

HoloMobile: Photorealistic Avatar on Mobile via Compact Dynamic Gaussians from a Monocular Video

Yuting He¹, Yi-Hua Huang², Xiaojuan Qi², Zhenyu Yan¹, Guoliang Xing^{1,*}

¹ The Chinese University of Hong Kong, Hong Kong SAR, China

² The University of Hong Kong, Hong Kong SAR, China

Abstract

Human avatars, defined as animatable 3D digital people, have diverse applications, including teleconferencing, motion guidance, remote care, and gaming. With the widespread adoption of smart mobile devices, the demand for real-time viewing of dynamic digital humans on mobile platforms has grown significantly. However, existing avatar reconstruction methods, such as MagicStream and ExAvatar, struggle to produce high-fidelity avatars from monocular videos captured by mobile devices. Furthermore, most related research focuses on individual algorithmic components and does not provide an end-to-end pipeline that jointly addresses avatar reconstruction, 4D data transmission, and mobile rendering. In this work, we introduce HoloMobile, the first end-to-end mobile avatar system with two key capabilities: (1) High-fidelity avatar reconstruction from monocular videos captured on mobile phones, using a hybrid explicit-implicit model. (2) Efficient polynomial compression, transmission, and web-based rendering of dynamic 3D avatars on mobile devices. With avatar reconstruction performed on a PC or cloud with a single consumer-grade GPU, HoloMobile provides an end-to-end solution for mobile users: from data capture to streaming and real-time viewing. Evaluation on 20 self-captured avatars reveals that HoloMobile achieves an average PSNR of 30.36 dB, a compression rate of 99.57%, and over 200 FPS for mobile rendering, demonstrating superiority over existing baselines. The demo video of HoloMobile is available at <https://youtu.be/kFOFV7AiP5A>.

CCS Concepts

• Human-centered computing → Ubiquitous and mobile computing systems and tools; • Information systems → Multimedia streaming.

Keywords

Photorealistic Avatar, 3D Gaussian Splatting, Mobile System.

ACM Reference Format:

Yuting He¹, Yi-Hua Huang², Xiaojuan Qi², Zhenyu Yan¹, Guoliang Xing^{1,*}. 2026. HoloMobile: Photorealistic Avatar on Mobile via Compact Dynamic Gaussians from a Monocular Video. In *The 24th Annual International Conference on Mobile Systems, Applications and Services (MobiSys '26)*, June 21–25, 2026, Cambridge, United Kingdom. ACM, New York, NY, USA, 14 pages. <https://doi.org/10.1145/3745756.3809219>

*Guoliang Xing is the corresponding author.



This work is licensed under a Creative Commons Attribution 4.0 International License. *MobiSys '26, Cambridge, United Kingdom*

© 2026 Copyright held by the owner/author(s).

ACM ISBN 979-8-4007-2027-7/26/06

<https://doi.org/10.1145/3745756.3809219>

1 Introduction

Mobile devices, including smartphones, tablets, and head-mounted displays, are becoming increasingly capable platforms for immersive visual communication. Recent advances in neural rendering [35] and 3D Gaussian Splatting (3DGS) [21] have made high-quality 3D content creation more practical. However, realistic human representation remains a key missing piece for mobile applications such as remote care, teleconferencing, training, and gaming. In these scenarios, 2D video and stylized avatars are often insufficient: 2D video fixes the viewpoint and limits interaction, while stylized avatars fail to faithfully preserve identity, appearance, and subtle nonverbal cues. Photorealistic 3D human avatars are therefore important for enabling viewpoint-free, immersive, and identity-preserving experiences on mobile devices. Despite this need, existing human avatar systems [3, 5, 39] largely assume controlled capture environments and desktop-class GPUs. They often rely on multi-view camera arrays, expensive reconstruction pipelines, or heavyweight rendering methods, making them impractical for everyday mobile use. We aim to design a mobile-first avatar system that allows users to create and experience high-fidelity 4D digital humans from monocular videos captured on commodity phones.

In this paper, we propose *HoloMobile*¹, the first mobile-oriented avatar system that builds photorealistic avatars from monocular captures and enables real-time rendering on diverse mobile devices, as shown in Fig. 1. *HoloMobile* comprises three core components: (1) monocular avatar reconstruction, (2) 4D human sequence compression, and (3) on-mobile rendering of dynamic human avatars. Users capture a monocular video using mobile devices and then upload it to a PC or cloud server to reconstruct a high-fidelity human avatar in a 3DGS representation. Once the avatar is reconstructed, it becomes a digital twin of the human, serving as a reusable asset with unlimited longevity. The avatar can be animated using various motion sources, such as real-time visual capture [7, 24], AI-generated motions [16, 54], retrieval from predefined motion libraries [34, 44], or user customization. The resulting 4D sequence is then compressed for efficient transmission, and our mobile rendering module enables interactive viewing across diverse applications, for example, free-viewpoint dance learning on a tablet at home or immersive, face-to-face 4D conversations with family members using VR headsets.

Designing such a system raises the following challenges.

• *Challenge 1: Accessible high-fidelity avatar reconstruction.* Human avatar reconstruction has been a long-standing challenge [18, 43, 50, 58]. Recently, 3DGS has become the dominant representation for human avatars due to its ability to deliver high-quality, real-time rendering. However, most existing 3DGS-based avatar reconstruction

¹ *HoloMobile* denotes **holographic mobile** avatar system, designed to deliver high-fidelity, animatable 3D human representations on mobile devices.

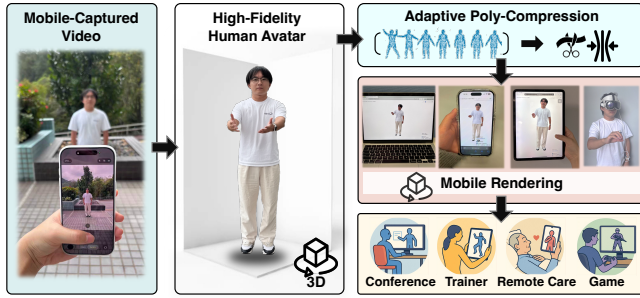


Figure 1: *HoloMobile* is a mobile-oriented system that enables users to customize and view dynamic avatars, supporting various applications such as teleconferencing, training, remote care, and gaming. The avatar can be created from mobile-captured monocular video, compressed for efficient transmission, and rendered interactively on various mobile devices.

methods [1–3, 38, 40] require large-scale camera-array facilities, typically hundreds of synchronized cameras in known and static poses. This requirement limits accessibility for ordinary users, who aim to reconstruct customized avatars using monocular videos captured on mobile phones. Monocular avatar methods [17, 23, 37] remove the need for camera arrays, but still fall short of high fidelity under challenging conditions with diverse clothing and rapid motion. A major limitation is their reliance on parametric human body models such as SMPL [33], which parameterize motion using 24 joints. This low-degree-of-freedom parameterization cannot fully capture complex real-world dynamics, including muscle bulging, cloth wrinkles, and nuanced facial expressions.

• **Challenge 2: Efficient transmission and mobile rendering of human avatars.** Avatars generate dynamic human sequences driven by motion sources, such as motion capture, AI-generated motions, or user control, selected based on the specific application. These sequences must be efficiently streamed to client devices. In a 3DGS-based setting, the content becomes a time-varying 3D Gaussian points sequence (4D data). A naive baseline that sends per-frame 3D Gaussian parameters leads to prohibitive bandwidth usage (e.g., tens of gigabytes for a short sequence), making real-time streaming infeasible. V³ [53] encodes Gaussian attributes into an MP4 video format, significantly reducing data size but still resulting in bandwidth-intensive streams for longer sequences. MagicStream [3], which leverages an SMPL-based representation and neural rendering, transmits only parametric SMPL poses and performs avatar inference and neural rendering locally, drastically reducing network load. However, this approach requires powerful locally deployed GPUs to run the neural network for rendering. Beyond transmission challenges, mobile rendering of long, dynamic 3DGS sequences also presents significant difficulties. Existing 3DGS pipelines are primarily optimized for desktop GPUs with CUDA-based rasterizers. Current web viewers [20, 25] support only static 3DGS in browsers, while per-frame 3DGS rendering incurs high data I/O overhead and leads to temporal flickering.

To address these challenges, we propose several novel approaches in *HoloMobile*. To overcome the limited expressiveness of parametric body models, we propose a **hybrid explicit-implicit reconstruction** framework. We incorporate an implicit pose model to enhance motion modeling and handle challenging variations. The implicit

pose parameters are constrained within a normal distribution, serving as a prior for these variations. This probabilistic modeling ensures that the common human appearances are captured, and the implicit pose parameters can be set to zero during inference. Furthermore, we optimize the base 3DGS shape progressively, sculpting it from coarse to fine by progressively unmasking features defined on multi-resolution hash grids. This gradual process enhances the quality and precision of reconstruction.

Based on the observation that human motion is continuous but varies in temporal intensity, we propose an **adaptive splitting and polynomial compression** method. The dynamic 3DGS sequence is adaptively split into segments using graph modeling and a minimal-cost path algorithm. Within each segment, the 3DGS trajectories are compressed using cubic polynomials. This adaptive segmentation identifies an optimal splitting strategy, grouping smoother motions within a single segment while isolating segments with more intense or rapid motion. Compared to per-frame 3DGS transmission or even splitting, our method achieves a significantly 99.57% compression ratio with minimal error, enabling faster and more efficient transmission of dynamic human avatars to mobile devices.

To enable cross-platform rendering, we develop a **web-based dynamic 3DGS rendering** module specifically designed for mobile devices. Over time, the cubic polynomial data are sequentially loaded and applied based on the segment’s order and time length. The positions of the human 3DGS are computed in real-time using the polynomial values corresponding to the current timestamp. The rendering tools provide users with interactive controls, allowing them to freely adjust the viewpoint, playback speed, and timeline of the dynamic human scene. This enables users to explore different views, review specific parts of the motion, slow down the display for learning, or jump and replay segments.

In summary, our main contributions are as follows:

- We propose *HoloMobile*, an end-to-end mobile-oriented system that reconstructs high-quality human avatars from monocular smartphone videos, efficiently compresses 4D human data and renders them on mobile devices, enabling a range of downstream applications.
- A hybrid explicit-implicit motion model to achieve high-fidelity reconstruction with monocular input.
- An adaptive splitting and polynomial compression scheme for dynamic 3DGS sequences, together with a web-based 3DGS rendering module that supports cross-platform, real-time rendering on mobile devices.
- We conduct extensive experiments on both public and self-captured datasets, demonstrating that *HoloMobile* achieves an average PSNR of 30.36 dB, a compression rate of 99.57%, and over 200 FPS for mobile rendering.

