4DGStream: Variable Bitrate Dynamic Gaussian Splatting Streaming

Zhicheng Liang, Dayou Zhang, Linfeng Shen, Miao Zhang, Jian Zhang, Bin Ju, Mallesham Dasari, Fangxin Wang, Jiangchuan Liu *Fellow, IEEE*

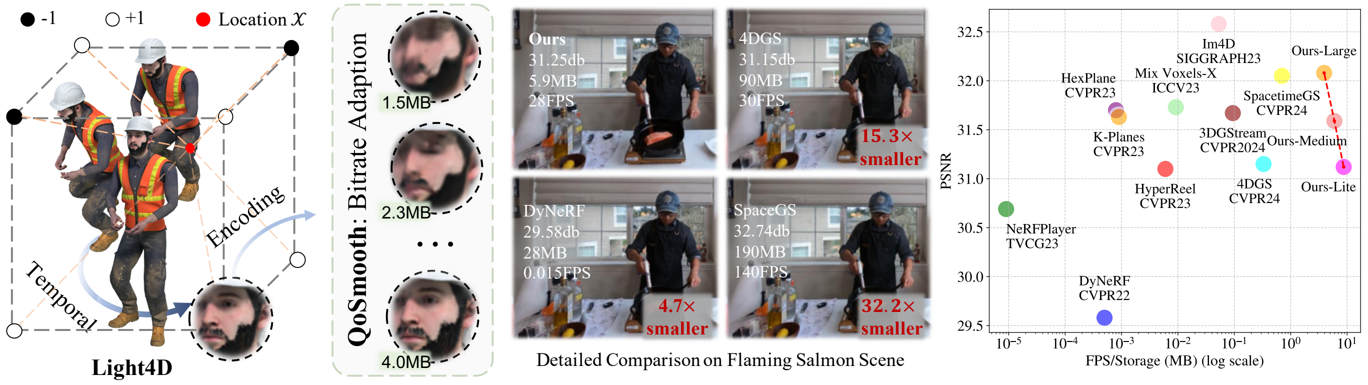


Fig. 1: `4DGStream` integrates `Light4D`, which utilizes binarization-assisted spatiotemporal optimization for compact representation of dynamic 3D Gaussians, and `QoSmooth` for adaptive bitrate streaming. The two rightmost figures show performance on the flaming salmon scene (from Neu3D) and the overall Neu3D dataset.

***Abstract*—While 3D Gaussian Splatting (3DGS) has revolutionized static scene representation, the extension to dynamic scene, i.e., 3DGS video (GSV), faces challenges related to reconstruction quality, rendering speed, and storage requirements. The substantial data volume of current GSV poses significant hurdles for streaming applications, particularly in the realm of AR, VR and MR. To tackle these challenges, we introduce `4DGStream`, a novel framework that integrates an efficient GSV compression method, `Light4D`, and a bitrate adaptation streaming strategy, `QoSmooth`, to ensure smooth playback while maintaining high visual quality. `Light4D` employs a binarization-assisted spatiotemporal deformation network to model the deformation of Gaussian primitive attributes over time, while a spatiotemporal-aware masking module prunes trivial Gaussians, further enhancing long-term reconstruction quality. To reduce storage, `Light4D` uses a binary hash grid to model the entropy of attributes for arithmetic coding, with its binary nature allowing efficient entropy modeling via a Bernoulli distribution. These components enable `Light4D` to improve the FPS/Storage metric by up to 12.4× over SpacetimeGS and 26.4× over 4DGS on the Neu3D dataset, with performance gains exceeding 3× orders of magnitude compared to other NeRF-based state-of-the-art (SOTA) methods. Here, FPS/Storage reflects the balance between rendering speed and data storage. Despite significant model size reductions, `Light4D` maintains or surpasses the reconstruction quality of 4DGS. Furthermore, `QoSmooth` provides effective rate control to enhance playback smoothness, reducing bitrate level switches by 61.6% and increasing time-average utility by 26.2%. All these improvements make `4DGStream` highly suited for GSV streaming, improving QoE by 36.7% compared to SOTA methods.**

***Index Terms*—3D Gaussian Splatting, Compression, Media Streaming, Quality of Experience, Virtual Reality.**

Zhicheng Liang and Dayou Zhang are with Shenzhen Future Network of Intelligence Institute and the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen. (email: {zhichengliang1, dayouzhang}@link.cuhk.edu.cn).

Jian Zhang and Bin Ju are with Shenzhen Research Institute of Migu Culture Technology Co., Ltd. (e-mail: {zhangjian_x, jubin}@migu.chinamobile.com).

Linfeng Shen, Miao Zhang and Jiangchuan Liu are with Simon Fraser University, British Columbia, Canada. (e-mail: {linfengs, mza94, jcliu}@sfu.ca).

Mallesham Dasari is with Northeastern University, United States. (e-mail: m.dasari@northeastern.edu).

Fangxin Wang is with School of Science and Engineering (SSE), Shenzhen Future Network of Intelligence Institute (FNii-Shenzhen) and The Guangdong Provincial Key Laboratory of Future Networks of Intelligence, The Chinese University of Hong Kong, Shenzhen. (email: wangfangxin@cuhk.edu.cn).

Manuscript received xxx; revised xxx.

The work was supported in part by the Guangdong S&T Programme with Grant No. 2024B0101030002, the Basic Research Project No. HZQB-KCZYZ-2021067 of Hetao Shenzhen-HK S&T Cooperation Zone, the NSFC with Grant No. 62293482 and Grant No. 62471423, the Shenzhen Science and Technology Program with Grant No. JCYJ20241202124021028 and Grant No. JCYJ20230807114204010, the Shenzhen Outstanding Talents Training Fund 202002, the Guangdong Research Projects No. 2019CX01X104 and 2024TQ08X346, the Young Elite Scientists Sponsorship Program of CAST (Grant No. 2022QNRC001), the Guangdong Provincial Key Laboratory of Future Networks of Intelligence (Grant No. 2022B1212010001) and the Shenzhen Key Laboratory of Big Data and Artificial Intelligence (Grant No. ZDSYS201707251409055).

I. Introduction

Recent advancements in 3D scene representations have revolutionized the field of novel view synthesis, with 3D Gaussian Splatting (3DGS) [1] emerging as a particularly promising approach. By representing scenes as explicit, learnable 3D Gaussians, 3DGS achieves rapid differentiable rendering and high photo-realistic fidelity, surpassing previous methods based on Neural Radiance Fields (NeRF) [2]. This breakthrough has led to widespread adoption and further developments in the field, including extensions to dynamic scenes [3], [4], [5].

The application of 3D Gaussian Splatting to dynamic scenes introduces significant challenges due to the increased complexity of modeling temporal changes. Existing compression techniques [6], [7], [8], [9], [10], [11] for static 3D Gaussian representations, such as parameter pruning and vector quantization, fail to efficiently handle the complex motion of points from sparse input in dynamic scenes. A straightforward approach involves constructing 3D Gaussians at each timestamp [3], [12] and applying compression to each frame of static 3DGS, but this dramatically increases storage and memory costs, especially for long input sequences.

Recent works have largely focused on constructing compact representations while maintaining both training and rendering efficiency [4], [5], leaving the streaming of GSV relatively unexplored yet important. While these methods could be adapted for streaming, challenges persist. First, the current GSV representations remain too large in size. Second, in video streaming, bitrate adaptation requires smooth transitions, but existing GSV representations do not incorporate rate-distortion joint optimization [13] during training, making it difficult to balance size and visual quality, which in turn complicates the smooth transmission between different bitrate levels. Third, existing methods [4], [5], [14], [15], [16], [17], [18] use large grid features for modeling, which cannot be progressively transmitted or gradually improved on the client side, leading to potential buffering issues as the entire grid must be transmitted successfully before rendering can proceed.

To tackle these issues, we propose `4DGStream`, a novel framework designed to efficiently compress GSV using `Light4D` and streaming using `QoSmooth`. Our `Light4D` introduces four key innovations, (1) utilizing a spatiotemporal deformation network with hybrid 2D-3D features generated by binary-encoded K-planes voxel grids to effectively capture scene dynamics; (2) implementing a spatiotemporal-aware learnable masking strategy to prune less crucial Gaussians over time, optimizing both storage efficiency and rendering quality, (3) leveraging binary hash grid to model the entropy of attributes for entropy coding, and (4) employing a Bernoulli modeling based technique to compress binary hash grid into compact bitstream for transmission. The rate-distortion joint loss is incorporated during optimization, enabling a more efficient, variable bitrate method for representing and compressing GSV. Additionally, the `QoSmooth` is a Lyapunov optimization-based method that adaptively selects bitrate level during GSV streaming, balances the trade-off between smooth playback and maintaining visual quality to maximize the QoE.

Through extensive experiments on various dynamic scenes, our GSV representation method `Light4D` achieves comparable visual quality to 4DGS while requiring only 8.9% of its storage on synthetic datasets. On real-world datasets, our approach matches or exceeds the performance of SOTA methods while using as little as 3.3% of their storage requirements. Additionally, our efficient encoding and decoding processes make `Light4D` particularly suitable for real-time streaming applications. Fig. 1 demonstrates an overall comparison that `Light4D` achieves an extremely compact size while delivering comparable visual quality in dynamic scenes. On the Neu3D dataset, `Light4D` achieves improvements of up to 12.4$\times$ compared to SpacetimeGS and 26.4$\times$ compared to 4DGS in the FPS/Storage metric, which evaluates the balance between rendering efficiency and data storage requirements. The main contributions of this work can be summarized as follows:

- We propose `4DGStream`, which, to the best of our knowledge, is the first variable bitrate streaming framework for dynamic Gaussian scenes. It enables efficient compression of dynamic 3D scenes and smooth playback under varying network conditions.
- We develop a deep context model to enable entropy coding for attributes and leverage Bernoulli modeling to capture the entropy of the binary hash grid for efficient compression. This, combined with rate-distortion optimization, balances the trade-off between reducing file size and maintaining acceptable visual quality.
- We propose a Lyapunov-optimization based bitrate adaption algorithm for GSV streaming.
- We conduct comprehensive experiments across various datasets, demonstrating SOTA performance in terms of visual quality, rendering speed, and storage efficiency, QoE, thus validating the effectiveness of our approach.

II. Related Work

A. Neural and Gaussian-based Volumetric Video

The advent of neural representations [19], [20], [21], [22] has revolutionized dynamic volumetric video reconstruction. Several NeRF-based methods [23], [24] have enhanced training and rendering speed through specialized tuning strategies, while others [25], [26], [20] have introduced novel encoding techniques to improve data efficiency. Despite these advancements, NeRF-based approaches, though achieving high compactness, still face significant challenges in rendering efficiency, even after being adapted into more efficient formulations [15], [27].

