Towards Full-scene Volumetric Video Streaming via Spatially Layered Representation and NeRF Generation

Jianxin Shi[1,2], Miao Zhang[2], Linfeng Shen[2], Jiangchuan Liu[2,*], Yuan Zhang[3], Lingjun Pu[1,*], Jingdong Xu[1]
[1]DISSec, ISN, CS, Nankai University, [2]Simon Fraser University, [3]Communication University of China

ABSTRACT

Immersive full-scene volumetric video (VV) showcases the richness and detail of the 3D world, yet poses significant streaming challenges given its massive data volume. Existing 3D tile-based viewport approaches struggle to effectively adapt to full-scene VV owing to their small video buffer limitation, high tile segmentation overhead, and lack of full-scene consideration.

In response, by exploiting spatially independent attributes of elements in VV, we present V^{2}NeRF, a novel full-scene VV streaming system featured by layered representation. It harmonizes the implicit neural radiance field (NeRF) with explicit point clouds to represent the static background and dynamic foreground, thereby avoiding large data transfer. Moreover, we propose a lightweight non-visible background removal method and a two-stage decoupled architecture to address the issues of intensive computation requirements and multiscale adaptation scheduling. An efficient buffer-aware simulated annealing algorithm is developed, alongside the utilization of a perceptually-learned metric, to enhance user experience. Extensive prototype evaluations demonstrate V^{2}NeRF's superior streaming and viewing performance.

CCS CONCEPTS

• **Information systems → Multimedia streaming**.

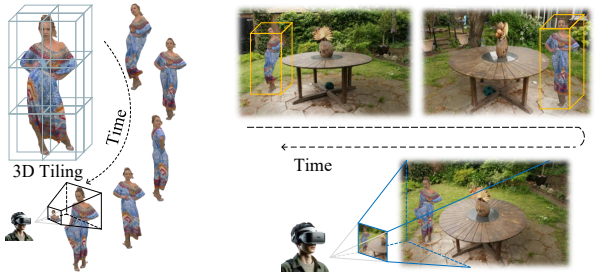
KEYWORDS

Full-scene volumetric video, Spatially layered representation, NeRF generation, Explicit point cloud

ACM Reference Format:
Jianxin Shi, Miao Zhang, Linfeng Shen, Jiangchuan Liu, Yuan Zhang, Lingjun Pu, Jingdong Xu. 2024. Towards Full-scene Volumetric Video Streaming via Spatially Layered Representation and NeRF Generation. In *The 34th edition of the Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV '24), April 15–18, 2024, Bari, Italy*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3651863.3651879

1 INTRODUCTION

Volumetric video (VV) captures content in 3D, enabling a six degrees-of-freedom (6DoF) motion for viewers with immersive experience.



(a) Object-based video. (b) Full-scene volumetric video.

Figure 1: Illustrations of the object-based and full-scene VV.

It has been suggested as the future media for a broad spectrum of applications from education and entertainment, to the business and manufacturing industry [21, 31]. Since the VV content is to be modeled with dense *explicit 3D data*, e.g., point cloud (PtCl), streaming VV over the Internet incurs huge bandwidth consumption. There have been studies leveraging tile-based viewport, occlusion, and distance visibility to reduce data usage [9, 27]. Most of them focus on streaming individual objects, e.g., a single person.

A full-scene VV could be re-constructed by the composition of individually streamed objects [12, 29], as depicted in Fig. 1. It is known, from the earlier experience of MPEG-4 object-based coding and streaming [13], that synchronization, visual compensation, and compatibility are critical issues in the reception and composition, even just for 2D video [32]. Given the much smaller video buffer size in VV (around 500ms, due to advanced motion prediction [8, 9]) and the high tile segmentation overhead (e.g., 1.6× the original video size [9]), a simple implementation of separate objects often lead to unpleasant viewing experiences like motion sickness and frequent stalls in a re-constructed full-scene VV.

In this work, we carefully examine full-scene VV streaming and arrive at two critical observations: (O1) The spatial features and independent attributes of elements can naturally facilitate the separation of content into the static background and dynamic foreground containing the moving objects; (O2) The static background content also needs to be updated as the viewport changes, yet delivering the entire static scene to the client in real time, all at once, is nearly impossible due to the sheer amount of 3D data.

As such, we suggest enabling the full-scene VV streaming by seamlessly fusing the latest advancements in *implicit neural radiance field* (NeRF) [23, 24] with explicit 3D data for background and foreground respectively. NeRF generates photorealistic rendering in representing the intricate and detailed static 3D scene through a lightweight deep model (e.g., compressed at MB level [28]) from any view. Recent exploratory studies [2, 18, 20] also consider NeRF as

*Corresponding authors (jcliu@sfu.ca, pulingjun@nankai.edu.cn).

NOSSDAV '24, April 15–18, 2024, Bari, Italy

ACM ISBN 979-8-4007-0613-4/24/04…$15.00
https://doi.org/10.1145/3651863.3651879

an attractive option. To incorporate it into full-scene VV streaming, however, we need to address the following non-trivial challenges:

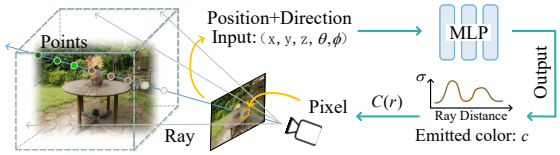
(C1) Intensive computation requirements. Explicit 3D data and implicit NeRF generation typically involve complex computations (e.g., require hardware acceleration) incurred by perspective projection [9, 30] and deep neural network [29, 38], making them highly resource-intensive and time-consuming. Meanwhile, foreground projection with various contents would compete with NeRF generation for the limited time budget (i.e., frame duration).

