STAvatar: Soft Binding and Temporal Density Control for Monocular 3D Head Avatars Reconstruction

Supplementary Material

1. Overview

This supplementary material presents additional methodological details, experimental results, and an ethical discussion. Sec. 2 elaborates on offset activation, UV sampling, silhouette coefficient, and the Summed-Area Table. Sec. 3 first provides extended details on the experimental setup for the baseline methods, then visualizes the FTC results to facilitate a more comprehensive understanding, and includes additional ablation studies such as hyperparameter analyses. It also presents further qualitative results, per-identity quantitative evaluations, and a visualization of the learned offset maps for spatial analysis. In addition, a dedicated section discusses the limitations of our optimization-based framework, including comparisons with recent generalizable single-image reconstruction methods. Finally, Sec. 4 discusses data handling practices, potential societal harms, and measures to prevent misuse.

2. Method

2.1. Offset Activation Details.

To ensure numerical stability and constrain the influence of the predicted offsets $\delta_i = \{\delta_\mu, \delta_s, \delta_r, \delta_\alpha, \delta_c\}$ on the final Gaussian parameters, we apply carefully chosen nonlinear activation functions to each component extracted from the UV feature offset map Δ_{map}. These activations are designed to restrict each predicted quantity within a semantically meaningful and physically valid range, as described below:

- **Position offset** δ_μ: Given the typical spatial extent of Gaussians, we constrain each component of the position offset to the interval $[-0.1, 0.1]$ using a hyperbolic tanh activation:

$$\delta_\mu = 0.1 \cdot \tanh(x), \tag{1}$$

  where $x \in \mathbb{R}^3$ is the raw network output.
- **Scaling Offset** δ_s: To ensure the scaling offset remains strictly positive—thus making the multiplication with the coarse scale $\tilde{s}$ valid—we apply an exponential activation to map the output to the range $(0, +\infty)$:

$$\delta_s = \exp(x), \tag{2}$$

  where $x \in \mathbb{R}^3$. The final scale is computed as the element-wise product of δ_s and the coarsely deformed scale $\tilde{s}$, ensuring both positivity and flexible control.
- **Rotation offset** δ_r: The raw rotation offset is represented as an axis-angle vector $x \in \mathbb{R}^3$. Its magnitude is first normalized to the interval $[-\pi, \pi]$ via:

$$\delta_r^{\text{angle}} = \pi \cdot \tanh(x), \tag{3}$$

  and then converted to a unit quaternion δ_r using the standard axis-angle to quaternion transformation. This guarantees a valid rotation that can be composed with the coarsely deformed rotation $\tilde{r}$ through the Hamilton product.
- **Opacity offset** δ_α: To avoid drastic changes in transparency, the opacity offset is constrained to $[-0.5, 0.5]$:

$$\delta_\alpha = 0.5 \cdot \tanh(x), \tag{4}$$

  where $x \in \mathbb{R}$. After adding δ_α to the coarse opacity $\tilde{\alpha}$, the final opacity α^* is further clamped to $[0, 1]$ to ensure physical plausibility.
- **Color offset** δ_c: To limit abrupt color shifts during Gaussian's deformation, we restrict each channel of the color offset to $[-0.7, 0.7]$ via:

$$\delta_c = 0.7 \cdot \tanh(x), \tag{5}$$

  with $x \in \mathbb{R}^3$.

These bounded activations not only prevent degenerate parameter updates during training (*e.g.*, negative scale or invalid quaternion norms), but also provide a consistent prior for Gaussian parameters, allowing more effective learning from the UV-space offset map Δ_{map}.

2.2. UV Sample.

As shown in Algorithm 1, we provide a more detailed algorithmic description of UV-Adaptive Sampling. Since certain mesh triangles may not cover any valid pixels after UV rasterization, it becomes infeasible to define the UV coordinates of the associated Gaussian primitives using pixel-based sampling. To address this, we incorporate an analytical fallback strategy that samples barycentric coordinates uniformly within the triangle, enabling the assignment of UV coordinates even when no valid pixels are present.

As illustrated in Fig. 1, after applying Temporal Adaptive Density Control, our UV-Adaptive Sampling adaptively allocates more Gaussians to regions requiring finer details across different identities. In particular, the central facial region of *biden* contains more points due to the presence of facial wrinkles, which demand higher density. Additionally, the density in *nf_01*'s eye region significantly increases because *nf_01* wears glasses, which typically require more Gaussians for accurate modeling.

