DGH: Dynamic Gaussian Hair

Junying Wang[1,2*] **Yuanlu Xu**[2] **Edith Tretschk**[2] **Ziyan Wang**[2] **Anastasia Ianina**[2]
Aljaz Bozic[2] **Ulrich Neumann**[1] **Tony Tung**[2]

[1]University of Southern California [2]Meta Reality Labs Research

Figure 1. Dynamic Gaussian Hair (DGH) is a framework that learns dynamic deformation and photorealistic novel-view synthesis of arbitrary hairstyles driven by head motions, while respecting upper-body collision. At runtime, given a hairstyle and head motion (a), DGH infers initial hair deformations (b), refines the deformations with dynamics (c), and generates 3D Gaussian Splats to achieve photorealistic novel-view synthesis (d).

Abstract

The creation of photorealistic dynamic hair remains a major challenge in digital human modeling because of the complex motions, occlusions, and light scattering. Existing methods often resort to static capture and physics-based models that do not scale as they require manual parameter fine-tuning to handle the diversity of hairstyles and motions, and heavy computation to obtain high-quality appearance. In this paper, we present Dynamic Gaussian Hair (DGH), a novel framework that efficiently learns hair dynamics and appearance. We propose: (1) a coarse-to-fine model that learns temporally coherent hair motion dynamics across diverse hairstyles; (2) a strand-guided optimization module that learns a dynamic 3D Gaussian representation for hair appearance with support for differentiable rendering, enabling gradient-based learning of view-consistent appearance under motion. Unlike prior simulation-based pipelines, our approach is fully data-driven, scales with training data, and generalizes across various hairstyles and head motion sequences. Additionally, DGH can be seamlessly integrated into a 3D Gaussian avatar framework, enabling realistic, animatable hair for high-fidelity avatar representation. DGH achieves promising geometry and appearance results, providing a scalable, data-driven alternative to physics-based simulation and rendering. Our project page: `https://junyingw.github.io/paper/dgh`

1 Introduction

Realistic, high-fidelity dynamic hair modeling and rendering are crucial for creating realistic avatars in animation and AR/VR applications. Adding hair dynamics can significantly enhance the realism of animated characters, as hair deformation and high-order motion effects contribute to physical

*Work performed during an internship at Meta Reality Labs Research

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

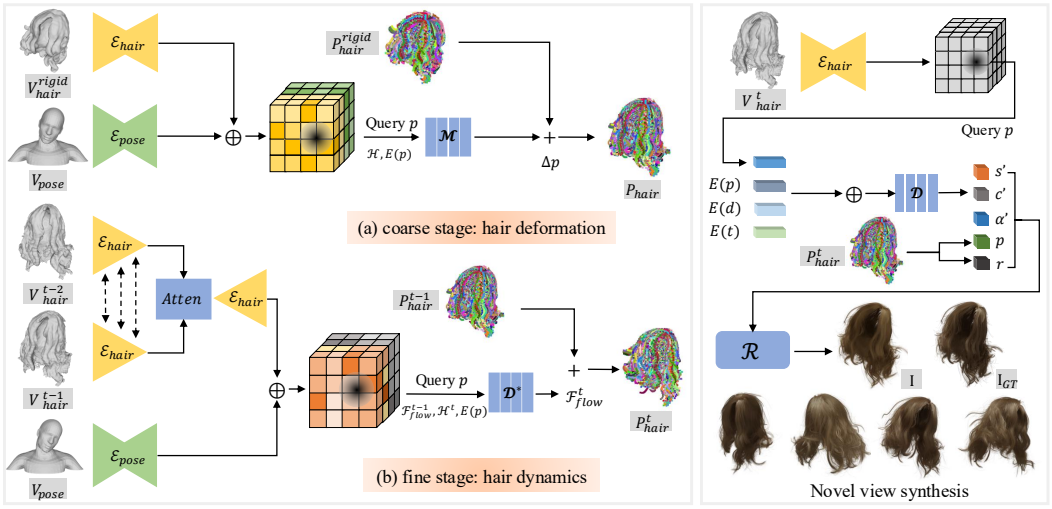

Figure 2. **Framework Overview.** DGH learns hair deformation dynamics and photorealistic appearance. **Stage I: Coarse-to-Fine Dynamic Hair Modeling.** The input hair model and the upper body are transformed into a canonical hair volume $V_{\text{hair}}^{\text{rigid}}$ and a pose volume V_{pose}, respectively. Then, a coarse-to-fine strategy deforms the hair model. At the coarse stage, points $\mathbf{p}_i$ are sampled from the rigidly transformed hair, and the interpolated features from $\mathcal{E}_{\text{pose}}$, $\mathcal{E}_{\text{hair}}$, head pose $\mathcal{H}$, and positional encoding $E(p)$ are concatenated and fed into an MLP $\mathcal{M}$ to predict displacements $\Delta\mathbf{p}$, producing deformed hair points P_{hair}. The fine stage refines hair deformation with dynamics by estimating flow $\mathcal{F}_{\text{flow}}^{t}$ through cross-attention between volumetric features from previous frames V_{hair}^{t-2} and V_{hair}^{t-1}, ensuring smooth temporal transitions. **Stage II: Appearance Optimization.** We train an MLP $\mathcal{D}$ to predict color c', scale s', and opacity α' of 3D Gaussian Splats from features of the deformed hair. Differentiable rasterization leverages the appearance model to synthesize high-quality renderings that adapt to hair movement and occlusion dynamics.

plausibility. Capturing and modeling these effects, however, poses substantial challenges due to the complexity of hair motion and the intricate interactions among hair strands and head/shoulder. Physics-based hair simulation methods produce high-quality hair dynamics but lack generalization as they require manual parameter fine-tuning per hairstyle (e.g., ponytail, curly hair) to reproduce realistic deformations (e.g., hair stiffness, density, damping). They also need to model interactions with the environment, such as collisions with shoulders, which are computationally expensive. The models are non-differentiable, and thus unsuitable for scalability with data-driven based solutions. Robust implementation of complex deformations is nontrivial, and unexpected behaviors usually break the experience. Most real-time applications model only simple or quasi-static deformations.

The computational cost to create photorealistic digital hair is very high. Realistic rendering requires path tracing of accurate hair modeling, which is translucent, contains occlusions, and accounts for light scattering effects [1]. Hair occlusion patterns change while in motion, posing significant challenges for dynamic hair rendering. The creation of a high-quality synthetic hair dataset at scale typically requires a rendering farm with multiple GPUs. Real-time hair rendering engines (Unreal, Unity) compromise on quality (e.g., approximate lighting) while still requiring a desktop GPU.

Methods relying on Neural Radiance Fields (NeRF) [2, 3] or 3D Gaussian Splatting (3DGS) [4–7] have enabled effective and high-quality view synthesis, baking complex hair rendering effects under constrained lighting environments. Although they enable photorealistic avatar re-animation, the hairstyle typically undergoes only rigid transformations without non-rigid or dynamic effects, especially for longer hair, due to weak control signals like head poses of a single timestep. Similarly, GaussianHair [8] and Gaussian Haircut [9] reconstruct hair geometry (from static scenes) using strand-aligned 3D Gaussian representations that can be animated with a physics-based simulation engine, which in turn is usually computationally intensive and requires tedious parameter tuning.

To address these limitations, we introduce *Dynamic Gaussian Hair*, a novel learning-based framework that learns hair deformation with dynamics given canonical hairstyles and head motions. By attaching 3D Gaussians to hair segments and leveraging the appearance optimization, DGH enables high-quality dynamic hair novel view synthesis at a fraction of the computational cost of a high-end rendering

system. Unlike physics-based simulators, which rely on explicit meshes and require additional rendering and conversion steps, our DGH framework only needs head rotation and a dense point cloud/pre-trained GS avatar. By converting static hair and upper-body into volume, we enable mesh-free deformation prediction, making our model more compatible with Gaussian-based avatars and easier to integrate into learning-based re-animation pipelines without additional rigging or simulation overhead.

Due to the lack of real captures of dynamic hair deformations with accurate strand tracking, we create a new synthetic dynamic hair dataset from scratch. Hairstyles with strands are modeled from industry experts, and animated with a physics-based simulation engine. Mutiple-view images are generated with a render farm. The dataset includes frame-by-frame hair geometry deformation for various hair styles, along with corresponding upper body geometry and head motions. The 3D models are rendered using fine-tuned hair shaders, resulting in photorealistic videos (see Appendices).

As shown in Fig. 2, our dynamic Gaussian hair framework learns hair deformation dynamics and photorealistic appearance. In the first stage, we introduce a coarse-to-fine framework for *dynamic hair modeling* using a feed-forward network, which, unlike traditional physics-based simulation pipelines, is fully data-driven, differentiable, and free of manual parameter tuning. Specifically, we learn a volumetric implicit deformation model in canonical space that maps static hair geometry to deformed hair, representing hair as a dense point cloud to support diverse hairstyles (e.g., long, curly, ponytail) and accommodate point, Gaussian, or mesh-based representations. We encode the head and upper body as volumetric features. During training, a random subset of static hair points is supervised using synthetic ground-truth displacements. The coarse stage is time-independent and serves as a physically plausible hair deformation initialization, leveraging upper-body features and an SDF constraint to prevent hair-body penetration. In the fine stage, we introduce time-dependent dynamics by predicting flow vectors and applying latent-space cross-attention over previous frames to capture temporal consistency and high-frequency hair motion effects such as inertia, oscillation, and damping. In the second stage, we optimize the *dynamic hair appearance* for novel view synthesis, enabling photorealistic rendering under complex motions and viewpoints. Each hair strand is represented as a sequence of stretched cylindrical Gaussian primitives as shown in Fig. 3, and we introduce a lightweight non-linear model that refines appearance using strand-level tangent information. Our data-driven framework robustly handles arbitrary hairstyle deformation under varying head motions and occlusions, achieving high-fidelity, view-consistent rendering. Our results are best viewed in our supplemental video.

We summarize our contributions as: (1) **Learning-based volumetric hair deformation:** We introduce a pose-driven, volumetric implicit deformation model that learns to map static hair to dynamic motion across diverse hairstyles and strand densities. Unlike physics-based methods, our approach is fully data-driven, requires no manual parameter tuning, and generalizes to novel head poses. (2) **Coarse-to-fine hair deformation dynamics:** We propose a coarse-to-fine learning framework that first predicts pose-dependent deformations and then refines temporal dynamics via flow-based residual learning. This approach ensures stability and generalizes well to unseen head motions. (3) **Differentiable dynamic hair appearance optimization:** We represent hair as cylindrical Gaussian primitives and optimize their dynamic appearance with a lightweight, strand-guided network. This differentiable formulation enables photorealistic, view-consistent rendering under motion and occlusion, and integrates seamlessly with neural avatar systems. We will release our synthetic dynamic-hair dataset to accelerate research on dynamic hair modeling.

2 Related Work

Creating realistic avatars includes several challenging steps. One of them is hair capture and animation. Modeling hair appearance and geometry has been studied in the static and dynamic setting.

Static Hair Modeling Handling diverse hairstyles is challenging. Some methods focus on specific cases like braided [10], curly [11], or generative approaches [12], while others aim to generalize across static and dynamic settings. They rely on single-view [13–15] or multi-view inputs with orientation maps [16–19], sometimes using simulated strands [18, 20] or geometric heuristics. NeuralStrands [21] constrains 3D surface orientation and uses point-based differentiable rendering [22, 23]. NeuralHD-Hair [24] and earlier work [25] leverage 2D supervision with 3D spatial cues. Some methods use real wigs [26], line-based 3D reconstruction [27], or OLAT images [28] for static hair modeling. Others

rely on 2D observations to learn hair growth fields [24, 29], or use optimization and differentiable rendering to extract strands [30]. Beyond image-based approaches, volumetric and neural representations have been explored [2, 31, 32]. Neural Radiance Fields (NeRF) enable high-fidelity rendering for static scenes [2, 3], but suffer from slow rendering speeds. Recent representations like Mixture of Volumetric Primitives (MVP) [33] and 3D Gaussian Splatting (3DGS) [34] improve quality and real-time performance, yet dynamic hair reconstruction remains challenging. INSTA [31] models dynamic neural radiance fields with neural graphics primitives around a face model, focusing on head geometry and efficiency. Sklyarova et al. [32] reconstruct hair using implicit volumes followed by strand-level refinement guided by hairstyle priors. Volumetric hair representations [35, 36] are able to encode 3D global spatial information for static hair modeling. However, all existing work focuses on static hair modeling, leaving the challenge of reanimating static hair for animatable avatars in AR/VR applications an unresolved problem.

Dynamic Hair Modeling The motion patterns of hair are hard to emulate. One option is hair simulation to advance between adjacent frames with temporal consistency [37]. In [38], per-frame reconstructions of hair strands are aligned with motion paths of hair strands from spatio-temporal slices of a video volume. Grid search over different simulation parameters can be used to determine the set of parameters that matches the visual observations best [39]. Current physics-based and neural hair simulators [40–42] enable real-time simulation of hundreds of thousands of strands. However, achieving this on AR/VR devices remains impractical due to their limited GPU memory and processing power. Recent work Quaffure [43] has only handled quasi-static simulation and does not integrate a tractable rendering solution. Dynamic hair can also be modeled with neural volumetric approaches. HVH [44] learns a volumetric representation for dynamic capturing, while NeuWigs [45] accounts for head movements and gravity to produce realistic animations across various hairstyles.