2 Related Work

Monocular Human Avatar Reconstruction. Reconstructing high-fidelity human avatars traditionally relied on calibrated multi-view RGB or RGB-D camera arrays [1–3, 38, 40, 46, 66], which provide strong geometric supervision but are impractical for general-purpose applications. For monocular video inputs, recent methods can be roughly grouped into mesh-based, NeRF-based, and 3DGS-based approaches. Mesh-based methods improve efficiency and drivability,

but often struggle to preserve clothing wrinkles and subtle facial appearance because they are tied to explicit surface topology [56]. NeRF-based methods extend articulated neural radiance fields to dynamic humans [10, 17, 18, 43, 50], producing detailed implicit representations from single videos, but they still face challenges in rendering efficiency and practical real-time deployment. More recent monocular 3DGS-based methods improve rendering speed and visual quality through animatable Gaussians, deformable Gaussian avatars, and pose-dependent appearance modeling [11, 13, 14, 23, 37, 45, 60]. Despite this progress, monocular methods in all three categories remain constrained by low-degree-of-freedom body priors and often struggle with pose-dependent deformation, fine appearance details, and out-of-distribution motion. A recent concurrent work, HRM² Avatar [47], is closest to our setting: it reconstructs high-fidelity avatars from monocular phone scans and supports real-time mobile rendering. Compared with *HoloMobile*, however, it assumes a two-stage phone scanning protocol, uses a clothed mesh-driven illumination-aware Gaussian representation, and targets fully GPU-driven on-device rendering. In contrast, *HoloMobile* focuses on end-to-end mobile delivery: after reconstructing a reusable avatar from a short monocular video, we generate motion-driven 3DGS sequences, compress them for transmission, and render them directly on heterogeneous mobile clients through a lightweight web pipeline.

Mobile Volumetric Video System. Another line of work studies how to capture, transmit, and render immersive 3D content on mobile devices under bandwidth and latency constraints. Early systems improve mobile free-viewpoint video delivery through visibility-aware fetching, multiview transcoding, neural enhancement, and multi-user bandwidth sharing [12, 31, 32, 62]. Closely related telepresence systems reduce communication cost through octree-based volumetric streaming, topology-aware alignment, semantic communication, and 2D-inferred delta encoding [3, 9, 26–28, 59]. These works are highly relevant from a mobile-systems perspective, but they primarily focus on end-to-end communication and delivery of captured volumetric content or communication-oriented proxies, rather than a reusable avatar reconstructed once from a monocular mobile video and later re-driven by arbitrary motions.

3DGS Delivery and Mobile Rendering. Recent works have further studied how to make 3DGS/4DGS content easier to deliver under bandwidth and device constraints. For static scenes, LapisGS [48], L3GS [52], and SGSS [67] support adaptive 3DGS delivery for 6-DoF navigation by transmitting scene-oriented Gaussian subsets or layers based on viewpoint and bandwidth. For dynamic scenes, LTS [51] extends DASH-style streaming to dynamic 3DGS, V³ [53] converts dynamic 3DGS into 2D Gaussian videos for hardware codecs, and Vega [22] targets mobile volumetric video streaming with selective computation. These systems primarily stream captured dynamic scenes, whereas our unit of transmission is a motion-driven avatar sequence whose per-Gaussian trajectories remain semantically tied to the same reusable avatar. Avatar-specific mobile execution is more limited. SqueezeMe [15] distills Gaussian full-body avatars for on-device animation and rendering, while EdgeGaussian [19] studies edge-client collaborative rendering for mobile-VR free-viewpoint video rather than reusable avatars. Compared with these methods, *HoloMobile* makes a different systems tradeoff: instead of relying on distilled on-device inference or edge-assisted neural rendering, it

pushes neural computation to the server and streams a directly renderable representation that can be decoded linearly in the browser. Related 4DGS methods, including Spacetime GS [30], 4DGS [57], 1000+ FPS 4DGS [61], and MEGA [65], focus on general dynamic-scene representation, efficiency, or memory reduction. Technically, they absorb temporal redundancy into learned spacetime features, deformation fields, lifespan modeling, or compact predictors. In contrast, *HoloMobile* does not learn a generic 4D scene representation for rendering-time deformation. It starts from a monocularly reconstructed reusable avatar, generates motion-driven 3DGS sequences offline or online, and compresses them with adaptive segmentation and polynomial trajectory fitting for heterogeneous mobile clients.

3 Background and Motivation

3.1 Background

3D Gaussian Splatting. 3DGS [21] has emerged as the mainstream 3D representation method, surpassing earlier explicit representations such as meshes, point clouds, voxels, and implicit neural radiance fields. This is due to its fast rendering speed, high-quality outputs, and efficient fitting to given observation views. 3DGS represents 3D scenes using Gaussian kernels with varying colors and decaying opacity distributed in 3D space. Millions of Gaussian particles can flexibly adapt to any target object, enabling the rendering of high-fidelity images from novel viewpoints. Due to its exceptional visual quality and fast rendering speed, it has become one of the most popular techniques for representing 3D objects.

Let us define a set of N Gaussians as $\{\mathcal{G}_n, n \in [N]\}$. Each Gaussian primitive is represented as $\mathcal{G}_n = (\mu_n, s_n, q_n, c_n)$, where $\mu_n \in \mathbb{R}^3$ is the center position, $s_n \in \mathbb{R}^{+3}$ are the scaling parameters along the three axes, $q_n \in \mathbb{R}^4$ is a unit quaternion representing the rotation matrix R , and $c_n \in \mathbb{R}^3$ is the color of the Gaussian. For simplicity, we use the Gaussian’s color attribute instead of view-dependent Spherical Harmonics to model its appearance. The 3D Gaussian distribution can be expressed as:

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

where $\Sigma = R S S^T R^T$ is the covariance matrix.

Rendering 3D Gaussians. To render a set of Gaussians into a 2D image from a target viewpoint, we project the 3D Gaussians onto a 2D image plane. Following Zwicker *et al.* [68], we use the EWA projection to map the 3D Gaussian distribution to a 2D Gaussian. This projection leverages the extrinsic view transformation matrix W and the intrinsic projection matrix J of the center point:

$$\Sigma_{2d} = J W \Sigma W^T J^T. \quad (2)$$

The color of each 2D Gaussian is then blended from front to back, with opacity determined by the Gaussian density at each pixel. The rendered image, I_{gs} , is compared with the ground truth image, I_{gt} , to compute the loss. This loss is used to optimize all Gaussian attributes, enabling high-fidelity reconstruction of the captured static or dynamic scenes.

Parametric Body Model. SMPL [33] is the most widely used full human body model for human avatar parameterization. Its extension, SMPL-X [42] is a unified parametric human body model with shape parameters trained jointly for the face, hands, and body. It uses standard vertex-based linear blend skinning (LBS) [29] with

learned corrective blend shapes, and comprises 10475 vertices and 54 joints, including joints for the neck, jaw, eyeballs, and fingers. The model is parameterized by pose $\theta \in R^{3 \times (54+1)}$, shape $\beta \in R^{100}$, and facial expression $\psi \in R^{50}$, and its geometry is defined as:

$$M(\beta, \theta, \psi) = \mathcal{W}(T_p(\beta, \theta, \psi), \mathcal{J}(\beta), \theta, \mathbf{W}) \quad (3)$$

where $\mathcal{W}(\cdot)$ is the standard LBS function, T_p represents the posed template, $\mathcal{J}$ is the joint regressor, and $\mathbf{W}$ denotes the blend weights.

3.2 A Motivation Study

HoloMobile targets an end-to-end mobile pipeline that reconstructs a high-fidelity human avatar from a monocular phone video and delivers animatable 3D humans efficiently to mobile clients. Existing methods still fall short in two coupled aspects: they cannot reliably recover high-fidelity avatars from casual monocular captures, and they lack an efficient representation for streaming long dynamic human sequences to resource-constrained devices.

Monocular Human Avatar Reconstruction. High-fidelity human avatar reconstruction is essential for applications that require realistic human representations. While multi-view videos provide sufficient supervision for full-body reconstruction and are commonly used in previous avatar reconstruction methods [3, 5, 27, 53], they often rely on depth sensors for more direct geometric supervision. However, such systems require carefully calibrated, synchronized camera arrays arranged on frames, which are expensive, complex to maintain, and unsuitable for in-the-wild or casual usage scenarios.

Monocular videos captured with mobile devices eliminate this hardware barrier. Nevertheless, existing methods based on monocular video still struggle to produce satisfactory results on challenging real-world sequences with large motions, diverse clothing, and expressive faces. To address this limitation, we propose a powerful avatar model that makes the system more practical and accessible through mobile data capture. We identified that the primary limitation of these methods lies in their heavy reliance on parametric models such as SMPL/SMPL-X to represent human shape and motion. While these models simplify the reconstruction process, they impose rigid constraints on the degrees of freedom, restricting the ability to capture diverse body shapes, subtle facial expressions, and fine-grained clothing textures. Consequently, these methods frequently fail in human videos with inconsistent and complex appearances, resulting in blurry reconstructions, loss of detail in clothing, and unrealistic facial reconstruction. To better understand these limitations, we qualitatively evaluate three representative monocular methods, InstantAvatar [17], HUGS [23], and ExAvatar [37], alongside a multi-view telepresence system, MagicStream [3], using only single-view data (see Sec.6 for details). As shown in Fig. 2, monocular methods struggled to preserve high-frequency details, especially in critical regions such as faces and clothing. This degradation stems from their inability to effectively model fine-grained textures and dynamic variations. These findings underscore the need for a novel approach that achieves high-fidelity reconstruction from monocular videos without solely relying on the parametric priors.

Compressed Transmission and Delay. Once the avatar is reconstructed, each new motion induces a dynamic 3DGS sequence that must be delivered from the server to the mobile client. The system's challenge is to preserve avatar fidelity while keeping transmission



Figure 2: Qualitative comparison of HUGS [23], InstantAvatar [17], ExAvatar [37] and MagicStream [3].

size and processing latency low enough for interactive playback. Existing solutions occupy very different points in this trade-off space. One natural baseline is to transmit a canonical avatar once and then stream only SMPL poses or small residual updates. However, existing 3DGS avatar pipelines [23, 37] rely on neural inference to animate 3DGS from SMPL parameters. As a result, this design shifts the burden to the client: reconstructing high-fidelity Gaussians from pose parameters still requires neural inference, while residual traffic grows quickly for large motions and pose-dependent clothing deformation. The baselines below illustrate these trade-offs. Another baseline is per-frame 3DGS, which transmits raw Gaussian attributes without compression, resulting in a prohibitively high transmission cost of 23.16 GB for a 600-frame sequence of yoga motion. V³ [53] encodes Gaussian attributes into a 2D Gaussian video, lowering transmission to 863.95 MB, though still bandwidth-intensive. V-PCC [8] decomposes point clouds onto multiple planes and compresses them using standard 2D video codecs (e.g., HEVC). It achieves the highest compression efficiency at just 480 KB, but its 70+ hour encoding latency makes it impractical for real-time use. In a setup utilizing locally deployed GPUs instead of mobile rendering, MagicStream [3] infers the avatar and neural rendering models locally, significantly reducing the data transmission size to 2.79 MB with SMPL parameters. However, due to its reliance on a large U-Net model for neural rendering and the absence of an efficient 3DGS representation, it still experiences a model inference delay of approximately 67ms on an NVIDIA RTX 3090 GPU.

