the encoder module, the fusion network, and the multi-scale point cloud synthesis module. The encoder module extracts multi-scale features from two consecutive frames. The fusion network employs a novel 4D feature learning technique to merge the multi-scale features from consecutive frames. Finally, the multi-scale point cloud synthesis module hierarchically reconstructs the interpolated point cloud intermediate frame at different resolutions. We evaluate our framework on high-resolution point cloud datasets used in MPEG, JPEG Pleno, and AVS standards. The quantitative and qualitative results demonstrate the effectiveness of the proposed method.

Index Terms— Dynamic Point Cloud, Interpolation

1. INTRODUCTION

Point clouds can be categorized as point cloud scenes and point cloud objects. Point cloud scenes are typically dynamically acquired using LiDAR (light detection and ranging) sensors and are commonly used in autonomous vehicles [1]. Point cloud objects can be further subdivided into static objects and dynamic objects. A static object is a single object, whereas a dynamic object is time-varying, where each instance of a dynamic point cloud is a static point cloud. Dynamic time-varying point clouds are used in AR/VR, volumetric video, and telepresence and can be generated using 3D models, i.e., CGI, or captured from real-world scenes using various methods such as multiple cameras with depth sensors surrounding the object and capturing movement over time. Frame rates of LiDARs are generally 10 to 20 Hz, resulting in a lower resolution, spatially and temporally sparse point cloud [2]. However, dynamic point clouds are denser photorealistic point clouds that contain a lot more points with high data rates. For example, a single instance of dynamic point cloud captured by 8i [3] contains between 1 million to 4 million points per frame which translates to a bitrate of around 1 Gbytes per second without compression for a 30 fps dynamic point cloud. The high data rate is one of the main problems faced by dynamic point clouds, and efficient interpolation techniques to synthesize intermediate frames would help in the distribution, processing, and compression of such content [4].

Video frame interpolation is commonly utilized in frame rate conversion, novel view synthesis, video streaming, and video compression pipeline to generate high frame rate videos from low frame rate ones (e.g. from 30 Hz to 240 Hz). Typical video frame interpolation methods [5, 6, 7, 8] perform two tasks: motion estimation, usually optical flow, and pixel synthesis. However, these methods cannot be directly applied to point clouds since 3D point clouds are unstructured and unordered. There are no direct correspondences between points in two point clouds like pixels in two images. The sparsity and large size of point clouds further complicate the point cloud interpolation problem. There has been limited work on 3D point cloud interpolation and all work has been performed on dynamically acquired LiDAR-based point cloud scenes. While optical flow represents 2D pixel movements on the image plane, 3D scene flow represents per point 3D movement. Optical flow can be considered the projection of scene flow into 2D. FlowNet3D [9] is a pioneering work of deep learning-based 3D scene flow estimation. FlowNet3D proposed a flow embedding layer to model the motion of points in different point cloud scenes. Following FlowNet3D, FlowNet3D++ [10] proposed geometric constraints in the form of point-to-plane distance and angular alignment to further improve the accuracy of scene flow estimation. There have been further works [11, 12, 13] that explore point cloud scene flow estimation or interpolation. PointINet [14] estimates bi-directional 3D scene flows, performs frame warping, followed by point fusions and intermediate point cloud generation. Even though these methods make a decent baseline in point cloud scene flow estimation, they are only limited to LiDAR-based point cloud scenes and are not applicable to photo-realistic dynamic point clouds.

Compared to dense dynamic point clouds, LiDAR point

[1]This work is supported in part through NSF-1747751 grant.

