Volumetric Video Compression Through Neural-based Representation

Yuang Shi, Ruoyu Zhao, Simone Gasparini, Géraldine Morin, Wei Tsang Ooi

Volumetric Video Compression Through Neural-based Representation

Yuang Shi
National University of Singapore
Singapore
yuangshi@u.nus.edu

Ruoyu Zhao
Tsinghua University
China
zhao-ry20@mails.tsinghua.edu.cn

Simone Gasparini
IRIT - University of Toulouse
France
simone.gasparini@toulouse-inp.fr

Géraldine Morin
IRIT - University of Toulouse
France
geraldine.morin@toulouse-inp.fr

Wei Tsang Ooi
National University of Singapore
Singapore
ooiwt@comp.nus.edu.sg

ABSTRACT

Volumetric video offers immersive exploration and interaction in 3D space, revolutionizing visual storytelling. Recently, Neural Radiance Fields (NeRF) have emerged as a powerful neural-based technique for generating high-fidelity images from 3D scenes. Building upon NeRF advancements, recent works have explored NeRF-based compression for static 3D scenes, in particular point cloud geometry. In this paper, we propose an end-to-end pipeline for volumetric video compression using neural-based representation. We represent 3D dynamic content as a sequence of NeRFs, converting the explicit representation to neural representation. Building on the insight of significant similarity between successive NeRFs, we propose to benefit from this temporal coherence: we encode the differences between consecutive NeRFs, achieving substantial bitrate reduction without noticeable quality loss. Experimental results demonstrate the superiority of our method for dynamic point cloud compression over geometry-based PCC codecs and comparable performance with state-of-the-art PCC codecs for high-bitrate volumetric videos. Moreover, our proposed compression based on NeRF generalizes to arbitrary dynamic 3D content.

CCS CONCEPTS

• **Information systems** → **Multimedia streaming**; • **Computing methodologies** → **Animation**.

KEYWORDS

Volumetric video; Point cloud compression; Neural radiance fields; Temporal coherence

1 INTRODUCTION

Volumetric video captures a 3D representation of a real-world scene or subject, allowing viewers to explore and interact with the captured content in six degrees of freedom (6DoF). Volumetric video is likely to play an increasingly important role in various industries, enabling new forms of visual storytelling and immersive experiences. In contrast to traditional 2D video, which has standard and mature forms of representation, 3D volumetric video has a plethora of representation formats. The representations of volumetric video can be categorized into explicit and implicit representations.

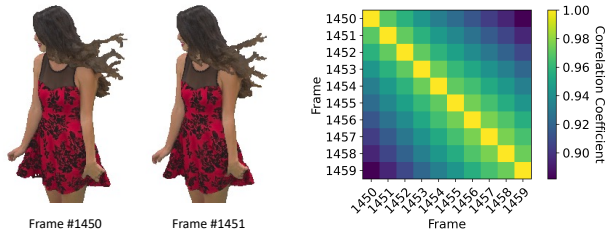
Most existing works are based on explicit 3D representations because they are easy to process through classical rendering pipelines.

(a) Point cloud representation. (b) NeRF reresentation.

Figure 1: Sample rendered images from a point cloud (left) showing visual artifacts due to its discrete nature which does not affect images rendered with NeRF (right).

Textured mesh is the most classical 3D model, but 3D point clouds, which consist of a set of 3D points with coordinates and color, have gained much popularity as the choice for high-quality representation for volumetric video as they are more adapted to dynamic acquisition. In order to provide a high-quality immersive experience with limited network conditions and computational resources, point cloud compression (PCC) techniques are paramount for volumetric video streaming. For example, Google's Draco [8], MPEG's video-based PCC (V-PCC), and MPEG's geometry-based PCC (G-PCC) [22] are three typical PCC codecs. Nevertheless, because of their discrete nature, point clouds can easily present visual artifacts that affect the visual quality [15]. For instance, point clouds may cause holes when being projected to screen space [16], which can be seen in Figure 1(a).

Given that explicit representations fail to achieve photo-realistic rendering quality, the latest advancements in implicit neural representations, especially neural radiance fields (NeRF) [18], have gained more popularity. NeRF [18] is a neural-based novel view synthesis technique. Given a set of 2D RGB images of a 3D scene, NeRF can model it as a neural radiance field with multilayer perceptrons (MLPs), and render immersive and high-fidelity novel views from this representation. Figure 1(b) shows an example of the rendered images from NeRF. Overall, NeRF has gained recognition as an effective approach [14, 27] for accurately representing the



(a) Neighboring frames. (b) Similarity matrix of NeRFs.

