

GENERAL INSTRUCTION

- **Authors:** Please check and confirm whether the name of the corresponding author is correct as set.
- **Authors:** When you submit your corrections, please either annotate the IEEE Proof PDF or send a list of corrections. Do not send new source files as we do not reconvert them at this production stage.
- **Authors:** Carefully check the page proofs (and coordinate with all authors); additional changes or updates **WILL NOT** be accepted after the article is published online/print in its final form. Please check author names and affiliations, funding, as well as the overall article for any errors prior to sending in your author proof corrections. Your article has been peer reviewed, accepted as final, and sent in to IEEE. No text changes have been made to the main part of the article as dictated by the editorial level of service for your publication.
- **Authors:** Per IEEE policy, one complimentary proof will be sent to only the Corresponding Author. The Corresponding Author is responsible for uploading one set of corrected proofs to the IEEE Author Gateway.

QUERIES

- Q1. Author: Please confirm or add details for any funding or financial support for the research of this article.
- Q2. Author: Please check and confirm whether the author affiliations in the first footnote are correct as set.
- Q3. Author: Please provide complete bibliographic details for Ref. [18].
- Q4. Author: Please update Refs. [28] and [49], if already published.

LivingAvatars: Robust Head Reconstruction With Gaussian Lifecycle Management and Neural Detail Synthesis

Xingyuan Lu , Mingyang Zhao , Chunyu Lin , *Member, IEEE*, and Dong-Ming Yan, *Member, IEEE*

Abstract—Creating robust, high-fidelity head avatars with 3D Gaussian Splatting (3DGS) remains constrained by two bottlenecks: one is the *representation bottleneck* that limits the modeling of dynamic details, and the other is the *dynamic robustness bottleneck* that causes information loss and point cloud inflation. To this end, we present LivingAvatars, which addresses both issues through two key designs: (i) a functionally decoupled Gaussian representation that separates identity-stable base Gaussians from detail Gaussians carrying high-frequency appearance, while a lightweight neural decoder injects features into the latter to synthesize expression-dependent details; and (ii) a Gaussian lifecycle management system that endows Gaussians with multiple states and replaces destructive pruning with probabilistic selective pruning and temporary hiding to suppress redundant growth, ensuring a compact and robust representation. On novel-view synthesis and self-reenactment, LivingAvatars achieves overall better metrics than existing methods. In cross-identity reenactment, it better preserves transient, expression-dependent signals, yielding higher perceptual fidelity. These results suggest that functionally decoupled Gaussian representations and lifecycle management are effective principles for building neural head avatars that remain robust and compact under dynamic occlusions, reliably reconstructing high-frequency details and improving overall realism.

Index Terms—3DGS, neural avatars, head reconstruction, Gaussian lifecycle management, neural detail synthesis.

I. INTRODUCTION

THE real-time creation of high-fidelity, expressive digital human avatars has been a long-standing goal in computer graphics, with applications in Virtual Reality (VR) and Augmented Reality (AR), and immersive communication. The core

challenge is to capture and render complex, dynamic details (e.g., expression-driven wrinkles and transient specular high-lights). Crucially, this must not sacrifice real-time performance.

Recently, Neural Radiance Fields (NeRF) [1] and their variants [2], [3], [4] have achieved impressive success in reconstructing static scenes. Subsequent work extended NeRF to model dynamic scenes, covering both arbitrary motion [5], [6], [7] and scenarios tailored to human subjects [8], [9]. However, despite high-quality view synthesis, these methods cannot generalize to new poses and expressions due to their lack of explicit, animatable control.

Subsequently, the recent advent of 3D Gaussian Splatting (3DGS) [10], [11] offers a compelling alternative, delivering an unprecedented combination of rendering quality and real-time speed. This breakthrough has spurred its adoption for creating drivable head avatars [12], [13], which typically operate by binding Gaussians to a parametric model like the Facial Linear Animation Model (FLAME) [14]. Despite this progress, existing methods face two fundamental limitations. We distill the gap into two orthogonal bottlenecks.

First, a *representation bottleneck* arises from using static per-Gaussian attributes, which fail to model high-frequency, expression-driven appearance changes. Simply increasing the number of Gaussians is inefficient and does not capture the intrinsic correlation between expression and detail.

Second, a *dynamic robustness bottleneck* stems from the destructive pruning in standard 3DGS, which permanently erases temporarily occluded Gaussians (e.g., teeth during mouth closure). This causes information loss and leads to point cloud inflation from a wasteful deletion-recreation cycle, degrading both quality and efficiency. Additionally, without any pruning mechanism, redundant and floating Gaussians accumulate during training, causing optimization difficulties and storage overhead.

Building on these observations, we introduce LivingAvatars, a framework that replaces static Gaussians and destructive pruning with two core innovations. The architecture is shown in Fig. 1. To address the representation bottleneck, we propose a *functionally decoupled Gaussian representation* that separates identity-stable *base Gaussians* from expression-driven *detail Gaussians*. Each detail Gaussian carries a learnable embedding to encode instance-specific attributes. A lightweight neural decoder conditions on the embedding and geometric context

Received 26 November 2025; revised 13 May 2026; accepted 28 June 2026. This work was supported in part by the Beijing Natural Science Foundation under Grant Z240002, in part by the Key Project of the National Natural Science Foundation of China under Grant 12494550 and Grant 12494553, and in part by NSFC-FDCT Joint Project under Grant 62461160259. Recommended for acceptance by X. Zhou. (*Corresponding author: Chunyu Lin.*)

Xingyuan Lu and Chunyu Lin are with the Beijing Jiaotong University, Beijing 100044, China (e-mail: 25140223@bjtu.edu.cn; cylin@bjtu.edu.cn).

Mingyang Zhao is with the State Key Laboratory of Mathematical Sciences, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing 100190, China, and also with the University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: zhaomingyang@amss.ac.cn).

Dong-Ming Yan is with the MAIS, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: yandongming@gmail.com).

Digital Object Identifier 10.1109/TVCG.2026.3710718

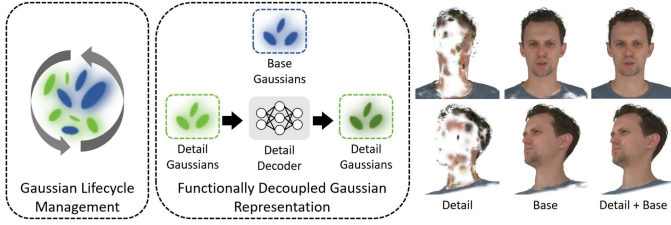


Fig. 1. LivingAvatars creates high-fidelity head avatars using two innovations: a functionally-decoupled Gaussian representation that separates identity (base Gaussians) from expression (detail Gaussians), and a Gaussian lifecycle management strategy for robust optimization. We visualize the base Gaussians, detail Gaussians, and the final rendered result.

to synthesize high-frequency, expression-dependent appearance (e.g., wrinkles, hair sheen). For robustness, *Gaussian lifecycle management* replaces destructive pruning with a non-destructive “Hibernation-Rejuvenation” cycle that preserves occluded information and prevents point-cloud inflation. Additionally, probabilistic selective pruning suppresses redundant detail Gaussians without harming fidelity, enabling efficient training with high-quality results.

We summarize our contributions as follows:

- *Functionally Decoupled Gaussian Representation*: We introduce a functionally decoupled representation that decomposes Gaussians into a stable geometric core (base Gaussians) and a dynamic detail layer (detail Gaussians). Each detail Gaussian carries learnable feature embeddings that encode its unique characteristics, while a neural decoder synthesizes appearance based on these embeddings and geometric context, enabling the efficient rendering of high-frequency, expression-driven details while preserving identity.
- *Gaussian Lifecycle Management*: We design a novel lifecycle management mechanism, featuring a “Hibernation-Rejuvenation” cycle, as a non-destructive alternative to standard pruning. This approach preserves information through occlusions and fundamentally reduces the point cloud inflation that plagues existing dynamic methods. In addition, we introduce probabilistic selective detail pruning to suppress redundant detail Gaussians without compromising fidelity, improving efficiency during training.
- *High-Fidelity Real-Time Avatar System*: By integrating these contributions, we build an end-to-end system that reconstructs high-fidelity digital avatars from multi-view video and operates in real-time.

Empirically, our system improves perceptual fidelity on novel-view synthesis and reenactment, and better preserves transient, expression-dependent signals in cross-identity scenarios.

II. RELATED WORK

This section reviews related work along these dimensions. We first survey foundational representations from parametric meshes to neural fields and modern 3DGS-based methods, and also note recent surveys on 3D Gaussian splatting, generative 3D reconstruction, diffusion-based 3D generation, and real-time

neural rendering pipelines [15], [16], [17], [18]. We then focus on recent advances in modeling expression-dependent details and handling transient occlusions that critically affect avatar quality.

A. 3D Head Representations

We group prior head representations into parametric mesh models, neural fields, and 3DGS-based head avatars. Parametric models like the 3D Morphable Model (3DMM) [19], [20], [21] and its successor the Facial Linear Animation Model (FLAME) [14] were foundational for head avatar reconstruction, offering low-dimensional control over identity and expression [22], [23], [24], [25], [26], [27]. However, their fixed mesh topology and linear expression bases struggled to represent unstructured details like hair and to generate non-linear, dynamic wrinkles. Neural fields such as NeRF [1] relaxed topological constraints but were typically too slow for real-time use. The advent of 3DGS [10] provided a breakthrough for real-time, high-fidelity avatars, spurring numerous Gaussian-based dynamic human representations [28], [29], [30], [31], [32], [33], [34], [35]. For head avatars, GaussianAvatars [12] binds Gaussians to a FLAME mesh for animatable rendering. TGA (Tensorial Gaussian Avatar) [36] proposes a compact tensorial Gaussian head representation that uses triplanes for neutral static appearance and lightweight 1D feature lines for expression-related dynamic texture, while RGBAvatar [37] introduces reduced Gaussian blendshapes, mapping 3DMM parameters to compact blendshape weights with a lightweight MLP to model personalized expressions with fewer primitives. Despite these advances, existing methods still struggle to capture rich dynamic details. Our work addresses this issue by functionally decoupling identity from dynamic details and replacing destructive pruning with a non-destructive lifecycle.

B. Dynamic Appearance and Occlusion Robustness in Gaussian Avatars

Building on 3DGS-based head avatars, a major thread targeted expression-dependent appearance modeling, while a complementary thread emphasized robustness to transient occlusions. Standard 3DGS relied on static Spherical Harmonics (SH), which underfitted expression-driven changes; subsequent works [12], [38], [39], [40] introduced corrective offsets or hybrid schemes, but such local adjustments under-expressed complex, correlated high-frequency details (e.g., wrinkles, specular micro-structure) and tended to couple detail capacity with Gaussian counts. TGA [36] addresses this issue with a compact tensorial representation that stores neutral appearance in triplanes and models expression-dependent dynamic texture with lightweight feature lines decoded into opacity offsets. RGBAvatar [37] instead adopts reduced Gaussian blendshapes, using a lightweight MLP to predict compact blendshape weights from 3DMM parameters so that personalized facial details can be captured with fewer Gaussian primitives. In parallel, SurfHead [41] proposed affine rig blending for 2D Gaussian surfels; with Jacobian Blend Skinning (JBS), it better handled stretching and shearing under extreme expressions and mitigated

specular-reflection and concave-cornea artifacts. HRAvatar [42] leveraged 3DGS with learnable blendshapes and learnable Linear Blend Skinning to enable flexible Gaussian deformation, and decomposed head appearance into physically grounded components (e.g., albedo and roughness) combined with a physically based shading model, achieving high-quality reconstructions. Dong and Zhang [43] proposed layered viewpoint simulation and continuous lighting simulation to construct global depth and brightness representations, addressing detail loss and blur caused by fixed viewpoints and constant lighting in training videos. Ma et al. [44] introduced a decoupled two-stage framework that generates facial landmarks from audio via a Transformer and employs Gaussian-landmark-based position encoding to optimize NeRF ray sampling for talking head rendering. GUAVA [45] proposes a generalizable single-image upper-body 3D Gaussian avatar framework, combining an expressive human model (EHM), inverse texture mapping, and projection sampling to enable feed-forward reconstruction of animatable avatars with real-time rendering.