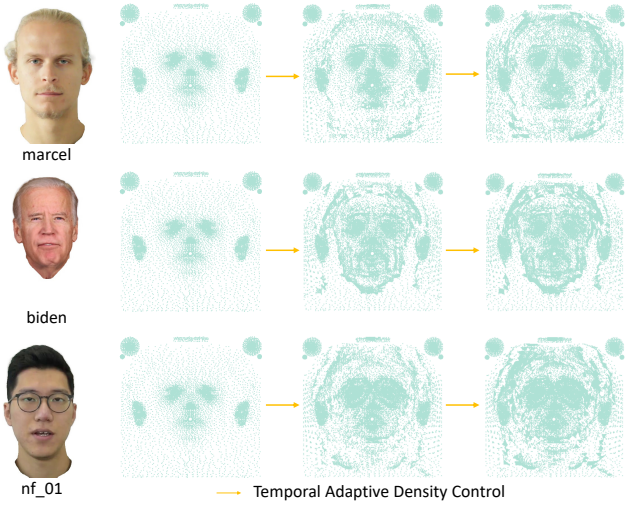

Figure 1. Visualization of UV-Adaptive Sampling's points.

2.3. Silhouette Coefficient.

To determine an appropriate number of clusters K for grouping structurally similar frames based on FLAME [6] parameters, we adopt the Silhouette Coefficient as the selection metric. Given a data point i, the silhouette score $s(i)$ is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i),\ b(i)\}}, \quad s(i) \in [-1, 1], \tag{6}$$

where $a(i)$ denotes the mean intra-cluster distance of point i, and $b(i)$ represents the smallest mean distance between point i and all points in the nearest neighboring cluster. These quantities are computed as:

$$a(i) = \frac{1}{|C_i| - 1} \sum_{\substack{j \in C_i \\ j \neq i}} d(i, j), \tag{7}$$

$$b(i) = \min_{k \neq i} \left(\frac{1}{|C_k|} \sum_{j \in C_k} d(i, j) \right), \tag{8}$$

where C_i is the cluster to which point i belongs, and $d(i, j)$ denotes the Euclidean distance in the PCA-projected FLAME parameter space. A higher average silhouette score across all points indicates better-defined and more compact clustering. We evaluate the silhouette score across varying $K \in [5, 12]$, and select the value that maximizes the overall score.

2.4. Summed-Area Table.

To efficiently compute aggregated error values over rectangular regions in the screen-space image domain, we adopt the two-dimensional *Summed-Area Table (SAT)* strategy, also known as the *integral image*. This data structure allows the summation of any axis-aligned rectangular region in constant time using only four memory accesses.

Given an input error map $E \in \mathbb{R}^{H \times W}$, where H and W denote the image height and width, respectively, the SAT

Algorithm 1 UV-Adaptive Sampling

Input:
$\mathcal{B} \in \mathbb{N}^N$: face binding index per Gaussian
$\mathcal{V}_{uv} \in \mathbb{R}^{V \times 2}$: UV coordinates of mesh vertices
$\mathcal{F}_{uv} \in \mathbb{N}^{F \times 3}$: triangle face indices over UV mesh
R: UV rasterization resolution (e.g., $R = 256$)
Output:
$\mathcal{U} \in \mathbb{R}^{N \times 2}$: UV coordinates per Gaussian

Rasterize UV mesh $(\mathcal{V}_{uv}, \mathcal{F}_{uv})$ into $R \times R$ grid
  $\rightarrow (\mathcal{M}_{face}, \mathcal{M}_{bary}) \in \mathbb{Z}^{R \times R}, \mathbb{R}^{R \times R \times 3}$
Initialize per-face pixel pool map $\mathcal{P}[f] \leftarrow \emptyset,\ \forall f \in \mathcal{F}_{uv}$

for each pixel (i, j) where $\mathcal{M}_{face}[i, j] \neq -1$ **do**
  $f \leftarrow \mathcal{M}_{face}[i, j]$
  Append $(i, j), \mathcal{M}_{bary}[i, j]$ into $\mathcal{P}[f]$
end for
Faces with valid pixels: $\mathcal{F}_{valid} \leftarrow \{f \mid \mathcal{P}[f] \neq \emptyset\}$
Initialize barycentric buffer $\mathbf{b} \in \mathbb{R}^{N \times 3}$, face index buffer $\mathbf{f}_{idx} \in \mathbb{N}^N$
for each face f that appears in bindings $\mathcal{B}$ **do**
  $\mathcal{I}_f \leftarrow \{i \mid \mathcal{B}[i] = f\}$ {Indices of points bound to f}
  $C_f \leftarrow |\mathcal{I}_f|$
  if $f \in \mathcal{F}_{valid}$ **then**
    Pixel-based Sampling
    Let $\mathbf{p}_f$ be sorted pixel list of face f in $\mathcal{P}[f]$
    if $|\mathbf{p}_f| \geq C_f$ **then**
      Select C_f evenly spaced pixels from $\mathbf{p}_f$
    else
      Select all $|\mathbf{p}_f|$ pixels and sample extra with replacement to reach C_f
    end if
    Extract barycentric coordinates $\mathbf{b}_f$ from $\mathcal{P}[f]$
  else
    Analytical Sampling
    for $i = 1$ to C_f **do**
      Sample $u, v \sim \mathcal{U}(0, 1)$
      Compute: $b_0 = 1 - \sqrt{u},\ b_1 = \sqrt{u}(1 - v),\ b_2 = \sqrt{u}v$
      Append $[b_0, b_1, b_2]$ to $\mathbf{b}_f$
    end for
  end if
  Assign: $\mathbf{b}[\mathcal{I}_f] \leftarrow \mathbf{b}_f, \quad \mathbf{f}_{idx}[\mathcal{I}_f] \leftarrow f$
end for
$\mathcal{U} \leftarrow \text{BarycentricReweight}(\mathcal{V}_{uv}, \mathcal{F}_{uv}, \mathbf{f}_{idx}, \mathbf{b})$