Figure 2: Example of temporal redundancy in (a) point clouds and (b) NeRFs.

dynamic interactions between light and color in three-dimensional space. Because of its ability to generate highly realistic images from 3D scenes, utilizing the recent advancements in NeRF for 3D content compression has become an attractive avenue. For instance, Bird *et al.* [3] adopt NeRF to represent 3D static scenes and apply an entropy penalty for model compression. Hu *et al.* [10] leverage NeRF to represent the geometry of 3D point clouds. Quantization and entropy encoding are then applied to compress neural networks, achieving comparable rate-distortion (R-D) performance with G-PCC. Although previous works make an effective step toward NeRF-based 3D content compression, there is still a big gap into practical volumetric video compression. The key challenge is to maintain the high-quality representation of volumetric videos while reducing the size of the representation itself [15].

In this paper, we present an end-to-end pipeline for volumetric video compression utilizing neural-based representation. We represent each frame of volumetric video as a NeRF, constructing a sequence of NeRFs. By representing volumetric video with neural networks, the problem of volumetric video compression becomes neural model compression.

Our work builds upon the key insight that there is significant similarity between successive NeRFs, which suggests the temporal redundancy in latent neural space. Figure 2(a) gives an example of temporal redundancy in explicit representation (*i.e.*, dynamic point clouds), where consecutive point cloud frames contain similar visual content. We find that such temporal redundancy still exists in latent neural space. For better illustration, we train ten NeRFs to represent ten consecutive point cloud frames and then measure their correlation. We present the similarity matrix of those neural representations in Figure 2(b). As shown, neighboring NeRFs share significantly high similarities, with over 0.98 correlation coefficient. Based on this observation, temporal compression is proposed for model compression. Specifically, instead of encoding each NeRF separately, we only encode the differences between consecutive NeRFs. This way, we achieve a significant reduction in bitrate without a noticeable loss of rendering quality. We apply an exponent-based non-uniform quantization scheme [5] to our temporal compression.

We propose an efficient, NeRFs-based representation for 3D dynamic scenes; the compression ratio benefits from the temporal coherence of the model. Here, we consider dynamic point cloud compression as a possible application scenario and thus compare

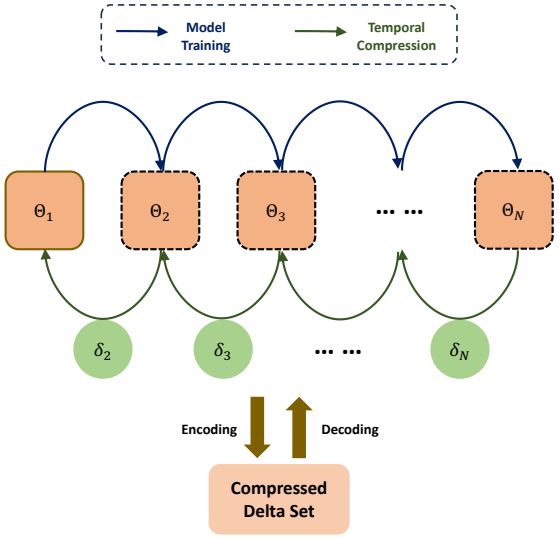


Figure 3: Overview of the proposed NeRF-based volumetric video compression.

the proposed method with state-of-the-art PCC codecs. We conduct extensive experiments on 8iVFBv2 and 8iVSLF Dataset [6, 13] with the original NeRF [18]. Experimental results demonstrate the superiority of our method compared with geometry-based PCC (*i.e.*, G-PCC and Draco). We also show that the proposed method can achieve comparable R-D performance w.r.t. V-PCC which is regarded as the state-of-the-art PCC codec, when dealing with high-bitrate volumetric videos.

The focus of our work lies in leveraging the high temporal coherence in neural models, independent of specific 3D representations and neural architectures. Specifically, our work is not limited to processing point clouds as the input source, but it can be applied to any dynamic content for which a sequence of images may be generated, and used as input of the NeRF sequence. Our approach inherently represents the scene using an implicit representation, thus allowing for the application of our NeRF model to a wide range of volumetric video-related tasks and scenarios. Meanwhile, it is worth highlighting that recent studies [4, 14] have shown that state-of-the-art NeRF variations, despite their improved rendering quality and speed, often sacrifice model size, hindering their suitability for streaming applications. Given our research's specific focus on the streaming context and the challenges associated with achieving a favorable rate-distortion trade-off, we chose to evaluate the original NeRF model as a baseline. Nonetheless, our methodology remains adaptable to incorporate other models, enabling further exploration of compression and trade-offs while considering specific application requirements.

