International Journal of Computer Vision
https://doi.org/10.1007/s11263-021-01503-6

Deep Learning Geometry Compression Artifacts Removal for Video-Based Point Cloud Compression

Wei Jia[1] · **Li Li**[2] · **Zhu Li**[1] · **Shan Liu**[3]

Received: 17 December 2020 / Accepted: 8 July 2021

Abstract

Point cloud is an essential format for three-dimensional (3-D) object modelling and interaction in Augmented Reality and Virtual Reality applications. In the current state of the art video-based point cloud compression (V-PCC), a dynamic point cloud is projected onto geometry and attribute videos patch by patch, each represented by its texture, depth, and occupancy map for reconstruction. To deal with occlusion, each patch is projected onto near and far depth fields in the geometry video. Once there are artifacts on the compressed two-dimensional (2-D) geometry video, they would be propagated to the 3-D point-cloud frames. In addition, in the lossy compression, there always exists a tradeoff between the rate of bitstream and distortion. Although some geometry-related methods were proposed to attenuate these artifacts and improve the coding efficiency, the interactive correlation between projected near and far depth fields has been ignored. Moreover, the non-linear representation ability of Convolutional Neural Network has not been fully considered. Therefore, we propose a learning-based approach to remove the geometry artifacts and improve the compressing efficiency. We have the following contributions. We devise a two-step method working on the near and far depth fields decomposed from geometry. The first stage is learning-based Pseudo-Motion Compensation. The second stage exploits the potential of the strong correlations between near and far depth fields. Our proposed algorithm is embedded in the V-PCC reference software. To the best of our knowledge, this is the first learning-based solution of the geometry artifacts removal in V-PCC. The extensive experimental results show that the proposed approach achieves significant gains on geometry artifacts removal and quality improvement of 3-D point-cloud reconstruction compared to the state-of-the-art schemes.

Keywords Point cloud compression · Denoising · Deep learning · Depth field

1 Introduction

Due to the massive demands for stereoscopic experience, three-dimensional (3-D) sensing and scanning instruments including Light Detection and Ranging (LIDAR) scanners (Jang et al. 2019; d'Eon et al. 2017) and RGB-D cameras (Guo et al. 2017; d'Eon et al. 2017) are developing unprecedentedly. Those 3-D devices daily generate an enormous amount of data. To visualize these 3-D data vividly, some 3-D representing methods, such as point clouds, light fields, and polygon meshes, progress rapidly. Those stereo expressing approaches are capable of representing the 3-D volumetric data in a realistic and immersive way. Point cloud is especially popular among these methods since we can acquire them more easily, render them more realistically, and manipulate them more feasibly.

Point cloud is an important format for various 3-D based volumetric technologies such as virtual reality (VR), aug-

Communicated by Dong Xu.

✉ Zhu Li
zhu.li@ieee.org

Wei Jia
wj3wr@umsystem.edu

Li Li
lil1@ustc.edu.cn

Shan Liu
shanl@tencent.com

[1] University of Missouri-Kansas City, Kansas City, MO 64110, USA

[2] University of Science and Technology of China, Hefei 230026, China

[3] Tencent Media Lab, Palo Alto, CA 94306, USA

mented reality (AR), and mixed reality (MR) (Bruder et al. 2014) all advancing rapidly. The number of 3-D applications (Tulvan et al. 2016; Chen et al. 2014) is therefore increasing significantly based on these immersive and realistic technologies.

For instance, an applying case is navigation (Chen et al. 2017; Mobile Mobile). The mobile navigation (Mobile Mobile) aims to create a 3-D map with localization data, global positioning system (GPS) images, and depth data. Other applications are VR/AR immsersive videos, games (Sportillo et al. 2017) and telecommunications (Fuchs et al. 2014). Since it is feasible to render and visualize point clouds, we can use a collection of 3-D point clouds to represent the low delay 3-D stream of the high quality (4K or 8K) immersive telecommunications. In addition, the historic relic (Culture Culture) is another interesting application. This type of heritage application is capable of providing an immersive stereo experience by visualizing the real historical relic to billions of points. However, point clouds contain a lot of digital data, so that storing and streaming such massive data is very difficult. The conflict between demands and capacity of the storage pushes an emergence of useful point cloud compression (PCC) solutions.

Driven by this technique requirement, Moving Pictures Experts Group (MPEG) starts the standardization (Schwarz et al. 2018) of PCC. Owing to the different densities of point clouds, there are two types of schemes incorporating video-based point cloud compression (V-PCC) (Schwarz et al. 2018) and geometry-based point cloud compression (G-PCC) (Schwarz et al. 2018). V-PCC mainly works on dense point clouds while G-PCC works on sparse point clouds such as large-scale point cloud maps that are produced by simultaneous localization and mapping (SLAM) algorithms (Sun et al. 2017). We mainly discuss the V-PCC development in this work. V-PCC divides a point cloud into many 3-D patches at first. V-PCC then projects the generated 3-D patches onto 2-D planes and packs them into a 2-D geometry video and a texture video. Subsequently, to encode 2-D videos efficiently, V-PCC maintains spatial continuity by padding the void area of geometry and texture videos before the 2-D compression. V-PCC eventually compresses the padded geometry and texture videos with 2-D video codecs such as Advanced Video Coding (AVC) (Wiegand et al. 2003), High Efficiency Video Coding (HEVC) (Sullivan et al. 2012), and Versatile Video Coding (VVC) (Bross et al. 2019) in the lossy mode.

Due to this lossy compression of V-PCC, distortions exist in the 2-D geometry reconstruction and 3-D point-cloud reconstruction. Essentially, the 2-D geometry video is the depth information of the 3-D point cloud. Once the 2-D geometry reconstruction distorts, the 3-D point-cloud reconstruction will also have artifacts. For instance, when the 2-D geometry reconstruction loses some pixels, the corresponding points of the 3-D point-cloud reconstruction will miss as well. Similarly, if the 2-D geometry reconstruction inserts a few noisy redundant pixels, the 3-D point-cloud reconstruction will introduce the corresponding redundant points. In addition, if some values of the 2-D geometry reconstruction verify, compared to the origin, the corresponding points of the 3-D reconstruction will locate in mistaken positions. All these cases degrade the quality of 3-D point-cloud reconstruction and result in artifacts.

To attenuate artifacts, researchers proposed some 2-D geometry-related methods (Andrivon et al. 2020; Cai et al. 2018; Dawar et al. 2018; Nakagami 2018). Among them, (Rhyu et al. 2018) as a geometry padding approach proposed padding the empty space between patches with neighboring patch information. This method is especially beneficial for coding efficiency in all the intra cases. To further decrease the distance between the point-cloud reconstruction and its origin, Graziosi and Tabatabai (2019) searched and picked up a depth value from a depth candidate list in the 2-D geometry video for padding the 3-D geometry. This encoder-only method is the first method using geometry reconstruction to process the inserted redundant positions.

Although these methods have achieved successes, the non-linear representation ability of the learning-based Voulodimos et al. (2018) approach has not been fully considered. There is still considerable space to develop a better learning-based geometry distortion removal algorithm.

Therefore, to take care of the tradeoff between the distortion and bitrate, in this work, we propose for the first time a learning-based approach removing the geometry artifacts for a better quality of the 3-D point-cloud reconstruction. We make the following contributions.

- To address the geometry artifacts problem, we propose a two-step method working on the near and far depth fields decomposed from geometry. The first stage is learning-based Pseudo-Motion Compensation. The second stage exploits the potential of the strong correlations between near and far depth fields. To the best of our knowledge, this is the first learning-based solution of the geometry artifacts removal in V-PCC.
- Our proposed algorithm is embedded in the V-PCC reference software for simulation. We have conducted extensive experiments to compare with state-of-the-art (SOTA) methods to demonstrate the effectiveness of the proposed method. We thoroughly analyze the experimental results to give more insights into the problem.

We organize the remainder of this paper as follows. We review the related works on point cloud compression in Sect. 2, followed by our motivation and observations on geometry in Sect. 3. We introduce the proposed geometry artifacts removal with two stages in Sect. 4. In Sect. 5, we report and analyze the experimental results comprehensively. We summarize this paper in Sect. 6 briefly.

2 Related Work

This section briefly reviews the previous point cloud compression (PCC) works and the geometry improvement methods in V-PCC.

2.1 Point Cloud Compression

The PCC methods can be roughly summarized into two groups, which are the group of 3-D-based and 2-D-based approaches and the group of deep learning-based approaches.

1) 3-D-based and 2-D-based methods. Because there is no strong time correlation between the neighboring frames, the 3-D-based methods can not precisely estimate a motion between points in neighboring frames. Kammerl et al. (2012) devised a lossy compression approach for dynamic point cloud streaming. The co-located octree node of the reference point-cloud frame predicted the current point-cloud frame. However, we can only apply this approach to a few moved point-cloud frames. Thanou et al. (2016) used a set of graphs to represent the time-varying geometry of these point-cloud frames. Based on this, they cast 3-D motion estimation as a feature-matching issue between consecutive point-cloud frames. However, this method did not precisely estimate the motion vectors of some objects in point-cloud frames.

de Queiroz and Chou (2017) proposed a simple codec. This coder segmented the voxelized point cloud at each frame into blocks of voxels. Their proposed method executed the 3-D translational motion estimation block by block to find the corresponding block in the reference point-cloud frame.

In addition, Mekuria et al. (2016) further imported iterative closest point (ICP) instead of translational motion model to better formulate the motions in neighboring point-cloud frames.

These methods could relieve the suffering from 3-D motion estimation and motion compensation to some extent.

To solve the bottleneck that streaming and caching the point clouds requires large bandwidth and storage space, Sun et al. (2019) proposed a clustering method starting with a range image-based 3-D segmentation. In addition, it introduced a prediction with the depth modeling modes for depth map coding. Nevertheless, without flexible block partition and more efficient motion estimation schemes, the coding efficiency of the dynamic point cloud compression (DPCC) still cannot be compatible with the 2-D-based approach.

Because codecs such as AVC, HEVC, and VVC have proven that the 2-D video compression algorithms are efficient, researchers proposed 2-D-based methods to transform the 3-D dynamic point cloud to 2-D videos for compression.

Budagavi et al. (2017) developed a method to code a projected 2-D video acquired from ordering points in a 3-D point cloud with HEVC. However, this work could not further utilize the inter-prediction information since the obtained video did not have many spatial and temporal correlations. To attenuate this deficiency, He et al. (2017) proposed a cubic projection method to convert a 3-D dynamic point cloud to a 2-D video. Although this work improved video coding performance, this approach resulted in missing points due to occlusion.

