

# FHAvatar: Fast and High-Fidelity Reconstruction of Face-and-Hair Composable 3D Head Avatar from Few Casual Captures

## Supplementary Material

### A. Implementation Details

In this section, we provide further details on the data pre-processing, model architecture, and experimental settings to facilitate reproducibility.

#### A.1. Data Pre-Processing

For data with varying capture conditions, our data pre-processing pipeline consists of four steps: (1) background removal, using the faster BackgroundMattingV2 [39] on multi-view datasets with clear boundaries such as NeRSemble [30], and the more robust Sapiens [27] on user-captured monocular images and videos; (2) keypoint detection, using STAR [82] to obtain facial landmarks and locate the face; (3) FLAME [36] parameters tracking following VHAP [51]; (4) based on tracking results, mesh projection and further cropping of the facial region to remove shoulders and torso.

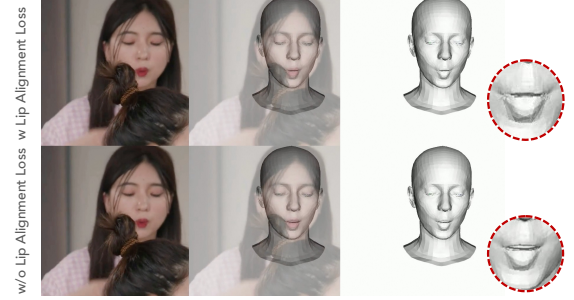
To better match the diverse real-world data captured by smartphones, we improve the original VHAP with three modifications: First, we discard the global FLAME head rotation and only optimize the neck rotation, which avoids entanglement between global and local pose and makes the recovered head motion easier to reuse with full-body models such as SMPL [40]. Second, we adopt a motion-aware iteration schedule: the number of optimization steps per iteration is increased when the average distance between the current landmarks and the reference frontal frame landmarks (determined by the distance between the eye landmarks) is large, and kept close to a small base value for iterations with little movement. Concretely, given the average distance  $d_{lmk}$  between the landmarks relative to the reference frontal frame, we compute a per-iteration proposal as follows:

$$N_t^{cur} = \begin{cases} N_0 + \lfloor \Delta (d_{lmk} - d_{th}) \rfloor, & d_{lmk} > d_{th}, \\ N_0, & \text{otherwise,} \end{cases} \quad (14)$$

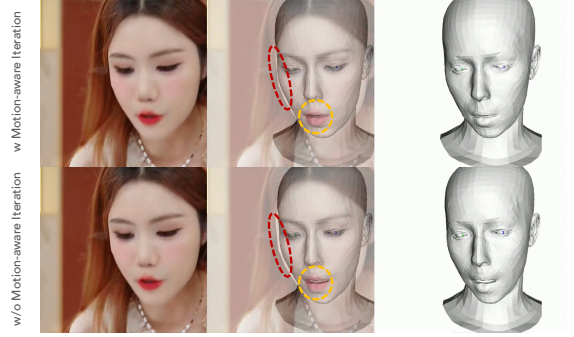
where  $d_{th}$  is a distance threshold,  $N_0$  is the base number of iterations, and  $\Delta$  is a scaling factor. For monocular video, we then apply exponential smoothing with an upper bound:

$$N_t = \min(\lfloor \lambda N_{t-1} + (1 - \lambda) N_t^{cur} \rfloor, N_{max}), \quad (15)$$

where  $\lambda$  is the smoothing weight, and  $N_{max}$  is the maximum iteration budget. This allows the tracker to better follow rapid head movements while keeping the overall computational cost moderate.



(a) Effect of lip alignment loss.



(b) Effect of motion-aware iteration schedule.

Figure 8. **Effects of our preprocessing improvements.** These modifications make the lip movement and facial shape tracking more accurate.