Orthogonal to appearance modeling, robustness work addressed transient occlusions. Standard destructive pruning caused irreversible information loss for temporarily occluded Gaussians (e.g., eyes and mouth), producing flicker and deletion-recreation cycles that inflate point counts. GaussianAvatar-Editor [46] down-weighted occluded regions, but reduced gradients still weakened parameters and could not prevent drift. Compared with methods that mainly emphasize compact appearance parameterization or efficient blendshape control, our method explicitly couples dynamic detail synthesis with a non-destructive lifecycle mechanism, targeting both transient high-frequency effects and robustness under severe self-occlusion.

III. PRELIMINARIES

A. 3D Gaussian Splatting

3DGS [10] represents a scene using a set of anisotropic 3D Gaussians. Each Gaussian is defined by a center μ and a covariance matrix Σ :

$$G(\mathbf{x}) = \exp\left(-\frac{1}{2}(\mathbf{x} - \mu)^T \Sigma^{-1}(\mathbf{x} - \mu)\right). \quad (1)$$

For an optimizable representation, the covariance matrix Σ is factored into a scaling matrix S and a rotation matrix R :

$$\Sigma = R S S^T R^T. \quad (2)$$

In practice, each Gaussian is parameterized by a position $\mu \in \mathbb{R}^3$, a scaling vector $s \in \mathbb{R}^3$, and a rotation quaternion $q \in \mathbb{R}^4$.

World-space distances are expressed in meters throughout this paper; all distance-like thresholds (e.g., search radii) follow this convention.

The color C for a pixel is rendered by alpha-blending the projected Gaussians, sorted by depth. Let N denote the number of visible Gaussians along the viewing ray; then

$$C = \sum_{i=1}^N \left(c_i \tilde{\alpha}_i \prod_{j=1}^{i-1} (1 - \tilde{\alpha}_j) \right). \quad (3)$$

Here c_i is the color derived from Spherical Harmonics (SH), and the blending weight $\tilde{\alpha}_i$ is defined as the product of the per-Gaussian opacity α_i (after Sigmoid) and the projected 2D Gaussian's screen-space density ρ_i .

B. FLAME Tracking

We directly use the data provided by GaussianAvatar [12]. To capture facial dynamics, the process begins by extracting 51 internal facial landmarks, excluding the contour, using an advanced detector [47]. Subsequently, an objective function is optimized to solve for the per-frame pose, expression, and translation parameters, along with a shared shape parameter. This objective function is composed of a keypoint loss, a color loss rendered by NVDiffRast [48], and regularization terms. The optimization process is conducted in two stages: an initial per-frame fitting followed by a global joint fine-tuning to improve inter-frame coherence and overall accuracy. To better capture the oral cavity, the FLAME template is extended by adding geometry for the teeth.

IV. METHOD

As shown in Fig. 2, LivingAvatars creates drivable, photorealistic head avatars from multi-view RGB videos by extending GaussianAvatars [12] with two complementary designs.

First, we introduce a *functionally-decoupled Gaussian representation*. A stable geometric core of base Gaussians preserves identity, while detail Gaussians model high-frequency, expression/pose-dependent effects (e.g., wrinkles, specular highlights). Detail appearance is synthesized by non-shared per-level MLP decoders (one decoder per detail level shared within that level) that take a learnable latent code, positional encoding, and geometric context as input. During rendering, decoder deltas are temporarily fused into the base attributes; only upon graduation are stable contributions baked, avoiding moving-target issues and preserving the intended functional split.

Second, to robustly handle transient occlusions and suppress redundant growth, we propose *Gaussian lifecycle management*. A four-state process (Baby, Mature, Decaying, Hibernating) replaces destructive pruning with a non-destructive Hibernation-Rejuvenation cycle guided by an error map, while probabilistic selective detail pruning reduces active detail Gaussians without deleting parameters. Validated detail Gaussians can graduate into the Base set, integrating both designs into a compact and robust representation.

A. Neural Dynamic Detail Synthesis

To model high-frequency, non-rigid details, ranging from expression-driven wrinkles to the complex geometries of hair and clothing, which static per-Gaussian appearance fails to capture, we reformulate detail learning as a structured, generative process instead of brute-force optimization.

1) *Functionally-Decoupled Gaussian Representation*: Our method decomposes the scene into *static base Gaussians* and *detail Gaussians*. The base Gaussians constitute a substantial,

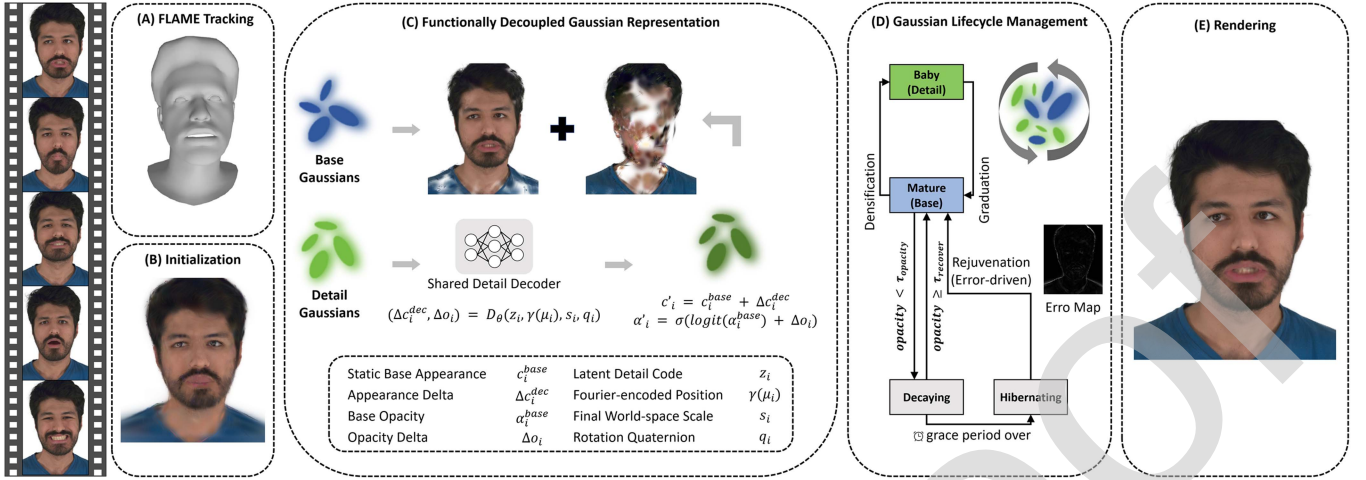


Fig. 2. Overview of the LivingAvatars framework. The pipeline begins by binding 3D Gaussians to a parametric FLAME mesh (A, B) [12], [14]. Our core representation, the functionally-decoupled Gaussian representation (C), decomposes these into two types: stable *base Gaussians* (blue) that define identity and dynamic *detail Gaussians* (green) for high-frequency effects. The appearance of detail Gaussians is synthesized by a *Detail Decoder* (non-shared). Concurrently, a *Gaussian lifecycle management* system (D) governs all Gaussians, using state transitions to handle occlusions and model evolution. This includes an error-driven rejuvenation process guided by an Error Map. Lastly, both Gaussian types are rendered together to produce the final image (E).

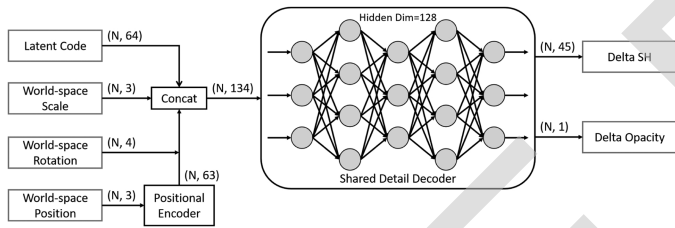


Fig. 3. Our Detail Decoder architecture (non-shared per level): a vector formed by concatenating the latent code, geometric attributes, and positional-encoded features is processed by an MLP to predict appearance and opacity variations that modulate the static base appearance.

stable geometric core with largely static attributes, capturing the subject’s core identity. Complementing this core, a population of detail Gaussians is distributed across the surface for high-frequency effects. Instead of being stored, their final appearance is dynamically modulated by a neural decoder. This focuses the model’s capacity on an efficient, uniformly controlled distribution of details.

2) *Detail Decoder Network*: We employ a *non-shared MLP decoder*, D_θ , that modulates detail Gaussians with a compact, learned prior (Fig. 3). Concretely, each Gaussian maintains an integer attribute $\ell_i \in \{0, 1, \dots, L\}$ called the *detail level*, where $\ell_i=0$ denotes a Base (Mature) Gaussian and $\ell_i>0$ denotes a Detail (Baby) Gaussian. The maximum level L (default $L=3$) bounds the recursive depth of detail spawning. When a new Detail Gaussian is spawned from a parent with level ℓ_p , its level is computed as $\ell_{\text{child}} = \min(\ell_p + 1, L)$. The system instantiates independent decoder heads per detail level (levels $1 \dots L$), while all detail Gaussians at the same level share that level’s decoder parameters; we do not instantiate a separate MLP per Gaussian. For a detail Gaussian i , the input concatenates: a

learnable latent code z_i that serves as an instance-specific feature embedding encoding the Gaussian’s unique characteristics, a Fourier-encoded world-space position $\gamma(\mu_i)$ (spatial context), and its world-space scale s_i and rotation q_i (geometric context). The network predicts deltas for SH appearance and logit-space opacity:

$$(\Delta c_i^{\text{dec}}, \Delta o_i) = D_\theta(z_i, \gamma(\mu_i), s_i, q_i), \quad (4)$$

where Δc_i^{dec} and Δo_i denote the deltas for SH coefficients and logit-space opacity.

We use the standard Fourier positional encoding $\gamma: \mathbb{R}^3 \rightarrow \mathbb{R}^{6B+3}$ with B frequency bands (default $B=10$), applied element-wise to 3D coordinates:

$$\gamma(\mu) = [\mu, \{\sin(2^k \pi \mu)\}_{k=0}^{B-1}, \{\cos(2^k \pi \mu)\}_{k=0}^{B-1}], \quad (5)$$

where $\sin(\cdot)$ and $\cos(\cdot)$ are applied component-wise to (μ_x, μ_y, μ_z) .