To minimize the number of occluded points, Lasserre et al. (2017) proposed an approach that combined the octree and projection. Mammou et al. (2017) devised a method that projected a 3-D dynamic point cloud to 2-D videos by a patch-based scheme. Their motivation was to consider projecting more points while reducing the bit cost on 2-D video coding as much as possible. Packing a group of patches that consists of 2-D pixels converted from 3-D points was the main philosophy of the patching method. This work packed these 2-D patches onto a video then compressed by codecs such as HEVC. Li et al. (2019) proposed a general model utilizing the 3-D motion and 3-D to 2-D correspondence to calculate the 2-D motion vector (MV). Compared to other proposals, the patch-based method (Preda 2017) performed better in coding efficiency. MPEG Immersive media working group (MPEG-I) adopted (Preda 2017) as a V-PCC standard. This approach has shown its efficiency with excellent performance. However, we have not improved the geometry video to its full potential extent, which intrinsically guides the 3-D point-cloud reconstruction process.

2) Deep learning-based methods. Point cloud processing is the fundamental component for any deep-learning-based point-cloud applications. Rather than compress point cloud data directly, Tu et al. (2016) proposed converting the packet data, which is raw point cloud data, losslessly into range images previously. To avoid unnecessarily voluminous rendering data, Qi et al. (2017) proposed a type of neural network that directly consumes point clouds, which well respects the permutation invariance of points in the input.

However, many works process the 3-D videos frame-by-frame either through 2-D convents or 3-D perception algorithms. Choy et al. (2019) proposed a new sparse tensor-based 3-D point-cloud processing method called Minkowski. This network for spatio-temporal perception can directly process such 3-D videos using high-dimensional convolutions.

Gojcic et al. (2020) proposed an end-to-end algorithm for joint learning of both parts of initial pairwise and the globally consistent refinement.

The deep learning-based PCC methods utilized the non-linear ability of Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) to improve the efficiency of PCC. To require less volume while giving the same decompression accuracy, Tu et al. (2019) used an RNN and residual blocks to compress one frame from 3-D LiDAR. However, these two methods did not use joint optimization. Quach et al. (2019) performed the joint optimization of both rate and distortion with a tradeoff parameter on a data-driven method. They created this method for the geometry static point cloud compression (SPCC) based on learned convolutional transforms and uniform quantization. Nevertheless, there is still space to improve the architecture of the network. Huang and Liu (2019) presented a new 3-D geometry PCC method based on an auto-encoder network. They used the extracted features of the raw model in the encoder to compress the original data to be bitstream. They further compressed the bitstream with sparse coding. Nevertheless, this method worked on the well-segmented objects.

To reduce the bitrate, Huang et al. (2020) proposed a deep compression method to decrease the memory footprint of LiDAR point clouds with the sparsity and structural redundancy between points. Nevertheless, the spatio-temporal relationships had not been fully considered. To reduce the bitrate of both geometry and intensity values, Biswas et al. (2020) exploited spatio-temporal relationships across multiple LiDAR sweeps and proposed a conditional entropy model. This models the probabilities of the octree symbols by utilizing both coarse level geometry, previous sweeps' geometric, and intensity information.

2.2 Geometry Related Methods in V-PCC

MPEG calls some geometry-related works to improve the point cloud quality and compression efficiency during the V-PCC standardization. Cai et al. (2018) proposed encoding the projected information as an absolute depth value instead of an error between the far layer and near layer.

However, this solution can not improve the quality of the point-cloud reconstruction. To obtain a better tradeoff between these artifacts and bitrate, Olivier and Llach (2018) proposed improving the projection of connected components into patches.

Nevertheless, it did not resolve the empty space between patches, which directly impacted the coding efficiency. In order to take care of the empty space, Rhyu et al. (2018) proposed dilating the gap between patches by expanding the geometry from boundaries of patches. Although this way minimized block artifacts in the decoded 2-D video, it still compressed the near and far geometry frames separately.

To get better encoding efficiency, Dawar et al. (2018) proposed using one frame instead of two frames to encode 2-D near and far layers.

However, this method interpolated pixels from spatial neighbors leading to distortions to some extent and did not save decoding time.

To keep the reconstruction quality but decrease the decoding time complexity, Nakagami (2018) proposed an upgraded geometry smoothing. The geometry smoothing of V-PCC aims at alleviating potential discontinuities that may arise at the patch boundaries due to compression artifacts. This proposed approach moved boundary points to the centroid of their nearest neighbors. It skipped the smoothing for points inside a patch and pre-calculated the neighbor points centroid instead of the nearest neighbor (NN) search. Nevertheless, this approach did not exploit the geometry information of the 3-D reconstruction.

Graziosi and Tabatabai (2019) therefore proposed padding the geometry with the reconstructed depth value for those positions introduced by occupancy map rescaling. To decrease the distance between the geometry reconstruction and origin, this encoder-only method searched and selected a depth value from a range of possible geometry depth values for an inserted point. This method only used some limited neighborhoods and geometry characteristics. The color consistency and surface consistency of the lines in the reconstruction could be beneficial. In addition, this method could still be improved further by occupancy map reconstruction methods.

Although V-PCC adopted these two works into the reference software due to their good performances, they did not fully exploit the strong correlations and interactions between the geometry near and far layers. In addition, the non-linear representation ability of CNN has not been considered carefully in the V-PCC geometry distortion removal.

3 Motivation

To explain the importance and necessity of the 2-D geometry improvement in V-PCC, we need to figure out the property of the geometry step by step. We first elaborate on how V-PCC converts the 3-D point cloud to the 2-D near and far layers of geometry frames in the projecting process. Then we state the strong correlations between near and far layers of the geometry. We explain all these above in Sect. 3.1. To better understand that how geometry propagates its impact on point-cloud reconstruction, we previously introduce the important role that geometry plays in the point cloud reconstructing process in Sect. 3.2. Finally, we explain the impact of 2-D geometry on 3-D point-cloud reconstruction in both objective and subjective performance in Sect. 3.3.

3.1 The Projection from 3-D to 2-D

A point-cloud frame consists of a collection of points within a 3-D volumetric space that is with its coordinates, geometry, and attributes information, as shown in Fig. 1a. To use the proven powerful 2-D codecs such as HEVC and VCC, V-PCC first projects the volumetric 3-D point-cloud frames to 2-D video frames. The whole projection process contains

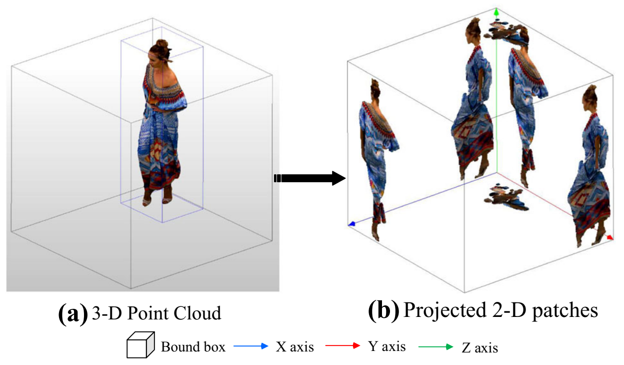


Fig. 1 The projection process (Committee 2020) from 3-D to 2-D in V-PCC. The segmented 3-D patches are projected to six planes of its bounding box.

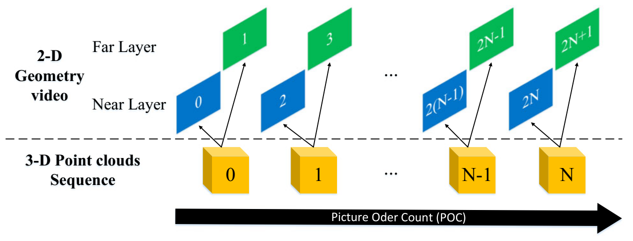


Fig. 2 The correlations between 3-D point clouds sequence and 2-D geometry video. A 2-D patch with picture order count (POC) $2N$ of near layer and one with $2N+1$ of far layer are both derived from the same 3-D point cloud patch with POC N.

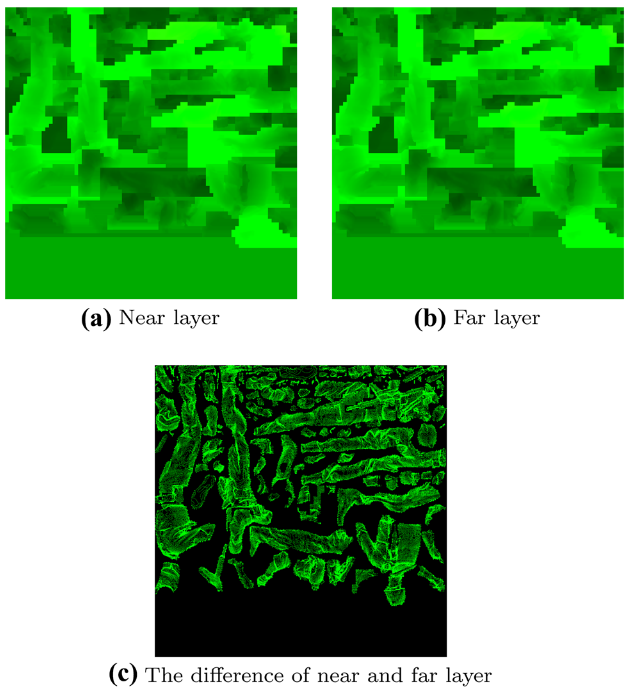


Fig. 3 Near **a** and far **b** layer frames in 2-D geometry video. **c** is the difference δ between the near layer frame with POC $2N$ and its corresponding POC $2N+1$ far layer frame. Since generally, the difference δ is a minimal value, it is difficult to directly see the difference between the near and far layer frames unless making subtraction with them as **c**.

three stages: patch segmentation, patch generation, and patch packing. The V-PCC patch denotes a set of information that describes the point cloud in a 3-D bounding box. This information incorporates points, corresponding geometry, and attribute descriptions along with the atlas information.

The patch segmentation aims to decompose the 3-D point clouds into many patches. Then during the process of patch generation, as shown in Fig. 1, V-PCC projects the segmented 3-D patches to six planes of its bounding box. To appropriately resolve the problem that different 3-D points are projected onto the identical 2-D pixel, V-PCC projects each patch onto two depth fields. More specifically, we assume that $P(u, v)$ is the collection of points of a patch projected to the identical sample (u, v). The near layer stashes the point of $P(u, v)$ with the lowest depth d_0. The far layer projects the point of $P(u, v)$ with the highest depth within $[d_0, d_0 + \delta]$, where δ denotes the thickness of the projected surface. Intrinsically, as illustrated in Fig. 2, a 2-D near layer with picture order count (POC) $2N$ and a far layer with POC $2N + 1$ are both derived from the same 3-D point-cloud frame with POC N. Fig. 3 visualizes an example of the near layer frame with POC $2N$, corresponding POC $2N + 1$ far layer frame and their difference. Since generally, the δ is a minimal value, it is difficult to directly see the difference between the near and far layer frames unless we make subtraction with them as Fig. 3c. Afterward, to generate the 2-D geometry and attributes videos, V-PCC arranges the projected 2-D patches compactly onto a 2-D frame with size $W \times H$. We call this stage patch packing. Once we obtain 2-D geometry videos, codecs can compress them efficiently.