3D Gaussian Head Avatars Modeling realistic 3D head avatars is crucial for immersive VR/AR applications. While 3D Gaussian Splatting (3DGS) [34] has enabled high-fidelity head avatars, dynamic hair modeling remains a significant challenge, as realistic hair must exhibit natural motion, deformation, and interactions. 3DGS-based head avatars [5, 6, 46–48] leverage learning-based deformation fields to animate facial expressions and head movements. However, with these models the hair remains a static transformation attached to the head, failing to exhibit natural flow or secondary motion. Several works have explored hair modeling within the Gaussian framework. GaussianHair [8] captures strand-level structure using cylindrical Gaussians. Gaussian HairCut [9] combines strand priors with 3DGS for photorealistic rendering, yet both do not model the dynamic hair appearance. To enable realistic hair dynamics in reanimation, we propose a dynamic Gaussian hair representation that models arbitrary hairstyle deformation driven by head motion, supporting dynamic hair novel view synthesis.

3 Method

In this section, we present our method for Dynamic Gaussian Hair modeling, as illustrated in Fig. 2. Our framework consists of two main stages that model hair dynamics and hair appearance. In the first stage, we learn pose-dependent hair deformation via a volumetric implicit function conditioned on canonical static hair, head motion, and upper body mesh. This time-independent model supports arbitrary head poses and strand counts, represented as dense point clouds. We further refine temporal hair dynamics by predicting 3D flow vectors and applying latent-space cross-attention across adjacent frames to ensure temporal consistency. In the second stage, we optimize time-varying hair appearance to handle motion-induced occlusions, enabling photorealistic novel view synthesis. During inference, we predict hair deformations in a recurrent manner based on the previous timesteps' deformations. Our appearance model then performs novel view synthesis, generating accurate and realistic hair appearance across frames.

3.1 Coarse-to-Fine Dynamic Hair Modeling

Modeling realistic hair dynamics is challenging due to complex motion and temporal variation. We address this with a coarse-to-fine learning framework, and show the following hypothesis:

Hypothesis Differentiable rendering of dynamic, photorealistic hair requires accurate and temporally consistent hair tracking. Instead of relying on explicit physics-based simulation, we decompose the simulation task into two stages: a coarse stage that learns pose-driven, time-independent deformation

for stable initialization, and a fine stage that refines high-frequency dynamics through 3D flow prediction and temporal cross-attention. We validate this hypothesis through the design of Alg. 1.

Algorithm 1 Coarse-to-Fine Dynamic Hair Modeling

Input: Canonical hair $P_{\text{hair}}^{\text{can}}$, proxy mesh (head and shoulders), head poses $\{\mathcal{H}^{t-2}, \mathcal{H}^{t-1}, \mathcal{H}^{t}\}$
Output: Dynamic hair P_{hair}^{t} at time t

Coarse Stage (Time-independent, pose-driven deformation)
Transform canonical hair: $P_{\text{hair}}^{\text{rigid}} \leftarrow \mathcal{T}_{\text{rigid}}(P_{\text{hair}}^{\text{can}}, \mathcal{H})$
Voxelize rigid hair and proxy mesh: $V_{\text{hair}}^{\text{rigid}} \leftarrow \text{SDF}(P_{\text{hair}}^{\text{rigid}}),\ V_{\text{pose}} \leftarrow \text{SDF}(\text{proxy mesh})$
Encode features: $\mathcal{V} \leftarrow \text{Concat}(\mathcal{E}_{\text{pose}}(V_{\text{pose}}), \mathcal{E}_{\text{hair}}(V_{\text{hair}}^{\text{rigid}}))$
for $\mathbf{p}_i \in P_{\text{hair}}^{\text{rigid}}$ **do**
  Sample feature: $\mathbf{v}_i \leftarrow \text{Interp}(\mathcal{V}, \mathbf{p}_i)$
  Predict displacement: $\Delta\mathbf{p}_i \leftarrow \mathcal{M}(\mathbf{v}_i, E(\mathbf{p}_i), \mathcal{H})$
  Update position: $\mathbf{p}_i \leftarrow \mathbf{p}_i + \Delta\mathbf{p}_i$
end for
Set $P_{\text{hair}} \leftarrow \{\mathbf{p}_i\}$

Fine Stage (Temporal flow refinement)
Voxelize past hair states: $V_{\text{hair}}^{t-2}, V_{\text{hair}}^{t-1} \leftarrow \text{SDF}(P_{\text{hair}}^{t-2}), \text{SDF}(P_{\text{hair}}^{t-1})$
Encode volumes: $\mathcal{V}_{t-2} \leftarrow \mathcal{E}_{\text{hair}}(V_{\text{hair}}^{t-2}),\ \mathcal{V}_{t-1} \leftarrow \mathcal{E}_{\text{hair}}(V_{\text{hair}}^{t-1})$
Cross-attend features: $\mathcal{V}_{\text{flow}} \leftarrow \text{CrossAttn}(\mathcal{V}_{t-1}, \mathcal{V}_{t-2})$
for $\mathbf{p}_i \in P_{\text{hair}}^{t-1}$ **do**
  Sample volume feature: $\mathbf{f}_i \leftarrow \text{Interp}(\mathcal{V}_{\text{flow}}, \mathbf{p}_i)$
  Fuse with pose and prior flow: $\hat{\mathbf{f}}_i \leftarrow \text{Concat}(\mathbf{f}_i, \mathcal{H}^t, \mathcal{F}_{\text{flow}}^{t-1}(\mathbf{p}_i), E(\mathbf{p}_i))$
  Predict flow vector: $\Delta\mathbf{p}_i^t \leftarrow \mathcal{D}^*(\hat{\mathbf{f}}_i)$
  Apply flow: $\mathbf{p}_i^t \leftarrow \mathbf{p}_i + \Delta\mathbf{p}_i^t$
end for
Update outputs: $P_{\text{hair}}^{t} \leftarrow \{\mathbf{p}_i^t\},\ \mathcal{F}_{\text{flow}}^{t} \leftarrow \{\Delta\mathbf{p}_i^t\}$
return P_{hair}^{t}

Coarse Stage. We first learn pose-dependent hair deformations from a canonical point cloud $P_{\text{hair}}^{\text{can}}$. The canonical hair is rigid transformed using a head pose $\mathcal{H}$ to produce $P_{\text{hair}}^{\text{rigid}}$, which is voxelized into an SDF volume $V_{\text{hair}}^{\text{rigid}}$. A proxy mesh of the posed head and shoulders is similarly voxelized into V_{pose}. These volumes are encoded via 3D CNNs $\mathcal{E}_{\text{hair}}$ and $\mathcal{E}_{\text{pose}}$, and concatenated to form the latent feature grid $\mathcal{V}$. For each point $\mathbf{p}_i \in P_{\text{hair}}^{\text{rigid}}$, we interpolate its feature $\mathbf{v}_i$ from $\mathcal{V}$, and input it along with positional encoding $E(\mathbf{p}_i)$ and pose $\mathcal{H}$ into an MLP $\mathcal{M}$ to predict the displacement $\Delta\mathbf{p}_i$. The updated position $\mathbf{p}_i$ is computed as $\mathbf{p}_i + \Delta\mathbf{p}_i$. We train using a random subset of hair points, with a total loss:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{p}}\mathcal{L}_{\text{point}} + \lambda_{\text{SDF}}\mathcal{L}_{\text{SDF}}, \tag{1}$$

where $\mathcal{L}_{\text{point}}$ is the MSE between predicted and ground-truth displacements, and $\mathcal{L}_{\text{SDF}}$ penalizes collisions with the body mesh using a hair-to-body Signed Distance Fields [49]. This stage yields physically plausible, pose-driven deformation that serves as initialization for dynamic refinement. Please see the Appendices for details on the 3D CNN architecture.

Fine Stage. The fine stage refines dynamic hair motion by predicting temporally consistent 3D flow vectors. We first voxelize the past hair states P_{hair}^{t-2} and P_{hair}^{t-1} into SDF volumes V_{hair}^{t-2} and V_{hair}^{t-1}, which are encoded via $\mathcal{E}_{\text{hair}}$ into latent feature grids. A cross-attention module aggregates these into a fused volume $\mathcal{V}_{\text{flow}}$. For each point $\mathbf{p}_i \in P_{\text{hair}}^{t-1}$, we sample its feature $\mathbf{f}_i$ from $\mathcal{V}_{\text{flow}}$, then concatenate it with the current pose $\mathcal{H}^t$ and previous flow $\mathcal{F}_{\text{flow}}^{t-1}(\mathbf{p}_i)$ to form $\hat{\mathbf{f}}_i$. These features are passed to a refinement network $\mathcal{D}^*$ to predict the per-point flow vector $\Delta\mathbf{p}_i^t$, which is applied to update the hair position $\mathbf{p}_i^t$. We supervise the predicted flow $\mathcal{F}_{\text{flow}}^t$ using an MSE loss against ground-truth flow $\mathcal{F}_{\text{GT}}^t$ from synthetic data, with a total loss:

$$\mathcal{L} = \mathcal{L}_{\text{flow}} = \text{MSE}(\mathcal{F}_{\text{flow}}^t, \mathcal{F}_{\text{GT}}^t) \tag{2}$$

This stage enables temporally consistent hair animation driven by data rather than explicit simulation.

3.2 Dynamic Hair Appearance Optimization

To achieve realistic rendering quality while ensuring speed and optimizability, we rely on 3DGS [34] to represent hair appearance. 3DGS assigns a learnable position $\mathbf{p}$, spatial scale $\mathbf{S}$ and rotation $\mathbf{R}$ (combined into covariance matrix Σ), opacity o, and color $\mathbf{c}$ to each Gaussian G:

$$G(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\mathbf{p})^T\Sigma^{-1}(\mathbf{x}-\mathbf{p})}, \quad \Sigma = \mathbf{RSS}^T\mathbf{R}^T, \\ \Sigma' = \mathbf{JW}\Sigma\mathbf{W}^T\mathbf{J}^T. \tag{3}$$

To render novel views, Gaussians are projected onto the image plane using a differentiable Surface Splatting method [50]. Given the viewing transform matrix $\mathbf{W}$ and the Jacobian $\mathbf{J}$ of the affine approximation, the covariance matrix Σ' of a 3D Gaussian's corresponding 2D Gaussian is defined as above. To determine a pixel's color $\mathbf{c}$, $N-1$ 2D Gaussians are sorted by depth and then composed with α-blending:

$$\mathbf{c} = \sum_{i=1}^{N} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{4}$$

where $\mathbf{c}_i$ is the color of each Gaussian, and α_i is given by multiplying the opacity with the value of the 2D Gaussian with covariance Σ' at the pixel location.

Inspired by [8, 9], we represent each hair strand as a sequence of connected cylindrical Gaussians (Fig. 3) for dynamic hair rendering, eliminating Gaussian densification [34], and fixing the number of primitives across frames. Because our dynamic hair model performs forward warping, we can transform each Gaussian from the canonical model to any frame t, which makes it trivial to directly propagate the Gaussian color $\mathbf{c}$ and scale $\mathbf{s}$ across time. The rendering is performed using a Differentiable Tile Rasterizer [34] $\mathcal{R}$, which yields image $\mathbf{I}$.

However, Fig. 5 shows that naively propagating the colors of the canonical frame to the dynamic sequence can lead to missing appearance details in dynamic frames due to complex hair self-occlusion and the interaction of light with hair in motion. To address this, we propose a lightweight non-linear model $\mathcal{D}$ that adjusts Gaussian parameters according to the hair dynamics. We train our model using multi-view video sequences under constant lighting. Hair scattering is complex and anisotropic due to strand interactions, as described by the hair BSDF [1, 51], which highlights the importance of strand tangents in appearance. To capture this, we incorporate the hair tangent vector $\mathbf{t}$ as additional geometric input for optimizing Gaussian parameters. Specifically, we use an MLP $\mathcal{D}$ that takes as input the per-point feature sampled from the encoded hair volume $\mathcal{E}_{\text{hair}}(V^t_{\text{hair}})$ at Gaussian mean $\mathbf{p}$, along with positional encodings E for $\mathbf{p}$, $\mathbf{t}$, $\mathbf{d}$, and $\mathbf{d}$ denotes the view direction.

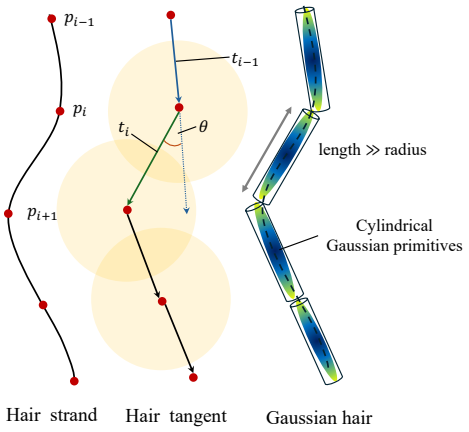

Figure 3. **Hair Representation.** We show different hair representations with tangent vectors and curvature. For Gaussian hair, we attach cylindrical Gaussian primitive [8] to each segment with a length much greater than its radius.

These enable $\mathcal{D}$ to express anisotropic effects. It outputs modified color $\mathbf{c}'$ in the form of spherical harmonic (SH) coefficients, scales $\mathbf{s}'$ and opacity α' :

$$\mathbf{c}', \mathbf{s}', \alpha' = \mathcal{D}\left(\mathcal{E}_{\text{hair}}(V^t_{\text{hair}}; \mathbf{p}), E(\mathbf{p}), E(\mathbf{t}), E(\mathbf{d})\right). \tag{5}$$

As present in Fig. 2, the predicted appearance is then rendered through differentiable rasterization $\mathcal{R}$ to produce the final image $\mathbf{I}$. Fig. 5 shows that our model enhances the hair appearance, achieving realistic renderings under consistent lighting. For training, we use common L_1, SSIM and LPIPS [52] reconstruction losses:

$$L = \lambda_{\text{rgb}} L_{\text{rgb}} + \lambda_{\text{ssim}} L_{\text{ssim}} + \lambda_{\text{lpips}} L_{\text{lpips}}. \tag{6}$$

Hypothesis Modeling hair strands with a fixed number of discrete Gaussian primitives leads to visual discontinuities in regions of high curvature. We hypothesize that local curvature is an effective cue for adaptively blending neighboring Gaussians to improve visual continuity.