Summary. Overall, existing methods fail to meet the specific requirements of a mobile-oriented system: achieving high-quality avatar reconstruction from mobile-captured monocular videos and delivering real-time avatar experiences through mobile rendering. They either rely on specialized multi-view capture setups, incur prohibitive bandwidth or encoding costs when streaming dynamic 3D content, or depend on heavyweight GPU rendering pipelines that exceed the capabilities of commodity mobile devices. *HoloMobile* addresses these gaps with (1) high-fidelity monocular avatar reconstruction beyond parametric-model limitations through a hybrid explicit-implicit design and per-Gaussian offset parameterization, (2) compact and streamable dynamic 3DGS representations tailored to avatar motion, and (3) real-time, web-based rendering on everyday mobile devices. In the following section, we introduce each component of *HoloMobile* and explain how they contribute to building a mobile-friendly system, from data capture to final rendering, to achieve the objectives listed above.

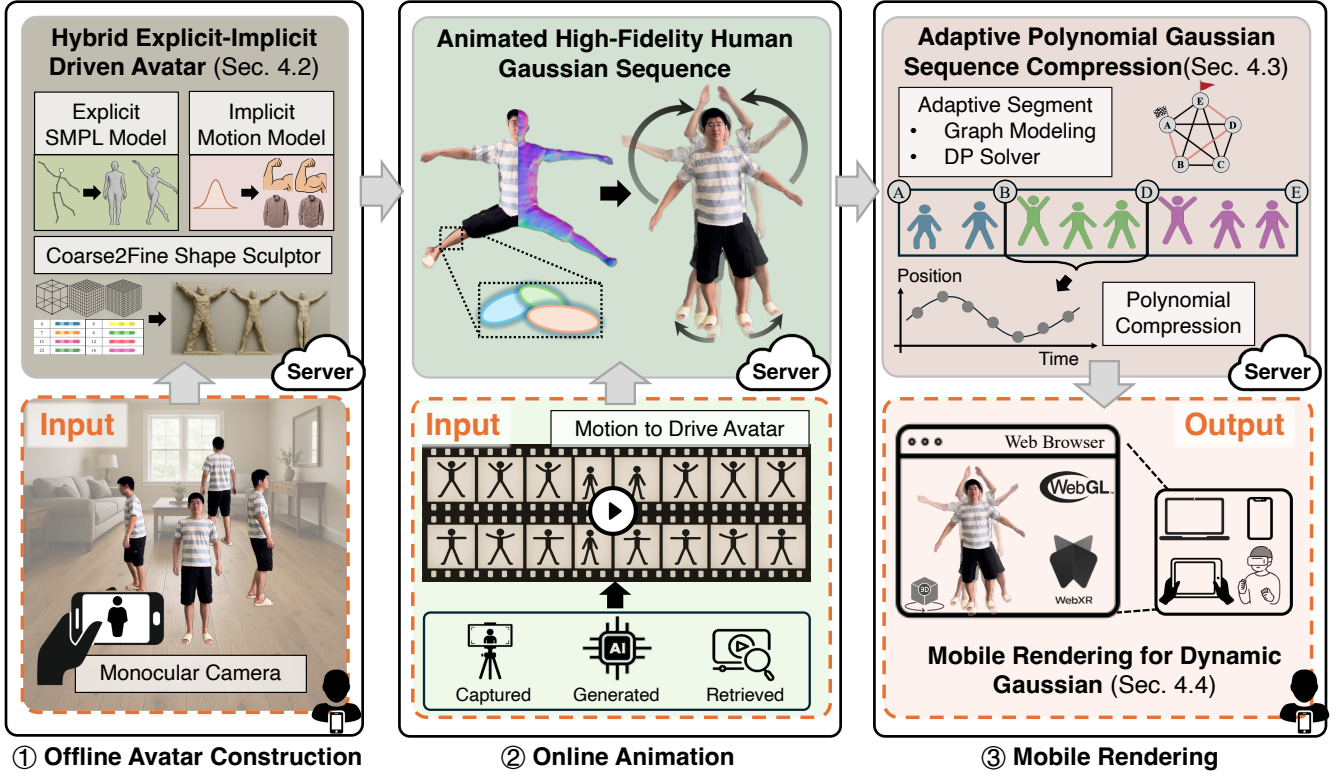


Figure 3: The system overview of *HoloMobile*. ① Offline avatar construction: from a monocular mobile video, we reconstruct a 3D human avatar using a hybrid explicit–implicit model that combines SMPL-X drivability with 3DGS photorealism. ② Online animation: the avatar is driven by captured, AI-generated, or retrieved motions to form a high-fidelity dynamic Gaussian sequence. ③ Mobile rendering: the sequence is adaptively segmented and polynomially compressed on the server, then transmitted and rendered in real time on mobile devices via web-side sorting and rasterization.

4 Design of HoloMobile

4.1 Overview

As illustrated in Fig. 3, given a mobile-captured monocular video of a human, *HoloMobile* learns a Gaussian representation to render the video from arbitrary viewpoints on various mobile devices. Additionally, it enables animating the avatar to perform a wide range of movements. The workflow consists of two temporally distinct stages. The first is a one-time offline avatar construction stage: a short monocular capture is uploaded to a server, which reconstructs a reusable canonical avatar. The second is an online animation and streaming stage: the reconstructed avatar is driven by live, retrieved, or AI-generated motions and then streamed to the client without repeating reconstruction. To construct a high-fidelity human avatar, we propose a novel hybrid explicit–implicit modeling approach. This approach integrates the parametric SMPL-X model for explicit pose control with an implicit motion model to capture subtle deformations and motions beyond the scope of SMPL-X. For real-time rendering with interactive view adjustments on mobile devices, we introduce an adaptive compression algorithm and a web-based rendering method tailored to the compressed representation. Dynamic 3DGS sequences are segmented using graph modeling and a dynamic programming solver, followed by polynomial fitting to achieve a

compact representation. These compressed sequences are then transmitted to mobile devices, where they are rendered and viewed by users. *HoloMobile* supports two deployment modes. In the Video-on-Demand (VoD) mode, a motion sequence is pre-animated on the server, compressed into a compact polynomial 3DGS representation, and delivered to the mobile client on demand. In the On-the-Fly real-time streaming mode, the server receives live motion capture or AI-generated pose parameters, animates the avatar in real time, compresses each segment incrementally, and streams it to the client with minimal latency. This architecture enables both asynchronous avatar delivery and synchronous bidirectional interaction, covering the full range of application requirements from pre-recorded fitness guidance to live teleconferencing.

Putting it all together, our end-to-end system captures a person with a single mobile camera, reconstructs an animatable high-fidelity avatar on a server, compresses its dynamic 3DGS sequence for efficient streaming, and finally delivers real-time, interactive rendering and control on various mobile devices.

4.2 Hybrid Explicit–Implicit Driven Avatar

In line with recent works [13, 18, 37], we leverage the explicit SMPL model to control human motion using interpretable skeletons and

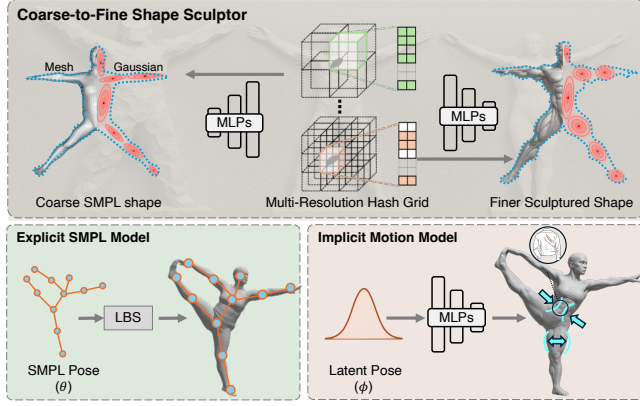


Figure 4: Overview of our coarse-to-fine shape sculptor and hybrid explicit-implicit motion model.

skinning weights. Gaussians are bound to the shape surface and move according to the SMPL model’s controls. The canonical shape of SMPL represents a human in a rest pose, with shape variations controlled by the β parameter. However, the diversity of real-world human shapes often exceeds the representational capacity of β . Similarly, the motion parameters θ in SMPL lack the flexibility required for precise motion reconstruction. Real-world variations, such as garment wrinkles and dynamic muscle deformations, introduce inconsistencies in geometry and appearance, often resulting in blurry reconstructions. To address these limitations in shape and pose representation, we propose: (1) a coarse-to-fine shape sculptor model to handle diverse human shapes and (2) an implicit motion model with a learnable latent pose assigned to each video frame, as illustrated in Fig. 4. These innovations enable our method to achieve significantly higher-quality Gaussian reconstructions of humans compared to existing approaches.

4.2.1 Explicit SMPL Model. We use the SMPL [33] model to represent body pose, while hand articulations are modeled using the corresponding parameters in its extended version, SMPL-X [42]. The SMPL-X model includes a shape parameter vector $\beta \in \mathbb{R}^{10}$, which models the shape of a canonical human body, and a pose parameter vector $\theta \in \mathbb{R}^{54 \times 3}$, which represents the 3D rotations of 54 joints in the human skeleton. The pose parameters control the deformation of a SMPL-X mesh with $N = 10475$ vertices using linear blend skinning. To achieve a higher vertex resolution, we apply subdivision [49] twice, increasing the vertex count to $N = 167390$. Each vertex is then equipped with a Gaussian primitive, which deforms alongside the vertices to enable various poses.

4.2.2 Coarse-to-fine Shape Sculptor. The SMPL surface usually does not perfectly align with real human bodies, even in the same zero-pose, due to the diversity in clothing and body shapes. To address this challenge, we leverage features stored in a hash table, indexed by multi-resolution grids, to model these variations in body shape. In our approach, Gaussians are allowed to move with offsets in the zero-pose canonical space, enabling them to flexibly fit the target human shape in a coarse-to-fine manner, thereby avoiding overfitting. For a 3DGS, its base position, defined as the corresponding vertex on the zero-pose mesh, is used to retrieve features from the hash

table. These features are indexed by the 8 grids of the voxel to which the vertex belongs. The features from these grids are tri-linearly interpolated to obtain a final feature representation. Multiple grids at different resolutions are used, and their features are concatenated from coarse to fine. During training, features from the coarsest grids are unmasked first, while features from higher-resolution grids are progressively unmasked. The features retrieved for each Gaussian are fed into a neural network to predict offsets relative to the base body shape. In the early stages of training, when only low-frequency features are active, all Gaussians share spatially similar features, allowing the model to learn spatially continuous and smooth offsets. During this phase, the model captures large-scale residuals of the target shape. As training progresses and higher-frequency features are unmasked, the Gaussians gain greater flexibility, enabling them to adapt to finer details of the target human body. This hierarchical design allows the model to fit the target shape with high precision, effectively overcoming the limitations of the SMPL β parameter.