2 NEURAL-BASED VOLUMETRIC VIDEO COMPRESSION

We represent the volumetric video with a sequence of NeRFs [18] and achieve volumetric video compression by compressing the

neural representations themselves. Figure 3 shows the overall architecture of our framework. Our framework consists of two key components: model training and temporal compression, which are elaborated in Section 2.1 and Section 2.2, respectively.

The system design stems from the insight that neighboring NeRF models share considerable similarities. In the model training stage, each NeRF is initialized based on the previous frame's NeRF, except for the first frame's NeRF, which is trained from the beginning as the starting point. This training strategy not only achieves significant time and resource savings but also further encourages temporal redundancy between subsequent NeRFs. Then, the temporal compression idea is applied to model compression. That is, for a sequence of trained NeRF models, we regard the first model as a "key-frame" or "I-frame" [2] and compress only the deltas between successive models. The models can be restored by accumulating the decoded deltas to the reference model.

2.1 NeRF-based Representation

Given a volumetric video with N frames, we can generate M views for each frame i, and construct a multi-view image set

$$\mathbf{D}_i = \left\{\left(V_m^i, X_m^i\right)\right\}_{m=1}^{M}, \tag{1}$$

where V_m^i is the camera pose and X_m^i is the corresponding image captured from this pose. A NeRF F_i is trained based on $\mathbf{D}_i$, and we simply use its weights $\mathbf{\Theta}_i$ to represent the model, to avoid cluttering the notation. Therefore, we represent a volumetric video with N frames as a NeRF sequence $\{\mathbf{\Theta}_i\}_{i=1}^{N}$.

We adopt a transfer-learning-like strategy for efficient model training. To be specific, each frame of the volumetric video is modeled by a NeRF initialized using the previous frame's NeRF. The NeRF representing the model at the starting time is trained from scratch and is then taken as a starting point for training subsequent times. By taking advantage of the learned knowledge from a previous model, considerable time and resources, which would have been required to train a model from scratch, can be saved. Meanwhile, such a training strategy further forces successive NeRFs to be closer and thus enhance the compression.

Formally, a NeRF $\mathbf{\Theta}_i$, which is trained to represent the i-th volumetric video frame, is initialized with the weights of its previous model $\mathbf{\Theta}_{i-1}$ and optimized by minimizing the distance from their renderings to the ground truth images:

$$\mathcal{L}_{\mathbf{\Theta}_i} = \sum_{m=1}^{M} \|\hat{X}_m^i - X_m^i\|_2^2, \tag{2}$$

where $\|\cdot\|_2^2$ is the Euclidean norm, $\hat{X}_m^i$ is the predicted image, and X_m^i is the ground truth.

2.2 Model Compression

In order to depict complex geometry and appearance, NeRF requires huge neural networks with billions of parameters, which poses a great challenge for the transmission with limited bandwidth. In this section, we introduce the proposed model compression techniques to considerably reduce the size of NeRF models while keeping good quality of the rendered images, to achieve good R-D performance.

Temporal Compression. To achieve efficient and scalable model compression, we propose to leverage high similarity between adjacent models and only encode and store the difference between them. Each model can be restored by applying the delta values to its previous model. Formally, we can represent a set of models $\{\mathbf{\Theta}_i\}_{i=1}^{N}$ as $\{\mathbf{\Theta}_1, \{\delta_i\}_{i=2}^{N}\}$, where $\mathbf{\Theta}_1$ is the first frame model and $\delta_i = \mathbf{\Theta}_i - \mathbf{\Theta}_{i-1}$ which is the delta values between $\mathbf{\Theta}_i$ and $\mathbf{\Theta}_{i-1}$. Our compression task is to compress the delta values while keeping good rendered image quality of the restored model $\hat{\mathbf{\Theta}}_i$, where $\hat{\mathbf{\Theta}}_i = \mathbf{\Theta}_1 + \sum_{t \le i} \hat{\delta}_t$ and $\hat{\delta}_t$ is decoded from the compressed δ_t.

An efficient and scalable model compression scheme, called LC-Checkpoint [5], is adopted in our proposed compression pipeline. The compression pipeline consists of two components. First, *exponent-based quantization* and then *priority promotion* are performed for lossy compression. The core idea of exponent-based quantization comes from the representation of floating points. Specifically, a floating point v is represented by $v = (-1)^s \times m \times 2^e$, where s is the sign, m is the mantissa, and e is the exponent. Exponent-based quantization partitions the floating-point numbers in δ_i into multiple buckets, based on their exponent e and sign s. Consequently, the elements with the same exponent and sign will be assigned to the same bucket. Then, the elements in each bucket are represented by the average of maximum and minimum values in the bucket. The number of buckets can be further limited with a priority promotion approach by keeping $2^{N_b} - 1$ buckets with larger exponent e only, where N_b is the number of bits for bucket indexing. The rest buckets are merged into one bucket, which is represented by 0. By doing so, only N_b bits are required to index buckets. Secondly, the quantized values are further compressed using *Huffman coding* [26]. We can trace the performance of the compressed model at different bitrates by changing the number of bits N_b to plot the R-D curves.