We adopt SH degree $d=3$, so each Gaussian stores $1 + [(d+1)^2 - 1] = 16$ SH coefficient groups per color channel (one DC and 15 rest), and the decoder’s color head outputs $\Delta c^{\text{dec}} \in \mathbb{R}^{15 \times 3}$. The decoder is a lightweight MLP with ReLU activations in hidden layers and linear heads for both color and opacity; we do not apply output clipping or normalization to the heads. For stability, the opacity delta Δo is applied in logit space and then passed through $\sigma(\cdot)$ at render time (cf. Eq (7)), which empirically avoids saturation and preserves a well-conditioned gradient flow across lifecycle states.

The MLP uses a lightweight trunk and two output heads for color and opacity. Adding these deltas to static base attributes yields direction-aware wrinkles and hair sheen without per-Gaussian overfitting.

3) *Hybrid Appearance Synthesis*: At render time, a Base Gaussian’s appearance is its static stored attributes. For a detail Gaussian i , its final dynamic appearance is a composite of a static

base component ($c_i^{\text{base}}, \alpha_i^{\text{base}}$) and a dynamic delta predicted by the decoder. The final SH coefficients c'_i and per-Gaussian opacity α_i are computed as:

$$c'_i = c_i^{\text{base}} + \Delta c_i^{\text{dec}}, \quad (6)$$

$$\alpha_i = \sigma(\text{logit}(\alpha_i^{\text{base}}) + \Delta o_i). \quad (7)$$

This decomposition into a static base and a dynamic function enables the model to learn a more robust and generalizable prior for dynamic facial appearance. As a result, our method avoids overfitting and enhances the fidelity of dynamic effects.

B. Dynamic Gaussian Lifecycle Management

Standard 3DGS optimization employs *destructive pruning*, which results in information loss due to transient occlusions and leads to point cloud inflation, characterized by the uncontrolled accumulation of redundant Gaussians. This results in significant temporal artifacts and inefficiency. To address this, we introduce a Gaussian lifecycle management mechanism. It replaces destructive pruning with a *non-destructive cycle* where temporarily invisible Gaussians enter a *hibernation* state and are later *rejuvenated* when needed, ensuring a compact and robust representation. Additionally, we incorporate a *probabilistic selective detail pruning strategy* to address redundancy in detail Gaussians during training.

1) *Gaussian State Machine*: We replace the traditional binary “exist or delete” paradigm with a four-state machine, enabling a functional decoupling between a stable core and a dynamic detail layer.

- *Baby*: A newly densified Gaussian, corresponding to a *detail Gaussian*. It belongs to the dynamic detail layer, with optimizable geometry but neurally-synthesized appearance to model high-frequency dynamic effects (e.g., wrinkles, hair).
- *Mature*: A graduated Gaussian that has proven its stability, corresponding to a *Base Gaussian*. It forms the stable geometric core with permanently baked appearance attributes, ensuring identity preservation.
- *Decaying*: A Mature Gaussian whose opacity falls below a threshold τ_{opacity} . It enters a probationary state, buffering against transient invisibility before potential hibernation.
- *Hibernating*: A Decaying Gaussian that persists for N_{grace} iterations. It is moved to a *hibernation pool*, removed from active optimization, with its parameters preserved for future rejuvenation.

2) *State Transitions and Management*: The state transitions, governed by heuristic thresholds like τ_{opacity} , are discrete and thus non-differentiable. This meta-control determines which Gaussians are active for gradient updates without interrupting backpropagation to their optimizable parameters. This explicit control is fundamental, enforcing the functional separation of our Gaussians and preventing the point cloud inflation and information loss inherent to destructive pruning, as validated by our ablation study (Section V-D4), and is governed by the following transition rules.

Graduation (Baby $\rightarrow$ Mature): We define a precise, per-Gaussian graduation criterion. For each Baby Gaussian i , we maintain running statistics over iterations in which it is visible: an accumulator $A^{(i)}$ of the ℓ_2 norm of its view-space position gradients and a visibility counter $D^{(i)}$. At iteration t , if the Gaussian is visible in the current view, we update

$$A^{(i)} \leftarrow A^{(i)} + \|\nabla_{\text{view}} \mathbf{p}^{(i)}\|_2, \quad D^{(i)} \leftarrow D^{(i)} + 1, \quad (8)$$

where $\mathbf{p}^{(i)} = \pi(\boldsymbol{\mu}_i)$ denotes the 2D view-space projection of the Gaussian center and $\nabla_{\text{view}} \mathbf{p}^{(i)}$ is the (screen-space) gradient magnitude provided by the renderer for the two projected coordinates. The projection $\pi: \mathbb{R}^3 \rightarrow \mathbb{R}^2$ is the standard pinhole model with intrinsics $\mathbf{K}$ and extrinsics $(\mathbf{R}, \mathbf{t})$:

$$\mathbf{p} = \pi(\boldsymbol{\mu}) = \Pi(\mathbf{K}(\mathbf{R}\boldsymbol{\mu} + \mathbf{t})),$$

$$\Pi([\mathbf{u}, \mathbf{v}, \mathbf{w}]^\top) = (u/w, v/w), \quad (9)$$

and its Jacobian $\mathbf{J}_\pi(\boldsymbol{\mu}) = \partial\pi/\partial\boldsymbol{\mu}$ relates world- and screen-space differentials. In practice, we directly use the renderer’s backpropagated screen-space gradient for stability and efficiency. The average accumulated view-space position gradient is then

$$\bar{g}^{(i)} = \frac{A^{(i)}}{\max(1, D^{(i)})}. \quad (10)$$

We do not use a fixed-length time window; the statistics are maintained since creation or the last state reset and are implicitly normalized by the number of visible updates $D^{(i)}$. A Baby Gaussian graduates to Mature when $\bar{g}^{(i)} > \tau_{\text{grad}}$; we use $\tau_{\text{grad}} = 10^{-3}$ by default and require the condition to hold on the current check only (i.e., $K=1$ consecutive satisfaction). Upon graduation, the neurally predicted appearance deltas ($\Delta c^{\text{dec}}, \Delta o$) are permanently baked into the base attributes, and the detail-specific parameters are reset:

$$c^{\text{base}} \leftarrow c^{\text{base}} + \Delta c^{\text{dec}}, \quad (11)$$

$$\alpha^{\text{base}} \leftarrow \sigma(\text{logit}(\alpha^{\text{base}}) + \Delta o), \quad (12)$$

and the latent code z_i and persistent deltas for the graduated Gaussian are zeroed (detail layer disabled for that Gaussian).

Hibernation (Mature $\rightarrow$ Decaying $\rightarrow$ Hibernating): A Mature Gaussian enters the Decaying state if its opacity falls below τ_{opacity} . If it remains in this state for a grace period of N_{grace} iterations, it transits to Hibernating, and its parameters are moved to a hibernation pool.

Recovery (Decaying $\rightarrow$ Mature): If the opacity of a Decaying Gaussian rises above a recovery threshold τ_{recover} (where $\tau_{\text{recover}} > \tau_{\text{opacity}}$), it immediately reverts to the Mature state.

Rejuvenation (Hibernating $\rightarrow$ Mature): Rejuvenation reactivates Hibernating Gaussians in high-error regions to prevent holes and flicker. We compute an error mask, collect visible positions that project inside it, and revive nearby Hibernating candidates by proximity-limited selection (Algorithm 1).

3) *Error Map and Rejuvenation Details*: Error map is computed per training view on linear RGB as a per-pixel L_1 residual: $E = \|\hat{I} - I\|_1$ averaged over channels. An optional temporal EMA is maintained to stabilize activation: $\bar{E}_t = \beta \bar{E}_{t-1}$

Algorithm 1: Error-Driven Rejuvenation of Hibernating Gaussians.

Input: Rendered image $\hat{\mathbf{I}}$, Ground-truth image $\mathbf{I}$;
Hibernating Gaussians $\mathcal{G}_{\text{hib}}$; Visible active Gaussians' positions $\mathcal{P}_{\text{vis}}$; Thresholds τ_{err} , r_{search} , k_{max} .

Output: Rejuvenated Gaussians $\mathcal{G}_{\text{rejuv}}$.

```

1:  $\mathcal{G}_{\text{rejuv}} \leftarrow \emptyset$ 
2:  $\mathbf{M}_{\text{err}} \leftarrow (|\hat{\mathbf{I}} - \mathbf{I}| > \tau_{\text{err}})$ 
3:  $\mathcal{P}_{\text{search}} \leftarrow \{\mathbf{p} \in \mathcal{P}_{\text{vis}} \mid \pi(\mathbf{p}) \in \mathbf{M}_{\text{err}}\}$ 
4: if  $\mathcal{P}_{\text{search}}$  is not empty then
5:    $\mathcal{G}_{\text{cand}} \leftarrow \emptyset$ 
6:   for each Gaussian  $g_h$  with center  $\mu_h$  in  $\mathcal{G}_{\text{hib}}$  do
7:      $d_{\min}(g_h) \leftarrow \min_{\mathbf{p}_s \in \mathcal{P}_{\text{search}}} \|\mu_h - \mathbf{p}_s\|_2$ 
8:     if  $d_{\min}(g_h) < r_{\text{search}}$  then
9:        $s(g_h) \leftarrow \exp(-\frac{d_{\min}(g_h)^2}{2\sigma^2}) \cdot \text{mean\_pool}_{5 \times 5}(E|_{\pi(\mu_h)})$ 
10:      Add  $(g_h, s(g_h))$  to  $\mathcal{G}_{\text{cand}}$ 
11:     end if
12:   end for
13:   Sort  $\mathcal{G}_{\text{cand}}$  by score  $s(\cdot)$  in descending order (ties broken by smaller  $d_{\min}$ , then id).
14:    $\mathcal{G}_{\text{rejuv}} \leftarrow$  First  $k_{\text{max}}$  Gaussians from sorted  $\mathcal{G}_{\text{cand}}$ .
15: end if
16: return  $\mathcal{G}_{\text{rejuv}}$ 

```

+ $(1 - \beta) E_t$ with $\beta \in [0, 1)$ (we use $\beta=0.8$ by default). Thresholding is applied to the smoothed map: $\mathbb{I}[\bar{E}_t > \tau_{\text{err}}]$.

High-error region selection: We optionally apply a 3×3 max-pooling on the binary mask to bridge thin gaps and then take connected components; when the mask is dense, we uniformly subsample up to N_{regions} components per iteration (default 64) to bound computation.

Spatial search and nearest-neighbor strategy: For each visible active Gaussian center whose projection falls into the error mask, we query hibernating candidates within a 3D radius r_{search} by batched pairwise distances. Candidates are scored by a distance-decayed error: $s(g_h) = \exp(-\frac{d_{\min}^2}{2\sigma^2}) \cdot \text{mean_pool}_{5 \times 5}(E|_{\pi(\mu_h)})$ with $\sigma = \frac{r_{\text{search}}}{2}$. We sort by $s(g_h)$ (desc.) as a refinement over pure distance in Algorithm 1.

Activation budget per iteration: We enforce a strict budget k_{max} per iteration and an optional global cap k_{frame} per frame across all views in a batch (default $k_{\text{frame}} = k_{\text{max}}$ when single-view training).

Tie-breaking: If candidates share identical scores up to numerical tolerance, we resolve ties deterministically by (i) smaller $d_{\min}$, then (ii) smaller Gaussian index id.

Anti-oscillation protection: In addition to opacity hysteresis ($\tau_{\text{recover}} > \tau_{\text{opacity}}$), we use a cooldown counter N_{cool} (default 2K iterations) that prevents a Gaussian from toggling states immediately after a transition. Furthermore, a debounce window N_{debounce} (default 1 K) requires a candidate to satisfy the rejuvenation condition consecutively before activation, which mitigates flicker from transient noise.