(C2) Multiscale adaptation scheduling. Foreground content is encoded based on the group of pictures (GoP) for better compression, while NeRF generation operates at the frame level. Ubiquitous network fluctuations and time-varying computing resources/requirements necessitate an adaptation scheme to ensure smooth playback [37]. Unfortunately, there is a lack of related research to support such coupled multiscale scheduling.

Solution. In our V^{2}NeRF system design, we first propose the *non-visible background removal scheme (C1).* The moving foreground objects always block the background content, as depicted in Fig 1(b). Thus, the fundamental idea is to avoid the NeRF inference of occluded parts. To this end, we devise a lightweight non-visible pixel skipping approach by modifying the ray transmittance of pixels. With the removal scheme, increased foreground content results in a high occlusion ratio and thus less NeRF computation. Conversely, in situations with reduced foreground, the PtCl projection is reduced, despite the associated lower occlusion, leaving more computing time for NeRF generation. This effectively mitigates the computation competition. Second, *two-stage optimization architecture (C2)* is developed where multiscale scheduling is decoupled into fine-grained computing adaptation by adjusting the NeRF-generated resolution, and coarse-grained network adaptation by controlling the PtCl density level. The consistency constraint and soft oversupply penalty are proposed to maintain the connection between the network and computing, thereby increasing resource utilization. An efficient, lightweight buffer-aware simulated annealing algorithm is developed, and the popular quality metric learned-perceptual-image-patch-similarity (LPIPS) [39] is utilized to maximize the quality of experience (QoE).

Our system prototype achieves full-scene VV streaming over varied broadband and 5G network traces. Moreover, extensive evaluations and ablation studies validate its effectiveness and demonstrate its robust performance. The results show that V^{2}NeRF improves perceptual quality by 24%, reduces rebuffering time by 83%, and increases user experience by 54% on average over benchmarks. The main contributions are summarized as:

- To the best of our knowledge, we are the first to enable free-viewpoint, photorealistic, full-scene VV streaming.
- We present V^{2}NeRF, which integrates implicit NeRF generation and explicit 3D PtCl to realize a layered full-scene VV representation.
- We develop a lightweight non-visible background removal method and a two-stage architecture to optimize the intensive computation and multiscale adaptation.
- We implement a prototype of V^{2}NeRF and verify its excellent performance on a wide variety of network conditions.



(a) Implicit neural radiance field generation.

Figure 2: An overview of neural radiance field representation.

2 BACKGROUND AND MOTIVATION

Exploration: explicit volumetric video. PtCls and polygon meshes are the primary explicit representations of VVs. In the context of full-scene VVs, PtCl coding is better suited and offers better flexibility and scalability, as its geometry information is unordered, unstructured, and unconnected [7, 37]. While compression methods (e.g., Draco [5] and G-PCC [6]) have been well-studied, the size of full-scene frame still reaches gigabytes level [11, 15]. Full-scene VV content can be captured in many ways, such as acquired by RGB-D cameras (e.g., Microsoft Azure Kinect) [12], or synthesized from 3D models (e.g., characters, objects, and environments) [34]. There is thus a key *insight* that 3D contents in a VV can be spatially separable based on its inherent spatial attributes [29, 38].

Motivation: implicit neural radiance field. Recent advances in NeRF [23, 24] have proven its huge potential in 3D representation. Given user motion, it can generate a photorealistic view, effectively avoiding large 3D data transmission. As shown in Fig. 2, NeRF models a 3D scene as a continuous volumetric field function parameterized by a multilayer perceptron (MLP), which it maps a 3D position $\mathbf{x} = (x, y, z)$ and a viewing direction $\mathbf{d} = (\theta, \phi)$ to a volume density σ and emitted color $\mathbf{c}$. To generate a single pixel color in user view, it first computes the ray corresponding to pixel $\mathbf{r}$, then evaluates the NeRF at a series of points $\{t_i\}$ along the ray. The outputs σ_i and $\mathbf{c}_i$ at each point are composited together into a single color value $\mathbf{C}(\mathbf{r})$ (i.e., pixel color) using the volume rendering. Yet, NeRF is mainly suitable for nearly static content. Vanilla NeRF fails to enable VV streaming, since its inherent limitations in handling large-scale movements and long duration [2, 23, 38].

Proposal: spatially layered representation. To enable streamable full-scene VVs, we advocate a spatially layered representation. A typical full-scene VV can be disentangled into a static background and a dynamic foreground according to the mobility of objects [29, 38]. In our design, NeRF is employed to generate backgrounds for different viewports, thereby significantly reducing the bandwidth requirements. The NeRF model only requires a single transfer over the network, while allowing long-term free-viewpoint rendering. For the unstructured foreground content with moving objects, PtCls are used to achieve the appearance of dynamic entities in a continuous manner. Meanwhile, foreground prefetching can take advantage of a long video buffer without being impeded by the short viewport prediction window.

3 V^{2}NERF MECHANISM

3.1 System Overview

The goal of V^{2}NeRF is to stream video-on-demand (VoD) full-scene VV stored on an Internet server to the client and render it in the 2D display plane. The system architecture is depicted in Fig. 3.