$I \in \mathbb{R}^{(H+1)\times(W+1)}$ is constructed as:

$$I(u,v) = \sum_{y=0}^{u-1} \sum_{x=0}^{v-1} E(y,x), \tag{9}$$

with the boundary conditions $I(0,v) = I(u,0) = 0$ for all u, v, which provide zero-padding to simplify index handling. The SAT is built in a single pass using two cumulative summations:

1. Compute the row-wise prefix sum for each row.
2. Compute the column-wise prefix sum of the row-wise sums.

This process is equivalent to:

$$\begin{aligned} I(u+1,v+1) = E(u,v) + I(u,v+1) \\ +I(u+1,v) - I(u,v) \end{aligned} \tag{10}$$

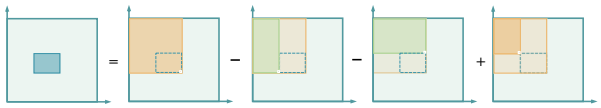

Figure 2. Inclusion–exclusion principle

Once the SAT is constructed, the sum over any rectangular region $[x_1, x_2] \times [y_1, y_2]$ can be computed in constant time using:

$$\begin{aligned} \sum_{x=x_1}^{x_2} \sum_{y=y_1}^{y_2} E(y,x) = I(y_2{+}1, x_2{+}1) - I(y_1, x_2{+}1) \\ - I(y_2{+}1, x_1) + I(y_1, x_1). \end{aligned} \tag{11}$$

This formula leverages the inclusion–exclusion principle to compute the sum via four corner indices, as illustrated in Fig. 2. The SAT reduces the computational complexity from $\mathcal{O}(R^2)$ per Gaussian to $\mathcal{O}(1)$ per region, enabling efficient error accumulation for tens or hundreds of thousands of Gaussians in real-time optimization.

3. Experiments

3.1. Baselines

As stated in the main paper, the number of training epochs is set to 10 for GA [7], RGBA [5] and Fate [14], 20 for FA [12], 30 for SA [8] and 100 for MGA [1] to ensure convergence. Except for the PointAvatar [16] dataset, we follow Fate's training schedule by training all baseline methods for 50 epochs and our method, STAvatar, for 30 epochs. Additionally, following the DECA-based [3] preprocessing pipeline, we further refine the FLAME coefficients for 50 epochs using a learning rate of 5×10^{-4} during testing, which is consistent with the IMAvatar [15] protocol.

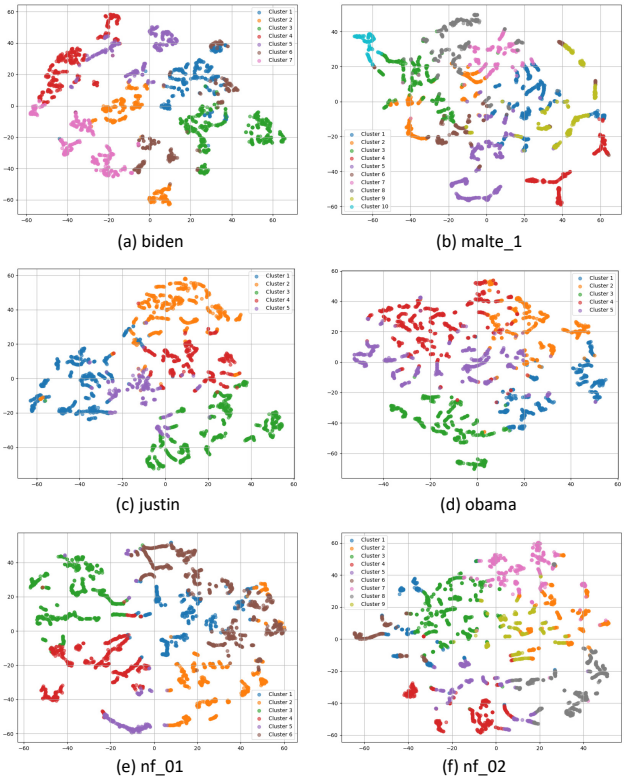

Figure 3. t-SNE visualization of different identities after applying FTC. Each color represents a distinct video frame cluster.