4.2.3 Implicit Motion Model. While the coarse-to-fine shape sculptor significantly improves shape fitting, the motion of the human avatar remains constrained by the linear blend skinning (LBS) of the SMPL model. This limitation stems from the SMPL motion parameters θ , which govern limb transformations based on joint rotations. As a result, the SMPL model struggles to capture finer and more flexible real-world motions, such as subtle garment deformations and nuanced expressions. Without addressing this limitation, the reconstructed human avatar may appear blurry due to inconsistencies in appearance caused by motion ambiguity. To overcome this challenge, we introduce an additional implicit motion model with a learnable pose parameter $\phi \in \mathbb{R}^{C_{im}}$ for each video frame to control human motion. For each Gaussian, its deformation under the influence of the implicit motion ϕ is decoded by a multi-layer perceptron (MLP): $\Delta\mu_{im} = \text{Dec}(\phi, \mu)$, where μ represents the Gaussian center in the canonical zero-pose space. To enhance the model’s ability to represent motions across a wide range of frequencies, we apply positional encoding to ϕ before feeding it into the decoder. The human avatar is now driven by both the explicit SMPL body model and the implicit motion model through deformation summation: $\Delta\mu = \Delta\mu_{ex} + \Delta\mu_{im}$. Here, the implicit deformations effectively compensate for the residual discrepancies between real-world motion and SMPL motion, resulting in a clearer and more accurate avatar reconstruction. Since the implicit motion model lacks explicit interpretability, we constrain the learnable latent pose parameters ϕ , which are associated with each video frame (similar to SMPL’s θ), to follow a normal distribution. This is achieved through a log-likelihood regularization loss: $\mathcal{L}_{im} = \lambda_{im} \|\phi\|_2^2$. This constraint ensures that, during inference, the implicit pose parameters can default to zero, thereby representing the most common characteristics across all views. From a Bayesian perspective, this formulation follows a standard prior-driven design: the prior is centered at zero to model the most common, shared appearance and motion components, while non-zero deviations are treated as latent variables that capture view-specific or outlier effects only when required by the data. This ensures that the implicit pose module activates selectively, representing residual variations that the explicit SMPL model cannot explain, rather than absorbing global motion patterns that are better handled by the interpretable

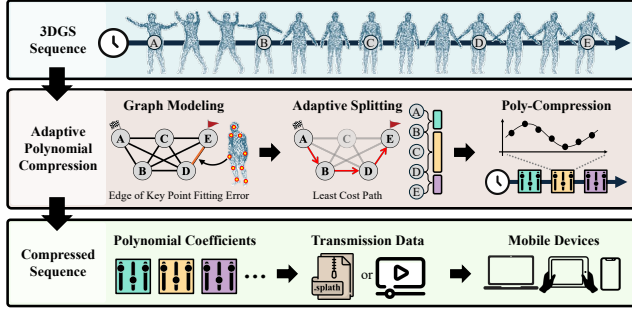


Figure 5: Illustration of the adaptive polynomial Gaussian sequence compression.

explicit scaffold. This design allows for more robust and consistent human avatar reconstruction.

4.3 Adaptive Polynomial Gaussian Sequence Compression

After constructing a high-fidelity avatar, we generate a 3DGS sequence representing a dynamic human with exceptional visual quality. However, transmitting this sequence directly to mobile devices will require substantial bandwidth and storage, making it impractical for resource-limited mobile environments. A seemingly simpler alternative is to transmit the canonical avatar once and then stream only SMPL poses or small residual updates. In practice, this is unsuitable for mobile clients: reconstructing a high-fidelity Gaussian point cloud from pose parameters still requires neural inference, and residual traffic grows quickly when motions are large or clothing deformation is complex. We therefore compress and stream a directly renderable 3DGS sequence, avoiding heavy on-device inference while preserving fidelity. To overcome this challenge, we develop a compression algorithm that approximates the Gaussian sequence efficiently using polynomials based on adaptive segment splits, as illustrated in Fig. 5. The algorithm first determines the optimal segmentation, ensuring high reconstruction quality while significantly reducing the sequence’s parameter count. Then each split is compressed into compact cubic polynomial coefficients.

4.3.1 Polynomial Fitting. Given a sequence of Gaussians $\{\mathcal{G}_n^t, n \in [N], t \in [T]\}$, obtained either from reconstruction or animated avatars, our goal is to compress this sequence using polynomials. Since the only time-dependent attribute of Gaussians is their positions $\{p_n^t, n \in [N], t \in [T]\}$, we aim to compress the sequence of positions by leveraging the continuity prior of motion tracks. For the n -th point $\{p_n^t, t \in [T]\}$, we fit a cubic polynomial $\hat{p}_n^t = \gamma_{n,3}t^3 + \gamma_{n,2}t^2 + \gamma_{n,1}t + \gamma_{n,0}$. Denoting the time matrix $\vec{T}$, coefficients $\vec{\gamma}_n$, and point tracks $\vec{p}_n$ as:

$$\vec{T} = \begin{bmatrix} 1 & t_1 & t_1^2 & t_1^3 \\ 1 & t_2 & t_2^2 & t_2^3 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & t_T & t_T^2 & t_T^3 \end{bmatrix}, \quad \vec{\gamma}_n = \begin{bmatrix} \gamma_{n,0} \\ \gamma_{n,1} \\ \gamma_{n,2} \\ \gamma_{n,3} \end{bmatrix}, \quad \vec{p}_n = \begin{bmatrix} p_n^{t_1} \\ p_n^{t_2} \\ \vdots \\ p_n^{t_T} \end{bmatrix}. \quad (4)$$

The optimization objective is defined as:

$$\vec{\gamma}_n = \underset{\vec{\gamma}}{\operatorname{argmin}} \|\vec{T} \cdot \vec{\gamma} - \vec{p}_n\|^2 = \left(\vec{T}^T \cdot \vec{T} \right)^{-1} \vec{T}^T \vec{p}_n. \quad (5)$$

Thus, we compress the trajectory of Gaussian points $\vec{p}_n \in \mathbb{R}^{T \times C}$ into cubic polynomial coefficients $\vec{\gamma}_n \in \mathbb{R}^{4 \times C}$.

When the motion sequence is long, a single cubic polynomial cannot accurately capture the motion. To address this, we divide the sequence into smaller segments and fit each segment with its own cubic polynomial. Due to the uneven distribution of deformation scales throughout the motion sequence, different time intervals exhibit varying levels of complexity. As a result, the segmentation must be adaptive to reflect these differences. This adaptive segmentation ensures both optimal compression and accurate motion representation.

Algorithm 1 Adaptive Sequence Splitting

Require: Sequence $\vec{p} \in \mathbb{R}^{T \times C}$; Max number of splits K

```

1: # 1. Dynamic Programming
2: Initialize  $E[t, t] \leftarrow \infty$  for all  $k \in [K], t \in [T]$ 
3: Initialize  $\text{Prev}[k, t] \leftarrow 1$  for all  $k \in [K], t \in [T]$ 
4: Initialize  $\text{Cost}[k, t] \leftarrow \infty$  for all  $k \in [K], t \in [T]$ 
5: for  $k = 1$  to  $K$  do
6:   for  $t = T$  down to 1 and  $s = t - 1$  down to 1 do
7:     if  $k = 1$ 
8:        $E[s, t] \leftarrow \left\| \left( \vec{T}_{[s:t]}^T \left( \vec{T}_{[s:t]}^T \vec{T}_{[s:t]} \right)^{-1} \vec{T}_{[s:t]}^T - I \right) \vec{p}_{[s:t]} \right\|^2$ 
9:       if  $s = 1$ :  $\text{Cost}[1, t] \leftarrow E[1, t]$ 
10:    else
11:      if  $\text{Cost}[k, t] > \text{Cost}[k-1, s] + E[s, t]$  do
12:         $\text{Cost}[k, t] \leftarrow \text{Cost}[k-1, s] + E[s, t]$ 
13:         $\text{Prev}[k, t] \leftarrow s$ 
14: # 2. Traceback
15: Initialize  $Tr \leftarrow \{T\}$ 
16: Initialize  $\hat{K} \leftarrow \operatorname{argmin}_k \text{Cost}[k, T]$ 
17: while  $Tr[1] \neq 1$  do
18:   Insert  $\text{Prev}[\hat{K}, Tr[1]]$  to the head of  $Tr$ .
19:    $\hat{K} \leftarrow \hat{K} - 1$ 
20: return  $Tr$ 

```

4.3.2 Adaptive Sequence Splitting. To achieve the global optimal segmentation-wise fitting results, it is essential to determine the optimal splitting strategy. The problem is formulated as follows: given a sequence $\{p^t, t \in [T]\}$ and a maximum number of split segments K , we aim to find a set of splitting points $\{\hat{t}_k, k \in [K+1]\}$ that satisfies the conditions $\hat{t}_k < \hat{t}_{k+1}$, $\hat{t}_1 = 1$, and $\hat{t}_{K+1} = T$. These splitting points divide the sequence into K segments, each of which is fitted by a cubic polynomial.

We solve this problem through graph modeling, reducing it to finding the least-cost path on the graph within a distance of K . Each point p^t is represented as a node, and the cubic fitting error $E[i, j]$ (for $i \leq j$) between p^i and p^j is treated as the cost from node j to node i . As a result, the problem is reformulated as finding the least-cost path from node T to node 0 within a distance of K . The nodes along the identified path correspond to the splitting points $\{\hat{t}_k \mid k \in [K+1]\}$. This optimization problem can be efficiently solved using dynamic programming. The pseudo-code for the algorithm is provided in

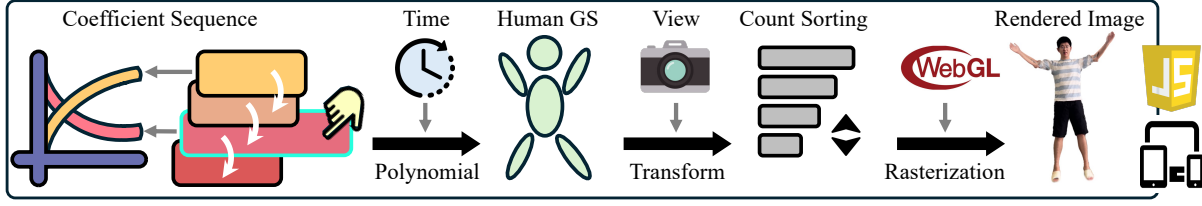


Figure 6: Overview of on-edge rendering for dynamic Gaussians. Two strategies are employed to accelerate Gaussian Splatting: (1) dynamic count sorting and (2) disentangled sorting and rasterization.

