An Interactive Conversational 3D Virtual Human

Richard Shaw†, Youngkyoon Jang†, Athanasios Papaioannou†, Arthur Moreau†, Helisa Dhamo, Zhensong Zhang, Eduardo Pérez-Pellitero

Noah's Ark Lab, Huawei, London, UK.

Contributing authors: {richard.shaw1; youngkyoonjang; athanasios.papaioannou; arthur.moreau3; helisa.dhamo; zhangzhensong; e.perez.pellitero}@huawei.com;
†These authors contributed equally to this work.

Abstract

This work presents *Interactive Conversational 3D Virtual Human* (ICo3D), a method for generating an interactive, conversational, and photorealistic 3D human avatar. Based on multi-view captures of a subject, we create an animatable 3D face model and a dynamic 3D body model, both rendered by splatting Gaussian primitives. Once merged together, they represent a lifelike virtual human avatar suitable for real-time user interactions. We equip our avatar with an LLM for conversational ability. During conversation, the audio speech of the avatar is used as a driving signal to animate the face model, enabling precise synchronization. We describe improvements to our dynamic Gaussian models that enhance photorealism: SWinGS++ for body reconstruction and HeadGaS++ for face reconstruction, and provide as well a solution to merge the separate face and body models without artifacts. We also present a demo of the complete system, showcasing several use cases of real-time conversation with the 3D avatar. Our approach offers a fully integrated virtual avatar experience, supporting both oral and written form interactions in immersive environments. ICo3D is applicable to a wide range of fields, including gaming, virtual assistance, and personalized education, among others.

Keywords: Neural rendering, facial animation, blendshapes, 3D reconstruction, virtual avatar, dynamic, LLM

1 Introduction

The development of interactive 3D virtual humans has gained significant traction in recent years, driven by advancements in artificial intelligence, computer vision, and real-time rendering technologies. These digital avatars are becoming integral to a wide range of applications, including gaming, entertainment, education, healthcare, and customer service. By enabling realistic, dynamic interactions between users and virtual characters, 3D virtual human avatars have the potential to redefine the way users experience virtual or augmented environments. Capturing, reconstructing, and reanimating an interactive virtual avatar that moves and behaves like a real human is a complex problem composed of a multiplicity of interconnected tasks, such as face reconstruction, speech and expression reanimation, body reconstruction and pose control, natural language understanding and generation. Prior art has generally focused on solving each of these problems individually, and often independently, in part as capturing multi-view sequences of body and head requires different hardware arrangements in terms of space, cameras and illumination.

Arguably, the face is the most attention-attracting element of a 3D virtual human avatar.

Fig. 1 Our method generates a photorealistic 3D full-body virtual avatar from posed multi-view images, with dynamic facial expressions driven by input audio and corresponding procedural body animation. A LLM enables the user to converse with the avatar in real-time. We refer the reader to our suppl. video to see our method in action.

A number of recent works [1–4] have obtained impressive results in photorealistic reconstruction of heads and faces, with increasingly faster rendering times using both monocular or multi-view input videos. Often, they retain a control interface for new expressions via parametric 3D Morphable Models (3DMM), such as FLAME [5] and ARKit [6]. Despite their success, these works are generally disconnected from the full body, sometimes even lacking neck articulation. Additionally, the expression reanimation often relies on pre-recorded face tracking or manually edited 3D blendshapes, making them not easily generative.

In this work, we build upon our previous work on head [7] and dynamic [8] 3D body reconstruction. We introduce technical improvements over both methods, and build a full-bodied conversational avatar that runs in real time.

Our contributions are: **(1)** we extend our head model HeadGaS to enable audio-driven head reanimation (Sec. 3.3), and **(2)** propose a novel intra-window spatio-temporal encoder that enhances temporal consistency and accuracy of our dynamic body model SWinGS (Sec. 5.1). **(3)** We then integrate both head and body reconstruction to produce a complete avatar (Sec. 3.5), and **(4)** introduce a Large Language Model (LLM) that provides interactive conversational capabilities (Sec. 3.7). Given the text response from the LLM, a Text-to-Speech (TTS) model generates speech audio that drives our head model, enabling realistic synchronized facial animations. Thanks to the inherent efficiency of 3D Gaussian Splatting at inference time, we can render the avatar in our real-time viewer, allowing users to chat with it while wearing a VR headset or watching a screen.

The remainder of this paper is organized as follows: In Sec. 2, we review related work in dynamic 3D human reconstruction, head reconstruction and animation, and Natural Language Processing (NLP). In Sec. 3, we describe the building blocks of our conversational virtual human, followed by Sec. 4 with further implementation details, specially relating to our real-time viewer. We present quantitative and qualitative results of our algorithms and system in Sec. 5. We discuss current limiation in Sec. 6 and conclude the paper in Sec. 7.

2 Related Work

2.1 Neural Rendering of Dynamic Human Bodies

Recent advancements in neural rendering have significantly improved the fidelity, photorealism, and efficiency of 3D human avatars compared to commonly used textured meshes. Pioneering works in human body modelling use statistical mesh templates such as SMPL [9, 10], which are convenient to use but lack flexibility to represent the exact body shape and appearance of a person, and thus are often used as building blocks of more advanced methods.

Since then, another important breakthrough has been the seminal work of Neural Radiance Fields (NeRF) [11] that enabled photorealistic rendering of static scenes via volume rendering. The computationally intensive first version has quickly been improved enabling real-time rendering by storing spatially distributed features [12, 13]. NeRF has also been extended to dynamic scenes by modeling the time dimension [14–16]. These dynamic radiance fields can be used to train and render volumetric representation of humans in motion [17, 18]. Alternatively, one can also use implicit surface representations [19, 20] to render a 3D human [21, 22].

Recently, 3D Gaussian Splatting (3DGS) [23] made significant strides by modeling scenes with 3D Gaussian primitives, which, when combined with tile-based differentiable rasterization, achieves very fast rendering while maintaining

high-quality reconstruction. Luiten et al. [24] extended this to dynamic scenes with tracked 3D Gaussian trajectories. Recent methods extend 3DGS to general dynamic scenes including [25–30] among others. Other works have focused specifically on adapting 3DGS to dynamic human reconstruction, including [31–39].

Regardless of the representation (e.g. mesh, NeRF, SDF, 3D Gaussians), one important design choice (explored in Sec. 5.1.2) is the driving signal for the body motion: learning time-dependent motion leads to 4D models that can replay the training videos [8, 17, 18], while using 3D body pose as input enables the creation of animatable body models [31–33, 35, 40] deformed with skinning algorithms [41, 42].

Our method specifically extends upon our prior work SWinGS [8] for multi-view 4D reconstruction. By incorporating recent advancements, we enable real-time and higher-fidelity full-body 3D reconstructions, providing a comprehensive solution for our interactive virtual human avatar.

2.2 Head Reconstruction and Facial Animation

Head reconstruction from a set of image observations has been a very active field in recent years. Approaches include models that generalize across subjects [43–45] or rely on multi-view head captures [46–49], which can have a static or dynamic form. The task of generating animatable facial models from images has progressed significantly, with recent works enabling high-fidelity facial reenactments and dynamic expressions. Deep Video Portraits [50] and Face2Face [51] enable real-time facial reenactment with high realism but rely on constrained source data, such as a frontal single-view image and limited expressions. Voice2Face [52] employs audio signals for facial animation, but the expressions and conversational integration are limited.

NeRF- and mesh-based approaches have also been developed for dynamic head modeling. NeRF-based methods such as Neural Head Avatars [53] and Semantic Facial NeRFs [54] reconstruct personalized radiance fields from monocular videos, capturing detailed geometry and appearance within a volumetric representation. Mesh-based techniques like Realistic One-shot Mesh-based Avatars [55] enable high-fidelity geometry and texture reconstruction from minimal input, supporting expressive and identity-preserving reenactment. Hybrid representations, such as the Mixture of Volumetric Primitives [49], combine structured primitives with neural rendering to balance visual realism and computational efficiency.

It is worth noting that most head reconstruction and facial animation models focus primarily on incomplete captures of the human body, i.e. with a field-of-view framing the head generally from the shoulders up (see Fig. 12 left). This achieves effectively higher resolution for the head and face regions, and the required capture systems (e.g. lightstage domes, multi-view camera rigs) are more accesible than their full-body equivalents. Designing full-body avatars that retain the same fidelity and control for the face is challenging, both from the data side, but also from the system design. Beyond the animatable body models discussed in Sec. 2.1, there are some notable examples in the prior art of methodologies achieving full-bodied avatars that have some degree of face expression control. The foundational work of Pavlakos et al. [10] SMPL-X is an extension of SMPL [9] that incorporates hands and head parametric control. More recently, ExAvatar [56] introduces 3D Gaussians in the surface of the SMPL-X mesh topology and thus inherits the same head control capabilities. Despite these works pioneering control over the face in full-body avatars, arguably this is fairly limited in terms of expressiveness, and can not reproduce nuances necessary for e.g. accurate speech, emotion. Concurrent to our work, TaoAvatar [57] proposes *SMPLX++*, an extension towards clothed avatars, and improves face expressiveness via learnable blend shapes for each Gaussian primitive.

Further recent 3D Gaussian Splatting-based approaches to facial animation modelling include [1, 2, 4, 58]. Our method extends our prior work HeadGaS [7] by integrating audio-driven facial animation with generative, LLM-based conversational capabilities. This enables our avatars to reason and generate language, and to exhibit contextually appropriate facial expressions and respond to user inputs with realistic movements in real time.

2.3 Natural Language Processing and Conversational Agents

Language is central to the way humans acquire, organize and communicate knowledge. Language modelling has been a long-standing field that aims to provide computational models that can understand and generate diverse human language [59, 60]. Language Models (LMs), such as the ubiquitous n-gram model [61], generally predict the likelihood of word sequences based on the surrounding context. Recently, Large Language Models (LLMs) have shown impressive performance in text understanding and generation. The term LLM refers to a family of models with a very large parameter count, which have been trained on an immense corpus of data, i.e. at web-scale. Notable examples of LLMs such as GPT-3 [62], InstructGPT [63], and GPT-4, [64] rely on the *Transformer* architecture [65], which, thanks to the self-attention mechanism, can flexibly capture context and long-range dependencies. In part due to their scale advantage, LLMs present strong generalization and unprecedented naturalness across different tasks and domains, specially those related to conversational capabilities, e.g. natural language understanding, reasoning, and generation. We refer the reader to Yupen et al. [66] for a more exhaustive literature review on LLMs.

In this work, we leverage publicly available LLMs to provide advanced conversational capabilities to our 3D human reconstruction, which effectively provides a 3D embodiment of the LLM. Qwen [67] provides an open-source LLM series consisting of various versions (1.5, 2, 2.5) and number of parameters (ranging from 0.5B to 110B) [68]. In our pipeline, we deployed Qwen2 which supports multiple languages, including English and Chinese, and whose model weights are available on Hugging Face and ModelScope.

3 Method

3.1 System Overview

Our complete system for building a full-body animatable and interactive 3D virtual avatar integrates separate reconstruction methods for head and body modeling, along with LLM integration for driving real-time facial animation. We divide the reconstruction process into three main components: the audio-driven animatable head model (Sec. 3.3), the dynamic body model (Sec. 3.4), and head-body integration (Sec. 3.5). The head model is first trained separately to obtain high-quality renderings and to learn person-specific facial expressions from the data. It is then fused with the body model during the integration process to generate the complete avatar. Each reconstruction method is built upon the 3D Gaussian Splatting framework.