4) **Densification Policy and Initialization:** When/where to insert new *Baby Gaussians*. We maintain three gradient-driven pathways: (i) **clone** high-gradient, small-scale Gaussians; (ii) **split** high-gradient, large-scale Gaussians; and (iii) **spawn detail** from eligible *Mature* parents when their average view-space gradient exceeds a detail-spawn threshold $\tau_{\text{grad_spawn}}$ and their current detail level is below the maximum L . We adopt $\tau_{\text{grad_spawn}} = 8 \times 10^{-4}$ and $L=3$ by default. Spawned children are positioned near the parent with a small Gaussian offset proportional to the parent's scale.

For each newly inserted *Baby Gaussian*, the center μ is initialized at the parent center plus a small random offset ($\sim \mathcal{N}(0, 0.01 s_{\text{parent}})$ in world space), the rotation quaternion q copies the parent's orientation, and the log-scale is set so that the absolute scale is $\frac{1}{4}$ of the parent's.

The SH DC, rest coefficients, and opacity are copied from the parent, with small Gaussian noise applied to the SH rest, while the persistent delta tensor and the latent code z_i are initialized to zeros. We use a 64-dimensional latent code by default ($\dim(z_i)=64$).

When a *Baby Gaussian* graduates, we bake its decoder deltas into the base SH/opacity and then reset both the persistent delta and the latent code to zero; i.e. the latent z_i is not retained for *Mature Gaussians*.

5) **Probabilistic Selective Detail Pruning:** To address the redundancy problem in *detail Gaussians* while maintaining reconstruction quality, we introduce a probabilistic selective pruning strategy that operates during training. Unlike traditional destructive pruning that permanently removes Gaussians and degrades quality, our approach temporarily hides detail Gaussians through *opacity modulation*, enabling efficient training without compromising fidelity.

Selective Application Rationale: Our pruning strategy is exclusively applied to *detail Gaussians*, not to the entire Gaussian population. This selective approach is crucial for maintaining reconstruction quality, particularly for critical facial structures, such as teeth. Teeth reconstruction primarily relies on *Mature Gaussians* that have been carefully optimized to capture the stable geometric structure. If these *Mature Gaussians* were accidentally pruned during training, the reconstruction of teeth would become increasingly difficult, as the model would lose the fundamental geometric foundation needed for accurate tooth representation. By restricting pruning to *detail Gaussians* only, we protect the stable *Mature Gaussian* core that preserves identity and critical structural information, while still addressing the redundancy problem in the dynamic detail layer.

During rendering, we apply probabilistic pruning exclusively to *detail Gaussians*. For each *detail Gaussian* i , we generate a random value $r_i \sim \mathcal{U}(0, 1)$ and compare it against a pruning probability p_t that varies with training iteration t :

$$\text{keep}_i = \begin{cases} 1 & \text{if } r_i > p_t \\ 0 & \text{if } r_i \leq p_t \end{cases}. \quad (13)$$

The pruning probability follows a linear annealing schedule:

$$p_t = p_{\text{start}} + (p_{\text{end}} - p_{\text{start}}) \cdot \min(1.0, t/T_{\text{anneal}}), \quad (14)$$

where $p_{\text{start}} = 0.0$, $p_{\text{end}} = 0.2$, and $T_{\text{anneal}} = 600,000$ iterations. Instead of permanently removing Gaussians, we modulate their per-Gaussian opacity during rendering:

$$\hat{\alpha}_i = \alpha_i \cdot \text{keep}_i, \quad (15)$$

where α_i is the per-Gaussian opacity (post-sigmoid) and $\hat{\alpha}_i$ is its pruning-modulated counterpart used for rendering. The final blending weight uses $\tilde{\alpha}_i = \hat{\alpha}_i \cdot \rho_i$. This non-destructive approach ensures that pruned Gaussians can contribute to rendering in subsequent iterations, maintaining the model’s capacity while introducing beneficial stochasticity.

During validation and testing, stochastic pruning is disabled to ensure deterministic evaluation.

C. Optimization Strategy

We train our framework with a multi-term objective function designed to enforce the functional decoupling between the stable core (*Mature Gaussians*) and the dynamic detail layer (*detail Gaussians*). The total loss $\mathcal{L}$ is:

$$\mathcal{L} = \mathcal{L}_{\text{render}} + \lambda_{\text{lpips}} \mathcal{L}_{\text{lpips}} + \lambda_{\text{embed}} \mathcal{L}_{\text{embed}} + \lambda_{\text{xyz}} \mathcal{L}_{\text{xyz}} + \lambda_{\text{scale}} \mathcal{L}_{\text{scale}} + \lambda_{\text{detail-delta}} \mathcal{L}_{\text{detail-delta}}, \quad (16)$$

where the weights are set to $\lambda_{\text{lpips}} = 0.1$, $\lambda_{\text{embed}} = 10^{-3}$, $\lambda_{\text{xyz}} = 10^{-2}$, $\lambda_{\text{scale}} = 1.0$, and $\lambda_{\text{detail-delta}} = 10^{-3}$.

The objective combines rendering losses with regularizers. The rendering loss includes a photometric term $\mathcal{L}_{\text{render}}$, which is a mix of L_1 and D-SSIM ($\lambda_{\text{ssim}} = 0.2$), and the perceptual loss LPIPS $\mathcal{L}_{\text{lpips}}$:

$$\mathcal{L}_{\text{render}} = (1 - \lambda_{\text{ssim}}) \mathcal{L}_1(\hat{\mathbf{I}}, \mathbf{I}) + \lambda_{\text{ssim}} \mathcal{L}_{\text{D-SSIM}}(\hat{\mathbf{I}}, \mathbf{I}), \quad (17)$$

$$\mathcal{L}_{\text{lpips}} = \text{LPIPS}(\hat{\mathbf{I}}, \mathbf{I}). \quad (18)$$

We introduce several regularization terms to constrain the model. Let $\mathcal{G}_{\text{detail}}$ denote the set of all *detail Gaussians*.

Neural Prior Regularization: To ensure the *Detail Decoder* learns a generalized appearance prior, we apply two regularizers exclusively to *detail Gaussians*. An L_2 penalty on the latent codes promotes a compact latent space:

$$\mathcal{L}_{\text{embed}} = \mathbb{E}_{i \in \mathcal{G}_{\text{detail}}} [\|\mathbf{z}_i\|_2]. \quad (19)$$

In these regularization terms, $\mathbb{E}$ denotes the expectation operator, which calculates the mean over the set of Gaussians specified in the subscript. To encourage sparse modifications, an L_1 penalty is applied to the decoder’s predicted appearance deltas, focusing its capacity on high-frequency details:

$$\mathcal{L}_{\text{detail-delta}} = \mathbb{E}_{i \in \mathcal{G}_{\text{detail}}} [\|\Delta \mathbf{c}_i^{\text{dec}}\|_1 + |\Delta o_i|]. \quad (20)$$

Gaussian Geometry Regularization: We define the parent-triangle local frame from the posed FLAME mesh at each timestep: for every face, we compute an orthonormal orientation matrix and a per-face anisotropic scale via per-face orientation computation; a Gaussian bound to face f stores its parameters $\mathbf{xyz}$ in this local frame. The world transform used for rendering is $\mathbf{x}^{\text{world}} = \mathbf{R}_f \mathbf{x}^{\text{local}} \odot \mathbf{s}_f + \mathbf{c}_f$, where $\odot$ denotes element-wise multiplication. Regularization is imposed in the local frame: we

penalize any Gaussian whose local center $\boldsymbol{\mu}_{l,i}$ or absolute scale $\exp(\mathbf{s}_{l,i})$ exceeds thresholds τ_{xyz} and τ_{scale} :

$$\mathcal{L}_{\text{xyz}} = \mathbb{E}_i [\text{ReLU}(\|\boldsymbol{\mu}_{l,i}\|_2 - \tau_{\text{xyz}})], \quad (21)$$

$$\mathcal{L}_{\text{scale}} = \mathbb{E}_i [\text{ReLU}(\|\exp(\mathbf{s}_{l,i})\|_2 - \tau_{\text{scale}})]. \quad (22)$$

Our binding keeps Gaussians anchored to the FLAME surface via the local frame above, but we allow *modest* local excursions under the default thresholds $\tau_{\text{xyz}}=0.55$ and $\tau_{\text{scale}}=0.6$, which suffice to capture thin structures (hairline, whiskers) without drifting.

V. EXPERIMENTS AND ANALYSIS

A. Setup

1) **Experimental Settings:** We systematically evaluate the quality of the generated head avatars under three experimental settings. 1) *Novel-View Synthesis*: Renders from held-out camera views using the head pose and expression parameters from the training sequence. 2) *Self-Reenactment*: Renders from all 16 camera views using unseen pose and expression parameters from the same subject’s test sequence. 3) *Cross-Identity Reenactment*: Drives the current subject’s model using motion parameters (pose and expression) from another subject. The source code is available at: <https://github.com/xgyutu/LivingAvatars>.

2) **Implementation Details:** For the single-view benchmark, training is limited to 60 K steps. All experiments run on a single NVIDIA RTX 4090 with PyTorch 2.0.1/CUDA 11.7.1. Following GaussianAvatars [12], we train for 600 k Adam [49] iterations, jointly fine-tune FLAME, start adaptive densification at 10 k iterations, and repeat it every 2 k iterations. The position learning rate decays from 5×10^{-3} to 5×10^{-5} ; scale, rotation, opacity, and SH use 1.7×10^{-2} , 1×10^{-3} , 0.05, and 2.5×10^{-3} , respectively. We disable periodic opacity resets and use probabilistic selective detail pruning, annealed from $p_{\text{start}}=0.0$ to $p_{\text{end}}=0.2$ over 600 k iterations. Unless otherwise specified, all results use one fixed-seed run per subject.

The detail decoder is a non-shared 2-layer MLP with 128 hidden units per detail level. It uses a 64-d latent code, 10 Fourier bands, and SH degree $d=3$, producing a 15-coefficient color residual per channel and an opacity-logit delta. We use $\tau_{\text{xyz}}=0.55$ and $\tau_{\text{scale}}=0.6$ for geometry regularization. Detail spawning uses $\tau_{\text{grad_spawn}}=8 \times 10^{-4}$ and initializes child scale to one quarter of the parent scale. Graduation uses $\tau_{\text{grad}}=10^{-3}$. Lifecycle management uses $\tau_{\text{opacity}}=0.025$, $\tau_{\text{recover}}=0.03$, $N_{\text{grace}}=15,000$, $\tau_{\text{err}}=0.5$, $\beta=0.8$, $r_{\text{search}}=0.5$ m, $k_{\text{max}}=50$, $N_{\text{cool}}=2,000$, and $N_{\text{debounce}}=1,000$.

