High-Quality Static Head Avatars with Gaussian Splatting

Jie Peng (Frank) Chen

Abstract—Static 3D head avatars are essential for augmented reality (AR), virtual reality (VR), and gaming applications, yet achieving high fidelity and efficient initialization under lightweight setups remains challenging. Gaussian Splatting has proven effective for general scene representation, but it struggles with robust initialization and fidelity when modeling smooth human head geometries. To address these challenges, we propose a hybrid approach that combines random point initialization guided by head avatar masks with iterative point refinement through addition and pruning, thereby eliminating reliance on Structure-from-Motion (SfM). For rendering, we enhance Gaussian Splatting by integrating a fully connected neural network inspired by NeRF to predict Gaussian attributes such as color and opacity from encoded spatial features. This integration achieves smoother transitions and slight improvements in rendering fidelity, addressing some of the known limitations. Experiments on the RenderMe-360 dataset demonstrate that our approach provides modest but meaningful advancements in quality and efficiency for lightweight static head avatar modeling and novel view synthesis.

Index Terms—Static 3D Head Avatars, Gaussian Splatting, Novel View Synthesis

1 Introduction

Static 3D head avatars are crucial for applications in augmented reality (AR), virtual reality (VR), gaming, and digital telepresence, offering immersive and interactive experiences. While Neural Radiance Fields (NeRFs) [1] excel in novel view synthesis, their reliance on implicit representations and dense neural networks makes them inefficient for real-time rendering. 3D Gaussian Splatting (3DGS) [2], an emerging alternative, utilizes explicit point-based representations to enable faster training and rendering with a lightweight and continuous structure. However, applying Gaussian Splatting to static head avatar modeling faces challenges in initialization and in representing smooth human head geometries.

Dynamic head avatar modeling has seen significant progress, but many existing approaches [3], [4], [5], are not well-suited for static, high-fidelity 360-degree novel view synthesis. These methods often prioritize animated expressions and real-time flexibility, which may come at the expense of computational efficiency and generalizability for static settings.

This project addresses these challenges with a lightweight and robust framework for static head avatar modeling. We propose a random point initialization method refined through image masks to replace computationally expensive techniques like Structure-from-Motion (SfM). For rendering, we integrate a hash-encoded neural network inspired by NeRF to predict Gaussian attributes such as color and opacity, enhancing rendering fidelity and addressing artifacts like popping and floating points. Our method achieves a balance between quality and computational efficiency, making it practical for novel view synthesis tasks.

The contributions of this work are as follows:

- Proposed a novel random point initialization method with mask-based refinement for static head avatars, replacing the reliance on SfM.
- Integrated a NeRF-like neural network to improve quality for unseen views and address Gaussian Splatting artifacts like popping and floating effects.
- Demonstrated the practicality and efficiency of the approach for 360-degree novel view synthesis of static head avatars.

Code is available:
https://frankjc2022.github.io/static-head-avatar/.

2 Related Work

Recent advancements in head avatar modeling have primarily focused on dynamic representations, addressing challenges such as expression animation and real-time rendering. Several notable methods have emerged in this domain, each offering unique contributions and limitations relevant to our task of static head avatar reconstruction.

MonoGaussianAvatar [3] utilizes a point-based initialization strategy tailored for monocular portrait videos. This method places points on a sphere surrounding the subject, simplifying initialization while effectively handling portrait setups. By leveraging temporal consistency in monocular videos, MonoGaussianAvatar achieves robust geometry representation for dynamic facial movements. However, its design inherently limits its applicability to monocular portrait setups, as the point-based sphere initialization does not generalize well to multi-view setups. Additionally, while it captures complex geometries such as hair and accessories, it cannot handle full 360-degree coverage required for novel view synthesis across static head avatars.