Alg. 1. To identify the most representative points, we first select a subset of points based on their importance, calculated as a combination of opacity and scale, which ensures that larger and more opaque points are prioritized. From these, we use farthest point sampling to select a diverse set of points that effectively capture the spatial structure. In practice, we collect traces of the 10 most representative points from the human limbs and head, concatenating them into $\vec{p} \in \mathbb{R}^{T \times 10 \times 3}$. To reduce computational complexity while maintaining results close to optimal, we uniformly sample key frames every 20 frames as nodes to build a sparse graph.

4.3.3 Data Transmission. The compressed polynomial coefficient sequence can be efficiently transmitted to mobile devices with minimal latency. Furthermore, the sequence supports streaming transmission, as the data format and the size of each frame are predefined and consistent. This enables progressive delivery and decoding, allowing playback to begin as soon as the initial frames are received while subsequent frames continue to stream seamlessly in the background.

4.4 Mobile Rendering for Dynamic Gaussian

With the poly-compressed 3DGS sequence transmitted to the mobile devices, the next step is to render these dynamic Gaussians efficiently, even with limited computational resources. To accomplish this, we propose two strategies to enhance rendering efficiency, as illustrated in Fig. 6.

4.4.1 Dynamic Count Sorting. One of the most computationally intensive tasks in Gaussian Splatting is sorting the Gaussians by depth to ensure correct occlusion relationships during rendering. Since both the Gaussians and the user-controlled camera are dynamic, the rendering order must be continuously updated to ensure accurate results. At each timestep, we select the polynomial parameters corresponding to the current split and compute the positions of the Gaussians. Using these positions, we calculate their depth relative to the camera and quantize the depth values into 256 bins. The Gaussians are then sorted using a counting sort algorithm. This approach involves two steps: first, traversing the Gaussian list to record indices corresponding to each bin, and second, iterating through the bin lists from front to back to place the Gaussians into a new array. By quantizing depth and using count sorting, we can reduce the computational complexity from $O(N \log(N))$ to $O(N)$, enabling faster rendering of dynamic Gaussians on resource-constrained mobile edge devices.

4.4.2 Disentangled Sorting and Rasterization. Despite the efficiency of dynamic count sorting, achieving real-time rendering of Gaussian Splatting on mobile devices, such as smartphones with limited computational power, remains challenging. However, mobile devices are

more frequently used due to their convenience, making it essential to minimize rendering latency and maintain high frame rates to avoid any perceptible lag during user interactions. To reduce rendering delays and deliver smooth real-time performance, we propose a parallel strategy that disentangles sorting from rasterization. Observing that both camera movement and Gaussian dynamics are continuous, we separate these processes into two independent threads. Rasterization, which is inherently fast, is implemented using WebGL2 with a custom shader for rendering dynamic Gaussian point clouds. Meanwhile, the sorting thread continuously updates the sorting results and stores them in a cache. The rasterization thread then reads the cached sorting results, ensuring real-time rendering without introducing computational bottlenecks. Since the changes in camera movement and Gaussian dynamics are gradual, the cached sorting results are usually sufficient for rendering, with minimal discrepancies. This approach can achieve smooth and efficient rendering even on devices with limited computational resources.

5 Implementation

Client. The client-side implementation is responsible for video capture, lightweight rendering, and user interaction. In our implementation, users record short monocular videos with a mobile phone, capturing full-body poses while spinning or performing casual movements to provide complete front and back views. No special capture skills are required beyond keeping the full body in view and avoiding excessively fast motions during the brief recording. The client devices render animated avatars streamed from the server. Rendering is implemented using JavaScript and WebGL2, ensuring compatibility across a wide range of devices, including desktops, laptops, tablets, and smartphones. The system is accessed via the Safari browser, requiring no additional software installation or specialized hardware such as GPUs, making it lightweight and accessible. We further provide a lightweight web-based *HoloMobile* Player for cross-platform playback and interaction. The player supports timeline scrubbing, viewpoint manipulation, and playback-speed adjustment for dynamic Gaussian avatars directly in the browser, as shown in Fig. 7.

Server. The server-side computer handles computationally intensive tasks of video processing, avatar reconstruction, and animation. The server hardware includes an NVIDIA RTX 3090 GPU and an Intel Xeon Gold 8375C CPU (2.9 GHz), providing the necessary computational resources for high-resolution video processing and real-time animation. The server software pipeline is built on PyTorch [41] and integrates key components to ensure accurate and efficient processing. Following ExAvatar [37], we utilize MMPose [4] to detect 2D keypoints from the input monocular video. DECA [6] is employed

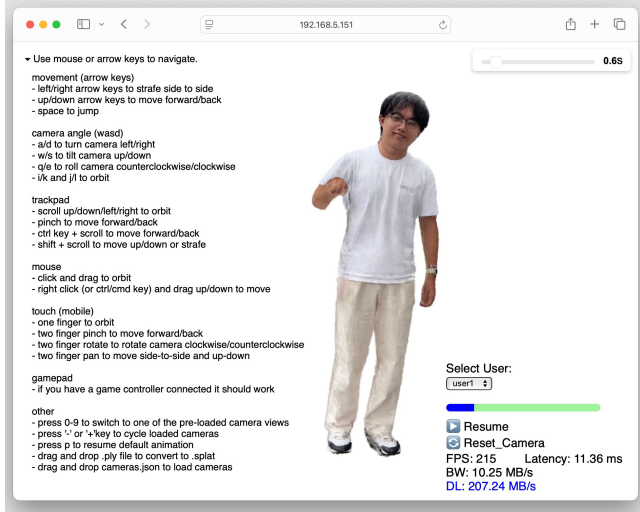


Figure 7: HoloMobile Player running in a browser on a MacBook Air. The interface supports timeline control, viewpoint manipulation, and real-time rendering of our dynamic Gaussian avatar.

to estimate facial geometry, while Hand4Whole [36] generates initial body and hand parameters. These steps produce coarse FLAME and SMPL-X models, which serve as the basis for detailed avatar reconstruction and animation.

6 Evaluation

6.1 Experimental Setup

Datasets. We evaluate our system on the NeuMan dataset [18] and a self-collected DIVERse Avatar (DIVA) dataset. NeuMan consists of short monocular videos (10–20 seconds) captured in real-world environments using mobile phones, featuring individuals performing motions such as walking. We collected DIVA with an iPhone 15 to reflect diverse downstream application scenarios. It includes 20 avatars (14 males and 6 females) with diverse body types (height: 155–190 cm, weight: 45–100 kg) and varied clothing styles, performing nine activities: dance, yoga, boxing, basketball, badminton, bicep curls, jumping jacks, squats, and a simulated conference. For training, participants performed a 5–10 second rotational motion, resulting in 300–600 frames with 1920×1080 resolutions. Testing involved performing the specified activities to evaluate the performance of avatars on novel motions across diverse scenarios.

Baselines. For monocular full-body avatar reconstruction, we compare *HoloMobile* with state-of-the-art methods, including InstantAvatar [17], HUGS [23], and ExAvatar [37]. Additionally, we compare against the streaming telepresence system MagicStream [3], which achieves streaming rendering with GPUs. Other telepresence systems, such as FarfetchFusion [27] and MonoFlex3D [63], are excluded from the comparison because they focus solely on facial reconstruction rather than full-body avatars. For Gaussian sequence compression and mobile rendering, we compare against V^3 [53], which bakes sequential Gaussian attributes into a 2D video by leveraging hardware video codecs with a high compression rate.

Table 1: Runtime analysis on different mobile devices for rendering at a resolution of 1920×1080 .

Metric	MacBook Air	iPhone 17 Pro	iPad Pro 3	Vision Pro
Latency (ms)	10.62	4.13	6.58	2.92
FPS	207	221	217	237

Metrics. We measure visual quality using three commonly used metrics: Peak Signal-to-Noise Ratio (PSNR), Structural Similarity Index Measure (SSIM) [55], and Learned Perceptual Similarity (LPIPS) [64]. We also record downloaded data volume (MB), bandwidth consumption, rendering latency, and FPS to quantify both network efficiency and on-device performance. For 3DGS compression, we additionally report the final data size, compression ratio, and compression time.

In this section, we first evaluate the complete system in **Sec. 6.2**, using a setup that aligns with real-world application scenarios. Both quantitative metrics and user feedback are analyzed to demonstrate the system’s practical usability and user satisfaction. In **Sec. 6.3.1**, we conduct a comprehensive evaluation of avatar reconstruction quality, leveraging state-of-the-art algorithms across various scenes that represent a wide range of real-world scenarios. Finally, in **Sec. 6.3.2**, we present experiments showcasing the efficiency and effectiveness of our compression strategy for dynamic 3DGS sequences.

6.2 System Evaluation

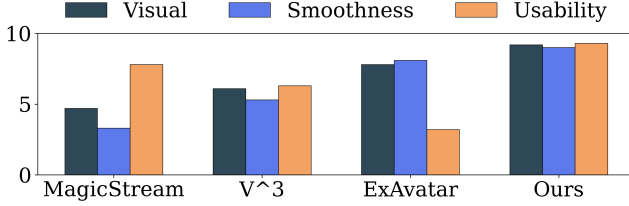
On-Mobile Runtime Analysis. We evaluate *HoloMobile* using key metrics such as latency, FPS, and bandwidth, which are critical for assessing user experience in real-world applications. The evaluation was conducted on four mobile devices: a laptop (MacBook Air with M3 chip), a smartphone (iPhone 17 Pro), a tablet (iPad Pro 3 with M1 chip), and Apple Vision Pro using the Safari browser. The results, summarized in Tab. 1, show that *HoloMobile* achieves a peak FPS of 221 and a minimum rendering latency of 4.13 ms on the iPhone 17 Pro. However, actual visual performance is influenced by device screen refresh rates (60 Hz or 120 Hz). The compressed representation requires only 10.22 Mbps, demonstrating efficiency suitable for downstream applications such as sports coaching, remote healthcare, and teleconferencing.

With-GPU Avatar Comparison. When the testbed has a consumer-grade GPU (e.g., PCs with an RTX 3090), our method can be rendered locally without requiring compression and transmission from the cloud. Using the locally deployed avatar, the incoming SMPL-X pose parameters transmitted via streaming are utilized to drive the avatar’s motion, enabling the presence of a 4D human viewable from any perspective. This setup aligns with the task defined in MagicStream [3]. We compare our method with MagicStream at the conference scene in the DIVA dataset across multiple dimensions, including model inference overhead, display fluency, and final rendering quality. As shown in Tab. 2, our method significantly reduces inference cost while achieving higher rendering FPS and superior visual quality. Although our method requires more GPU memory, it remains compatible with most consumer-grade GPUs with 10+ GB of memory.