Curvature-based Gaussian blending. Unlike continuous surfaces, Gaussian hair treats each segment as an independent shading unit, each with its own tangent vector $\mathbf{t}_i$. In high-curvature regions, large angular differences between adjacent tangents ($\mathbf{t}_i$, $\mathbf{t}_{i+1}$) can cause shading discontinuities. Although increasing segment density can alleviate this issue, hair tracking typically uses a fixed number of segments per strand. Here we present a simplified hair diffuse shading formulation:

$$\begin{aligned} I_i &\propto \max(0, \mathbf{t}_i \cdot \mathbf{l}), \\ |I_i - I_{i+1}| &\propto |\max(0, \mathbf{t}_i \cdot \mathbf{l}) - \max(0, \mathbf{t}_{i+1} \cdot \mathbf{l})| \,. \end{aligned} \tag{7}$$

where I_i is the shading intensity of the i-th hair segment, $\mathbf{t}_i$ is its tangent direction, and $\mathbf{l}$ is the light direction. The shading discontinuity between adjacent Gaussians is represented by $|I_i - I_{i+1}|$, and this value increases with curvature. To address this, we introduce a curvature-based blending algorithm that adaptively adjusts the interpolation of Gaussian parameters (color and opacity) based on local strand curvature. For each segment i, we compute the tangent vector $\mathbf{t}_i$, curvature κ_i, and normalized curvature $\tilde{\kappa}_i$, which is used as the blending weight w_i:

$$\begin{aligned} \mathbf{t}_i &= \frac{\mathbf{p}_{i+1} - \mathbf{p}_i}{\|\mathbf{p}_{i+1} - \mathbf{p}_i\|}, \quad \kappa_i = \|\mathbf{t}_i - \mathbf{t}_{i-1}\|, \\ \tilde{\kappa}_i &= \frac{\kappa_i}{\kappa_{\max} + \epsilon}, \quad w_i = \tilde{\kappa}_i, \end{aligned} \tag{8}$$

where $\mathbf{p}_i$ denotes the 3D position of the i-th point along the strand, $\kappa_{\max}$ is the maximum curvature across the strand, and ϵ is a small constant for numerical stability. We apply w_i to blend spherical harmonics (SH) coefficients and opacity α between adjacent Gaussians:

$$\begin{aligned} \text{SH}_{\text{blended},i} &= \text{SH}_i \cdot (1 - w_i) + \text{SH}_{i-1} \cdot w_i, \\ \alpha_{\text{blended},i} &= \alpha_i \cdot (1 - w_i) + \alpha_{i-1} \cdot w_i. \end{aligned} \tag{9}$$

As shown in Fig. 8, our curvature-based blending significantly improves the visual continuity of hair rendering, particularly in curved regions. This blending is jointly optimized with other Gaussian parameters (e.g., position, scale, SH, opacity) during training, enabling more realistic and temporally coherent appearance in dynamic hair sequences.

3.3 Inference and Implementation Details

We train our model on a single A100 GPU using the Adam optimizer and a learning rate of 1×10^{-4} for both stages. In Stage I, each iteration samples 200k points from the hair point cloud. Here we provide the formal definitions for Eq. 1, including the point loss $\mathcal{L}_{\text{point}} = \frac{1}{N}\sum_{i=1}^{N} \|\hat{\mathbf{p}}_i - \mathbf{p}_i^{\text{GT}}\|_2^2$ and the SDF penalty loss $\mathcal{L}_{\text{SDF}} = \frac{1}{N}\sum_{i=1}^{N} \max(0, -\text{SDF}(\hat{\mathbf{p}}_i))$, where $\hat{\mathbf{p}}_i$ is the predicted 3D hair point, $\mathbf{p}_i^{\text{GT}}$ is the corresponding ground-truth hair point, and $\text{SDF}(\hat{\mathbf{p}}_i)$ denotes the signed distance to the body surface. We penalize points inside the mesh via the ReLU (i.e., $\max(0, \cdot)$) operation. For Eq. 1, we set λ_{p}=1.0 and λ_{SDF}=0.01; for Eq. 6, we set λ_{rgb}=1.0, λ_{ssim}=0.1, and λ_{lpips}=0.1.

During inference, we are given a head motion sequence and a canonical hair groom. For $t = 0$, we apply only the coarse stage to initialize the dynamic hair. At $t = 1$, the fine stage is used with self-attention over the result from $t = 0$, and the input flow $\mathcal{F}_{\text{flow}}^0$ is set to 0. For $t > 1$, the pipeline operates recurrently as described. Once the dynamic hair sequence is obtained, we infer hair appearance based on the tracked hair positions to achieve dynamic hair rendering. For further implementation details (runtime and memory analysis), please refer to the Appendices.

4 Experiments

Dataset. Due to the lack of real hair tracking data, we generate a synthetic dataset using XPBD-based physics simulation [53]. It includes dynamic sequences across diverse hairstyles, driven by motion-captured head movements. Each simulated groom consists of $1500k$ hair strand with 24 vertices per strand. For each hairstyle, we simulate 100 motion sequences of 100 frames, totaling $10k$ frames. Each frame includes deformed hair positions, head motion parameters, and the upper-body mesh. To generate our dynamic hair appearance dataset, we simulate 500 frame sequences per groom and render multi-view videos from 24 camera angles in Blender. See the Appendices for dataset details.

Figure 4. **Hair Deformation Comparison.** From left to right column, given a canonical groom and upper body pose, rigid transformations result in unrealistic hair deformation with upper-body penetration, while our method achieves natural deformation across different grooms with correct collisions.

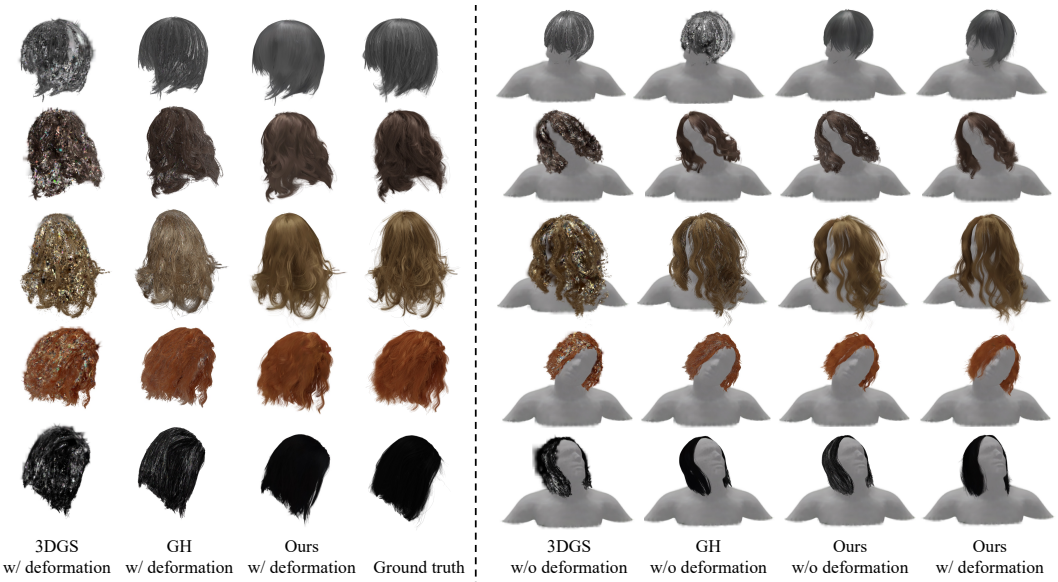

Figure 5. **Dynamic hair appearance evaluation.** The left section compares dynamic hair rendering using our hair deformation/tracking model against baseline methods: 3DGS [34] and Gaussian Haircut (GH) [9]. The right section shows rendering results merging hair and body Gaussian primitives. Without our hair tracking model, the hair appears rigid and unrealistic, while our appearance model enhances hair rendering quality.

Evaluation. We evaluate our method on deformation and appearance, comparing it with baselines across 5 hair subjects using our synthetic hair dataset. Each groom's training dataset consists of 90 motion sequences, while testing is performed on the remaining 10 sequences. For dynamic appearance, we assess unseen hair motions (100 frames per subject) and novel views, capturing 100 views via horizontal camera rotation. See the Appendices for more dataset and evaluation details.

	Ours		Rigged hair	
Subject	**Error↓**	**Chamfer↓**	**Error↓**	**Chamfer↓**
Subject 1	0.0738	0.0233	0.1411	0.0382
Subject 2	0.0998	0.0294	0.1911	0.0474
Subject 3	0.1187	0.0342	0.2891	0.0637
Subject 4	0.0679	0.0260	0.1054	0.0334
Subject 5	0.0562	0.0201	0.0928	0.0296
Average	0.0832	0.0266	0.1639	0.0424

Table 2. **Hair deformation comparison.** L_2 and Chamfer distances across normalized subjects.

	Ours			Gaussian Haircut [9]			3D GS [34]		
Subject	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**	**PSNR↑**	**SSIM↑**	**LPIPS↓**
Subject 1	28.026	0.906	0.101	23.248	0.873	0.101	20.747	0.852	0.132
Subject 2	24.817	0.820	0.169	20.921	0.791	0.180	19.972	0.772	0.215
Subject 3	24.987	0.744	0.227	20.960	0.742	0.236	19.972	0.687	0.287
Subject 4	27.534	0.955	0.071	24.181	0.906	0.078	20.246	0.894	0.101
Subject 5	29.681	0.933	0.069	26.053	0.924	0.058	23.605	0.906	0.083
Average	27.009	0.871	0.127	23.073	0.847	0.131	20.908	0.822	0.164

Table 1. **Comparison with others baselines on hair appearance.** We report PSNR↑, SSIM↑, and LPIPS↓ for each groomed subject to compare our method with other baselines in rendering quality.

Metrics. We evaluate the quality of rendered images via reconstruction fidelity (PSNR [54]), local structural similarity (SSIM [54]), and perceptual similarity (LPIPS [52]) between the synthesized images and the ground truth. We evaluate per-frame motion via the $L2$ error, Chamfer distance [55], between our dense hair point cloud and the ground truth. We evaluate the temporal consistency of the hair motion via the $L2$ error between the flow vectors in the predicted and ground-truth point cloud sequences, and we show the flow error of different settings in Fig. 7

Baselines. Our framework learns dynamic hair motion and time-varying appearance using Gaussians in a differentiable manner. We compare against 3DGS [34] and Gaussian Haircut [9], retraining both on our synthetic dataset for each static hairstyle. Since dynamic hair modeling is underexplored, we integrate our dynamics model into each baseline, optimize their canonical hair appearance, and re-animate hair using our estimated motion for fair comparison. We further benchmark dynamics against two references: rigid-transformed canonical hair (lower bound) and physics-based XPBD [53] results (upper bound). Full baseline-training and implementation details are provided in the Appendices.

Setting	**Error↓**
Ours w/o SDF	0.1269
Ours w/o motion	0.0964
Ours w/o atten.	0.0909
Ours full	0.0832

Table 3. **Hair dynamics ablation.** We show the $L2$ error across settings on deformed hair test sets.

Setting	**PSNR↑**	**SSIM↑**	**LPIPS↓**
Ours w/o tan. & blend	20.89	0.80	0.19
Ours w/o blend	25.08	0.88	0.18
Ours full	28.12	0.90	0.19

Table 4. **Hair appearance ablation.** We compare different appearance model settings on rendering quality.

Sample frames: camera view 1 Sample frames: camera view 2

Figure 6. **Dynamic Gaussian Hair qualitative evaluation.** We present qualitative visual results across various hairstyles and camera views, where the driven motion is obtained from the Mixamo MoCap data [56].

4.1 Results

Tab. 2 shows that our deformation model significantly enhances realistic hair deformations and reduces errors compared to rigid transformations on static hair. As shown in Fig. 4, applying rigid transformations to static hair leads to unnatural deformations, such as the absence of gravity effects and hair penetrating the body mesh. Tab. 1 presents a quantitative comparison of our method with GH (Gaussian Haircut) [9] and 3DGS [34] on our synthetic dynamic hair dataset. Our approach consistently achieves the highest PSNR and SSIM across all subjects, indicating high-fidelity rendering quality. In Fig. 5 and 6 we further demonstrate dynamic hair appearance across different frames and hairstyles. Qualitative comparisons demonstrate that without our appearance optimization, both 3DGS and strand-based Gaussian representations suffer from degraded rendering quality due to hair self-occlusions. In contrast, our motion-dependent appearance model is robust

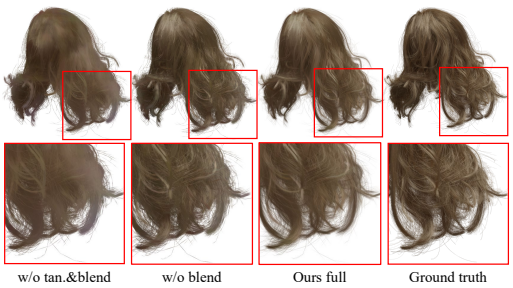

Figure 8. **Ablation on appearance.** Top: results of appearance model variants. Bottom: zoomed details.

