Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

UniResAvatar: Multi-Level One-shot Animatable Gaussian Avatars with Adjustable Quality--Cost Trade-offs

Zihao Zhang
Northeastern University
Wei Zhang
zhangwei1@mail.neu.edu.cn
Northeastern University
Weimin Lei
Northeastern University
Runbin Liu
Northeastern University
Xuan Zhang
Northeastern University

Research Article

Keywords:

Posted Date: November 27th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11744592/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

UniResAvatar: Multi-Level One-shot Animatable Gaussian Avatars with Adjustable Quality–Cost Trade-offs

Zihao Zhang[1], Wei Zhang[1*], Weimin Lei[1], Runbin Liu[1], Xuan Zhang[1]

[1]Northeastern University, Shenyang, China.

*Corresponding author(s). E-mail(s): zhangwei1@mail.neu.edu.cn;

Abstract

Existing methods for reconstructing Gaussian head avatars from a single image are typically trained with a fixed number of Gaussians, making it difficult for a single model to accommodate different requirements for reconstruction quality and construction cost. We present UniResAvatar, a shared multi-level model that reconstructs animatable Gaussian head avatars from a single image. Starting from a FLAME mesh, we construct geometric templates at different densities through subdivision and coarsening while preserving their corresponding deformation attributes. The vertices of the selected template are positionally encoded and combined with a level embedding to form query features. These features interact with image features in a shared Transformer to predict canonical Gaussian attributes. We obtain the image features by concatenating DINOv2 semantic features with reconstruction-oriented features from a variational autoencoder and applying nonlinear gated residual fusion. We train the model with joint multi-level supervision and cross-level consistency constraints. The resulting canonical avatar is animated using FLAME expression and pose parameters, and target images are synthesized through Gaussian rasterization. Experiments show that the shared model reconstructs avatars at different quality levels: higher-density configurations achieve higher reconstruction quality, whereas lower-density configurations substantially reduce canonical avatar construction time. The shared model thus allows the representation density to be chosen to suit different quality and cost requirements.

Keywords: Single-image 3D avatars, 3D Gaussian Splatting, multi-level representations, FLAME, quality–cost trade-off

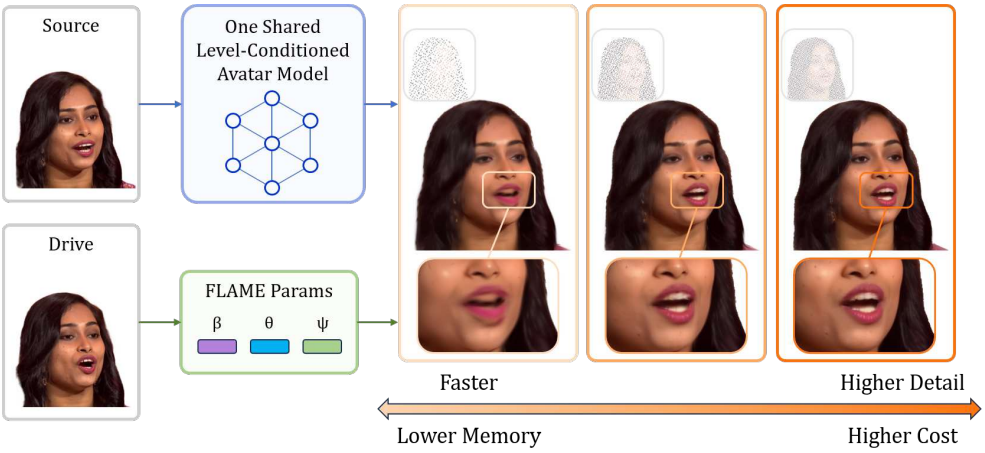


Fig. 1 One source image, multiple representation densities. The shared level-conditioned model generates an avatar at each selected level, with the source identity shape, driving motion, and camera held fixed across examples. The Gaussian visualizations and mouth close-ups show differences in representation and rendered detail across densities, illustrating the trade-off between reconstruction quality and avatar construction cost

1 Introduction

The reconstruction and animation of head avatars from images constitute a research area in computer graphics and computer vision, with applications in remote communication, digital content creation, and immersive interaction. This task requires recovering a person's appearance and head shape and synthesizing images with different facial expressions, head poses, and viewpoints. Different applications impose different requirements on the visual quality of synthesized images and the available computational resources. A variety of methods have been proposed for the reconstruction and animation of head avatars.

Two-dimensional head avatar generation methods learn appearance representations from images of a person and combine them with driving motion to generate target images. They use motion fields to guide the resampling of images or features, followed by a generative network that synthesizes images with the target expressions and head poses [1–4]. These methods can generate portrait animations from limited image observations, but maintaining consistent head geometry and identity remains challenging under large pose changes, occlusion, and the exposure of previously unseen regions.

Three-dimensional head avatar reconstruction provides a geometric basis for multi-view-consistent image synthesis and motion control. Neural radiance field avatars, exemplified by NeRFace, use continuous implicit appearance fields and volume rendering to synthesize images [5]. Such methods can produce high-quality images, but incur considerable training and rendering time. In contrast, 3D Gaussian Splatting (3DGS) uses explicit Gaussians and differentiable rasterization. Although explicit Gaussian representations generally require more storage than compact implicit appearance

fields, they enable efficient, high-quality image synthesis [6]. GaussianAvatars and SplattingAvatar associate Gaussians with deformable geometry to support controllable avatar animation [7, 8]. These person-specific avatar methods rely on multiple images or video frames to optimize a separate representation for each person. More recently, feed-forward methods, including Portrait4D, Portrait4D-v2, Real3D-Portrait, and LAM, have learned generalizable reconstruction models that directly predict avatars of unseen identities without per-identity optimization [9–12].

For example, LAM encodes the vertices of the parametric head model FLAME as query features, performs cross-attention with image features, and predicts a canonical Gaussian avatar in a single forward pass. It then animates the avatar using FLAME's linear blend skinning (LBS) and deformation corrections and generates images through Gaussian rasterization [9, 13]. However, the model is trained with a fixed number of Gaussians, making it difficult for a single model to meet reconstruction requirements across different density configurations.

We introduce UniResAvatar, a multi-level framework for one-shot animatable Gaussian avatars. Starting from a FLAME mesh template, we construct head avatar templates at different sampling densities through subdivision and coarsening. Deformation attributes are interpolated during subdivision and aggregated during coarsening, allowing the resulting templates to be controlled by the same set of identity shape, expression, and pose parameters. To combine head structure with local appearance, we integrate the semantic and spatial features of DINOv2 [14] with latent features learned for image reconstruction by a pretrained variational autoencoder (VAE) [15]. Channel-wise concatenation and nonlinear residual fusion combine the two types of features while preserving the semantic feature path. To accommodate different sampling densities within one generative model, we use level conditioning to identify the current geometric configuration and jointly learn to reconstruct avatars across levels. Cross-level consistency further aligns the appearance and foreground structure of the same person across densities. Shared prediction and unified animation incorporate density selection into a single model, allowing users to choose a representation according to reconstruction needs and construction cost. Figure 1 shows the effects of selecting different representation densities for the same source person and driving motion.

Our main contributions are as follows:

- We construct a multi-level FLAME–Gaussian representation that preserves deformation attributes across sampling densities, providing selectable geometric templates for avatar generation.
- We propose a shared, level-conditioned generative model that jointly learns multi-level Gaussian avatars with a single set of network parameters, performs one-shot reconstruction at the selected sampling density, and reuses a unified animation process.
- We construct image conditioning that combines semantics and appearance through nonlinear residual fusion of aligned semantic features and reconstruction-oriented latent features, providing information on both head structure and local appearance for multi-level Gaussian attribute prediction.