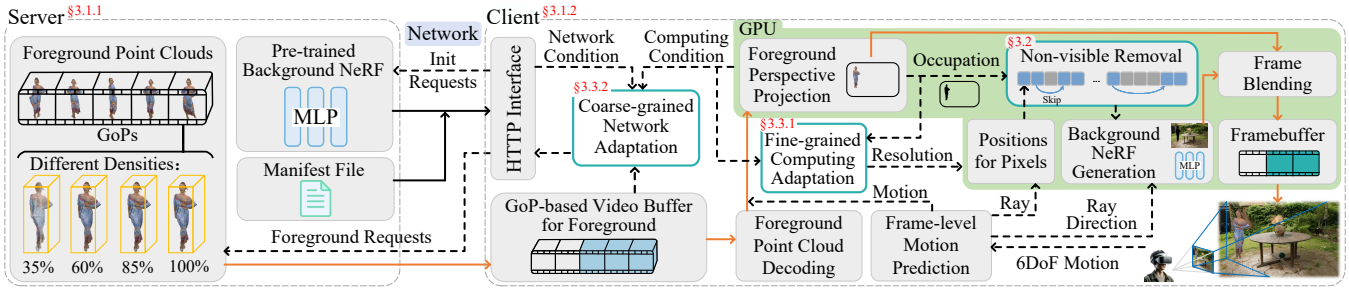


Figure 3: The system architecture of V^{2}NeRF. Solid lines represent data flows, while dashed lines represent information flows.

• *3.1.1 Content Server.* Unlike object-based VVs, our volumetric content consists of foreground PtCls and a pre-trained background NeRF model. According to the Dynamic Adaptive Streaming over HTTP (DASH) protocol, the foreground PtCl video is partitioned into serialized GoPs with a fixed duration, denoted as $C = \{C_1, C_2, ..., C_N\}$. Each GoP is encoded into multiple density levels (i.e., similar to 2D resolution), indicated as $\mathcal{K} = \{1, 2, ..., K\}$. The higher density PtCl contains more details and thus provides better visualization, making it suitable for higher resolution rendering [9, 37]. The background NeRF model is trained based on multi-view RGB images. With techniques such as hashing transformation [24] and dictionary fields [10], the training process for NeRF is lightweight, incurring only minute-level of time cost on the consumer-level GPU. Furthermore, the size of the NeRF model in our system is only approximately equivalent to the 2-second foreground PtCls. The information related to the VV and NeRF, including model features, content size, and content quality, is recorded in the manifest file and accessible as an HTTP resource.

• *3.1.2 Client.* It first requests the manifest file and the pre-trained NeRF model to allow the player to initialize. The network adaptation module then formulates requests for the foreground PtCl GoPs, taking into account the network and computing conditions, and transmits them via the HTTP interface. Then, the downloaded GoPs are filled into the video buffer, which can be configured with a large size to handle network fluctuations. During data transmission, the cached GoPs are decoded into raw PtCl frames.[1] The motion prediction forecasts the immediate user's 6DoF motion. Unlike GoP-based works [8, 9], this prediction is only for the next $\rho = 3$ frames, which is well-studied [19, 21]. Here, the motion prediction is used to migrate later computation uncertainty without compromising the effectiveness of the video buffer in handling network uncertainty like the previous viewport-aware schemes (e.g., [9, 27]). Next, the client projects the foreground PtCl onto the 2D viewing plane using perspective projection. The NeRF model generates the corresponding background image. Finally, the foreground and background are blended into a unified viewing frame, then filled into the inherent Framebuffer [33] in the video card, and prepared for rendering.

Despite the great advantages in terms of transmission and photorealistic representation, the system implementation in practice needs to address the following challenges: *(C1) Intensive computation requirements* and *(C2) Multiscale adaptation scheduling.*

3.2 Non-visible Background Removal

To address C1, the main idea is to avoid the NeRF inference of non-visible background. For NeRF inference, the two-dimensional image view is flattened into one-dimensional vectors. Each pixel obtains its ray position according to its location in the image, which serves as part of the NeRF input. Subsequently, each pixel queries the MLP model to generate its RGB color. To avoid NeRF inference of the non-visible region, an intuitive scheme is to prevent the ray positions of occluded pixels from also being taken as NeRF inputs. Yet, this would involve extra pixel deletion and recovery of the original pixel arrangement, resulting in an unnecessary time cost. As such, we propose a *non-visible pixel skipping* method. During the ray casting process for a pixel, if the transmittance of the ray is lower than the minimum threshold, the ray is marked as converged and the marching process is terminated. Motivated by this, we proactively set the transmittance of occluded pixels to the minimum value and integrate this attribute into the ray vector through matrix operations, to realize the skipping operations in NeRF generation, as depicted in Fig. 4(a). The time cost of our removal approach running on RTX 3090 GPU ranges from 0.16ms to 0.185ms at various resolutions.[2] Meanwhile, it maintains a slow increase with growing resolution because of well-designed matrix operations.

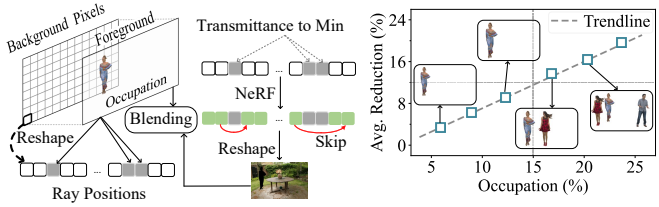
As video content and user viewing behavior evolve, the amount of occlusion is significantly affected by factors such as the viewing distance and the number of foreground objects. Fig. 4(b) illustrates the degree of occlusion corresponding to different scenarios and shows the influence of the occlusion ratio on the background computation (i.e., the generation time of frame). As the occlusion ratio increases, there is a significant reduction in the NeRF computation, demonstrating a linear relationship. It is represented as $f(o)$, where o is the occlusion ratio, quantifying the percentage reduction in NeRF computation. In turn, reduced foreground means fewer PtCl projections and more available computation time for NeRF.