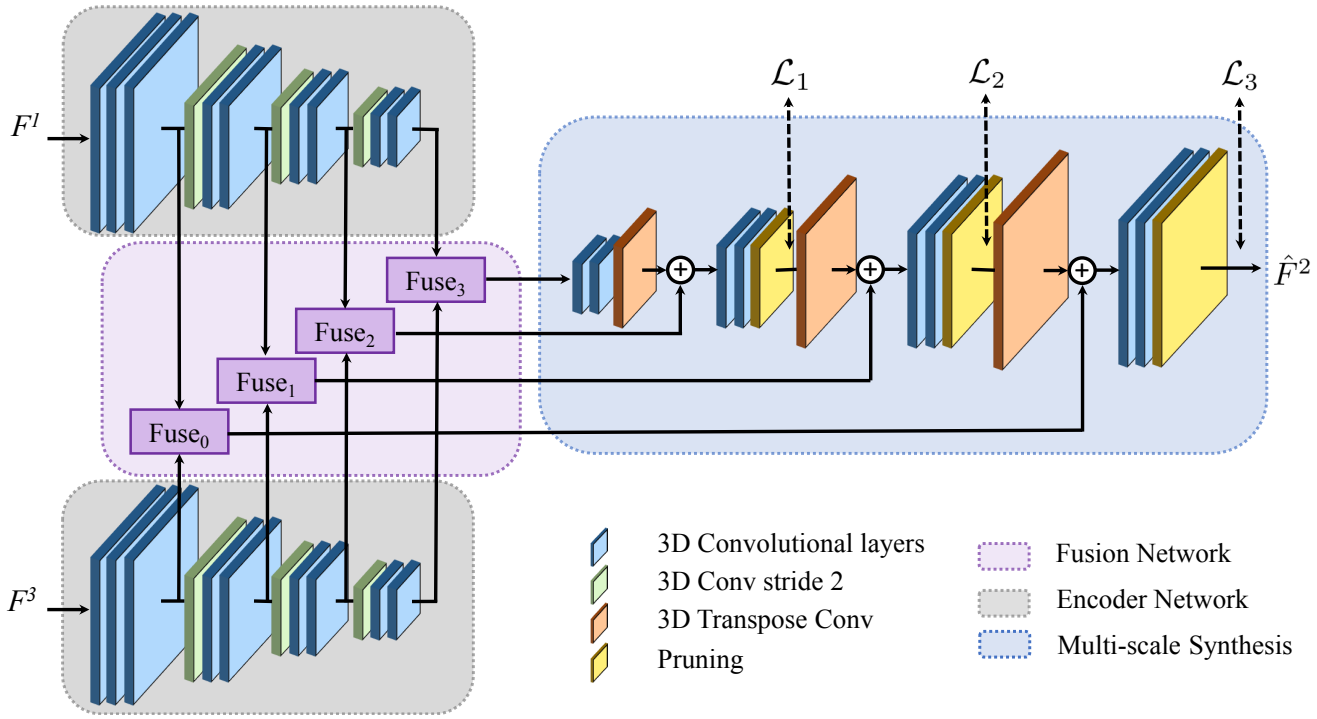

Fig. 1. System Model. F^1 and F^3 are the input frames while $\hat{F}^2$ is the interpolated frame.

clouds tend to be sparser, with a lower frame rate and a smaller number of points. Since point cloud scenes are dynamically acquired, there is more scene rigidity as well as point correspondences. However, photo-realistic dynamic point clouds tend to be denser where the object moves and changes shape making the point correspondences difficult and thus the scene flow methods are inapplicable to dynamic point clouds. Moreover, the large size of the photo-realistic dynamic point clouds makes the task further challenging [15]. To address these issues we propose a first of its kind dynamic point cloud interpolation framework. Given two consecutive dynamic point cloud frames, our framework aims to generate intermediate frame(s) between them. We propose three different modules: the encoder network, the fusion network, and the multi-scale point cloud synthesis module. The encoder module extracts features from frames at four different scales. The fusion network takes features at different scales from consecutive frames, concatenates them into 4D features, then utilizes 4D convolutions to merge consecutive frame features. Finally, the multi-scale point cloud synthesis module hierarchically interpolates the target frame at different resolutions.

2. POINT CLOUD INTERPOLATION

In this section, we first introduce the overall architecture of the proposed point cloud interpolation network and then explain the details of the key components of our framework.

The overall system model is shown in Fig. 1. Given two point cloud frames $F^1 \in \mathbb{R}^{N\times 3}$ and $F^3 \in \mathbb{R}^{N\times 3}$, the goal is to predict the intermediate point cloud frame $\hat{F}^2$.

2.1. Network

The proposed framework contains three different modules: the encoder network, the fusion network, and the multi-scale point cloud synthesis module. We use sparse tensors and sparse convolution using Minkowski Engine [16]. The input frames are pre-processed where they are voxelized and converted into sparse tensors.

2.1.1. The Encoder Module

We employ the encoder module for multi-scale point cloud feature extraction. This module learns the point cloud features at four different scales for both frames F^1 and F^3. Both encoder modules shown in Fig. 1 are identical and share the same weights. As shown in the evaluation results, a pre-trained encoder module performs much better than learning the weights of the encoder along with the rest of the network. We pre-train our encoder module in a typical encoder-decoder architecture using reconstruction loss (binary voxel occupancy loss) on a static point cloud objects dataset (ShapeNet).