2 Related Work

2.1 2D Avatar Reconstruction

Two-dimensional avatar reconstruction represents a person's appearance using image features and combines it with driving motion to synthesize target frames. The source image provides the identity-specific appearance, while a driving image, video, or sequence of facial parameters specifies expressions and head poses. X2Face disentangles appearance and motion by learning source and driving representations [1]. The First Order Motion Model and the Thin-Plate Spline Motion Model estimate motion from learned keypoints and spatially resample source image features to synthesize target frames [2, 16]. Beyond keypoint-based motion estimation, PIRenderer encodes facial motion parameters into motion descriptors to guide feature warping and image editing [17], whereas DaGAN uses estimated depth to guide motion estimation and feature resampling [18]. Face-vid2vid introduces free-viewpoint control [3], and MegaPortraits extends synthesis to megapixel images [4]. These methods combine appearance recovery with controllable image generation. However, large pose changes, occlusion, and newly revealed regions still pose challenges for geometric and identity consistency. Three-dimensional representations provide a geometric basis for addressing these issues.

2.2 3D Avatar Reconstruction

Three-dimensional avatar reconstruction combines appearance, geometry, and deformation to support motion control and rendering from different viewpoints. Parametric models describe facial geometry and appearance using a compact set of parameters: the 3D Morphable Model (3DMM) statistically models facial shape and appearance [19], whereas FLAME combines identity shape, expression blendshapes, joint articulation, and pose-dependent corrections [13]. Image-based estimation methods such as DECA and EMOCA recover geometry and control parameters from portraits [20, 21]. Richer appearance representations include the dynamic radiance field of NeRFace, the implicit surface of IMAvatar, and the deformable points of PointAvatar; these methods reconstruct the geometry, appearance, and deformation of a specific person from video observations [5, 22, 23]. Feed-forward methods extend reconstruction to unseen identities: Portrait4D learns an animatable triplane representation from synthetic multi-view data [10]; Portrait4D-v2 is trained on pseudo-multi-view videos, enabling a reconstructed avatar of the same person to reproduce the driving frame's motion from different viewpoints [11]; and Real3D-Portrait combines image-to-triplane reconstruction with motion-conditioned animation [12]. For explicit representations, 3D Gaussian Splatting models geometry and appearance using anisotropic Gaussians and differentiable rasterization [6]. Animating Gaussians requires linking these representation elements to facial deformation. GaussianAvatars and SplattingAvatar use local triangle coordinates or surface embeddings on deformable meshes [7, 8], while FlashAvatar introduces learned offsets into a surface-embedded Gaussian field [24]. These methods optimize an avatar for each person from video observations. Gaussian-based methods also use feed-forward models to construct avatars from a single source image:

GAGAvatar predicts Gaussian attributes and offsets in both forward and backward directions from an image feature plane to place Gaussians in 3D space, and controls expressions through a separate branch based on parametric face vertices [25]; LAM uses features associated with FLAME vertices as cross-attention queries to interact with image features and predict canonical Gaussian attributes, and animates the avatar using blendshapes and linear blend skinning [9]. In this one-shot generation setting, representation density becomes a design choice for balancing an avatar's capacity to represent geometric and appearance details against its construction cost.

2.3 Multi-Level Methods

Level of Detail (LOD) balances visual quality and computational cost by adjusting representation complexity. Progressive meshes and quadric error simplification reduce geometric complexity, while subdivision refines surface representations [26–28]. Gaussian-based methods adjust complexity at different stages: LightGaussian compresses existing scene representations through pruning, appearance distillation, and quantization [29]; Octree-GS organizes Gaussians hierarchically and selects levels based on viewing conditions [30]. For head avatars, ArchitectHead learns a UV feature field for each person from monocular video and enables continuous control of Gaussian density through post-training resampling [31]. These methods illustrate ways to control density through compression, view-dependent level selection, or resampling of learned representations. UniResAvatar focuses on density selection during one-shot avatar construction: a FLAME template with deformation attributes is selected before prediction, and a shared model then generates the corresponding Gaussian avatar. The template's vertex count determines the number of queries processed during prediction and thus affects avatar construction cost; the number of generated Gaussians also affects the projection and color-blending workload during per-frame rendering.

3 Method

3.1 Problem Definition and Overview

Given a single source image I_s, UniResAvatar reconstructs an animatable Gaussian head avatar at a selected representation level $l \in \mathcal{L}$. Here, $\mathcal{L}$ is the configured set of density levels, and each level l corresponds to a FLAME template with N_l vertices. The source image I_s is preprocessed to estimate the identity shape $\boldsymbol{\beta}_s$, which provides stable identity-specific geometry for video generation. From frame t, denoted by I_t, of the driving video, we estimate expression parameters $\boldsymbol{\psi}_t$ and pose parameters $\boldsymbol{\theta}_t$ to control the target expression and head pose, respectively; the camera intrinsics $\mathbf{K}_t$ and extrinsics $\mathbf{T}_t$ determine the target viewpoint.

As shown in Figure 2, the image feature encoder first encodes the source image I_s into appearance conditioning F. Meanwhile, the FLAME template for the selected level l is combined with identity shape $\boldsymbol{\beta}_s$ to construct a set of canonical vertices $Q_l \in \mathbb{R}^{N_l \times 3}$. These vertices are encoded as geometric query features and combined with the corresponding level embedding. A shared Transformer fuses geometric queries with image features through joint attention, and pointwise attribute heads then predict

one Gaussian per vertex. Following the 3D Gaussian Splatting representation [6], each canonical Gaussian comprises a center $\boldsymbol{\mu}_i^l$, a rotation quaternion $\mathbf{q}_i^l$, standard deviations along its principal axes $\mathbf{s}_i^l$, opacity o_i^l, and RGB color $\mathbf{c}_i^l$:

$$\begin{aligned} \mathcal{G}_l &= \{(\boldsymbol{\mu}_i^l, \mathbf{q}_i^l, \mathbf{s}_i^l, o_i^l, \mathbf{c}_i^l)\}_{i=1}^{N_l}, \\ \boldsymbol{\mu}_i^l &= Q_{l,i} + \Delta\boldsymbol{\mu}_i^l. \end{aligned} \tag{1}$$

In Eq. (1), $\mathcal{G}_l$ is the set of canonical Gaussians at the selected level, and $Q_{l,i}$ denotes a vertex position in Q_l, indexed by i. The center offset $\Delta\boldsymbol{\mu}_i^l$ adjusts the spatial position of a template vertex to fit identity-specific geometry; color, opacity, and the other attributes jointly describe appearance. For a given source image and selected level, the model predicts $\mathcal{G}_l$ only once and reuses this set for subsequent frames through deformation and rendering.

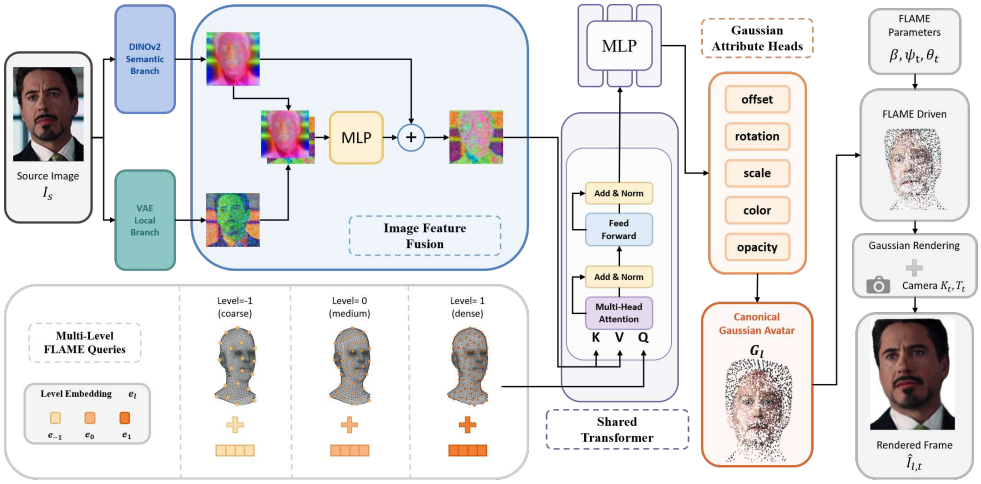


Fig. 2 Overview of UniResAvatar. DINOv2 and VAE features are aligned and concatenated, then fused through an MLP and a gated residual connection to form image conditioning. The canonical vertices of the selected FLAME template are encoded as queries and combined with a level embedding; a shared Transformer and attribute heads predict a canonical Gaussian head avatar. With the source identity shape held fixed, FLAME expression and pose parameters drive the Gaussian centers, and Gaussian rendering produces the target view