3.2 3D Gaussian Splatting

The original 3D Gaussian Splatting (3DGS) method [23] models a static scene using a set of 3D Gaussians. Given a set of images and associated camera poses, 3DGS initializes its representation with a sparse point cloud, typically generated using the COLMAP structure-from-motion pipeline [69]. Each 3D Gaussian is represented by its center $\boldsymbol{\mu} \in \mathbb{R}^3$, covariance matrix $\mathbf{\Sigma} \in \mathbb{R}^{3\times3}$, opacity $o \in \mathbb{R}$, and color $c \in \mathbb{R}^{3(k+1)^2}$, where k denotes the degree of spherical harmonics used to model the color, capturing view-dependent appearance. The Gaussians are placed in world coordinates and evaluated at a point $\boldsymbol{x}$ using the following equation:

$$G(\boldsymbol{x}) = e^{-\frac{1}{2}\boldsymbol{x}^T\mathbf{\Sigma}^{-1}\boldsymbol{x}}. \quad (1)$$

The covariance matrix $\mathbf{\Sigma}$ is computed as $\mathbf{\Sigma} = \mathbf{RSS}^T\mathbf{R}^T$, where $\mathbf{S}$ and $\mathbf{R}$ are the scaling and rotation matrices, respectively. To represent the scene, 3DGS optimizes the parameters of the 3D Gaussians using differentiable rasterization and Gaussian Adaptive Density Control (ADC). The loss function used for training is defined as:

$$\mathcal{L} = (1-\lambda)\mathcal{L}_1(I_r, I_{gt}) + \lambda\mathcal{L}_{\text{SSIM}}(I_r, I_{gt}), \quad (2)$$

where I_r and I_{gt} denote the rendered image and ground-truth image, respectively. This framework is highly effective for static scenes but lacks the ability to handle dynamics, such as the nuanced facial expressions and body movements of our avatar, which we address in the subsequent sections.

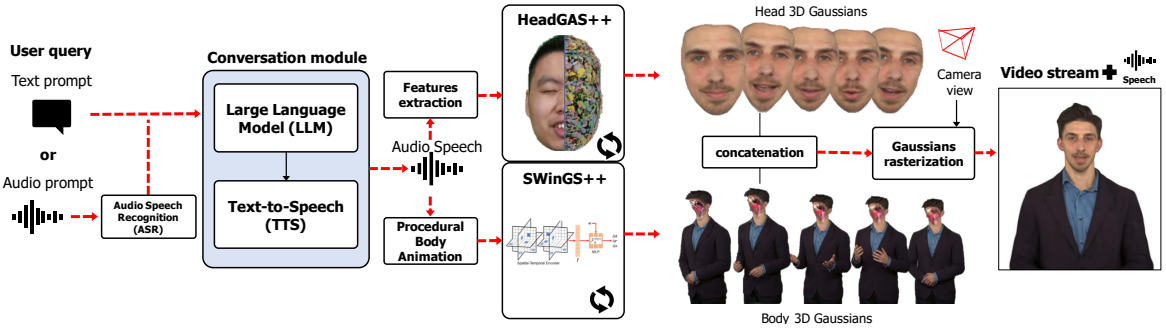


Fig. 2 Overview of our conversational human pipeline ICo3D. Users interact with the avatar via text or audio queries, which are processed by a LLM to produce a textual answer, then converted to audio. The audio speech is the driving signal for the audio-driven head model *HeadGaS++* (Fig. 3). It also serves to determine the body dynamics through procedural body animation. 3D Gaussians are generated at each timestep by both the head and the body model *SWinGS++* (Fig. 4), which are then integrated and rendered from novel views, producing a free viewpoint video stream synchronized with the audio speech.

3.3 HeadGaS++: Audio-driven 3D Head Model

In this section, we describe our method HeadGaS++, our approach to generating high-fidelity, audio-driven 3D head models from multi-view video sequences and speech inputs. The goal is to synthesize novel views and expressions of the avatar's face given a novel audio feature vector and camera pose as input. An overview of our method is shown in Fig. 3. The method is an extension of our prior work on animatable head modelling HeadGaS [7], but with enhanced audio-driven capabilities and higher fidelity reconstructions (see Table 2).

As in HeadGaS, HeadGaS++ extends the static 3DGS framework to create animatable avatars by introducing audio-dependent Gaussian properties. We enable each Gaussian to change its color c_i and opacity o_i based on the input audio-visual features. Rather than moving the Gaussian positions directly, this framework results in 3D Gaussians with dynamic appearances. These Gaussians can occasionally appear and vanish depending on the current audio-visual features, and additionally allow color changes to simulate non-rigid appearance effects.

To this end, we augment every Gaussian primitive with a learned latent feature basis $\mathbf{F} \in \mathbb{R}^{B\times f}$, as shown in Fig. 3 (Audiovisual Feature Basis), where B is the number of blendshape expressions. This feature basis is blended with an audio-visual feature vector to enable dynamic expression control. During training, the latent basis $\mathbf{F}$ is optimized together with the other parameters of the 3D Gaussians. At each iteration, we blend the respective audio-visual weights $\mathbf{e}_i \in \mathbb{R}^B$ corresponding to a particular expression at the current frame index i, with the feature basis $\mathbf{F}$ into a 1D feature vector $\mathbf{f}_i \in \mathbb{R}^f$ as:

$$\mathbf{f}_i = \mathbf{F}^T \mathbf{e}_i + \mathbf{f}_0, \tag{3}$$

where $\mathbf{f}_0$ is a bias term. Then, feature $\mathbf{f}_i$ is fed into a small MLP $\phi(\cdot)$, to compute the Gaussian's color c_i and opacity o_i:

$$c_i, o_i = \phi(\mathbf{f}_i, \gamma(\boldsymbol{\mu})), \tag{4}$$

where γ denotes the sinusoidal positional encoding function, and $\boldsymbol{\mu}$ denotes the Gaussian mean position.

We use a compact MLP composed of two linear layers with LeakyReLU activation; the hidden layer has 64 channels, and the last layer consists of two branches for color and opacity prediction. The model is optimized by comparing the rendered image I_r with the ground truth image I_{gt}, using the following loss function:

$$L = \lambda_1 L_1(I_r, I_{gt}) + \lambda_s L_s(I_r, I_{gt}) + \lambda_p L_p(I_r, I_{gt}), \tag{5}$$

where, λ_1, λ_s, and λ_p are weighting factors, and L_s and L_p are SSIM and perceptual losses [70], respectively.

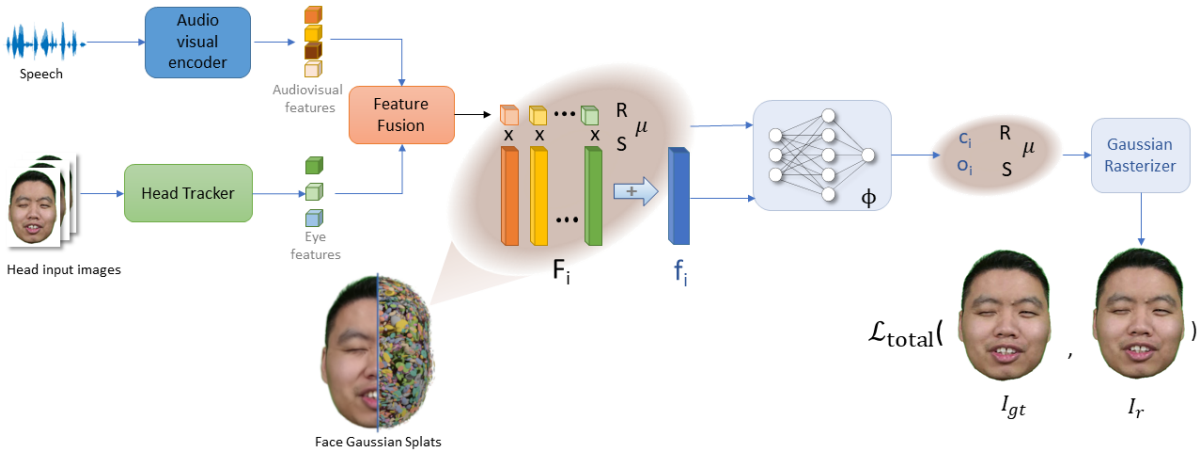


Fig. 3 An overview of our method for 3D animatable head reconstruction. *HeadGaS++*, is an extension of our work HeadGaS [7], which uses features extracted from input audio speech to drive the facial expressions.

3.4 SWinGS++: Dynamic 3D Body Reconstruction

Once the animatable head model is trained, our focus shifts to 4D body reconstruction to complete the avatar. To achieve the level of fidelity required to create a convincing and realistic virtual human, we extend our prior work SWinGS [8]. SWinGS enables high-quality rendering of general dynamic scenes given multi-view, calibrated and time-synchronized videos as input. SWinGS++ extends the method by shifting the focus to dynamic humans with the introduction of a spatial-temporal encoder to provide better human motion estimation. An overview of our extended method is shown in Fig. 4.

The body reconstruction pipeline comprises three main components: 1) We divide an input sequence into overlapping variable-length sliding windows, based on the human's motion. This allows us to handle long sequences while preserving render quality. 2) We train independent 4DGS models for each sliding window, where per-window tunable MLPs [71] learn deformations from temporally-local canonical Gaussians to each frame. This allows us to capture significant geometric changes as well as the emergence of new objects. 3) A final temporal fine-tuning stage enforces temporal consistency throughout the sequence. We discuss each of these steps in the following sections.

3.4.1 Sliding Window Sampling

Representing the dynamic human with a single 4DGS model is impractical, in terms of computational load and performance degradation, especially with fast human motion and long sequences. To address this, following [8], we introduce a sliding-window strategy; partitioning the sequence into smaller overlapping windows. Each window is an independent 4DGS model, allowing all Gaussian parameters–positions, rotations, scaling, colors, and opacities–to change between windows. Our adaptive sampling method dynamically adjusts the window size based on the human's motion, balancing training efficiency and performance. Given a sequence of length N_f frames, we split it into a set of sliding windows of lengths $\{N_w\}$, depending on the motion: high-motion regions are sampled more frequently (shorter windows), low-motion regions are sampled less frequently (longer windows). The motion is estimated using 2D optical flow from a pre-trained RAFT model [72]. For each camera view $c \in C$, we accumulate the optical flow magnitude $\hat{v}_i$ across frames:

$$\hat{v}_i = \frac{1}{C} \sum_{c}^{C} \sum_{i}^{N_f - 1} ||\boldsymbol{v}(I_i^c, I_{i+1}^c)||_2^2, \tag{6}$$

and a new window is spawned at frame i when the accumulated flow exceeds a predefined threshold $\hat{v}_i > v_{thresh}$.