Third, for the authenticity of novel expression reenactments, details such as eye closure and lip alignment are crucial. We introduce two additional structural perceptual loss terms: the eye closure loss  $\mathcal{L}_{eye}$  and the lip alignment loss  $\mathcal{L}_{lip}$ . These losses not only enhance the supervision of landmark position errors but also penalize the discrepancy in distances between corresponding eye and lip landmarks.

$$\mathcal{L}_{eye} = \gamma \cdot \left\| \hat{\mathbf{K}}_{eye} - \mathbf{K}_{eye} \right\|_2 + (1 - \gamma) \cdot \left| \hat{d}_{eye} - d_{eye} \right|, \quad (16)$$

$$\mathcal{L}_{lip} = \gamma \cdot \left\| \hat{\mathbf{K}}_{lip} - \mathbf{K}_{lip} \right\|_2 + (1 - \gamma) \cdot \left| \hat{d}_{lip} - d_{lip} \right|, \quad (17)$$

where  $\gamma = 0.95$ .  $\mathbf{K}_{lip}$ ,  $\hat{\mathbf{K}}_{lip}$ ,  $\mathbf{K}_{eye}$ , and  $\hat{\mathbf{K}}_{eye}$  represent the ground truth and predicted landmark vectors for the lip and eye regions in both the original image and the rendered FLAME model.  $d_{eye}$  and  $d_{lip}$  correspond to the distances between the left and right eyes and the upper and lower lips, respectively.

## A.2. Network Architecture

**Adaptive Hair Sampler.** In generating strand Gaussians for hair, we introduce an adaptive sampler strategy that uses a varying number of Gaussians for different hairstyles, reducing rendering pressure while maintaining quality. This optimization is achieved by downsampling both the total number of strands and the number of Gaussians per strand simultaneously:

For the aggregated hair token  $\mathbf{T}_{\text{hair}}$ , we first decode it into  $S = 256$  direction vectors and corresponding vertex positions for each strand, as described in Eq. (6). From this, we can compute the average strand length  $\|\bar{\mathbf{d}}\|_2$ , classifying it into short, medium, and long hair categories. Then we apply the following steps: (1) The last dimension of the raw DiffLocks [55] feature  $f_{\text{hair}}$  is treated as a coarse density map, which is scaled and adjusted under different hair lengths. A density-based mask is then used to randomly sample and remove excess strands, effectively reducing the total number of strands in a scalp-aware manner; (2) Based on pre-defined hyperparameters for base vertices number  $S_0 = 24$  and base radius  $r_0$ , we use a piecewise linear relationship to compute new vertex count  $S'$  and Gaussian radius  $r'$ . Sampling vertices at intervals reduces the number of Gaussians per strand. It is noted that the strand Gaussian radius increases linearly with the average strand length for short hair, maintaining scalp coverage, though there is no strict growth relationship between categories of different lengths.

The computation is performed at the full UV map scale. While patch-scale adaption provides more refined regional control, the improvement is minimal and comes at the cost of increased forward time.

## A.3. Experiment Details

**Data Selection and Augmentation.** During training, for each sample, we randomly select between 1 and 6 images with different views and expressions as input, and 4 images with the same expression as supervision. This means that within a single iteration, the input images have varying expressions, while the supervision images maintain the same expression, allowing the Gaussians reconstructed from casual captures to require only one pose and expression transformation per iteration. To enhance model generalization, we apply random data augmentations, including small perturbations to brightness, contrast, and saturation. The probability of triggering augmentations is set to 0.6, with upper and lower thresholds of 0.2, 0.15, and 0.15 respectively.

During the optional refinement stage, we perform fine-tuning and supervision only on the few input images. The intermediate results from encoding and the forward pass of the transformer backbone are saved and fixed, with subsequent fine-tuning running the bidirectional forward pass only on the Gaussian decoder and renderer. For cases with more than 6 images, we randomly select multiple sets of 6

images, calculate and save the intermediate results for each set, ensuring that all images are used.