B. Datasets and Metrics

We evaluate on the NeRsemble dataset [9], which contains 9 subjects captured from 16 views at 802×550 resolution. Following GaussianAvatars [12], we use 10 sequences per subject for training, one held-out sequence for Self-Reenactment, and one free-form sequence for Cross-Identity Reenactment. Images are downsampled to 802×550 , backgrounds are removed with Background Matting V2 [50], and pixels below the shoulders are masked using the tracked FLAME model to focus reconstruction

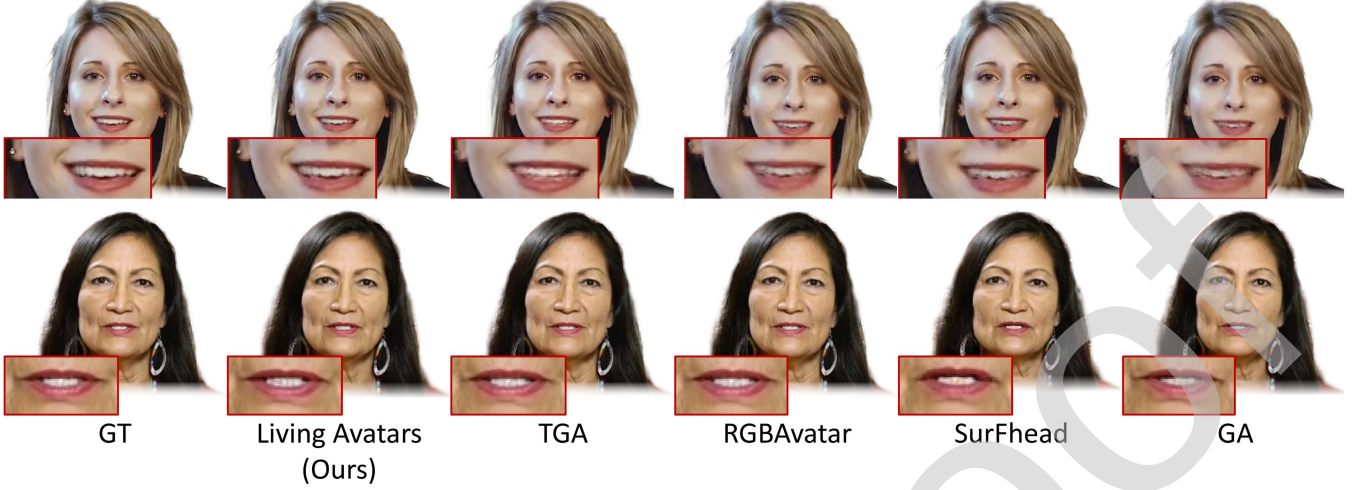


Fig. 4. Qualitative single-view reconstruction results on the HDTF-based benchmark. The models are trained on videos from HDTF, and we show representative reconstructions with zoomed-in mouth regions for visual comparison.

TABLE I
QUANTITATIVE COMPARISON WITH REPRESENTATIVE STATE-OF-THE-ART METHODS ON NOVEL-VIEW SYNTHESIS AND SELF-REENACTMENT. BEST RESULTS ARE IN BOLD AND SECOND-BEST RESULTS ARE UNDERLINED

Model	Novel-View Synthesis			Self-Reenactment		
	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
AvatarMAV	29.5	0.913	0.152	24.3	0.887	0.168
PointAvatar	25.8	0.893	0.097	23.4	0.884	0.102
INSTA	26.7	0.899	0.122	<u>26.3</u>	0.906	0.110
GA	31.0	<u>0.936</u>	<u>0.066</u>	25.8	0.910	<u>0.076</u>
GA-Editor	30.3	0.933	0.071	26.2	0.914	0.079
SurFHead	30.3	0.930	0.080	26.2	<u>0.912</u>	0.086
TGA	26.7	0.911	0.102	25.6	0.909	0.106
RGBAvatar	<u>31.7</u>	0.948	0.081	25.9	0.904	0.113
Ours	32.3	<u>0.936</u>	0.040	26.9	0.910	0.060

TABLE II
SINGLE-VIEW RECONSTRUCTION RESULTS ON HELD-OUT SUBJECTS. HIGHER IS BETTER FOR PSNR AND SSIM, WHILE LOWER IS BETTER FOR LPIPS

Model	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$
GA	22.0	0.918	0.064
SurFHead	27.0	0.949	<u>0.027</u>
TGA	<u>30.5</u>	<u>0.963</u>	0.024
RGBAvatar	28.4	0.953	0.030
Ours	31.7	0.967	0.029

on the head region. To further test reconstruction from sparse observations, we build a single-view benchmark using three videos selected from the dataset of [51]. These videos are processed with the same GaussianAvatars pipeline and downsampled to 767×431 resolution. We report PSNR, SSIM, and LPIPS for Novel-View Synthesis, Self-Reenactment, and the single-view benchmark.

C. Comparison With State-of-The-Art Methods

We compare LivingAvatars with representative baselines for Novel-View Synthesis and Self-Reenactment, covering both non-3DGS and recent 3DGS-based methods (Table I and Figs. 6 and 5). The results for INSTA [52], PointAvatar [53], and AvatarMAV [54] are taken from GaussianAvatars [12]. Existing methods often trade off fine detail and robustness under dynamic facial motion.

LivingAvatars gives the best overall trade-off. It achieves the highest PSNR and LPIPS on both tasks, while remaining

competitive in SSIM. This advantage comes from the combination of a non-shared Detail Decoder for high-frequency appearance and lifecycle control that avoids the flickering and point inflation caused by destructive deletion–recreation cycles. Fig. 6 shows the same trend for Novel-View Synthesis and Self-Reenactment. LivingAvatars reconstructs teeth with more plausible geometry and cleaner appearance, while the eyes retain natural specular highlights. It also preserves hair texture, facial wrinkles, and small accessories such as necklaces more clearly. In contrast, these details are blurred or barely visible in competing methods.

Among the non-3DGS baselines, INSTA, PointAvatar, and AvatarMAV lag behind the Gaussian-based methods, especially on detail-sensitive metrics. GaussianAvatars [12] introduced a dynamic 3D Gaussian representation bound to FLAME [14], but its static per-Gaussian appearance limits rich dynamic details. GaussianAvatar-Editor [46] improves robustness to motion occlusion and temporal inconsistency, yet at the cost of weaker texture detail. SurFHead [41] handles large deformations better, but tends to lose sharpness in other facial regions. TGA [36] and RGBAvatar [37] are more compact but still weaker on dynamic details. Overall, LivingAvatars better balances fidelity, detail, and transient-occlusion robustness.

Table II shows that LivingAvatars transfers well to the single-view setting, with the best PSNR and SSIM. Its LPIPS is slightly

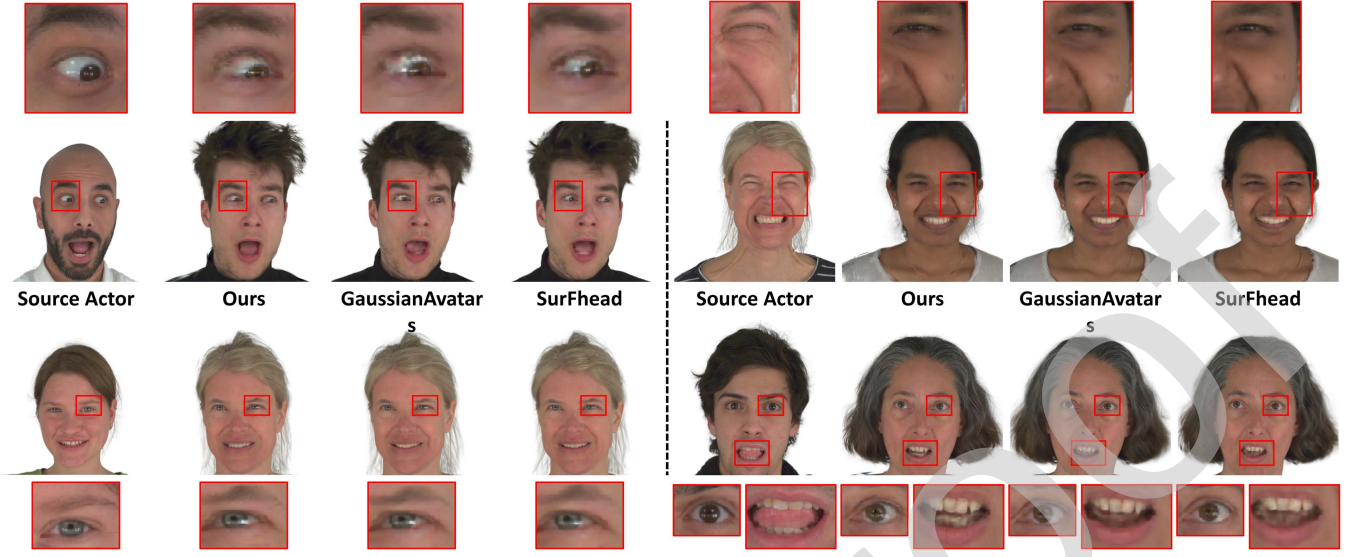


Fig. 5. Additional qualitative comparisons on harder cases with stronger pose and expression changes. LivingAvatars remains more faithful in the highlighted facial regions.

TABLE III
ABLATION ON CORE COMPONENTS AVERAGED OVER ALL 9 SUBJECTS. BEST RESULTS ARE IN BOLD AND SECOND-BEST RESULTS ARE UNDERLINED

Model	Novel-View Synthesis			Self-Reenactment			Num (K)↓
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	
Ours	32.3	0.936	0.040	26.9	0.910	0.060	77.0
w/o Detail System	29.9	0.925	0.049	26.0	0.908	0.060	67.5
w/o Lifecycle	<u>32.1</u>	0.936	0.040	26.9	0.910	0.060	89.4

behind TGA, but the gap is small, so the main gain lies in pixel accuracy and structural consistency. Fig. 4 confirms this on in-the-wild monocular HDTF videos: LivingAvatars reconstructs clearer teeth and better preserves the first-row subject’s ear accessory. These details are easy to lose under monocular supervision, showing stronger recovery of fine-scale identity cues. Fig. 5 further shows stable Cross-Identity Reenactment, where LivingAvatars preserves facial details more faithfully than GaussianAvatars and RGBAvatar across all three metrics.

D. Ablation Studies

1) *Ablation of Core Components*: We reran the ablation on all 9 subjects and report the averages in Table III. As shown in Fig. 7, removing either component core degrades the quality of reconstruction.

Qualitatively, Fig. 7 highlights where degradations occur. Without the Detail Decoder, hair strands collapse into blurry clumps, facial wrinkles become faint or disappear, and specular eye highlights lose sharpness and liveliness, yielding a flatter overall appearance. Without the Lifecycle mechanism, we also observe a mild yet consistent drop: subtle temporal flicker and softened eye/teeth details reduce perceptual crispness even when peak PSNR is similar.