Avatar Construction Time. To provide a clear assessment of practical deployment feasibility, we analyze the computational requirements for avatar construction. Avatar reconstruction is performed offline on a server equipped with a single NVIDIA RTX 3090 GPU

Table 2: Comparison of Streaming Avatar Rendering with Model Inference on a GPU (RTX 3090).

Method	GPU Memory ↓	FLOPs (G) ↓	FPS ↑	PSNR ↑
MagicStream [3]	32.98 MB	1047.45	14.83	26.97
Ours	1.68 GB	23.70	729.03	30.13

**Figure 8: Experience evaluation based on user ratings.**

and an Intel Xeon Gold 8375C CPU. For a 10-second monocular capture video (600 frames at 1920×1080 resolution), the reconstruction pipeline requires approximately one hour of training time, comprising 15 mins for pose estimation. Once reconstructed, the avatar can be used indefinitely without further training. This one-time offline cost is acceptable for the target applications, such as creating a personal avatar for long-term use in teleconferencing or fitness training. Moreover, consumer-grade GPU hardware is sufficient; no specialized multi-GPU clusters are required.

User Study. In addition to quantitative metrics, which may not fully capture the user experience, we conducted an IRB-approved in-person user study² with 10 participants (5 females and 5 males) to collect subjective scores for each system. Each system was tested by reconstructing a human avatar or dynamic human using its own implementation. Participants viewed the rendered results in person and rated visual quality, motion smoothness, and overall usability. To minimize bias, we used a standardized script for all sessions and provided the same method-level information for every system, including input modality, avatar support, cross-platform accessibility, mobile support, and streamability. The results, shown in Fig. 8, demonstrate that our system achieved the highest scores for visual satisfaction and smoothness. In contrast, MagicStream [3] and V³ rendered dynamic scenes on a per-frame basis rather than using a unified 3D representation, leading to noticeable flickering and inconsistencies in the results. We also observed that users tended to give higher ratings to systems with avatars, as these offer flexible, animatable human assets compared to "static" volumetric videos. Moreover, mobile accessibility also plays a crucial role in usability ratings, as evidenced by the lower usability rating of ExAvatar.

6.3 Evaluation Results of Key Modules

6.3.1 Avatar Reconstruction Evaluation. To evaluate our hybrid explicit-implicit avatar model, we conduct a comprehensive comparison of the reconstruction quality achieved by our system and state-of-the-art methods for monocular avatar reconstruction. The performance of each method is evaluated on both public and self-collected datasets, with results presented for each scene. Notably, our

²This study was approved by the Institutional Review Board (IRB) of the authors' institution.

Table 3: Quantitative comparison of dynamic motion reconstruction on the NeuMan dataset [18].

	PSNR↑	SSIM↑	LPIPS↓
InstantAvatar [17]	24.19	0.9411	0.0707
HUGS [23]	22.23	0.7478	0.1187
ExAvatar [37]	34.61	0.9848	0.0092
MagicStream [3]	30.44	0.9753	0.0228
Ours	35.87	0.9866	0.0086

Table 4: Quantitative comparison of novel motion animation on the DIVA dataset.

	PSNR↑	SSIM↑	LPIPS↓
InstantAvatar [17]	18.49	0.9001	0.1387
HUGS [23]	15.35	0.6877	0.3446
ExAvatar [37]	25.56	0.9606	0.0563
MagicStream [3]	22.62	0.9511	0.0722
Ours	30.36	0.9785	0.0413

collected DIVA dataset encompasses a wide range of diverse scenes, reflecting practical scenarios commonly encountered in real-world applications.

NeuMan Dataset. Following ExAvatar [37], we evaluate and compare the rendering quality of the foreground (human) only. As shown in Tab. 3, our method surpasses all baseline methods across all metrics. The qualitative comparison in Fig. 9 further highlights these differences, where baseline methods often introduce artifacts, such as distorted faces, and fail to capture fine details, including clothing wrinkles and patterns. In contrast, our approach produces clean, accurate, and visually realistic human renderings while preserving intricate details like facial features and clothing patterns.

DIVA Dataset. To further evaluate our method, we conducted experiments on the more challenging DIVA dataset, which contains complex scenarios. As shown in Tab. 4, our method continues to outperform all baselines across all metrics. The disparity is even more evident in the qualitative results presented in Fig. 10. Notably, even ExAvatar [37], which performs comparably to our method on the NeuMan dataset, struggles with reconstructing fingers, muscle contours, and clothing patterns in DIVA. Thanks to our hybrid explicit-implicit model, our method generalizes effectively to out-of-distribution poses while producing plausible pose-dependent surface and appearance deformations.

6.3.2 Gaussian Sequence Compression Evaluation. We evaluate the effectiveness of our compression method on the Yoga scene from the DIVA dataset, which consists of 600 frames captured at 60 FPS, representing 10 seconds of dynamic 3D Gaussian sequences.

Comparison of Sequential 3DGS Compression. We compare our method with V³ [53] on 3DGS sequence compression. The results, summarized in Tab. 5, demonstrate that our method significantly outperforms the V³ in terms of compression efficiency and bandwidth reduction. Specifically, our method reduces the original 3DGS sequence data size from 23.16 GB to 102.17 MB, achieving a compression ratio of 0.43%, which is approximately 7.7 times smaller than the compressed size achieved by V³ (3.3%). This substantial reduction

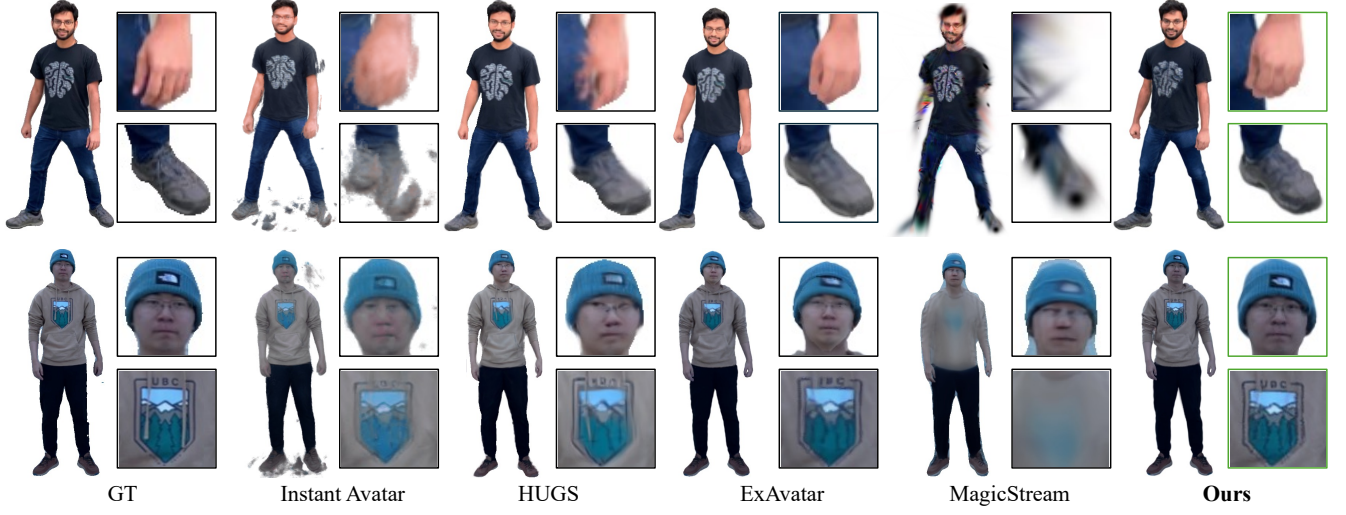


Figure 9: Qualitative Results on NeuMan [18] Dataset. Our method achieves state-of-the-art performance on widely used public datasets, delivering significantly clearer details.

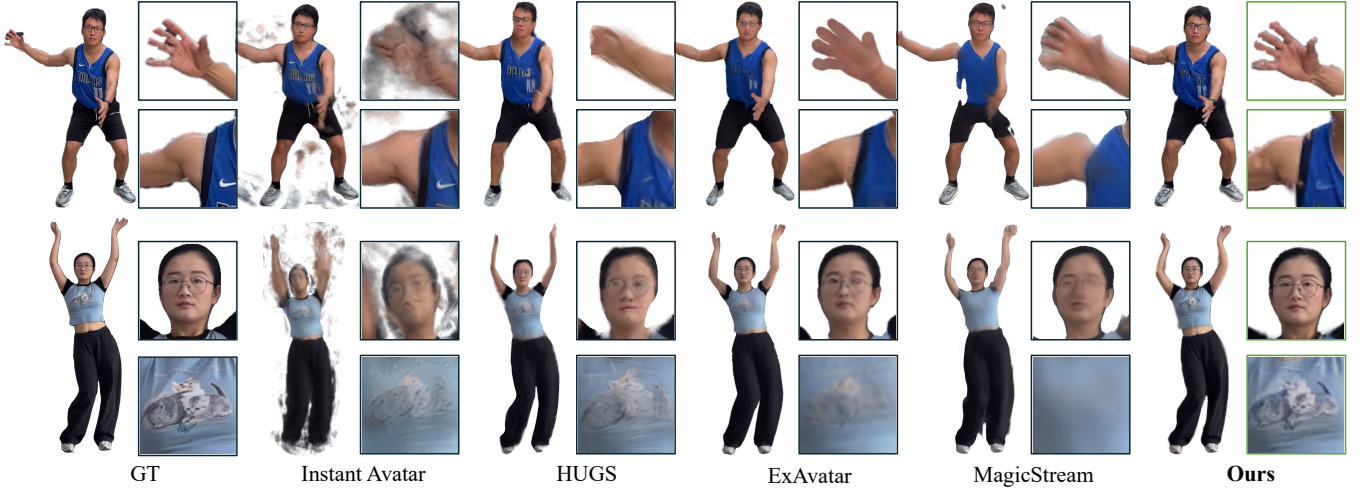


Figure 10: Qualitative results on the DIVA dataset. Our method effectively handles inconsistencies caused by facial expressions and clothing wrinkles, whereas other methods appear blurry in these regions.

also translates into a lower bandwidth requirement of 10.22 Mbps compared to 78.41 Mbps for V^3 , highlighting the suitability of our approach for bandwidth-constrained applications such as real-time streaming or telepresence systems. Moreover, our method reduces sequence compression time by 7.4 $\times$ compared with V^3 . This improvement is achieved through the CUDA-based parallel polynomial fitting. The compressed 3DGS sequence demonstrates a very similar rendering PSNR. However, it is important to note that we use the 3DGS sequence generated by our reconstructed avatar for a purely compression-based comparison. This is because V^3 is designed for videos captured using multi-view camera arrays and performs poorly when reconstructing dynamic scenes in a monocular setting. These results demonstrate the effectiveness of cubic polynomial fitting for compressing motion tracks in dynamic 3D Gaussian sequences,

Table 5: Gaussian sequence compression.