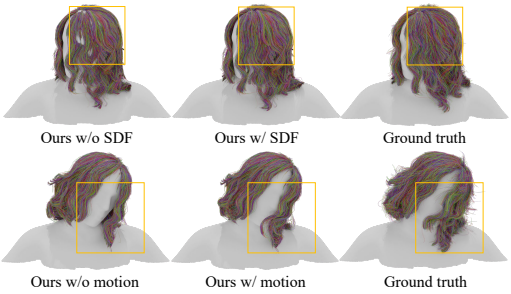

Figure 9. **Ablation on dynamics.** Top: hair w/wo SDF. Bottom: motion model effect.

to occlusions, ensuring realistic rendering, while our hair deformation model can further enhance dynamic hair realism. Our dynamic Gaussian hair representation is flexible, allowing integration with body Gaussian primitives for avatar reanimation and high-quality rendering. Additional visual results are provided in the supplemental video.

4.2 Ablation Study

• ***Dynamic Hair Modeling.*** Fig. 9 shows how our proposed approaches impact hair deformation and dynamics. **Ours w/ SDF** uses our hair-to-body mesh SDF constraint during training, reducing hair penetration into the body mesh. **Ours w/ motion** shows enhanced hair dynamics effects, such as inertia and gravity, due to the hair motion model. **Ours w/o SDF** removes the SDF constraint, **Ours w/o motion** uses only the coarse hair deformation model, and **Ours w/o atten** removes the cross-attention component in our fine stage. These results indicate that our full coarse-to-fine approach achieves the best quality. Detailed quantitative results in Tab. 3 confirm this.

To evaluate temporal consistency, Fig. 7 plots the flow vector error across a sequence for different settings. While our coarse stage shows similar performance to rigid hair in terms of dynamics, the fine model significantly improves motion effects. Additionally, the attention module further enhances hair dynamics and temporal consistency. • ***Dynamic Hair Appearance Optimization***. Fig. 8 compares different configurations of our appearance model: Ours without tangent features or curvature-based blending (**w/o tan.&blend**), ours without curvature-based blending (**w/o blend**), and our full model (**Ours full**). It demonstrates that conditioning on structural features like hair tangent vectors enhances local detail and sharpness, while curvature-based blending further improves smoothness and scattering between hair segments, creating more realistic transitions. As indicated in Tab. 4, incorporating tangent features improves natural hair perception, while curvature blending significantly enhances rendering quality, resulting in more visually appealing hair.

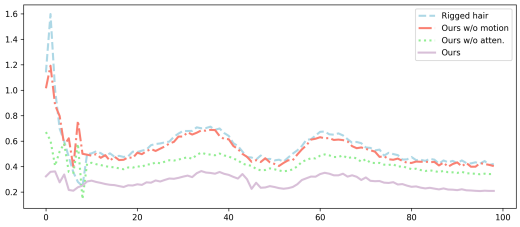

Figure 7. **Hair dynamics ablation.** Flow error ($\times 10^{-3}$) per timestep for different settings.

5 Conclusion

We propose **Dynamic Gaussian Hair** (DGH), a novel data-driven framework for dynamic hair generation with animatable 3D Gaussians. DGH models a wide range of hairstyle deformation driven by arbitrary head motions (including long hair, curly hair, ponytails, etc.), handling dynamics and hair-body collision, within a coarse-to-fine strategy. Hair appearance is represented by motion-dependent connected Gaussians to handle variations under intricate motions at render time, enabling high-fidelity dynamic hair modeling and rendering. Future work and limitations are addressed in the Appendices.

6 Acknowledgment

We sincerely thank Gene Lin for his help and discussions regarding the synthetic data generation pipeline, and we also appreciate Yu Ding for his assistance with dynamic hair and body merging.

References

[1] Stephen R Marschner, Henrik Wann Jensen, Mike Cammarano, Steve Worley, and Pat Hanrahan. Light scattering from human hair fibers. *ACM Transactions on Graphics (TOG)*, 22(3):780–791, 2003. 2, 6

[2] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2, 4, 18

[3] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. *ICCV*, 2021. 2, 4

[4] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. *arXiv preprint arXiv:2312.02069*, 2023. 2

[5] Yuelang Xu, Benwang Chen, Zhe Li, Hongwen Zhang, Lizhen Wang, Zerong Zheng, and Yebin Liu. Gaussian head avatar: Ultra high-fidelity head avatar via dynamic gaussians. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 4

[6] Alfredo Rivero, ShahRukh Athar, Zhixin Shu, and Dimitris Samaras. Rig3dgs: Creating controllable portraits from casual monocular videos. *arXiv preprint arXiv:2402.03723*, 2024. 4

[7] Helisa Dhamo, Yinyu Nie, Arthur Moreau, Jifei Song, Richard Shaw, Yiren Zhou, and Eduardo Pérez-Pellitero. Headgas: Real-time animatable head avatars via 3d gaussian splatting. In *European Conference on Computer Vision*, pages 459–476. Springer, 2024. 2

[8] Haimin Luo, Min Ouyang, Zijun Zhao, Suyi Jiang, Longwen Zhang, Qixuan Zhang, Wei Yang, Lan Xu, and Jingyi Yu. Gaussianhair: Hair modeling and rendering with light-aware gaussians. *arXiv preprint arXiv:2402.10483*, 2024. 2, 4, 6

[9] Egor Zakharov, Vanessa Sklyarova, Michael Black, Giljoo Nam, Justus Thies, and Otmar Hilliges. Human hair reconstruction with strand-aligned 3d gaussians. In *European Conference on Computer Vision*, pages 409–425. Springer, 2024. 2, 4, 6, 8, 9, 18, 21

[10] Liwen Hu, Chongyang Ma, Linjie Luo, Li-Yi Wei, and Hao Li. Capturing braided hairstyles. *ACM Transactions on Graphics (TOG)*, 33(6):1–9, 2014. 3

[11] Fei Shao, Xingce Wang, Qianqian Jiang, Zhongke Wu, and Mingquan Zhou. Modeling curly hair based on static super-helices. In *2015 International Conference on Cyberworlds (CW)*, pages 306–313. IEEE, 2015. 3

[12] Yuxiao Zhou, Menglei Chai, Alessandro Pepe, Markus Gross, and Thabo Beeler. Groomgen: A high-quality generative hair model using hierarchical latent representations. *ACM Transactions on Graphics (TOG)*, 42(6):1–16, 2023. 3

[13] Menglei Chai, Linjie Luo, Kalyan Sunkavalli, Nathan Carr, Sunil Hadap, and Kun Zhou. High-quality hair modeling from a single portrait photo. *ACM Trans. Graph.*, 34(6):204–1, 2015. 3

[14] Menglei Chai, Tianjia Shao, Hongzhi Wu, Yanlin Weng, and Kun Zhou. Autohair: Fully automatic hair modeling from a single image. *ACM Transactions on Graphics*, 35(4), 2016.

[15] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Single-view hair modeling using a hairstyle database. *ACM Transactions on Graphics (ToG)*, 34(4):1–9, 2015. 3

[16] Sylvain Paris, Hector M Briceno, and François X Sillion. Capture of hair geometry from multiple images. *ACM transactions on graphics (TOG)*, 23(3):712–719, 2004. 3

[17] Yichen Wei, Eyal Ofek, Long Quan, and Heung-Yeung Shum. Modeling hair from multiple views. In *ACM SIGGRAPH 2005 Papers*, pages 816–820, 2005.

[18] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Robust hair capture using simulated examples. *ACM Transactions on Graphics (TOG)*, 33(4):1–10, 2014. 3

[19] Yuxiao Zhou, Menglei Chai, Daoye Wang, Sebastian Winberg, Erroll Wood, Kripasindhu Sarkar, Markus Gross, and Thabo Beeler. Groomcap: High-fidelity prior-free hair capture. *arXiv preprint arXiv:2409.00831*, 2024. 3

[20] Linjie Luo, Hao Li, Sylvain Paris, Thibaut Weise, Mark Pauly, and Szymon Rusinkiewicz. Multi-view hair capture using orientation fields. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1490–1497. IEEE, 2012. 3

[21] Radu Alexandru Rosu, Shunsuke Saito, Ziyan Wang, Chenglei Wu, Sven Behnke, and Giljoo Nam. Neural strands: Learning hair geometry and appearance from multi-view images. In *European Conference on Computer Vision*, pages 73–89. Springer, 2022. 3

[22] Darius Rückert, Linus Franke, and Marc Stamminger. Adop: Approximate differentiable one-pixel point rendering. *ACM Transactions on Graphics (ToG)*, 41(4):1–14, 2022. 3

[23] Wang Yifan, Felice Serena, Shihao Wu, Cengiz Öztireli, and Olga Sorkine-Hornung. Differentiable surface splatting for point-based geometry processing. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019. 3

[24] Keyu Wu, Yifan Ye, Lingchen Yang, Hongbo Fu, Kun Zhou, and Youyi Zheng. Neuralhdhair: Automatic high-fidelity hair modeling from a single image using implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1526–1535, 2022. 3, 4

[25] Lingchen Yang, Zefeng Shi, Youyi Zheng, and Kun Zhou. Dynamic hair modeling from monocular videos using deep neural networks. *ACM Transactions on Graphics (TOG)*, 38(6): 1–12, 2019. 3

[26] Yuefan Shen, Shunsuke Saito, Ziyan Wang, Olivier Maury, Chenglei Wu, Jessica Hodgins, Youyi Zheng, and Giljoo Nam. Ct2hair: High-fidelity 3d hair modeling using computed tomography. *ACM Transactions on Graphics (TOG)*, 42(4):1–13, 2023. 3

[27] Giljoo Nam, Chenglei Wu, Min H Kim, and Yaser Sheikh. Strand-accurate multi-view hair capture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 155–164, 2019. 3

[28] Tiancheng Sun, Giljoo Nam, Carlos Aliaga, Christophe Hery, and Ravi Ramamoorthi. Human hair inverse rendering using multi-view photometric data. In *Eurographics Symposium on Rendering (EGSR)*. The Eurographics Association, 2021. 3

[29] Zhiyi Kuang, Yiyang Chen, Hongbo Fu, Kun Zhou, and Youyi Zheng. Deepmvshair: Deep hair modeling from sparse views. In *SIGGRAPH Asia 2022 Conference Papers*, pages 1–8, 2022. 4

[30] Yusuke Takimoto, Hikari Takehara, Hiroyuki Sato, Zihao Zhu, and Bo Zheng. Dr. hair: Reconstructing scalp-connected hair strands without pre-training via differentiable rendering of line segments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20601–20611, 2024. 4

[31] Wojciech Zielonka, Timo Bolkart, and Justus Thies. Instant volumetric head avatars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4574–4584, 2023. 4

[32] Vanessa Sklyarova, Jenya Chelishev, Andreea Dogaru, Igor Medvedev, Victor Lempitsky, and Egor Zakharov. Neural haircut: Prior-guided strand-based hair reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19762–19773, 2023. 4

[33] Stephen Lombardi, Tomas Simon, Gabriel Schwartz, Michael Zollhoefer, Yaser Sheikh, and Jason Saragih. Mixture of volumetric primitives for efficient neural rendering. *ACM Trans. Graph.*, 40(4), July 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459863. URL `https://doi.org/10.1145/3450626.3459863`. 4

[34] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 4, 6, 8, 9, 21

[35] Ziyan Wang, Timur Bagautdinov, Stephen Lombardi, Tomas Simon, Jason Saragih, Jessica Hodgins, and Michael Zollhofer. Learning compositional radiance fields of dynamic human heads. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5704–5713, 2021. 4

[36] Ziyan Wang, Giljoo Nam, Aljaz Bozic, Chen Cao, Jason Saragih, Michael Zollhöfer, and Jessica Hodgins. A local appearance model for volumetric capture of diverse hairstyles. In *2024 International Conference on 3D Vision (3DV)*, pages 190–200. IEEE, 2024. 4

[37] Qing Zhang, Jing Tong, Huamin Wang, Zhigeng Pan, and Ruigang Yang. Simulation guided hair dynamics modeling from video. *Computer Graphics Forum*, 31(7):2003–2010, 2012. 4

[38] Zexiang Xu, Hsiang-Tao Wu, Lvdi Wang, Changxi Zheng, Xin Tong, and Yue Qi. Dynamic hair capture using spacetime optimization. *To appear in ACM TOG*, 33:6, 2014. 4

[39] Liwen Hu, Derek Bradley, Hao Li, and Thabo Beeler. Simulation-ready hair capture. *Computer Graphics Forum*, 36(2):281–294, 2017. 4

[40] Qing Lyu, Menglei Chai, Xiang Chen, and Kun Zhou. Real-time hair simulation with neural interpolation. *IEEE Transactions on Visualization and Computer Graphics*, 28(4):1894–1905, 2020. 4

[41] Jerry Hsu, Tongtong Wang, Zherong Pan, Xifeng Gao, Cem Yuksel, and Kui Wu. Real-time physically guided hair interpolation. *ACM Transactions on Graphics (TOG)*, 43(4):1–11, 2024.

[42] Gilles Daviet. Interactive hair simulation on the gpu using admm. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023. 4

[43] Tuur Stuyck, Gene Wei-Chin Lin, Egor Larionov, Hsiao-yu Chen, Aljaz Bozic, Nikolaos Sarafianos, and Doug Roble. Quaffure: Real-time quasi-static neural hair simulation. *arXiv preprint arXiv:2412.10061*, 2024. 4

[44] Ziyan Wang, Giljoo Nam, Tuur Stuyck, Stephen Lombardi, Michael Zollhöfer, Jessica Hodgins, and Christoph Lassner. Hvh: Learning a hybrid neural volumetric representation for dynamic hair performance capture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6143–6154, 2022. 4

[45] Ziyan Wang, Giljoo Nam, Tuur Stuyck, Stephen Lombardi, Chen Cao, Jason Saragih, Michael Zollhöfer, Jessica Hodgins, and Christoph Lassner. Neuwigs: A neural dynamic model for volumetric hair capture and animation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8641–8651, 2023. 4

[46] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20299–20309, 2024. 4

[47] Tobias Kirschstein, Javier Romero, Artem Sevastopolsky, Matthias Nießner, and Shunsuke Saito. Avat3r: Large animatable gaussian reconstruction model for high-fidelity 3d head avatars. *arXiv preprint arXiv:2502.20220*, 2025.