3.2 The Reconstruction Process

Before reconstructing point clouds, the V-PCC decoder first demultiplexes the compressed bitstream into geometry, attributes, occupancy map, and atlas streams. The atlas mainly contains auxiliary patch information. The occupancy map is a binary signal video implicating whether a 2-D pixel exists in the original 3-D point cloud as a point. Once the reconstructing process starts, as described in Fig. 4, V-PCC reconstructs the atlas first. We can see from Fig. 4a that V-PCC only builds padded patch rough sketches. After V-PCC merges the decoded occupancy map into the reconstruction, the occupancy status of the points becomes clear because the occupancy map removes the padded area.

However, there is still no clear person shape information. When V-PCC adds the geometry information into the reconstruction, all 3-D point clouds are almost created except the

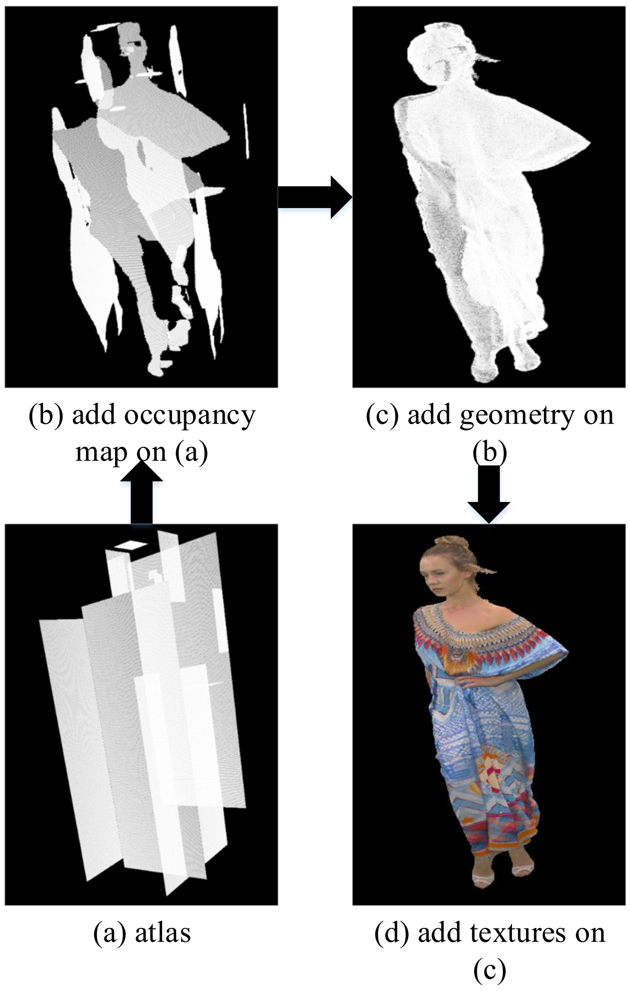


Fig. 4 The process of 3-D point-cloud reconstruction (Committee 2020) in V-PCC. The atlas **a** is first reconstructed. Only padded patch rough sketches are built. After the decoded occupancy map is merged into the reconstruction **b**, the points occupancy status becomes clearly shown because the occupancy map has removed the padded area. When the geometry information is added into the reconstruction **c**, all 3-D point clouds are almost created except the color attributes. Finally, the attributes are drawn on the reconstruction **d**. This visualizing reconstruction process demonstrates how the 2-D geometry enormously impacts the subjective quality of 3-D point clouds reconstruction.

color attributes. Finally, V-PCC draws the attributes on the reconstruction.

3.3 The Impact of the 2-D Geometry to the 3-D Point Cloud

The visualizing reconstruction process above has demonstrated that how the 2-D geometry enormously impacts the subjective quality of 3-D point clouds reconstruction. Furthermore, Fig. 5 shows that how the V-PCC propagates the 2-D geometry artifacts significantly to the 3-D point-cloud reconstruction. The sub-figures of the first row (a), (b), and (c) are the 2-D geometry, attributes, and 3-D point cloud of ground truths, respectively. With the same arrangement, the sub-figures of the second row are from anchor reconstructions. Because it is difficult to directly recognize the difference of the geometry value between the anchor and ground truth with our eyes, we visualize their difference in (d). We can clearly see the geometry value distortion of the gun in the enlarged area of (d). As explained in Sect. 3.2, the V-PCC reconstructs the 2-D attributes video with a 2-D reconstructed geometry video, containing useful pixel location information. The V-PCC finally reconstructs the 3-D point clouds with recolored and smoothed attributes video in 3-D space. From the enlarged areas, we can clearly see that the V-PCC propagates the 2-D geometry distortion to attributes reconstruction. Then, the V-PCC brings the attributes artifacts into the 3-D point-cloud reconstruction.

On the aspect of objective influence, we can find the same importance of geometry video in Table 1. The higher bitrate $r2$ outperforms the lower one $r1$ on the PSNR of 2-D geometry video on either near or far layer. These gains are obviously propagated to 3-D point clouds on either point-to-point or point-to-plane (Tian et al. 2017) PSNR. These altogether demonstrate the consistency of objective geometry qualities on both 2-D and 3-D sides. Based on the analysis and observations above, if an efficient algorithm improving 2-D geometry can be carefully devised, we can expect similar ideal performance on the 3-D point cloud.

4 The Proposed Algorithm

Based on the observations and analysis above in Sect. 3, the 2-D geometry impacts the 3-D point-cloud reconstruction significantly. Therefore, to remove artifacts of the near and far layers in geometry, we develop an algorithm with a two-step strategy. In the first step, we not only improve the near and far layers with individual CNNs, but also use the enhanced near reconstruction as the Pseudo-Motion Compensation (PMC) for augmenting the far layer. In the second step, we dive into the interactions between near and far layers and devise an X-like interacting network (XInteractNet) to fully use their strong similarities for further enhancement. Specifically, we elaborate an in-depth discussion on the two-step scheme in Sect. 4.1, design of XInteractNet in Sect. 4.2, loss function in Sect. 4.3, dataset in Sect. 4.4, and training process in Sect. 4.5.

4.1 Artifacts Removal of Geometry with Two Steps

We carefully devise our algorithm on two aspects. On the one hand, as observed in Sect. 3.2 and Sect. 3.3, the geometry plays an important role in the 3-D point-cloud reconstruction. We, therefore, focus on removing artifacts of geometry to improve the quality of 3-D point clouds further. On the other hand, as elaborated in Sect. 3.1, in the projection process of

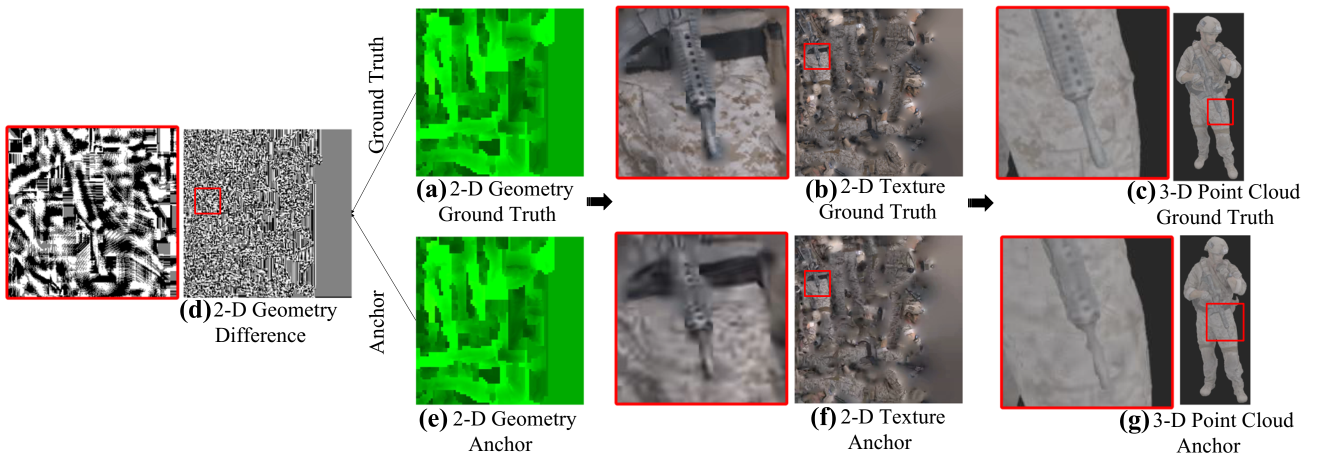

Ground Truth
Anchor
(a) 2-D Geometry Ground Truth
(b) 2-D Texture Ground Truth
(c) 3-D Point Cloud Ground Truth
(d) 2-D Geometry Difference
(e) 2-D Geometry Anchor
(f) 2-D Texture Anchor
(g) 3-D Point Cloud Anchor

Fig. 5 Impact of the 2-D geometry artifacts to the 3-D point cloud artifacts. The sub-figures of the first row **a**, **b**, and **c** are the 2-D geometry, attributes, and 3-D point cloud of ground truths, respectively. With the same arrangement, the sub-figures of the second row are from anchor reconstructions. Because it is difficult to directly recognize the difference of the geometry value between the anchor and ground truth with our eyes, we visualize their difference in **d**. White and black pixels represent the different ones, while gray pixels are the same ones. We can clearly see the geometry value distortion of the gun in the enlarged area of **d**. From the enlarged areas, we can clearly see that the V-PCC propagates the 2-D geometry distortion to attributes reconstruction. Then, the V-PCC brings the attributes artifacts into the 3-D point-cloud reconstruction.

Table 1 2-D and 3-D geometry PSNR comparison between lower and higher bitrates of V-PCC Anchor (Point Point) within soldier first 32 frames

PSNR (dB)	Metrics	$r1$ (lower)	$r2$ (higher)
2-D geometry video	Near layer	46.8062	**49.1117**
	Far layer	46.4561	**48.7152**
	Avg. all	46.6312	**48.9134**
3-D point clouds	Point-to-point error	65.66	**67.45**
	Point-to-plane error	67.42	**69.48**