Gaussian Head Avatar [4] builds on Gaussian Splatting by employing Signed Distance Functions (SDF) and Deep Marching Tetrahedra (DMTet) [6] for high-precision initialization. This strategy generates detailed 3D head representations, including intricate facial features and smooth surfaces, making it suitable for high-fidelity animated expressions. Gaussian properties, such as color and opacity, are predicted using a multi-layer perceptron (MLP), ensuring realistic appearance even in dynamic scenarios. Furthermore, an

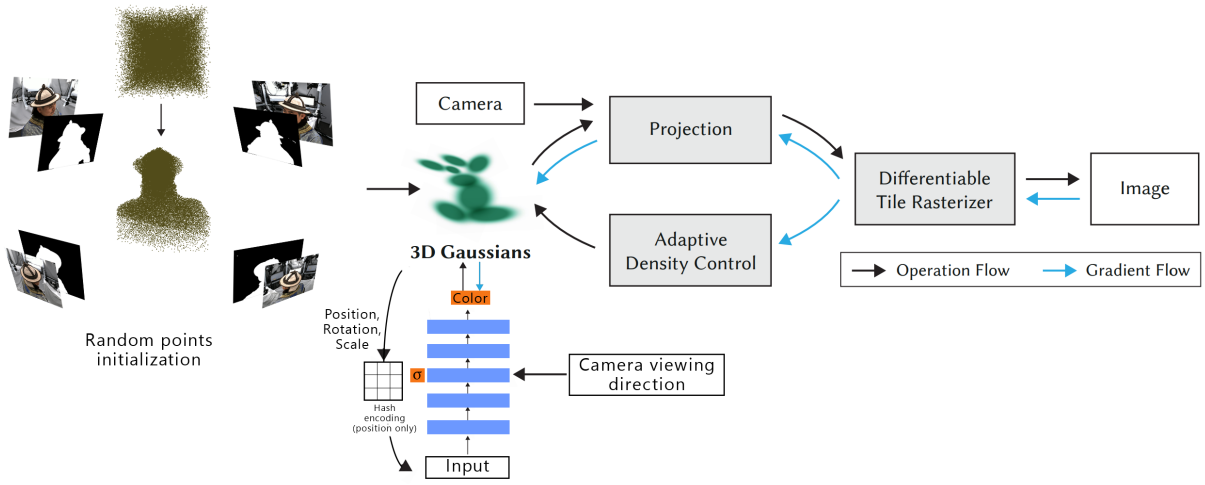

Fig. 1. The overview of the whole pipeline. The process begins with a randomized initial point cloud refined using image masks. An MLP with hash-encoded positions predicts the color and opacity of the Gaussians. The remainder of the process follows the original Gaussian Splatting pipeline.

additional MLP models deformation fields, allowing accurate representation of expression-dependent changes in geometry. This pipeline excels at capturing fine details, such as accessories, hair, and other non-facial structures, due to the robust initialization and deformation modeling provided by the SDF and DMTet frameworks. However, the computational demands of this approach are significant, and its primary focus is on dynamic animation rather than static, full 360-degree novel view synthesis of head avatars. This makes it less optimized for applications that prioritize static modeling over dynamic flexibility.

PSAvatar [5] introduces a point-based morphable shape model designed for real-time high-fidelity head avatar animation. By utilizing discrete geometric primitives, PSAvatar effectively models complex geometries such as eyeglasses and hairstyles, addressing limitations of traditional 3DMM-based methods. It employs 3D Gaussian representation for fine detail modeling and high-fidelity rendering, enabling detailed reconstructions of surface-like and intricate structures. However, PSAvatar primarily focuses on expression-dependent deformations and real-time animation. While it achieves impressive fidelity and real-time performance for dynamic expressions, it lacks comprehensive support for static 360-degree novel view synthesis, limiting its applicability in scenarios requiring detailed reconstructions of non-visible regions, such as the side of the head or accessories beyond facial regions.