Method	Gaussian Sequence	V^3	Ours
Data Size (MB) ↓	23715.05	784.15	102.17
Compression Ratio ↑	N/A	96.70%	99.57%
Compression Time (s) ↓	N/A	196.5	26.5
PSNR (dB) ↑	31.66	31.08	31.14

particularly for human-centered scenes with smooth and continuous trajectories.

Sequence Splitting Strategy Comparison. We evaluate three sequence splitting strategies: Even Split, Graph Dense, and Graph Sparse, analyzing their performance in terms of fitting error and computation time, as illustrated in Fig. 11. As described in Sec. 4.3.2,

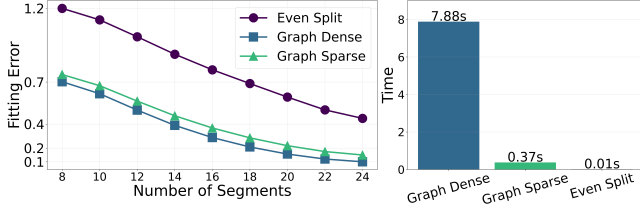


Figure 11: Splitting strategy analysis.

we construct a sparse graph using keyframes as graph nodes every 20 frames. Without this sparse sampling strategy, the graph becomes dense, leading to a significant increase in computational complexity. The Even Split method, which divides the sequence into equal-length intervals without accounting for motion complexity, consistently shows the highest fitting error across all segment numbers. In contrast, Graph Dense achieves the lowest fitting error by identifying the optimal splitting points. However, this comes at the cost of significantly higher computation time, requiring 7.88 seconds. Graph Sparse strikes an effective balance by sampling boundaries less frequently. While it results in a slightly higher fitting error compared to Graph Dense, it drastically reduces computation time to just 0.37 seconds. This makes it 21× faster than Graph Dense while maintaining comparable segmentation quality. Notably, Even Split requires twice as many segments to achieve a similar error.

Effect of the number of segments. We also investigate the impact of the number of splits K on rate–distortion performance. As shown in Fig. 12, using only $K = 20$ segments, our method achieves 27.50 dB PSNR with a data size of 204 MB, corresponding to over 70× compression compared to the 14.89 GB per-frame Gaussian upper bound, while incurring negligible perceptual degradation. The performance gains saturate beyond $K = 10$, indicating that a moderate number of segments is sufficient to capture most temporal motion continuity.

7 Discussion

Limitations of Avatar Reconstruction. While HoloMobile delivers high-fidelity avatar reconstruction, more challenging scenarios remain for future study. Although the hybrid explicit–implicit motion model enables precise representation of the human body, it does not yet account for geometry beyond the body surface, especially during human–object interactions. HoloMobile currently cannot represent 4D humans interacting with external objects such as sports equipment, teaching aids, or musical instruments, limiting its use in applications like dumbbell-based fitness training and virtual classroom instruction. A promising direction for future work is to incorporate universal objects into the avatar representation, allowing joint modeling of humans and objects and thereby expanding the system’s applicability.

Web-based Rendering Constraints. Rendering dynamic 3DGS sequences on mobile devices must account for inherent constraints in computational and memory resources. In our current design, the web-based rendering module achieves real-time performance for single-user scenarios, but scaling to a large number of simultaneous avatars or extremely complex scenes remains an open challenge. Future improvements may further integrate native rendering APIs and hardware-specific optimizations.

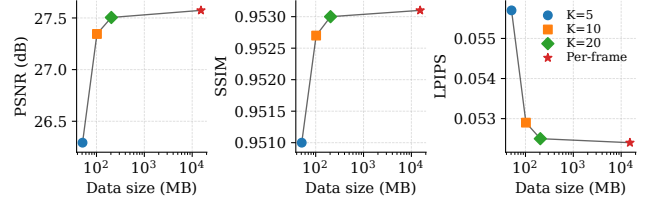


Figure 12: Splitting analysis with quality-size curves.

Multi-Avatar Scalability. While the current system is designed for single-user avatar scenarios, extending it to support multi-avatar applications—such as multi-party teleconferencing or collaborative virtual environments—requires only minimal architectural modifications. The primary adjustment involves reformulating the graph construction in the adaptive segmentation module (Sec. 4.3.2) to jointly model all avatars in a shared graph, enabling globally optimal segmentation across multiple subjects. Since each avatar’s polynomial-compressed 3DGS sequence is independently decodable, the client rendering pipeline can be extended to load and render multiple avatar streams in parallel. The main bottleneck in multi-avatar scenarios shifts from compression efficiency to rendering throughput and bandwidth aggregation: rendering M simultaneous avatars requires approximately $M \times$ the Gaussian count and bandwidth of a single avatar. With our current compression achieving 10.22 Mbps for a single avatar, multi-party scenarios (e.g., $M = 4$) would require approximately 40 Mbps—within the range of modern mobile broadband connections. Exploring tighter integration with hardware-accelerated rendering APIs and dynamic Gaussian culling strategies for multi-avatar scenes represents a promising direction for future work.

8 Conclusion

In this paper, we present *HoloMobile*, an end-to-end mobile system for reconstructing high-fidelity human avatars from mobile-captured monocular videos and enabling efficient compression, transmission, and rendering on mobile devices. By leveraging a hybrid explicit–implicit motion model, *HoloMobile* achieves photorealistic avatar reconstruction while capturing fine details such as cloth wrinkles and soft-tissue deformation. To support efficient mobile delivery, we introduce adaptive segmentation and polynomial compression for dynamic 3DGS sequences, together with a web-based rendering module that enables real-time interactive playback across heterogeneous mobile platforms. These results demonstrate the feasibility of practical mobile avatar applications such as teleconferencing, remote care, and fitness training.

Acknowledgments

This paper is supported in part by the Research Grants Council (RGC) of Hong Kong under grants STG1/E-403/24-N and CRF C4032-25E.

References

- [1] Jianchuan Chen, Jingchuan Hu, Gaige Wang, Zhonghua Jiang, Tiansong Zhou, Zhiwen Chen, and Chengfei Lv. 2025. TaoAvatar: Real-Time Lifelike Full-Body Talking Avatars for Augmented Reality via 3D Gaussian Splatting. In *Proceedings of the Computer Vision and Pattern Recognition Conference*. 10723–10734.