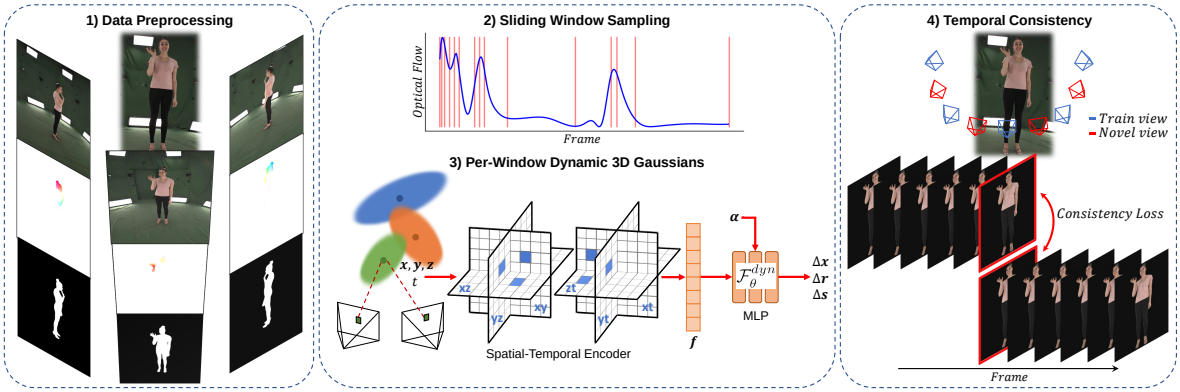


Fig. 4 An overview our dynamic 3DGS method for body reconstruction.. *SWinGS++* is an extension of our work SWinGS [8], using temporally-local dynamic MLPs on a sliding window basis. The method is extended using a spatial-temporal encoder to help reconstruct larger or faster human movements.

3.4.2 Dynamic 3D Body Gaussians

To extend 3DGS for dynamic scenes, [8] introduced temporally-local dynamic MLPs $\phi(\cdot)$ for each sliding window, where each MLP learns the deformation mapping from a per-window canonical 3D Gaussian representation to the set of 3D Gaussians for each frame i within the window. We use a compact MLP which takes as input the normalized frame time $t \in [0, 1]$ and 3D Gaussian positions $\boldsymbol{\mu}$ (normalized by the scene's mean and standard deviation), and outputs displacements to their positions $\Delta\boldsymbol{\mu}$, rotations $\Delta\boldsymbol{r}$ and scaling $\Delta\boldsymbol{s}$ at each time step:

$$\Delta\boldsymbol{\mu}(t), \Delta\boldsymbol{r}(t), \Delta\boldsymbol{s}(t) = \phi(\gamma(\boldsymbol{\mu}), \gamma(t)). \quad (7)$$

However, we found this model insufficient to capture human motion when the inter-frame displacement is large. Therefore, SWinGS++ extends the method by introducing per-window spatial-temporal encoders, similar to [37, 73], as shown in Fig. 4. The input 3D position $\boldsymbol{\mu}$ and time t are passed through the spatial-temporal encoder $\mathcal{H}$ producing voxel features $\mathbf{f}_v = \mathcal{H}(\boldsymbol{\mu}, t)$, which are then passed to the MLP:

$$\Delta\boldsymbol{\mu}(t), \Delta\boldsymbol{r}(t), \Delta\boldsymbol{s}(t) = \phi(\mathbf{f}_v), \quad (8)$$

resulting in better motion estimation and reconstruction, as shown in Table 1 and our qualitative results in Fig. 8.

The MLP uses the same architecture as [8], i.e. a *tunable MLP* [71] consisting of M sets of weights with blending parameters $\{\boldsymbol{\alpha} \in \mathbb{R}^{M \times N_g}\}$. These sets of weights enable the MLP to learn M *motion modes*, smoothly weighted by blending parameters $\boldsymbol{\alpha}$ for each input Gaussian. The output of a single layer of the dynamic MLP can be expressed as:

$$\boldsymbol{y} = \psi\left(\sum_{m=1}^{M} \left(\alpha_m \boldsymbol{w}_m^T \boldsymbol{x} + \alpha_m \boldsymbol{b}_m\right)\right), \quad (9)$$

where $\boldsymbol{w}$ and $\boldsymbol{b}$ are the weights and bias, and ψ is an activation function. Thus, the MLP from Eq. (8) becomes ϕ_{dyn}, a function of spatial-temporal features $\mathbf{f}_v$ and blending parameters $\boldsymbol{\alpha}$:

$$\Delta\boldsymbol{\mu}(t), \Delta\boldsymbol{r}(t), \Delta\boldsymbol{s}(t) = \phi_{dyn}(\mathbf{f}_v, \boldsymbol{\alpha}). \quad (10)$$

We optimize the model using the following loss function:

$$L = \lambda_1 L_1(I_r, I_{gt}) + \lambda_s L_s(I_r, I_{gt}) + \lambda_m L_m(M_r, M_{gt}), \quad (11)$$

where L_m is a mask loss which is simply the L_1 loss on the rendered M_r and ground truth M_{gt} human masks.

3.4.3 Temporal Consistency Fine-tuning

After training each sliding window independently, optimization discrepancies lead to temporal artifacts on the body, such as flickering, mainly apparent in novel views. To mitigate this, we employ

temporal fine-tuning to enforce inter-window consistency. We fine-tune each sliding window model, starting from the first and proceeding sequentially through the sequence. We randomly sample novel views $\mathbf{P}_{novel}$ by interpolating the training camera poses in $\mathbf{SE}(\mathbf{3})$:

$$\mathbf{P}_{novel} = \exp_{\mathrm{M}} \sum_{c}^{C} \beta_c \log_{\mathrm{M}}(\mathbf{P}_c), \tag{12}$$

where $\exp_{\mathrm{M}}$ and $\log_{\mathrm{M}}$ are the matrix exponential and logarithm respectively, and $\beta_c \in [0, 1]$ is a uniformly sampled weighting. We apply consistency loss on novel view renders, which is simply the $L1$ loss on the overlapping frames of the current w and previous $w-1$ windows:

$$L_{consistency} = |I^{w}_{r,t=0} - I^{w-1}_{r,t=N_w-1}|_1. \tag{13}$$

During fine-tuning, we freeze the previous window's model and the MLPs, allowing only the canonical Gaussians to optimize. We use an alternating training strategy which optimizes for temporal consistency on overlapping frames for 75% of the time, while using the remaining 25% for standard training.

3.5 Head-Body Model Integration

Once the head model is trained (Sec. 3.3), the next step is to integrate the head with the body to produce a full-bodied avatar that can be rendered in real-time. The integration process happens jointly with the optimization of the body (Sec. 3.4) as this enables smooth unification of the two models. Our integration method includes the following steps: 1) 3D head-body pose alignment (see Sec. 3.6 on cross-setup alignment), 2) Gaussian merging, 3) Gaussian pruning and blending, and lastly 4) face color optimization (if necessary). In this section, we describe each of these processes in detail.

First, we align the head model with the body model for each frame in the sequence. If the body and the face are visible in the same capture (as in Fig. 10), the head and body models can be simply aligned using the tracked head poses, as the head and body segmentations are extracted from the same input images and both models use the same camera poses. In this case, the head Gaussians are transformed by the corresponding head pose $\mathbf{H}_i \in \mathbb{R}^{4\times4}$ at each frame i. In the case of head and body data from different sources, i.e. different capture setups (Fig. 7), the head must be rigidly aligned to the body with additional transformation matrices. This is explained in Sec. 3.6.

Secondly, to render the head and body models together, we merge their respective Gaussian representations into a unified representation by concatenating each of the Gaussian parameters together, before being fed into the rasterizer. This includes their positions, rotations, scales, colors and opacities. Thus, the complete avatar can be rendered efficiently with a single pass of the Gaussian Splatting rasterizer.

Thirdly, simply merging the Gaussians of each model together as stated results in significant artifacts. The reason is that Gaussians on the body model can penetrate the face model. Additionally, 2D head segmentation errors (or multi-view mask inconsistencies) can lead to artifacts at the face boundary, creating a noticeable *transition* between face and body. To address these two issues, we apply Gaussian blending and pruning strategies. To prevent body Gaussians penetrating the face, we prune Gaussians from the body, every $n = 100$ iterations, that lie within a spherical region positioned at the centroid of the face. We also prune Gaussians around the jaw to prevent "double-chin" artifacts appearing, e.g. when the face model opens the mouth but on the body it is closed. To address face border issues, at the start of training we inject randomly sampled Gaussians positioned around the border (extracted from the FLAME mesh) to help blend the edges.

The final integration stage is color or appearance optimization, i.e. we optimize for the colors of the face Gaussians to match the body. For captures with visible head and body, this step is not necessary as the head and body models are trained using the same input images and any differences in color due to optimization discrepancies are negligible. In the case of data from different sources with different lighting conditions, the appearance of the head must be adjusted to match the body. We do this by unfreezing the final layer of the HeadGaS++ model, i.e. the parameters that control the color of the Gaussians, and allow these to optimize concurrently with training of the body.

3.6 Cross-setup Head Alignment

When training with the RenderMe-360 [74] and DNA-Rendering [75] datasets for head and body reconstruction respectively, we face the challenge of having two separate captures of our avatar, acquired at different times with distinct capture setups (Fig. 7). Each setup has different world origins, camera poses and intrinsics, different motion sequences with changing head poses, facial expressions, and different lighting conditions, and other varying factors (e.g. appearance and clothing) that make integrating head and body challenging.

For the task of cross-setup head alignment, the aim is to combine the 4DGS models so that the head can be rendered in the reference system of the body. In other words, transform the HeadGaS++ model trained with capture setup 1's head and camera poses (RenderMe-360), so it can be observed in capture 2's reference system (DNA-Rendering) and aligned to the per-frame head poses of capture 2. Generally, this would not have a closed-form solution, given that the captures are entirely independent. However, we can exploit the fact that the FLAME fittings from both sequences puts both heads (from capture setups 1 and 2) into a unified canonical reference system.

First, we can project a 3D point $\boldsymbol{x}$ from each of the 3D models into an image in their individual reference systems as:

$$\boldsymbol{p}_{w_1} = \mathbf{K}_1 \mathbf{C}_{w_1}^{w2c} \mathbf{C}_{ref,w_1}^{-1} \mathbf{H}_{i,w_1} \boldsymbol{x}_c \tag{14}$$

$$\boldsymbol{p}_{w_2} = \mathbf{K}_2 \mathbf{C}_{w_2}^{w2c} \mathbf{C}_{ref,w_2}^{-1} \mathbf{H}_{i,w_2} \boldsymbol{x}_c, \tag{15}$$

where $\boldsymbol{p}$ is a 2D image point and $\boldsymbol{x}_c$ is the corresponding 3D point in canonical FLAME coordinates. $\mathbf{H}_i$ is the head pose, $\mathbf{K}$ and $\mathbf{C}^{w2c}$ are camera intrinsics and extrinsics, and $\mathbf{C}_{ref}$ is the head tracker's reference world origin. Subscripts w_1 and w_2 refer to the world coordinate systems of capture setups 1 and 2 respectively.

However, what we need to represent in world reference system w_2 is not the canonical head $\boldsymbol{x}_c$, but the observed head $\boldsymbol{x}_i$ at frame i. The relation between canonical pose and real observation is:

$$\boldsymbol{x}_c = \mathbf{H}_{ref}^{-1} \mathbf{H}_i^{-1} \mathbf{C}_{ref} \boldsymbol{x}_i. \tag{16}$$

We write this equation generally as it applies to both world references. Note, head poses are applied relative to the first frame ($\mathbf{H}_{ref} = \mathbf{H}_0$).

Fig. 5 Cross-setup head-body integration. The audio-driven head model is trained using HeadGaS++ on the RenderMe-360 dataset [74], while the body is trained on the same subject from the DNA-Rendering dataset [75]. Left: the RenderMe-360 head is aligned to the DNA-Rendering body using Eq. 20. To reduce artifacts, we prune Gaussians from the body model as discussed in Sec. 3.5. Right: we optimize for the face colors by un-freezing the last layer of the HeadGaS++ MLP and optimizing jointly with the body model.