Recent research has made significant strides in extending 3D Gaussian Splatting to dynamic scenes, enabling high-quality novel view synthesis and real-time rendering of moving objects and environments. These approaches [3], [28], and [5] introduced methods that allow Gaussians to move and rotate over time while maintaining persistent attributes, employing various techniques such as local-rigidity constraints, neural deformation models, and spacetime Gaussian representations. Other researchers, including [29], [30], and [4], focused on improving the efficiency of dynamic scene reconstruction and rendering, achieving real-time performance through techniques like dual-domain deformation modeling, neural transformation caches, and 4D neural voxels. Additionally, [31], [32], and [33] proposed methods that enhance the editability, geometric coherence, and reconstruction quality of dynamic 3D Gaussian representations, further advancing the field of dynamic scene modeling and rendering. While these existing methods have expanded capabilities, they often rely on large-scale data representations that are difficult to transmit efficiently. This

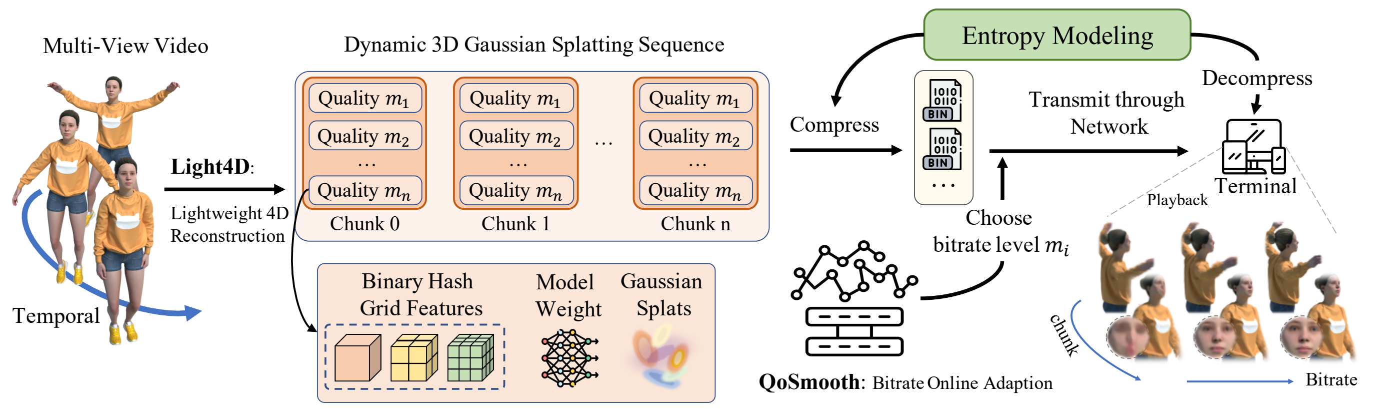


Fig. 2: Overview of our proposed system - 4DGStream

makes the development of compression techniques increasingly urgent.

B. 3D Data Compression

NeRF Compression. Recent advancements in Neural Radiance Field (NeRF) compression have significantly improved the efficiency and applicability of these models. The Vector Quantized Radiance Fields (VQRF) framework [34] achieves a 100× compression ratio through voxel pruning, vector quantization, and post-processing, reducing model size to 1 MB with minimal quality loss. Similarly, Binary Radiance Fields (BiRF) [35] proposes a binary feature encoding method inspired by Binarized Neural Networks, achieving impressive PSNR results while using only 0.5 MB of storage space. The Compressible-composable NeRF [36] approach enables efficient manipulation of 3D models through hybrid tensor rank decomposition and rank-residual learning, allowing for dynamic model size adjustment and model composition. These NeRF compression techniques build upon broader research in model compression and efficient neural network architectures [37], [38], [39].

3D Gaussian Splatting compression. As 3D Gaussian Splatting techniques have advanced, researchers have also focused on developing efficient compression methods to reduce storage requirements and enable practical applications on devices with limited resources. Various approaches have been proposed to achieve this goal. These approaches [6], [7], [8] introduced methods that significantly reduce the storage requirements of 3DGS models. By employing approaches such as hash-grid assisted contexts, self-organizing grids, and compact radiance field representations, these methods achieve compression ratios ranging from 17× to 75× without substantial loss in rendering quality. Other researchers, including [9], [40], [11], and [41], focused on optimizing the representation of 3D Gaussians through techniques like anchor point-based distribution, rate-distortion optimization, and sensitivity-aware vector clustering. These methods have demonstrated impressive compression rates of up to 40×, while maintaining high rendering quality and even improving rendering speed in some cases.

C. Volumetric Video Streaming

Traditional volumetric video streaming. Streaming volumetric video is a challenging task and attracts many research interests [42], [43], [44], [45], [46], [47], [48], [49], [50], [51] Volumetric video streaming faces high bandwidth demands due to its panoramic nature. Nasrabadi et al. [42] supported layered streaming using Scalable High efficiency Video Coding standard [52]. Additionally, tile-based viewport-adaptive techniques have been proposed to address this issue by predicting users' viewports and allocating higher bitrates to the most relevant areas [53], [54], [55], [56], [57], [58], [59].

Dynamic 3D Gaussians Scene Streaming. There is limited work on streaming dynamic 3D Gaussians. While dynamic 3D Gaussian representations [1], [4], [5], [60] are not explicitly designed for streaming, they could be adapted for such scenarios. However, these methods still pose significant storage demands, making their application to streaming challenging.

Several notable online training methods have been developed for on-the-fly reconstruction and real-time streaming. Dynamic 3D Gaussians [3] track dense 3D Gaussians by modeling their motion over time, but still suffer from high temporal redundancy. 3DGStream [30] achieves efficient Free-Viewpoint Video streaming by leveraging 3D Gaussians combined with a Neural Transformation Cache strategy, enabling fast per-frame reconstruction and real-time rendering of dynamic scenes. This approach reduces both training time and storage requirements while maintaining high rendering speed and visual quality. QUEEN [61] updates and compresses all 3DGS attributes without structural constraints, using a quantization-sparsity framework to achieve better memory efficiency and faster training times. However, despite these advancements, these methods still require storage sizes ranging from hundreds of megabytes to several gigabytes for a 10-second video. Furthermore, online training methods still fall short of meeting the real-time standard of 30 FPS. As a result, both online and offline dynamic Gaussian splatting methods are currently unsuitable for GSV live streaming, where the entire pipeline—from data capture to scene reconstruction—must operate at a minimum of 30 FPS. In this context, our focus shifts from live streaming to video-on-demand (VoD) scenarios, which do not require online reconstruction. Instead, VoD requires online video bitrate adaptation to accommodate varying network conditions and ensure smooth and reliable playback.

III. Background and Motivation

A. 3D Gaussians

Starting from a set of Structure-from-Motion (SfM) [62] points, each point is designated as the position (mean) μ of a 3D Gaussian.

$$G(x) = e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)} \tag{1}$$

where $\mu \in \mathbb{R}^3$ is its center and $\Sigma \in \mathbb{R}^{3\times3}$ denotes covariance matrix, which can be decomposed into a scaling factor S and a rotation quaternion R.

$$\Sigma = RSS^T R^T \tag{2}$$

Each Gaussian is characterized by attributes including position $X \in \mathbb{R}^3$, color $C \in \mathbb{R}^{(k+1)^2\times3}$ (where k represents the degrees of freedom), opacity $\alpha \in \mathbb{R}$, rotation factor $R \in \mathbb{R}^4$, and scaling factor $S \in \mathbb{R}^3$. The final rendered color $C(x')$ at a pixel position x' is determined using a tile-based rasterizer that employs alpha-blending with sorted 2D Gaussians:

$$C(x') = \sum_{i\in N} c_i \sigma_i \left(\prod_{j=1}^{i-1}(1-\sigma_j)\right) \tag{3}$$

where N denotes the number of sorted 2D Gaussians associated with the queried pixel, $\sigma_i = \alpha_i G_i'(x')$, and the terms are sorted in descending order of opacity.

B. Superiority of 3DGS Video and Network challenges

Point cloud, mesh, NeRF, and 3D Gaussian videos (GSV) are different techniques for 3D scene rendering. Point cloud videos capture scenes as discrete 3D points for real-time rendering but lack surface detail. Mesh videos use polygons for detailed surfaces but struggle with reflective or transparent elements. NeRF videos generate volumetric scenes using neural networks, enabling high-quality view synthesis but are computationally intensive. GSV, leveraging 3DGS, offers several advantages over traditional volumetric formats like NeRFs and meshes. Its real-time rendering is efficient due to the direct projection of 3D Gaussians onto 2D planes, avoiding the computationally heavy ray-marching of NeRFs. It also shows robustness in handling transparency and reflectiveness [32]. However, with traditional GSV requiring over 10GBps of storage, significant challenges remain in network performance—such as memory and bandwidth demands, compression needs, latency, and compatibility with existing infrastructure—necessitating further optimization for effective streaming [6], [8], [30], [63], [61].

To bridge this gap, we propose `4DGStream`, as illustrated in Fig. 2. `4DGStream` integrates `Light4D`, an effective compression method that utilizes binary hash grid features to model GSV into a sequence of lightweight data chunks. Each chunk offers multiple quality levels for bitrate allocation by `QoSmooth` according to network conditions, an online bitrate adaptation streaming strategy. This integration enables 3D Gaussian video to be applied in real-world scenarios.

IV. Light4D: Compact Dynamic Gaussian Splatting Optimization Framework

In this section, we introduce `Light4D`, as illustrated in Fig. 3, which shows its detailed pipeline: (a) A spatiotemporal deformation network leverages a binary hybrid feature grid, combining 3D voxels and 2D planes across spatial and temporal axes, to predict attribute deformations from input location X and timestamp t (§ IV-B). (b) A spatiotemporal-aware learnable masking module prunes less important Gaussians over time (§ IV-C). (c) A multi-resolution binary 3D voxel hash grid with multi-head MLPs predicts entropy of attributes for arithmetic coding (§ IV-D). (d) A Bernoulli distribution models the entropy of the binary hash grids in (a) and (c) for arithmetic encoding (§ IV-E). (e) Hash grids are decoded first, then used to model attribute entropy for decoding. (f) `Light4D` is optimized with a rate-distortion joint loss to balance rendering quality and bitstream efficiency (§ IV-G).

A. Binary Hash Grid Representation

Hash grids [27] have been instrumental in the fast training and rendering of 3D scene representation, but it concurrently imposes a storage burden. Inspired by BiRF [35], we introduce an innovative approach by employing binary feature encoding to achieve a storage-efficient and compact representation of attributes. Specifically, the hash feature embeddings are binarized to $\{-1,+1\}$ using a sign function, and backpropagation is carried out through a straight-through estimator (STE) [64]. The 3D voxel hash grid $\mathbf{H}$ consists of binary values $\theta_l^i \in \{-1,+1\}$ across multiple levels l. For each level l, the hash grid $\mathbf{H}_l$ is defined as:

$$\mathbf{H}_l = \{\theta_l^i \mid i = 1,\ldots,T_l\}, \quad \theta_l^i \in \{-1,+1\} \tag{4}$$

where T_l represents the number of entries in the hash table at level l. We obtained final hash feature $\mathbf{H}$ through interpolation within multi-resolution hash grid [27]. We then build our spatiotemporal deformation network (§ IV-B) and context model (§ IV-D) upon on such binary hash grids representation strategy.