**Training Setup.** Our model predicts a set of planar or strand-based Gaussians for each UV pixel. The total number of Gaussians can be controlled by adjusting the size of the texture map. In our experiments, the head UV has dimensions  $H_{\text{uv}}^{\text{head}} = W_{\text{uv}}^{\text{head}} = 224$ , while the hair scalp UV has dimensions  $H_{\text{uv}}^{\text{scalp}} = W_{\text{uv}}^{\text{scalp}} = 112$ .

For training, we employ the Distributed Data Parallel (DDP) strategy for multi-GPU training, with L2 norm gradient clipping set to 1 for all learnable parameters in each iteration. A cosine learning rate scheduler is used during the training phase with 600 warm-up iterations, while a linear learning rate scheduler with a decay factor of 0.1 is applied during the fine-tuning phase for 100 epochs. The initial learning rate for both phases is set to  $1e-4$ .

## B. More Results

**Results on Tracking Improvements.** Fig. 8 shows the improvement in FLAME tracking due to lip alignment loss and motion-aware iteration. The test images are monocular data captured by mobile phones from the internet. The addition of lip alignment loss ensures that the lips of the tracked FLAME model close accurately when the person closes their mouth, while the motion-aware iteration schedule improves the shape accuracy, better aligning with the facial and lip shapes. This provides a good prior for avatar reconstruction and new expression reenactment.

**Results on Hair-Face Disentanglement.** We separate face and hair using semantic masks [27] and report region-wise results under 6-input setting in Tab. 3. Our method achieves the best PSNR, SSIM, and LPIPS for each region independently. Moreover, we apply face parsing to both the rendered results and the ground truth and compute region-wise IoU, where our method achieves 0.92 ( $\uparrow 6.7\%$ ) and 0.83 ( $\uparrow 36\%$ ) for the face and hair regions, respectively, validating the effectiveness of our hair-face disentanglement.

**Challenging Results.** Additional rendering results of uncovered novel viewpoints under more challenging settings, including cases without near-frontal input and with sparse input, are shown in Fig. 9, further demonstrating the robustness of the proposed method.

**Adaptive Hair Branch** We conduct additional experiments on the adaptive hair branch, starting with parameter selection. The number of hair Gaussians  $N$  is jointly determined by the scalp UV resolution  $H_{\text{scalp}}^{\text{uv}}$  and the per-strand segment count  $S' \propto S_0$ , with detailed definitions in

Table 3. **Face-Region and Hair-Region Evaluation Results.**

Method	Overall PSNR $\uparrow$	PSNR $\uparrow$	Face (w/o Neck)			Hair			
			SSIM $\uparrow$	LPIPS $\downarrow$	IoU $\uparrow$	PSNR $\uparrow$	SSIM $\uparrow$	LPIPS $\downarrow$	IoU $\uparrow$
DiffusionRig [15]	16.55	13.78	0.671	0.359	0.506	17.50	0.807	0.216	0.419
FlashAvatar [71]	16.08	14.64	0.651	0.336	0.220	16.91	0.773	0.210	0.179
GaussianAvatars [51]	<u>23.44</u>	<u>21.48</u>	<u>0.784</u>	<u>0.200</u>	<u>0.864</u>	<u>23.54</u>	<u>0.830</u>	<u>0.163</u>	<u>0.608</u>
MeGA [64]	17.29	16.92	0.649	0.302	0.624	19.60	0.759	0.193	0.476
Ours	<b>23.71</b>	<b>24.60</b>	<b>0.880</b>	<b>0.102</b>	<b>0.922</b>	<b>24.46</b>	<b>0.867</b>	<b>0.157</b>	<b>0.826</b>