3.3 Multiscale Adaptation Scheduling.

A holistic scheme to tackle the multiscale challenge is unreliable because of the time-varying network and computation conditions. Thus, we propose a two-stage optimization architecture to decouple the multiscale scheduling, while maintaining connections between the transmission and computation.

[1]The decoding process can be done concurrently with the data transfer and run in the CPU without additional time overhead (e.g., average 6.87ms per frame decoding [37]).

[2]Here, the maximum resolution is 864×560, and a fine-grained resolution ladder {1, 12/13, 6/7, 4/5, 3/4} is used to avoid sharp quality fluctuations for a better experience.



(a) Non-visible pixel skipping method. (b) Computation reduction

Figure 4: Overview of non-visible background removal.

• *3.3.1 Fine-grained computing adaptation.* As shown in Fig. 3, foreground perspective projection and NeRF generation both operate at the frame level and require hardware acceleration (e.g., GPU). To maintain the stable frame rate (e.g., $\eta = 24$), a frame-level computing adaptation is required here to cope with changing computing requirements and resources. For example, the foreground PtCl projection with different density levels will affect available hardware time for NeRF generation. Meanwhile, the non-visible background removal also impacts the generation computation in real time. We accomplish this adaptation by adjusting the resolution of the background image generated by NeRF. A higher resolution provides a better visual experience but requires a longer inference time due to the increased number of pixels, and vice versa. Here, the frame sequences are denoted as $\mathcal{F} = \{F_1, F_2, ..., F_I\}$. The control variable $x_{ij}, \forall j \in \{1, 2, ..., J\}$ is used for resolution selection, meaning that the generated resolution is j for the i-th frame.

Recall that the system aims to maximize the user viewing experience, which is typically captured using three aspects [19, 37]: *(i) Perceptual quality* (Q_i^F). The popular perceptually-learned metric (LPIPS) [39] is used instead of traditional metrics, such as PSNR and SSIM, to evaluate the viewing quality, given its superior ability to capture the nuances of human perception. Notably, a lower LPIPS value indicates better perceptual quality. Thus, the viewing quality of the generated i-th frame is expressed as $Q_i = -LPIPS(x_{ij})$. *(ii) Rebuffering* (R_i^F) that represents how long the video stalls because the Framebuffer is empty. The PtCl projection time is indicated as $T(F_{ik})$, where the i-th frame is in the n'-th chunk and its density level is k. The NeRF generation time is denoted as $T(x_{ij})(1-f(o_i))$, where $f(o_i)$ is the percentage reduction in computation due to occlusion. The rebuffering time incurred by computation is given by $R_i^F = [T(x_{ij})(1 - f(o_i)) + T(F_{ik}) - B_i^F]^+$, where $[.]^+ = \max\{., 0\}$; and $B_{i+1}^F = [B_i^F - T(x_{ij})(1-f(o_i)) - T(F_{ik})]^+ + 1/\eta$ is the available playback time (i.e., Framebuffer occupancy). *(iii) Smoothness* (S_i^F). To avoid frequent viewing quality changes in frame level, we consider the quality switching between all adjacent frames within a given period τ, instead of traditional variations between only two consecutive frames. It is denoted as $S_i^F = 1/\tau \sum_{t=i}^{i-\tau} |Q_t^F - Q_{t-1}^F|$.

The *objective* of this stage is formulated as $\mathcal{P}_1 = \max \frac{1}{I} \sum_{i=1}^{I} (Q_i^F - \alpha R_i^F - \beta S_i^F)$, where α and β are adjustment factors [19, 21, 37]. For a seamless frame blending, the perceptual quality of the generated background should be consistent with the foreground PtCls. To achieve this, we propose the plane quality alignment scheme and establish the mapping relationship $h(C_{n'k})$ between various PtCl density levels and their plane viewing quality by careful enumeration method. The high-density PtCls can be backward-compatible with lower-quality background. For the *consistency representation*, the background quality is constrained by the highest quality perceptible on PtCls: $x_{ij} \leq h(C_{n'k})$.

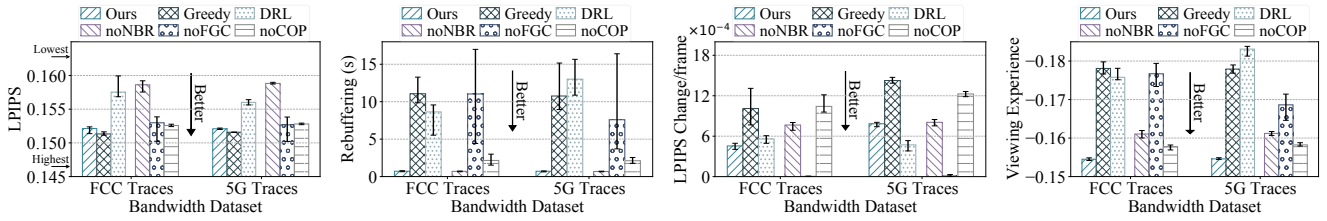
• *3.3.2 Coarse-grained network adaptation.* V^{2}NeRF adapts to varying network conditions by controlling the density of foreground PtCls and utilizing a long video buffer operating at the GoP level. This control variable is represented as $y_{nk}, \forall k \in \mathcal{K}$, meaning that the desired density level is k for the n-th GoP. The video buffer occupancy is B_n^G. While traditional network adaptation expects to prefetch high-quality content while avoiding playback stalls, overly dense PtCls are not always beneficial in the system, as their high projection computation would potentially squeeze the available GPU time for NeRF generation, resulting in unnecessary bandwidth waste and viewing quality reduction.