Neural Architecture Search. The ability of 3D representation of NeRF is determined by its model architecture. Generally speaking, with a larger number of parameters, NeRF can represent more complex detailed scenes. Meanwhile, as reported by previous works [11, 19, 29], not all parameters are crucial for accurate rendering and one can significantly reduce the model size with a limited impact on performance by properly adapting the number of network layers.

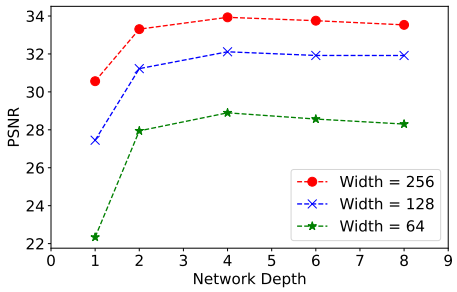


Figure 4: The performance of NeRFs with different neural architectures.

Therefore, inspired by such observation, we explore the effect of model architecture on the rendering performance of NeRF, and

analyze the R-D performance by tracking the change of model architecture. Specifically, we choose the network depth (number of layers) and the network width (number of neurons in each layer) as our neural architecture search space, which are set to $\{1, 2, 4, 6, 8\}$ and $\{64, 128, 256\}$, respectively. We first conducted experiments to narrow down the search space. We trained NeRFs with different combinations of network depth and width on a volumetric video frame (*i.e.* RedAndBlack Frame #1450) and measured the quality of images rendered from NeRFs using the peak signal-to-noise ratio (PSNR), as shown in Figure 4. Our crucial observation from Figure 4 is that the rendering quality of NeRF monotonically decreases when we reduce the network width, while with the decrease of the network depth, the rendering quality first improves and then drops when the network depth is smaller than 2. These results can be explained by previous works [9, 24, 30], which suggest that network depth provides the model with the ability to learn hierarchical representations, while width provides the model with the capacity to "memorize" the training data. Hence, we narrow down the search space of network depth and network width to $\{2, 4\}$ and $\{128, 256\}$, respectively, for the following evaluation.

3 EVALUATION

3.1 Experimental Settings

Dataset. We use four dynamic point cloud sequences for evaluation: RedAndBlack, Loot, Soldier, and Thaidancer. The first three sequences are from the 8iVFBv2 Dataset [6], and Thaidancer is from the 8iVSLF Dataset [13] which has a much greater number of points and higher bitrate. We select the first 30 frames of each sequence for the experiment. The average number of points per frame and corresponding bitrates (in Gbps) of the uncompressed volumetric videos are summarized in Table 1.

Table 1: Dynamic 3D Point Cloud Dataset

	RedAndBlack	Loot	Soldier	Thaidancer
Points ($\times 10^6$)	0.7	0.8	1.1	3.1
Bitrate (Gbps)	3.6	3.9	5.5	20.7

Rendering and Evaluation Parameters. Open3D [31] version 0.15.1 [1] is used for 2D rendering, where the width and height of the rendered images are 600 and 600, respectively. For NeRF training and testing, we generate 100 views as the training set and 200 views as the testing set for each frame of each point cloud sequence, where the camera settings are the same as [18]. Similarly, to evaluate the performance of other PCC codecs, we generate 200 rendered images for each decoded point cloud using the same camera settings of the testing set. PCC Arena [28] is used to measure the 2D quality of volumetric videos. PSNR and structural similarity index (SSIM) are used to quantify the 2D quality of rendered images. We calculate the average quality among the testing set for every frame.

Comparison Methods. Three PCC codecs are introduced:

i. *V-PCC* stores point cloud frames into 2D video frames and passes the 2D videos to 2D video codecs for compression. As defined in the V-PCC common test condition (CTC) [7], five compression rates controlled by the geometry and texture quantization parameter are used to generate the R-D curve.
ii. *G-PCC* utilizes an octree [17] or spatial data structures and applies arithmetical encoding to attributes. G-PCC quantizes the coordinates from floating-point numbers to integers with the parameter *positionQuantizationScale*. As defined in PCC Arena [28], eight compression rates, which are controlled by the quantization parameter, are used in our experiment.
iii. *Draco* adopts the K-D tree [1] data structure to compress point clouds. It employs quantization to reduce the number of bits, controlled by the quantization bit and compression level. The quantization bit determines the level of precision for the data. The compression level strikes a balance between the rate of compression and the computational complexity involved. According to PCC Arena [28], we use eight compression rates to track the R-D performance.