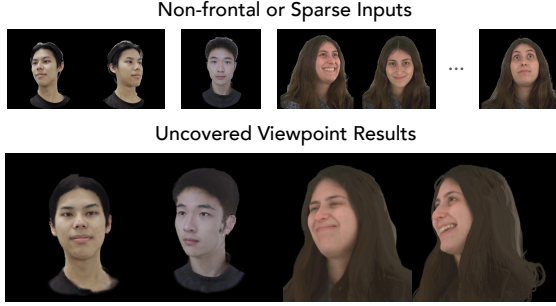


Figure 9. **Results under challenging inputs and viewpoints**

Sec. 3.1.3 and App. A.2. We have supplemented parameter sensitivity analysis. As shown in Fig. 10, we find that  $H_{\text{scalp}}^{\text{uv}} = 112$  and  $S_0 = 24$  strike a favorable balance between visual quality and memory usage.

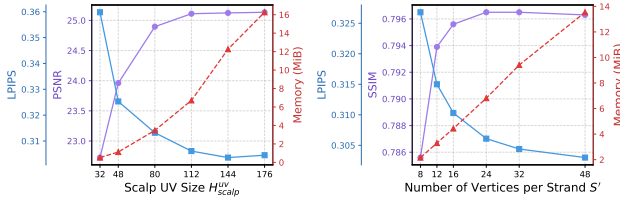


Figure 10. **Quality-Efficiency Trade-off in the Hair Branch.**

Moreover, Tab. 4 explicitly reports the hair-sampling data for the three individual cases shown in Fig. 11, where longer hair is expected to correspond to a larger number of Gaussians. Fig. 12 further provides results on long, wavy, and curly hairstyles, demonstrating the robustness of our hair-branch design.

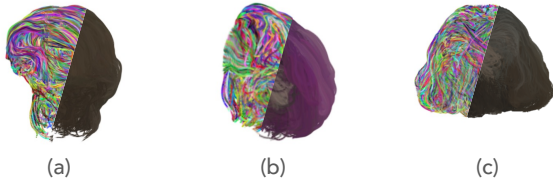


Figure 11. **Cases of Different Hairstyles:** (a) male curly hair, (b) female dyed long hair, (c) elderly female medium-length hair.

**Video Presentation.** We also provide additional results of FHAAvatar in the **attached video**, including more renderings

Table 4. **Adaptive Hair Gaussians.**

Case	$\ \bar{\mathbf{d}}\ _2$	Density Scale	$S'$	Total Gaussians
(a)	0.121	1.235	21	54080
(b)	0.228	1.000	42	84132
(c)	0.201	1.000	36	77455

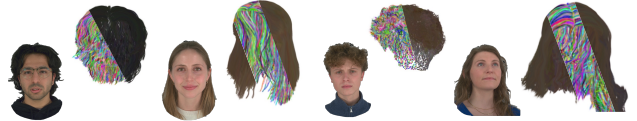


Figure 12. **Cases of Challenging Hairstyles.**

of new identities, expressions, and viewpoints on both the NeRsemble dataset [30] and real-world in-the-wild data. We also visualize the same generated avatar under different background colors to ensure objectivity and introduce greater variability.

## C. Additional Discussion

**Limitations and Future Work.** Since our reconstruction and animation of 3D Gaussian avatars are based on the FLAME model and its parameters, it remains difficult to represent static regions and dynamic details that are not modeled by FLAME, such as the tongue and fine facial wrinkles. In addition, although we achieve strand-level modeling for hair, our training set still exhibits bias in hairstyle distribution, leading to occasional failures on complex accessories or uncommon hairstyles.

For future work, the generated Gaussian avatars can be exported to TaoAvatar’s [8] optimized renderer, enabling efficient rendering and Text-to-Speech (TTS) integration on mobile devices such as Apple Vision Pro. Such integration would further support applications including online meetings, virtual companions, and VR gaming, providing a high-quality foundation for interactive digital experiences.

**Potential Social Impact.** FHAAvatar can generate realistic 3D head avatars and synthetic renderings from casual captures. As with other generative methods capable of producing lifelike digital humans, responsible use is advised to avoid unintended misuse or misrepresentation.