B. Spatiotemporal Deformation Network

Based on the attributes A initialized from points derived from SfM, where $A = \{C,R,S,\alpha\}$, we now extend the scene from static to dynamic by introducing a spatiotemporal deformation field, $\delta = \{\Delta X, \Delta R, \Delta S, \Delta\alpha\}$. Inspired by [16], we employ K-planes feature encoder to capture the intricate dynamics of a scene across both spatial and temporal dimensions. The dynamic scene is represented by 4D voxels, which are units in a 4D grid, each containing a set of Gaussians characterized by attributes that vary over time. The 4D voxel grid is encoded by hybrid feature grid that combines both 3D voxels and 2D planes across spatial and temporal axes. There are six 2D feature planes, each spanning a pair of coordinate axes. To achieve a storage-efficient and compact representation, we integrate binary feature encoding into the hybrid feature grid as described in § IV-A. Given a Gaussian $\mathscr{G}$ at position $X = (x,y,z)$ and time t, the features

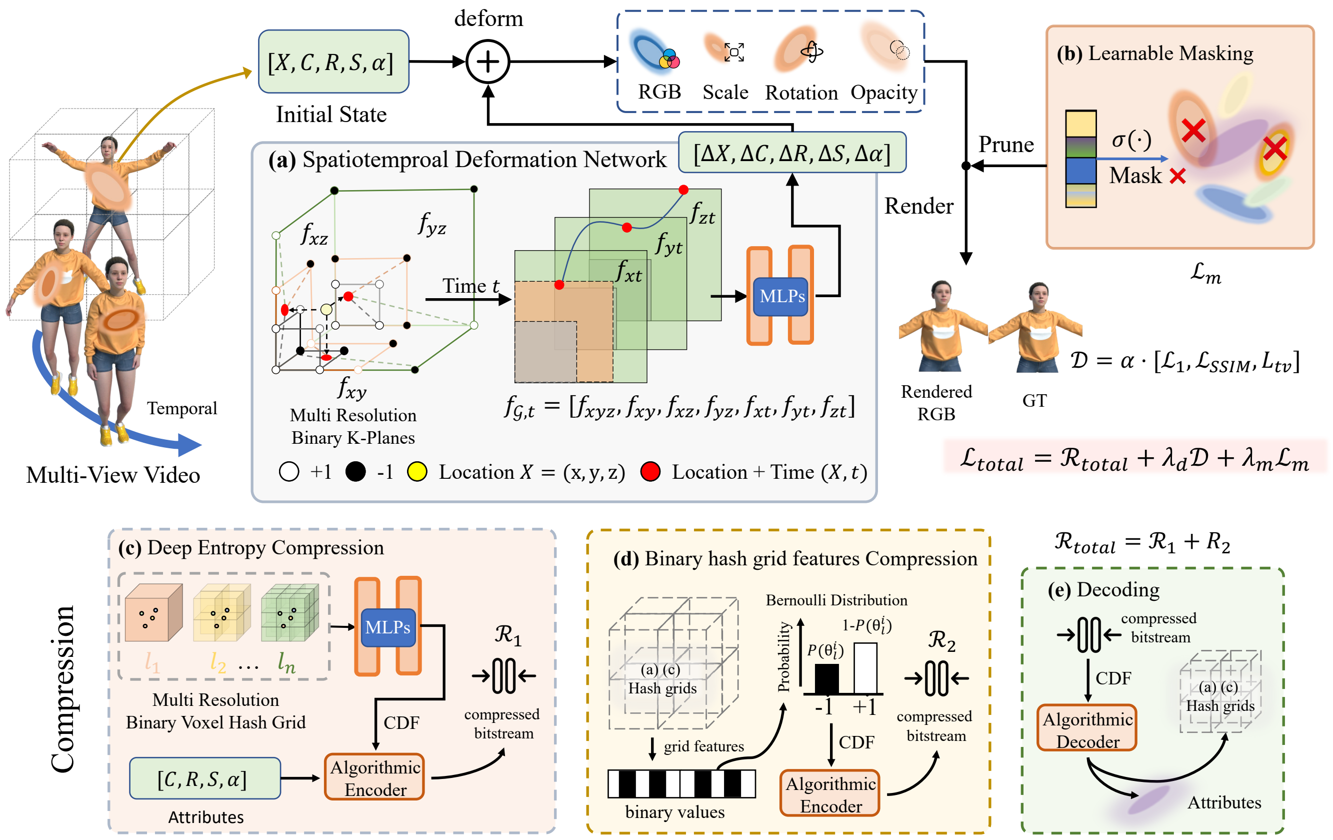


Fig. 3: `Light4D` Framework: a `4D` compression framework for encoding dynamic 3D Gaussian Splatting into `Lightweight` data chunks.

$f_{\mathcal{G},t}$ are computed as follows: For each of the six planes c (representing xy, xz, yz, xt, yt, zt), project the spatiotemporal coordinate $e = (X,t)$ onto the plane:

$$f(e)_c = \psi(P_c, \pi_c(e)) \tag{5}$$

where $\pi_c(e)$ is the projection of e onto the c-th plane, ψ denotes bilinear interpolation on the 2D grid corresponding to the plane P_c. Next, the features from all six planes are combined using the Hadamard product and then concatenated over different resolutions:

$$f_{\mathcal{G},t} = \left[f(e)_{xyz}, \bigcup_{s \in \mathcal{S}} \prod_{c \in \mathcal{C}} f(e)_c^s \right] \tag{6}$$

where $f(e)_{xyz}$ denotes the 3D grid feature, $\mathcal{C}$ represents the set of all six planes, and $\mathcal{S}$ represents the set of all resolutions. The deformation field δ is subsequently obtained through a multi-heads decoder $\Phi(f_{\mathcal{G},t}) = \{\phi_X, \phi_C, \phi_R, \phi_S, \phi_\alpha\}$, where each component decoder $\Phi_i(\cdot)$ generates deformation $\{\Delta X, \Delta C, \Delta R, \Delta S, \Delta\alpha\}$. Finally, the Gaussians are deformed:

$$\mathcal{G}' = \{X + \Delta X, C + \Delta C, R + \Delta R, S + \Delta S, \alpha + \Delta\alpha\} \tag{7}$$

C. *Spatiotemporal Aware Learnable Masking*

Lossy compression of the geometric information X of Gaussians results in a significant degradation of reconstruction quality as the geometric information is notably more sensitive to precision than other attributes. However, effective lossless compression requires complex deep context entropy modeling [65], which is computationally intensive. To address this challenge, we propose a learnable masking strategy to prune Gaussians and then losslessly compress coordinate X. 4D Gaussians represent dynamic scenes over time, and not all Gaussians contribute significantly to the visual fidelity of the rendered scenes across both spatial and temporal dimensions. This redundancy occurs when Gaussians have minimal spatial volume or exhibit little change over time. We extend the volume masking strategy introduced in [6] from static scene to dynamic one. A learnable mask parameter $\theta \in \mathbb{R}^N$ is introduced and followed by a tiny MLP ϕ_m, based on which we generate binary masks $M \in \{0,1\}^N$. The masking process evaluates both the spatial significance and the consistency of the Gaussian's attributes over time. The binary mask is generated as follows:

$$M = \mathrm{sg}(\mathbb{I}[(\sigma(\phi_m(\theta,t)) > \varepsilon] - \sigma(\phi_m(\theta,t))) + \sigma(\phi_m(\theta,t)) \tag{8}$$

where $\mathrm{sg}(\cdot)$ is the stop gradient operator, $\mathbb{I}(\cdot)$ is the indicator function, $\sigma(\cdot)$ is the sigmoid function, and ε is the masking threshold. We use the masking loss L_m in [6] to balance the accurate rendering and the number of Gaussians eliminated during training.

D. Deep Context Modeling for Entropy Coding

Ballé et al. [13] employed deep learning methodologies to gauge the entropy of data targeted for compression. In the realm of information theory, entropy quantifies the inherent uncertainty or randomness in a dataset, acting as a constraint on the maximal average compression rate attainable by any lossless compression algorithm applied to specific data. Once the entropy is discerned, an Algorithmic Encoder (AE) [66], [67] proceeds to compress the data losslessly, adhering to the entropy values.

We design a deep entropy model to estimate the conditional probability of attributes of Gaussian primitives, incorporating both the learnable binary 3D hash grid feature f_e and a set of MLPs parameterized by Θ_e. The feature f_e is the input to the MLPs, which are optimized to maximize the conditional probability $p(A|f_e;\Theta_e)$. Both f_e and Θ_e are optimized using a rate-distortion loss function [13], as detailed in § IV-G.

E. Hash Grid Compression with Bernoulli Modeling

Efficient compression of the hash grid is crucial for reducing storage requirements, as hash grid features occupy a significant portion of the storage. Inspired by [35], the binary nature of each hash grid entry θ_l^i allows us to employ a Bernoulli distribution to accurately model the occurrence of $+1$ (or -1), i.e., $\Theta_l^i \sim \text{Bernoulli}(p)$. The probability p that a given hash grid entry θ_l^i is $+1$ is estimated based on the empirical distribution of the binary values across the entire hash grid:

$$p = \frac{\sum_{i=1}^{N} \mathbf{1}(\theta_l^i = +1)}{N} \tag{9}$$

where N is the total number of entries and $\mathbf{1}(\cdot)$ is the indicator function. This estimation allows us to precisely define the distribution of Θ_l^i for subsequent compression steps.

F. Entropy and Bitstream Calculation

The entropy for each entry θ_l^i in the hash grid is calculated using the Bernoulli distribution:

$$H(\theta_l^i) = -(p(\theta_l^i)\log_2(p(\theta_l^i)) + (1-p(\theta_l^i))\log_2(1-p(\theta_l^i))) \tag{10}$$

Finally, the bitstream required to encode the entire hash grid across all levels is obtained using the AE, which leverages the entropy values to achieve efficient compression:

$$R_H = \frac{1}{N}\sum_{l=1}^{L}\sum_{i=1}^{T_l} H(\theta_l^i) \tag{11}$$

where N is the total number of binary features across all levels of the hash grid. Similarly, the bitstream required to encode the attributes A, based on the learned feature f_e and the model parameters Θ_e, is calculated as the entropy of A:

$$R_a = -\sum_{A} p(A|f_e;\Theta_e)\log p(A|f_e;\Theta_e) \tag{12}$$

Finally, the total bitstream required for the entire system is the sum of the bitstreams for both the attributes A and the hash grid:

$$R = R_a + R_H \tag{13}$$

G. Loss Function

Our model is optimized using a rate-distortion joint loss that balances rendering quality and bitstream efficiency. This loss function integrates entropy loss R, distortion loss D and masking loss L_m to ensure high-quality output while controlling bitrate consumption:

$$L = R + \lambda_d D + \lambda_m L_m \tag{14}$$

where λ_d and λ_m control the trade-off between rate and distortion.
Distortion Loss. D measures the reconstruction accuracy:

$$D = \alpha_1 L_1 + \alpha_2 L_{SSIM} + \alpha_3 L_{tv} \tag{15}$$

where L_1 is L1 norm difference between rendered and ground truth images, L_{SSIM} measures loss of structural similarity, and L_{tv} is a grid-based total-variational loss [68].