3 Proposed Method

The pipeline for reconstructing static head avatars is illustrated in Fig. 1, including point cloud initialization, color and opacity prediction, and the Gaussian Splatting pipeline. The goal is to reconstruct a static 3D head avatar from a set of multi-view RGB images of a subject. Our method introduces two main improvements to 3D Gaussian Splatting. First, we replace the reliance on Structure-from-Motion (SfM) techniques like COLMAP for initial point cloud generation with a random point initialization approach refined using image masks. Second, we enhance the Gaussian attributes' prediction by introducing a fully connected neural network combined with hash encoding for RGB color and opacity.

3.1 Point Cloud Initialization

In the original Gaussian Splatting pipeline, SfM tools like COLMAP are used to generate a point cloud. The positions of the extracted points are used as Gaussian means, and their RGB values initialize the spherical harmonic (SH) coefficients for color. While effective, this process is computationally intensive and prone to failure when dealing with smooth human head geometries. Our approach eliminates the dependency on SfM by introducing a random point initialization method refined using image masks.

To initialize the point cloud, we randomly sample 50,000 points within a cube centered at the calculated scene center, with the edge length determined by the scene's radius. Both the center and radius are derived from the training camera parameters. The sampled points are projected onto the 2D image plane and filtered based on their inclusion within a provided mask, which is generated using BackgroundMattingV2 [7]. Points outside the mask are pruned, and new points are iteratively resampled and reprojected until the final set is fully contained within all masks. This iterative refinement ensures a robust 3D representation of the head, capturing complex features such as hair, shoulders, and accessories. The illustration of this process is shown in Fig. 2. The process completes within seconds, bypasses the limitations of SfM by not relying on key point detection across images, and provides a robust initialization without its computational overhead.

The remaining Gaussian properties—opacity, rotation, and scale—are initialized following the conventions of the original Gaussian Splatting method. However, unlike the original approach, which uses the RGB color of the extracted point cloud for initialization, our method assigns randomized colors to the points due to the lack of explicit color information in the random sampling.

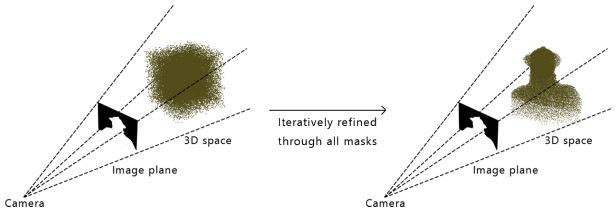

Fig. 2. Illustration of the iterative process of pruning points on a cube and refining them into the shape of a head using masks.

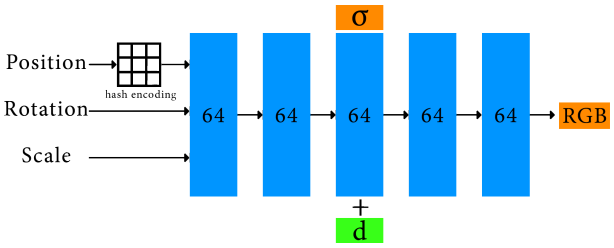

Fig. 3. MLP architecture for predicting color (RGB) and opacity (σ) using position (hash encoded), rotation, scale, and camera view direction (d) as inputs. The intermediate output is a 64-dimensional feature vector that encodes spatial and geometric properties of the input.

3.2 Color and Opacity Prediction

To predict Gaussian attributes, we replace the SH-based method of the original Gaussian Splatting pipeline with a neural network inspired by NeRF. This approach incorporates multi-resolution hash encoding from Instant Neural Graphics Primitives (NGP) [8] to efficiently encode the Gaussian's spatial positions into compact embeddings. Hash encoding is particularly effective in representing high-frequency details, allowing for efficient and accurate predictions.

The neural network architecture consists of two stages. In the first stage, a MLP predicts the opacity and a feature vector based on the encoded spatial position, rotation, and scale of the Gaussian. The second stage uses the predicted feature vector, concatenated with the viewing direction, to predict the RGB color through another MLP. The MLP architecture for this process is illustrated in Fig. 3.