As such, the goal is captured by: *(i) Density quality* (Q_n^G), which denotes the downloaded density level of PtCl: $Q_n^G = y_{nk}$. *(ii) Stall penalty* (R_n^G) that represents the stall duration when the video buffer is empty. It is denoted as $R_n^G = [D_{nk}/W_n - B_n^G]^+$, where D_{nk} signifies volume of GoP C_{nk}; and W_n is the available bandwidth. The buffer occupancy is updated by $B_{n+1}^G = [B_n^G - D_{nk}/W_n]^+ + \delta$, where δ is GoP duration. *(iii) Oversupply penalty* ($\triangle_n^G$). This strives to prevent prefetching the oversupply of PtCl density, which would not be fully utilized in display due to the need for compatibility with low-quality background. It helps ensure that NeRF generation is not squeezed unnecessarily. To this end, the penalty is calculated as the difference between the perceptual quality of PtCl and the generable quality by NeRF, represented as $\triangle_n^G = [h(y_{nk}) - 1/M \sum_{m=1}^{M} x'_{mj}]^+$, where m is the index of last $M = 6$ viewed frames. Note that x'_{mj} denotes generated background quality without the consistency representation constraint. Thus, the *objective* represents as $\mathcal{P}_2 = \max \frac{1}{N} \sum_{n=1}^{N} (Q_n^G - \lambda R_n^G - \gamma \triangle_n^G)$, where λ and γ are adjustment factors.

3.4 Buffer-aware Optimization

The system expects to efficiently tackle the above nonlinear discrete optimization problems ($\mathcal{P}_1$ and $\mathcal{P}_2$), which have a large solution space and inter-decision interaction. One potential option is to explore a deep reinforcement learning (DRL) approach like Pensieve [22]. Yet, this requires pre-training and customization for a specific environment [35]. It also results in a significant inference overhead in terms of time and computing resources. Meanwhile, recent works [35, 36] indicate that buffer-based approaches (e.g., [26]) are more practical than the learning-based methods.

As such, we propose the *buffer-aware Simulated Annealing (SA)* algorithm. SA is a probabilistic approach that can approximate global optimum and reduce search complexity [14]. At runtime, we consider a finite horizon of the next ω frames or ε GoPs. For fine-grained computing adaptation, the algorithm begins with a buffer-aware scheme. When the Framebuffer occupancy exceeds the reserved value Res^F, it is reset to $B_i^F -= Cus^F$, where Cus^F is a cushion value designed to offset the fluctuation effects of computation. If not, the system directly selects the lowest quality to quickly recover buffer occupancy to avoid rebuffering. Next, the SA algorithm first sets the generated quality of all the next ω frames to the minimum level. It proceeds to improve the quality of each frame by one level. If the optimized objective for the finite horizon increases as a result of the change, it will be accepted. Otherwise, it



(a) Perceptual quality. (b) Total rebuffering time. (c) Quality smoothness. (d) Quality of user experience.

Figure 5: The performance comparison of proposed V^{2}NeRF with other benchmarks under various network conditions.

may still accept the new decision with a probability of $\exp(-\triangle/t)$. Here, $\triangle$ denotes the decrease in the optimized goal, and t is the current annealing temperature, to prevent potential local maximum. Finally, the decided quality should satisfy the consistency constraint. For coarse-grained network adaptation, the network is constantly changing. It firstly predicts the bandwidth of the next ε GoPs. Bandwidth prediction has been widely studied [31]. Here, we adopt the existing method in [26]. Then, a similar buffer-aware SA algorithm is used to obtain the density decision of PtCls, where Res^G and Cus^G denote the reserved value and cushion value for the video buffer. To make the algorithm fast and lightweight, the finite horizons ω and ε are empirically set to 2 and 3. Based on the Framebuffer/video buffer size, the Res^F and Res^G are set to $0.1/\eta$ and 4δ; and the Cus^F and Cus^G are set to $0.3/\eta$ and 6δ.

4 PROTOTYPE EVALUATION

We perform extensive experiments to evaluate the system's streaming and viewing performance from the following aspects:

▹ **Exp. 1:** Evaluation of detailed and overall user experiences compared to other schemes under time-varying network conditions.

▹ **Exp. 2:** Analysis of the individual contributions of each component to the V^{2}NeRF's overall performance.

▹ **Exp. 3:** Assessment of the overhead associated with V^{2}NeRF.

4.1 System Configurations

The test platform comprises two desktop computers, designated as the server and the client. They are interconnected via a gigabit router. The server runs on Ubuntu 20.04 LTS and utilizes Apache Tomcat software for DASH services. The client is equipped with an NVIDIA RTX 3090 GPU and Intel i5-12600K CPU and operates a script player in Python to facilitate the NeRF generation and full-scene VV playback. It maintains a 5-second video buffer for the foreground PtCl prefetching and a 3-frame Framebuffer for content rendering. The influence of network conditions, such as delay, jitter, and packet loss, is abstracted as end-to-end throughput. Thus, we utilize two real-world throughput datasets: *(i) FCC dataset* [3], representing the American fixed broadband measurements (i.e., 2023.5-2023.7); *(ii) 5G dataset* [25], comprising 5G data collected during video streaming with an Irish mobile operator. They respectively represent two typical network fluctuations. In line with PtCl data volume, we randomly select 10 throughput traces averaging around 60Mbps, corresponding to the 80th and 60th percentile values of CDF in FCC and 5G. On the server, all traces are emulated using the built-in traffic control (TC) tool of Linux. Given the lack of publicly accessible, high-fidelity full-scene VVs, inspired by digital asset composition [34], we built a synthetic dataset that integrates 8i PtCls [4] and Google 360° scenes [1]. The dynamic 8i PtCls are used as the foreground content, while the static 360° scenes are as the background environment. We chose the three PtCl videos from 8i: Longdress, Redandblack, and Loot as illustrated in Fig. 4(b); and the Garden scene from the Google dataset. Since original PtCl videos only have 300 frames, we loop them to extend the duration to 3 minutes. Each PtCl video is encoded using G-PCC [6] into five density levels from r01 to r05. Similar to [30], user viewing traces are generated using pre-defined random motion paths. The system only requires the next 3-frame (i.e., Framebuffer size) motion prediction. As a result, a lightweight linear regression method is proved sufficient [9, 21].