V. Bitrate Online Adaption for Dynamic 3DGS Scene Streaming

In this section, we outline the design of the streaming optimization strategy in `4DGStream`. We begin by explaining how ABR streaming operates in dynamic 3DGS scene streaming in § V-A, followed by the formulation of the optimization problem in § V-B. Finally, we introduce `QoSmooth` as a solution to this problem in § V-E.

A. ABR Streaming Modeling

Scene Model. The GSV is divided into N chunks, each representing a fixed time window τ. These chunks are encoded at multiple bitrate levels controlled by distortion parameters, producing M bitrate versions per chunk. The m-th version of the i-th chunk, denoted as $c_{i,m}$, has a bitrate b_m. Higher bitrates provide better visual quality, represented by u_m. such that:

$$u_1 \le u_2 \le \cdots \le u_M. \tag{16}$$

Client Buffer Dynamics. At the client side, a buffer stores downloaded chunks of the 4DGS scene, ready for playback, with a maximum capacity of $Q_{\max}$ chunks. Chunks are consumed at a fixed rate of one every τ seconds, where τ is the chunk duration. Each downloading slot n has a duration T_n, corresponding to the download time of the selected chunk. The download rate depends on the available average network bandwidth $\bar{c}_n$ during the slot n, which fluctuates due to factors like congestion or signal strength. The time to download a chunk at bitrate b_m at time slot n, denoted as T_n^m, is given by:

$$T_n^m = \frac{b_m}{\bar{c}_n} \tag{17}$$

The buffer level at slot n, denoted by $Q(n)$, evolves as chunks are consumed and downloaded. The buffer level $Q(n)$ is updated based on the playback rate and the download rate as follows:

$$Q(n+1) = \max\left(Q(n) - \frac{T_n^m}{\tau}, 0\right) + x_m(n) \tag{18}$$

where $x_m(n) \in \{0,1\}$ indicates whether a chunk at bitrate b_m finishes downloading within the slot n. The term $\frac{T_n^m}{\tau}$ represents how many chunks are consumed by playback during the download time T_n^m. This equation accounts for the removal of chunks for playback and the arrival of newly downloaded chunks.

To ensure smooth playback, the system must avoid buffer underflow (i.e., running out of chunks and causing rebuffering) and buffer overflow (i.e., exceeding the buffer capacity $Q_{\max}$, which wastes network resources). Hence, the system dynamically adjusts the bitrate b_m for new chunk downloads based on the current buffer level $Q(n)$ and the available average bandwidth $\overline{c}(n)$ to minimize rebuffering and maintain continuous playback.

B. Problem Formulation

The objective of the 4DGS streaming system is to maximize the playback performance, defined as a combination of maximizing visual quality and minimizing interruptions due to rebuffering. This goal is framed as an optimization problem balancing *time-average utility* and *smooth playback*. Here, time-average utility measures the average quality of the streamed content based on the chosen bitrates, while playback smoothness reflects the proportion of uninterrupted playback time relative to the total duration (including rebuffering).

We consider a discrete horizon of N time steps. Over these N steps, the time-average normalized utility is defined as:

$$U(N) = \frac{1}{N}\sum_{n=1}^{N} \frac{x_m(n)u_m}{u_M} \tag{19}$$

where $m \in \{1,2,\dots,M\}$ is the index of the bitrate chosen at time step n, and u_M is the utility of the highest available bitrate. This term captures the cumulative utility derived from the quality of the downloaded chunks over the period N. While playback smoothness is defined as the ratio of time spent playing chunks to the total time, including rebuffering:

$$S(N) = \frac{k\cdot\tau}{T} \tag{20}$$

where k is the number of played-out chunks, τ is the playback time of each chunk, and T is the total real time elapsed, including both normal playback and any rebuffering. To achieve both high quality and smooth playback, the system aims to maximize the joint utility:

$$\max\left(U(N) + \alpha S(N)\right) \tag{21}$$

where α is a tunable parameter balancing smoothness and quality.

C. Relaxation of the Problem and Rate Stability

Managing the buffer under dynamic network conditions with finite size constraints $Q_{\max}$ is complex, particularly because network bandwidth $\overline{c}_n$ fluctuates unpredictably over time. Traditional dynamic programming (DP) methods, which attempt to make optimal decisions over time, require precise knowledge of the distribution of the bandwidth process $\overline{c}_n$. Specifically, DP methods rely on forecasting future states to compute expected long-term rewards or costs. However, in real-world scenarios, accurately predicting the distribution of $\overline{c}_n$ is infeasible due to its inherent variability and unpredictability. The complexity of predicting future bandwidth makes traditional DP approaches impractical for real-time video streaming systems, where decisions must be made quickly without knowing future conditions.

To address this challenge, we introduce a problem relaxation by focusing on rate stability rather than imposing strict buffer constraints at every time step. Instead of directly enforcing $0 \le Q(n) \le Q_{\max}$, the system is designed to maintain long-term balance between the download rate and the playback rate, ensuring that the buffer neither overflows nor depletes on average over time. This approach simplifies the problem by avoiding the need for exact knowledge of $\overline{c}_n$. The rate stability condition is expressed as:

$$\lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N} x_m(n)\tau \le \lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N} T_n^m \tag{22}$$

where $m \in \{1,2,\dots,M\}$ is the index of the bitrate chosen at time step n, and the left-hand side represents the time-average playback rate and the right-hand side represents the time-average download rate. The system aims to ensure that, over time, the rate of chunk consumption does not exceed the rate of chunk downloading, thus preventing both buffer overflow and underflow.

D. Optimization Objective

The system must maximize the time-average utility $U(N)$ while maintaining smooth playback $S(N)$:

$$\max_{x_m(n)}\ \lim_{N\to\infty}\left(U(N) + \gamma S(N)\right) \tag{23}$$

subject to rate stability constraint Eq. 22. Here, γ adjusts the trade-off between prioritizing high video quality and minimizing rebuffering.

E. `QoSmooth`*: Lyapunov Optimization-based for Bitrate Online Adaption*

To solve this problem, we propose `QoSmooth`, a method designed to deliver a **smooth** playback experience with a focus on **Qo**E. It leverages Lyapunov optimization to ensure buffer stability while dynamically optimizing chunk selection. The Lyapunov function, $L(Q(n))$, quantifies the buffer's deviation from an ideal state:

$$L(Q(n)) = \frac{1}{2}Q(n)^2 \tag{24}$$

penalizing large deviations that may lead to rebuffering or buffer overflow. The Lyapunov drift measures the expected change in $L(Q(n))$ over time:

$$\Delta L(Q(n)) = \mathbb{E}\left[L(Q(n+1)) - L(Q(n)) \mid Q(n)\right] \tag{25}$$

The drift term $\Delta L(Q(n))$ quantifies how much the buffer deviates over time, and the goal is to minimize this deviation while maximizing utility. To capture this trade-off, we introduce a drift-plus-penalty formulation:

$$\Delta L(Q(n)) - \beta\cdot x_m(n)u_m \tag{26}$$

where β is a control parameter that balances buffer stability and utility maximization. To derive the control policy, we minimize the drift-plus-penalty expression:

$$\min_{x_m(n)} \mathbb{E}\left[\frac{Q(n+1)^2 - Q(n)^2}{2}\right] - \beta \cdot x_m(n) u_m \tag{27}$$

This policy dynamically selects the bitrate b_m for each chunk by evaluating the impact on both the buffer $Q(n)$ and the utility u_m, ensuring smooth playback and high video quality. The algorithm adjusts chunk selection based on current buffer occupancy and network conditions.

Rebuffering and Buffer Overflow Penalties. To further improve system performance, penalties can be introduced for undesirable events such as rebuffering and buffer overflow. Rebuffering occurs when the buffer is depleted, leading to playback stalls. To penalize this, we define a rebuffering penalty:

$$P_{\mathrm{r}}(n) = \lambda \cdot \mathbb{I}(Q(n) = 0) \tag{28}$$

where λ is a penalty coefficient that discourages buffer depletion, and $\mathbb{I}(Q(n) = 0)$ is an indicator function that activates when the buffer is empty. Similarly, buffer overflow, where the buffer exceeds its capacity $Q_{\max}$, results in wasted bandwidth. This can be penalized using:

$$P_o(n) = \mu \cdot \mathbb{I}(Q(n) \geq Q_{\max}) \tag{29}$$

where μ penalizes downloading chunks that cannot be stored.

Incorporating both the rebuffering and buffer overflow penalties into the optimization framework:

$$\min_{x_m(n)} \mathbb{E}\left[\frac{Q(n+1)^2 - Q(n)^2}{2} + \lambda \cdot P_r(n) + \mu \cdot P_o(n)\right] - \beta \sum_{m=1}^{M} x_m(n) u_m \tag{30}$$

subject to the rate stability constraint Eq. 22

Real-Time Decision Making. To address this optimization problem in real time, the system continuously monitors both the buffer level $Q(n)$ and the available network bandwidth $\overline{c}(n)$ at each time slot n. Based on these inputs, the system dynamically selects the bitrate b_m for each video chunk using the drift-plus-penalty framework described previously.

VI. Experiments

In this section, we begin by detailing the experimental setting of our `4DGStream` framework, followed by a series of evaluation experiments to benchmark `Light4D` and `4DGStream` against existing SOTA methods. We also perform ablation studies to highlight the contribution of each technical component in `Light4D`.

A. Experimental Setting

1) Implementation Details: Our implementation is built upon the PyTorch framework, with all training and testing carried out on an Intel(R) Xeon(R) Silver 4210 CPU @ 2.20GHz with one NVIDIA A100 GPU with 80GB of memory. Further hyper-parameters can be found in the appendix.

2) Hyperparameter Settings: In Eq. 14, λ_d is set from 5×10^{-5} to 1×10^{-3} for variable bitrates, and λ_m is set to 5×10^{-4}. In Eq. 21, α is set to 1, and $\alpha_1, \alpha_2, \alpha_3$ in Eq. 15 are set to 1, 0.95, 0.95, respectively. λ and μ are both set to 1 and β is set to 2 in Eq. 30. The details of the feature dimension and size of hash feature map in `Light4D` please refer to the Appendix.