3.2 Multi-Level FLAME Geometric Priors

FLAME describes head geometry and deformation through identity shape, expression deformation bases, and a shared kinematic structure [13]. We use FLAME mesh vertices as geometric anchors for Gaussian centers, so the vertex count of the selected topology determines the number of Gaussians to predict. Starting from the base FLAME topology, subdivision and coarsening produce denser and sparser templates, respectively. Because these operations change the mesh vertex count and triangle connectivity, deformation bases and skinning weights must be associated with the new

vertices so that templates at different densities can still be animated using the same set of FLAME parameters.

To construct a denser template while preserving its animation attributes, midpoint subdivision retains the existing vertices, inserts a new vertex at the midpoint of each edge, and splits each triangle into four subtriangles. The deformation attributes of each new midpoint are the equally weighted average of the corresponding endpoint attributes. Coarse templates are constructed by spatial vertex clustering, with the clustering scale adjusted according to the target vertex count. The mean coordinates of the vertices in each spatial cluster define the merged coarse vertex, and faces are reconstructed according to the cluster mapping. Because coarsening merges multiple vertices into one coarse vertex, deformation attributes are aggregated by minimizing the reconstruction error obtained when coarse-level attributes are mapped back to the base mesh. Figure 3 illustrates these two template construction operations using coarsening and one step of midpoint subdivision of the base template; the sizes of the levels used in the experiments and their training settings are given in Section 4.1.

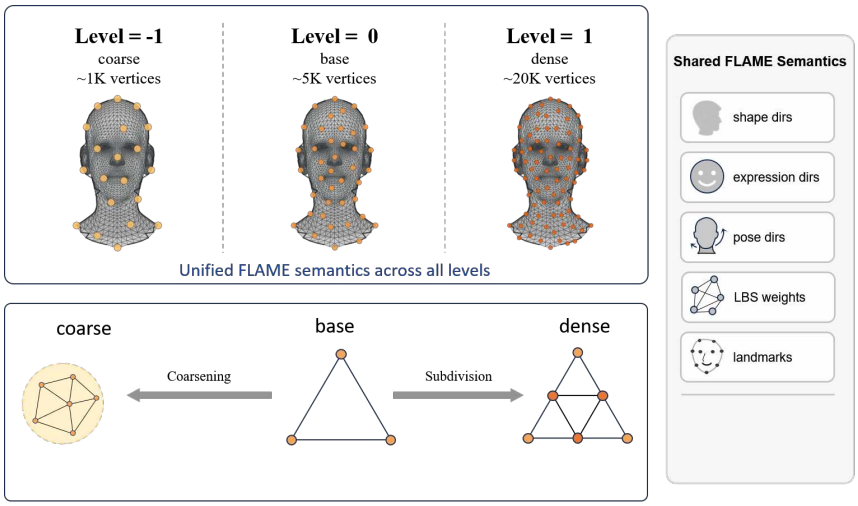


Fig. 3 Examples of multi-level FLAME template construction. The upper panel shows templates at different sampling densities, with sample points indicating relative density; the lower panel illustrates coarsening and one step of midpoint subdivision of the base mesh. Coarsening aggregates deformation attributes through a reconstruction mapping from the coarse level to the base level, whereas subdivision interpolates attributes to newly inserted edge midpoints. All levels retain the same FLAME parameterization and vertex attributes corresponding to their respective topologies

For a coarse template, we construct a sparse mapping $\mathbf{P}_l \in \mathbb{R}^{N_0 \times N_l}$ that reconstructs base vertices from coarse vertices, where N_0 is the number of base vertices. Each base vertex is associated with a nearby coarse triangle, and nonnegative barycentric weights are determined with respect to its three vertices. These weights sum to one and represent the interpolation contribution of each coarse vertex. Placing them in the corresponding row of $\mathbf{P}_l$ allows the coarse vertex coordinates to approximately reconstruct base vertex positions and interpolates coarse attributes back to the base

level. Let $\mathbf{X}_0$ be the per-vertex attribute matrix at the base level and $\mathbf{X}$ the coarse-level attribute matrix to be solved for; each row corresponds to a vertex, and each column to a deformation basis component or a joint skinning weight. The numbers of rows in the base- and coarse-level attribute matrices equal their respective vertex counts, while the number of columns is unchanged. We solve the following weighted least-squares problem for the optimal coarse-level attributes $\mathbf{X}_l^*$, so that interpolation back to the base level preserves the original deformation response as closely as possible:

$$\mathbf{X}_l^* = \arg\min_{\mathbf{X}} \left\| \mathbf{W}_0^{1/2} (\mathbf{P}_l \mathbf{X} - \mathbf{X}_0) \right\|_F^2 + \lambda_{agg} \|\mathbf{X}\|_F^2. \tag{2}$$

In Eq. (2), $\| \cdot \|_F$ denotes the Frobenius norm, whose square is the sum of squared matrix entries. The first term measures the difference between the coarse-level attributes interpolated back to the base level and the original attributes. Because vertex density varies across the base mesh, equally weighting errors at all vertices would give dense regions greater influence in fitting simply because they contain more vertices. We distribute each triangle's area equally among its three vertices and sum the contributions from adjacent triangles to obtain the surface area represented by each vertex. These areas define the diagonal entries of $\mathbf{W}_0$, weighting each region's error by its covered area to reduce the influence of sampling-density differences on coarse-level attribute estimation. The second term penalizes the squared norm of the coarse-level attribute matrix. Its regularization coefficient satisfies $\lambda_{agg} > 0$ and is used to suppress excessively large solutions and improve numerical stability. This least-squares procedure aggregates deformation bases and skinning weights separately. The resulting skinning weights are clamped to zero if negative and then normalized per vertex so that the joint weights sum to one.

To accommodate the reduced vertex count at the coarse level, we also construct a corresponding joint regression matrix that computes skeletal joint positions as weighted combinations of vertex coordinates. Let $\mathbf{J}_0$ and $\mathbf{J}_l$ denote the base- and coarse-level joint regression matrices, respectively. For the base and subdivided templates, joints are regressed on the base topology after adding identity-shape and target-expression displacements; for the coarse template, joints are regressed on its own topology, with the regression matrix satisfying $\mathbf{J}_l = \mathbf{J}_0 \mathbf{P}_l$. This composition is equivalent to reconstructing coarse geometry on the base topology and then regressing joints using the base-level matrix, allowing the coarse template to use the original skeletal animation mechanism.

After this construction, each level has geometry and animation attributes corresponding to its topology, while retaining the same parameter dimensions and kinematic joint tree. The model adds shape displacements generated from the identity-shape parameters to the template coordinates at each level, yielding the canonical vertex set $Q_l \in \mathbb{R}^{N_l \times 3}$ as geometric input for subsequent Gaussian generation. Additional density levels can be introduced by adjusting the coarsening target or subdivision depth, constructing the corresponding attributes, and training the associated level conditioning.