NeRF Settings. We keep the same settings for the NeRF model as [18], except for the model architecture. The Adam optimizer [12] is used for optimization. The learning rate begins at 5×10^{-4} and decays exponentially to 5×10^{-5}. ReLU [20] is used as the activation function. The first NeRF corresponding to the first frame is trained from scratch, with 300k iterations. The following NeRFs are initialized with the previous NeRF, and we find the optimization typically only takes 20k to converge.

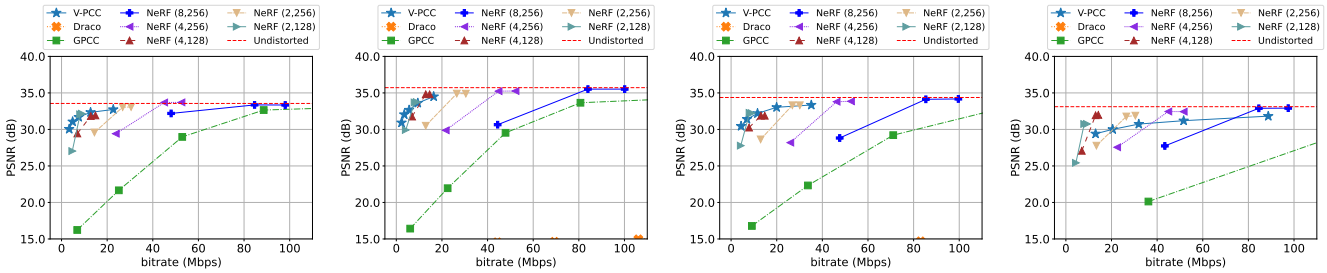
As discussed in Section 2.2, we compressed the NeRF-based representation by simplifying the neural architecture to further improve its performance. Hence, we train five NeRF sequences for every dynamic point cloud sequence by changing the model architecture. Specifically, according to the experiments, we narrowed down the search space of network depth d and width w to $\{2, 4\}$ and $\{128, 256\}$ so that four model architectures are considered. We additionally train the NeRF with default model architecture ($d = 8$, $w = 256$) for sanity check. We denote these five model settings as $NeRF(8, 256)$, $NeRF(4, 256)$, $NeRF(4, 128)$, $NeRF(2, 256)$, and $NeRF(2, 128)$.

Encoder Settings. As mentioned in Section 2.2, two trade-off parameters are utilized to balance between bitrate and distortion in the proposed method: (i) number of bits N_b, which determines the quantization level for temporal compression, and (ii) model architecture which controls the size of neural networks. Based on our observations, we found that the rendered quality of the compressed NeRF remained stable when the number of bits N_b exceeded 5, while dropping significantly below 3. This observation is reasonable since using less than $2^2 - 1$ buckets to store the weights can result in substantial distortion. Therefore, we limited the number of bits to a range of 3 to 5 to ensure a reasonable trade-off between bitrate and distortion. For five NeRF architectures, we perform temporal compression with different number of bits to trace the R-D performance, which gives us a total of five R-D curves for each point cloud sequence.

3.2 Experimental Results

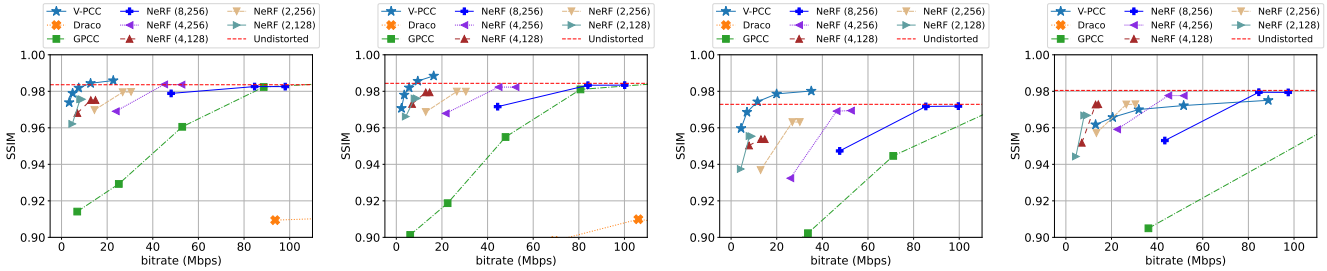
We report the R-D curves in Figure 5 and Figure 6 showing how the quality of the four point cloud sequences changes w.r.t. the

[1] https://github.com/isl-org/Open3D/releases/tag/v0.15.1



(a) RedAndBlack. (b) Loot. (c) Soldier. (d) Thaidancer.