3) Network Traces: To simulate network conditions, we collected 1,000 throughput traces, each lasting between 200 and 1,000 seconds, from two real-world datasets: (1) the FCC [69] dataset, which includes over one million throughput traces gathered in natural environments; and (2) the 3G/HSDPA dataset [70], encompassing mobile throughput data across various usage scenarios such as buses, trains, and urban areas. To avoid scenarios where bitrate selection is trivial—specifically, situations where selecting the maximum bitrate is always optimal or where the network cannot support any available bitrate for extended periods—we filtered the original traces based on the compressed data size to be transmitted. Specifically, we chose traces whose average throughput ranged from 0.75 to 1 times 5‰ of the size required to transmit 30 frames, assuming a frame rate of 30 FPS. The chunk size was set to 2 seconds, and the buffer size was set to 20 seconds.

4) Training: We adopt a progressive training strategy to get a more stable training performance. In the first 3,000 iterations, we optimize 3D Gaussians for frames selected from distributed intervals without deformation, ensuring stable training of Gaussian attributes and distribution across spatiotemporal regions. This reduces the burden on the deformation network for later stages. From iteration 3,000, we introduce spatiotemporal deformation, allowing joint training with the deformation network. At iteration 20,000, we incorporate entropy loss to optimize bitstream consumption. The training concludes after 40,000 iterations, achieving a balance between Gaussian attributes, deformation, and compression efficiency.

5) Datasets: We follow the methodology outlined in 4DGS [4] to evaluate our approach across multiple datasets. These include real-world datasets such as the Neural 3D Video Dataset [14], which comprises six indoor multi-view video sequences captured by 18 to 21 cameras at a resolution of 2704×2028, and the HyperNeRF Dataset [71], captured with fewer than 2 cameras in feed-forward settings. For synthetic data, we experiment with the D-NeRF [20] datasets, designed for monocular settings with random camera poses per timestamp.

6) Baseline methods: **Scene Representation.** To assess the quality of novel view syn-thesis and the storage of the model taken, we compare most recent NeRF-based and Gaussian Splatting-based SOTAs in the field with `Light4D`, including TiNeuVox-B [72], StreamNeRF [23], DyNeRF [14], Im4D [73], KPlane [16], HexPlane-Slim [68], 3DGS [1], V4D [18], NeRFPlayer [15], HyperReel [74], MixVoxels [75], FFDNeRF [17], MSTH [76], 4DGS [4], Spacetime [5], CompD3D [60], E-D3DGS [77], QUEEN [61]. The results from these baseline methods originate from their papers, and some are from papers [4], [5].

System. To the best of our knowledge, we are the first to propose a distortion joint optimization framework for streaming GSV. To validate the performance of `4DGStream`, we design

TABLE I: Quantitative results on the synthesis dataset. The **best** and the **second best** results are denoted by pink and yellow. The rendering resolution is set to 800×800. "Time" in the table stands for training times.

Model	PSNR (dB) ↑	SSIM ↑	LPIPS ↓	Time ↓	FPS ↑	Storage (MB) ↓	FPS/Storage ↑
TiNeuVox-B	32.67	0.97	0.04	28 mins	1.5	48	0.03125
KPlanes	31.61	0.97	-	52 mins	0.97	418	0.000255
HexPlane-Slim	31.04	0.97	0.04	11m 30s	2.5	38	0.06579
3DGS	23.19	0.93	0.08	10 mins	170	10	17.00
FFDNeRF	32.68	0.97	0.04	-	< 1	440	0.00227
MSTH	31.34	0.98	0.02	6 mins	-	-	-
V4D	33.72	0.98	0.02	6.9 hours	2.08	377	0.00552
4DGS	34.05	0.98	0.02	8 mins	82	18	4.56
CompD3D	32.19	0.97	0.04	8 mins	150	$\sim$ 159	0.94
Ours-Lite	33.12	0.97	0.03	19 mins	69.2	1.6	43.25
Ours-Medium	33.45	0.98	0.02	29 mins	61.7	2.5	24.68
Ours-Large	33.89	0.98	0.02	41 mins	47.6	4.1	11.60

three baseline methods for comparison, drawing from existing SOTA approaches and well-developed tools in the industry:

- Vanilla Streamable 3DGS (denoted as 3DGS†): Inspired by DynamicGaussian [3] and MGA [12], we reconstruct 3DGS on a per-frame basis, creating a sequence of 3DGS representations to form a volumetric video. To reduce storage consumption, we apply the strategy from [11], achieving a 25× reduction in overall size. Further, we customize the third-party compression codec Draco [78], originally designed for compressing 3D objects like point clouds and meshes, to compress 3D Gaussians into variable bitrates. Since different attributes exhibit varying sensitivities to compression, careful parameter selection for each attribute is necessary. To handle this, we employ a similar Lyapunov optimization approach as described in [57], enabling dynamic parameter selection under fluctuating network conditions to ensure smooth playback while preserving visual quality.
- 3DGStream [30]: An innovative approach that leverages 3D Gaussians and a neural transformation cache for efficient and rapid streaming of GSV in photo-realistic free-viewpoint videos.
- Streamable 4DGS (denoted as 4DGS†): 4DGS [4] is a SOTA method that reconstructs GSV with high visual quality while maintaining a small storage footprint. We generate multiple bitrate levels of scene representation by capping the number of Gaussian primitives and adjusting the size of the hash grid feature in the deformation network. This is combined with the streaming strategy `QoSmooth` (§ V-E) to enhance streaming performance.

To further reduce storage requirements of 4DGS† and 3DGStream, we apply a codebook encoding strategy [11], enhancing their practicality in streaming scenarios. More implementation details of baseline methods please refer to the Appendix of our full paper.

7) Evaluation Metrics: **Visual quality.** Following common standards, training and evaluation are performed at half resolution, with the initial camera reserved for evaluation. Evaluation metrics include peak-signal-to-noise ratio (PSNR), perceptual quality measure LPIPS [79], structural similarity index (SSIM) [80] and its extensions dissimilarity index measure (DSSIM). For DSSIM, we use the structural similarity function from scikit-image library, and set data range to 2.0.

Evaluation metrics of video streaming. We evaluate the effectiveness of the video streaming system through measuring various metrics that reflect QoE for the user, including average video quality, playback smoothness, average rendering speed over time duration T. More specifically, average video quality is quantified by the average PSNR, denoted as $V(N)$ and playback smoothness is measured by the ratio of playback time without rebuffering $S(N)$ in Eq. 20. We treat average rendering speed $P(N)$ as a summation operator for the weighted combination of $V(N)$ and $S(N)$ to evaluate its overall impact on QoE. We combine these key QoE metrics into a single score that reflects the overall experience:

$$\text{QoE} = \sum_{P(N)} (\alpha \cdot V(N) + \beta \cdot S(N)) \tag{31}$$

where α and β are the importance weight of each factor and are both set to 1 in our experiments.

B. *Experiment Evaluation:* `Light4D`

1) Performance on Synthetic Dataset: The results in synthetic dataset [20] are summarized in Tab. I. While current dynamic hybrid representations can produce high-quality results, they often come with the drawbacks of slow rendering speeds and high data storage requirements. Although 3DGS [1] is fast and compact, it fails to reconstruct GSV due to its lack of dynamic motion modeling. 4DGS [4] enjoys both the highest rendering quality within the synthetic dataset and fast rendering speeds but its storage consumption is still non-negligible. We report our methods in three different sizes, balancing reconstruction quality and model size. Specifically, we adjust the parameters λ_d, λ_m, and the warm-up steps during training to achieve different model sizes; for more details, please refer to our supplementary file. Our `Lite` version achieves comparable visual quality (with only a 2.7% drop in PSNR compared to 4DGS), while requiring just 8.9% of the storage consumed by 4DGS. This results in a 9.5× improvement in FPS/Storage. By increasing the model size to 22.7%, the `Large` version achieves 99.5% of PSNR of 4DGS and achieves a 2.54× improvement in FPS/Storage. These enhancements demonstrate the effectiveness of our models in balancing quality and storage efficiency.

TABLE II: Quantitative comparisons on Neu3D's dataset. The **best** and the **second best** results are denoted by pink and yellow.

Method	PSNR (dB) ↑	DSSIM ↓	LPIPS ↓	Time ↓	FPS ↑	Storage (MB)↓	FPS/Storage ↑
DyNeRF	29.58	0.020	0.083	-	0.015	28	$5e^{-4}$
HexPlane	31.71	-	0.075	-	-	200	-
StreamRF	28.26	-	-	-	10.9	5310	0.002
NeRFPlayer	30.69	-	0.111	6 hours	0.05	5130	$< 9e^{-6}$
HyperReel	31.10	-	0.096	9 hours	2	360	0.006
HexPlane-all *	31.70	0.014	0.075	12 hours	0.2	250	$8e^{-4}$
K-Planes	31.63	0.018	-	1.8 hours	0.3	311	$9e^{-4}$
Mix Voxels-L	31.34	0.017	0.096	-	37.7	500	0.075
Mix Voxels-X	31.73	0.015	0.064	-	4.6	500	0.0092
Im4D	32.58	-	0.208	-	5	93	0.053
3DGS	32.08	-	-	41.5 hours	390	14130	0.027
3DGStream	31.67	-	-	1 hour	215	2280	0.094
SpacetimeGS	32.05	0.014	0.044	≈ 18.5 mins †	140	200	0.7
4DGS	31.15	0.016	0.049	40 mins	30	90	0.33
CompD3D	30.46	-	0.15	1 hour	118	338	0.349
E-D3DGS	31.31	-	0.037	1.87 hours	74.5	35	2.12
QUEEN-s	31.89	-	0.139	23.25 mins	345	204	1.69
QUEEN-m	32.03	-	0.137	29.8 mins	321	207	2.55
QUEEN-l	32.19	-	0.136	39.5 mins	248	225	1.10
Ours-Lite	30.92	0.017	0.053	45 mins	27	3.1	8.7
Ours-Medium	31.53	0.016	0.049	58 mins	25	4.2	5.95
Ours-Large	31.88	0.016	0.047	1.1 hours	22	5.6	3.92

* The metrics of the model are tested without "coffee martini" and resolution is set to 1024×768.
† Corresponds to the model trained on longer sequence on the Flame Salmon scene.