3.3 Image Conditioning with Semantics and Appearance

Source-image conditioning must describe a person's identity and head structure, as well as the appearance of facial features, hair, and boundaries. We therefore combine features from DINOv2 and a pretrained variational autoencoder (VAE). During pretraining, DINOv2 learns semantic and structural information by matching the visual representations of teacher and student networks under different augmentations and masking conditions [14]; the VAE compresses an image into latent variables and reconstructs the input through decoding, retaining appearance information needed for reconstruction in its latent features [15]. We fuse representations learned through these two pretraining tasks to form image conditioning that captures semantic structure and local appearance for Gaussian attribute prediction.

Both branches receive the same source-image crop. The DINO branch extracts patch features from multiple intermediate layers and integrates them into $F_D \in \mathbb{R}^{N_F \times d}$ through a trainable projection module within the branch, where N_F is the number of spatial tokens and d is the feature dimension. The VAE branch uses the encoder posterior mean to obtain deterministic latent features, which are aligned with the DINO patch grid through channel projection and bilinear resampling to obtain $F_V \in \mathbb{R}^{N_F \times d}$. The two feature sets therefore share spatial indices and channel dimensions and can be fused at corresponding locations. During training, the pretrained DINOv2 and VAE backbones remain frozen, while the branch projections and cross-branch fusion module are trained.

The aligned features from the two branches are concatenated along the channel dimension, allowing both representations at each spatial location to enter a multilayer perceptron (MLP) jointly. The MLP learns a joint transformation of the two feature types, and the resulting complementary features are added to the DINO features through a gated residual connection to form the final image conditioning:

$$F = F_D + \sigma(g)\,\mathrm{MLP}\left(\mathrm{LN}([F_D; F_V])\right). \tag{3}$$

Here, $[\,;\,]$ denotes channel concatenation, LN denotes layer normalization, and σ is the sigmoid function. The learnable scalar g is mapped through this function to obtain a gating coefficient that controls the contribution of the complementary features to the final image conditioning. To start fusion from the existing DINO conditioning, the weights and bias of the MLP's final linear layer are initialized to zero. At initialization, the residual is zero and the fusion output equals the DINO branch features; subsequent training learns the joint transformation of the two feature types and adjusts the residual gate strength.

3.4 Level-Conditioned Shared Generation and Animation

A pointwise positional encoding Φ maps the canonical vertices Q_l to geometric features. A learnable level embedding $e_l \in \mathbb{R}^d$ is then added to each vertex token to condition the shared generator on the selected level:

$$\begin{aligned} U_l &= \Phi(Q_l) + \mathbf{1}_{N_l} e_l^\top, \\ Z_l &= \mathcal{T}(U_l, F), \qquad \mathcal{G}_l = \mathcal{H}(Z_l, Q_l). \end{aligned} \tag{4}$$

Here, $U_l \in \mathbb{R}^{N_l \times d}$ is the query sequence, and $\mathbf{1}_{N_l}$ is an all-ones column vector with length equal to the number of query points, used to broadcast the level embedding; the superscript $\top$ denotes transposition. The shared Transformer $\mathcal{T}$ in Eq. (4) comprises joint attention and a token-wise feed-forward network. Its attention layers project geometric queries and image tokens into queries, keys, and values, then compute joint attention over the combined sequence, allowing geometric queries to interact with one another and exchange information with image features. The token-wise feed-forward network then applies linear mappings and nonlinear transformations at each position and, together with residual connections, updates the features to $Z_l \in \mathbb{R}^{N_l \times d}$ for attribute prediction.

The pointwise attribute heads $\mathcal{H}$ predict the Gaussian attributes defined in Eq. (1). To satisfy the requirements of Gaussian geometry and image synthesis, the center offset $\Delta\boldsymbol{\mu}_i^l$ is bounded within a preset range to limit how far a Gaussian center deviates from its template vertex; the quaternion $\mathbf{q}_i^l$ is normalized to define a valid rotation; and the scale $\mathbf{s}_i^l$ takes positive values with an upper bound to control the Gaussian's spatial extent. Opacity o_i^l and color $\mathbf{c}_i^l$ are mapped to $[0, 1]$ using sigmoid functions, consistent with the definitions of opacity and normalized color. The image encoder and fusion module, Transformer, and Gaussian attribute heads share parameters across levels; each level uses its corresponding template and level embedding. The vertex count N_l of the selected template determines the query sequence length. Changes in the number of query features alter the computational scale and activation memory requirements of attention and pointwise attribute prediction, while the parameters of the shared modules remain unchanged.

During animation, target-expression displacements from the selected level's expression deformation basis are added to canonical Gaussian centers. Per-vertex linear blend skinning (LBS) then produces their center positions in the target pose. Here, $\mathbf{B}_{E,i}^l(\boldsymbol{\psi}_t)$ denotes the displacement produced for vertex i by the selected level's expression deformation basis under the target expression. We use $\mathcal{D}_{l,i,t}$ to denote the skinning transformation of Gaussian i, which includes FLAME joint transformations weighted by the vertex's skinning weights and a global translation. Animation and rendering can be written as

$$\begin{aligned} \boldsymbol{\mu}_{i,t}^l &= \mathcal{D}_{l,i,t}\left(\boldsymbol{\mu}_i^l + \mathbf{B}_{E,i}^l(\boldsymbol{\psi}_t)\right), \\ (\widehat{I}_{l,t}, \widehat{M}_{l,t}) &= \mathcal{R}(\mathcal{G}_{l,t}; \mathbf{K}_t, \mathbf{T}_t). \end{aligned} \tag{5}$$

In Eq. (5), $\boldsymbol{\mu}_{i,t}^l$ denotes the deformed center of Gaussian i. Replacing the canonical centers with the deformed centers while keeping all other predicted attributes unchanged yields the posed Gaussian set $\mathcal{G}_{l,t}$. The Gaussian rendering operator $\mathcal{R}$ projects and composites this set using the target camera parameters $\mathbf{K}_t$ and $\mathbf{T}_t$, producing the RGB image $\widehat{I}_{l,t}$ and soft foreground mask $\widehat{M}_{l,t}$. All levels use the same deformation and rendering process.

3.5 Joint Multi-Level Optimization

We train the shared model using a source frame I_s and target frame I_t from the same video sequence. Each training level generates $\widehat{I}_{l,t}$ under the same motion and camera

conditions, with supervision provided by I_t and the foreground mask M_t. Its supervised objective $\mathcal{L}^l_{\text{sup}}$ combines image and foreground-coverage supervision, a geometric offset constraint, and full-image and local appearance supervision. An RGB L_1 loss normalized by foreground area supervises color, while a full-image mask L_1 loss supervises foreground coverage. The mean squared center offset constrains the deviation of Gaussian centers from template vertices. Full-image LPIPS [32] supervises perceptual similarity, and image-gradient differences within the target foreground constrain local detail. We also define a common crop using the target foreground bounding box and an extended margin, crop and resize the prediction, target, and mask together, and compute local gradient and perceptual losses. These seven loss terms are weighted by coefficients specified for each training stage.

To coordinate predictions across densities, we use the current prediction from a high-density training level r as a consistency reference to constrain lower levels. In the following, we use the rendering outputs from Eq. (5) and omit the frame subscript. For a student level l, the consistency loss $\updownarrow^l_{cons}$ compares RGB values within the reference prediction's foreground, full-image masks, and features obtained by re-encoding rendered outputs:

$$\begin{aligned}\updownarrow^l_{cons} =& w_r \updownarrow_{rgb} \left(\widehat{I}_l, \mathrm{sg}(\widehat{I}_r); \mathrm{sg}(\widehat{M}_r)\right) \\ &+ w_m \updownarrow_{mask} \left(\widehat{M}_l, \mathrm{sg}(\widehat{M}_r)\right) \\ &+ w_f \left(1 - v_l^\top \mathrm{sg}(v_r)\right).\end{aligned} \tag{6}$$

Here, sg denotes stop-gradient, $\updownarrow_{rgb}$ is the foreground-normalized absolute difference, and $\updownarrow_{mask}$ is the mean absolute mask error. The nonnegative weights w_r, w_m, and w_f balance RGB, mask, and feature consistency. The normalized vectors v_l and v_r are obtained by re-encoding the rendered images with the shared image encoder, followed by feature pooling and L_2 normalization. The feature term measures the cosine distance between these normalized vectors. The consistency loss stops gradients only through the reference path, while retaining backpropagation through the low-density path; all training levels, including the reference level, still receive supervision from the ground-truth target. This one-way constraint works together with each level's reconstruction objective to encourage low-density outputs to retain the appearance and foreground structure of denser outputs.