[48] Kartik Teotia, Hyeongwoo Kim, Pablo Garrido, Marc Habermann, Mohamed Elgharib, and Christian Theobalt. Gaussianheads: End-to-end learning of drivable gaussian head avatars from coarse-to-fine representations. *ACM Transactions on Graphics (TOG)*, 43(6):1–12, 2024. 4

[49] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. 5

[50] Matthias Zwicker, Hanspeter Pfister, Jeroen Van Baar, and Markus Gross. Surface splatting. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 371–378, 2001. 6

[51] Eugene d'Eon, Guillaume Francois, Martin Hill, Joe Letteri, and Jean-Marie Aubry. An energy-conserving hair reflectance model. *Computer Graphics Forum*, 30(4):1181–1187, 2011. 6

[52] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 6, 9

[53] Miles Macklin, Matthias Müller, and Nuttapong Chentanez. Xpbd: position-based simulation of compliant constrained dynamics. In *Proceedings of the 9th International Conference on Motion in Games*, pages 49–54, 2016. 7, 9, 15, 19

[54] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010. 9

[55] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017. 9

[56] Adobe Systems Inc. Mixamo. `https://www.mixamo.com/`, 2020. Online character animation and motion capture dataset. Accessed 2025-10-21. 9

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 18

[58] NVIDIA Corporation. *NVIDIA TensorRT: High-Performance Deep Learning Inference*, 2025. URL `https://developer.nvidia.com/tensorrt`. Accessed: 2025-03-05. 19

[59] Xueting Li, Ye Yuan, Shalini De Mello, Gilles Daviet, Jonathan Leaf, Miles Macklin, Jan Kautz, and Umar Iqbal. Simavatar: Simulation-ready avatars with layered hair and clothing. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 26320–26330, 2025. 21

[60] Chengan He, Jorge Alejandro Amador Herrera, Yi Zhou, Zhixin Shu, Xin Sun, Yao Feng, Sören Pirk, Dominik L Michels, Meng Zhang, Tuanfeng Y Wang, and Holly Rushmeier. Digital salon: An ai and physics-driven tool for 3d hair grooming and simulation. *ACM SIGGRAPH Asia 2024 Real-Time Live!*, 2024. URL `https://digital-salon.github.io/`. 22

[61] Chengan He, Xin Sun, Zhixin Shu, Fujun Luan, Sören Pirk, Jorge Alejandro Amador Herrera, Dominik L Michels, Tuanfeng Y Wang, Meng Zhang, Holly Rushmeier, and Yi Zhou. Perm: A parametric representation for multi-style 3d hair modeling. *arXiv preprint arXiv:2407.19451*, 2024. 22

[62] Thomas Müller, Fabrice Rousselle, Jan Novák, and Alexander Keller. Real-time neural radiance caching for path tracing. *arXiv preprint arXiv:2106.12372*, 2021. 22

[63] Stephan J Garbin, Marek Kowalski, Matthew Johnson, Jamie Shotton, and Julien Valentin. Fastnerf: High-fidelity neural rendering at 200fps. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 14346–14355, 2021. 22

- Appendices -

In this document, we provide more details for the dataset, method, experiments, and more qualitative results, as an extension of Sec. 3 and Sec. 4 in the main paper. Please also refer to the video demo for dynamic hair results, comparison, ablation study, and more results.

A Dataset Details

As mentioned in the Sec. 1, due to the lack of real captures of dynamic hair deformations with accurate strand tracking, we create a new synthetic dataset capturing both hair geometry and appearance. The hair dataset breakdown is shown below: in Fig. 10, we present a variety of hairstyles in our synthetic hair dataset, including curly, wavy, blowout, ponytail and etc, and we show the distributions of hairstyle types and hair lengths.

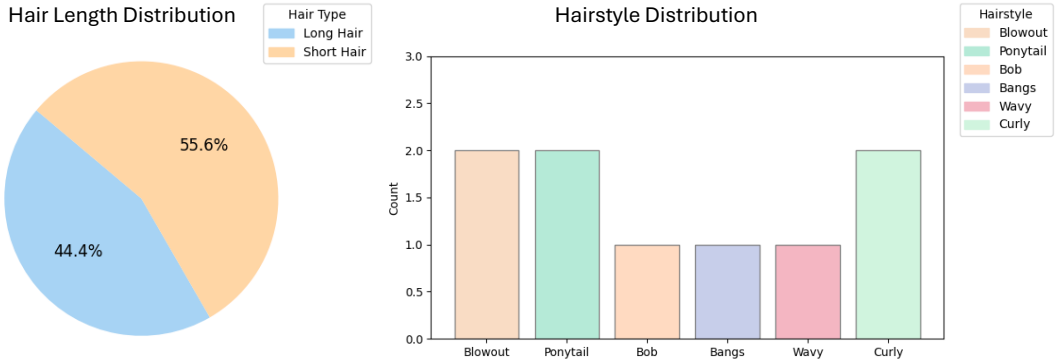

Figure 10. **Hair length and hairstyle distribution**

A.1 Geometry Dataset (Sec. 4)

To capture head motions, we rotate the head to record head movements. Instead of linear speed between two head positions, we apply spline-based interpolation to simulate varying motion speeds. This approach allows us to more effectively capture secondary hair motions when simulating the hair with head motions with realistic damping and inertial dynamics. For each hairstyle, we animated with XPBD-based physics simulation [53] driven by captured head motions, and for each hairstyle, we generate 100 motion sequences, and each motion sequence contains 100 frames. For each frame, we record the head mesh, deformed hair strands, and then post-process to get hair volume and pose volume. We show the hair geometry dataset generation pipeline in Fig. 11

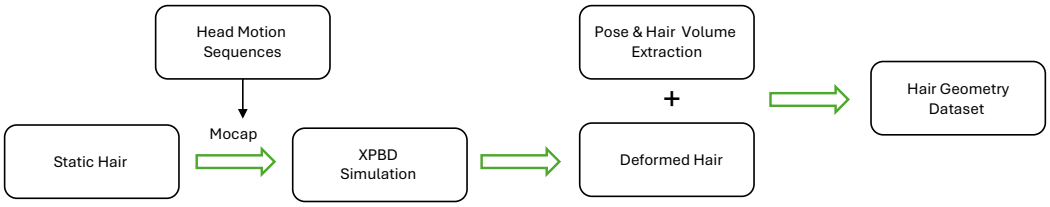

Figure 11. **Hair Geometry Dataset Generation Pipeline**

Fig. 12 presents samples from our hair deformation dataset, including static hair with strands, upper body SDF volume, static hair SDF volume, deformed hair strands, and deformed hair SDF volumes for each hair groom. For hair SDF volumes, hair point cloud (PC) is converted to an SDF grid by computing the distance from each voxel to the nearest point in the PC.

Training dataset. We train each hairstyle independently to obtain a hairstyle-specific hair deformation model. For each hairstyle, we use 90 motion subjects, resulting in a geometry training dataset of 9K frames. In the coarse stage, we randomly sample a frame and apply rigid transformations to the

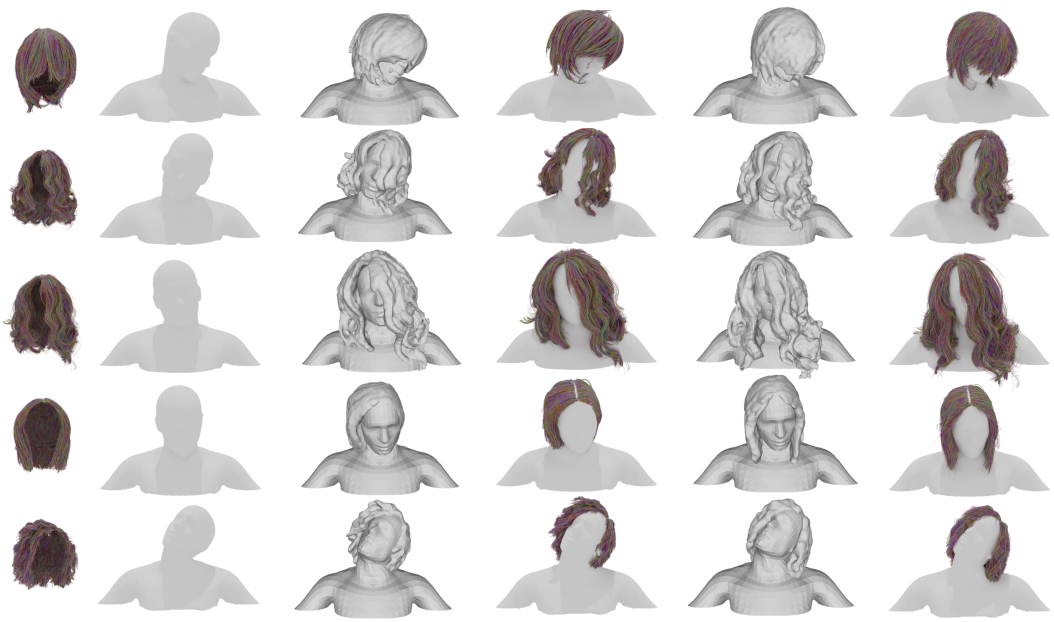

Figure 12. Training samples from the hair deformation dataset, including static hair, pose volume, rigidly transformed static hair, recorded hair volume, deformed hair, and corresponding recorded hair volume.

static hair using the sampled head pose. The network learns per-point displacements by iteratively sampling different poses. In the fine stage, we learn a flow vector field to deform the hair from frame $t-1$ to frame t, using randomly sampled points from the deformed hair at time $t-1$. Each training batch consists of 200K randomly sampled hair points.

Testing dataset. We evaluate our dynamic hair model across different hairstyles using 10 motion sequences, resulting in 1K frames. We report the average L_2 error between the predicted and ground-truth deformed hair, as well as the L_2 error of the estimated flow vectors (displacements between frames $t-1$ and t).

A.2 Appearance Dataset (Sec. 4)

Once obtaining the deformed hair sequences, we render multi-view videos of various hair grooms in Blender (particle system) under a consistent lighting setup to train our appearance model. Each frame has a resolution of 1024 $\times$1024. For each hairstyle, we include diverse hair colors. Fig. 13 (left) shows the color variations for each hairstyle, (middle) illustrates the hair appearance dataset generation pipeline, and (right) presents a sample setup demonstrating how deformed hair is positioned for multi-view rendering. Each dynamic frame includes 24 views: 12 cameras evenly distributed horizontally around the object and 12 randomly positioned on the upper semi-sphere to capture diverse angles. Fig. 14 illustrates multi-view rendering samples for different hair grooms.

Training dataset We train our motion-dependent hair appearance model using multi-view video sequences. Each sequence contains 500 motion frames, and for each frame, we render images from 24 camera viewpoints distributed over a semi-sphere. This setup yields a total of 12K training images per hairstyle.

Testing dataset To evaluate our appearance model, we generate a testing dataset consisting of 100 frames featuring unseen hair motion. The test sequence is captured using a horizontal camera rotation arc, with each camera looking at the hair center. Viewpoints are sampled across a vertical range of 0°–45° on the viewing semi-sphere, providing diverse angular coverage and challenging view-dependent appearance variations.

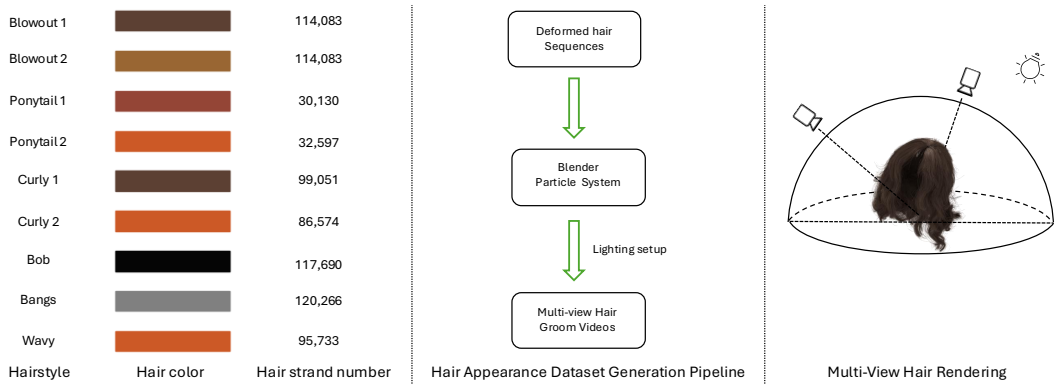

Figure 13. **Hairstyle Color Variations and Hair Appearance Dataset Generation Pipeline**

B Network and Evaluation Details

Hair, as a complex and dynamic structure, exhibits intricate spatial patterns and deformations that are challenging to model with lower-dimensional representations, the hair appearance also suffer challenges as the moving pattern caused occlusion. To address this, we propose two stage modeling: which Stage I and II as Hair Dynamics Model and Appearance Model. As summarized in Tab. 5, Stage I adopts a coarse-to-fine design: the coarse stage learns hair deformation from static hair, while the fine stage refines secondary dynamics (e.g., inertia and damping).

Module	**Function**
Stage I (coarse stage)	Learns hair deformation given head motions
Stage I (fine stage)	Adds secondary motion refinement
Stage II	Optimizes dynamic Gaussian hair appearance

Table 5. **Overview of model stages and their corresponding functions.**

For Hair Dynamics Model, we adopt a coarse-to-fine framework, representing static hair as a SDF volume to capture its complete 3D structure and spatial relationships. In this section, we present the architecture of our 3D CNN networks, $\mathcal{E}_{\text{pose}}$ and $\mathcal{E}_{\text{hair}}$, the implicit networks $\mathcal{M}$, $\mathcal{D}$, and $\mathcal{D}^*$, along with dataset and evaluation details.