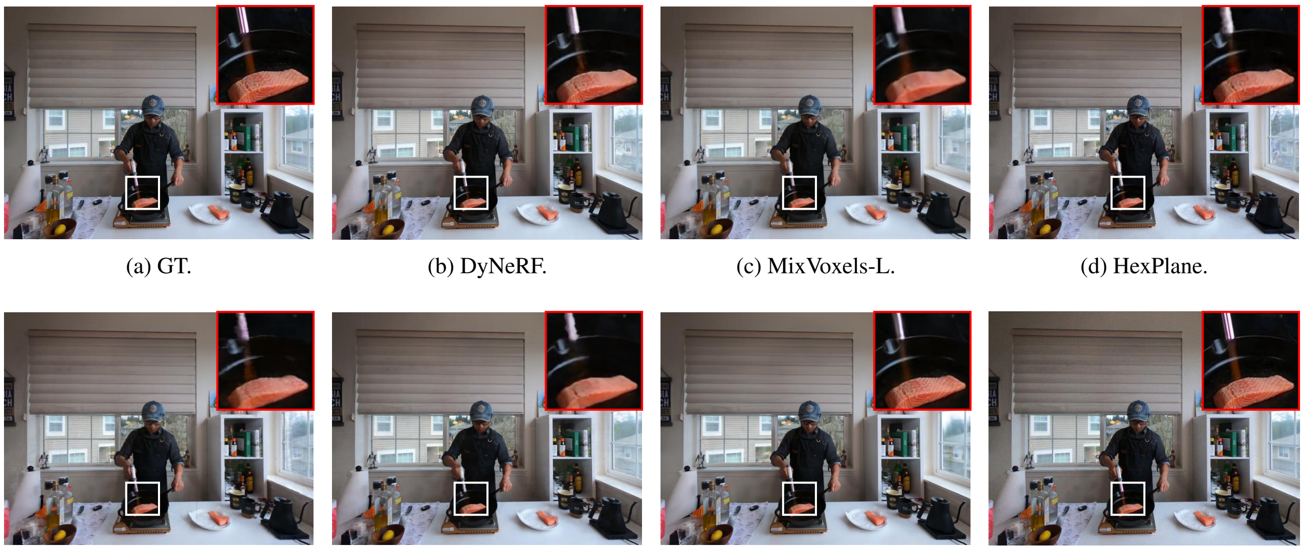
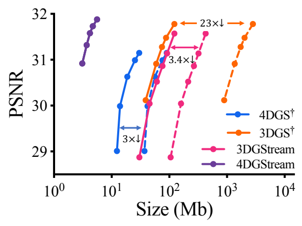

(a) GT. (b) DyNeRF. (c) MixVoxels-L. (d) HexPlane.
(e) HyperReel. (f) K-Plane. (g) SpaceGS. (h) Ours.

Fig. 4: Qualitative comparisons on the Neural 3D Video Dataset.

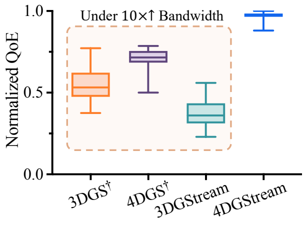
TABLE III: Ablation study on different components.'C', 'M', and 'H' denote deep compression for attributes, learnable masking, binary hash grid feature, respectively. '#Gauss' denotes the number of Gaussians.

Method \Dataset			Flame Salmon					Flame Steak				
C	M	H	PSNR	#Gauss	Storage	FPS	FPS/Storage	PSNR	#Gauss	Storage	FPS	FPS/Storage
Vanilla			30.87	109 K	93 MB	33.7	0.66	31.65	97 K	89 MB	35	1.37
✓			30.87	131 K	16 MB	29.4	1.83	31.65	118 K	17 MB	30.5	1.79
✓	✓		31.23	87 K	15 MB	30.3	2.02	32.08	76 K	16 MB	31.7	1.98
✓	✓	✓	31.25	92 K	5.9 MB	28.1	2.81	32.36	94 K	5.4 MB	27.9	2.79

2) Performance on Neu3D's Dataset: We summarize the quantitative results in Tab. II and the qualitative comparisons in Fig. 4. Our methods achieve similar reconstruction quality while the `Lite` version requires only 3.3%, 1.6%, and 3.4% of the data storage used by Im4D, SpacetimeGS, and 4DGS, respectively. The `Large` version offers the third-best visual

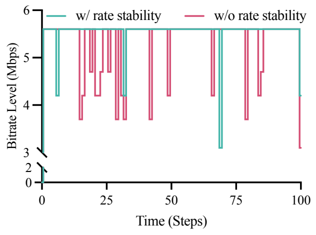


(a)

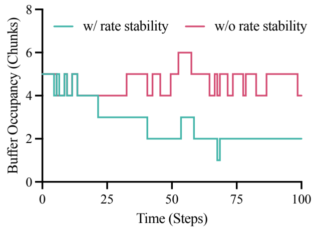


(b)

Fig. 5: Results of Neu3D's Dataset Streaming. (a) Visual Quality vs. Size Curve of Methods: Visual quality is measured in PSNR, and size is calculated for one 2-second chunk across various bitrate levels. (b) Comparison of Overall Normalized QoE: Baseline methods are tested under 10× larger bandwidth.



(a)



(b)

Fig. 6: Comparison of selected bitrate levels (a) and buffer occupancy (b) with and without rate stability control in `QoSmooth` using FCC dataset [69]

quality while requiring just 6.0%, 2.7%, and 6.2% of the data storage compared to these methods.

3) Performance on HyperNeRF's Dataset: We present the quantitative results on HyperNeRF's dataset in Tab. IV, respectively. Our `Large` version achieves the best reconstruction quality while maintaining relatively small storage consumption (9.6% of that of 4DGS). Additionally, the `Medium` version has the second smallest data size while achieving reconstruction quality comparable to that of 4DGS.

C. Analysis of Streaming Performance of `4DGStream`

Chunk size. We generate multiple bitrate levels for each method by controlling the number of Gaussian primitives and adjusting model size. To make the baseline methods more compact and streamable, we apply effective pruning, quantization, and codebook encoding strategies from [11]. Fig. 5a shows the bitrate levels for each method, with the dashed line representing the original uncompressed versions of the baseline methods. Despite significant efforts to reduce storage requirements, achieving approximately 3×, 3.4×, and 23× data reduction for 4DGS†, 3DGStream, and 3DGS†, respectively, the baseline methods remain at least an order of magnitude larger than our proposed method `4DGStream`.

Rendering Speed. Rendering speed is a critical metric that strongly impacts QoE, as outlined in our QoE model (Eq. 31). Tab. II presents the average rendering speed for each method. While `Light4D` may have slower rendering performance, its compactness compensates for the potential QoE drop caused

TABLE IV: Quantitative results on HyperNeRF's vrig dataset. Rendering resolution is set to 960×540. The **best** and the **second best** results are denoted by pink and yellow.

Model	PSNR ↑	MS-SSIM ↑	FPS ↑	Storage↓
Nerfies	22.2	0.803	< 1	-
HyperNeRF	22.4	0.814	< 1	-
TiNeuVox-B	24.3	0.836	1	48
3DGS	19.7	0.680	55	52
FFDNeRF	24.2	0.842	0.05	440
V4D	24.8	0.832	0.29	377
4DGS	25.2	0.845	34	61
CompD3D	25.6	0.89	188	∼ 720
E-D3DGS	25.43	-	139.3	33
Ours-Lite	24.2	0.833	27	2.3
Ours-Medium	24.9	0.840	24	3.6
Ours-Large	25.4	0.847	19	5.9

TABLE V: Average Total Runtime (hash grid coding time) in seconds for encoding/decoding across different datasets under varying λ_d. The values are displayed in the format: avg. (std.).

λ_d \Dataset	Synthesis	Neu3D's	HyperNeRF's
1e-3	∼0.09 (0.04)	∼0.15 (0.09)	∼0.16 (0.1)
5e-4	∼0.09 (0.04)	∼0.14 (0.09)	∼0.15 (0.09)
1e-4	∼0.09 (0.04)	∼0.13 (0.08)	∼0.15 (0.09)
5e-5	∼0.08 (0.03)	∼0.13 (0.08)	∼0.14 (0.09)

TABLE VI: Comparison of average performance with and without rate stability control in `QoSmooth` using entire network traces. The values are displayed in the format: avg. (std.).

	$U(N)$	$S(N)$	Bitrate ↑	Buffer ↓	Swiches ↓
w/ Stability	0.978 (0.02)	0.948 (0.03)	5.479 (0.44)	0.324 (1.63)	0.06 (<0.01)
w/o Stability	0.904 (0.07)	0.976 (0.01)	5.08 (0.53)	0.550 (1.44)	0.161 (0.04)

by the slower rendering. In contrast, the higher rendering speeds of 3DGStream and 3DGS† help offset the QoE reduction caused by the transmission latency due to their larger data sizes.

Coding Time. `4DGStream` is the first to integrate a distortion joint optimization framework into GSV streaming, requiring additional encoding and decoding compared to baseline methods due to entropy modeling and algorithmic coding. Our codec proceeds in two stages. *First*, during continuous-attribute compression (§ IV-D), a single forward pass of the binary 3D hash-grid feature f_e through a lightweight set of MLPs Θ_e produces the probability model for each attribute; this step accounts for roughly 70% of the total encoding/decoding time. *Second*, in hash-grid feature compression (§ IV-E), the binary nature of hash grid values allows for fast encoding/decoding using the Bernoulli distribution. The Bernoulli parameter$p$ is computed in $O(N)$ time by counting the proportion of positive entries, as shown in Eq. 9. The aggregate runtimes are reported in Table V. In our experiments, the coding time for each chunk is only 5% to 10% of the chunk duration, making `4DGStream` highly suitable for real-time streaming.

Comparison on overall QoE. The baseline methods were tested under a bandwidth setting that was 10× higher (approximately 40 to 60 Mbps) than that used for `4DGStream`.

Despite this significant advantage for the baseline methods, 4DGStream still achieved QoE improvements of 36.71%, 76.47%, and 158.73% compared to 4DGS†, 3DGS†, and 3DGStream, respectively, when considering all key metrics from the QoE model in Eq. 31.

D. Ablation Study

Effectiveness of each component in Light4D. To assess the effectiveness of the proposed components in Light4D, we performed an ablation study summarized in Tab. III, using the Flame Salmon and Flame Steak scenes from Neu3D's dataset. We used a customized version of 4DGS, incorporating hash grid features [27], as the baseline for our method. The component labeled 'C' introduces deep entropy coding for attributes. This approach results in an average data size reduction of 87.2%. The spatiotemporal learnable masking component further reduces less important Gaussians over time, contributing an additional decrease in data size and improving visual quality by eliminating noisy Gaussians. Binarizing the hash grid values enables precise Bernoulli distribution modeling of the CDF for AE compression, resulting in a 12.7% reduction in data size.

Effectiveness of rate stability control in QoSmooth. Fig. 6a and Fig. 6b showcase a detailed comparison of the streaming statistics trending curve between with and without using rate stability control using FCC dataset, while Tab. VI illustrates the detailed metrics using the entire network traces. With the help of rate stability control (§ V-E), the selected bitrate remains stable even under high bandwidth variance (up to 30%), with the number of bitrate switches (proportion of changes across consecutive chunks) as low as 6.1%, representing a 62.5% reduction compared to the case without rate stability control. Additionally, rate stability control improves the time-average utility $U(N)$ by 8.1%, ensuring a stable playback experience while maintaining high visual quality.

VII. Conclusion

We proposed 4DGStream, a comprehensive framework that integrates Light4D for compact dynamic 3D scene representation and QoSmooth for effective online bitrate adaptation during video streaming. Light4D efficiently compresses dynamic 3D scenes using binarization-assisted spatiotemporal optimization, incorporating binary 3D hash grids, spatiotemporal deformation networks, and a spatiotemporal-aware masking strategy. These innovations result in over $10\times$ data reduction compared to existing SOTA methods on synthetic datasets, and reduce storage requirements to as low as 3.3% on real-world datasets, all without compromising visual quality or rendering speed. QoSmooth significantly enhances playback smoothness by reducing bitrate level switches by 61.6%, while maintaining high visual quality by increasing time-average utility by 26.2%. With all these advantages combined, 4DGStream achieves QoE improvements of up to 158.73% compared to baseline methods, making it well-suited for real-time streaming applications with limited storage and network bandwidth.