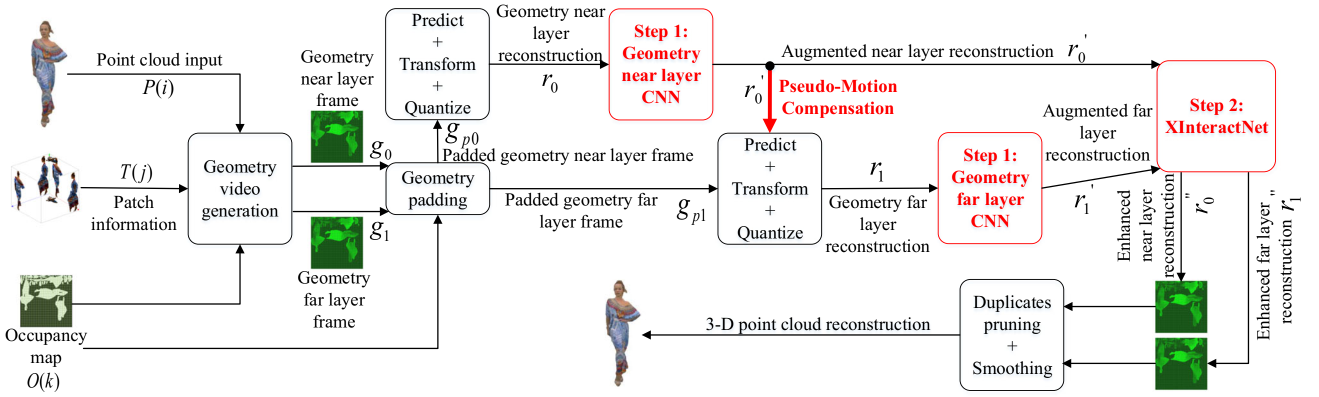

Point cloud input
$P(i)$
$T(j)$
Patch information
Occupancy map
$O(k)$
Geometry video generation
Geometry near layer frame
g_0
Geometry far layer frame
g_1
Predict + Transform + Quantize
Geometry padding
g_{p0}
Padded geometry near layer frame
Padded geometry far layer frame
g_{p1}
Geometry near layer reconstruction
r_0
Step 1: Geometry near layer CNN
Augmented near layer reconstruction r_0'
r_0'
Pseudo-Motion Compensation
Predict + Transform + Quantize
r_1
Geometry far layer reconstruction
Step 1: Geometry far layer CNN
Augmented far layer reconstruction
r_1'
Step 2: XInteractNet
Enhanced near layer reconstruction r_0''
Enhanced far layer reconstruction r_1''
Duplicates pruning + Smoothing
3-D point cloud reconstruction

Fig. 6 The proposed two-step method is embedded in the V-PCC encoding scheme. The target of the first step is to denoise the coarse artifacts of the reconstructed near layer r_0 and far layer r_1 first. The augmented near layer reconstruction r_0' as PMC iteratively participates into the far layer g_{p1} prediction to generate the far layer reconstruction r_1. Afterward, r_1 is fed into the far layer step one CNN to produce a better reconstruction r_1'. The objective of the second step is to utilize the interactive information $I_s(r_0', r_1')$ between near layer r_0' and far layer r_1' in full extent for further removing artifacts. We input the outputs of step one including r_0' and r_1' into XInteractNet to achieve enhanced corresponding near layer reconstruction r_0'' and far layer reconstruction r_1''. They are finally used for reconstructing the 3-D point cloud.

Algorithm 1 The flow of two-step approach

```
Input: The near depth field reconstruction $r_0$, the far depth field $g_{p1}$.
Output: The artifacts removed near depth field $r_0^{''}$ and far depth field $r_1^{''}$.

if Step one initialization successes then
    Input $r_0$ into the step one $S_0(\cdot)$ CNN and output the augmented near reconstruction $r_0^{'}$;
    Input $g_{p1}$ and Pseudo-Motion Compensation $r_0^{'}$ into the predictor and output the far reconstruction $r_1$;
    Input $r_1$ into the step one $S_1(\cdot)$ CNN and output the augmented far reconstruction $r_1^{'}$;
end
if Step two initialization successes then
    Input $r_0^{'}$ and $r_1^{'}$ into the step two XInteractNet;
    Compute the down features $x_0^d$ and $x_1^d$ with X Down Block $\Phi^d(x_0(i), x_1(i))$;
    Compute the up features $y_0^u$ and $y_1^u$ with X Up Block $\Phi^u(y_0(i), y_1(i))$;
    Output the artifacts removed near depth field $r_0^{''}$ and far depth field $r_1^{''}$;
end
```

geometry, V-PCC projects a 3-D point-cloud frame to two 2-D different layer frames, including a near one denoted as g_0 with depth d_0 and a far one denoted as g_1 with depth d_1. Since we essentially acquire these two 2-D frames from the same 3-D point-cloud frame, we consider utilizing their similarities and interactive information $I_s(g_0, g_1)$ as much as possible for denoising geometry. Hence, we propose a CNN-based two-step method that uses the interaction $I_s(g_0, g_1)$ to enhance near layer g_0 and far layer g_1 as frequently as possible.

Figure 6 describes the upgraded V-PCC encoding scheme embedded in the proposed two-step approach. Initially, the upgraded V-PCC uses the 3-D input of point clouds $P(i)$ to generate patch information $T(j)$. V-PCC then generates the occupancy map $O(k)$ with these patches information $T(j)$ during the patch packing process. Afterwards, V-PCC generates the 2-D near layer g_0 and far layer g_1 from the 3-D input $P(i)$, patch information $T(j)$ and occupancy map $O(k)$ for further padding. V-PCC then pads far layer g_0 and near layer g_1 to generate corresponding g_{p0} and g_{p1} that are beneficial for predicting, transforming and quantizing.

The target of the first step is to denoise the coarse artifacts of the reconstructed near layer r_0 and far layer r_1 first. This way can provide step two with reconstructions of better quality as dual inputs. The inter prediction in HEVC adopts motion compensation technologies. The principle of motion compensation in codec is to search out a reference frame containing reference blocks for predicting the current frame. Since we replace the reference frame with an enhanced near layer reconstruction in the upgraded codec for predicting the far layer reconstruction, we call this process as PMC. Specifically, at first, we feed r_0 into a CNN $S(\cdot)$ (near CNN) that is an autoencoder structure with four convolutional layers. Then the augmented near layer reconstruction $r_0^{'}$ as PMC iteratively participates into the padded far layer g_{p1} prediction to generate the far layer reconstruction r_1. We embed the algorithm of step one into the V-PCC and HECV encoders, configuring the near layer reconstruction $r_0^{'}$ as an I frame while the geometry far layer reconstruction r_1 as a P frame. The overhauled codec sets the near layer $r_0^{'}$ (I frame) as a reference frame for predicting the far layer reconstruction r_1 (P frame). The updated encoder estimates and compensates the motions for the far layer reconstruction r_1 with the enhanced near layer reference frame $r_0^{'}$. Accordingly, due to this PMC, if the quality of the near layer reference frame $r_0^{'}$ is better, the quality of the far layer reconstruction r_1 will be better. Afterward, we feed the far reconstruction r_1 into the far layer step one CNN (far CNN), which has the same architecture as near CNN, to produce a better reconstruction $r_1^{'}$.

The objective of the second step is to utilize the interactive information $I_s(r_0^{'}, r_1^{'})$ between near layer $r_0^{'}$ and far layer $r_1^{'}$ in full extent for further removing artifacts. Accordingly, we devise a network with two inputs and two outputs, namely XInteractNet mining the interactive information $I_s(r_0^{'}, r_1^{'})$ as much as possible. We input the outputs of step one including $r_0^{'}$ and $r_1^{'}$ into XInteractNet to achieve enhanced corresponding near layer reconstruction $r_0^{''}$ and far layer reconstruction $r_1^{''}$. We finally use $r_0^{''}$ and $r_1^{''}$ for reconstructing the 3-D point cloud. We elaborate the proposed two-step artifacts removal algorithm of near and far depth fields in Algorithm 1.

4.2 XInteractNet

The architecture of the proposed XInteractNet in the second step is illustrated in Fig. 7. Let us define $x_0(i)$ and $x_1(i)$ are the near and far layer feature map of No. i level of the XInteractNet, respectively. In order to share the interactive information $I_s(x_0(i), x_1(i))$ between near and far layers as much as possible, we devise a network model pair called X interactive down and up blocks as the basic unit of XInteractNet. According to the observations above in Sect. 3, for a given near and far layer pairs, there exists a high correlation between the near and far layer feature maps at every level in the network. Therefore, the main idea behind the design of the proposed x interactive down and up blocks is sharing the interactive information $I_s(x_0(i), x_1(i))$ after every convolutional computation level.

Specifically, in the X down block, to share the interactive information $I_d(x_0(i), x_1(i))$ at level i before the max-pooling operation, we merge the previous convolutional near layer's feature $x_0(i-1)$ and far layer's feature $x_1(i-1)$ into $x(i-1)$ first. Then we fed $x(i-1)$ into the max-pooling model $f(x(i-1); w)$ to decrease its size from $H \times W$ to $(H/2) \times$

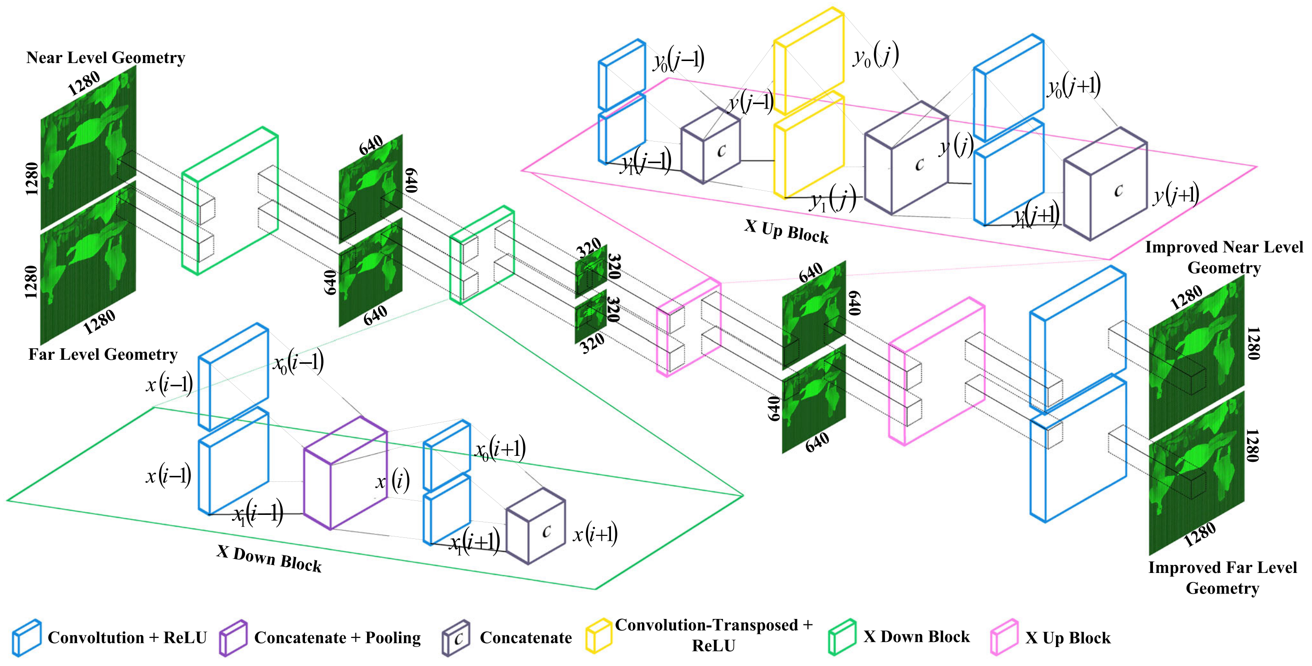


Fig. 7 XInteractNet architecture. In the X down block, to share the interactive information $I_d(x_0(i), x_1(i))$ at level i before the max-pooling operation, the previous convolutional near layer's feature $x_0(i-1)$ and far layer's feature $x_1(i-1)$ are merged into $x(i-1)$ first. The No. $i+1$ convolutional near and far layers take the pooled $x(i)$ feature as input to tackle the next convolution computation. Similarly, the X up block shares the interactive information $I_u(y_0(j), y_1(j))$ at level j before the transposed-convolution operation. The previous convolutional near layer's feature $y_0(j-1)$ and far layer's feature $y_1(j-1)$ are shared and merged into $y(j-1)$. The obtained $y_0(j)$ and $y_1(j)$ features are concatenated again and fed into the next NO. $j+1$ convolutional near and far layers for computation.