B.1 Coarse-to-Fine Framework (Sec. 3.1)

In the **Coarse Stage**, we utilize a 3D U-Net architecture for $\mathcal{E}_{\text{pose}}$ and $\mathcal{E}_{\text{hair}}$ with input volumes of resolution $128 \times 128 \times 128$. We present the network structure in Fig. 15. The network employs separate encoders for static hair and pose, each comprising four convolutional blocks with filter sizes $F = 4$, $2F$, $4F$, and $8F$, utilizing 3D convolutions, LeakyReLU activations, and Instance Normalization to effectively extract hierarchical features.

The decoder mirrors the encoder with four deconvolutional blocks, progressively upsampling and reconstructing volumetric features while reducing the number of filters from $8F$ to F. Skip connections between corresponding encoder and decoder blocks preserve high-resolution details. With a base filter size of $F = 4$ and output channels of 16, the 3D U-Net effectively captures global context and fine-grained details of static hair, enabling precise learning of hair deformation by leveraging the static hair structure.

In the **Fine Stage**, the encoders $\mathcal{E}_{\text{pose}}$ and $\mathcal{E}_{\text{hair}}$ retain the same structure, with $\mathcal{E}_{\text{hair}}$ sharing weights across timesteps during training. The MLPs ($\mathcal{M}, \mathcal{D}^*$) consist of six fully connected layers with dimensions $[176, 512, 512, 256, 128, 3]$ and $[239, 512, 512, 256, 128, 3]$, respectively. Residual connections are employed to enhance feature representation by concatenating input features with intermediate outputs. Leaky ReLU is used as the activation function for all layers except the final one, ensuring non-linearity and stability. The final layer outputs a 3-dimensional vector, directly supervised by the ground truth displacement and 3D flow vector.

Figure 14. Training samples from the appearance dataset, which presents a static frame of various hair grooms and rendering results from six different camera views.

The fine stage estimates the flow vector from V_{hair}^{t-1} to V_{hair}^{t}. Since V_{hair}^{t} is not available, cross-attention [57] between V_{hair}^{t-1} and V_{hair}^{t-2} infers dynamics. "Ours w/o atten" uses only V_{hair}^{t-1} to estimate the flow vector. In the cross-attention, Q, K, V is V_{hair}^{t-2}, V_{hair}^{t-1}, V_{hair}^{t-1}. This approach leverages the immediate context from V_{hair}^{t-1} and longer-term dynamics from V_{hair}^{t-2}, enabling the model to capture motion patterns and ensure temporal consistency for a smooth and reliable estimation of V_{hair}^{t}.

B.2 Dynamic Hair Appearance Network and Implementation(Sec. 3.2)

For dynamic hair appearance optimization, the encoder $\mathcal{E}_{\text{hair}}$ encodes the deformed hair to capture global motion features and 3D spatial information. A lightweight MLP $\mathcal{D}$ with dimensions $[169, 256, 256, 256, 128, 50]$ decodes the Spherical Harmonics coefficients for color, the scale factor along the hair length direction, and opacity. Positional encoding E for the position $\mathbf{p}$, tangent vector $\mathbf{t}$, and view direction $\mathbf{d}$ follows the encoding function proposed in NeRF [2]. For Gaussian primitive initialization, we make a cylindrical Gaussian primitive attached to each hair segment with a length much greater than its radius. Similar to Gaussian Haircut [9], the scale of Gaussian primitives has only one degree of freedom, which is proportional to the length of the line segment, while the other two are fixed to a small predefined value.

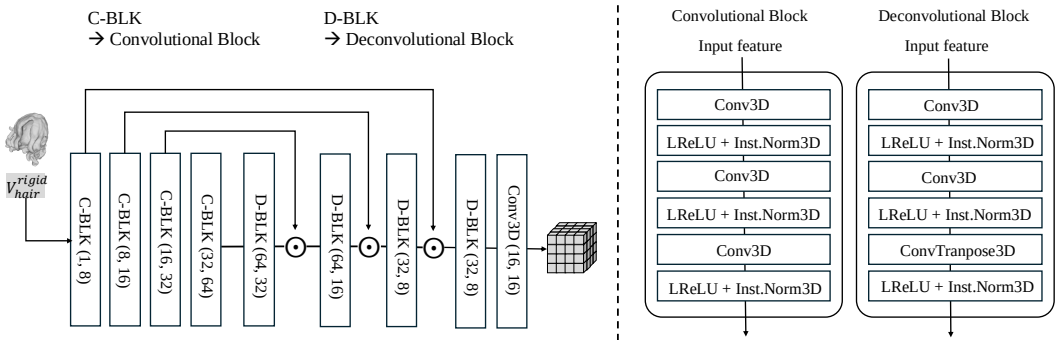

Figure 15. Left: Our 3D CNN networks for $\mathcal{E}_{\text{pose}}$ and $\mathcal{E}_{\text{hair}}$; Right: Convolutional and deconvolutional blocks.

B.3 Evaluation Details (Sec. 4)

Metrics. We evaluate hair motion over time by comparing the flow vectors between consecutive frames in the predicted and ground truth point cloud sequences. The flow vector for each point i at frame t in the predicted point cloud is defined as $\mathcal{F}_i^t = \mathbf{P}_i^{t+1} - \mathbf{P}_i^t$, where $\mathbf{P}_i$ represents the predicted point cloud positions. Similarly, the flow vector in the ground truth point cloud is defined as $\hat{\mathcal{F}}_i^t = \hat{\mathbf{P}}_i^{t+1} - \hat{\mathbf{P}}_i^t$, where $\hat{\mathbf{P}}_i$ represents the GT point cloud positions. The total flow error at each timestep t over all points in the point cloud is given by $f_{\text{error}}^t = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathcal{F}_i^t - \hat{\mathcal{F}}_i^t\right\|_2$, where N is the total number of points in the point cloud.

Method	PSNR ↑	SSIM ↑	LPIPS ↓
Ours w/o LPIPS & SSIM	27.82	0.8874	0.2100
Ours w/o LPIPS	28.01	0.8923	0.1987
Ours	**28.12**	**0.9004**	**0.1881**

Table 6. Ablation of different losses for appearance optimization.

Inference. We train and test our model on a single A100 GPU, with training times of approximately 20 hours for the dynamic hair coarse stage, 20 hours for the fine stage, and 26 hours for the appearance model. During inference, the dynamic hair model achieves 2.0 FPS for approximately 150K hair strands, significantly outperforming the XPBD [53] physics-based simulation engine that is used to generate the synthetic hair dataset, which runs at approximately 0.33 FPS. Meanwhile, dynamic hair novel view synthesis reaches 2.22 FPS, with faster speeds for fewer strands.

Runtime and memory performance analysis. DGH design is suitable for low-compute devices. Our implementation is an upper bound. Neural net inference can be accelerated with TensorRT [58] and additional strands linearly interpolated. For hair deformation runtime analysis, we report runtime with various strand numbers on RTX4090 GPU, compared with XPBD, as shown in Fig 16. Hair inference required 5GB, while high-quality grooms (150K+ strands) required 20GB. For dynamic hair rendering runtime analysis, our appearance model renders 1K images in 0.45 seconds (2.22 FPS), whereas Blender's hair shader rendering takes 6 seconds per image (0.17 FPS).

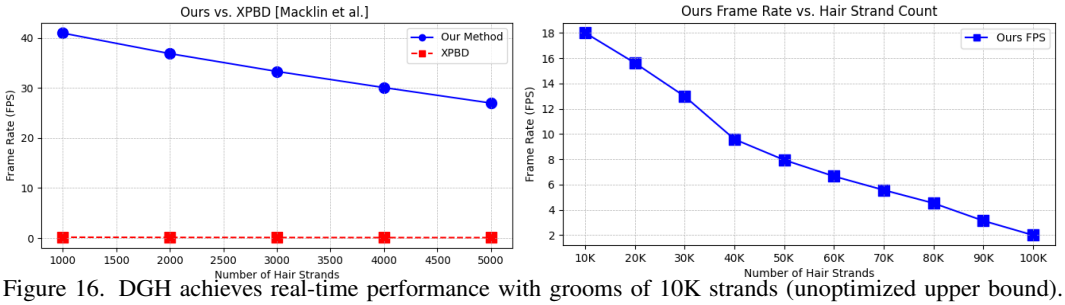

Figure 16. DGH achieves real-time performance with grooms of 10K strands (unoptimized upper bound). Physics-based simulation (XPBD) runs below 1 FPS with hair interpolation and equal high-quality photoreal rendering in Blender 3D engine.

Appearance loss ablations. We present the appearance loss ablation study in Tab. 6. For the final loss constraint, we incorporate both perceptual loss (LPIPS) and structural loss (SSIM), demonstrating that our full method achieves the best performance.

Figure 17. Hair deformation with varying strand counts. From left to right: increasing the inferred number of hair strands, totaling approximately 100K to 150K strands.

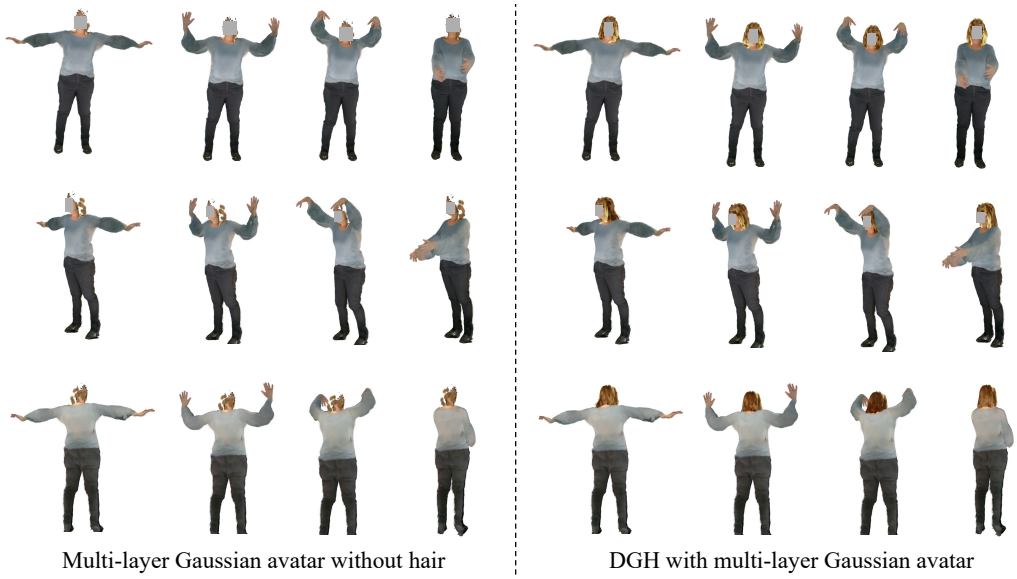

Figure 18. Our method merges the dynamic hair layer with a per-frame animatable Gaussian avatar. By employing a multi-layer Gaussian representation (left), we obtain an avatar without the hair layer. Applying DGH within this multi-layer framework enables realistic avatar re-animation (right). *Note: The human face is masked to comply with privacy regulations.*

C Qualitative Results

C.1 Hair Deformation.

In Fig. 17, we present the hair deformation test on arbitrary head poses and various hairstyles. We also present hair deformation results tested with varying numbers of hair strands, ranging from 1K to the full set of approximately 150K strands. Our volumetric implicit function robustly infers hair deformation for arbitrary strand numbers without requiring post-processing or hair interpolation.

C.2 Dynamic Hair Novel-View Synthesis.

Baseline settings. We compare our DGH model with two baseline methods: 3DGS [34] and Gaussian Haircut [9]. We train both baselines using their publicly released code on our canonical synthetic hair dataset, which includes 48 camera views covering the full hair. For 3DGS, we initialize Gaussian primitives by attaching them to the center of each hair segment. To maintain consistency with our dynamic sequences, we fix the number of Gaussians and disable density growth. We fix the opacity and positions of Gaussians, compute their orientation using hair tangents, and optimize three degrees of freedom of scale, color (represented using spherical harmonics). We use the Adam optimizer with a learning rate of 0.00025 for SH color features and 0.0005 for scale optimization. For Gaussian Haircut, we similarly optimize both color and scale, and the scale is constrained to be proportional to the length of each line segment to better capture hair geometry. Once each baseline model is trained on the static canonical hairstyle, we propagate the learned Gaussian parameters to dynamic sequences using our hair tracking model, which provides per-frame hair geometry.

Comparison and additional results. We present further results on dynamic hair novel-view synthesis in Fig. 19 and Fig. 20. To better evaluate appearance quality, we use ground-truth hair tracking and compare our appearance model against other Gaussian-based hair representation methods. Results are shown across multiple dynamic sequences and diverse hairstyles.

C.3 Multi-Layer Gaussian Avatars with Dynamic Hair.

Our dynamic hair Gaussian representation can be seamlessly integrated with pre-trained Gaussian avatar models for avatar re-animation and novel-view rendering, as shown in Fig. 21, 22, 23, 24 the render image resolution is 1024×1024. Our appearance model supports only hair rendering, while the body is rendered using a separate pre-trained Gaussian model. For visualization, both hair and body Gaussians are rendered together. **Note that the static hair lacks gravity effects.** Using our hair tracking model, we represent hair strands as connected cylindrical Gaussians and achieve high-quality novel view synthesis through our motion-dependent appearance model. Unlike static hair, our dynamic hair model implicitly learns gravity effects from the dataset distribution, ensuring realistic rendering across dynamic hair motion sequences.