References

[1] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering." *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[2] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[3] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan, "Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis," *arXiv preprint arXiv:2308.09713*, 2023.

[4] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang, "4d gaussian splatting for real-time dynamic scene rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 310–20 320.

[5] Z. Li, Z. Chen, Z. Li, and Y. Xu, "Spacetime gaussian feature splatting for real-time dynamic view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 8508–8520.

[6] J. C. Lee, D. Rho, X. Sun, J. H. Ko, and E. Park, "Compact 3d gaussian representation for radiance field," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 719–21 728.

[7] W. Morgenstern, F. Barthel, A. Hilsmann, and P. Eisert, "Compact 3d scene representation via self-organizing gaussian grids," *arXiv preprint arXiv:2312.13299*, 2023.

[8] Y. Chen, Q. Wu, J. Cai, M. Harandi, and W. Lin, "Hac: Hash-grid assisted context for 3d gaussian splatting compression," *arXiv preprint arXiv:2403.14530*, 2024.

[9] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai, "Scaffold-gs: Structured 3d gaussians for view-adaptive rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 654–20 664.

[10] Z. Fan, K. Wang, K. Wen, Z. Zhu, D. Xu, and Z. Wang, "Lightgaussian: Unbounded 3d gaussian compression with 15x reduction and 200+ fps," *arXiv preprint arXiv:2311.17245*, 2023.

[11] P. Papantonakis, G. Kopanas, B. Kerbl, A. Lanvin, and G. Drettakis, "Reducing the memory footprint of 3d gaussian splatting," *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 7, no. 1, pp. 1–17, 2024.

[12] Y.-C. Sun, Y. Shi, W. T. Ooi, C.-Y. Huang, and C.-H. Hsu, "Multi-frame bitrate allocation of dynamic 3d gaussian splatting streaming over dynamic networks," in *Proceedings of the 2024 SIGCOMM Workshop on Emerging Multimedia Systems*, 2024, pp. 1–7.

[13] J. Ballé, V. Laparra, and E. P. Simoncelli, "End-to-end optimized image compression," in *5th International Conference on Learning Representations, ICLR*, 2017.

[14] T. Li, M. Slavcheva, M. Zollhoefer, S. Green, C. Lassner, C. Kim, T. Schmidt, S. Lovegrove, M. Goesele, R. Newcombe *et al.*, "Neural 3d video synthesis from multi-view video," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5521–5531.

[15] L. Song, A. Chen, Z. Li, Z. Chen, L. Chen, J. Yuan, Y. Xu, and A. Geiger, "Nerfplayer: A streamable dynamic scene representation with decomposed neural radiance fields," *IEEE Transactions on Visualization and Computer Graphics*, vol. 29, no. 5, pp. 2732–2742, 2023.

[16] S. Fridovich-Keil, G. Meanti, F. R. Warburg, B. Recht, and A. Kanazawa, "K-planes: Explicit radiance fields in space, time, and appearance," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 12 479–12 488.

[17] X. Guo, J. Sun, Y. Dai, G. Chen, X. Ye, X. Tan, E. Ding, Y. Zhang, and J. Wang, "Forward flow for novel view synthesis of dynamic scenes," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 16 022–16 033.

[18] W. Gan, H. Xu, Y. Huang, S. Chen, and N. Yokoya, "V4d: Voxel for 4d novel view synthesis," *IEEE Transactions on Visualization and Computer Graphics*, 2023.

[19] V. Sitzmann, J. Thies, F. Heide, M. Nießner, G. Wetzstein, and M. Zollhofer, "Deepvoxels: Learning persistent 3d feature embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2437–2446.

[20] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer, "D-nerf: Neural radiance fields for dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 318–10 327.

[21] E. Tretschk, A. Tewari, V. Golyanik, M. Zollhöfer, C. Lassner, and C. Theobalt, "Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 959–12 970.

[22] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, W. Yifan, C. Lassner, V. Sitzmann, R. Martin-Brualla, S. Lombardi *et al.*, "Advances in neural rendering," in *Computer Graphics Forum*, vol. 41, no. 2. Wiley Online Library, 2022, pp. 703–735.

[23] L. Li, Z. Shen, Z. Wang, L. Shen, and P. Tan, "Streaming radiance fields for 3d video synthesis," *Advances in Neural Information Processing Systems*, vol. 35, pp. 13 485–13 498, 2022.

[24] H. Lin, S. Peng, Z. Xu, Y. Yan, Q. Shuai, H. Bao, and X. Zhou, "Efficient neural radiance fields for interactive free-viewpoint video," in *SIGGRAPH Asia 2022 Conference Papers*, 2022, pp. 1–9.

[25] L. Wang, Q. Hu, Q. He, Z. Wang, J. Yu, T. Tuytelaars, L. Xu, and M. Wu, "Neural residual radiance fields for streamably free-viewpoint videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 76–87.

[26] L. Wang, K. Yao, C. Guo, Z. Zhang, Q. Hu, J. Yu, L. Xu, and M. Wu, "Videorf: Rendering dynamic radiance fields as 2d feature video streams," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 470–481.

[27] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM transactions on graphics (TOG)*, vol. 41, no. 4, pp. 1–15, 2022.

[28] D. Das, C. Wewer, R. Yunus, E. Ilg, and J. E. Lenssen, "Neural parametric gaussians for monocular non-rigid object reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 10 715–10 725.

[29] Y. Lin, Z. Dai, S. Zhu, and Y. Yao, "Gaussian-flow: 4d reconstruction with dynamic 3d gaussian particle," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 136–21 145.

[30] J. Sun, H. Jiao, G. Li, Z. Zhang, L. Zhao, and W. Xing, "3dgstream: On-the-fly training of 3d gaussians for efficient streaming of photo-realistic free-viewpoint videos," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 675–20 685.

[31] Y.-H. Huang, Y.-T. Sun, Z. Yang, X. Lyu, Y.-P. Cao, and X. Qi, "Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4220–4230.

[32] Z. Lu, X. Guo, L. Hui, T. Chen, M. Yang, X. Tang, F. Zhu, and Y. Dai, "3d geometry-aware deformable gaussian splatting for dynamic view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 8900–8910.

[33] Z. Yang, X. Gao, W. Zhou, S. Jiao, Y. Zhang, and X. Jin, "Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 331–20 341.

[34] L. Li, Z. Shen, Z. Wang, L. Shen, and L. Bo, "Compressing volumetric radiance fields to 1 mb," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4222–4231.

[35] S. Shin and J. Park, "Binary radiance fields," *Advances in neural information processing systems*, vol. 36, 2024.

[36] J. Tang, X. Chen, J. Wang, and G. Zeng, "Compressible-composable nerf via rank-residual decomposition," *Advances in Neural Information Processing Systems*, vol. 35, pp. 14 798–14 809, 2022.

[37] D. He, Y. Zheng, B. Sun, Y. Wang, and H. Qin, "Checkerboard context model for efficient learned image compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 771–14 780.

[38] R. Rigamonti, A. Sironi, V. Lepetit, and P. Fua, "Learning separable filters," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 2754–2761.

[39] A. Polino, R. Pascanu, and D. Alistarh, "Model compression via distillation and quantization," *arXiv preprint arXiv:1802.05668*, 2018.

[40] H. Wang, H. Zhu, T. He, R. Feng, J. Deng, J. Bian, and Z. Chen, "End-to-end rate-distortion optimized 3d gaussian representation," *arXiv preprint arXiv:2406.01597*, 2024.

[41] S. Niedermayr, J. Stumpfegger, and R. Westermann, "Compressed 3d gaussian splatting for accelerated novel view synthesis," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 10 349–10 358.

[42] A. T. Nasrabadi and R. Prakash, "Layer-assisted adaptive video streaming," in *Proceedings of the 28th ACM SIGMM Workshop on Network and Operating Systems Support for Digital Audio and Video*, 2018, pp. 31–36.

[43] Y. Jin, J. Liu, and F. Wang, "Ebublio: Edge assisted multi-user 360-degree video streaming," in *2022 IEEE Conference on Virtual Reality and 3D User Interfaces Abstracts and Workshops (VRW)*. IEEE, 2022, pp. 600–601.

[44] Y. Jin, J. Liu, F. Wang, and S. Cui, "Where are you looking? a large-scale dataset of head and gaze behavior for 360-degree videos and a pilot study," in *Proceedings of the 30th ACM International Conference on Multimedia*, 2022, pp. 1025–1034.

[45] K. Hu, H. Yang, Y. Jin, J. Liu, Y. Chen, M. Zhang, and F. Wang, "Understanding user behavior in volumetric video watching: Dataset, analysis and prediction," in *Proceedings of the 31st ACM International Conference on Multimedia*, 2023, pp. 1108–1116.

[46] N. Wu, K. Liu, R. Cheng, B. Han, and P. Zhou, "Theia: Gaze-driven and perception-aware volumetric content delivery for mixed reality headsets," in *Proceedings of the 22nd Annual International Conference on Mobile Systems, Applications and Services*, 2024, pp. 70–84.

[47] R. Cheng, N. Wu, V. Le, E. Chai, M. Varvello, and B. Han, "Magicstream: Bandwidth-conserving immersive telepresence via semantic communication," in *Proceedings of the 22nd ACM Conference on Embedded Networked Sensor Systems*, 2024, pp. 365–379.

[48] K. Hu, Y. Jin, H. Yang, J. Liu, and F. Wang, "Fsvvd: A dataset of full scene volumetric video," in *Proceedings of the 14th Conference on ACM Multimedia Systems*, 2023, pp. 410–415.

[49] Y. Jin, J. Liu, K. Hu, and F. Wang, "A networking perspective of volumetric video service: Architecture, opportunities and case study," *IEEE Network*, 2024.

[50] Y. Jin, X. Duan, K. Hu, F. Wang, and X. Liu, "3d video conferencing via on-hand devices," *IEEE Transactions on Circuits and Systems for Video Technology*, 2024.

[51] K. Hu, Y. Chen, K. Han, B. Li, H. Yang, Y. Jin, J. Liu, and F. Wang, "Livevv: Human-centered live volumetric video streaming system," *IEEE Internet of Things Journal*, 2025.

[52] J. M. Boyce, Y. Ye, J. Chen, and A. K. Ramasubramonian, "Overview of shvc: Scalable extensions of the high efficiency video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 26, no. 1, pp. 20–34, 2015.

[53] L. Xie, Z. Xu, Y. Ban, X. Zhang, and Z. Guo, "360probdash: Improving qoe of 360 video streaming using tile-based http adaptive streaming," in *Proceedings of the 25th ACM international conference on Multimedia*, 2017, pp. 315–323.