We can also write the following relations between the learned head geometry $\boldsymbol{x}_h$, canonical points $\boldsymbol{x}_c$ and observed points $\boldsymbol{x}_i$:

$$\boldsymbol{x}_h = \mathbf{C}_{ref}^{-1} \mathbf{H}_i^{-1} \mathbf{C}_{ref} \boldsymbol{x}_i, \tag{17}$$

and

$$\boldsymbol{x}_c = \mathbf{H}_{ref}^{-1} \mathbf{C}_{ref} \boldsymbol{x}_h. \tag{18}$$

Exploiting the fact that canonical points $\boldsymbol{x}_c$ are are same in both worlds, and referring to Eq. (16) and 18, we can write:

$$\mathbf{H}_{ref,w_1}^{-1} \mathbf{C}_{ref,w_1} \boldsymbol{x}_h^{w_1} = \mathbf{H}_{ref,w_2}^{-1} \mathbf{H}_{i,w_2}^{-1} \mathbf{C}_{ref,w_2} \boldsymbol{x}_i^{w_2}. \tag{19}$$

This can be rearranged, to obtain the final transformation:

$$\boldsymbol{x}_i^{w_2} = \mathbf{C}_{ref,w_2}^{-1} \mathbf{H}_{i,w_2} \mathbf{H}_{ref,w_2} \mathbf{H}_{ref,w_1}^{-1} \mathbf{C}_{ref,w_1} \boldsymbol{x}_h^{w_1}. \tag{20}$$

Using this transformation we can transform the learned HeadGaS++ geometry $\boldsymbol{x}_h^{w_1}$ trained with the head-centric captures from capture 1, into the observed full-body captures $\boldsymbol{x}_i^{w_2}$ of capture 2. An example of this alignment is shown in Fig. 5.

3.7 Language Understanding and Generation

To implement our interactive pipeline, we capitalize on the publicly available tools for natural language processing, namely Audio Speech Recognition (ASR), Large Language Models (LLM) and Text-to-Speech (TTS) models. We will briefly describe the specific models we used in our framework.

3.7.1 Audio Speech Recognition (ASR)

ASR is the part of the pipeline responsible for understanding input human audio speech and feeding the LLM with the correct question with low latency. We select Whisper [76], a state-of-the-art model which demonstrates a strong ability to generalize to many datasets and domains in a zero-shot setting. In our pipeline, we incorporate the Whisper Large V3 model to have support for multiple languages and its distilled version, which is much faster but with support only for English.

3.7.2 Large Language Model (LLM)

One of the most important parts of our pipeline is the LLM module. As latency is a crucial factor for the real-time nature of our application, we need to have a fast and accurate model. To this end, we deployed the quantized model of Qwen2 with 0.5B parameters and Activation-aware Weight Quantization (AWQ) [77]. It is based on the Transformer architecture with SwiGLU activation, attention QKV bias, and group query attention.

3.7.3 Text-to-Speech (TTS)

TTS consists of the last part of our pipeline. For our framework, we deployed the OpenVoice V2 [78], which has native multilingual support (English, Spanish, French, Chinese, Japanese, and Korean).

4 Implementation Details

We implement our method in PyTorch, building upon the 3DGS codebase and its differentiable rasterizer. We initialize the body Gaussian centers using point clouds derived from SMPL-X fittings, while the head points are initialized with 2500 vertices derived from the tracked FLAME mesh [3, 79]. In scenarios where no mesh data is available, the points are sampled randomly within predefined near and far bounds.

The learning rates for the head MLP, Gaussian positions $\boldsymbol{\mu}$, latent features $\mathbf{F}$, scale $\mathbf{S}$, and rotation $\mathbf{R}$ are set to 1.6×10^{-4}, 1.6×10^{-4}, 0.0025, 0.005, and 0.001, respectively, while the body MLP and blending parameters $\boldsymbol{\alpha}$ learning rates are initialized to 1×10^{-4}. The body is trained with the Adam optimizer, while head optimization is performed using Stochastic Gradient Descent (SGD), with exponential decay scheduling applied to the Gaussian centers $\boldsymbol{\mu}$ and the MLP parameters. The latent feature $\mathbf{F}$ dimensionality is fixed at 32 and initialized to zero. The expression feature vector consists of 32-dimensional audio features from SyncTalk [80] and 7-dimensional eye parameters from the ARKit tracking [6], resulting in 39 elements in total (i.e., $B = 39$).

The head model is trained on a GPU with 14 TFLOPS of FP32 performance for 50,000 iterations, a process that takes roughly one hour to complete. The perceptual loss L_p is applied after 10,000 iterations with a weight of 0.1, while the weights for L_1 and SSIM are set to 0.8 and 0.2, respectively. For the body, each sliding window model is trained for 30,000 iterations in parallel across eight GPUs with 14 TFLOPS (FP32). The first 1,000 iterations are designated as a warm-up phase, where only the central frame of each window is trained, with the MLP weights frozen to allow stabilization of the canonical body representation. The final fine-tuning step is performed sequentially on a single GPU for 20,000 iterations per window to ensure temporal consistency. For both models, adaptive densification begins at iteration 500 and concludes by the 15,000th iteration.

4.1 Procedural Body Animation

While the dynamic body model produced by SWinGS++ enables replay of the training video sequence, it does not support novel pose generation. To create novel sequences, we use a simple procedural animation technique. At inference time, given speech audio input, our method generates a plausible body-pose sequence to complement the audio-driven facial expressions. The body animation should match the length of the

speech. Otherwise, the head and body movements can appear disjointed.

We do this by defining two states for the avatar: (i) rest pose and (ii) action, each defined by a set of preselected keyframes from the training video. In rest pose, the avatar is in an idle state, waiting to converse with the user when provided input. The rest pose follows an infinite loop, constantly replaying until a user input is detected. The rest pose loop is created by defining start and end keyframes and looping the rendered frames back and forth. Random eye blinks are inserted so that the avatar does not feel lifeless. To create an action sequence, we define a set of keyframes that signify key body movements from the training video, e.g. hands gesticulating, carefully selected so that the reversed the motion still feels natural. Our animation method randomly selects from these keyframes to generate a sequence of movements that match the length of the input audio, finishing at the rest pose end keyframe so that the movement smoothly transitions back into the idle state.

4.2 Interactive Viewer

In this section, we detail our interactive viewer implementation, serving as the central platform for integrating our prebuilt avatar models with interactive Text-to-Speech (TTS) and Large Language Model (LLM) functionalities. The system combines these components to deliver a smooth and immersive conversational human avatar experience. We build upon the original viewer from 3DGS [23], which excels in rendering high-fidelity static scenes but lacks dynamic and interactive capabilities. To bridge this gap, we extend it to support real-time interactivity and conversation by adding modules for: (1) text or microphone input, (2) Automatic Speech Recognition (ASR), (3) LLM processing, (4) TTS audio generation, (5) audio-to-SyncTalk expression parameter generation, and (6) our HeadGaS++ and dynamic body models.

The key features of our interactive viewer include:

4.2.1 Modular Integration with a Remote Server

The viewer adopts a modular architecture allowing for integration with remote server modules, including the ASR, LLM, TTS, and SyncTalk components. Each component functions independently while maintaining synchronized communication between viewer and server. By keeping the ASR, LLM, TTS, and SyncTalk modules on a remote server, we minimize the computational load on the local viewer. The local viewer focuses solely on rendering and minimal communication, including:

- Sending: user inputs (audio from a microphone or text).
- Receiving: processed outputs (ASR text, LLM text responses, TTS audio, SyncTalk parameters).
- HeadGaS++ inference: using SyncTalk and head pose inputs to predict Gaussian attributes (Sec. 3.3).
- Rendering: background, body, and face Gaussians.

This setup minimizes communication between the local viewer and the remote server to essential data exchanges, while computationally intensive processes run concurrently on the remote server. The final output from the server is the SyncTalk expression parameters, which the local viewer feeds into the embedded HeadGaS++ module (using LibTorch). Frame-wise Gaussian parameters are generated to produce audio-synchronized facial expressions.

4.2.2 Real-Time Synchronization

The viewer synchronizes facial expressions, lip movements, and body gestures with generated speech in real time. Given a user input, the TTS module converts a LLM-generated text response into an audio signal, driving the facial animation. Our procedural animation module (Sec. 4.1) generates natural idle motion to ensure the avatar displays natural body movement (e.g. eye-blinking) while awaiting user input or server response. To smoothly transition between idle and talking states, audio playback and facial animation start simultaneously. The avatar's Gaussians are rendered at fixed 30 FPS to synchronize with the audio, while background Gaussians utilize the remaining processing capacity, enabling our system to run at **105.27 FPS**.

5 Experimental Results

This section presents visual and quantitative results supporting our method. We recommend that readers inspect the supplementary video, which shows high-quality renderings and several user interaction scenarios with our avatar. The model presented in this video was captured using a synchronized multi-view system with 24 cameras, recording a subject performing speech and gestures simultaneously. To our knowledge, no public dataset exists containing similar data (both speech and body motion). As a result, we evaluate our body and face models separately using dedicated datasets in Sec. 5.1 and 5.2. As these datasets have been released by the same group, we select sequences that depict the same subject in both datasets, enabling us to build an interactive avatar from different captures. This procedure is described in Sec. 3.6.

5.1 Body Model Evaluation

We use the public DNA-Rendering dataset [75] to evaluate our proposed body model. This dataset contains full body human performances recorded by 60 synchronized cameras at 15 FPS in a light stage environment. Despite dense camera coverage, this is a challenging dataset due to complex human motion, loose clothing with rich textures, a large diversity of scanned individuals (ethnicity, age, height, gender, etc.), and various object interactions. An overview of the camera positions and images is given in Fig. 7.

5.1.1 Novel View Synthesis Evaluation

Following prior work, we evaluate the quality of our model through novel view synthesis (NVS). DNA-Rendering [75] includes an NVS benchmark but does not provide the exact data splits to enable direct comparison to their results. However, we replicate it as closely as possible while comparing to the most recent baselines.

We train one model for each sequence of 225 time steps using 48 cameras evenly distributed around the subject. 12 cameras are held out for evaluation. We use the six sequences shown in the benchmark website[1]. To save computation, we preprocess training images by cropping based on segmentation masks to avoid rendering background pixels. Evaluation is done using PSNR, SSIM [81], and LPIPS [82] metrics. PSNR-M refers to masked PSNR, i.e. PSNR computed only on the ground truth human segmentation mask. We compare our body model SWinGS++ (Sec. 3.4) to the following baselines:

- SWinGS [8]: our prior method without spatial-temporal feature encoders.
- 4D Gaussian Splatting [37]: a 4D model that proposed the spatial-temporal feature encoder for 4DGS. In contrast with our method, it does not use sliding windows.
- SpaceTime Gaussians [83]: a 4D model using time-conditioned parametric functions to model the evolution of Gaussians.
- HuGS [31]: An animatable GS model, i.e. deformations are driven by the SMPL-X [10] skeleton (Sec. 5.1.2).
- Better Together [84]: Another animatable Human Body Model, that optimizes jointly skeletal motion with gaussians rigged on a personalized human mesh, resulting in improved quality compared to HuGS.