We occasionally observe an increase in SSIM alongside a decrease in PSNR and LPIPS, typically due to the preference of

SSIM for smooth and well-aligned structures, even when fine high-frequency details, reflected more sensitively by PSNR and LPIPS, are lost. Therefore, we report all three metrics and emphasize LPIPS and qualitative results to assess high-frequency fidelity.

w/o Neural Dynamic Detail Synthesis: The average results for 9 subjects show that after removing the detail system, the PSNR and LPIPS metrics for Novel-View Synthesis deteriorate. This is due to the loss of high-frequency structural information such as wrinkles and hair strands. The number of Gaussians decreases, and this result reflects insufficient Gaussian density, leading to smoother but less faithful reconstruction outcomes. This also confirms that the Detail Decoder can influence the increase in the number of Gaussians, as more Gaussians are allocated to complex regions with rich details.

w/o Dynamic Gaussian Lifecycle Management: After replacing our proposed Lifecycle mechanism with the original 3D Gaussian management and pruning method, the average number of Gaussians increased from 77.0 K to 89.4 K. This is due to repeated optimization of the “deletion-reconstruction” cycle in regions prone to occlusion (e.g., teeth). The original 3D Gaussian management method struggles to optimize and manage Gaussians more efficiently. Although the peak PSNR remains at a similar level, both the perceptual quality and temporal stability significantly degrade, which is consistent with the visual artifacts (e.g. flicker around the eyes and teeth) shown

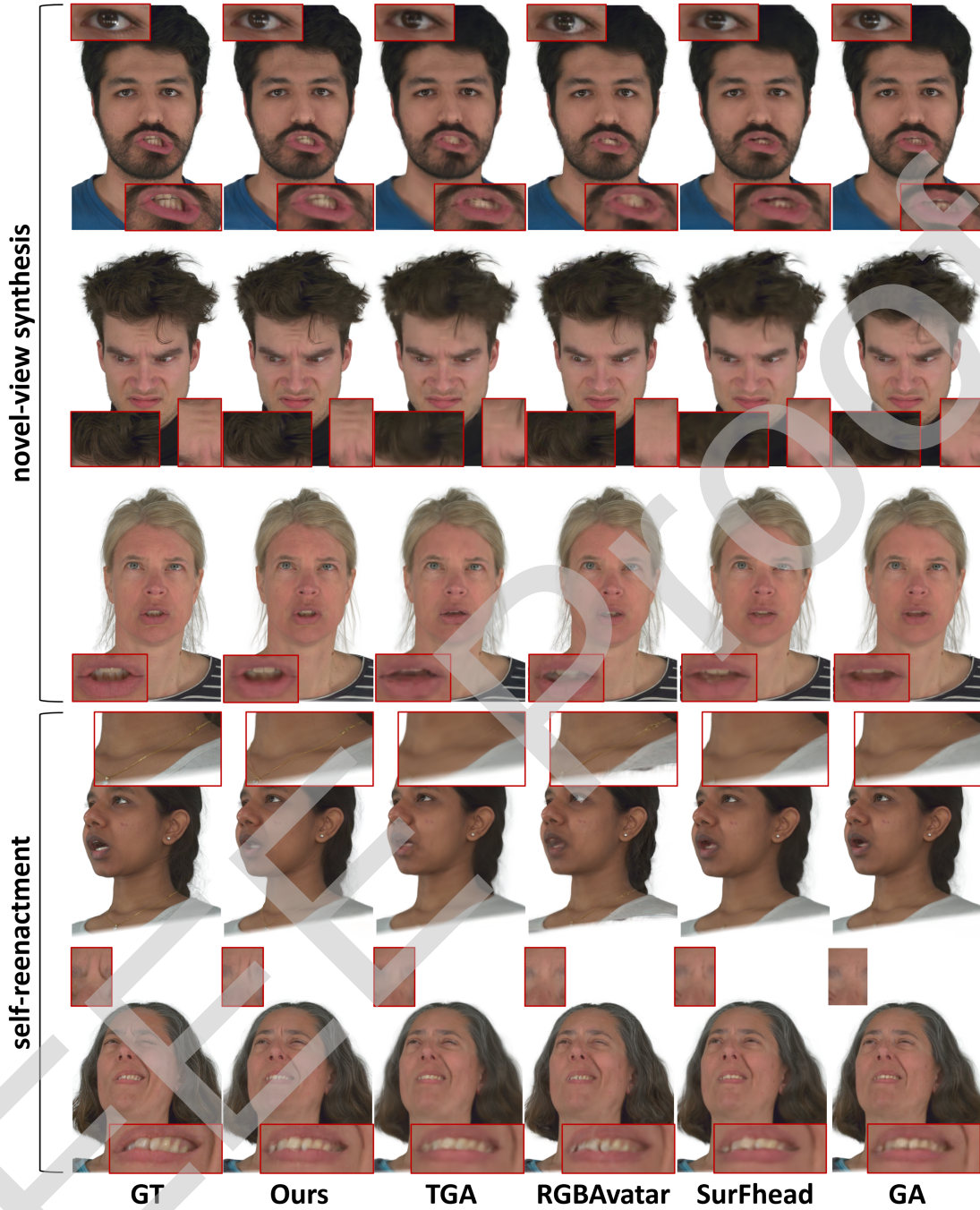


Fig. 6. Representative qualitative comparisons on held-out reconstructions. The red boxes mark zoomed-in regions, where LivingAvatars preserves sharper local detail than the baselines.

in Fig. 7. This result confirms the crucial role of the Lifecycle mechanism in maintaining the compactness and stability of the representation.

2) *Analysis of the Functionally-Decoupled Gaussian Representation*: As shown in Fig. 8, the two Gaussian populations play distinct roles. Base (Mature) Gaussians form a stable geometric core that preserves identity-defining structures, such as facial contours and organs. Detail (Baby) Gaussians capture dynamic, high-frequency signals, including hair and clothing geometry, while refining the base representation. This separation

between a stable identity foundation and a flexible detail layer supports our design.

We compare two ways of using the decoder’s predicted deltas: *Permanent baking*, which writes the deltas into the base SH/opacity at every iteration, and *Temporary fusion*, which applies the deltas during rendering and keeps the base attributes unchanged until graduation. Averaged over 9 subjects, temporary fusion outperforms permanent baking on both tasks (Table IV).

Permanent baking creates a moving target for the decoder. Because the deltas are folded into the static base at every step,

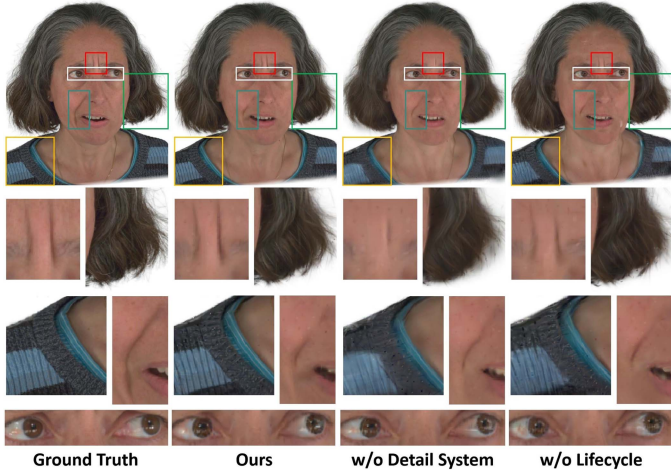


Fig. 7. Visualization results of the ablation study on core components.



Fig. 8. Visual analysis of functionally-decoupled Gaussian representation on partial subjects.

 TABLE IV
COMPARATIVE ANALYSIS OF PERMANENTLY BAKING DELTAS AND TEMPORARY FUSION WITH BAKING

Method	Novel-View Synthesis			Self-Reenactment		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
Permanent	29.1	0.916	0.067	25.4	0.898	0.078
Temporary	32.3	0.936	0.040	26.9	0.910	0.060

the residual changes continuously, gradients become less stable, and the base absorbs dynamic, view- and pose-dependent effects. The result is weaker decoupling, smoother textures, and less crisp direction-sensitive detail. Temporary fusion avoids this by keeping the residual stationary during optimization and limiting the dynamic effect to the detail layer; only stable contributions are baked into the core at graduation. This yields higher fidelity and better generalization to novel views and reenactment tasks.

3) *Decoder Design Analysis*: To validate our Detail Decoder, we ablated 9 subjects. As in Table V and Fig. 9, removing any core component degrades perceptual and accuracy metrics, confirming the design. Each component plays a distinct role.

Analysis of the Geometric Context: Removing xyz input leads to a significant performance drop, indicating that spatial coordinate information is crucial for detail synthesis. Without the

 TABLE V
COMPREHENSIVE ABLATION STUDY OF THE DETAIL DECODER ARCHITECTURE COMPONENTS

Method	Novel-View Synthesis			Gaussian Count	
	PSNR↑	SSIM↑	LPIPS↓	Detail (K)	Base (K)
Ours	32.3	0.936	0.040	15.8	61.2
w/o Embedding Input	31.9	0.933	0.043	16.7	62.9
w/o xyz Input	30.3	0.928	0.046	14.7	69.4
w/o Scale&Rot Input	31.3	0.931	0.045	16.7	56.6
w/o Opacity Output	31.3	0.931	0.045	21.1	61.7
w/o SH Output	32.0	0.934	0.043	11.7	60.7
w/o Non-shared Decoder	31.8	0.934	0.043	15.5	70.6

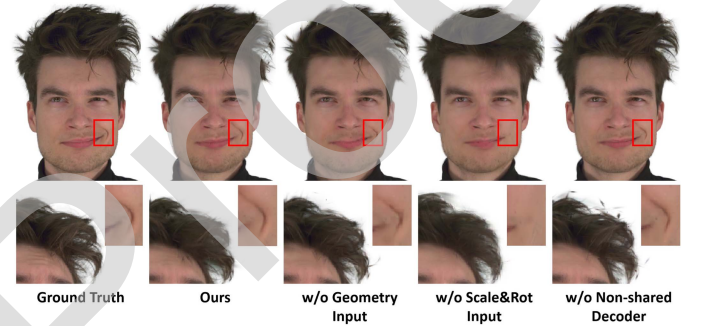


Fig. 9. Visualization of the ablation results for the Detail Decoder architecture.

geometric xyz input, the model cannot accurately align directional features, resulting in blurry structures such as beards and nasolabial folds. Furthermore, removing the scale and rotation input also causes a certain degree of performance degradation, but the impact is relatively smaller. This suggests that scale and rotation information helps the model better adapt to various geometric details and assists in the accurate restoration of facial textures at different scales and orientations, but its importance is not as high as the basic spatial coordinate information.

Analysis of the Learnable Latent Codes: The ablation confirms that the learnable latent codes z_i are indeed instance-specific features for each detail Gaussian. Removing embedding input results in measurable performance degradation, demonstrating that these codes provide crucial semantic information for detail synthesis. Meanwhile, without embeddings, the model compensates for lost spatial precision by generating more Gaussians.

Analysis of the Output Component: The two output components exhibit different efficiency patterns. Without opacity correction prediction, the model generates the largest number of detail Gaussians but achieves lower quality. Additionally, removing the SH correction prediction also leads to a decline in reconstruction performance.

Analysis of the Number of Detail Levels: We further evaluate the capacity of the detail layer by sweeping the maximum detail level L from 0 to 5, where $L=0$ disables both detail Gaussians and Detail Decoder. Table VI reports the averages for the 9 subjects. The results improve steadily from $L=0$ to

TABLE VI
COMPARATIVE ANALYSIS OF THE NUMBER OF DETAIL LEVELS

detail level	PSNR↑	SSIM↑	LPIPS↓
0	29.9	0.925	0.049
1	31.8	0.934	0.043
2	32.1	0.935	0.041
3	32.3	0.936	0.040
4	32.0	0.934	0.042
5	31.9	0.934	0.042

TABLE VII
ABLATION ON THE NUMBER OF FOURIER FREQUENCY BANDS IN THE POSITIONAL ENCODER OF THE DETAIL DECODER

Frequency Bands	PSNR↑	SSIM↑	LPIPS↓
Low (5 bands)	30.4	0.928	0.046
Mid (10 bands)	32.3	0.936	0.040
High (20 bands)	30.7	0.927	0.048

$L=3$, indicating that deeper detail hierarchies better capture high-frequency effects (wrinkles, hair sheen). Beyond $L=3$, metrics degrade slightly, suggesting diminishing returns and mild over-parameterization that introduces noisy fine structures and interferes with optimization. We therefore adopt $L=3$ as the default, striking a balance between fidelity and stability.