4.2 Experiment Setup

NeRF model. We employ recently proposed Instant-NGP [24] as the NeRF model, which exhibits superiority in training/generation speed (i.e., minute-level/millisecond-level) and generation quality (i.e., photorealistic-level) for the static full-scene content, thanks to its multiresolution hash encoding technique. Consistent with [23, 24], the given scene is equipped with an individual model.

Metrics. *(i) Perceptual quality*: LPIPS [39]. *(ii) Rebuffering time*, the accumulated rebuffering (s) during video playback. *(iii) Smoothness penalty*, the frame-average quality variation in LPIPS. *(iv) Viewing experience (QoE)*, the optimized goal of fine-grained computing adaptation. Refer to [19, 21, 37], the adjustment factors α and β are empirically set to 10 and 1; while λ and γ are set to 1 and 1.1 to trade-off the quality benefits and penalties of poor experience. Those parameters can be easily adjusted by content providers to suit their service patterns.

Baselines. *(i) Greedy*, a standard bandwidth and computing resource-based greedy approach, tries to get the maximum reward possible in multiscale adaptation decisions. *(ii) DRL* [22], which applies a DRL method to address the adaptation problem. Similar to [22], we train the DRL model offline with a simulation program, followed by online fine-tuning and deployment.

Ablation strategies. To assess the impact of each system component, we consider: *(i) noNBR*, which only involves the multiscale adaptation without implementing the Non-visible Background Removal. *(ii) noFGC*. Here, the Fine-Grained Computing is replaced by a fixed generated quality selected based on end-device performance. *(iii) noCOP*. Since previous studies have underscored the necessity of network adaptation, this scheme is to evaluate the effect of excluding the Coarse-grained Oversupply Penalty component.

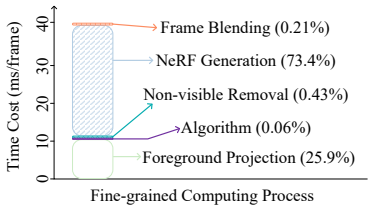


Figure 6: The average time cost of each system component.

4.3 Results and Analysis

(Exp. 1) Testing on various network conditions. Fig. 5 presents the performance comparison of detailed metrics and overall user experience. Since LPIPS is a learned perceptual metric, to facilitate a fair comparison among various schemes, we utilize the normalized LPIPS growth metric, i.e., $(\text{LPIPS} - \text{LPIPS}_{\text{lowest}})/(\text{LPIPS}_{\text{highest}} - \text{LPIPS}_{\text{lowest}})$, which allows for a quantifiable assessment of quality differences. As shown in Fig. 5(a)-(c), the Greedy scheme maximizes the current optimized goal, but it fails to resist the fluctuations in transmission and computation. As a result, it incurs the unaccepted rebuffering time (i.e., 11.05s in FCC and 10.75s in 5G) and quality variation (i.e., with an increase of 1.22× in FCC and 82% in 5G over Ours). Thanks to the buffer-aware SA algorithm, our scheme can effectively deal with the uncertainty in the system. Compared with the DRL approach, V^{2}NeRF improves the perceptual quality by 1.03× in FCC and by 57% in 5G. The primary reason for this disparity lies in the significant inference overhead involved in DRL, both in terms of time and computing resources which significantly affect the NeRF generation computation. Meanwhile, DRL's inherent limitations in adapting to various environments lead to a substantial increase in rebuffering time, i.e., 8.6s in FCC and 13s in 5G. From the perspective of user experience, as shown in Fig. 5(d), our approach achieves a 74.4% increment over Greedy and a 75.3% increment over DRL. This indicates the effectiveness and robustness of our algorithm under various network conditions.

(Exp. 2) Testing on ablation studies. To gain a deeper insight into performance gains, we assess the impact of each design component. From Fig. 5(a), we can find that the perceptual quality of noNBR decreases by 15.8% and quality change increases by 35.4% compared to full V^{2}NeRF setup. This is attributed to noNBR's inability to remove non-visible background computation, necessitating a reduction in generated quality to save computation time. Benefiting from the proposed algorithm, the noNBR maintains a low rebuffering time, i.e., 0.68s. Compared with the noFGC, Ours achieves an 8.6% increment in perceptual quality in FCC and a 5.7% increment in 5G. Since noFGC adopts a fixed generated quality, which fails to counter the computation fluctuations and incurs a long rebuffering time, for instance, 11.02s in FCC and 7.6s in 5G. The noCOP, lacking the oversupply penalty, weakens the connection between the network and computing, resulting in inefficient resource utilization. For example, its perceptual quality decreases by 5.7%, rebuffering time increases by 2×, and quality change increases by 2.1×. Fig. 5(d) shows that Ours achieves a 45% improvement in viewing experience over noNBR, a 68.4% improvement over noFGC, and a 29.7% improvement over noCOP.