Quantitative comparison with all other methods is shown in Table 1 and qualitative comparisons are provided in Fig. 8. We observe that fitting this dataset is difficult for most methods. Spacetime Gaussians [83] is limited to short sequences as it is memory demanding; we had to split the video into 5 segments of 45 frames and train a dedicated model for each. 4DGaussians [37] is compatible with longer sequences but presents quite low rendering quality with poor detail reconstruction. HuGS [31] supports training on long sequences but, as it relies on a template-body model, it is not able to capture objects and loose clothing. It is generally more blurry overall, due to noise in the input body pose parameters. Better Together [84] presents improved rendering results compared to HuGS thanks to optimized body poses but still performs worse than 4D methods, especially on loose clothing sequences. SWinGS [8] and SWinGS++ perform better than these baselines thanks to the sliding windows approach.

[1] https://dna-rendering.github.io/inner-benchmark-novel-view.html, accessed 23/10/2025.

Table 1 Quantitative results of the body reconstruction for novel view synthesis on the DNA Rendering dataset [75]. We report PSNR, masked PSNR (PSNR-M), SSIM and LPIPS metrics.

Method	PSNR ↑	PSNR-M ↑	SSIM ↑	LPIPS ↓
SWinGS++ (ours)	**30.1701**	**23.6950**	**0.9699**	**0.0807**
SWinGS [8]	29.4488	22.9739	0.9647	0.0902
SpacetimeGauss.[83]	29.3039	22.8494	0.9621	0.0924
4DGS [37]	26.8197	20.2145	0.9441	0.1152
Better Together [84]	25.9881	19.5538	0.9437	0.1089
HuGS [31]	25.0400	18.9540	0.9404	0.1118

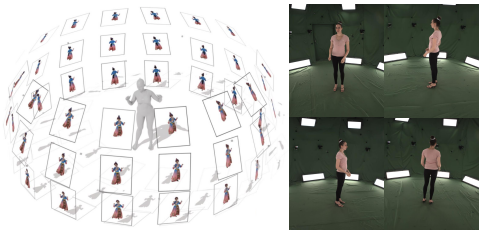
(a) SWinGS++ (b) HuGS (c) Novel pose

Fig. 6 Comparison between our dynamic body model SWinGS++ and HuGS. SWinGS++ provides notably better novel view synthesis results than HuGS on observed motion. However, HuGS also enables novel pose synthesis (c) which is not possible with a 4D reconstruction model like SWinGS++.

Finally, the spatial-temporal feature encoder of SWinGS++ improves the rendering quality further by 0.7dB PSNR and better reconstructs faster movements, e.g. the subject's leg in Fig. 8 row 1.

5.1.2 Comparison with Animatable Body Models

As presented in Sec. 3.4, SWinGS++ learns a time-dependent Gaussian Splatting model for the dynamic human. However, another approach is to use an animatable model where 3D Gaussians are skinned to a human skeleton [32, 34, 35, 38, 39].



(a) DNA-Rendering

(b) RenderMe-360

Fig. 7 Datasets used for cross-setup avatar creation. We can use data of the same subject captured from two different environments to train our Gaussian avatar. We used DNA-Rendering for the body model and RenderMe-360 for the head model. The alignment procedure is discussed in section 3.6.

The advantage is the ability to render the subject in any target pose, rather than being limited to replay of the training motions. We explored this using HuGS [31] and Better Together [84] for the body of our avatar. However, we observed that animatable models present reduced render quality compared to SWinGS++, although they are able to extrapolate to novel gestures with reasonable quality, as shown in Fig. 6. While the render quality is reduced compared to 4D methods, we note that this benchmark contains human-object interactions (e.g. holding a sword) which cannot be represented by a human body model only.

Given the difference in render quality, we decided to build our interactive avatar using SWinGS++ to maximize photorealism and generate a convincing avatar. However, 4D models limit us to the procedural body animation described in Sec. 4.1 and thus the motion of the avatar can sometimes look repetitive or not well adapted to the speech. Animatable Gaussian-based avatars would enable driving the body based on speech input using 3D motion synthesis methods [85, 86]. We see this direction as a promising future work.

Fig. 8 Qualitative comparison of novel view synthesis. Our method SWinGS++ displays notably sharper rendering with better motion reconstruction.

5.2 Head Model Evaluation

To evaluate our proposed animatable head model, we follow [87], using four high-definition talking portrait video clips from previous publicly released video sets (three males, one female) for testing. Each portrait video consists of 6,500 frames on average that are divided into training and test sets in a ratio of 10 : 1. The video clips are cropped and resized to 512×512, apart from the Obama video, which has a resolution of 450×450.

We compare our model with two state-of-the-art talking head NeRF-based methods, namely ER-NeRF [88], RAD-NeRF [89], and two 3D Gaussian Splatting-based methods, TalkingGaussian [87] and GaussianTalker [90]. The evaluation settings for the reconstruction quality and lip-audio synchronization ability are as follows:

1. *The self-reconstruction setting:* Each video is split into training and test sets. We trained each model using the audio, expression parameters, and head-pose sequences from the training set. For testing, the model synthesizes new instances of talking face using input audio, expressions, and pose sequences from the unseen test set. PSNR, SSIM [81], LPIPS [82], LMD [91] are used in quantitative comparison. Lip synchronization is evaluated using the confidence score (Sync-C) and error distance (Sync-E) from SyncNet [92, 93].

   Table 2 shows the quantitative comparison with all other methods and qualitative comparisons are provided in Fig. 9. Our method outperforms the state-of-the art methods in rendering quality, motion quality, and efficiency. It achieves the highest PSNR and SSIM, indicating superior image fidelity and structural similarity, while maintaining a competitive LPIPS score. In motion quality, it leads with the highest Sync-C, reflecting better synchronization. Efficiency-wise, it achieves the highest FPS (250) with a reasonable training time of 1 hour, making it highly suitable for real-time applications. The quality of the reconstructed faces can be seen in the qualitative results in Fig. 9.
2. *The cross-driven setting:* In this case, each pre-trained model is driven by audio tracks from other videos. Specifically, we used the same audio samples as in the previous work SynObama [94] to evaluate the "Macron" and "Lieu" portraits. Table 3 highlights the effectiveness of our method. It achieves high scores in Sync-C even in cross-lingual scenarios. Despite the Macron model being trained in French, it scores the lowest Sync-C by a considerable margin, indicating superior alignment of our lip movements. This emphasizes our proposed method's robustness and adaptability in handling cross-lingual lip synchronization, outperforming the other methods.

5.3 Head-Body Integration Ablation

In this section, we present an ablation study of our head-body integration framework, including quantitative and qualitative evaluation of each component described in Section 3.5. Our aim is to isolate each processing stage's contribution to the overall rendering fidelity of the fully-integrated avatar. To do so, we evaluate the image renders of each integrated avatar across all test views and every 10 frames of the input sequence, comparing them to the ground-truth images under the following cumulative configurations:

1. **Baseline (No Head):** Body-only reconstruction without the head.
2. **+ Head Reconstruction:** Directly attaching the independently reconstructed head.
3. **+ Gaussian Pruning:** Removing overlapping or redundant facial splats to minimize visual artifacts.
4. **+ Head-Pose Alignment:** Applying the head alignment transformation (Eq. 20) based on a shared FLAME mesh to better align head and body geometry.
5. **+ Face Color Optimization:** Harmonizing appearance in the overlapping face region via local color attribute optimization.

We evaluate the integration process for two separate capture scenarios: *same-setup* and *cross-setup*. *Same-setup* refers to when head and body are sourced from the same capture setup, i.e. the input images to the head and body reconstructions are extracted from the same input multi-view video sequences. For this we use the DNA-Rendering dataset which contains full-body captures. *Cross-setup* refers to when the head and body images are sourced from distinct capture setups, e.g. DNA-Rendering for the body and

Fig. 9 Qualitative comparison of lip synchronization. Our method HeadGaS++ displays better performance in synthesizing synchronized talking heads compared to the baseline methods. Please zoom in for better visualization.

Table 2 Quantitative results on head reconstruction quality and lip-synchronization for the *self-reconstruction setting*. We compare to different methods and report rendering quality, motion quality, and efficiency. Each model is trained using the audio, expression parameters, and head-pose sequences from the training set and tested on unseen input audio, expressions, and poses.

Methods	**Rendering Quality**			**Motion Quality**		**Efficiency**	
	PSNR ↑	SSIM ↑	LPIPS ↓	LMD ↓	Sync-C ↑	Training Time	FPS
Ground Truth	N/A	0	1.000	6.861	7.584	-	-
RAD-NeRF [89]	26.794	0.901	0.083	2.907	4.988	3h	25
ER-NeRF [88]	27.350	0.904	0.063	2.836	5.172	1h	35
TalkingGaussian [87]	29.317	0.920	**0.046**	**2.688**	5.802	0.8h	110
GaussianTalker [90]	29.127	0.911	0.085	2.814	5.350	5h	130
Ours	**30.398**	**0.935**	0.051	2.793	**5.928**	1h	250

Table 3 Quantitative results on lip synchronization for the *cross-driven setting*. Following previous works, we extract two audio clips from the SynObama demo [94] to drive each method and evaluate lip synchronization using the confidence score (Sync-C) and error distance (Sync-E) from SyncNet [92, 93].

Methods	Test Audio A				Test Audio B			
	"Macron"		"Lieu"		"Macron"		"Lieu"	
	Sync-E ↓	Sync-C ↑	Sync-E ↓	Sync-C ↑	Sync-E ↓	Sync-C ↑	Sync-E ↓	Sync-C ↑
Ground Truth	0	7.463	0	7.463	0	7.37	0	7.37
RAD-NeRF [89]	7.999	**6.419**	9.910	4.051	7.875	6.894	8.728	6.135
ER-NeRF [88]	8.618	6.110	11.105	2.828	7.826	**7.271**	11.241	3.168
TalkingGaussian [87]	8.597	6.038	10.389	3.242	8.008	6.803	10.429	3.736
GaussianTalker [90]	9.381	5.399	10.691	3.138	8.738	6.522	11.142	3.285
Ours	**7.566**	6.235	**7.431**	**6.487**	**7.231**	7.251	**7.463**	**7.491**

RenderMe-360 for the head. Each capture setup has different number of cameras with different viewpoints and fields-of-view, varying illumination and backgrounds, and changing avatar appearance. We explore the cross-setup scenario as it enables us to train a high-fidelity animatable head model on close-up images of the face, while capturing the body separately. High-resolution capture of both face and body with detectable facial expressions and audio speech is challenging to achieve with a single capture setup.

Evaluation Protocol and Selection Criteria. We perform the ablation study using 5 avatars with varying appearances: 3 for same-setup and 2 for cross-setup. We select avatars from the DNA-Rendering dataset who exhibit a presenter-style appearance, with visible facial expressions or speech and limited body motion. For cross-setup, we select avatars who are present in both the DNA-Rendering and RenderMe-360 datasets. For each setup, we follow the evaluation protocol of DNA-Rendering by training the body model with 48 cameras, with 12 hold-out cameras for testing. 8 training cameras are used to train the HeadGaS model. For cross-setup, we follow the same process to train the body, but the HeadGaS model is trained using 20 cameras from RenderMe-360.

Ablation Results and Discussion. Quantitative results for the same-setup and cross-setup scenarios are shown in Tables 4 and 5, respectively. Each additional component yields consistent improvements in PSNR, SSIM, and LPIPS, with the final configuration achieving a PSNR of 33.64 dB for same-setup and 32.00 dB for cross-setup. These quantitative results are complemented by qualitative comparisons in Fig. 11 and 12, where the progressive reduction of visible seams around the face, geometric head misalignments, and face-color discrepancies clearly illustrates the effectiveness of each step.