$(W/2)$ while aggregate more features from C channels to $2 \times C$ channels. The No. $i+1$ convolutional near and far layers take the pooled $x(i)$ feature as input to tackle the next convolution computation. We set the kernel size to 3×3, stride to 1, padding to 1 for all convolutional layers of X down block. Formally, the X interactive down block is able to be concluded as

$$\Psi^d\,(x_0(i), x_1(i)) = C(\Phi_0^d * x_0\,(i)\,, \Phi_1^d * x_1\,(i)) \tag{1}$$

where $x_0(i)$ and $x_1(i)$ are the near and far features input that stands on the No. i layer. Φ_0^d and Φ_1^d are denoted as the the near and far layers model parameters including corresponding weights and bias matrices, respectively. C denotes the concatenation function.

Similarly, the X up block shares the interactive information $I_u(y_0(j), y_1(j))$ at level j before the transposed-convolution operation. The previous convolutional near layer's feature $y_0(j-1)$ and far layer's feature $y_1(j-1)$ share interactions for each other and we merge them into $y(j-1)$. Afterward, the No. j transposed-convolutional layer takes it as input to recover its size from $(H/2) \times (W/2)$ back to $H \times W$ while decreasing the feature channels from $2C$ to C. Then we concatenate the achieved $y_0(j)$ and $y_1(j)$ features again and feed it into the next NO. $j+1$ convolutional near and far layers for computation. Table 2 shows the convolutional layer parameters of X up block. Mathematically, the X interactive up block is able to be represented by

$$\Psi^u\,(y_0(j), y_1(j)) = C(\Phi_0^u * y_0\,(j)\,, \Phi_1^u * y_1\,(j)) \tag{2}$$

where $y_0(j)$ and $y_1(j)$ are the near and far features input that stands on the No. j layer. Φ_0^u and Φ_1^u denote the the near and far layers model parameters including corresponding weights and bias matrices, respectively. C denote the concatenation function.

4.3 Interactive Loss Function

To properly fit the second step of XInteractNet, we devise a loss function called Interactive loss function to measure the XInteractNet. During the design process of the interactive loss function, we have two primary considerations.

First, as a dual inputs and dual outputs supervised network, XInteractNet should be capable of recovering the near and far depth fields to be close to their original ones on the pixel level, respectively. We, therefore, define the near Mean

Table 2 The convolutional and transposed convolutional layers parameters of the first X Up Block in Fig. 7

Layer	1	2	3
Level	Near conv	Near transposed conv	Near conv
	Far conv	Far transposed conv	Far conv
Kernel size	3×3	2×2	3×3
	3×3	2×2	3×3
Feature map	64	64	32
Number	64	64	32
Stride	1	2	1
	1	2	1
Padding	1	0	1
	1	0	1

Square Error (MSE):

$$L_0(\Theta_0) = \frac{1}{N}\sum_{i=1}^{N} ||\Upsilon_0(r_0^{''}(i)|\Theta_0) - g_0(i)||_2^2 \quad (3)$$

where Θ_0 encapsulates the whole near depth field parameter set of the XInteractNet, that contains weights and bias. $\Upsilon_0(r_0^{''}(i)|\Theta_0)$ is denoted as corresponding near depth field modules in the XInteractNet that output $r_0^{''}$. As explained in Sect. 4.2, $g_0(i)$ is the original near depth field, where i indexes each of them. $r_0^{'}(i)$ is the near depth field output of the XInteractNet. N is the number of frames. Similarly, the far MSE is defined as

$$L_1(\Theta_1) = \frac{1}{N}\sum_{i=1}^{N} ||\Upsilon_1(r_1^{''}(i)|\Theta_1) - g_1(i)||_2^2 \quad (4)$$

Hence, the sum of near MSE $L_0(\Theta_0)$ and far MSE $L_1(\Theta_1)$ can be defined as Dual MSE (DMSE)

$$L_D(\Theta_0, \Theta_1) = L_0(\Theta_0) + L_1(\Theta_1) \quad (5)$$

Second, as explained in Sect. 4.2 above, the XInteractNet aims to utilize the interactive information $I_s(r_0, r_1)$ between the near and far depth fields as much as possible. Relying on the interactive information, the XInteractNet should be able to handle two types of depth field problems well. Specifically, one non-occlusive case is that the original near depth field is the same as the original far depth field. V-PCC essentially converts the same 3-D point to the original near and far pixel with the same value. The XInteractNet should recognize this non-occlusive case and do its best to assimilate them with their $I_s(g_0, g_1)$. In this way, the XInteractNet can be helpful to reconstruct only one point in 3-D space correctly. In the other occlusive case, the correlation between near and far depth fields is quite weak, and this means that the near depth field is quite different from the far depth field. The XInteractNet should then learn from $I_s(g_0, g_1)$ to discriminate them and help reconstruct two different points in 3-D space. Based on the considerations above, we need to design a special term in the loss function that can recover the interactive status of $I_s(r_0, r_1)$ to be as close as possible to the original interactive status $I_s(g_0, g_1)$. We, therefore, devise an interactive term

$$L_{IT}(r_0^{''}, r_1^{''}, g_0, g_1) = \lambda L_{s1}(|r_0^{''} - r_1^{''}|, |g_0 - g_1|) \quad (6)$$

where L_{s1} denotes smooth L one loss function. $r_0^{''}$ and $r_1^{''}$ are the near and far depth fields outputs of XInteractNet while g_0 and g_1 are their corresponding origins. λ is a tuning coefficient and is set to 0.001 by default. This interactive term pushes the XInteractNet to learn from interactive information so that it could squeeze the difference between near and far reconstructions and the difference of their labels.

We finally propose the interactive loss function:

$$\begin{aligned} L_I((\Theta_0, \Theta_1); (r_0^{''}, r_1^{''}, g_0, g_1)) = L_D(\Theta_0, \Theta_1) + \\ L_{IT}(r_0^{''}, r_1^{''}, g_0, g_1) \end{aligned} \quad (7)$$

The interactive loss function $L_I((\Theta_0, \Theta_1); (r_0^{''}, r_1^{''}, g_0, g_1))$ effectively constrain the XInteractNet to learn from the interactive information $I_s(g_0, g_1)$ for correctly recognizing the correlation between the near and far depth fields.

4.4 Dataset

We adopt the Common Test Conditions(CTC) (Schwarz et al. 2018) consisting of dynamic point cloud sequences recommended by MPEG for training, validating, and testing. 8*i* captured and collected these raw 3-D point cloud sequences. For our two-step algorithm, to train and validate our near and far $S(\cdot)$ CNNs of step one and XInteractNet of step two models, we use Queen sequence. We test the proposed models with the other four sequences containing Loot, RedandBlack, Soldier, and Longdress, as shown in Sect. 5. On step one, V-PCC generates 250 near depth field and 250 far depth field origins and reconstructions of Queen. Among these data, for both near and far depth fields, we use 192 frames for training while 58 frames for validating. Then, we extract the 64×64 Coding Tree Units (CTU) from the luminance component of the generated near and far depth field of origins and reconstructions. Finally, we generate a total of 76, 800 near and far depth field frames for training, and 23, 200 frames for validating our $S(\cdot)$ CNNs. In step two, we use the V-PCC embedded step one models, instead of anchor V-PCC reference software, to generate both training and validating data. The other data preparation process is the same as the step one generation process. The amount of training or validating CTUs is the same as the one of step one as well.

Table 3 Proposed two-step method and geometry padding method (Graziosi and Tabatabai 2019) against the geometry smoothing method (Nakagami 2018) respectively on BD-rate and time complexity within the first 32 frames of sequences under all intra

Class	Sequence	V-PCC with geometry padding method (SOTA) (Graziosi and Tabatabai 2019)					Proposed two-step method				
		Geom.BD-totalrate		Attr.BD-totalrate			Geom.BD-totalrate		Attr.BD-totalrate		
		D1 ↓	D2 ↓	Luma ↓	Cb ↓	Cr ↓	D1 ↓	D2 ↓	Luma ↓	Cb ↓	Cr ↓
A	Loot	−4.4%	−9.9%	4.4%	4.8%	5.4%	−20.8%	−18.2%	−1.2%	−1.4%	−0.5%
	Redandblack	−0.8%	−7.5%	4.4%	5.9%	5.0%	−10.1%	−11.3%	−1.1%	−0.9%	−2.4%
	Soldier	−1.5%	−7.8%	4.4%	7.3%	6.4%	−13.8%	−12.8%	−1.9%	0.1%	−0.8%
B	Longdress	−1.3%	−8.2%	2.3%	3.5%	3.2%	−12.8%	−13.7%	−2.9%	−1.4%	−2.3%
	Class a	−2.2%	−8.4%	4.4%	6.0%	5.6%	−14.9%	−14.1%	−1.4%	−0.7%	−1.2%
	Class b	−1.3%	−8.2%	2.3%	3.5%	3.2%	−12.8%	−13.7%	−2.9%	−1.4%	−2.3%
Avg.	All	−2.0%	−8.3%	3.9%	5.4%	5.0%	−14.4%	−14.0%	−1.8%	−0.9%	−1.5%
	Enc.Self			102%					103%		
	Enc.Child			101%					101%		
	Dec.Self			102%					121%		
	Dec.Child			101%					212%		

4.5 Training

To train the first step $S(\cdot)$ CNNs, we feed the near and far depth field reconstruction CTUs obtained from V-PCC anchor into the near and far $S(\cdot)$ CNNs, respectively. The corresponding near and far origins generated from the anchor supervise the training procedure. During training the XInteractNet of the second step, we feed the generated reconstruction CTUs of near and far depth fields from step one $S(\cdot)$ CNNs into XInteractNet by batch-size of 16, respectively. The corresponding near and far origins generated from $S(\cdot)$ CNNs supervise the XInteractNet training procedure. Table 4 shows the parameters of the XInteractNet training process. Once the interactive loss is convergent, the training state is considered as completion. According to our experiments and observations, the loss is convergent before 60 epochs so we set the total XInteractNet training epochs to 60. Additionally, we set the base learning rate to $1e^{-4}$. We degrade the learning rate by multiplying γ of 0.1 after each interval of 50 epochs. In fact, γ just means the learning rate adjusting coefficient. We apply the Adaptive Moment Estimation (Adam) (Kingma and Ba 2014) algorithm as the gradient optimizer. We set Adam momentum to 0.9 and the weight decay to $1e^{-4}$. We use these hyper-parameters for the first step $S(\cdot)$ training as well.