In Fig. 18, we further demonstrate how DGH can be seamlessly integrated with a multi-layer Gaussian avatar for realistic avatar re-animation. Since our DGH model primarily takes head motion as input, the resulting motion focuses on head-driven hair dynamics without incorporating body motion effects. Inspired by recent advancements in multi-layer Gaussian avatar modeling [59], realistic avatar editing and re-animation can be achieved by introducing additional Gaussian layers to represent various components such as hair and clothing. To merge an additional hair layer with the body avatar, our framework begins with a pre-trained head GS avatar re-animated using head motion inputs. The DGH module predicts dynamic Gaussian hair deformations directly from a half-body GS representation or a dense point cloud, which are then merged with the head GS to produce realistic, data-driven hair motion.

In contrast, physics-based methods re-animate a GS avatar by converting it into an explicit mesh, simulating hair motion with a physics engine, rendering it through a 3D pipeline, and re-converting the result into a 3DGS format before merging. Unlike such mesh-based pipelines that require multiple conversion and rendering stages, our DGH framework operates entirely in the Gaussian domain. By representing both hair and the upper body as volumetric Gaussians, it enables mesh-free deformation prediction, ensuring compatibility with Gaussian-based avatars and seamless integration into learning-based re-animation pipelines without rigging or simulation overhead.

D Limitation and Future Work

DGH currently handles hair-upper-body collisions. Environmental constraints and relighting of DGH (via albedo inference) could be handled in future work. Due to the challenges of accurately tracking real hair dynamics, precise deformation and per-timestep positions of hair strands are difficult to record. Our model is currently trained and evaluated on synthetic data, focusing on generalizing to head motions. In future work, we aim to create a large dynamic hair dataset based on static 3D hair datasets [60, 61], leveraging a learned deformed hair prior. This identity-independent model could robustly infer diverse hairstyles and perform well on real-world data.

In the second stage, a lightweight MLP combined with a differentiable rasterizer achieves high-fidelity novel view rendering, balancing efficiency and quality. While fast for dense Gaussian primatives, the current version is not real-time for dense hair (100K-150K hair strands). Future work could focus on optimizing the MLP or leveraging caching methods [62, 63], such as viewpoint-aware or temporal caching, to achieve real-time performance.

For multi-layer Gaussian avatars with dynamic hair applications, our model currently supports integration with head avatars driven solely by head motion. Full-body effects such as jumping and squirming under gravity are not yet modeled. In future work, we plan to incorporate explicit material conditioning and full-body motion inputs to improve generalization across diverse hairstyles, particularly with access to a more comprehensive 3D hairstyle dataset.

E Broader Impacts

As a positive impact, our work enhances the realism of Gaussian-based digital humans, enabling more expressive virtual interactions in telepresence, AR/VR, and virtual production. As a negative impact, increased photorealism may also raise ethical concerns, including potential misuse for identity manipulation or deceptive content generation.

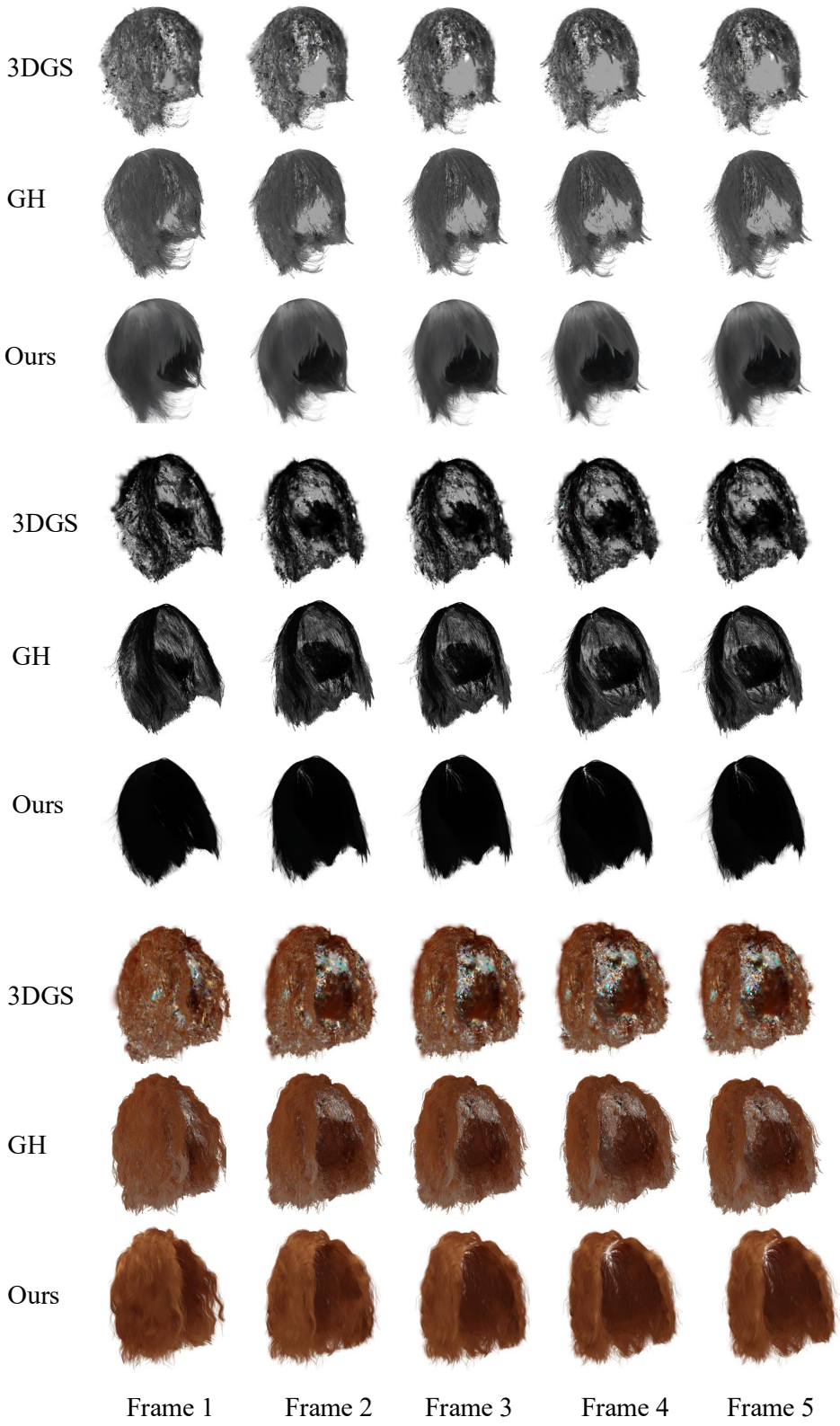

Figure 19. **Comparison results for dynamic hair rendering**

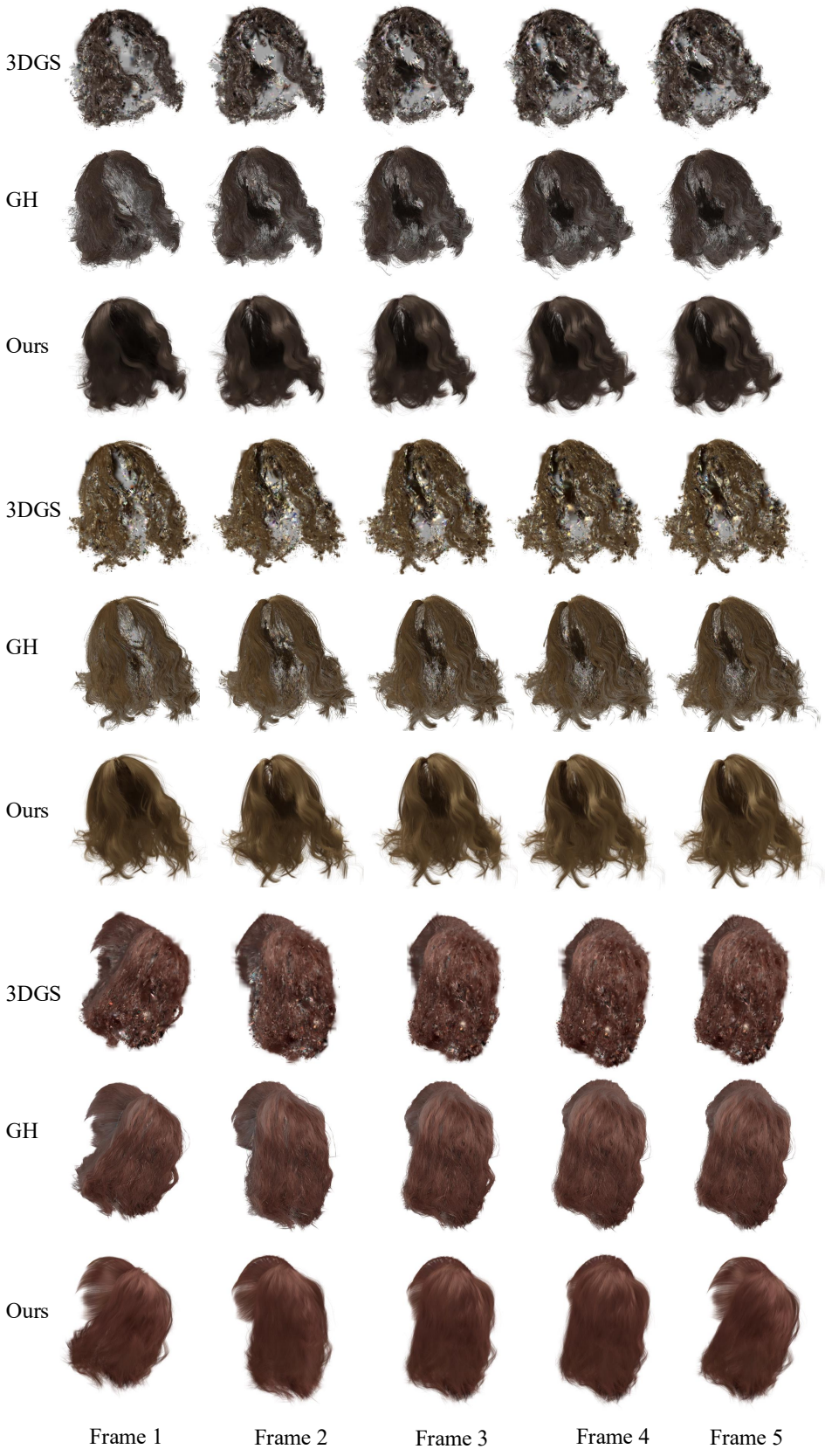

Figure 20. **Comparison results for dynamic hair rendering**

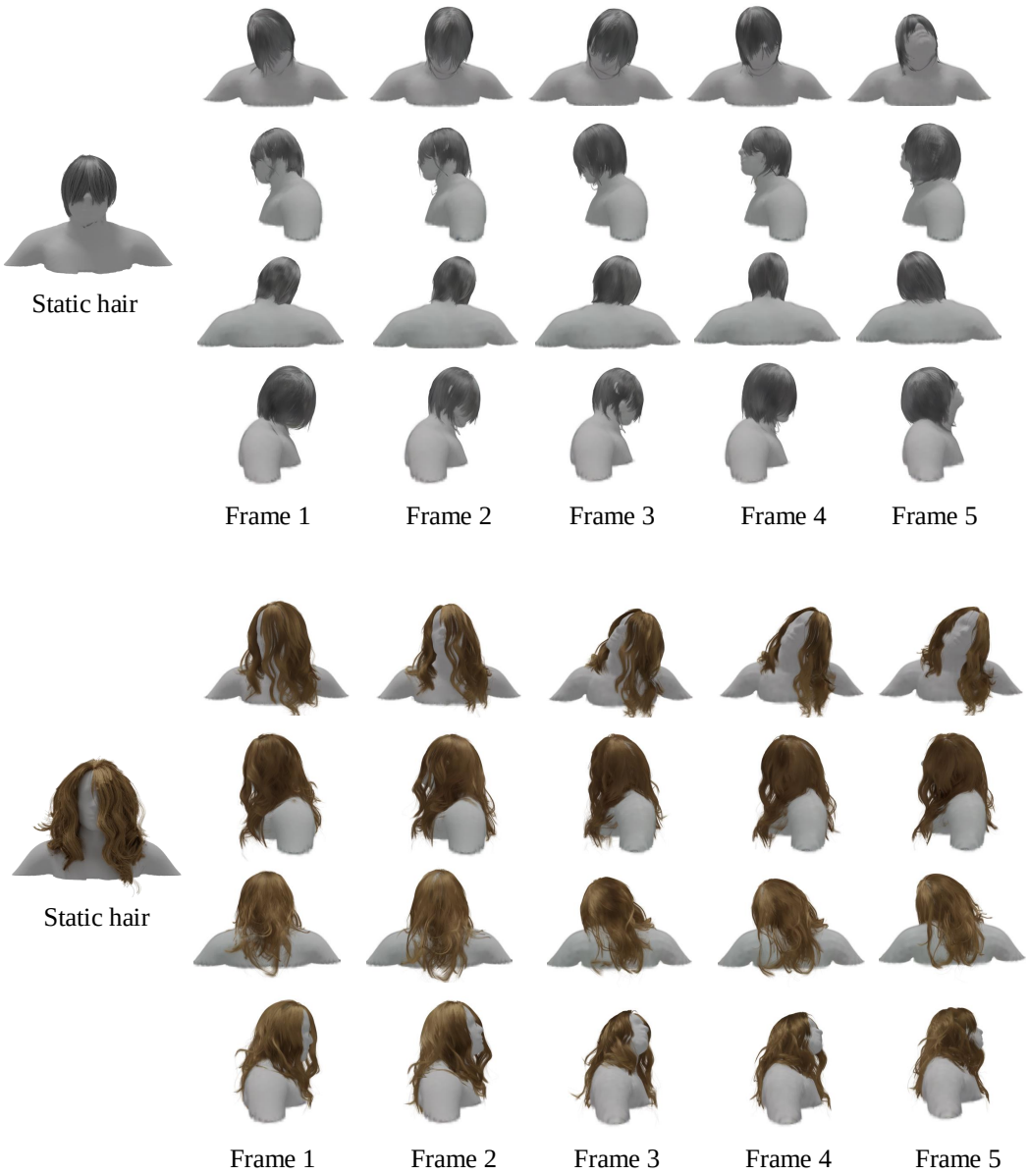

Figure 21. Our dynamic Gaussian hair appearance model enables novel view synthesis from static hair using our hair tracking model. Note: Static hair lacks gravity, while the dynamic model implicitly learns gravity from the dataset distribution.