[54] P. K. Yadav and W. T. Ooi, "Tile rate allocation for 360-degree tiled adaptive video streaming," in *Proceedings of the 28th ACM International Conference on Multimedia*, 2020, pp. 3724–3733.

[55] J. Song, F. Yang, W. Zhang, W. Zou, Y. Fan, and P. Di, "A fast fov-switching dash system based on tiling mechanism for practical omnidirectional video services," *IEEE transactions on multimedia*, vol. 22, no. 9, pp. 2366–2381, 2019.

[56] J. Liu, B. Zhu, F. Wang, Y. Jin, W. Zhang, Z. Xu, and S. Cui, "Cav3: Cache-assisted viewport adaptive volumetric video streaming," in *2023 IEEE Conference Virtual Reality and 3D User Interfaces (VR)*. IEEE, 2023, pp. 173–183.

[57] Z. Liang, J. Liu, M. Dasari, and F. Wang, "Fumos: Neural compression and progressive refinement for continuous point cloud video streaming," *IEEE Transactions on Visualization and Computer Graphics*, 2024.

[58] D. Zhang, Z. Liang, Z. Cao, L. Wei, D. Wang, and F. Wang, "3dgstreaming: Spatial heterogeneity aware 3d gaussian splatting compression and streaming," *IEEE Internet of Things Journal*, 2025.

[59] D. Zhang, Z. Liang, Z. Cao, D. Wang, and F. Wang, "Srbf-gaussian: Streaming-optimized 3d gaussian splatting," in *Proceedings of the IEEE Conference Virtual Reality and 3D User Interfaces (VR)*, 2025, pp. 461–471.

[60] K. Katsumata, D. M. Vo, and H. Nakayama, "A compact dynamic 3d gaussian representation for real-time dynamic view synthesis," in *European Conference on Computer Vision*. Springer, 2025, pp. 394–412.

[61] S. Girish, T. Li, A. Mazumdar, A. Shrivastava, S. De Mello *et al.*, "Queen: Quantized efficient encoding of dynamic gaussians for streaming free-viewpoint videos," *Advances in Neural Information Processing Systems*, vol. 37, pp. 43 435–43 467, 2024.

[62] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.

[63] J. Li, J. Zhang, X. Bai, J. Zheng, X. Ning, J. Zhou, and L. Gu, "Dngaussian: Optimizing sparse-view 3d gaussian radiance fields with global-local depth normalization," in *Proceedings of the IEEE/CVF*

Conference on Computer Vision and Pattern Recognition, 2024, pp. 20 775–20 785.
[64] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," *arXiv preprint arXiv:1308.3432*, 2013.
[65] Y. He, X. Ren, D. Tang, Y. Zhang, X. Xue, and Y. Fu, "Density-preserving deep point cloud compression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 2333–2342.
[66] J. Rissanen and G. G. Langdon, "Arithmetic coding," *IBM Journal of research and development*, vol. 23, no. 2, pp. 149–162, 1979.
[67] I. H. Witten, R. M. Neal, and J. G. Cleary, "Arithmetic coding for data compression," *Communications of the ACM*, vol. 30, no. 6, pp. 520–540, 1987.
[68] A. Cao and J. Johnson, "Hexplane: A fast representation for dynamic scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 130–141.
[69] F. C. Commission, "Raw data – measuring broadband america," Federal Communications Commission, Tech. Rep., 2016. [Online]. Available: https://www.fcc.gov/reports-research/reports/
[70] H. Riiser, P. Vigmostad, C. Griwodz, and P. Halvorsen, "Commute path bandwidth traces from 3g networks: Analysis and applications," in *Proceedings of the 4th ACM Multimedia Systems Conference*, 2013, pp. 114–118.
[71] K. Park, U. Sinha, P. Hedman, J. T. Barron, S. Bouaziz, D. B. Goldman, R. Martin-Brualla, and S. M. Seitz, "Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields," *arXiv preprint arXiv:2106.13228*, 2021.
[72] J. Fang, T. Yi, X. Wang, L. Xie, X. Zhang, W. Liu, M. Nießner, and Q. Tian, "Fast dynamic radiance fields with time-aware neural voxels," in *SIGGRAPH Asia 2022 Conference Papers*, 2022, pp. 1–9.
[73] H. Lin, S. Peng, Z. Xu, T. Xie, X. He, H. Bao, and X. Zhou, "High-fidelity and real-time novel view synthesis for dynamic scenes," in *SIGGRAPH Asia 2023 Conference Papers*, 2023, pp. 1–9.
[74] B. Attal, J.-B. Huang, C. Richardt, M. Zollhoefer, J. Kopf, M. O'Toole, and C. Kim, "Hyperreel: High-fidelity 6-dof video with ray-conditioned sampling," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 16 610–16 620.
[75] F. Wang, S. Tan, X. Li, Z. Tian, Y. Song, and H. Liu, "Mixed neural voxels for fast multi-view video synthesis," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 19 706–19 716.
[76] F. Wang, Z. Chen, G. Wang, Y. Song, and H. Liu, "Masked space-time hash encoding for efficient dynamic scene reconstruction," *Advances in Neural Information Processing Systems*, vol. 36, 2024.
[77] J. Bae, S. Kim, Y. Yun, H. Lee, G. Bang, and Y. Uh, "Per-gaussian embedding-based deformation for deformable 3d gaussian splatting," in *European Conference on Computer Vision*. Springer, 2025, pp. 321–335.
[78] "Draco 3d data compression." https://google.github.io/draco, 2023.
[79] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 586–595.
[80] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

Zhicheng Liang received his B.Eng. degree in Internet of Things from Jinan University in 2019 and his M.Sc. degree in Data Science from the City University of Hong Kong in 2021. He is currently pursuing a Ph.D. in the School of Science and Engineering at The Chinese University of Hong Kong, Shenzhen, China. His research interests broadly include machine learning, multimedia networking, computer vision, and data mining.

Dayou Zhang received the B.S. degree from the School of Information and Electronics, Beijing Institute of Technology, Beijing, China, in 2017. He is currently pursuing the Ph.D. degree in the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, China. His research interests include multimedia networking, virtual reality, and machine learning.

Linfeng Shen is currently a Ph.D. student in the School of Computing Science at Simon Fraser University, BC, Canada. He received BEng in information security from Beijing University of Posts and Telecommunications in 2019, and MSc in computing science from Simon Fraser University in 2021. His research interests include edge computing and multimedia.

Miao Zhang received her B.Eng. degree from Sichuan University in 2015, and her M.Eng. degree from Tsinghua University in 2018. She is currently a Ph.D. student at Simon Fraser University, British Columbia, Canada. Her research areas include cloud and edge computing, and multimedia systems and applications.

Jian Zhang currently serves as the Vice President at the Migu Culture Technology Co., Ltd. He graduated from Nanjing University of Posts and Telecommunications in 2010. His primary research interests encompass 3D reconstruction, computer vision and Multimodal Large Language Models.

Bin Ju currently holds the position of Vice President at the Shenzhen Research Institute of Migu Culture Technology Co., Ltd. He graduated from Xidian University in 2008. His research area includes stereoscopic video, three-dimensional reconstruction, Large Language Model, and the metaverse technologies.

Mallesham Dasari is an Assistant Professor at Northeastern University, affiliated with both the Institute for the Wireless Internet of Things and the Department of Electrical and Computer Engineering. Dr. Dasari's research interests include Augmented and Virtual Reality (AR/VR) Systems, Computer Networks, Wireless Sensing and Communications, and Mobile and Wearable Computing. He has published extensively in premier conferences and journals, such as IEEE VR, IEEE ISMAR, ACM SIGCOMM, and USENIX NSDI. Dr. Dasari actively contributes to the academic community, he has served as a Program Committee Member for numerous prestigious conferences, including USENIX NSDI, IEEE VR, IEEE COMSNETS, IEEE ICDCS, USENIX ATC, ACM IMC, ACM CoNEXT, ACM MM, ACM MMSys, and ACM SIGCOMM. He has also contributed as a Program Co-Chair for the ACM SIGCOMM Workshop on Emerging Multimedia Systems and the ACM Student Workshop at MobiSys.

Fangxin Wang (Member, IEEE) is an assistant professor at The Chinese University of Hong Kong, Shenzhen (CUHKSZ). He received his Ph.D., M.Eng., and B.Eng. degree all in Computer Science and Technology from Simon Fraser University, Tsinghua University, and Beijing University of Posts and Telecommunications, respectively. Before joining CUHKSZ, he was a postdoctoral fellow at the University of British Columbia. Dr. Wang's research interests include Multimedia Systems and Applications, Cloud and Edge Computing, Deep Learning, and Distributed Networking and System. He leads the intelligent networking and multimedia lab (INML) at CUHKSZ. He has published more than 50 papers at top journal and conference papers, including INFOCOM, Multimedia, VR, ToN, TMC, IOTJ, etc. He entered the Stanford World's Top 2% Scientists List in 2023. He was selected in the 8th Young Elite Scientist Sponsorship Program, CUHKSZ Presidential Young Scholar, and a recipient of SFU Dean's Convocation Medal for Academic Excellence. He is an associate editor of IEEE Transactions on Mobile Computing. He served as the TPC chairs of IEEE Satellite 2023, TPC members of IWQoS, ICC, BigCom and reviewer of many top conference and journals, including INFOCOM, ToN, TMC, JSAC, etc.

Jiangchuan Liu (S'01-M'03-SM'08-F'17) is a University Professor in the School of Computing Science, Simon Fraser University, British Columbia, Canada. He is a Fellow of The Canadian Academy of Engineering, an IEEE Fellow, and an NSERC E.W.R. Steacie Memorial Fellow. He was an EMCEndowed Visiting Chair Professor of Tsinghua University (2013-2016). In the past, he worked as an Assistant Professor at The Chinese University of Hong Kong and as a research fellow at Microsoft Research Asia. He received the BEng degree (cum laude) from Tsinghua University, Beijing, China, in 1999, and the PhD degree from The Hong Kong University of Science and Technology in 2003, both in computer science. He is a co-recipient of the inaugural Test of Time Paper Award of IEEE INFOCOM (2015), ACM SIGMM TOMCCAP Nicolas D. Georganas Best Paper Award (2013), and ACM Multimedia Best Paper Award (2012). His research interests include multimedia systems and networks, cloud and edge computing, social networking, online gaming, and Internet of things/RFID/backscatter. He has served on the editorial boards of IEEE/ACM Transactions on Networking, IEEE Transactions on Big Data, IEEE Transactions on Multimedia, IEEE Communications Surveys and Tutorials, and IEEE Internet of Things Journal. He is a Steering Committee member of IEEE Transactions on Mobile Computing and Steering Committee Chair of IEEE/ACM IWQoS (2015-2017). He is TPC Co-Chair of IEEE INFOCOM'2021 and General Co-Chair of INFOCOM'2024.interests include edge computing and multimedia.