3.2. t-SNE of FTC.

To intuitively demonstrate the effectiveness of our FTC strategy, we visualize the clustering results using t-SNE [9] in Fig. 3. The number of clusters K is adaptively determined using the silhouette coefficient, which allows FTC to dynamically adjust to the inherent data distribution across identities. As shown in the Fig. 3, embeddings belonging to different identities are clearly separated in the latent space, suggesting that FTC effectively groups structurally similar frames into coherent clusters, which facilitates the computation of the average densification criterion.

3.3. Additional Ablation Study

We conduct additional ablation studies to further validate the effectiveness of each proposed component and the selection of key hyper-parameters. All results are summarized in Tab. 1, where we compare multiple candidate settings for the FPE's $\mathcal{L}_{\text{d-ssim}}$ weight, the FTC's randomly training epochs M, and the FTC's cluster weights. The configuration adopted in the main paper is indicated in bold.

FPE: $\mathcal{L}_{\text{d-ssim}}$ weight. We examine three choices for the perceptual weighting coefficient. The setting $\lambda_1 = 0.2$ consistently achieves the highest PSNR, SSIM, and lowest LPIPS, demonstrating that it provides a more balanced trade-off between pixel-level reconstruction and perceptual

Table 1. Hyperparameter ablation results. The row in **bold** indicates the configuration used in the main paper: FPE's $\mathcal{L}_{\text{d-ssim}}$ weight $\lambda_1 = 0.2$, FTC's randomly training epochs $M = 1$, and FTC's cluster weights $(0.3, 0.6, 0.1)$.

Hyperparameter	Setting	Metrics		
		PSNR↑	SSIM↑	LPIPS↓
λ_1	0.1	30.52	0.9577	0.0312
	0.3	30.48	0.9579	0.0311
	0.2	30.63	0.9587	0.0304
M	2	30.53	0.9582	0.0310
	0	30.42	0.9572	0.0311
	1	30.63	0.9587	0.0304
cluster weights	(0.5, 0.4, 0.1)	30.44	0.9584	0.0312
	(0.4, 0.5, 0.1)	30.50	0.9583	0.0311
	(0.3, 0.6, 0.1)	30.63	0.9587	0.0304

fidelity. This result aligns with the intuition that moderate perceptual emphasis yields better alignment with human visual sensitivity.

FTC: Randomly training epochs M. We further evaluate the influence of the FTC training strategy by varying the number of random training epochs M. The configuration $M = 1$ produces the best performance, indicating that performing clustered training followed by a single fine-tuning epoch on the entire training set is sufficient and effective. This observation verifies the necessity and contribution of FTC in facilitating the computation of the densification criterion which benefits the reconstruction of transiently visible regions.

FTC: Cluster weights. Besides, we test different FLAME-based cluster weight combinations to assess their impact on the clustering quality. The best-performing setting $(0.3, 0.6, 0.1)$ (corresponding to expression, pose, and translation components) assigns a larger weight to pose-related components, highlighting that pose cues dominate the temporal variations relevant to our densification criterion. This confirms that emphasizing pose information improves the reliability of cluster formation and benefits the downstream optimization process.

3.4. Visualization of Offset Maps

To further clarify the spatial effect of the learned offsets, we visualize the distribution of the predicted UV attribute offset maps in Fig. 4. The position map is constructed based on the canonical FLAME mesh topology, which does not explicitly model fine-grained structures such as hair. Therefore, it is important to examine whether the learned offsets compensate for these missing details and to identify the regions that benefit most from offset learning.

As illustrated in Fig. 4, Gaussians corresponding to hair are primarily associated with the scalp-related UV regions. Notably, the learned offsets exhibit significantly larger magnitudes in high-frequency and highly deformable areas, such as expression-related facial regions (e.g., mouth and eye surroundings) and hair regions. In contrast, offsets remain relatively small in geometrically rigid regions, such as the cheeks and forehead. These observations indicate that the offset learning mechanism adaptively concentrates its modeling capacity on spatially and temporally complex regions, enabling the representation of fine-scale structures (including hair) and dynamic deformations that are not explicitly captured by the underlying FLAME topology.