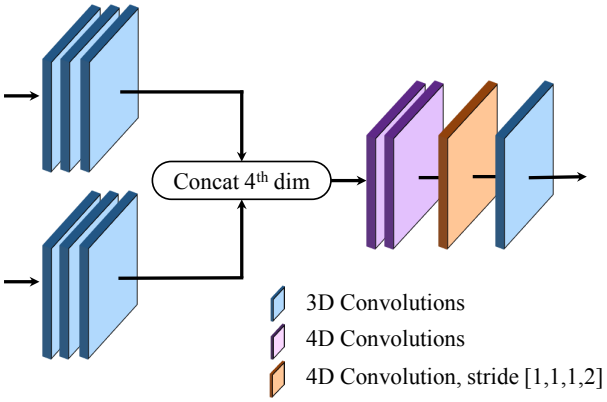

Fig. 2. A single fuse block from the fusion network.

2.1.2. *The Fusion network*

The fusion network utilizes a novel 4D fuse block that merges features from two consecutive point cloud frames into a single feature. The fusion module takes features at four different scales from frames F^1 and F^3. Different scale features are processed individually by a fuse block as shown in Fig. 1. The goal here is to merge features at the same scale together. The individual fuse block is shown in Fig. 2. In the fuse block, the features pass through 3D convolutions and then are concatenated in the 4^{th} dimension resulting in the feature size of $(x, y, z, 2)$. Where x, y, z is the size of the *x, y* and z coordinates respectively. The 4D features are then passed through 4D convolutions so the inter-frame features could be learned. Afterward, a 4D convolution with stride in only the 4^{th} dimension $(stride = [1, 1, 1, 2])$ is applied. This convolution acts as a learnable pooling in a single dimension where the resulting feature size becomes $(x, y, z, 1)$. This is converted back into 3D features so the 3D convolutions can be applied. Since 4D convolutions tend to have higher computational complexity and extra memory consumption, we tend to limit the amount of 4D convolutions we employ.

2.1.3. *Multi-scale Point Cloud Synthesis Module*

The fused features at four different scales are fed into the multi-scale point cloud synthesis module that hierarchically interpolates the intermediate frame $\hat{F}^2$. Sparse transpose convolutions are used to upscale the features. After each upscaling, the features from the fuse block of the same scale are added to the synthesis module network. Upscaling using transpose convolution generates a lot of new points. To choose the best points and dispose of invalid points, pruning is performed. The pruning layer implements classification by converting $N \times C$ features to $N \times 1$ features and choosing the best features above a threshold.

2.2. Loss Function

Rather than upscaling the features directly to the full scale, our network synthesizes the interpolated point cloud at different resolutions by employing multiple loss functions. The multi-scale synthesis module produces the interpolated point cloud at three different resolutions. As shown in Fig. 1, we train our network using three different loss functions:

$$\mathcal{L} = \mathcal{L}_1 + \mathcal{L}_2 + \mathcal{L}_3 \ , \tag{1}$$

We perform voxel classification using binary cross-entropy loss to compare the voxel occupancy prediction from the network and the ground truth (original) point cloud. The ground truth point cloud is downscaled to three resolutions, one for each loss function.

3. EVALUATION RESULTS

In this section, we will go over our datasets, the training environment, evaluation metrics employed, and both the quantitative and visual results of our dynamic point cloud interpolation framework.

3.1. Datasets

The encoder network is pretrained before being utilized in the framework. The encoder is pretrained on **ShapeNet** dataset which is a static objects dataset. We randomly selected ≈ 24000 3D mesh models from the core dataset of ShapeNet. The mesh model is sampled into point clouds by randomly generating points on the surfaces of the mesh, then randomly rotated and quantized the point cloud to 7-bit precision. For dynamic point clouds we employed sequences *loot, longdress* and *redandblack* from JPEG Plenos 8i Voxelized Full Bodies dataset (**8iVFB v2**) [17] with about a million points per frame. We also used sequences *basketball* and *exercise* from MPEGs 8i Voxelized Surface Light Field dataset (**8iVSLF**) [3] with about 2.6 million points per frame. Finally we used the sequence *Queen* produced by **Technicolor** (https://www.technicolor.com/fr) with about 1 million points per frame. We train on *loot* and *longdress* sequences and test on the rest of the four sequences.