The examples in Fig. 11 and 12 show particularly challenging cases of *cross-setup* integration, where the head and body are reconstructed from two distinct datasets: RenderMe-360 and DNA-Rendering, respectively. Although the same avatar identity appears in both, the differences in camera

Fig. 10 Qualitative results showing the interactive avatar's different modes of variation. Top row: the avatar's facial animation is driven by novel expression parameters generated from input audio speech, while the body undergoes corresponding procedural animation. Middle row: a static human pose is held while the face is animated with novel expressions. Bottom row: the user has full free-viewpoint navigation of the fully animated 3D avatar in real time.

systems, lighting conditions, hair style, make-up, and expressions introduce significant inconsistency in the 3DGS outputs. Naïve fusion of these assets can lead to discontinuities and degraded visual quality.

To address this, as discussed in Section 3.6, we exploit the shared canonical FLAME geometry to align both reconstructions. This alignment is critical for initializing a consistent coordinate frame, but due to dataset-specific variations in appearance and lighting, further processing is required. We thus perform Gaussian pruning in the overlapping region to prevent redundancy and eliminate interference between splats, followed by color optimization to achieve a seamless transition in shading and appearance between the head and body.

Altogether, our ablation results demonstrate the necessity of each integration step. Without careful handling, merging head and body reconstructions from different sources leads to perceptual and geometric inconsistencies. Our proposed pipeline successfully overcomes these challenges, enabling high-fidelity, real-time, and photorealistic full-body avatars—even in mismatched capture scenarios.

Head-Body Integration Limitations.

1. *Multi-view tracking failures* of head pose and facial keypoints, in both same-setup and cross-setup configurations, can result in alignment errors between the head and body. Head tracking may fail if the avatar's movements are too fast, motion-blurred, or if the face becomes occluded, e.g. by limbs, hair, or clothing. Such tracking errors also degrade the quality of the resulting 3DGS head reconstruction, leading to blurrier textures and sometimes spiky or floating Gaussian artifacts. Incorrect tracking can further cause the head to appear disembodied from the body.
2. *Appearance differences* between capture setups in the cross-setup scenario, e.g. variations in illumination or avatar appearance (makeup or facial hair) can result in artifacts when the head is integrated with the body. Our face-color optimization step helps close the gap in appearance, but it can only model so much by changing the colors of the Gaussians. Significant color differences or geometry changes, e.g. caused

by different hairstyles or clothing (e.g. headwear or glasses) that partially occludes the face, cannot be modeled by our method. These topological differences will result in blurry face textures when the face color is optimized.

3. *Expression differences* between the head and body can cause Gaussian intersection artifacts. For example, if the jaw of the underlying body model is open (in the DNA-Rendering body sequence) while the mouth of the expression-driven head model remains closed, their corresponding Gaussian representations may intersect, leading to a "double-chin" effect or visible protrusions. Our Gaussian pruning step alleviates this issue, though minor artifacts may still appear, slightly reducing the perceived realism of the avatar.
4. *Expression transfer* between the source (RenderMe-360) and target (DNA-Rendering) in the cross-setup scenario can lead to inaccurately modeled facial expressions in the fully-integrated avatar. This issue can be mitigated by normalizing the expression parameters between the source and target to be of similar scale.

Table 4 Ablation study of *same-setup* head-body integration. In this scenario, the head and body inputs are extracted from the same video sequences for each avatar from the DNA-Rendering dataset. We evaluate each component of the integration method across 3 avatars with varying appearance; comparing the renders of each combined avatar with their corresponding ground-truth images for 12 hold-out test views and all 225 frames.

	PSNR ↑	SSIM ↑	LPIPS ↓
Baseline Body Reconstruction	28.67	0.9527	0.0953
+ Head Reconstruction	31.23	0.9531	0.0933
+ Gaussian Pruning	31.74	0.9542	0.0922
+ Head Pose Alignment	**33.64**	**0.9586**	**0.0883**
+ Color Optimization	–	–	–

5.4 Computational Speed and Complexity

As we aim to achieve a real-time interactive conversational avatar, we evaluate our viewer's performance on two different systems: a Linux and a Windows system with different hardware configurations, to assess computational speed and efficiency. On the Linux system running Ubuntu equipped with an Intel Core i9-9900X CPU, a GPU with 10.7 TFLOPS (FP32) and 64 GB of RAM, we achieve an average frame rate of 70.60 FPS when rendering the head, body, and background (BG) together—equivalent to 14.16 ms per frame. On the Windows system running Windows 11, equipped with an Intel Core i9-13900HX CPU, GPU with 33 TFLOPS (FP32) and 32 GB of RAM, we achieved an average frame rate of **105.27 FPS** under the same conditions—equivalent to **9.50 ms** per frame. These results confirm that our system operates well above the typical threshold for real-time performance (30 FPS), ensuring smooth and responsive interactions. Detailed performance metrics are presented in Table 6.

Table 5 Ablation study of *cross-setup* head-body integration. In this scenario, for each avatar the body is extracted from the DNA-Rendering dataset while the head is extracted from RenderMe-360 for the same identity. We evaluate each component of the integration method across 2 avatars with varying appearance; comparing the renders of each combined avatar with their corresponding ground-truth images for 12 hold-out test views and all 225 frames.

	PSNR ↑	SSIM ↑	LPIPS ↓
Baseline Body Reconstruction	30.32	0.9581	0.1173
+ Head Reconstruction	26.52	0.9515	0.1208
+ Gaussian Pruning	27.23	0.9529	0.1195
+ Head Pose Alignment	28.96	0.9583	0.1133
+ Color Optimization	**32.00**	**0.9613**	**0.1115**

Table 6 Performance metrics of our interactive viewer on two system configurations. A ✓ indicates inclusion of a component; absence of a mark indicates exclusion.

BG	**Body**	**Head** (HeadGaS++)	**FPS (ms)** **Linux**	**Windows**
✓			125.68 (7.96)	169.92 (5.89)
✓	✓		114.86 (8.70)	152.23 (6.57)
✓		✓	74.60 (13.40)	112.53 (8.89)
✓	✓	✓	70.60 (14.16)	105.27 (9.50)

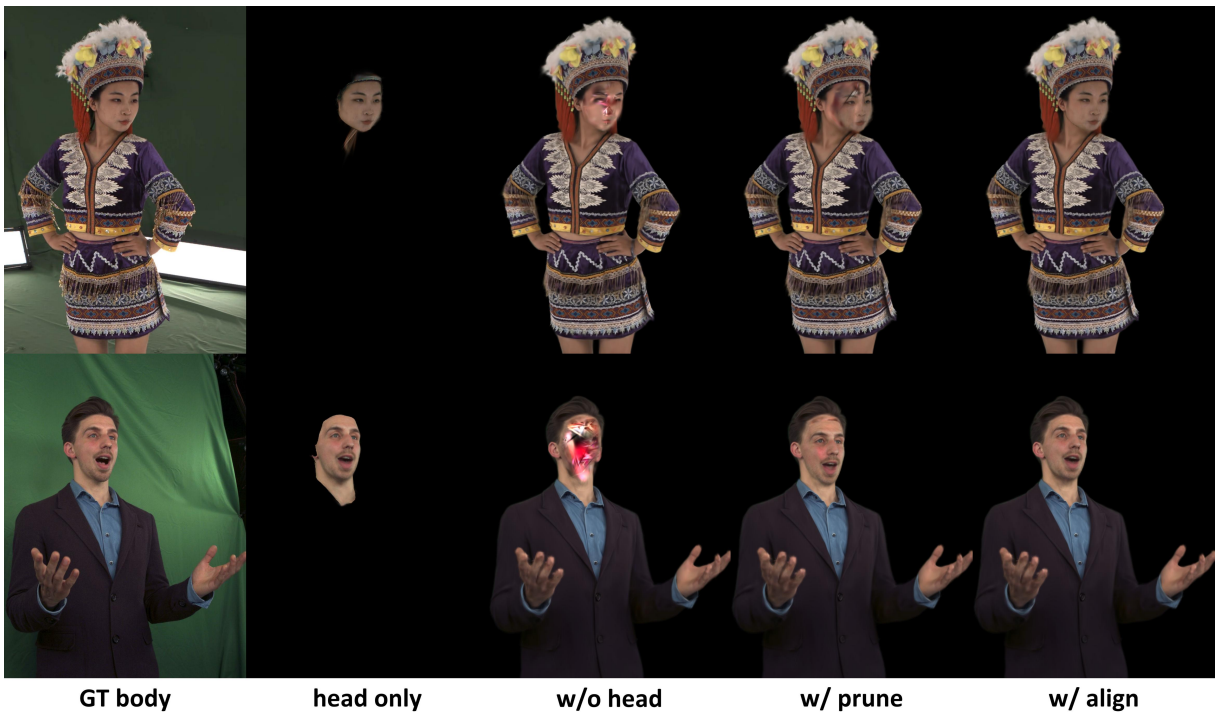


Fig. 11 Ablation study of *same-setup* head-body integration. Qualitative results for each integration component for the combined avatar compared to the ground truth.

6 Limitations

While our system achieves high-fidelity face and body reconstruction and animation, with real-time performance and conversational interactivity, several limitations remain that point to promising directions for future work.

Pre-captured Body Motion. Although our system supports full-body animation through procedural motion graphs, the body movements are currently generated from a dictionary of pre-captured motion segments and do not incorporate audio- or text-driven control. As a result, the body motion is not semantically aligned with the spoken content or conversational intent, which can reduce perceived realism and responsiveness in interactive settings. While this approach ensures real-time performance and gesture diversity, it lacks the contextual expressivity required for more natural avatar behavior. Future work could integrate speech- or language-conditioned gesture synthesis models (e.g., Chhatre et al. [95]) to generate plausible, emotionally grounded body movements in response to audio or text prompts. However, this is an ill-posed problem and achieving this without sacrificing real-time performance remains an open challenge, particularly due to the data and modeling requirements of current gesture synthesis pipelines.