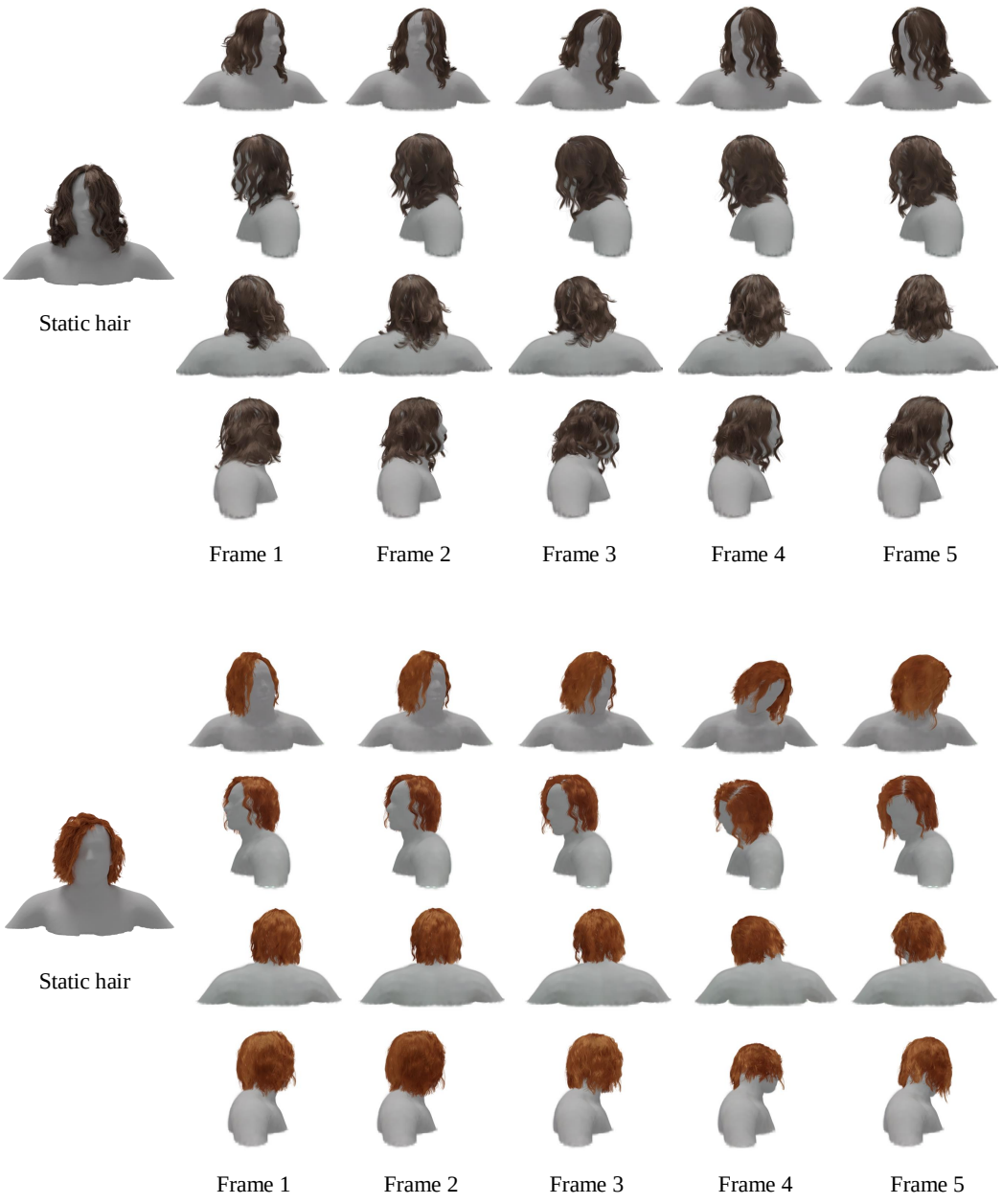

Figure 22. Novel view synthesis results from static hair using our hair tracking model across various hair grooms.

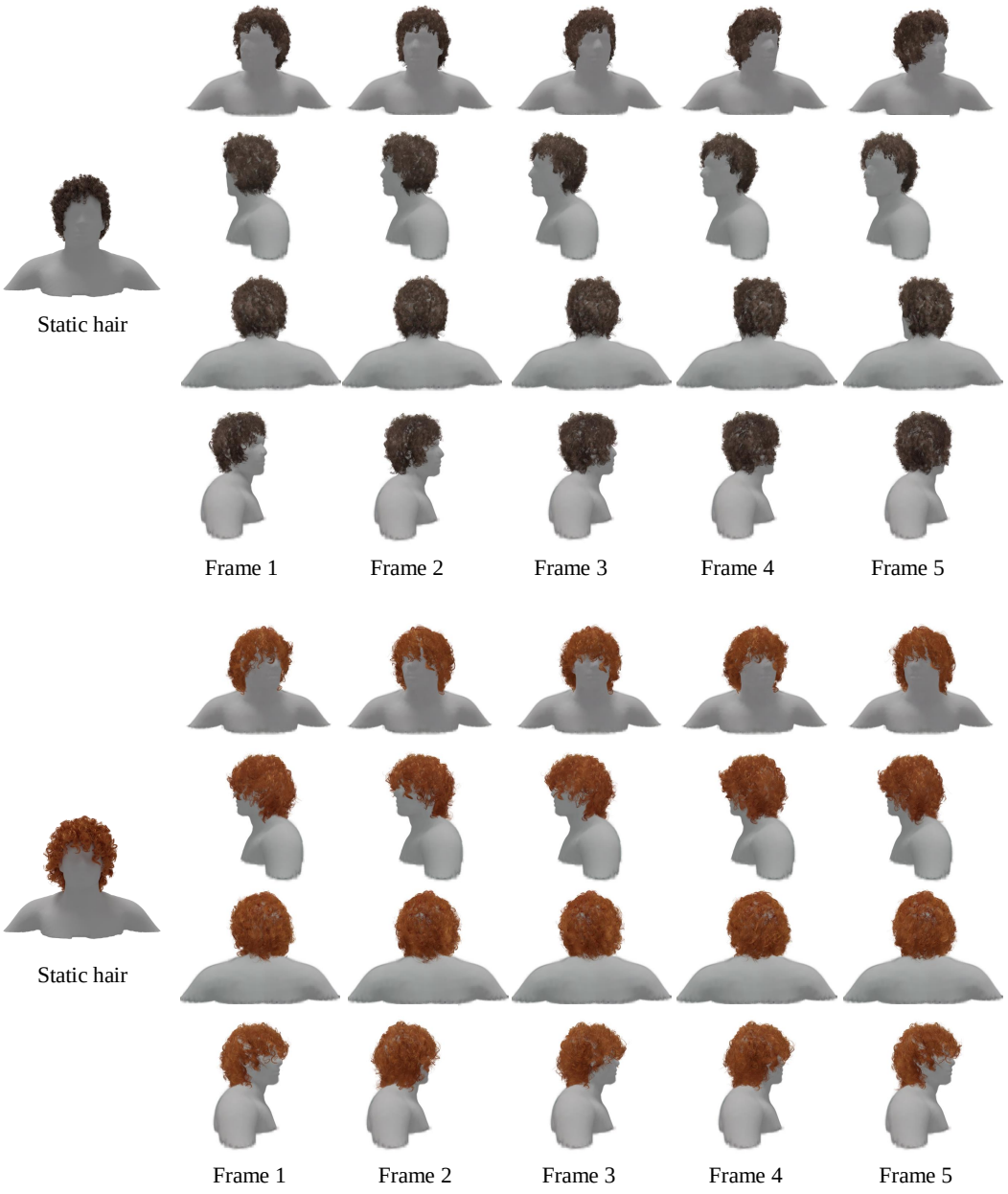

Figure 23. Given our hair tracking model and body pose, our framework enables reanimation and novel-view rendering with hair dynamics, including curly hair.

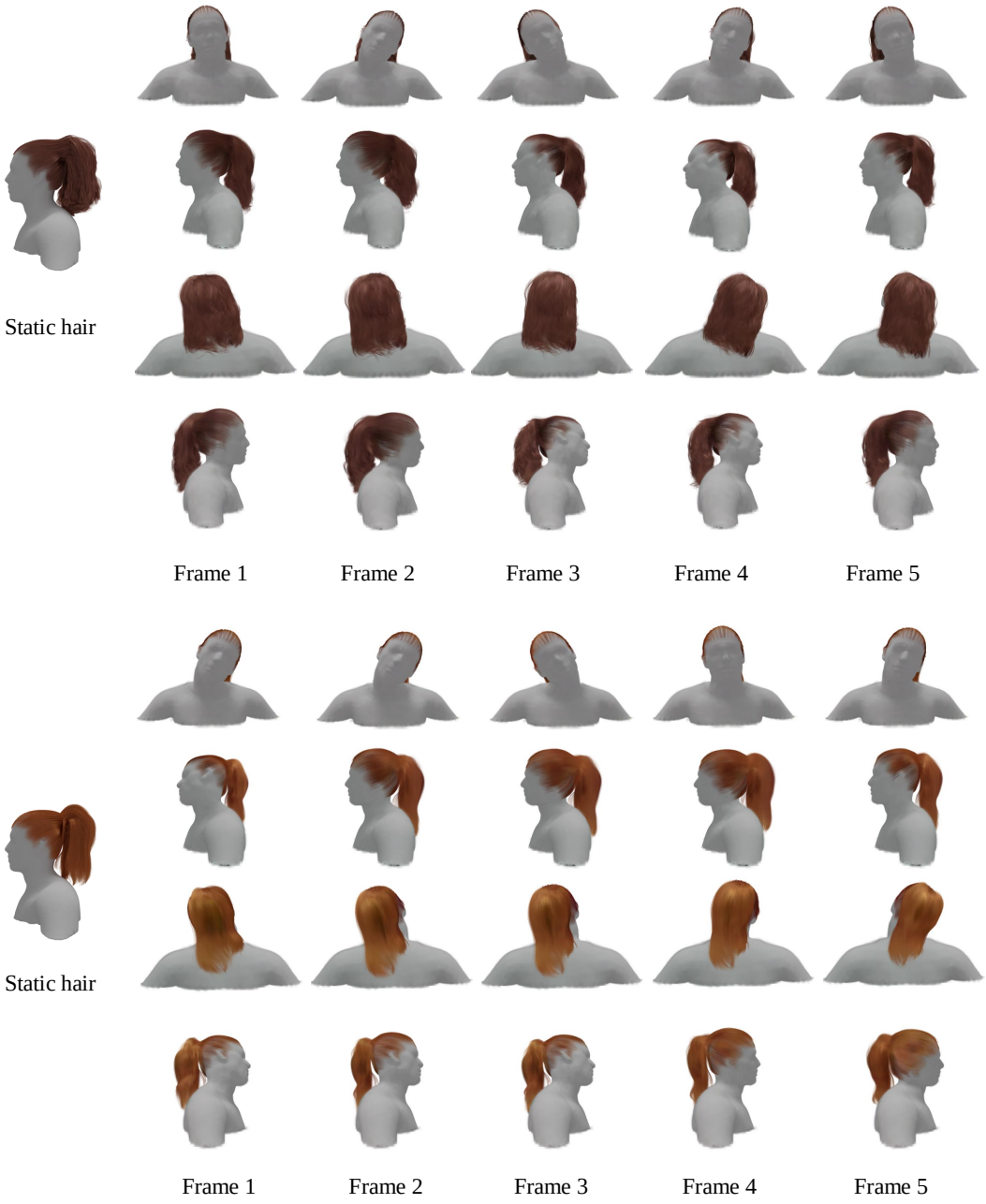

Figure 24. Given our hair tracking model and body pose, our framework enables reanimation and novel-view rendering with hair dynamics, including ponytail hairstyles.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: It presents our novel framework, Dynamic Gaussian Hair, which includes coarse-to-fine hair dynamics modeling and dynamic hair appearance optimization.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: We included the limitations at the end of Appendices document.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

Justification: We state the main assumption in the Introduction and Methods, and provide their proof in the Experiments as a form of ablation study and comparison.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide all technical details, including the model, inference algorithm, training, loss functions, and dataset information. Additional network and evaluation details are provided in the Appendices.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [NA]

Justification: The main contribution of this work lies in the proposed method rather than the dataset or experiments, however, we plan to release our dataset in the future to support further research in dynamic hair modeling.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We provide the training and testing data details in the Experiments section of the main paper. Hyperparameters and training settings are described in the Implementation Details, with additional dataset information included in the Appendices.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: We present the errors in both numerical tables and plots, included in the main paper and the Appendices.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide sufficient information about the computational resources in the main paper Implementation Details as well as the Appendices.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This paper complies with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We discuss boarder positive and negative impacts at the end of the Appendices.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Our paper poses no such risks.

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We provide license details in the experiments and properly cite all datasets and code used from prior work.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: This paper does not release new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor direct research with human subjects

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: This paper does not involve crowdsourcing nor direct research with human subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.

- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: The core method development in this research does not involve LLMs as any important, original, or non-standard components.

    Guidelines:

    - The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.