3.2. Training

The encoder architecture is pretrained on ShapeNet dataset. We train the rest of the framework on *loot* and *longdress* sequences. To be able to feed our network three frames each containing about a million points per frame, we subdivide the point cloud using kd-tree partitioning. Points are extracted from the same locations within the cube across multiple frames. In our training, we use a kd-tree depth of 4 to divide each frame into 16 cubes. During evaluation, the whole point cloud can be fed into the network.

Method	*redandblack*		*queen*		*basketball*		*exercise*		Avg	
	CD↓	PSNR↑	CD↓	PSNR↑	CD↓	PSNR↑	CD↓	PSNR↑	CD↓	PSNR↑
Identity	1623.69	53.68	45.29	70.75	113.65	71.73	146.54	71.19	482.29	66.84
Ours	**386.22**	**56.88**	**30.44**	**76.08**	**80.96**	**74.54**	**110.20**	**75.48**	**151.96**	**70.75**
Ours-w/o Pretrained	502.86	55.44	35.64	74.81	90.37	73.30	117.62	74.34	186.63	69.47
Ours-Single Loss	575.91	55.40	36.64	74.00	89.74	73.03	121.54	73.37	205.96	68.95
Ours-Fuse3D	865.72	54.64	37.01	73.64	100.18	72.14	125.47	73.12	282.10	68.39

Table 1. Evaluation results of our interpolation method using Chamfer Distance (CD (10^{-2})) and MSE PSNR (dB).

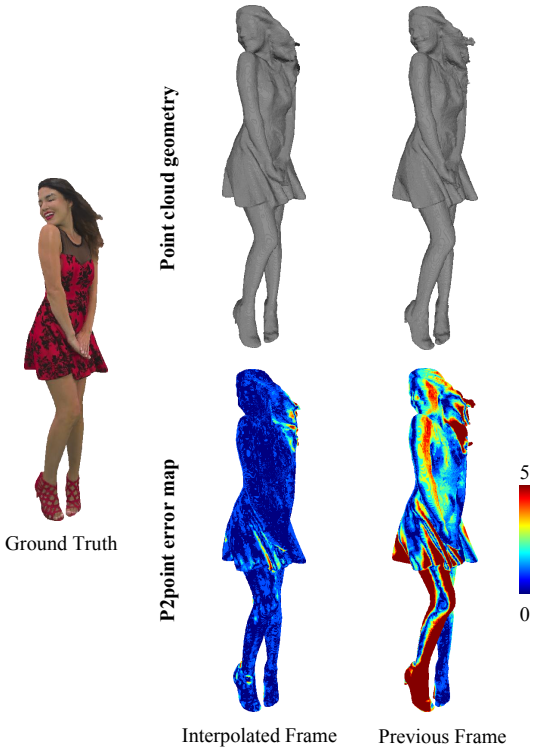

Fig. 3. Visual Results on *Redandblack* dataset.

3.3. Evaluation Metrics

We consider two commonly-used evaluation metrics that compare the reconstructed point cloud to the ground truth point cloud to quantitatively evaluate the performance of our method. These metrics are Chamfer distance (CD) and point-to-point (D1) based mean squared error peak signal-to-noise ratio (MSE PSNR). MSE PSNR has been adopted by both MPEG and AVS standards as an evaluation metric for dynamic point cloud quality [18]. We obtain the point-to-point geometry PSNRs using MPEG's *pc_error* tool [19].

3.4. Objective Evaluation and Visual Results

We compare our method with different variations of our framework as well as with **Identity** where we simply duplicate the first point cloud frame as the intermediate point cloud. The objective results of our evaluations are shown in Table 1. **Ours** is the framework described in this paper. **Ours-w/o Pretrained** is the framework where the Encoder is not pretrained on ShapeNet. **Ours-Single Loss** framework employs only a single loss ($\mathcal{L}_3$) and no longer employs losses $\mathcal{L}_1$ and $\mathcal{L}_2$. **Ours-Fuse3D** framework utilizes 3D convolutions in the fuse block rather than the 4D convolutions. In this method, the features are simply added and 3D convolutions are employed. As can be seen, our method considerably performs better than *Identity*. Furthermore, the results show that each sub-component of our framework is essential and improves the results considerably. The results show that the fusion using 4D convolutions is much more efficient in learning the intermediate frame features compared to 3D convolutions.

An example of visual results for sequence *redandblack* is shown in Fig. 3. We can see that our method generates limited outliers while populating points very close to the surface of the ground truth point cloud. This results in a high-resolution point cloud with considerably better quality than the *Identity* where the previous frame is used.