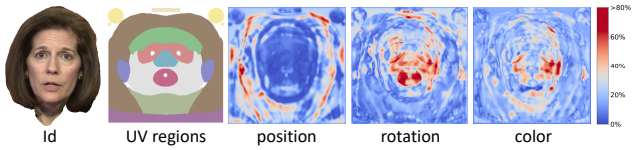

Figure 4. Visualization of learned UV attribute offset maps. Larger magnitudes are primarily observed in high-frequency and highly deformable regions, such as hair and expression-related areas.

3.5. Limitations and Discussion

In recent years, numerous generalizable methods [2, 4, 10, 11, 13] for single-image 3D reconstruction have achieved remarkable performance. Compared with these approaches, our optimization-based framework requires a monocular video sequence and per-subject training, which may limit its applicability in certain scenarios. Although generalizable methods demonstrate strong cross-identity generalization, they are inherently designed for single-image settings and therefore cannot exploit temporal information for subject-specific optimization. As a result, their capacity to learn fine-grained identity-specific characteristics and to capture complex dynamic motions is constrained. Nevertheless, optimization-based methods typically achieve higher identity fidelity due to subject-specific optimization.

To illustrate this trade-off, we conduct additional experiments on the INSTA dataset, selecting GAGAvatar and LAM as representative baselines. As reported in Tab. 2 and illustrated in Fig. 5, these methods struggle to faithfully reconstruct subtle identity details, such as teeth geometry and fine facial wrinkles, whereas our method preserves such high-frequency structures more accurately.

Method	PSNR↑	SSIM↑	LPIPS↓
LAM-20K	21.71	0.8655	0.1292
GAGAvatar	22.81	0.8942	0.1042
Ours	30.63	0.9587	0.0304

Table 2. Quantitative comparison with GAGAvatar and LAM on the INSTA dataset.

3.6. Self-Reenactment

As shown in Fig. 6, STAvatar reconstructs fine-grained geometric and appearance details (e.g., hair strands, eye con-

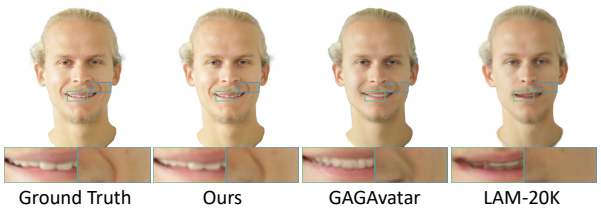

Figure 5. Qualitative comparisons on the INSTA dataset.

tours, and glasses frames) more faithfully than prior approaches. With the assistance of Temporal Adaptive Density Control, our model allocates higher density to transient or frequently occluded regions, leading to improved reconstruction of structures such as teeth. As summarized in Tab. 3 and Tab. 4, although STAvatar does not outperform all baselines on every metric, it achieves the best overall performance on all datasets.

3.7. Cross-Reenactment

As illustrated in Fig. 7, our method accurately transfers diverse expressions and motions—such as eye-closing, mouth asymmetry, and pouting—from the source actor while preserving the target identity's geometry and appearance. Even under significant pose variations and complex non-rigid deformations, the reconstructed avatars retain stable identity-related geometry and appearance, demonstrating the robustness of the proposed soft binding and temporal density control.

4. Ethical Discussion

All datasets used in this study are publicly available and collected in compliance with their respective licenses. While our method enables the generation of realistic and controllable 3D head avatars, we acknowledge its potential misuse in fabricating synthetic content that may infringe on privacy or mislead the public. We explicitly oppose any use of this technology for malicious or unauthorized purposes, including identity fraud or deceptive media creation. Developers and practitioners should act responsibly and follow ethical guidelines to prevent potential harm from misuse.

References

[1] Yufan Chen, Lizhen Wang, Qijing Li, Hongjiang Xiao, Shengping Zhang, Hongxun Yao, and Yebin Liu. Monogaussianavatar: Monocular gaussian point-based head avatar. In *ACM SIGGRAPH 2024 Conference Papers*, 2024. 3

[2] Xuangeng Chu and Tatsuya Harada. Generalizable and animatable gaussian head avatar. *Advances in Neural Information Processing Systems*, 37:57642–57670, 2024. 4

[3] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *ACM Transactions on Graphics (ToG)*, 40(4): 1–13, 2021. 3

[4] Yisheng He, Xiaodong Gu, Xiaodan Ye, Chao Xu, Zhengyi Zhao, Yuan Dong, Weihao Yuan, Zilong Dong, and Liefeng Bo. Lam: large avatar model for one-shot animatable gaussian head. In *Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Papers*, pages 1–13, 2025. 4

[5] Linzhou Li, Yumeng Li, Yanlin Weng, Youyi Zheng, and Kun Zhou. Rgbavatar: Reduced gaussian blendshapes for online modeling of head avatars. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10747–10757, 2025. 3