- [2] Yushuo Chen, Zerong Zheng, Zhe Li, Chao Xu, and Yebin Liu. 2024. MeshAvatar: Learning High-quality Triangular Human Avatars from Multi-view Videos. arXiv:2407.08414 [cs.CV] <https://arxiv.org/abs/2407.08414>
- [3] Ruizhi Cheng, Nan Wu, Vu Le, Eugene Chai, Matteo Varvello, and Bo Han. 2024. Magicstream: Bandwidth-conserving immersive telepresence via semantic communication. In *Proceedings of the 22nd ACM Conference on Embedded Networked Sensor Systems*. 365–379.
- [4] MMPose Contributors. 2020. OpenMMLab Pose Estimation Toolbox and Benchmark. <https://github.com/open-mmlab/mmpose>.
- [5] Tiancheng Fang, Chaoyue Niu, Yujie Sun, Chengfei Lv, Xiaotang Jiang, Ben Xue, Fan Wu, and Guihai Chen. 2024. An End-to-End, Low-Cost, and High-Fidelity 3D Video Pipeline for Mobile Devices. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. 1162–1176.
- [6] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. 2021. Learning an Animatable Detailed 3D Face Model from In-The-Wild Images. *ACM Transactions on Graphics (Proc. SIGGRAPH)* 40, 8. <https://doi.org/10.1145/3450626.3459936>
- [7] Shubham Goel, Georgios Pavlakos, Jathushan Rajasegaran, Angjoo Kanazawa, and Jitendra Malik. 2023. Humans in 4d: Reconstructing and tracking humans with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14783–14794.
- [8] Danilo Graziosi, Ohji Nakagami, Satoru Kuma, Alexandre Zaghetto, Teruhiko Suzuki, and Ali 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.
- [9] Yongjie Guan, Xueyu Hou, Nan Wu, Bo Han, and Tao Han. 2023. Metastream: Live volumetric content capture, creation, delivery, and rendering in real time. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.
- [10] Chen Guo, Tianjian Jiang, Xu Chen, Jie Song, and Otmar Hilliges. 2023. Vid2Avatar: 3D Avatar Reconstruction from Videos in the Wild via Self-supervised Scene Decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [11] Chen Guo, Junxuan Li, Yash Kant, Yaser Sheikh, Shunsuke Saito, and Chen Cao. 2025. Vid2Avatar-Pro: Authentic Avatar from Videos in the Wild via Universal Prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- [12] Bo Han, Yu Liu, and Feng Qian. 2020. ViVo: Visibility-aware mobile volumetric video streaming. In *Proceedings of the 26th annual international conference on mobile computing and networking*. 1–13.
- [13] Liangxiao Hu, Hongwen Zhang, Yuxiang Zhang, Boyao Zhou, Boning Liu, Shengping Zhang, and Liqiang Nie. 2024. Gaussianavatar: Towards realistic human avatar modeling from a single video via animatable 3d gaussians. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 634–644.
- [14] Yi-Hua Huang, Yang-Tian Sun, Ziyi Yang, Xiaoyang Lyu, Yan-Pei Cao, and Xiaojuan Qi. 2024. Sc-gs: Sparse-controlled gaussian splatting for editable dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 4220–4230.
- [15] Forrest Iandola, Stanislav Pidhorskyi, Igor Santesteban, Divam Gupta, Anuj Pahuja, Nemanja Bartolovic, Frank Yu, Emanuel Garbin, Tomas Simon, and Shunsuke Saito. 2025. SqueezeMe: Mobile-Ready Distillation of Gaussian Full-Body Avatars. In *SIGGRAPH ACM*. <https://arxiv.org/abs/2412.15171>
- [16] Biao Jiang, Xin Chen, Wen Liu, Jingyi Yu, Gang Yu, and Tao Chen. 2023. Motiongpt: Human motion as a foreign language. *Advances in Neural Information Processing Systems* 36 (2023), 20067–20079.
- [17] Tianjian Jiang, Xu Chen, Jie Song, and Otmar Hilliges. 2023. Instantavatar: Learning avatars from monocular video in 60 seconds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 16922–16932.
- [18] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. 2022. NeuMan: Neural Human Radiance Field from a Single Video. In *Proceedings of the European conference on computer vision (ECCV)*.
- [19] Zhihui Ke, Xiaobo Zhou, Yuyang Liu, Zhizhuo Pang, and Tie Qiu. 2025. EdgeGaussian: Real-time Free-Viewpoint Video for Mobile VR via Edge-Client Collaborative Neural Rendering. In *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking*. 773–787.
- [20] Mark Kellogg. 2024. GaussianSplat3D: Three.js-based implementation of 3D Gaussian splatting. <https://github.com/mkkellogg/GaussianSplat3D>
- [21] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.* 42, 4 (2023), 139–1.
- [22] Gunjoong Kim, Seonghoon Park, Jeho Lee, Chanyoung Jung, Hyungchol Jun, and Hojung Cha. 2025. Vega: Fully Immersive Mobile Volumetric Video Streaming with 3D Gaussian Splatting. In *Proceedings of the 31st Annual International Conference on Mobile Computing and Networking*. 1106–1120.
- [23] Muhammed Kocabas, Jen-Hao Rick Chang, James Gabriel, Oncel Tuzel, and Anurag Ranjan. 2024. Hugs: Human gaussian splats. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 505–515.
- [24] Nikos Kolotourous, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. 2019. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *Proceedings of the IEEE/CVF international conference on computer vision*. 2252–2261.
- [25] Kevin Kwok. 2023. splat: WebGL implementation of a real-time renderer for 3D Gaussian Splatting for Real-Time Radiance Field Rendering. <https://github.com/antimatter15/splat>
- [26] Hojeong Lee, Yu Hong Kim, Sangwoo Ryu, James Won-Ki Hong, Sangtae Ha, and Seyeon Kim. 2025. DeltaStream: 2D-Inferred Delta Encoding for Live Volumetric Video Streaming. In *Proceedings of the 23rd Annual International Conference on Mobile Systems, Applications and Services*. 361–373.
- [27] Kyungjin Lee, Juheon Yi, and Youngki Lee. 2023. Farfetchfusion: Towards fully mobile live 3d telepresence platform. In *Proceedings of the 29th Annual International Conference on Mobile Computing and Networking*. 1–15.
- [28] Kyungjin Lee, Juheon Yi, Youngki Lee, Sunghyun Choi, and Young Min Kim. 2020. GROOT: A real-time streaming system of high-fidelity volumetric videos. In *Proceedings of the 26th Annual International Conference on Mobile Computing and Networking*. 1–14.
- [29] John P Lewis, Matt Corder, and Nickson Fong. 2023. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *Seminal Graphics Papers: Pushing the Boundaries, Volume 2*. 811–818.
- [30] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. 2024. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 8508–8520.
- [31] Yu Liu, Bo Han, Feng Qian, Arvind Narayanan, and Zhi-Li Zhang. 2022. Vues: practical mobile volumetric video streaming through multiview transcoding. In *Proceedings of the 28th Annual International Conference on Mobile Computing and Networking*. 514–527. doi:10.1145/3495243.3517027
- [32] Yu Liu, Puqi Zhou, Zejun Zhang, Anlan Zhang, Bo Han, Zhenhua Li, and Feng Qian. 2024. MuV2: Scaling up Multi-user Mobile Volumetric Video Streaming via Content Hybridization and Sharing. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*. 327–341. doi:10.1145/3636534.3649364
- [33] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-Person Linear Model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)* 34, 6 (Oct. 2015), 248:1–248:16.
- [34] Naureen Mahmood, Nima Ghorbani, Nikolaus F Troje, Gerard Pons-Moll, and Michael J Black. 2019. AMASS: Archive of motion capture as surface shapes. In *Proceedings of the IEEE/CVF international conference on computer vision*. 5442–5451.
- [35] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.
- [36] Gyeongsik Moon, Hongsuk Choi, and Kyoung Mu Lee. 2022. Accurate 3D Hand Pose Estimation for Whole-Body 3D Human Mesh Estimation. In *Computer Vision and Pattern Recognition Workshop (CVPRW)*.
- [37] Gyeongsik Moon, Takaaki Shiratori, and Shunsuke Saito. 2024. Expressive whole-body 3D gaussian avatar. In *European Conference on Computer Vision*. Springer, 19–35.
- [38] Arthur Moreau, Jifei Song, Helisa Dharmo, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero. 2024. Human gaussian splatting: Real-time rendering of animatable avatars. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 788–798.
- [39] Sergio Orts-Escolano, Christoph Rhemann, Sean Fanello, Wayne Chang, Adarsh Kowdle, Yury Degtyarev, David Kim, Philip L Davidson, Sameh Khamis, Mingsong Dou, et al. 2016. Holoportation: Virtual 3d teleportation in real-time. In *Proceedings of the 29th annual symposium on user interface software and technology*. 741–754.
- [40] Haokai Pang, Heming Zhu, Adam Kortylewski, Christian Theobalt, and Marc Habermann. 2024. Ash: Animatable gaussian splats for efficient and photoreal human rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 1165–1175.
- [41] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).
- [42] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. 2019. Expressive Body Capture: 3D Hands, Face, and Body from a Single Image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. 10975–10985.
- [43] Sida Peng, Juntao Dong, Qianqian Wang, Shangzhan Zhang, Qing Shuai, Xiaowei Zhou, and Hujun Bao. 2021. Animatable neural radiance fields for modeling dynamic human bodies. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 14314–14323.
- [44] Gerard Pons-Moll, Javier Romero, Naureen Mahmood, and Michael J Black. 2015. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–14.
- [45] Zhiyin Qian, Shaofei Wang, Marko Mihajlovic, Andreas Geiger, and Siyu Tang. 2024. 3dgs-avatar: Animatable avatars via deformable 3d gaussian splatting. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5020–5030.
- [46] Kaiyue Shen, Chen Guo, Manuel Kaufmann, Juan Zarate, Julien Valentin, Jie Song, and Otmar Hilliges. 2023. X-Avatar: Expressive Human Avatars. *Computer Vision and Pattern Recognition (CVPR)*.

- [47] Chao Shi, Shenghao Jia, Jinhui Liu, Yong Zhang, Liangchao Zhu, Zhonglei Yang, Jinze Ma, Chaoyue Niu, and Chengfei Lv. 2025. HRM² 2Avatar: High-Fidelity Real-Time Mobile Avatars from Monocular Phone Scans. *arXiv preprint arXiv:2510.13587* (2025).
- [48] Yuang Shi, Géraldine Morin, Simone Gasparini, and Wei Tsang Ooi. 2025. Lapisgs: Layered progressive 3d gaussian splatting for adaptive streaming. In *2025 International Conference on 3D Vision (3DV)*. IEEE, 991–1000.
- [49] Jos Stam and Charles Loop. 2003. Quad/triangle subdivision. In *Computer Graphics Forum*, Vol. 22. Wiley Online Library, 79–85.
- [50] Shih-Yang Su, Frank Yu, Michael Zollhöfer, and Helge Rhodin. 2021. A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose. *Advances in neural information processing systems* 34 (2021), 12278–12291.
- [51] Yuan-Chun Sun, Yuang Shi, Cheng-Tse Lee, Mufeng Zhu, Wei Tsang Ooi, Yao Liu, Chun-Ying Huang, and Cheng-Hsin Hsu. 2025. LTS: A DASH streaming system for dynamic multi-layer 3D Gaussian splatting scenes. In *Proceedings of the 16th ACM Multimedia Systems Conference*. 136–147.
- [52] Yi-Zhen Tsai, Xuechen Zhang, Zheng Li, and Jiasi Chen. 2025. L3GS: Layered 3D Gaussian Splats for Efficient 3D Scene Delivery. *arXiv preprint arXiv:2504.05517* (2025).
- [53] Penghao Wang, Zhirui Zhang, Liao Wang, Kaixin Yao, Siyuan Xie, Jingyi Yu, Minye Wu, and Lan Xu. 2024. V³: Viewing Volumetric Videos on Mobiles via Streamable 2D Dynamic Gaussians. *ACM Transactions on Graphics (TOG)* 43, 6 (2024), 1–13.
- [54] Yuan Wang, Di Huang, Yaqi Zhang, Wanli Ouyang, Jile Jiao, Xuetao Feng, Yan Zhou, Pengfei Wan, Shixiang Tang, and Dan Xu. 2024. Motiongpt-2: A general-purpose motion-language model for motion generation and understanding. *arXiv preprint arXiv:2410.21747* (2024).
- [55] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.
- [56] Jing Wen, Xiaoming Zhao, Zhongzheng Ren, Alexander G. Schwing, and Shenlong Wang. 2024. GoMAvatar: Efficient Animatable Human Modeling from Monocular Video Using Gaussians-on-Mesh. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2059–2069.
- [57] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 2024. 4d gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 20310–20320.
- [58] Hongyi Xu, Thiemo Alldieck, and Cristian Sminchisescu. 2021. H-nerf: Neural radiance fields for rendering and temporal reconstruction of humans in motion. *Advances in Neural Information Processing Systems* 34 (2021), 14955–14966.
- [59] Donggyu Yang, Wooseung Nam, Byunggu Kang, and Kyunghan Lee. 2025. Empilo: Realizing Immersive Mobile 3D Video Conferencing through Parameterized Communication. In *Proceedings of the 23rd Annual International Conference on Mobile Systems, Applications and Services*. 374–387.
- [60] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. 2024. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 20331–20341.
- [61] Yuheng Yuan, Qiuhong Shen, Xingyi Yang, and Xinchao Wang. 2025. 1000+ fps 4d gaussian splatting for dynamic scene rendering. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.
- [62] Anlan Zhang, Chendong Wang, Bo Han, and Feng Qian. 2022. YuZu: Neural-Enhanced Volumetric Video Streaming. In *19th USENIX Symposium on Networked Systems Design and Implementation (NSDI 22)*. 137–154.
- [63] Chongli Zhang, Liyuan Cuan, Yufan Yang, Liang Qiao, Xiuquan Qiao, and Yakun Huang. 2025. MonoFlex3D: Low-Latency Glasses-Free 3D Teleconference from a Single Camera. In *Proceedings of the 3rd ACM Workshop on Mobile Immersive Computing, Networking, and Systems*. 1–8.
- [64] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. 2018. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 586–595.
- [65] Xinjie Zhang, Zhening Liu, Yifan Zhang, Xingtong Ge, Dailan He, Tongda Xu, Yan Wang, Zehong Lin, Shuicheng Yan, and Jun Zhang. 2025. Mega: Memory-efficient 4d gaussian splatting for dynamic scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 27828–27838.
- [66] Zerong Zheng, Xiaochen Zhao, Hongwen Zhang, Boning Liu, and Yebin Liu. 2023. Avatarrex: Real-time expressive full-body avatars. *ACM Transactions on Graphics (TOG)* 42, 4 (2023), 1–19.
- [67] Mufeng Zhu, Mingju Liu, Cunxi Yu, Cheng-Hsin Hsu, and Yao Liu. 2025. SGSS: Streaming 6-DoF Navigation of Gaussian Splat Scenes. In *Proceedings of the 16th ACM Multimedia Systems Conference*. 46–56.
- [68] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. 2001. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. 371–378.