Fixed Avatar Illumination. Our method assumes fixed lighting conditions during both training and inference, and the resulting avatars are not inherently relightable. As with most 3DGS pipelines, lighting is baked into the per-splat color attributes, limiting generalization to novel illumination environments. That said, because our training data is captured under calibrated, relatively diffuse lighting conditions in a controlled lightstage, the resulting splat colors approximate diffuse albedo. This makes it possible to perform approximate visual adaptation using global color corrections (e.g., white balance or affine color transforms) when compositing the avatar into new scenes. More advanced relighting capabilities could be incorporated via techniques from recent work [58, 96], though these approaches require significantly more complex training data — such as OLAT multi-view face captures or

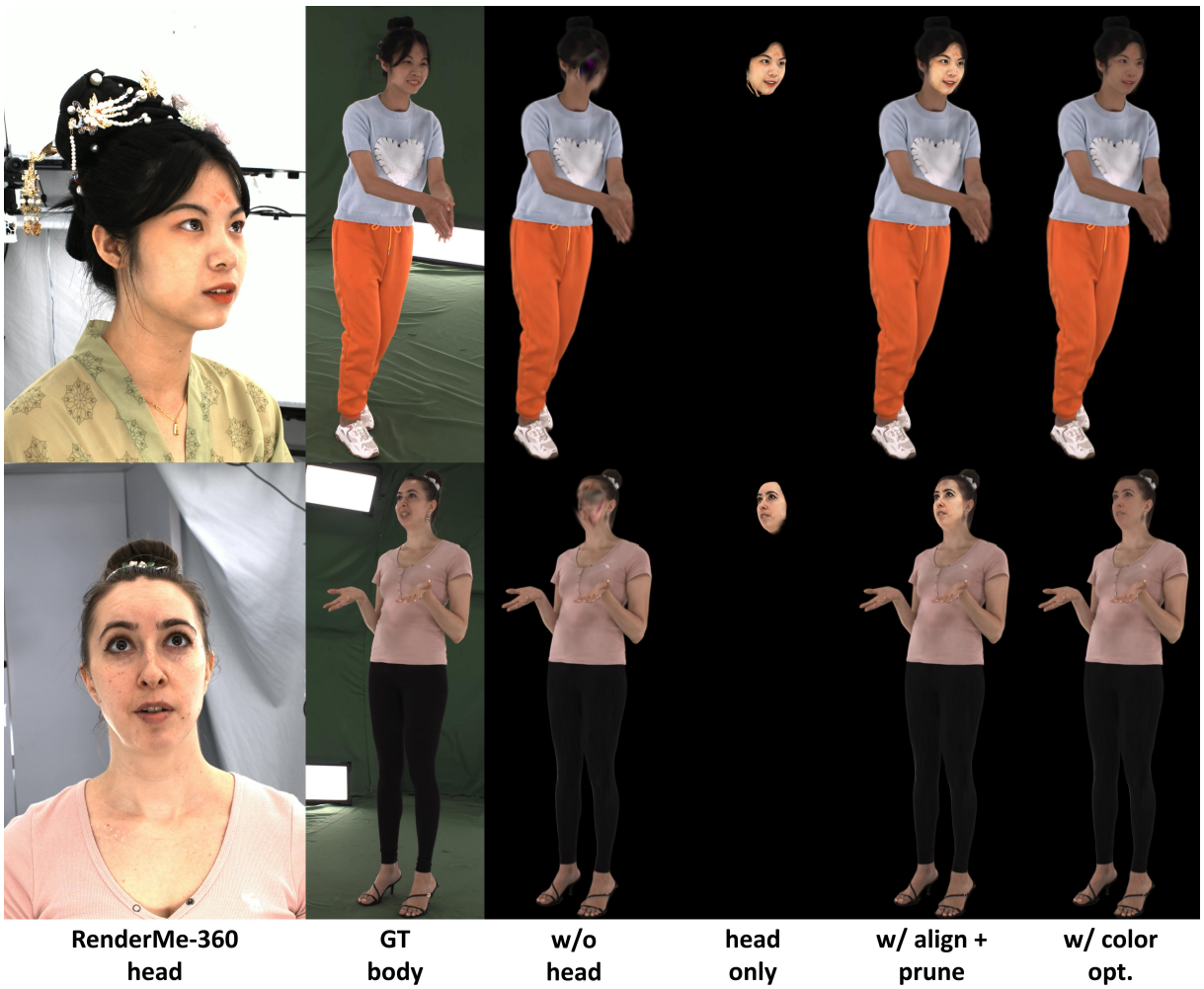


Fig. 12 Ablation study of *cross-setup* head-body integration. Qualitative results for each integration component for the combined avatar compared to the ground truth, where the head model is extracted from the RenderMe-360 dataset and the body is from DNA-Rendering.

neural reflectance models — which our current datasets (DNA-Rendering and RenderMe-360) do not provide. Alternatively, appearance embedding techniques such as BilaRF [97] or WildGaussians [98] may offer a practical compromise by modeling lighting variation implicitly across views.

Fixed Identity Avatars. Our current integration pipeline assumes that the head and body come from the same subject, an assumption that limits flexibility in applications such as virtual telepresence or avatar composition. Enabling cross-identity integration (e.g., combining a user's face with a generic animated body) would require robust normalization and alignment strategies that reconcile differing identity traits, including facial structure, hairstyle, and body proportions.

Controlled Capture Setup. Our system is designed for high-quality, studio-captured input data, typically acquired under controlled conditions with uniform lighting, synchronized cameras, and minimal occlusion. As such, performance may degrade in more challenging real-world scenarios — for instance, in-the-wild video with limited viewpoint coverage, rapid subject motion, or occlusions from props or loose clothing. Adapting our pipeline to handle such unconstrained inputs, including mobile capture or

casual home recording, would require enhanced robustness in geometry estimation, texture reconstruction, and temporal consistency modeling.

In summary, while our system offers a novel, photorealistic, and interactive avatar framework, future work could focus on integrating audio-conditioned body motion generation, improving robustness to diverse lighting and capture conditions, and enabling identity-flexible avatar composition to broaden applicability and realism in everyday use.

7 Conclusion

We have presented *Interactive Conversational 3D Virtual Human* (ICo3D), a comprehensive system for creating interactive, photorealistic 3D virtual human avatars. Our system seamlessly integrates dynamic full-body 3D reconstruction, dynamic facial expressions, and audio-driven conversational capabilities. We address limitations in the state-of-the-art by unifying novel methods for audio-driven facial animation, LLM-powered conversational interaction, and dynamic human rendering. Additionally, our method enables real-time rendering of avatars with free-viewpoint camera navigation, delivering an immersive and interactive user experience.

Our experimental results demonstrate the effectiveness of ICo3D in producing lifelike avatars with high-quality 3D reconstructions surpassing state-of-the-art methods. Our extended body reconstruction method SWinGS++ improves upon its predecessor with better motion estimation and sharper reconstructions, even with significant human motion. Similarly, our extended audio-driven 3D head model HeadGaS++ outperforms state-of-the-art methods in terms of rendering quality and lip synchronization in both self-driven and cross-driven settings. Qualitative results show our avatars are capable of engaging in natural, LLM-driven conversations, with precise synchronization between audio and realistic facial expressions.

Supplementary information. Please refer to our supplementary video for more visualizations and results.

Declarations

Data Availability

The findings and experimental section of this paper are based on multiple publicly accessible datasets which are freely available. These include the DNA-Rendering dataset [75], which can be accessed at https://dna-rendering.github.io/, the RenderMe-360 dataset [74], available at https://renderme-360.github.io/, and the TalkingGaussian dataset [87], available at https://fictionarry.github.io/TalkingGaussian/. Requests for access can be made to the corresponding authors, subject to approval in accordance with applicable regulations.Additionally, this manuscript includes visualizations on data not publicly available due to internal privacy regulations.

Acknowledgment: This version of the article has been accepted for publication, after peer review (when applicable) but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record will be available online at the publisher website (a link will be provided here once it is online). Use of this Accepted Version is subject to the publisher's Accepted Manuscript terms of use https://www.springernature.com/gp/open-research/policies/accepted-manuscript-terms

References

[1] Qian, S., Kirschstein, T., Schoneveld, L., et al.: GaussianAvatars: Photorealistic Head Avatars with Rigged 3D Gaussians. CVPR (2024)

[2] Xiang, J., Gao, X., Guo, Y., et al.: FlashAvatar: High-Fidelity Head Avatar with Efficient Gaussian Embedding. CVPR (2024)

[3] Zielonka, W., Bolkart, T., Thies, J.: Instant Volumetric Head Avatars. CVPR (2022)

[4] Xu, Y., Chen, B., Li, Z., et al.: Gaussian Head Avatar: Ultra High-Fidelity Head Avatar via Dynamic Gaussians. CVPR (2024)

[5] Li, T., Bolkart, T., Black, M.J., et al.: Learning a Model of Facial Shape and Expression from 4D Scans. ACM ToG (2017)

[6] ARKit. https://developer.apple.com/augmented-reality/arkit

[7] Dhamo, H., Nie, Y., Moreau, A., et al.: HeadGaS: Real-Time Animatable Head Avatars via 3D Gaussian Splatting. ECCV (2024)

[8] Shaw, R., Nazarczuk, M., Song, J., et al.: SWinGS: Sliding Windows for Dynamic 3D Gaussian Splatting. ECCV (2024)

[9] Loper, M., Mahmood, N., Romero, J., et al.: SMPL: A Skinned Multi-Person Linear Model. SIGGRAPH Asia (2015)

[10] Pavlakos, G., Choutas, V., Ghorbani, N., et al.: Expressive Body Capture: 3D Hands, Face, and Body from a Single Image. CVPR (2019)

[11] Mildenhall, B., Srinivasan, P.P., Tancik, M., et al.: NeRF. ECCV (2020)

[12] Müller, T., Evans, A., Schied, C., et al.: Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. ACM ToG (2022)

[13] Yu, A., Li, R., Tancik, M., et al.: PlenOctrees for Real-Time Rendering of Neural Radiance Fields. ICCV (2021)

[14] Li, Z., Niklaus, S., Snavely, N., et al.: Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. CVPR (2021)

[15] Pumarola, A., Corona, E., Pons-Moll, G., et al.: D-NeRF: Neural Radiance Fields for Dynamic Scenes. CVPR (2020)

[16] Park, K., Sinha, U., Barron, J.T., et al.: Nerfies: Deformable Neural Radiance Fields. ICCV (2021)

[17] Peng, S., Yan, Y., Shuai, Q., et al.: Representing Volumetric Videos as Dynamic MLP Maps. CVPR (2023)

[18] Işık, M., Rünz, M., Georgopoulos, M., et al.: HumanRF: High-Fidelity Neural Radiance Fields for Humans in Motion. ACM ToG (2023)

[19] Wang, P., Liu, L., Liu, Y., et al.: NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-View Reconstruction. arXiv preprint arXiv:2106.10689 (2021)

[20] Takikawa, T., Litalien, J., Yin, K., et al.: Neural Geometric Level of Detail: Real-Time Rendering with Implicit 3D Shapes. CVPR (2021)

[21] Saito, S., Simon, T., Saragih, J.M., et al.: PIFuHD: Multi-Level Pixel-Aligned Implicit Function for High-Resolution 3D Human Digitization. CVPR (2020)

[22] Jiang, B., Hong, Y., Bao, H., et al.: SelfRecon: Self Reconstruction Your Digital Avatar From Monocular Video. CVPR (2022)

[23] Kerbl, B., Kopanas, G., Leimkühler, T., et al.: 3D Gaussian Splatting for Real-Time Radiance Field Rendering. ACM ToG (2023)

[24] Luiten, J., Kopanas, G., Leibe, B., et al.: Dynamic 3D Gaussians: Tracking by Persistent Dynamic View Synthesis. 3DV (2024)

[25] Yang, Z., Yang, H., Pan, Z., et al.: Real-Time Photorealistic Dynamic Scene Representation and Rendering with 4D Gaussian Splatting. ICLR (2024)

[26] Yang, Z., Gao, X., Zhou, W., et al.: Deformable 3D Gaussians for High-Fidelity Monocular Dynamic Scene Reconstruction. CVPR (2024)

[27] Huang, Y.-H., Sun, Y.-T., Yang, Z., et al.: SC-GS: Sparse-Controlled Gaussian Splatting for Editable Dynamic Scenes. CVPR (2024)

[28] Lin, Y., Dai, Z., Zhu, S., et al.: GaussianFlow: 4D Reconstruction with Dynamic 3D Gaussian Particle. CVPR (2024)

[29] Li, Z., Chen, Z., Li, Z., et al.: Spacetime Gaussian Feature Splatting for Real-Time Dynamic View Synthesis. CVPR (2024)

[30] Sun, J., Jiao, H., Li, G., et al.: 3DGStream: On-the-fly Training of 3D Gaussians for

Efficient Streaming of Photo-Realistic Free-Viewpoint Videos. CVPR (2024)

[31] Moreau, A., Song, J., Dhamo, H., et al.: Human Gaussian Splatting: Real-Time Rendering of Animatable Avatars. CVPR (2024)