[6] Tianye Li, Timo Bolkart, Michael J Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4d scans. *ACM Trans. Graph.*, 36(6):194–1, 2017. 2

[7] Shenhan Qian, Tobias Kirschstein, Liam Schoneveld, Davide Davoli, Simon Giebenhain, and Matthias Nießner. Gaussianavatars: Photorealistic head avatars with rigged 3d gaussians. *CVPR*, 2024. 3

[8] Zhijing Shao, Zhaolong Wang, Zhuang Li, Duotun Wang, Xiangru Lin, Yu Zhang, Mingming Fan, and Zeyu Wang. SplattingAvatar: Realistic Real-Time Human Avatars with Mesh-Embedded Gaussian Splatting. In *CVPR*, 2024. 3

[9] Laurens van der Maaten, Geoffrey Hinton, and Yoesoep Rachmad. Viualizing data using t-sne. *Journal of Machine Learning Research*, 9:2579–2605, 2008. 3

[10] Zidu Wang, Xiangyu Zhu, Jiang Yu, Tianshuo Zhang, and Zhen Lei. S2td-face: Reconstruct a detailed 3d face with controllable texture from a single sketch. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 6453–6462, 2024. 4

[11] Zidu Wang, Meng Xu, Miao Xu, Hengyuan Ma, Jiankuo Zhao, Xutao Li, Xiangyu Zhu, and Zhen Lei. Bfsm: 3d bidirectional face-skull morphable model. *arXiv preprint arXiv:2509.24577*, 2025. 4

[12] Jun Xiang, Xuan Gao, Yudong Guo, and Juyong Zhang. Flashavatar: High-fidelity head avatar with efficient gaussian embedding. In *CVPR*, 2024. 3

[13] Dongbin Zhang, Yunfei Liu, Lijian Lin, Ye Zhu, Yang Li, Minghan Qin, Yu Li, and Haoqian Wang. Guava: Generalizable upper body 3d gaussian avatar. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14205–14217, 2025. 4

[14] Jiawei Zhang, Zijian Wu, Zhiyang Liang, Yicheng Gong, Dongfang Hu, Yao Yao, Xun Cao, and Hao Zhu. Fate: Full-head gaussian avatar with textural editing from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 3

[15] Yufeng Zheng, Victoria Fernández Abrevaya, Marcel C. Bühler, Xu Chen, Michael J. Black, and Otmar Hilliges. I M Avatar: Implicit morphable head avatars from videos. In *Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[16] Yufeng Zheng, Wang Yifan, Gordon Wetzstein, Michael J. Black, and Otmar Hilliges. Pointavatar: Deformable point-based head avatars from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 3

Figure 6. Qualitative comparisons of head reconstruction results across different state-of-the-art methods. Our method STAvatar produces more accurate geometry and fine-grained details, particularly in challenging regions such as occluded inner mouth areas.