Density levels are introduced in stages. We first adapt image fusion and the shared predictor at the high-density reference level, then progressively add lower-density levels for joint optimization on the same batches. Let $\mathcal{L}_k$ denote the set of levels participating in training at stage k, including the reference level r; $\mathcal{S}_k = \mathcal{L}_k \setminus \{r\}$ contains the remaining lower-density levels after excluding the reference level from the stage's training set, and consistency constraints are applied to these levels. For each batch, the model performs forward passes at every level participating in the current stage, and all loss terms contribute to a single update of the shared parameters:

$$\mathcal{L}^k = \sum_{l \in \mathcal{L}_k} \omega_l^k \mathcal{L}_{\text{sup}}^l + \gamma_k \mathcal{C}^k,$$
$$\mathcal{C}^k = \frac{1}{|\mathcal{S}_k|} \sum_{l \in \mathcal{S}_k} \updownarrow_{cons}^l. \tag{7}$$

In Eq. (7), $\mathcal{L}^k$ is the joint objective for the current stage, $\mathcal{C}^k$ is the mean consistency loss over the levels in $\mathcal{S}_k$, and $|\mathcal{S}_k|$ is the number of levels in this set. When only the reference level participates in training, $\mathcal{S}_k$ is empty and $\mathcal{C}^k$ is defined as zero. The coefficients ω_l^k and γ_k control each level's supervised contribution and the consistency strength, respectively, together balancing the loss terms in a joint update on the same batch. We initialize the existing DINO feature projection module, vertex encoding module, Transformer, and Gaussian attribute heads using the pretrained parameters of the official single-level LAM model [9]. The new VAE projection and fusion modules and level embeddings are initialized separately and then trained jointly with the shared predictor across levels.

4 Experiments

We first evaluate the trade-off between reconstruction quality and representation cost across three levels of a shared model. We then compare our method with existing methods in same-frame reconstruction and cross-frame animation, followed by ablation studies of image conditioning and multi-level training.

4.1 Experimental Settings

Data and training.

We constructed the training dataset from consecutive frames of interview videos in VFHQ [33] by extracting foreground masks and estimating FLAME shape, expression, pose, and camera parameters. The processed dataset contains 14,900 complete clips, with 13,410 for training and 1,490 for validation. Each sample comprises eight frames drawn from the same clip. One frame serves as the source image I_s, while the remaining frames are candidate driving frames that provide target motion, camera parameters, and corresponding image supervision (Table 2).

We initialized the existing modules with the official single-level LAM model weights [9] and performed staged joint training as described in Section 3.5: we first optimized $l = 1$, then progressively introduced $l = 0$ and $l = -1$. The DINOv2 and VAE backbones remained frozen, while the image feature projection and fusion modules, vertex encoding module, level embeddings, Transformer, and Gaussian attribute heads were trained. Table 3 summarizes the main optimization settings. After 4,000 warm-up updates, the learning rate followed a cosine decay schedule. Table 4 provides the complete stage configuration.

We trained and evaluated three representative levels, $l = -1, 0, 1$, covering coarse, base, and denser representations. Their topologies and sizes are listed in Table 1.

Table 4 lists the stage configurations for 18 epochs of joint training. Within each minibatch, all currently active levels are processed sequentially, and their level weights

Table 1 Discrete FLAME–Gaussian representation levels.

Level	Topology source	Vertices	Faces
-1	Coarsening of the base mesh and attribute aggregation	1256	2638
0	Base FLAME head topology	5023	9976
1	One step of midpoint subdivision of the base mesh	20018	39904

Table 2 VFHQ-derived training dataset and frame sampling settings.

Item	Value
Complete training-ready clips	14,900
Training / validation clips	13,410 / 1,490
Input and supervision resolution	512×512
Sampled frames per sample	8

Table 3 Main settings for staged multi-level joint training.

Configuration	Setting
Active levels	$-1/0/1$, introduced progressively by stage
Optimizer	AdamW, weight decay 0.05
Learning rate	1.5×10^{-4}, cosine decay after warm-up
Mixed precision and gradient clipping	bf16, maximum gradient norm 0.8
Data parallelism	3 L40 GPUs, minibatch size 2 per GPU
Training duration	18 epochs, 40,230 updates

determine the relative contributions of the supervised losses. The seven supervised loss weights correspond, in order, to foreground RGB, foreground mask, center offset regularization, full-image perceptual, foreground gradient, local-crop gradient, and local-crop perceptual losses.

The level embedding for $l = 1$ is initialized to a zero vector as an anchor. The initialization standard deviations for $l = 0$ and $l = -1$ are 0.008 and 0.016, respectively. All three embedding vectors remain trainable.

Evaluation protocol.

We evaluated on 50 clips from VFHQ-Test. For same-frame reconstruction, eight frames were selected from each clip, with each frame serving as both source and target. For same-identity cross-frame animation, one source image was fixed and eight other frames were selected as targets. Each task thus contains 400 pairs. Our method obtains both identity appearance conditioning and FLAME identity shape solely from the source image I_s; the driving frame I_t provides expression, pose, and camera parameters. External methods receive the corresponding information through their official interfaces. We compared with the public models of Real3D-Portrait [12], Portrait4D

Table 4 Stage configurations for 18 epochs of multi-level joint training.

Stage	Epochs	Active levels	Level weights	Supervised loss weights	Consistency coefficient
A: Anchor stabilization	2	1	1.00	$(1, 1, 0.04, 0, 0, 0, 0)$	0
B: Local branch warm-up	2	1	1.00	$(1, 1, 0.05, 0.08, 0.03, 0.05, 0.02)$	0
C: Base-level alignment	3	$0, 1$	$0.42, 0.58$	$(1, 1, 0.08, 0.15, 0.05, 0.08, 0.03)$	0.75
D: Shared three-level optimization	5	$-1, 0, 1$	$0.23, 0.30, 0.47$	$(1, 1, 0.10, 0.30, 0.08, 0.12, 0.05)$	1.00
E: Three-level detail enhancement	4	$-1, 0, 1$	$0.22, 0.30, 0.48$	$(1, 1, 0.10, 0.42, 0.13, 0.22, 0.10)$	1.15
F: Pixel-quality rebalancing	2	$-1, 0, 1$	$0.23, 0.30, 0.47$	$(1.2, 1, 0.10, 0.25, 0.08, 0.10, 0.04)$	1.00

[10], Portrait4D-v2 [11], and LAM. All methods share the same source and target frame lists while retaining their required preprocessing and motion interfaces. Outputs are standardized to 512×512 images on a white background. Quality comparisons use a common set of valid pairs. For same-frame reconstruction, one pair lacked valid face analysis, leaving 399 pairs across all 50 clips; cross-frame animation uses all 400 pairs. Failed pairs are excluded from the common set rather than assigned zero scores.

Quality metrics.