Figure 5: R-D curves for PSNR: (a) RedAndBlack, (b) Loot, (c) Soldier, and (d) Thaidancer.



(a) RedAndBlack. (b) Loot. (c) Soldier. (d) Thaidancer.

Figure 6: R-D curves for SSIM: (a) RedAndBlack, (b) Loot, (c) Soldier, and (d) Thaidancer.

encoded bit-rate. To have better insights on how much quality is lost during the quantization process, we also plot the quality of rendered images from $NeRF(8, 256)$ without compression and denote it as Undistorted, which serves as the upper bound of visual quality. These figures specifically focus on bit-rates below 100 Mbps, so not all the R-D curves are displayed. By truncating the curves, we can have a clearer visualization of the R-D performance of the proposed method and thus make a better comparison with other PCC codecs. Notably, most of the points in R-D curves for Draco are not within the shown bit-rate range due to its significantly lower compression ratios. This indicates that Draco falls behind another geometry-based PCC, *i.e.* G-PCC, in terms of R-D performance. Consequently, we have chosen not to include Draco in the quantitative comparison to focus on the codecs that are more relevant and competitive in the displayed bit-rate range.

As observed in Figure 5 and Figure 6, the proposed method clearly outperforms G-PCC, always achieving better quality at the same bitrate for all the point cloud sequences. Specifically, our method outperforms G-PCC by at most 15.92 dB in PSNR and 6.15% in SSIM on RedAndBlack, 17.33 dB in PSNR and 7.48% in SSIM on Loot, 15.45 dB in PSNR and 7.41% in SSIM on Soldier, and 11.76 dB in PSNR and 7.94% in SSIM on Thaidancer. Furthermore, the R-D curves of our method demonstrate that, by utilizing the proposed temporal compression technique, the NeRF size can be effectively reduced with only a minor increase in rendering distortion. This finding highlights the effectiveness of our method in achieving substantial compression gains while maintaining acceptable quality.

Particularly, in the case of high-bitrate sequences, *i.e.* Thaidancer with a bitrate exceeding 20 Gbps, our method achieves better R-D performance compared to V-PCC, with up to 2.59 dB improvement in PSNR and 1.11% improvement in SSIM. The advantages of neural-based representation in this context are well-founded. Neural networks empower NeRF with the capability to capture fine-grained details by learning a compact implicit representation. Unlike explicit representations like point clouds, the size of the implicit representation in NeRF is determined by the neural networks and is not directly proportional to the complexity of geometry and attribute of the volumetric video. This decoupling allows for more efficient storage and transmission of the video data. In contrast, as an explicit representation, point cloud requires larger data size and potentially higher bitrate requirements.

We also employ the Soldier sequence as an example, presenting the compression ratio vs. SSIM in Figure 7. The compression ratio denotes the ratio of the compressed model's bitrate to that of the uncompressed baseline model ($NeRF(8, 256)$, labeled as Undistorted). Notably, our method achieves a wide range of compression ratios, spanning from 35 to 442, while exhibiting minimal degradation in quality, with quality drop ranging from 0.1% to 3.6% in SSIM. These compelling results underscore the exceptional performance of our proposed method in effectively balancing efficient compression and preservation of visual quality.

Besides the quantitative analysis above, we also show the sample rendered images of RedAndBlack and Thaidancer using compressed NeRF models under different bitrates in Figure 8, for qualitative

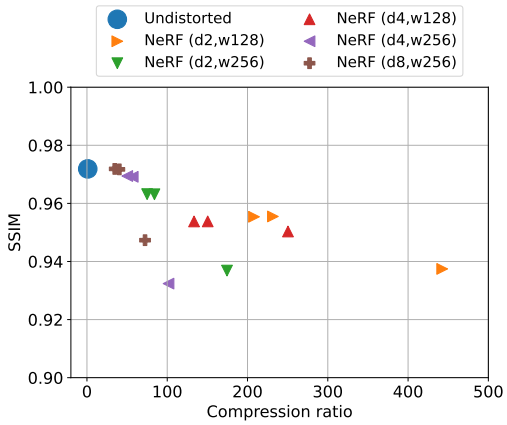


Figure 7: Compression ratio vs. SSIM for Soldier. Each NeRF architecture has three points corresponding to three different N_b: 3, 4, and 5.