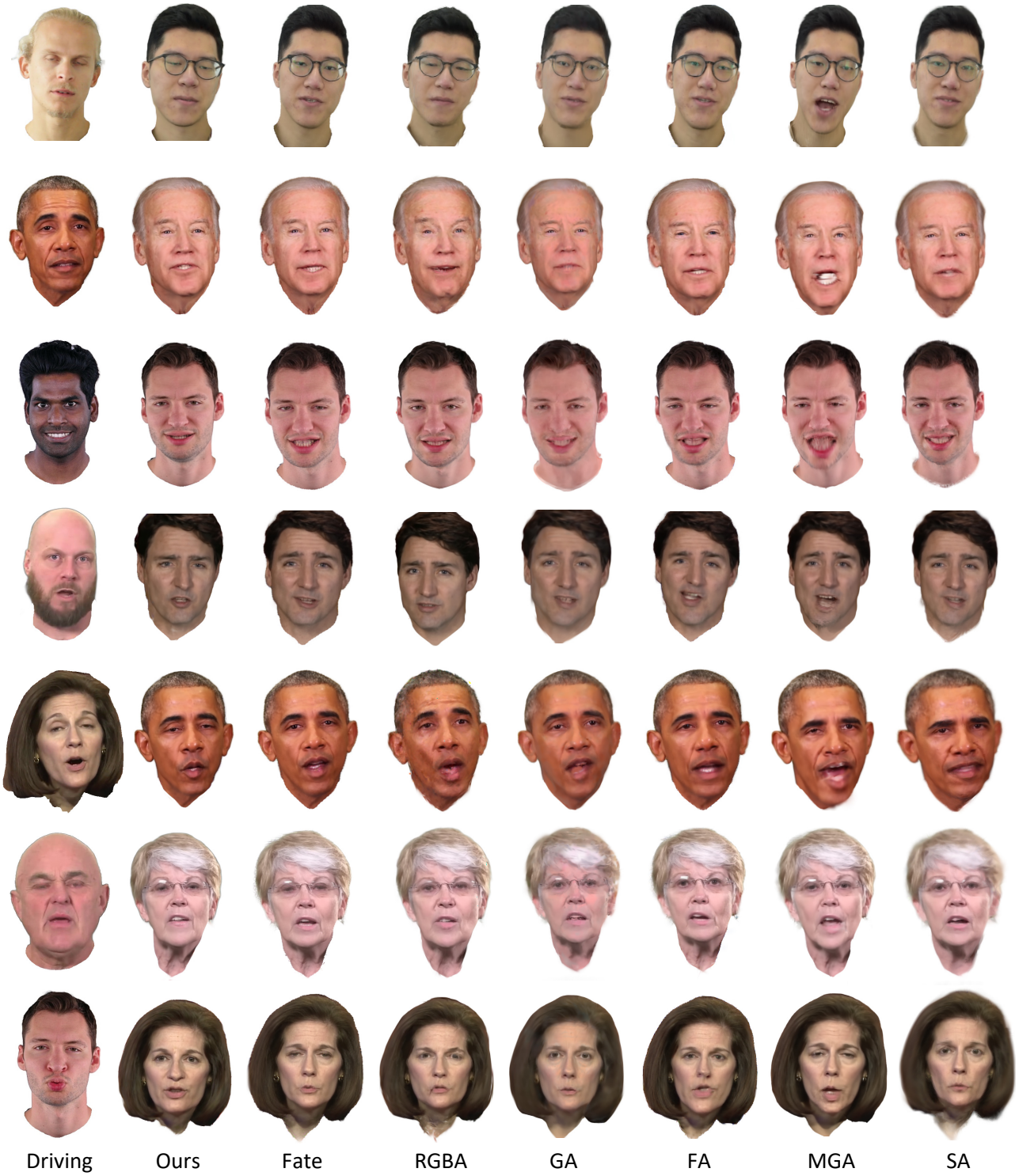

Figure 7. Qualitative comparison of cross-reenactment results with state-of-the-art methods. Our approach achieves achieves faithful expression transfer while maintaining consistent identity-specific geometry and appearance.

Table 3. Full comparison of quantitative results with state-of-the-art methods on INSTA dataset.

Datasets		INSTA Dataset									
		bala	biden	justin	malte_1	marcel	nf_01	nf_03	obama	person_4	wojtek_1
PSNR↑	SplattingAvatar	27.01	29.25	28.01	27.97	29.02	24.59	27.59	23.83	27.19	30.35
	MonoGaussianAvatar	25.39	29.38	27.32	28.25	28.27	25.09	28.01	27.10	24.48	30.16
	GaussianAvatars	27.41	29.11	27.49	27.69	27.84	23.71	27.27	23.40	26.04	29.89
	FlashAvatar	28.86	28.95	27.20	28.46	29.07	24.90	27.87	27.24	25.80	30.61
	RGBAvatar	29.11	30.45	28.48	28.39	30.56	26.44	27.95	27.53	24.53	30.65
	FateAvatar	29.02	30.28	28.54	28.35	29.21	25.40	28.20	27.93	25.40	30.94
	Ours	29.26	32.79	29.16	31.73	32.54	28.38	30.72	31.48	29.23	30.98
SSIM↑	SplattingAvatar	0.9138	0.9498	0.9570	0.9326	0.9325	0.9225	0.9133	0.9158	0.9452	0.9466
	MonoGaussianAvatar	0.8956	0.9546	0.9577	0.9386	0.9322	0.9324	0.9205	0.9426	0.9273	0.9498
	GaussianAvatars	0.9287	0.9551	0.9589	0.9356	0.9370	0.9242	0.9179	0.9205	0.9488	0.9516
	FlashAvatar	0.9187	0.9519	0.9513	0.9345	0.9298	0.9307	0.9162	0.9401	0.9377	0.9457
	RGBAvatar	0.9432	0.9685	0.9705	0.9470	0.9452	0.9488	0.9314	0.9542	0.9255	0.9591
	FateAvatar	0.9299	0.9646	0.9641	0.9438	0.9403	0.9384	0.9193	0.9511	0.9398	0.9552
	Ours	0.9406	0.9757	0.9706	0.9584	0.9541	0.9524	0.9481	0.9724	0.9575	0.9571
LPIPS↓	SplattingAvatar	0.1386	0.0771	0.0811	0.0839	0.1042	0.1325	0.1294	0.1126	0.1101	0.0769
	MonoGaussianAvatar	0.1135	0.0486	0.0539	0.0639	0.1338	0.1058	0.1098	0.0581	0.1151	0.0563
	GaussianAvatars	0.0976	0.0593	0.0655	0.0714	0.1032	0.1049	0.1138	0.0854	0.0908	0.0593
	FlashAvatar	0.0405	0.0315	0.0384	0.0401	0.0775	0.0692	0.0642	0.0407	0.1258	0.0347
	RGBAvatar	0.0535	0.0260	0.0327	0.0409	0.0655	0.0577	0.0666	0.0364	0.1029	0.0361
	FateAvatar	0.0534	0.0338	0.0360	0.0412	0.0681	0.0714	0.0603	0.0400	0.0691	0.0351
	Ours	0.0378	0.0166	0.0249	0.0234	0.0432	0.0406	0.0346	0.0182	0.0386	0.0256