PSNR, SSIM [34], and LPIPS [32] measure pixel fidelity, structural similarity, and perceptual distance, respectively. In the quality–cost comparison and ablation studies, PSNR and SSIM are computed over the foreground region of the original evaluation canvas, while LPIPS is computed over the full frame on a white background. For comparisons with external methods, predictions and ground truth are first normalized to a common face coordinate system using a shared rule. PSNR and SSIM are then computed over the face foreground, and LPIPS over the full normalized face canvas. Within each table, all methods use the same evaluation region and sample set. CSIM measures the cosine similarity between the identity features of the output and the source image I_s. AED and APD compare the expression and pose coefficients, respectively, estimated by a shared face analyzer from the output and target frame I_t, using the L_2 distance. AKD measures their landmark distance normalized by the interocular distance. Frame-level scores are first averaged within each clip, and the 50 clip scores are then averaged with equal weights.

Resource measurement.

Resource measurements for our model cover canonical avatar construction, including source image encoding, query construction, the Transformer, and Gaussian

Table 5 Quality and construction cost of three levels of a shared model. Quality metrics are computed over 400 cross-frame target frames from 50 clips, and construction time is aggregated over the 50 source images. Time and peak GPU memory refer to canonical avatar construction.

Level	Gaussians	PSNR ↑	SSIM ↑	LPIPS ↓	Construction time (ms)↓	Memory (MB) ↓
$l = -1$	1256	22.2987	0.7114	0.1943	**131.122**	**3117.4**
$l = 0$	5023	22.7482	0.7334	0.1784	270.612	3155.8
$l = 1$	20018	**22.8933**	**0.7462**	**0.1528**	1565.106	3713.9



Fig. 4 Local reconstruction results at three levels with identical target motion, camera, and crop regions. The examples show differences around hats, eyeglass frames, beards, and mouth boundaries

attribute prediction. We report the Gaussian count, construction time, and peak GPU memory. Measurements were performed on an NVIDIA L40 at 512×512 resolution in fp32, with one warm-up run followed by three CUDA-synchronized repetitions. Timing excludes model loading, image preprocessing, metric networks, and output writing.

4.2 Multi-Level Quality–Cost Trade-off

To examine the effect of representation density, we vary only the template level while fixing the model weights, source image, identity shape estimated from the source image, target motion, camera, and crop. Table 5 summarizes reconstruction quality over 400 cross-frame targets from 50 clips and the corresponding avatar construction cost. All three levels use the same trained model.

Table 6 Single-image same-frame reconstruction on VFHQ-Test using 399 common valid pairs from 50 clips. Bold and underlined values indicate the best and second-best results, respectively.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	CSIM ↑	AED ↓	APD ↓	AKD ↓
Real3D-Portrait	22.5842	0.7328	0.2282	**0.9225**	**0.8782**	0.0359	0.0582
Portrait4D	19.8923	0.6105	0.3033	0.8684	1.0634	0.0695	0.0813
Portrait4D-v2	20.6307	0.6375	0.2305	0.8983	<u>0.9477</u>	0.0663	0.0727
LAM	19.9091	0.6344	0.2572	0.8544	1.5855	0.0552	0.0787
UniResAvatar ($l = -1$)	22.4516	0.7099	0.2333	0.8786	1.2448	0.0246	0.0603
UniResAvatar ($l = 0$)	<u>23.2833</u>	<u>0.7460</u>	<u>0.2081</u>	0.9127	0.9640	**0.0211**	<u>0.0509</u>
UniResAvatar ($l = 1$)	**23.3089**	**0.7501**	**0.1895**	<u>0.9205</u>	0.9610	<u>0.0214</u>	**0.0506**

Table 7 Single-image same-identity cross-frame animation on VFHQ-Test using 400 pairs from 50 clips. Bold and underlined values indicate the best and second-best results, respectively.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	CSIM ↑	AED ↓	APD ↓	AKD ↓
Real3D-Portrait	18.6267	0.5685	0.2979	**0.7720**	<u>1.5862</u>	0.1140	0.1308
Portrait4D	18.3682	0.5531	0.3369	0.7072	1.6485	0.0716	0.0973
Portrait4D-v2	<u>19.3144</u>	0.5923	<u>0.2698</u>	0.7287	**1.4109**	0.0682	<u>0.0833</u>
LAM	19.0449	<u>0.5969</u>	0.2821	0.7096	1.8526	<u>0.0601</u>	0.0878
UniResAvatar ($l = 1$)	**20.4152**	**0.6555**	**0.2474**	<u>0.7500</u>	1.6064	**0.0355**	**0.0713**

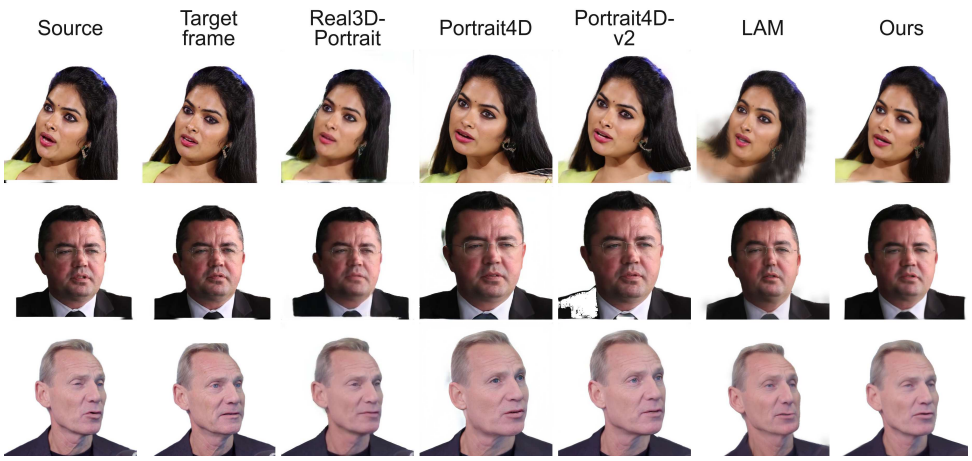


Fig. 5 Qualitative comparison of same-identity cross-frame animation. Each row uses one source image and another frame of the same identity as the driving target. All results are shown on a white background

Among the three evaluated levels, $l = 1$ achieves the highest PSNR. In comparison, $l = -1$ has a PSNR that is 0.5946 dB lower, but requires only 131.122 ms to construct a canonical avatar, approximately one-twelfth of the time required by $l = 1$, with lower peak GPU memory. From $l = -1$ to $l = 1$, the Gaussian count increases by approximately a factor of 16, while LPIPS decreases by 21.36% relative to the coarser level. Figure 4 shows the corresponding differences in fine textures and facial boundaries. Users can select a level within the shared model according to their reconstruction quality requirements and avatar construction budget.

4.3 Comparison with Existing Methods

Same-frame reconstruction.

Table 6 evaluates reconstruction of the input appearance. Our $l = 1$ model achieves the highest PSNR and SSIM and the lowest LPIPS and AKD. Compared with Real3D-Portrait, it improves PSNR by approximately 0.725 dB and reduces LPIPS by 0.0387. Real3D-Portrait achieves the highest CSIM and lowest AED, indicating that our model performs better in image reconstruction metrics but is not the best in identity similarity or expression error. Within our model, increasing the level from $l = 0$ to $l = 1$ reduces mean LPIPS by 0.0186.

Cross-frame animation.

Fixing the source image I_s while varying the driving frame I_t tests how the reconstructed avatar responds to motion. We compare our $l = 1$ configuration with external methods in Table 7. Our model achieves the highest PSNR and SSIM and the lowest LPIPS: compared with Portrait4D-v2, PSNR increases by 1.101 dB and LPIPS decreases by 0.0224; compared with LAM, SSIM increases by 0.0586. It also achieves the lowest APD and AKD, indicating closer agreement with the target pose and landmarks. Portrait4D-v2 has a lower expression error, while Real3D-Portrait has higher identity similarity. Figure 5 supplements the quantitative comparison with qualitative examples of facial appearance and motion transfer on a shared white background.

Avatar generation cost.