Table 4 Training parameters

Parameters	Value
Base learning rate	$1e^{-4}$
γ adjusting coefficient	0.1
Adjusting epochs interval	50
Weight decay	$1e^{-4}$
Momentum	0.9
Total epochs	60

5 Experimental Results

To evaluate the performance of the proposed approaches, we implement our proposed two-step and one-step algorithms into the V-PCC and HEVC reference software. As explained in Sect. 4, the two-step method represents the proposed geometry artifacts removal approach in two stages, including step one with near and far CNNs and step two using our designed XInteractNet. The one-step method of near and far CNNs means it executes one stage utilizing the near CNN and far CNN for corresponding depth fields. The one-step method of mixed geometry is inputted with a mixture of near and far depth fields. In this section, we compare the V-PCC geometry smoothing method (Nakagami 2018), state-of-the-art(SOTA), namely geometry padding method (Graziosi and Tabatabai 2019), the one-step methods above, and the proposed two-step method. On the aspect of test data, we experiment within the first 32 frames of four V-PCC CTC (Schwarz et al. 2018) sequences, as mentioned in Sect. 4.5.

5.1 Comparison with SOTA Under All Intra

As Table 3 shown, we compare the proposed two-step method and SOTA geometry padding (Graziosi and Tabatabai 2019) against the V-PCC geometry smoothing (Nakagami 2018) under all intra case within the first 32 frames of four CTC sequences (Schwarz et al. 2018). Note that, to be fair, we set both the proposed two-step method and the other two V-PCC methods to 8 bits for the geometry encoder. We experiment with all these methods under four level bitrate settings. On point-to-plane D2 Geom.BD-TotalRate, we can see that the proposed two-step approach outperforms V-PCC geometry smoothing by an average of −14.0% while the SOTA geometry padding method gains −8.3% on this metric. In addition, the proposed two-step method surpasses SOTA and geometry smoothing methods in every sequence on point-to-plane D2. Specifically, the proposed two-step approach performs better than SOTA by −5.7% and −5.5% related to Class A and B on point-to-plane D2, respectively. Especially, the peak difference even reaches −8.3% on Loot of Class A.

Compared to the SOTA geometry padding method, the proposed two-step method gains an average of −12.4% on point-to-point D1 Geom.BD-TotalRate. We can see that the proposed two-step method outperforms SOTA and geometry smoothing methods in all sequences on point-to-point D1. Specifically, the proposed two-step method surpasses SOTA on Class A and B correspondingly by −12.7% and −11.5% at the point-to-point error D1. In addition, the top difference climbs to −16.4% on Loot of Class A.

Two reasons lead to these comparison results. Both geometry smoothing (Nakagami 2018) and geometry padding (Graziosi and Tabatabai 2019) do not exploit the strong correlations and interactions between the geometry near and far layers. In addition, the non-linear representation ability of CNN has not been considered in these two geometry distortion removal methods. All these above implicate that our proposed two-step mechanism and designed XInteractNet achieves a clear improvement on the problem of geometry artifacts removal. Besides, the design of XInteractNet effectively exploiting the strong correlations between near and far depth fields is beneficial for the enhancement of geometry.

As illustrated in Fig. 8, we count the number of points n_r of the reconstructed 3-D point cloud within the first 32 frames of four CTC sequences. From level $r1$ to level $r4$, the bitrate labeled on the Y axis gradually increases. Our proposed algorithm aims to remove the artifacts and restore the points that initially exist in ground truth as much as possible.

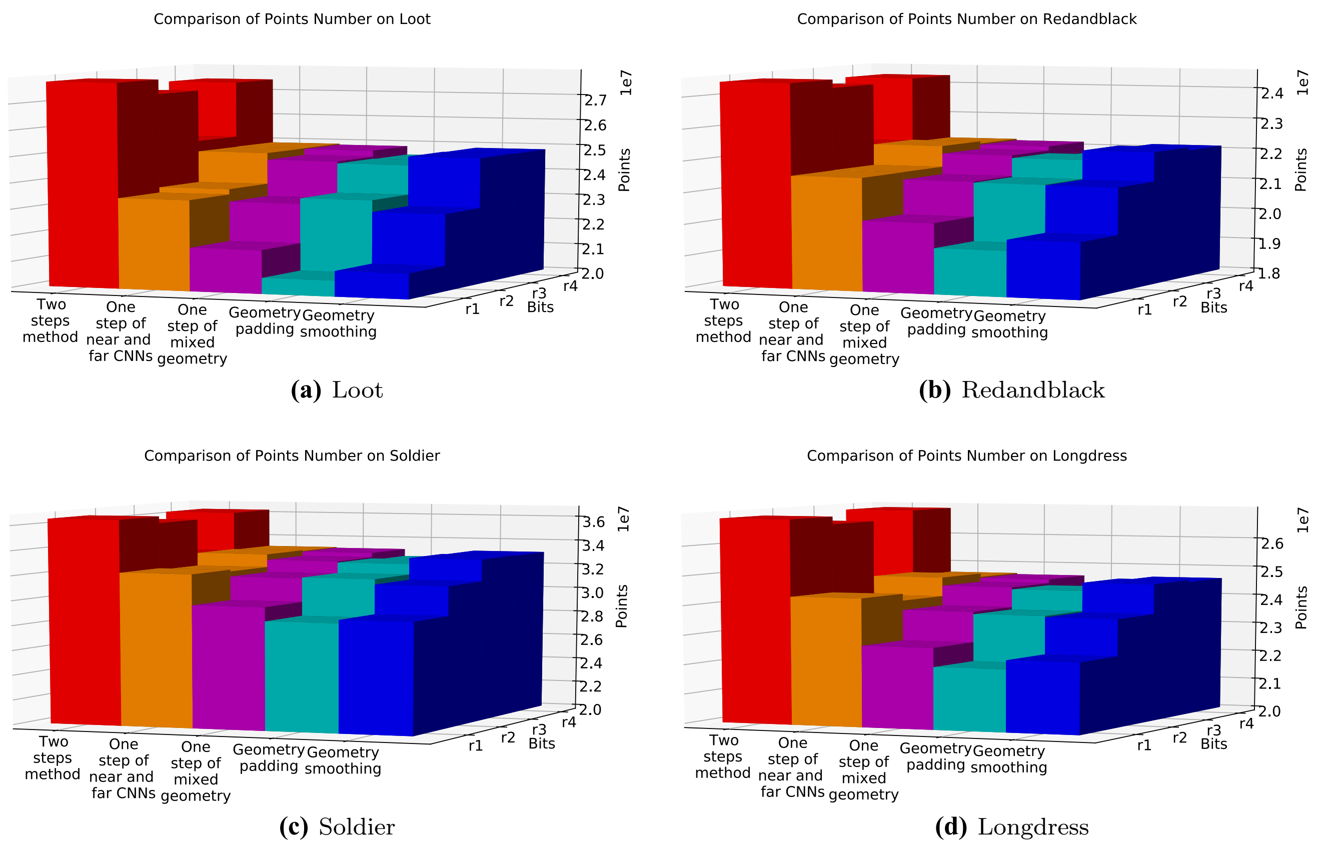


Fig. 8 Number of points N_R in reconstructed 3-D point clouds within first 32 frames of four CTC sequences. The bitrate of Y axis gradually increases from level $r1$ to level $r4$. The number of points N_R of our proposed two-step method clearly restore more points than V-PCC geometry pading and smoothing methods. These statistics fully demonstrate that the proposed two-step method effectively restores the 3-D point clouds.

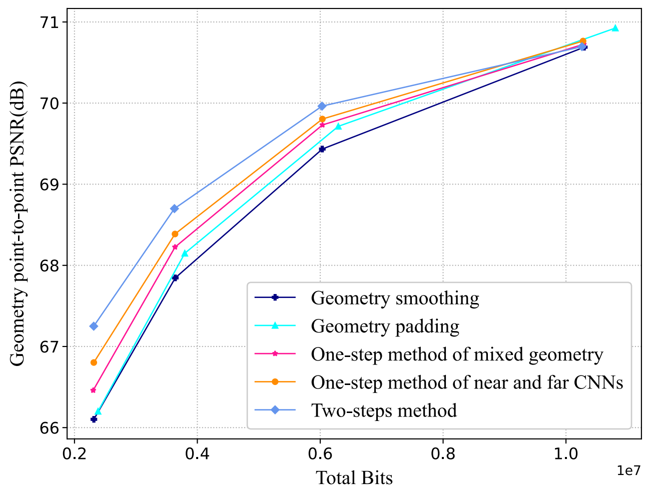
We can obviously see that the number of points n_r of our proposed two-step method is more than the one of SOTA, geometry smoothing. These statistics fully demonstrate that the proposed two-step methods efficiently restore the original points in the 3-D point-cloud reconstruction to improve its quality.

Figure 9 shows the comparison of Geometry Rate-Distortion (RD) curves in all the intra cases on Loot sequence. As shown, the point-to-plane D2 PSNRs of all 4 bitrate settings in the proposed two-step method is higher than the ones of the SOTA and geometry smoothing method. Similarly, the point-to-point D1 PSNRs of $r1$, $r2$, and $r3$ bitrate levels in the proposed two-step method are higher than those of the SOTA and geometry smoothing method as well. These results significantly prove that the proposed two-step method is superior to the SOTA and geometry smoothing on removing geometry artifacts and improving the PCC quality.

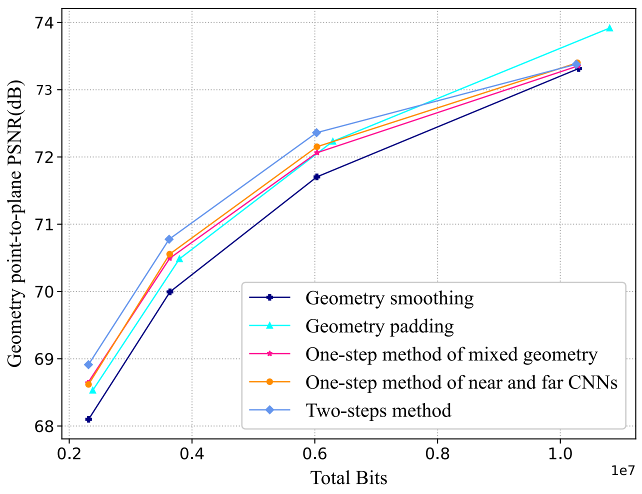
The time complexity (Li et al. 2019) is shown in Table 3 as well. For all methods, we configure the same environments. Specifically, the CPU configuration is Intel core i5-8400 CPU @ 2.80GHz, and the GPU configuration is GTX 1080ti. The ‘Enc.Self’ represents the encoder side of V-PCC, and ‘Enc.Child’ means the encoder side of HEVC. Similarly, the ‘Dec.Self’ represents the decoder side of V-PCC, and ‘Dec.Child’ means the decoder side of HEVC. Under the all intra case, on the encoder side of V-PCC, the proposed two-step method takes 103% time of geometry smoothing method. On the decoder side of V-PCC, it takes 121% time of geometry smoothing method. Their time complexities are similar.