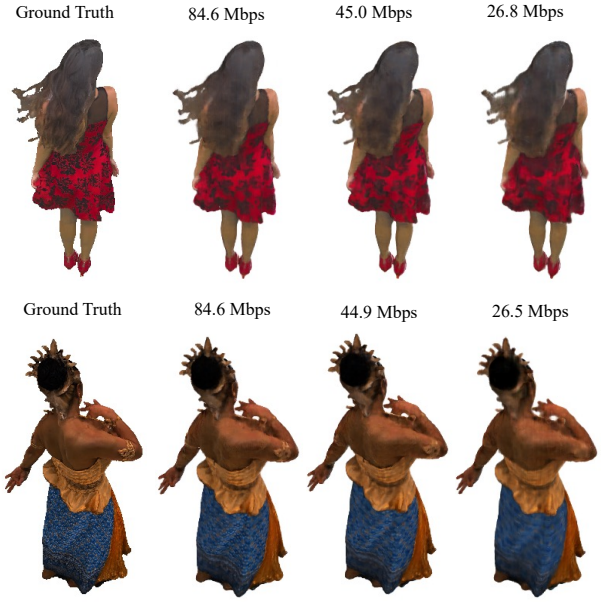


Figure 8: Sample images of RedAndBlack and Thaidancer using the compressed NeRF models at different bitrates.

analysis. The figure shows a view of Frame #1454 of RedAndBlack and Frame #6487 of Thaidancer. As can be found, the proposed method well restores the details of cloth texture. Moreover, the visual quality of the renderings remains relatively intact even when compressing the NeRF model from 84 Mbps to 26 Mbps, which suggests that our method can provide similar levels of detail and visual quality even at low bitrate.

In summary, the objective results reported in Figures 5, 6, and 7, and the sample rendered images from compressed NeRFs shown in Figure 8 demonstrate that our proposed method achieves high compression for volumetric video with minimal loss of detail.

4 CONCLUSION AND DISCUSSION

In this paper, we introduce an extendable and general pipeline for compressing volumetric video using a neural-based representation, which leverages the similarities between consecutive NeRFs and exploits temporal coherence and neural architecture to achieve effective and efficient model compression. We primarily tested our method on point cloud compression. Through experimental evaluations, we demonstrate the superiority of our method compared to geometry-based PCC codecs. Moreover, our approach achieves comparable results with state-of-the-art PCC codecs for high-bitrate volumetric videos. However, it is important to note that our proposed approach has wider applicability and is not restricted to point clouds as the sole input source. The inherent nature of our method, which models the scene using an implicit representation, enables its usage with any volumetric video data. The advantage of NeRF representation of offering a joint, realistic geometry and appearance model holds for our proposed solution. Therefore, our NeRF-based compression framework applies to any dynamic 3D content that may be rendered, expanding its compression performance to potential applications including a diverse range of tasks and scenarios.

There are several directions for future research based on the limitations and opportunities identified in our work. Firstly, although the PSNR and SSIM results offer valuable insights into the visual quality, conducting user studies would provide a more comprehensive understanding of the effectiveness of our method. Secondly, our current approach models volumetric videos frame by frame, which essentially represents the scene as a set of static NeRFs. Recent research efforts [21, 23, 25] have extended static NeRF to dynamic NeRF, enabling the representation of dynamic scenes with a single model. However, in the context of NeRF-based volumetric video streaming, modeling a dynamic scene with one model can make rate and viewport adaptation impractical [15]. One potential solution could be to split the video into several equal-size groups of frames (GOF) and train dynamic NeRFs for each group. Then, exploring the temporal redundancy among consecutive dynamic NeRFs and the relationship between the size of GOF and the level of temporal redundancy would be an interesting avenue for future research.

ACKNOWLEDGMENTS

This work was supported by the Singapore Ministry of Education Academic Research Fund Tier 1 (T1 251RES2038). The authors kindly thank Axel Carlier for his initial inspirational idea.

REFERENCES

[1] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, Sept 1975.

[2] Vasudev Bhaskaran and Konstantinos Konstantinides. *Image and video compression standards: algorithms and architectures*. Springer Science & Business Media, 1997.

[3] Thomas Bird, Johannes Ballé, Saurabh Singh, and Philip A. Chou. 3D scene compression through entropy penalized neural representation functions. In *Proceedings of the 2021 Picture Coding Symposium, PCS 2021, Bristol, United Kingdom, June 29 - July 2, 2021*, pages 1–5. IEEE, 2021.

[4] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. TensoRF: Tensorial radiance fields. In *Proceesdings of the 17th European Conference on Computer Vision, ECCV 2022, Tel Aviv, Israel, October 23-27, 2022,*, volume 13692,

pages 333–350. Springer, 2022.

[5] Yu Chen, Zhenming Liu, Bin Ren, and Xin Jin. On efficient constructions of checkpoints. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119, pages 1627–1636. PMLR, 2020.

[6] 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*, 7(8):11, 2017.