Analysis of the Positional encoding frequency: We conduct ablations to assess how the number of Fourier bands for the Detail Decoder’s positional encoding (see Table VII) affects performance. A moderate band count yields the best accuracy and perceptual quality: too few bands miss fine spatial details and cause over-smoothing; too many overfit noise and irrelevant patterns, producing artifacts and unstable training. We are using 10 bands, which strikes a good balance, reconstructing subtle wrinkles and skin textures without introducing noise, enabling the model to focus on semantically meaningful geometry while avoiding both underfitting and excessive sensitivity. Qualitatively, low-frequency encoding lacks spatial discriminability, blurring region-specific structures; high-frequency encoding is overly sensitive to noise, leading to mild overfitting and inconsistent fine textures; a moderate setting offers sufficient spatial expressivity and stable training.

Analysis of Probabilistic selective detail pruning: We ablate the pruning probability while keeping all other settings fixed. As shown in Table VIII, a probability of 0.2 yields the best trade-off, effectively suppressing redundant Gaussians with negligible quality loss. When the probability is 0.0 (no pruning), clear redundancy emerges; using the optimal 0.2 reduces the point count while preserving reconstruction fidelity. Moreover, probabilistic selective pruning encourages core information to migrate into base Gaussians, allowing detail Gaussians to concentrate on high-frequency, expression-driven effects. The additional five-seed experiment in Table IX shows very small variance across repeated runs, indicating that this stochastic pruning strategy does not introduce noticeable instability during training. The

TABLE VIII
COMPARATIVE ANALYSIS OF THE PROBABILITY IN PROBABILISTIC SELECTIVE DETAIL PRUNING

Probability	Novel-View Synthesis			Gaussian Count	
	PSNR↑	SSIM↑	LPIPS↓	Detail (K)	Base (K)
0.0	32.3	0.936	0.041	73.4	36.8
0.1	32.0	0.934	0.042	7.6	34.8
0.2	32.3	0.936	0.040	15.8	61.2
0.3	32.0	0.934	0.042	9.5	43.1
0.4	31.9	0.934	0.043	7.9	37.5
0.5	31.9	0.934	0.042	9.0	42.1

TABLE IX
STABILITY ANALYSIS ACROSS FIVE RANDOM SEEDS ON THE REPRESENTATIVE SUBJECT ID306. WE REPORT NOVEL-VIEW SYNTHESIS AND SELF-REENACTMENT RESULTS FOR REPEATED RUNS OF OUR METHOD, WHICH INCLUDES STOCHASTIC PROBABILISTIC SELECTIVE DETAIL PRUNING DURING TRAINING

Seed	Novel-View Synthesis			Self-Reenactment		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
1	34.4	0.967	0.023	31.2	0.957	0.025
2	34.5	0.967	0.023	31.3	0.957	0.025
3	34.5	0.966	0.024	31.2	0.957	0.026
4	34.6	0.967	0.023	31.0	0.957	0.025
5	34.6	0.967	0.022	31.1	0.957	0.025
Avg.	34.5	0.967	0.023	31.2	0.957	0.025

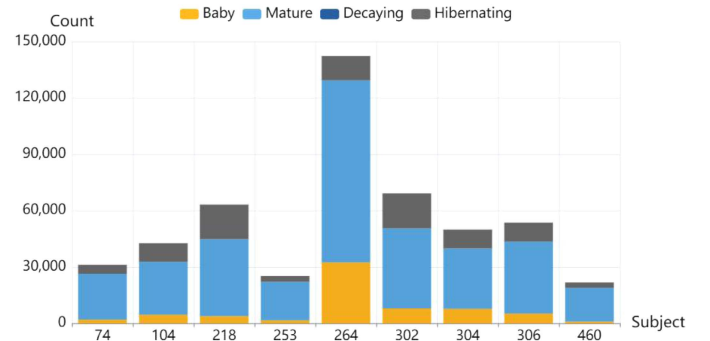


Fig. 10. Distribution of Gaussians across different lifecycle states for all subjects.

final reconstruction quality also remains highly consistent across seeds in both Novel-View Synthesis and Self-Reenactment.

4) *Analysis of the Gaussian Lifecycle Mechanism:* Table X shows that each lifecycle component matters: removing any one hurts either reconstruction quality or model compactness. The full design keeps the lowest Gaussian count while preserving quality, and its efficiency cost is reported separately in Section V-E.

Figs. 10 and 11 show the intended state flow. Baby Gaussians are abundant early, when the model inserts new points in high-gradient regions. Mature Gaussians then dominate and

TABLE X

ABLATION ON LIFECYCLE MECHANISMS AVERAGED OVER ALL 9 SUBJECTS. WE REPORT NOVEL-VIEW SYNTHESIS AND SELF-REENACTMENT TOGETHER WITH THE AVERAGE NUMBER OF GAUSSIANS (IN THOUSANDS, LOWER IS BETTER), TRAINING TIME, AND MEMORY CONSUMPTION. ALL VARIANTS ARE TRAINED FOR 600 K ITERATIONS ON A SINGLE NVIDIA RTX 4090 GPU. BEST RESULTS ARE IN BOLD AND SECOND-BEST RESULTS ARE UNDERLINED

Model	Novel-View Synthesis			Self-Reenactment			Num	Train	Mem
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	(K)↓	(h)↓	(GB)↓
Ours	32.3	0.936	0.040	26.9	0.910	0.060	77.0	13.6	4.46
w/o Graduation	31.5	0.934	0.044	26.8	0.909	0.062	<u>86.3</u>	13.1	4.50
w/o Decaying	32.3	0.935	0.040	26.9	0.909	<u>0.061</u>	94.3	12.7	4.47
w/o Hibernation	32.0	0.935	<u>0.041</u>	26.9	0.910	0.060	88.8	13.5	<u>4.50</u>
w/o Rejuvenation	32.0	0.934	0.042	26.8	0.910	<u>0.061</u>	89.4	13.7	4.48

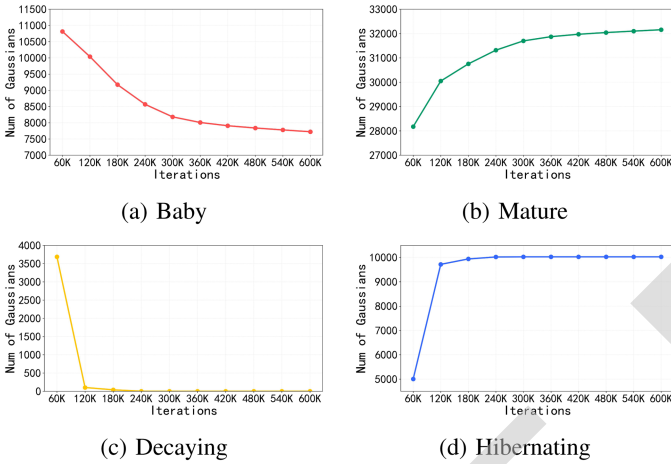


Fig. 11. The number of Gaussian points in each stage.

TABLE XI

TRAINING EFFICIENCY COMPARISON. ALL METHODS ARE TRAINED FOR 600 K ITERATIONS ON A SINGLE NVIDIA RTX 4090 GPU

Method	Training Time (h)	Memory (GB)	Points (K)
GA	9.3	1.2	61
SurFhead	14.6	20.0	437
TGA	8.7	2.3	76
RGBAvatar	4.3	0.7	83
Ours	13.6	1.7	90

form the stable core. Decaying Gaussians stay sparse because inactive points are moved into Hibernating, where they can be revived when local error rises. In the representative trajectory, Baby counts drop as they graduate, Mature counts plateau, and Hibernating counts stabilize. Together, these trends confirm the four-state graduate-hide-reactivate cycle and show that the model preserves recoverable detail without uncontrolled growth.

E. Efficiency Analysis

1) *Training Efficiency*: Table XI compares LivingAvatars with recent compact 3DGS head-avatar methods. RGBAvatar is the fastest and most memory-efficient baseline, TGA is

TABLE XII

INFERENCE EFFICIENCY COMPARISON AT 802×550 RESOLUTION. ALL METHODS ARE INFERENCE ON A SINGLE NVIDIA RTX 4090 GPU

Method	Total Points	Detail Points	Mem (MB)	FPS	Detail Ratio (%)
GaussianAvatars	89,021	-	2,008	71.83	-
Ours (online)	43,502	5,257	1,108	104.15	12.3
Ours (offline)	43,502	5,257	1,108	145.88	-

also compact, and SurFhead has the largest memory footprint and point count because of its surfel representation and Jacobian Blend Skinning. Our method trains more slowly than GA, TGA, and RGBAvatar because of Detail Decoder forward passes and lifecycle management, but it keeps memory low and delivers stronger fidelity on both Novel-View Synthesis and Self-Reenactment. Although it uses slightly more points than several compact baselines, the extra cost is offset by better reconstruction quality.

The lifecycle ablation in Table X shows that the training-time and memory overhead of the lifecycle components is modest, and the overall cost stays stable across variants. The full lifecycle system therefore gives the best quality-complexity trade-off.

2) *Inference Efficiency*: Table XII shows that LivingAvatars preserves competitive rendering speed at 802×550 . Online fusion adds minor runtime overhead from the Detail Decoder but improves fidelity; offline fusion removes this cost and increases FPS further. The smaller Gaussian set is the main source of the speedup and memory savings.

VI. LIMITATIONS AND DISCUSSIONS

Our framework has several limitations. 1) *Parametric Model Constraints*. Expression is constrained by the linear basis of the FLAME model, limiting geometric accuracy for extreme facial actions. A key direction is to explore more expressive non-linear or model-free representations. 2) *Hibernating Pool Scalability*. For long sequences, the Hibernating Pool can grow substantially, posing a challenge in scalability. Investigating efficient retrieval or forgetting mechanisms is necessary to balance consistency and resource usage.

VII. CONCLUSION

This study proposes a new paradigm for robust and efficient head avatar reconstruction, centered on two key innovations. First, the functionally decoupled Gaussian representation decomposes the Gaussian model into a stable geometric core (base Gaussians) and a dynamic detail layer (detail Gaussians). Detail Gaussians are endowed with learnable feature embeddings and utilize a Detail Decoder to refine Gaussian attributes. Second, a comprehensive Gaussian lifecycle management mechanism introduces novel states for Gaussian points and enables efficient management. In contrast to the destructive pruning behavior of conventional 3DGS, we replace it with a Hibernation-Rejuvenation cycle and a probabilistic selective detail pruning strategy. This leads to a significant improvement in the performance of dynamic head modeling. Extensive quantitative and qualitative experiments demonstrate that our framework achieves state-of-the-art performance on Novel-View Synthesis, Self-Reenactment, and Cross-Identity Reenactment tasks. We hope that this work will inspire further advances in efficient, high-fidelity dynamic avatar systems and their deployment in real-world scenarios.