(Exp. 3) Testing on system overhead. To evaluate the overhead associated with V^{2}NeRF, we measure the average time cost per frame for each component involved in the fine-grained computing process, as shown in Fig. 6. We can find that the proportion of additional time cost is only 0.7%, including the algorithm time (0.06%), non-visible removal time (0.43%), and frame blending time (0.21%). In comparison, the time cost of the DRL is 3.01ms (7.6%), which imposes a significant system overhead. The total time cost for computation is measured at 39.4ms per frame, which means our system achieves real-time processing at 24 frame rates.

5 RELATED WORK

Volumetric video streaming. Previous studies, such as ViVo [9] and Vues [21], mainly focus on 3D tile-based viewport adaptation approaches, which involve adjusting the density levels of tiles to optimize the viewing experience. The cache-assisted strategy (e.g., [19]), compression-based method (e.g., [15, 31]), and super-resolution-enhanced (e.g., [37]) are further investigated to reduce data transfer. Yet, they mainly focus on object-based VV and lack the consideration for full-scene content. In addition, the viewing patterns of users for full-scene VVs are significantly different from regular videos. Users often expect to view content in different spaces at a given time point for a better spatial experience. This is difficult to support with existing viewport-aware schemes due to their incomplete data delivery.

Implicit neural radiance field. NeRF has shown an impressive ability to learn to represent static or nearly static 3D objects and scenes [16, 23, 24, 28]. However, it is difficult to deal with objects with large-scale movements (e.g., topological changes and articulated objects) and capture appearance details of dynamic scenes [29, 38]. Although some studies (e.g., [17, 29]) attempt to modify NeRF for some given VVs, they typically require integrating the time dimension as an additional input to NeRF or employing a secondary MLP to model and learn deformations for each video frame. They are far from being practical for full-scene VV streaming due to the slow rendering speed (e.g., 2 mins per 1080p frame [38]), large model size alongside lengthy training periods, and short video duration limitation (e.g., second-level [29]).

6 CONCLUSION

We presented the V^{2}NeRF, which integrates the implicit neural radiance field and explicit 3D point cloud to realize a streamable full-scene volumetric video representation. Furthermore, we proposed a lightweight non-visible background removal method to migrate intensive computation requirements and a two-stage decoupled strategy to optimize the multiscale network/computing scheduling. A suitable buffer-aware simulated annealing algorithm is developed and the perceptually-learned LPIPS metric is utilized to maximize the quality of user experience. Extensive evaluations with prototype implementation verify the superior performance of V^{2}NeRF.

ACKNOWLEDGMENTS

We appreciate the constructive comments from the reviewers. Jianxin Shi, Lingjun Pu, and Jingdong Xu are partially supported by the National Natural Science Foundation of China (No. 62172241).

REFERENCES

[1] J. T Barron, B. Mildenhall, D. Verbin, et al. 2022. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR '22)*. 5470–5479.

[2] R. Cheng, K. Liu, N. Wu, and B. Han. 2023. Enriching telepresence with semantic-driven holographic communication. In *Proceedings of the ACM Workshop on Hot Topics in Networks (HotNets '23)*. 147–156.

[3] Federal communications commission (FCC). 2023. Measuring broadband raw data releases. https://www.fcc.gov/oet/mba/raw-data-releases/. [Online; accessed 3-Dec-2023].

[4] E. d'Eon, B. Harrison, T. Myers, and P. Chou, A. Geneva, January, 2017. 8i voxelized full bodies - A voxelized point cloud dataset. *ISO/IEC JTC1/SC29 Joint WG11/WG1 (MPEG/JPEG) input document WG11M40059/WG1M74006* (Geneva, January, 2017).

[5] Google. 2023. Draco 3D Data Compression. https://github.com/google/draco. [Online; accessed 3-Oct-2023].

[6] D. Graziosi, O. Nakagami, S. Kuma, A. Zaghetto, T. Suzuki, and A. Tabatabai. 2020. An overview of ongoing point cloud compression standardization activities: Video-based (V-PCC) and geometry-based (G-PCC). *APSIPA Transactions on Signal and Information Processing* 9 (2020), e13.

[7] Y. Guan, X. Hou, N. Wu, B. Han, and T. Han. 2023. MetaStream: Live volumetric content capture, creation, delivery, and rendering in real time. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom '23)*. 1–15.

[8] S. Gül, D. Podborski, T. Buchholz, et al. 2020. Low-latency cloud-based volumetric video streaming using head motion prediction. In *Proceedings of the ACM Workshop on Network and Operating Systems Support for Digital Audio and Video (NOSSDAV '20)*. 27–33.

[9] B. Han, Y. Liu, and F. Qian. 2020. ViVo: Visibility-aware mobile volumetric video streaming. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom '20)*. 1–13.

[10] A. hen, Z. Xu, X. Wei, S. Tang, H. Su, and A. Geiger. 2023. Dictionary fields: Learning a neural basis decomposition. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '23)*.

[11] K. Hu, Y. Jin, H. Yang, J. Liu, and F. Wang. 2023. FSVVD: A dataset of full scene volumetric video. In *Proceedings of the Conference on ACM Multimedia Systems (MMSys '23)*. 410–415.

[12] K. Hu, H. Yang, Y. Jin, et al. 2023. Understanding user behavior in volumetric video watching: Dataset, analysis and prediction. In *Proceedings of the ACM international conference on multimedia (MM '23)*. 1108–1116.