Table 8 complements the quality comparison with avatar generation time. The two lower-density levels require less construction time, while the time for $l = 1$ is of the same order as that of external methods. Values for external methods are aggregated from the available same-frame timing records. Since these methods use different avatar representations, Gaussian counts are compared only within our representation.

Cross-identity examples.

For cross-identity animation, we use expression, pose, and camera parameters from a different identity while retaining the source person's appearance and identity shape. In these qualitative examples, the shape is obtained by fitting FLAME in canonical space to the source clip. Figure 6 shows that all three levels preserve recognizable source identity cues, such as hair and accessories, while responding to changes in the driving head pose and mouth shape.

Table 8 Avatar asset generation time on the same hardware. Each external method has 400 timing records, except Real3D-Portrait with 399. Values for our method are taken from Table 5.

Method	**Time (ms)** ↓
Real3D-Portrait	1646.244
Portrait4D	1785.437
Portrait4D-v2	1737.995
LAM	1844.406
UniResAvatar ($l = -1$)	**131.122**
UniResAvatar ($l = 0$)	<u>270.612</u>
UniResAvatar ($l = 1$)	1565.106

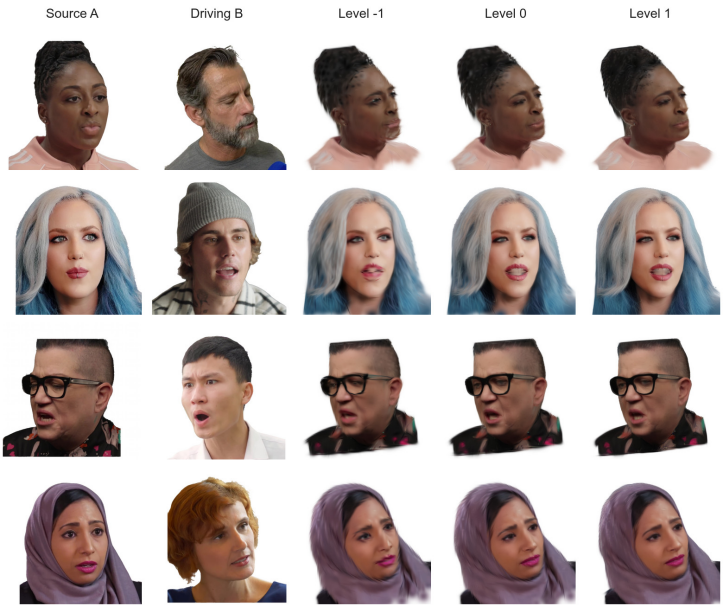


Fig. 6 Qualitative results of cross-identity animation. The shared three-level model preserves the source identity's appearance while responding to target motion from another identity

4.4 Ablation Studies

We examine image conditioning and multi-level training separately under the same-frame protocol on 50 test clips. All variants use the same data, the same initialization for shared modules, and the same optimization budget; each comparison changes only the component under study. Each variant is evaluated using its final weights obtained at the end of the same training schedule. Results are reported as equally weighted averages over the three levels.

Image conditioning and fusion.

Table 9 compares four image conditioning configurations: DINOv2 only, residual addition of DINOv2 and VAE features, linear concatenation projection, and the

Table 9 Image conditioning ablation, averaged equally over three levels. The full configuration uses concatenation–MLP–gated residual fusion.

Image conditioning	PSNR ↑	SSIM ↑	LPIPS ↓
DINOv2 only	21.932	0.7408	**0.1751**
Residual addition	21.579	0.7327	0.1807
Concatenation projection	21.841	0.7403	0.1754
Full fusion	**22.846**	**0.7442**	0.1811

full concatenation–MLP–gated residual fusion. Simply adding VAE features does not effectively improve reconstruction quality: residual addition and concatenation projection achieve PSNR values of 21.579 dB and 21.841 dB, respectively, neither showing a clear improvement. In contrast, the full fusion strategy achieves the highest PSNR and SSIM, improving them by 0.914 dB and 0.0034, respectively, over DINOv2 only. This indicates that the way the two feature types are combined affects reconstruction performance. However, the full fusion strategy has an LPIPS of 0.1811, which is 0.006 higher than that of DINOv2 only. Its advantages therefore lie mainly in pixel fidelity and structural similarity, without a corresponding improvement in perceptual similarity.

Table 10 Training component ablation, averaged equally over three levels. Removing detail supervision disables both gradient-based losses.

Training configuration	PSNR ↑	SSIM ↑	LPIPS ↓
Full model	**22.846**	**0.7442**	**0.1811**
Without level embedding	21.860	0.7298	0.1877
Without cross-level consistency	21.679	0.7250	0.1872
Without gradient detail supervision	21.747	0.7284	0.1871

Multi-level training components.

Table 10 evaluates the contribution of each training component to reconstruction quality by removing level embeddings, cross-level consistency, or gradient-based detail supervision in turn. The variant without gradient-based detail supervision disables image gradient supervision over both the full foreground and local crops, while retaining the corresponding perceptual losses. The full model achieves the best PSNR, SSIM, and LPIPS, at 22.846 dB, 0.7442, and 0.1811, respectively. Removing level embeddings reduces PSNR by 0.986 dB and SSIM by 0.0144. Removing cross-level consistency reduces PSNR by 1.167 dB and SSIM by 0.0192, the largest decreases in these two metrics among the three ablated variants. Removing gradient-based detail supervision reduces PSNR and SSIM by 1.099 dB and 0.0158, respectively. The LPIPS values of the three ablated variants also increase to 0.1877, 0.1872, and 0.1871, respectively,

further indicating that each training component contributes positively to perceptual reconstruction quality.

5 Limitations

The model trained and evaluated in this work primarily covers three predefined density levels. Although the framework can accommodate other densities by constructing templates with the corresponding deformation attributes, newly introduced levels still require training to adapt the model, and their reconstruction quality and one-time construction cost remain to be evaluated experimentally. The current animation process updates only Gaussian centers without adjusting their scales or orientations in response to local deformation, which may limit how closely the Gaussian distribution fits the deformed surface. For complex hairstyles, accessories, or large-amplitude motions, boundaries of regions such as hair and accessories may exhibit texture stretching or contour distortion; these potential issues require targeted experimental verification.

The image conditioning ablation shows that the full fusion configuration achieves higher PSNR and SSIM than the DINOv2-only configuration, but LPIPS increases from 0.1751 to 0.1811. This indicates that the current fusion strategy improves pixel fidelity and structural similarity without a corresponding reduction in perceptual distance.

6 Conclusion

We present UniResAvatar, a framework for reconstructing animatable head avatars from a single image that provides a choice of multi-level Gaussian representations within one shared model. It combines semantic and local appearance features from the source image, FLAME templates with geometric and deformation attributes, and level-conditioned prediction, enabling the shared network to generate FLAME-driven Gaussian avatars at the trained levels. Experiments at three levels on VFHQ-Test show that reducing the Gaussian count from 20,018 to 1,256 decreases canonical avatar construction time from 1565.106 ms to 131.122 ms and peak GPU memory from 3713.9 MB to 3117.4 MB. Higher-density configurations achieve better PSNR, SSIM, and LPIPS among these three levels.

Future work will construct and train additional coarsened and subdivided levels to evaluate their reconstruction quality and one-time construction cost. We will also explore animation that updates Gaussian scales and orientations in response to local deformation, and investigate feature fusion and optimization strategies that reduce LPIPS while preserving the gains in PSNR and SSIM.

Statements and Declarations

Author Contributions. Z.Z., W.Z., W.L., R.L., and X.Z. contributed to the conception and design of the study. Z.Z. prepared the materials, collected and analyzed the data, and wrote the initial draft. W.Z. and W.L. contributed to the experiments

and reviewed and revised the manuscript. All authors read and approved the final manuscript.

Funding. No funding was received for this work.

Data Availability. The VFHQ dataset, including VFHQ-Test, is available from the official VFHQ project page for non-commercial research, subject to the dataset's terms of use.