[7] Eugene d'Eon, Bob Harrison, Taos Myers, and Philip A Chou. Common test conditions for point cloud compression. *ISO/IEC JTC1/SC29/WG11 w17766*, 2018.

[8] Google. Draco 3D data compression. https://github.com/google/draco. Accessed: 2023-11-28.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, , Las Vegas, NV, USA, June 27-30, 2016*, pages 770–778. IEEE Computer Society, 2016.

[10] Yueyu Hu and Yao Wang. Learning neural volumetric field for point cloud geometry compression. In *Proceedings of the 2022 Picture Coding Symposium, PCS 2022, San Jose, CA, USA, December 7-9, 2022*, pages 127–131. IEEE, 2022.

[11] Yongdong Huang, Yuanzhan Li, Xulong Cao, Siyu Zhang, Shen Cai, Ting Lu, Jie Wang, and Yuqi Liu. An efficient end-to-end 3D voxel reconstruction based on neural architecture search. In *Proceedings of the 26th International Conference on Pattern Recognition, ICPR 2022, Montreal, QC, Canada, August 21-25, 2022*, pages 3801–3807. IEEE, 2022.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

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

[14] Junhua Liu, Yuanyuan Wang, Yan Wang, Yufeng Wang, Shuguang Cui, and Fangxin Wang. Mobile volumetric video streaming system through implicit neural representation. In *Proceedings of the 2023 Workshop on Emerging Multimedia Systems, EMS 2023, New York, NY, USA, 10 September 2023*, pages 1–7. ACM, 2023.

[15] Kaiyan Liu, Ruizhi Cheng, Nan Wu, and Bo Han. Toward next-generation volumetric video streaming with neural-based content representations. In *Proceedings of the 1st ACM Workshop on Mobile Immersive Computing, Networking, and Systems, ImmerCom 2023, Madrid, Spain, 6 October 2023*, pages 199–207. ACM, 2023.

[16] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhöfer, Yaser Sheikh, and Jason M. Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Transactions on Graphics*, 40(4):59:1–59:13, Jul 2021.

[17] Donald Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129–147, Jun 1982.

[18] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, Dec 2021.

[19] Saeejith Nair, Yuhao Chen, Mohammad Javad Shafiee, and Alexander Wong. NAS-NeRF: Generative neural architecture search for neural radiance fields. *CoRR*, abs/2309.14293, 2023.

[20] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning, ICML 2010, June 21-24, 2010, Haifa, Israel*, pages 807–814. Omnipress, 2010.

[21] Sida Peng, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Representing volumetric videos as dynamic MLP maps. In *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 4252–4262. IEEE, 2023.

[22] Sebastian Schwarz, Marius Preda, Vittorio Baroncini, Madhukar Budagavi, Pablo César, Philip A. Chou, Robert A. Cohen, Maja Krivokuca, Sebastien Lasserre, Zhu Li, Joan Llach, Khaled Mammou, Rufael Mekuria, Ohji Nakagami, Ernestasia Siahaan, Ali J. Tabatabai, Alexis M. Tourapis, and Vladyslav Zakharchenko. Emerging MPEG standards for point cloud compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(1):133–148, Mar 2019.

[23] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4D: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *Proceedings of the 2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 16632–16642. IEEE, 2023.

[24] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[25] Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. NeRFPlayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742, May 2023.

[26] Jan van Leeuwen. On the construction of huffman trees. In *Proceedings of the 3rd International Colloquium on Automata, Languages and Programming, ICALP 1976, University of Edinburgh, UK, July 20-23, 1976*, pages 382–410. Edinburgh University Press, 1976.

[27] Hanchen Wang, Qi Liu, Xiangyu Yue, Joan Lasenby, and Matt J. Kusner. Unsupervised point cloud pre-training via occlusion completion. In *Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9762–9772. IEEE, 2021.

[28] Cheng-Hao Wu, Chih-Fan Hsu, Tzu-Kuan Hung, Carsten Griwodz, Wei Tsang Ooi, and Cheng-Hsin Hsu. Quantitative comparison of point cloud compression algorithms with PCC arena. *IEEE Transactions on Multimedia*, 25:3073–3088, Feb 2023.

[29] Guo-Wei Yang, Wen-Yang Zhou, Hao-Yang Peng, Dun Liang, Tai-Jiang Mu, and Shi-Min Hu. Recursive-NeRF: An efficient and dynamically growing nerf. *IEEE Transactions on Visualization and Computer Graphics*, 29(12):5124–5136, Dec 2023.

[30] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *Proceedings of the 2016 British Machine Vision Conference, BMVC 2016, York, UK, September 19-22, 2016*. BMVA Press, 2016.

[31] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing, 2018.