Table 4. Full comparison of quantitative results with state-of-the-art methods on the HDFT dataset, PointAvatar dataset, and NerFace dataset.

Datasets		HDFT Dataset						PointAvatar Dataset			NerFace Dataset		
		subject1	subject2	subject3	subject4	subject5	subject6	yufeng	marcel	soubhik	person1	person2	person3
PSNR↑	SplattingAvatar	27.30	31.46	30.04	29.08	19.71	18.52	25.06	26.17	23.55	24.59	26.24	27.59
	MonoGaussianAvatar	28.57	32.89	30.30	30.15	20.42	19.92	28.86	25.90	28.97	25.09	26.46	28.01
	GaussianAvatars	26.21	31.06	29.63	28.69	18.16	16.74	24.93	26.15	22.77	23.71	26.25	27.27
	FlashAvatar	28.88	32.77	29.25	30.13	20.37	19.60	25.86	26.04	26.67	24.90	28.10	27.87
	RGBAvatar	28.83	30.30	29.57	29.44	20.97	21.23	26.34	27.35	26.32	26.44	26.99	27.95
	FateAvatar	28.98	33.10	30.59	30.24	20.52	19.66	29.07	26.77	29.25	25.40	27.75	28.20
	Ours	30.38	34.39	32.54	28.45	20.76	21.11	27.85	28.81	28.10	28.38	31.14	30.72
SSIM↑	SplattingAvatar	0.8827	0.9679	0.9556	0.9421	0.8366	0.7684	0.8757	0.9198	0.8858	0.9225	0.9357	0.9133
	MonoGaussianAvatar	0.9164	0.9745	0.9588	0.9553	0.8743	0.8231	0.9258	0.9154	0.9419	0.9324	0.9469	0.9205
	GaussianAvatars	0.8826	0.9687	0.9571	0.9447	0.8400	0.7341	0.8904	0.9302	0.9014	0.9242	0.9417	0.9179
	FlashAvatar	0.9198	0.9728	0.9488	0.9538	0.8592	0.8114	0.8870	0.9126	0.9169	0.9307	0.9524	0.9162
	RGBAvatar	0.9347	0.9705	0.9618	0.9598	0.8806	0.8588	0.9020	0.9288	0.9343	0.9488	0.9513	0.9314
	FateAvatar	0.9256	0.9747	0.9593	0.9554	0.8700	0.8271	0.9223	0.9254	0.9385	0.9384	0.9515	0.9193
	Ours	0.9456	0.9826	0.9719	0.9562	0.8799	0.8522	0.9236	0.9348	0.9426	0.9524	0.9697	0.9481
LPIPS↓	SplattingAvatar	0.1753	0.0638	0.1167	0.1200	0.3096	0.3070	0.1512	0.1388	0.1534	0.1325	0.0817	0.1294
	MonoGaussianAvatar	0.1081	0.0471	0.0977	0.0997	0.1776	0.2049	0.0958	0.1552	0.0830	0.1058	0.0545	0.1098
	GaussianAvatars	0.1502	0.0617	0.1115	0.1237	0.2446	0.3661	0.1397	0.1263	0.1340	0.1049	0.0709	0.1138
	FlashAvatar	0.0639	0.0296	0.0794	0.0578	0.1200	0.1139	0.1100	0.1051	0.0690	0.0692	0.0331	0.0642
	RGBAvatar	0.0595	0.0327	0.0617	0.0503	0.1532	0.1351	0.0872	0.0767	0.1011	0.0577	0.0408	0.0666
	FateAvatar	0.0638	0.0320	0.0643	0.0694	0.1389	0.1413	0.0709	0.0960	0.0660	0.0714	0.0356	0.0603
	Ours	0.0356	0.0160	0.0275	0.0410	0.1162	0.0893	0.0505	0.0576	0.0404	0.0406	0.0180	0.0346