5.2 Comparison with One-Step Methods

Table 5 shows the BD-rate and time complexity comparison between the proposed two-step method and the one-step method of near and far CNNs all against the one-step method inputted into mixed geometry within the first 32 frames of four CTC sequences. The proposed two-step method outperforms the one-step method of near and far CNNs by an average of −4.2% and −3.6% on point-to-point error D1 and point-to-plane D2, respectively. Meanwhile, the proposed two-step method performs better than the one-step



(a) Point-to-pointRD Curves on Loot



(b) Point-to-plane RD Curves on Loot

Fig. 9 Comparison of Geometry point-to-point and point-to-plane Rate-Distortion (RD) curves on Loot sequence in all the intra cases. As shown, the point-to-point and point-to-plane PSNRs of first three rate points in the proposed two-step method is higher than the ones of all other V-PCC geometry padding and smoothing methods. This proves that the proposed algorithm performs obviously better than SOTA methods on improving coding efficiency.

method inputted into mixed geometry −8.1% and −5.4% on point-to-point error D1 and point-to-plane error D2, respectively. These performances significantly demonstrate that the proposed two-step method outperforms one-step methods on objective qualities and PCC coding efficiency. These results also prove that XInteractNet in the second step explores the similarities between the near layer and far layer effectively. In addition, thanks to the architecture design with X similarity down and up blocks, XInteractNet could mine the interactions between the near layer and far layer in a full extent.

Regarding time complexity, the proposed two-step method takes almost the same time as the geometry smoothing method on the encoder side of V-PCC and HEVC. Meanwhile, on the decoder side of V-PCC and HEVC, it takes 118% and 113% time of geometry smoothing method, respectively.

As exhibited in Fig. 8, we compare the number of points n_r in the reconstructed 3-D point cloud between the proposed two-step method and one-step methods within the first 32 frames of four CTC sequences as well. We can see that the number of points n_r of the proposed two-step method is more than either the one of the one-step method of near and far CNNs or the one of the one-step method inputted into mix geometry in all sequences. These statistics fully prove that

Table 5 Proposed two-step method and one-step method of near and far CNNs against the one-step method of mixed geometry respectively on BD-rate and time complexity within the first 32 frames of sequences under the all intra case

Class	Sequence	One-step method of near and far CNNs					Proposed two-step method				
		Geom.BD-Totalrate		Attr.BD-Totalrate			Geom.BD-Totalrate		Attr.BD-Totalrate		
		D1 ↓	D2 ↓	Luma ↓	Cb ↓	Cr ↓	D1 ↓	D2 ↓	Luma ↓	Cb ↓	Cr ↓
A	Loot	−4.4%	−1.9%	0.0%	1.1%	0.6%	−12.2%	−8.0%	−0.3%	−0.1%	−1.0%
	Redandblack	−3.3%	−2.1%	−0.1%	−0.2%	−0.1%	−6.1%	−4.9%	−0.4%	−1.1%	−1.2%
	Soldier	−3.8%	−1.5%	−0.1%	0.4%	0.0%	−8.1%	−4.8%	−0.8%	−0.7%	−1.3%
b	Longdress	−4.2%	−1.8%	−0.2%	0.1%	0.2%	−5.9%	−3.9%	−1.2%	−1.0%	−1.1%
	Class A	−3.9%	−1.8%	−0.1%	0.4%	0.1%	−8.8%	−5.9%	−0.5%	−0.6%	−1.1%
	Class B	−4.2%	−1.8%	−0.2%	0.1%	0.2%	−5.9%	−3.9%	−1.2%	−1.0%	−1.1%
Avg.	All	−3.9%	−1.8%	−0.1%	0.3%	0.2%	−8.1%	−5.4%	−0.7%	−0.7%	−1.1%
	Enc.Self	100%					100%				
	Enc.Child	103%					103%				
	Dec.Self	108%					118%				
	Dec.Child	101%					113%				

Table 6 Proposed two-step method against the geometry smoothing (Nakagami 2018) on BD-rate and Time complexity under the random access case

Sequence	Geom.BD-TotalRate		Attr.BD-TotalRate		
	D1 ↓	D2 ↓	Luma ↓	Cb ↓	Cr ↓
A.Loot	−20.2%	−19.2%	−6.0%	−3.8%	−5.9%
A.Red&black	−10.3%	−11.5%	−2.0%	−1.2%	−2.2%
A.Soldier	−6.1%	−6.6%	−5.0%	−1.6%	−1.9%
B.Longdress	−12.1%	−14.1%	−3.9%	−2.4%	−2.8%
Class A	−12.2%	−12.4%	−4.3%	−2.2%	−3.3%
Class B	−12.1%	−14.1%	−3.9%	−2.4%	−2.8%
Avg. All	−12.2%	−12.8%	−4.2%	−2.3%	−3.2%
Enc.Self			103%		
Enc.Child			100%		
Dec.Self			119%		
Dec.Child			211%		

the proposed two-step method significantly performs better than other one-step methods on restoring points for artifact removal. Additionally, Fig. 9 shows the RD curves comparison between the proposed two-step method and the other one-step methods. The proposed two-step method leads the higher PSNRs of point-to-point error D1 and points to plane error D2 than other one-step methods.

5.3 Performances of the Proposed Two-Step Algorithm Under Random Access Case

As shown in Table 6, in the random access case, we can see that compared to the V-PCC geometry smoothing method, the proposed two-step method gains an average of −12.2% and −12.8% on Geom.BD-TotalRate point-to-point error D1 and point-to-plane error D2, respectively. Again, we can also see that the proposed two-step method leads the geometry smoothing method on point-to-point error D1 and point-to-plane error D2 in every class. Additionally, the top difference between the two-step method and geometry smoothing method climbs to −20.2% and −19.2% at Loot on point-to-point error D1 and point-to-plane D2, respectively. This comparison fully proves that the benefits brought by the proposed two-step methods can be similarly propagated to random access case.

5.4 Subjective Results

Figure 10 shows the visual comparisons of ground truth, point-cloud reconstructions of geometry smoothing, geometry padding (SOTA), and the proposed two-step method. These figures are from sequences of Loot, RedandBlack, and Longdress. We generate the figures (a), (b), (c), and (d) from the point cloud ground truth, the point-cloud reconstructions of geometry smoothing, geometry padding (SOTA), and the proposed two-step methods, respectively. From these sequences, we can clearly see from the zoomed red and green blocks that there are apparent distortions in the reconstructions of geometry smoothing, geometry padding. However, the reconstructions of the proposed two-step method exhibit better visual qualities. For instance, the green rectangles of Loot and Longdress and red ones of Redandblack and Longdress show significantly smoother boundary edges processed by the proposed two-step method. Meanwhile, the green rectangle of Redandblack and the red one of Loot exhibits obviously more abundant graph information as well. The subjective results demonstrate that compared to the geometry smoothing and geometry padding (SOTA) methods, the proposed two-step approach could significantly bring better visual qualities.

6 Conclusion

The geometry video intrinsically represents the depth fields of 3-D point clouds. Once there are artifacts on the compressed 2-D geometry video, they would be propagated to the 3-D point-cloud frames. In the lossy compression, there always exists a tradeoff between the rate of bitstream and distortion. This paper proposes a learning-based approach to remove the geometry artifacts and improve the compressing efficiency. We devise a two-step method working on the near and far depth fields decomposed from geometry. The first stage is learning-based Pseudo-Motion Compensation. The second stage exploits the potential of the strong correlations between near and far depth fields. We embed the proposed algorithm into the V-PCC reference software. To the best of our knowledge, this is the first learning-based solution of the geometry artifacts removal in V-PCC. The extensive experimental results show that the proposed approach achieves

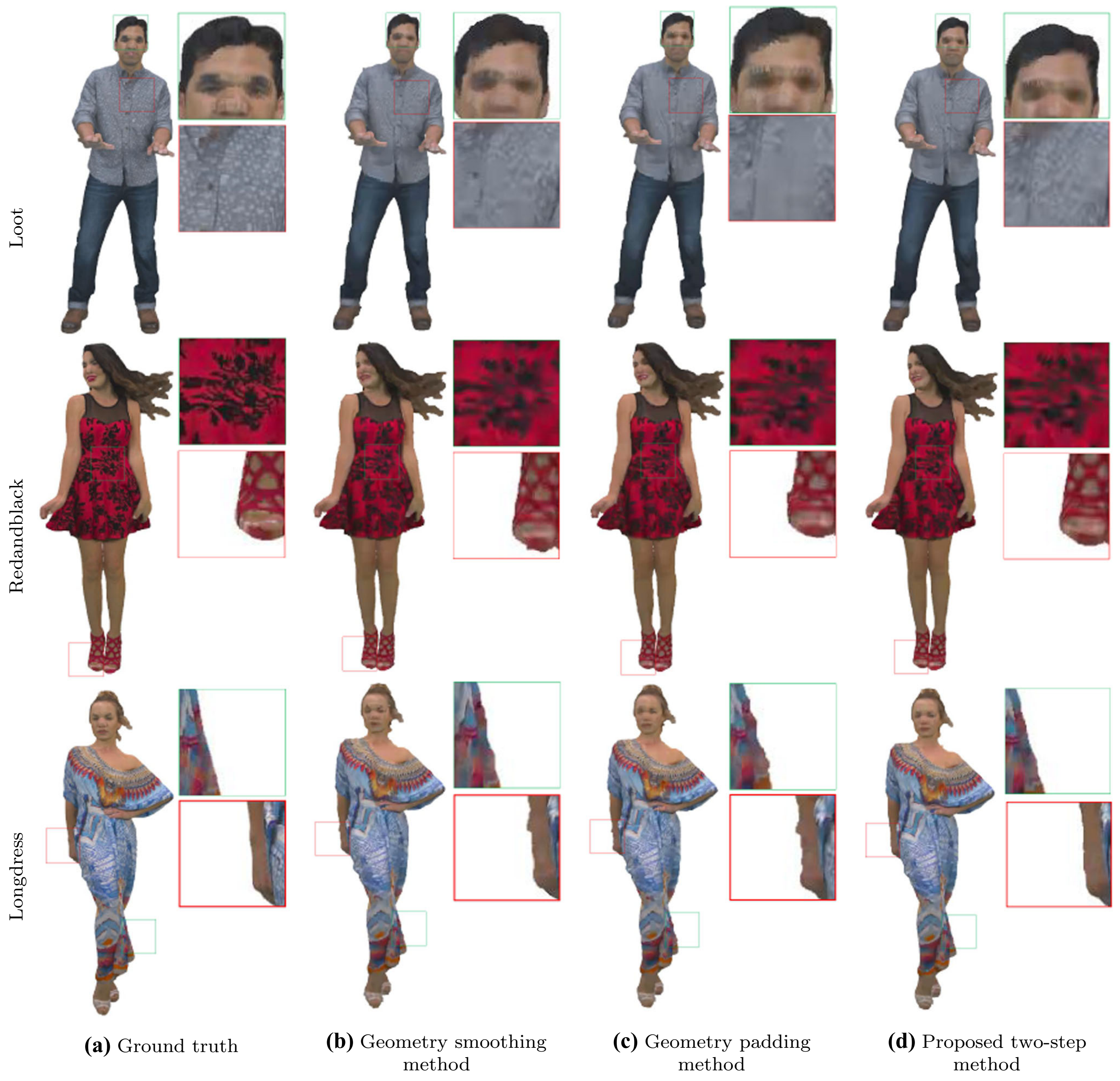

Loot
Redandblack
Longdress
(a) Ground truth
(b) Geometry smoothing method
(c) Geometry padding method
(d) Proposed two-step method

Fig. 10 Visual comparisons of ground truth, point-cloud reconstructions of geometry smoothing, padding and proposed two-step methods. The figures are derived from Loot, RedandBlack, and Longdress. From the three sequences, we can clearly see from the red and green rectangles that there are significant artifacts and noises in the reconstructions of V-PCC Geometry smoothing and padding methods, while the reconstructions of the proposed two-step method show a smoother effect. The visual results obviously demonstrate that compared to the SOTA, the proposed two-step method brings better subjective qualities.

significant gains on geometry artifacts removal and quality improvement of 3-D point-cloud reconstruction compared to state-of-the-art schemes. In the future, we can still exploit the temporal relationships of the 2-D geometry and 3-D point-cloud frames to assist the artifact removal further. The multi-frames-based method is another reasonable potential solution utilizing the learned features for motion compensation. In addition, we consider using sparse convolution for PCC, which is a promising direction as well.