Code Availability. The code for UniResAvatar will be released on GitHub upon publication.

Ethics Approval. Not applicable.

Consent for Publication. Not applicable.

References

[1] Wiles O, Koepke AS, Zisserman A (2018) X2Face: A network for controlling face generation using images, audio, and pose codes. In: Computer Vision – ECCV 2018, Springer, pp 690–706. https://doi.org/10.1007/978-3-030-01261-8_41

[2] Siarohin A, Lathuilière S, Tulyakov S, et al (2019) First order motion model for image animation. In: Advances in Neural Information Processing Systems, vol. 32. URL https://proceedings.neurips.cc/paper/2019/hash/31c0b36aef265d9221af80872ceb62f9-Abstract.html

[3] Wang TC, Mallya A, Liu MY (2021) One-shot free-view neural talking-head synthesis for video conferencing. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 10034–10044. https://doi.org/10.1109/CVPR46437.2021.00991

[4] Drobyshev N, Chelishev J, Khakhulin T, et al (2022) MegaPortraits: One-shot megapixel neural head avatars. In: Proceedings of the 30th ACM International Conference on Multimedia, pp 2663–2671. https://doi.org/10.1145/3503161.3547838

[5] Gafni G, Thies J, Zollhöfer M, et al (2021) Dynamic neural radiance fields for monocular 4D facial avatar reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 8645–8654. https://doi.org/10.1109/CVPR46437.2021.00854

[6] Kerbl B, Kopanas G, Leimkühler T, et al (2023) 3D Gaussian Splatting for real-time radiance field rendering. ACM Transactions on Graphics 42(4):139:1–139:14. https://doi.org/10.1145/3592433

[7] Qian S, Kirschstein T, Schoneveld L, et al (2024) GaussianAvatars: Photorealistic head avatars with rigged 3D Gaussians. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 20299–20309. https://doi.org/10.1109/CVPR52733.2024.01919

[8] Shao Z, Wang Z, Li Z, et al (2024) SplattingAvatar: Realistic real-time human avatars with mesh-embedded Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 1606–1616. https://doi.org/10.1109/CVPR52733.2024.00159

[9] He Y, Gu X, Ye X, et al (2025) LAM: Large avatar model for one-shot animatable Gaussian head. In: Proceedings of ACM SIGGRAPH 2025 Conference Papers, Article 27. https://doi.org/10.1145/3721238.3730706

[10] Deng Y, Wang D, Ren X, et al (2024) Portrait4D: Learning one-shot 4D head avatar synthesis using synthetic data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 7119–7130. https://doi.org/10.1109/CVPR52733.2024.00680

[11] Deng Y, Wang D, Wang B (2024) Portrait4D-V2: Pseudo multi-view data creates better 4D head synthesizer. In: Computer Vision – ECCV 2024, Springer, pp 316–333. https://doi.org/10.1007/978-3-031-72643-9_19

[12] Ye Z, Zhong T, Ren Y, et al (2024) Real3D-Portrait: One-shot realistic 3D talking portrait synthesis. In: International Conference on Learning Representations. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/c74b624843218d9b6713fcf299d6d5e4-Abstract-Conference.html

[13] Li T, Bolkart T, Black MJ, et al (2017) Learning a model of facial shape and expression from 4D scans. ACM Transactions on Graphics 36(6):194:1–194:17. https://doi.org/10.1145/3130800.3130813

[14] Oquab M, Darcet T, Moutakanni T, et al (2024) DINOv2: Learning robust visual features without supervision. Transactions on Machine Learning Research. URL https://openreview.net/forum?id=a68SUt6zFt

[15] Rombach R, Blattmann A, Lorenz D, et al (2022) High-resolution image synthesis with latent diffusion models. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 10674–10685. https://doi.org/10.1109/CVPR52688.2022.01042

[16] Zhao J, Zhang H (2022) Thin-plate spline motion model for image animation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 3647–3656. https://doi.org/10.1109/CVPR52688.2022.00364

[17] Ren Y, Li G, Chen Y, et al (2021) PIRenderer: Controllable portrait image generation via semantic neural rendering. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 13739–13748. https://doi.org/10.1109/ICCV48922.2021.01350

[18] Hong FT, Zhang L, Shen L, et al (2022) Depth-aware generative adversarial network for talking head video generation. In: Proceedings of the IEEE/CVF

Conference on Computer Vision and Pattern Recognition, pp 3387–3396. https://doi.org/10.1109/CVPR52688.2022.00339

[19] Blanz V, Vetter T (1999) A morphable model for the synthesis of 3D faces. In: Proceedings of SIGGRAPH, pp 187–194. https://doi.org/10.1145/311535.311556

[20] Feng Y, Feng H, Black MJ, et al (2021) Learning an animatable detailed 3D face model from in-the-wild images. ACM Transactions on Graphics 40(4):88:1–88:13. https://doi.org/10.1145/3450626.3459936

[21] Daněček R, Black MJ, Bolkart T (2022) EMOCA: Emotion driven monocular face capture and animation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 20279–20290. https://doi.org/10.1109/CVPR52688.2022.01967

[22] Zheng Y, Fernández Abrevaya V, Bühler MC, et al (2022) I M Avatar: Implicit morphable head avatars from videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 13535–13545. https://doi.org/10.1109/CVPR52688.2022.01318

[23] Zheng Y, Yifan W, Wetzstein G, et al (2023) PointAvatar: Deformable point-based head avatars from videos. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 21057–21067. https://doi.org/10.1109/CVPR52729.2023.02017

[24] Xiang J, Gao X, Guo Y, et al (2024) FlashAvatar: High-fidelity head avatar with efficient Gaussian embedding. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 1802–1812. https://doi.org/10.1109/CVPR52733.2024.00177

[25] Chu X, Harada T (2024) Generalizable and animatable Gaussian head avatar. In: Advances in Neural Information Processing Systems, vol. 37, pp 57642–57670. https://doi.org/10.52202/079017-1838

[26] Hoppe H (1996) Progressive meshes. In: Proceedings of SIGGRAPH, pp 99–108. https://doi.org/10.1145/237170.237216

[27] Garland M, Heckbert PS (1997) Surface simplification using quadric error metrics. In: Proceedings of SIGGRAPH, pp 209–216. https://doi.org/10.1145/258734.258849

[28] Loop C (1987) Smooth subdivision surfaces based on triangles. Master's thesis, University of Utah

[29] Fan Z, Wang K, Wen K, et al (2024) LightGaussian: Unbounded 3D Gaussian compression with 15x reduction and 200+ FPS. In: Advances in Neural Information Processing Systems, vol. 37, pp 140138–140158. https://doi.org/10.52202/079017-4447

[30] Ren K, Jiang L, Lu T, et al (2025) Octree-GS: Towards consistent real-time rendering with LOD-structured 3D Gaussians. IEEE Transactions on Pattern Analysis and Machine Intelligence, Early Access, pp 1–15. https://doi.org/10.1109/TPAMI.2025.3568201

[31] Yan P, Ward R, Tang Q, et al (2026) ArchitectHead: Continuous level of detail control for 3D Gaussian head avatars. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, pp 1632–1642. https://doi.org/10.1109/WACV61042.2026.00164

[32] Zhang R, Isola P, Efros AA, et al (2018) The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp 586–595. https://doi.org/10.1109/CVPR.2018.00068

[33] Xie L, Wang X, Zhang H, et al (2022) VFHQ: A high-quality dataset and benchmark for video face super-resolution. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pp 656–665. https://doi.org/10.1109/CVPRW56347.2022.00081

[34] Wang Z, Bovik AC, Sheikh HR, et al (2004) Image quality assessment: From error visibility to structural similarity. IEEE Transactions on Image Processing 13(4):600–612. https://doi.org/10.1109/TIP.2003.819861