[13] ISO/IEC 14496-2 Information Technology — Coding of Audio-visual Objects — Part 2: Visual. 1999. https://api.semanticscholar.org/CorpusID:14775904

[14] S. Kirkpatrick, C D. Gelatt Jr, and M. P Vecchi. 1983. Optimization by simulated annealing. *Science* 220, 4598 (1983), 671–680.

[15] K. Lee, J. Yi, Y. Lee, , et al. 2020. GROOT: A real-time streaming system of high-fidelity volumetric videos. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom '20)*. 1–14.

[16] L. Li, Z. Shen, Z. Wang, L. Shen, and L. Bo. 2023. Compressing volumetric radiance fields to 1 MB. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR '23)*. 4222–4231.

[17] T. Li, M. Slavcheva, M. Zollhoefer, S. Green, et al. 2022. Neural 3D video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR '22)*. 5521–5531.

[18] J. Liu, Y. Wang, Y. Wang, Y. Wang, S. Cui, and F. Wang. 2023. Mobile volumetric video streaming system through implicit neural representation. In *Proceedings of the Workshop on Emerging Multimedia Systems (EMS '23)*. 1–7.

[19] J. Liu, B. Zhu, F. Wang, et al. 2023. CaV3: Cache-assisted viewport adaptive volumetric video streaming. In *IEEE Conference Virtual Reality and 3D User Interfaces (VR '23)*. 173–183.

[20] Kaiyan Liu et al. 2023. Toward next-generation volumetric video streaming with neural-based content representations. In *Proceedings of the ACM Workshop on Mobile Immersive Computing, Networking, and Systems (ImmerCom '23)*. 199–207.

[21] Y. Liu, B. Han, et al. 2022. Vues: Practical mobile volumetric video streaming through multiview transcoding. In *Proceedings of the ACM Annual International Conference on Mobile Computing and Networking (MobiCom '22)*. 514–527.

[22] H. Mao et al. 2017. Neural adaptive video streaming with pensieve. In *Proceedings of ACM Special Interest Group on Data Communication (SIGCOMM '17)*. 197–210.

[23] B. Mildenhall, P. P Srinivasan, M. Tancik, et al. 2020. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV '20)*. 4700–4708.

[24] T. Müller, A. Evans, C. Schied, and A. Keller. 2022. Instant neural graphics primitives with a multiresolution hash encoding. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '22)*.

[25] D. Raca, D. Leahy, C. J. Sreenan, and J. J. Quinlan. 2020. Beyond throughput, the next generation: A 5G dataset with channel and context metrics. In *Proceedings of the ACM Multimedia Systems Conference (MMSys '20)*. 303–308.

[26] K. Spiteri et al. 2020. BOLA: Near-optimal bitrate adaptation for online videos. *IEEE/ACM Transactions On Networking (TON)* 28, 4 (2020), 1698–1711.

[27] S. Subramanyam, I. Viola, J. Jansen, E. Alexiou, A. Hanjalic, and P. Cesar. 2022. Evaluating the impact of tiled user-adaptive real-time point cloud streaming on VR remote communication. In *Proceedings of the ACM International Conference on Multimedia (MM '22)*. 3094–3103.

[28] T. Takikawa, J. Evans, A.and Tremblay, T. Müller, M. McGuire, A. Jacobson, and S. Fidler. 2022. Variable bitrate neural fields. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '22)*. 1–9.

[29] H. Turki et al. 2023. SUDS: Scalable urban dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR '23)*.

[30] L. Wang, C. Li, W. Dai, et al. 2022. QoE-driven adaptive streaming for point clouds. *IEEE Transactions on Multimedia (TMM)* 25 (2022), 2543–2558.

[31] Y. Wang, D. Zhao, H. Zhang, C. Huang, T. Gao, Z. Guo, L. Pang, and H. Ma. 2023. Hermes: Leveraging implicit inter-frame correlation for bandwidth-efficient mobile volumetric video streaming. In *Proceedings of the ACM international conference on multimedia (MM '23)*. 9185–9193.

[32] M. Wijnants, G. Rovelo, P. Quax, and W. Lamotte. 2016. A pragmatically designed adaptive and web-compliant object-based video streaming methodology: Implementation and subjective evaluation. In *Proceedings of the ACM international conference on multimedia (MM '16)*. 1267–1276.

[33] Wiki. 2023. The introduction of Framebuffer. https://en.wikipedia.org/wiki/Framebuffer. [Online; accessed 26-Nov-2023].

[34] J. Wu, Y. Guan, Q. Mao, Y. Cui, Z. Guo, and X. Zhang. 2023. ZGaming: Zero-latency 3D cloud gaming by image prediction. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM '23)*. 710–723.

[35] Z. Xia, Y. Zhou, F. Y Yan, and J. Jiang. 2022. Genet: Automatic curriculum generation for learning adaptation in networking. In *Proceedings of the ACM Special Interest Group on Data Communication (SIGCOMM '22)*. 397–413.

[36] F. Y Yan, H. Ayers, Ch. Zhu, S. Fouladi, J. Hong, et al. 2020. Learning in situ: A randomized experiment in video streaming. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI '20)*. 495–511.

[37] A. Zhang, C. Wang, B. Han, and F. Qian. 2022. YuZu: Neural-enhanced volumetric video streaming. In *USENIX Symposium on Networked Systems Design and Implementation (NSDI '22)*. 137–154.

[38] J. Zhang, X. Liu, X. Ye, et al. 2021. Editable free-viewpoint video using a layered neural representation. In *Proceedings of the Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '21)*.

[39] R. Zhang, P. Isola, A. A Efros, E. Shechtman, and O. Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR '18)*.