References

Andrivon, P., Ricard, J., Guede, C., Nakagami, O., Graziosi, D., & Tabatabai, A. (2020). Patch border filtering specification in V-PCC. Document ISO/IEC JTC1/SC29/WG11 m51501, Geneva, CH.

Biswas, S., Liu, J., Wong, K., Wang, S., Urtasun, R. (2020). Muscle: Multi sweep compression of lidar using deep entropy models. In: Larochelle, H., Ranzato, M., Hadsell, R. Balcan, M.F. Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 22170–22181. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2020/file/fc152e73692bc3c934d248f639d9e963-Paper.pdf.

Bross, B., Chen, J., Liu, S. (2019). Versatile Video Coding (Draft 4). Document ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 JVET-M1001-v6, Marrakech, MA.

Bruder, G., Steinicke, F., Nüchter, A. (2014). Poster: Immersive point cloud virtual environments. In: 2014 IEEE symposium on 3D user interfaces (3DUI), pp. 161–162. IEEE.

Budagavi, M., Faramarzi, E., Ho, T., Najaf-Zadeh, H., Sinharoy, I. (2017). Samsungs response to cfp for point cloud compression (category 2). Document ISO/IEC JTC1/SC29/WG11 m41808, Macau, China.

Cai, K., & Ricard, J., C.G., Llach, J., Chevet, J.C. (2018). Geometry image coding improvements. Document ISO/IEC JTC1/SC29/WG11 m42111, Gwangju, Korea.

Chen, X., Ma, H., Wan, J., Li, B., Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1907–1915.

Chen, J., Lin, C., Hsu, P., & Chen, C. (2014). Point cloud encoding for 3d building model retrieval. *IEEE Transactions on Multimedia*, *16*(2), 337–345.

Choy, C., Gwak, J., Savarese, S. (2019). 4d spatio-temporal convnets: Minkowski convolutional neural networks. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 3075–3084.

Committee, M. (2020). V-PCC codec description. Document ISO/IEC JTC1/SC29/WG11 w19526, Italy.

Culture 3D cloud. http://c3dc.fr/.

Dawar, N., Najaf-Zadeh, H., Joshi, R., Budagavi, M. (2018). PCC TMC2 Interleaving in geometry and texture layers. Document ISO/IEC JTC1/SC29/WG11 m43723, Ljubljana, Slovenia.

de Queiroz, R. L., & Chou, P. A. (2017). Motion-compensated compression of dynamic voxelized point clouds. *IEEE Transactions on Image Processing*, *26*(8), 3886–3895.

d'Eon, E., Harrison, B., Myers, T., Chou, P. (2017). Input to ad hoc groups on mpeg point cloud compression and jpeg pleno. Document ISO/IEC JTC1/SC29/WG11 m40059, Geneva, Switzerland.

Fuchs, H., State, A., & Bazin, J. C. (2014). Immersive 3d telepresence. *Computer*, *47*(7), 46–52.

Gojcic, Z., Zhou, C., Wegner, J.D., Guibas, L.J., Birdal, T. (2020). Learning multiview 3d point cloud registration. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 1759–1769.

Graziosi, D., Tabatabai, A. (2019). [V-PCC] New contribution on geometry padding. Document ISO/IEC JTC1/SC29/WG11 m47496, Geneva, CH.

Guo, K., Xu, F., Yu, T., Liu, X., Dai, Q., & Liu, Y. (2017). Real-time geometry, albedo, and motion reconstruction using a single rgb-d camera. *ACM Transactions on Graphics (ToG)*, *36*(4), 1.

He, L., Zhu, W., Xu, Y. (2017). Best-effort projection based attribute compression for 3d point cloud. In: 2017 23rd Asia-Pacific conference on communications (APCC), pp. 1–6. IEEE.

Huang, T., Liu, Y. (2019). 3d point cloud geometry compression on deep learning. In: Proceedings of the 27th ACM international conference on multimedia, pp. 890–898.

Huang, L., Wang, S., Wong, K., Liu, J., Urtasun, R. (2020). Octsqueeze: Octree-structured entropy model for lidar compression. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 1313–1323.

Jang, E. S., Preda, M., Mammou, K., Tourapis, A. M., Kim, J., Graziosi, D. B., et al. (2019). Video-based point-cloud-compression standard in mpeg: from evidence collection to committee draft [standards in a nutshell]. *IEEE Signal Processing Magazine*, *36*(3), 118–123.

Kammerl, J., Blodow, N., Rusu, R.B., Gedikli, S., Beetz, M., Steinbach, E. (2012). Real-time compression of point cloud streams. In: 2012 IEEE international conference on robotics and automation, pp. 778–785. IEEE.

Kingma, D.P., Ba, J. (2014). Adam: A method for stochastic optimization. Preprint arXiv:1412.6980

Lasserre, S., Llach, J., Guede, C., Ricard, J. (2017). Technicolor's response to the cfpp for point cloud compression. Document ISO/IEC JTC1/SC29/WG11 m41822, Macau, China.

Li, Y., Liu, S., Kawamura, K. (2019). Methodology and reporting template for neural network coding tool testing. Document ITU-T SG 16 WP 3 and ISO/IEC JTC 1/SC 29/WG 11 JVET-M1006-v1, Marrakech, MA.

Li, L., Li, Z., Zakharchenko, V., Chen, J., & Li, H. (2019). Advanced 3d motion prediction for video-based dynamic point cloud compression. *IEEE Transactions on Image Processing*, *29*, 289–302.

Mammou, K., Tourapis, A.M., Singer, D., Su, Y. (2017). Video-based and hierarchical approaches point cloud compression. Document ISO/IEC JTC1/SC29/WG11 m41649, Macau, China.

Mekuria, R., Blom, K., & Cesar, P. (2016). Design, implementation, and evaluation of a point cloud codec for tele-immersive video. *IEEE Transactions on Circuits and Systems for Video Technology*, *27*(4), 828–842.

Mobile Mapping System. http://www.mitsubishielectric.com/bu/mms/index.html.

Nakagami, O. (2018). PCC TMC2 low complexity geometry smoothing. Document ISO/IEC JTC1/SC29/WG11 m43501, Ljubjana, SI.

Olivier, Y., Llach, J. (2018). Per patch projection optimization for TMC2. Document ISO/IEC JTC1/SC29/WG11 m43723, San Diego, CA, US.

Point Cloud Compression Category 2 Reference Software TMC2-8.0. http://mpegx.int-evry.fr/software/MPEG/PCC/TM/mpeg-pcc-tmc2.

Preda, M. (2017). Report on pcc cfp answers. Document ISO/IEC JTC1/SC29/WG11 w17251, Macau, China.

Qi, C.R., Su, H., Mo, K., Guibas, L.J.(2017) 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.

Quach, M., Valenzise, G., Dufaux, F.(2019). Learning convolutional transforms for lossy point cloud geometry compression. In: 2019 IEEE international conference on image processing (ICIP), pp. 4320–4324. IEEE.

Rhyu, S., Oh, Y., Woo, J.(2018). PCC CE2.13 report on texture and depth padding improvement. Document ISO/IEC JTC1/SC29/WG11 m43667, Ljubjana, SI.

Schwarz, S., Martin-Cocher, G., Flynn, D., Budagavi, M. (2018). Common test conditions for point cloud compression. Document ISO/IEC JTC1/SC29/WG11 w17766, Ljubljana, Slovenia.

Schwarz, S., Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P. A., et al. (2018). Emerging mpeg standards for point cloud compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *9*(1), 133–148.

Sportillo, D., Paljic, A., Boukhris, M., Fuchs, P., Ojeda, L., Roussarie, V.(2017). An immersive virtual reality system for semi-autonomous driving simulation: A comparison between realistic and 6-dof controller-based interaction. In: Proceedings of the 9th international conference on computer and automation engineering, pp. 6–10.

Sullivan, G. J., Ohm, J. R., Han, W. J., & Wiegand, T. (2012). Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on circuits and systems for video technology*, *22*(12), 1649–1668.

Sun, Y., Liu, M., & Meng, M. Q. H. (2017). Improving rgb-d slam in dynamic environments: A motion removal approach. *Robotics and Autonomous Systems*, *89*, 110–122.

Sun, X., Ma, H., Sun, Y., & Liu, M. (2019). A novel point cloud compression algorithm based on clustering. *IEEE Robotics and Automation Letters*, *4*(2), 2132–2139.

Thanou, D., Chou, P. A., & Frossard, P. (2016). Graph-based compression of dynamic 3d point cloud sequences. *IEEE Transactions on Image Processing*, *25*(4), 1765–1778.

Tian, D., Ochimizu, H., Feng, C., Cohen, R., Vetro, A. (2017). Geometric distortion metrics for point cloud compression. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 3460–3464. IEEE.

Tu, C., Takeuchi, E., Carballo, A., Takeda, K. (2019). Point cloud compression for 3d lidar sensor using recurrent neural network with residual blocks. In: 2019 International Conference on Robotics and Automation (ICRA), pp. 3274–3280. IEEE.

Tu, C., Takeuchi, E., Miyajima, C., Takeda, K. (2016). Compressing continuous point cloud data using image compression methods. In: 2016 IEEE 19th international conference on intelligent transportation systems (ITSC), pp. 1712–1719. IEEE.

Tulvan, C., Mekuria, R., Li, Z., Laserre, S. (2016). Use cases for point cloud compression (pcc).

Voulodimos, A., Doulamis, N., Doulamis, A., & Protopapadakis, E. (2018). Deep learning for computer vision: A brief review. *Computational Intelligence and Neuroscience,2018*.

Wiegand, T., Sullivan, G. J., Bjontegaard, G., & Luthra, A. (2003). Overview of the H. 264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *13*(7), 560–576.