[32] Li, Z., Zheng, Z., Wang, L., et al.: Animatable Gaussians: Learning Pose-Dependent Gaussian Maps for High-Fidelity Human Avatar Modeling. CVPR (2024)

[33] Kocabas, M., Chang, J.-H.R., Gabriel, J., et al.: HUGS: Human Gaussian Splatting. CVPR (2024)

[34] Hu, L., Zhang, H., Zhang, Y., et al.: GaussianAvatar: Towards Realistic Human Avatar Modeling from a Single Video via Animatable 3D Gaussians. CVPR (2024)

[35] Pang, H., Zhu, H., Kortylewski, A., et al.: ASH: Animatable Gaussian Splats for Efficient and Photoreal Human Rendering. CVPR (2024)

[36] Qian, Z., Wang, S., Mihajlovic, M., et al.: 3DGS-Avatar: Animatable Avatars via Deformable 3D Gaussian Splatting. CVPR (2024)

[37] Wu, G., Yi, T., Fang, J., et al.: 4D Gaussian Splatting for Real-Time Dynamic Scene Rendering. CVPR (2024)

[38] Wen, J., Zhao, X., Ren, Z., et al.: GoMAvatar: Efficient Animatable Human Modeling from Monocular Video Using Gaussians-on-Mesh. CVPR (2024)

[39] Jung, H., Brasch, N., Song, J., et al.: Deformable 3D Gaussian Splatting for Animatable Human Avatars. arXiv preprint arXiv:2312.15059 (2023)

[40] Bagautdinov, T., Wu, C., Simon, T., et al.: Driving-Signal Aware Full-Body Avatars. ACM ToG (2021)

[41] Merry, B., Marais, P., Gain, J.: Animation Space: A Truly Linear Framework for Character Animation. ACM ToG (2006)

[42] Kavan, L., Collins, S., Žára, J., *et al.*: Skinning with Dual Quaternions. In: Symposium on Interactive 3D Graphics and Games (2007)

[43] Hong, Y., Peng, B., Xiao, H., et al.: HeadNeRF: A Realtime NeRF-based Parametric Head Model. CVPR (2021)

[44] Mihajlovic, M., Bansal, A., Zollhoefer, M., et al.: KeypointNeRF: Generalizing Image-Based Volumetric Avatars using Relative Spatial Encoding of Keypoints. arXiv preprint arXiv:2205.04992 (2022)

[45] Wang, D., Chandran, P., Zoss, G., et al.: MoRF: Morphable Radiance Fields for Multiview Neural Head Modeling. SIGGRAPH (2022)

[46] YoungkyoonJang, JialiZheng, Song, J., et al.: VSCHH 2023: A Benchmark for the View Synthesis Challenge of Human Heads. ICCVW (2023)

[47] Kirschstein, T., Qian, S., Giebenhain, S., et al.: NeRSemble: Multi-View Radiance Field Reconstruction of Human Heads. ACM ToG (2023)

[48] Lombardi, S., Saragih, J.M., Simon, T., et al.: Deep Appearance Models for Face Rendering. ACM ToG (2018)

[49] Lombardi, S., Simon, T., Schwartz, G., et al.: Mixture of Volumetric Primitives for Efficient Neural Rendering. ACM ToG (2021)

[50] Kim, H., Garrido, P., Tewari, A.K., et al.: Deep Video Portraits. ACM ToG (2018)

[51] Thies, J., Zollhöfer, M., Stamminger, M., et al.: Face2Face: Real-Time Face Capture and Reenactment of RGB Videos. CVPR (2016)

[52] Aylagas, M.V., Leon, H.A., Teye, M., et al.: Voice2Face: Audio-Driven Facial and Tongue Rig Animations with cVAEs. Comput. Graph. Forum (2022)

[53] Grassal, P.-W., Prinzler, M., Leistner, T., et al.: Neural Head Avatars from Monocular

RGB Videos. CVPR (2021)

[54] Gao, X., Zhong, C., Xiang, J., et al.: Reconstructing Personalized Semantic Facial NeRF Models from Monocular Video. ACM ToG (2022)

[55] Khakhulin, T., Sklyarova, V., Lempitsky, V.S., Zakharov, E.: Realistic one-shot mesh-based head avatars. ECCV (2022)

[56] Moon, G., Shiratori, T., Saito, S.: Expressive whole-body 3D gaussian avatar. In: ECCV (2024)

[57] Chen, J., Hu, J., Wang, G., Jiang, Z., Zhou, T., Chen, Z., Lv, C.: Taoavatar: Real-time lifelike full-body talking avatars for augmented reality via 3d gaussian splatting. In: CVPR (2025)

[58] Saito, S., Schwartz, G., Simon, T., et al.: Relightable Gaussian Codec Avatars. CVPR (2024)

[59] Devlin, J., Chang, M.-W., Lee, K., *et al.*: BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. In: Conf. of the North American Chapter of the Association for Computational Linguistics (2019)

[60] Gao, J., Lin, C.-Y.: Introduction to the Special Issue on Statistical Language Modeling. ACM Transactions on Asian Language Information Processing (2004)

[61] Brown, P.F., Della Pietra, V.J., deSouza, P.V., et al.: Class-Based n-gram Models of Natural Language. Computational Linguistics (1992)

[62] Floridi, L., Chiriatti, M.: GPT-3: Its Nature, Scope, Limits, and Consequences. Minds Mach. (2020)

[63] Ouyang, L., Wu, J., Jiang, X., et al.: Training Language Models to Follow Instructions with Human Feedback. NeurIPS (2024)

[64] OpenAI, et al.: GPT-4 Technical Report. arXiv preprint arXiv:2303.08774 (2024)

[65] Vaswani, A., Shazeer, N., Parmar, N., *et al.*: Attention is All You Need. In: 31st International Conf. on Neural Information Processing Systems (2017)

[66] Chang, Y., Wang, X., Wang, J., et al.: A Survey on Evaluation of Large Language Models. ACM Trans. Intell. Syst. Technol. (2024)

[67] Bai, J., Bai, S., Chu, Y., et al.: Qwen Technical Report. arXiv preprint arXiv:2309.16609 (2023)

[68] Zhao, W.X., Zhou, K., Li, J., et al.: A Survey of Large Language Models. arXiv preprint arXiv:2303.18223 (2023)

[69] Schönberger, J.L., Frahm, J.-M.: Structure-from-Motion Revisited. CVPR (2016)

[70] Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual Losses for Real-Time Style Transfer and Super-Resolution. ECCV (2016)

[71] Maggioni, M., Tanay, T., Babiloni, F., et al.: Tunable Convolutions with Parametric Multi-Loss Optimization. CVPR (2023)

[72] Teed, Z., Deng, J.: RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. ECCV (2020)

[73] Fridovich-Keil, S., Meanti, G., Warburg, F., et al.: K-Planes: Explicit Radiance Fields in Space, Time, and Appearance. CVPR (2023)

[74] Pan, D., Zhuo, L., Piao, J., *et al.*: RenderMe-360: Large Digital Asset Library and Benchmark Towards High-Fidelity Head Avatars. In: 37th Conf. on Neural Information Processing Systems Datasets and Benchmarks Track (2023)

[75] Cheng, W., Chen, R., Fan, S., et al.: DNA-Rendering: A Diverse Neural Actor Repository for High-Fidelity Human-Centric Rendering. ICCV (2023)

[76] Zhang, Z., Geiger, J., Pohjalainen, J., et al.: Deep Learning for Environmentally Robust Speech Recognition: An Overview of Recent Developments. ACM Trans. Intell. Syst.

Technol. (2018)

[77] Lin, J., Tang, J., Tang, H., et al.: AWQ: Activation-Aware Weight Quantization for On-Device LLM Compression and Acceleration. Proceedings of Machine Learning and Systems (2024)

[78] Qin, Z., Zhao, W., Yu, X., et al.: OpenVoice: Versatile Instant Voice Cloning. arXiv preprint arXiv:2312.01479 (2023)

[79] Zheng, Y., Wang, Y., Wetzstein, G., et al.: PointAvatar: Deformable Point-Based Head Avatars from Videos. CVPR (2022)

[80] Peng, Z., Hu, W., Shi, Y., et al.: SyncTalk: The Devil is in the Synchronization for Talking Head Synthesis. CVPR (2024)

[81] Wang, Z., Bovik, A.C., Sheikh, H.R., et al.: Image Quality Assessment: From Error Visibility to Structural Similarity. IEEE Trans. on Image Processing (2004)

[82] Zhang, R., Isola, P., Efros, A.A., et al.: The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. CVPR (2018)

[83] Li, Z., Chen, Z., Li, Z., et al.: Spacetime Gaussian Feature Splatting for Real-Time Dynamic View Synthesis. CVPR (2024)

[84] Moreau, A., Brahimi, M., Shaw, R., Papaioannou, A., Tanay, T., et al.: Better together: Unified motion capture and 3d avatar reconstruction. ArXiv **abs/2503.09293** (2025)

[85] Habibie, I., Elgharib, M., Sarkar, K., et al.: A Motion Matching-Based Framework for Controllable Gesture Synthesis from Speech. SIGGRAPH (2022)

[86] Mir, A., Puig, X., Kanazawa, A., et al.: Generating Continual Human Motion in Diverse 3D Scenes. 3DV (2024)

[87] Li, J., Zhang, J., Bai, X., et al.: TalkingGaussian: Structure-Persistent 3D Talking Head Synthesis via Gaussian Splatting. ECCV, 127–145 (2024)

[88] Li, J., Zhang, J., Bai, X., et al.: Efficient Region-Aware Neural Radiance Fields for High-Fidelity Talking Portrait Synthesis. ICCV, 7568–7578 (2023)

[89] Tang, J., Wang, K., Zhou, H., et al.: Real-time Neural Radiance Talking Portrait Synthesis via Audio-spatial Decomposition. arXiv preprint arXiv:2211.12368 (2022)

[90] Cho, K., Lee, J., Yoon, H., et al.: GaussianTalker: Real-Time High-Fidelity Talking Head Synthesis with Audio-Driven 3D Gaussian Splatting (2024)

[91] Chen, L., Li, Z., Maddox, R.K., et al.: Lip Movements Generation at a Glance. ECCV, 520–535 (2018)

[92] Son Chung, J., Senior, A., Vinyals, O., et al.: Lip Reading Sentences in the Wild. CVPR, 6447–6456 (2017)

[93] Chung, J.S., Zisserman, A.: Out of Time: Automated Lip Sync in the Wild. ACCV 2016 International Workshops, 251–263 (2016)

[94] Suwajanakorn, S., Seitz, S.M., Kemelmacher-Shlizerman, I.: Synthesizing Obama: Learning Lip Sync from Audio. ACM ToG **36**(4), 1–13 (2017)

[95] Chhatre, K., Danecek, R., Athanasiou, N., Becherini, G., Peters, C., Black, M.J., Bolkart, T.: Emotional speech-driven 3d body animation via disentangled latent diffusion. CVPR, 1942–1953 (2023)

[96] Schmidt, J., Giebenhain, S., Nießner, M.: Becominglit: Relightable gaussian avatars with hybrid neural shading. ArXiv **abs/2506.06271** (2025)

[97] Wang, Y., Wang, C., Gong, B., Xue, T.: Bilateral guided radiance field processing. ACM ToG **43**, 1–13 (2024)

[98] Kulhánek, J., Peng, S., Kúkelová, Z., Pollefeys, M., Sattler, T.: Wildgaussians: 3d gaussian splatting in the wild. NeurIPS (2024)