4. CONCLUSION

In this work, we propose the first dynamic point cloud interpolation framework for dense high-resolution point clouds. While the previous point cloud interpolation methods are limited to point cloud scenes, our framework is able to process and interpolate frames on a high-resolution dynamic point cloud. We employ a pretrained multi-scale encoder module to extract features at multiple scales. The encoder module is pre-trained on a static object dataset (ShapeNet). We introduce a novel 4D feature fusion module that utilizes 4D learning to merge 3D features from two consecutive frames at multiple scales. Finally, our multi-scale point cloud synthesis module hierarchically reconstructs the interpolated point cloud frame at different resolutions. We test our framework on a diverse set of high-resolution dynamic point cloud sequences. The evaluation results validate our network design and demonstrate the effectiveness of our method.

5. REFERENCES

[1] Anique Akhtar, Birendra Kathariya, and Zhu Li, "Low latency scalable point cloud communication," in *2019 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2019, pp. 2369–2373.

[2] Anique Akhtar, Junchao Ma, Rubayet Shafin, Jianan Bai, Lianjun Li, Zhu Li, and Lingjia Liu, "Low latency scalable point cloud communication in VANETs using V2I communication," in *ICC 2019-2019 IEEE International Conference on Communications (ICC)*. IEEE, 2019, pp. 1–7.

[3] Maja Krivokuca, Philip A Chou, and Patrick Savill, "8i voxelized surface light field (8iVSLF) dataset," *ISO/IEC JTC1/SC29/WG11 MPEG, input document m42914*, 2018.

[4] Anique Akhtar, Wen Gao, Li Li, Zhu Li, Wei Jia, and Shan Liu, "Video-based Point Cloud Compression Artifact Removal," *IEEE Transactions on Multimedia*, 2021.

[5] Simon Niklaus, Long Mai, and Feng Liu, "Video frame interpolation via adaptive separable convolution," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 261–270.

[6] Simon Niklaus, Long Mai, and Feng Liu, "Video frame interpolation via adaptive convolution," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 670–679.

[7] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang, "Depth-aware video frame interpolation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3703–3712.

[8] Simon Niklaus and Feng Liu, "Context-aware synthesis for video frame interpolation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1701–1710.

[9] Xingyu Liu, Charles R Qi, and Leonidas J Guibas, "FlowNet3D: Learning Scene Flow in 3D Point Clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 529–537.

[10] Zirui Wang, Shuda Li, Henry Howard-Jenkins, Victor Prisacariu, and Min Chen, "FlowNet3D++: Geometric Losses For Deep Scene Flow Estimation," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 91–98.

[11] Xiuye Gu, Yijie Wang, Chongruo Wu, Yong Jae Lee, and Panqu Wang, "HPLFlowNet: Hierarchical Permutohedral Lattice FlowNet for Scene Flow Estimation on Large-scale Point Clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3254–3263.

[12] Wenxuan Wu, Zhiyuan Wang, Zhuwen Li, Wei Liu, and Li Fuxin, "PointPWC-Net: A Coarse-to-Fine Network for Supervised and Self-Supervised Scene Flow Estimation on 3D Point Clouds," *arXiv preprint arXiv:1911.12408*, 2019.

[13] Himangi Mittal, Brian Okorn, and David Held, "Just go with the flow: Self-supervised scene flow estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 11177–11185.

[14] Fan Lu, Guang Chen, Sanqing Qu, Zhijun Li, Yinlong Liu, and Alois Knoll, "PointINet: Point Cloud Frame Interpolation Network," *arXiv preprint arXiv:2012.10066*, 2020.

[15] Anique Akhtar, Wen Gao, Xiang Zhang, Li Li, Zhu Li, and Shan Liu, "Point Cloud Geometry Prediction Across Spatial Scale using Deep Learning," in *2020 IEEE International Conference on Visual Communications and Image Processing (VCIP)*. IEEE, 2020, pp. 70–73.

[16] Christopher Choy, JunYoung Gwak, and Silvio Savarese, "4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3075–3084.

[17] 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*, vol. 7, pp. 8, 2017.

[18] Dong Tian, Hideaki Ochimizu, Chen Feng, Robert Cohen, and Anthony Vetro, "Geometric distortion metrics for point cloud compression," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3460–3464.

[19] Dong Tian, Hideaki Ochimizu, Chen Feng, Robert Cohen, and Anthony Vetro, "Geometric distortion metrics for point cloud compression," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 3460–3464.