REFERENCES

- [1] B. Mildenhall, P.P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," *Commun. ACM*, vol. 65, no. 1, pp. 99–106, 2021.
- [2] J. T. Barron, B. Mildenhall, M. Tancik, P. Hedman, R. Martin-Brualla, and P.P. Srinivasan, "Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 5855–5864.
- [3] A. Chen, Z. Xu, A. Geiger, J. Yu, and H. Su, "TensorRF: Tensorial radiance fields," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 333–350.
- [4] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–15, 2022.
- [5] T. Li et al., "Neural 3D video synthesis from multi-view video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 5521–5531.
- [6] K. Park et al., "NeRFies: Deformable neural radiance fields," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2021, pp. 5865–5874.
- [7] K. Park et al., "HyperNeRF: A higher-dimensional representation for topologically varying neural radiance fields," *ACM Trans. Graph.*, vol. 40, no. 6, Dec. 2021.
- [8] M. Isik et al., "HumanRF: High-fidelity neural radiance fields for humans in motion," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 1–12, 2023.
- [9] T. Kirschstein, S. Qian, S. Giebenhain, T. Walter, and M. Nießner, "NeRsemble: Multi-view radiance field reconstruction of human heads," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 1–14, 2023.
- [10] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 1–14, 2023.
- [11] B. Fei, J. Xu, R. Zhang, Q. Zhou, W. Yang, and Y. He, "3D Gaussian splatting as a new era: A survey," *IEEE Trans. Vis. Comput. Graph.*, vol. 31, no. 8, pp. 4429–4449, Aug. 2025.
- [12] S. Qian, T. Kirschstein, L. Schoneveld, D. Davoli, S. Giebenhain, and M. Nießner, "Gaussian Avatars: Photorealistic head avatars with rigged 3D Gaussians," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 20299–20309.
- [13] Y. Xu et al., "Gaussian head avatar: Ultra high-fidelity head avatar via dynamic Gaussians," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 1931–1941.
- [14] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero, "Learning a model of facial shape and expression from 4D scans," *ACM Trans. Graph.*, vol. 36, no. 6, pp. 194–1, 2017.
- [15] T. Wu et al., "Recent advances in 3D Gaussian splatting," *Comput. Vis. Media*, vol. 10, no. 4, pp. 613–642, 2024.
- [16] S.-S. Huang, S.-K. Zhang, S. Yang, J.-W. Guo, and H. Huang, "A survey of recent advances in generative 3D reconstruction," *J. Comput. Sci. Tech.*, vol. 40, no. 5, pp. 1236–1254, 2025.
- [17] C. Wang, H.-Y. Peng, Y.-T. Liu, J. Gu, and S.-M. Hu, "Diffusion models for 3D generation: A survey," *Comput. Vis. Media*, vol. 11, no. 1, pp. 1–28, 2025.
- [18] X.-K. Yan, S.-L. Mai, K.-W. Ji, Y.-C. Huo, and H.-J. Bao, "NRX: An end-to-end real-time neural rendering pipeline optimized for modern ai GPUs," *J. Comput. Sci. Tech.*
- [19] V. Blanz and T. Vetter, "Face recognition based on fitting a 3D morphable model," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 9, pp. 1063–1074, Sep. 2003.
- [20] S. Giebenhain, T. Kirschstein, M. Georgopoulos, M. Rünz, L. Agapito, and M. Nießner, "Learning neural parametric head models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 21003–21012.
- [21] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter, "A 3D face model for pose and illumination invariant face recognition," in *Proc. IEEE Int. Conf. Adv. Video Signal-Based Surveill.*, 2009, pp. 296–301.
- [22] Y. Feng, H. Feng, M. J. Black, and T. Bolkart, "Learning an animatable detailed 3D face model from in-the-wild images," *ACM Trans. Graph.*, vol. 40, no. 4, pp. 1–13, 2021.
- [23] R. Danecek, M. J. Black, and T. Bolkart, "EMOCA: Emotion driven monocular face capture and animation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 20311–20322.
- [24] W. Zielonka, T. Bolkart, and J. Thies, "Towards metrical reconstruction of human faces," in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 250–269.
- [25] Z. Bai et al., "Learning personalized high quality volumetric head avatars from monocular RGB videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 16890–16900.
- [26] G. Gafni, J. Thies, M. Zollhofer, and M. Nießner, "Dynamic neural radiance fields for monocular 4D facial avatar reconstruction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 8649–8658.
- [27] Y. Zheng, V. F. Abrevaya, M. C. Bühler, X. Chen, M. J. Black, and O. Hilliges, "IM Avatar: Implicit morphable head avatars from videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 13545–13555.
- [28] Z. Zhao, Z. Bao, Q. Li, G. Qiu, and K. Liu, "PSAvatar: A point-based shape model for real-time head avatar animation with 3D Gaussian splatting," 2024, *arXiv:2401.12900*.
- [29] Z. Li, Z. Zheng, L. Wang, and Y. Liu, "Animatable Gaussians: Learning pose-dependent Gaussian maps for high-fidelity human avatar modeling," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 19711–19722.
- [30] M. Kocabas, J.-H. R. Chang, J. Gabriel, O. Tuzel, and A. Ranjan, "HUGS: Human Gaussian splats," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 505–515.
- [31] J. Lei, Y. Wang, G. Pavlakos, L. Liu, and K. Daniilidis, "GART: Gaussian articulated template models," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 19876–19887.
- [32] S.-Y. Hu and Z. Liu, "GauHuman: Articulated Gaussian splatting from monocular Human videos," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 20418–20431.
- [33] Y. Zhi, W. Sun, J. Chang, C. Ye, W. Feng, and X. Han, "StruGauAvatar: Learning structured 3D Gaussians for animatable avatars from monocular videos," *IEEE Trans. Vis. Comput. Graph.*, vol. 31, no. 10, pp. 7820–7833, Oct. 2025.
- [34] J. Wang, J.-C. Xie, X. Li, F. Xu, C.-M. Pun, and H. Gao, "GaussianHead: High-fidelity head avatars with learnable Gaussian derivation," *IEEE Trans. Vis. Comput. Graph.*, vol. 31, no. 7, pp. 4141–4154, Jul. 2025.
- [35] Y. Zhan, T. Shao, H. Wang, Y. Yang, and K. Zhou, "Interactive rendering of relightable and animatable gaussian avatars," *IEEE Trans. Vis. Comput. Graph.*, vol. 31, no. 10, pp. 8491–8502, Oct. 2025.
- [36] Y. Wang et al., "3D Gaussian head avatars with expressive dynamic appearances by compact tensorial representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 21117–21126.
- [37] L. Li, Y. Li, Y. Weng, Y. Zheng, and K. Zhou, "RGBAvatar: Reduced Gaussian blendshapes for online modeling of head avatars," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 10747–10757.
- [38] J. Xiang, X. Gao, Y. Guo, and J. Zhang, "FlashAvatar: High-fidelity head avatar with efficient Gaussian embedding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 1802–1812.
- [39] Z. Shao et al., "SplattingAvatar: Realistic real-time human avatars with mesh-embedded Gaussian splatting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 1606–1616.
- [40] S. Ma, Y. Weng, T. Shao, and K. Zhou, "3D Gaussian blendshapes for head avatar animation," in *Proc. ACM Trans. Graph.*, 2024, pp. 1–10.

- [41] J. Lee et al., “SurFhead: Affine rig blending for geometrically accurate 2D Gaussian surfel head avatars,” in *Proc. Int. Conf. Learn. Representations*, 2025, pp. 7735–7759.
- [42] D. Zhang et al., “HRAvatar: High-quality and relightable Gaussian head avatar,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 26285–26296.
- [43] B. Dong and L. Zhang, “Talking head generation via viewpoint and lighting simulation based on global representation,” in *Proc. ACM Int. Conf. Multimedia*, 2025, pp. 10258–10267.
- [44] B. Ma, Y. Cao, and L. Zhang, “Decoupled two-stage talking head generation via Gaussian-landmark-based neural radiance fields,” *Comput. Vis. Med*, vol. 11, no. 4, pp. 799–816, Aug. 2025.
- [45] D. Zhang et al., “Guava: Generalizable upper body 3D Gaussian avatar,” in *Proc. IEEE Int. Conf. Comput. Vis.*, 2025, pp. 14205–14217.
- [46] X. Liu et al., “Gaussianavatar-editor: Photorealistic animatable gaussian head avatar editor,” in *Proc. Int. Conf. 3D Vis.*, 2025, pp. 1124–1133.
- [47] Z. Zhou, H. Li, H. Liu, N. Wang, G. Yu, and R. Ji, “STAR loss: Reducing semantic ambiguity in facial landmark detection,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 15475–15484.
- [48] S. Laine, J. Hellsten, T. Karras, Y. Seol, J. Lehtinen, and T. Aila, “Modular primitives for high-performance differentiable rendering,” *ACM Trans. Graph.*, vol. 39, no. 6, pp. 1–14, 2020.
- [49] D. P. Kingma, “Adam: A method for stochastic optimization,” 2014, *arXiv:1412.6980*.
- [50] S. Lin, A. Ryabtsev, S. Sengupta, B. L. Curless, S. M. Seitz, and I. Kemelmacher-Shlizerman, “Real-time high-resolution background matting,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 8762–8771.
- [51] Z. Zhang, L. Li, Y. Ding, and C. Fan, “Flow-guided one-shot talking face generation with a high-resolution audio-visual dataset,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 3661–3670.
- [52] W. Zielonka, T. Bolkart, and J. Thies, “Instant volumetric head avatars,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 4574–4584.
- [53] Y. Zheng, W. Yifan, G. Wetzstein, M. J. Black, and O. Hilliges, “PointAvatar: Deformable point-based head avatars from videos,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 21057–21067.
- [54] Y. Xu, L. Wang, X. Zhao, H. Zhang, and Y. Liu, “AvatarMAV: Fast 3D head avatar reconstruction using motion-aware neural voxels,” in *Proc. ACM Trans. Graph.*, 2023, pp. 1–10.



Xingyuan Lu received the bachelor's degree from the Tianjin University of Technology, in 2023. He is currently working toward the MS degree with the School of Computer Science and Technology, Beijing Jiaotong University. His research interests include 3D vision and deep learning.



the Grand PhD Scholarship of Saudi Arabian and the Excellent Student Prize of President Fellowships, Chinese Academy of Sciences in 2020 and 2021, respectively, and the Outstanding PhD Thesis Award of the Chinese Academy of Sciences in 2022.

Mingyang Zhao received the PhD degree from the Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science (AMSS), Chinese Academy of Sciences (CAS), in 2021. From 2023 to 2025, he was an assistant professor with the Centre for Artificial Intelligence and Robotics Hong Kong Institute of Science & Innovation, Hong Kong. He is currently an associate professor with AMSS, CAS, and also focuses on his research and development on probabilistic modeling and geometric computation. He was the recipient of



Chunyu Lin (Member, IEEE) received the Doctor degree from Beijing Jiaotong University (BJTU), Beijing, China, in 2011. From 2009 to 2010, he was a visiting researcher with ICT Group, Delft University of Technology, Netherlands. From 2011 to 2012, he was a postdoctoral researcher with the Multimedia Laboratory, Gent University, Belgium. He is currently a professor with BJTU. His research interests include image/video compression and robust transmission, 3D vision, virtual reality video processing, and ADAS.



Dong-Ming Yan (Member, IEEE) received the bachelor's and master's degrees from Tsinghua University, in 2002 and 2005, respectively, and the PhD degree from Hong Kong University, in 2010. He is currently a professor with the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include computer graphics, computer vision, geometric processing, and pattern recognition.