The neural networks are implemented using the Tiny-CUDA-NN [9] framework, ensuring fast training and inference. The position encoding utilizes a hash grid with four levels, eight features per level, a logarithmic hash map size of 19, and a base resolution of 16. Both the opacity and color networks feature two hidden layers with 64 neurons each, ReLU activation functions, and specific output activations (Sigmoid for opacity and linear for RGB color). The feature vector size is 64 dimensions, balancing efficiency and quality.

This architecture addresses key artifacts of Gaussian Splatting, such as popping and floating effects, and enhances the quality of unseen views in novel view synthesis. By leveraging hash encoding and efficient neural networks, our method achieves real-time rendering performance while maintaining high visual fidelity.

4 Experimental Results

Dataset and Preprocessing. We utilize the RenderMe-360 dataset [10], which offers comprehensive multimodal data for each subject, including synchronous multi-view image frames, audio, calibrations, and annotations. For our experiments, we extracted 60 multi-view images at the same frame ID along with corresponding masks generated using BackgroundMattingV2. Each image, originally sized at 2448×2048, was resized to 1600×1338 during training, maintaining high resolution while reducing computational overhead. Consistent with the original Gaussian Splatting pipeline, we followed its train-test split strategy, using 52 views for training and 8 views for testing.

Point Cloud Initialization. We experimented with different numbers of initial points, testing 20,000, 50,000, and 100,000 points. The configuration with 50,000 points produced slightly better results across the tested subjects. The iterative pruning and projection process, ensuring all points fall within the masks across all views, typically converges within 500 iterations. However, we set a maximum of 1000 iterations to ensure completion. This process is highly efficient, completing within seconds.

While this initialization method ensures robust coverage of the head geometry, its quality heavily depends on mask accuracy. Deformable accessories or inconsistencies in masks across views may result in missing regions in the point cloud, as shown in Fig. 4. Nonetheless, the Gaussian Splatting densification step compensates for missing parts by generating additional points. Furthermore, our method effectively eliminates floating points caused by inaccurate masks in the original Gaussian Splatting pipeline. These floating points, often resulting from the inclusion of non-head regions in masks, are pruned during our iterative refinement process, as demonstrated in Fig. 5.

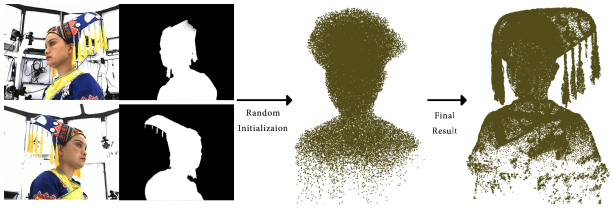

Fig. 4. Effect of mask accuracy and deformable accessories on point cloud initialization. Missing regions caused by mask errors or deformable elements are compensated by Gaussian Splatting densification.

Impact of Random Initialization. Our random initialization method addresses the limitations of SfM-based point cloud generation, particularly for subjects where SfM fails. However, inaccuracies in point placement—where some points do not align with the geometry surface—introduce artifacts during unseen view rendering, as shown in Fig. 6. These artifacts are propagated through the Gaussian Splatting densification step, highlighting the importance of accurate initial point distribution.

Despite these limitations, random initialization significantly reduces the number of Gaussians compared to COLMAP-generated point clouds. The evenly distributed random points efficiently cover the subject geometry, eliminating the need for densification in regions with insufficient initial points. For example, SfM often fails on smooth surfaces like facial regions, leading to gaps filled through

Fig. 5. Elimination of floating points caused by inaccurate masks in the original Gaussian Splatting pipeline. Non-head regions included in the masks are pruned during the iterative refinement process.

Fig. 6. Artifacts caused by random initialization, with points inside the body instead of on the geometry surface, resulting in noticeable errors in unseen views.

densification from nearby points. This process increases the total number of Gaussians. In contrast, our method achieves comprehensive coverage with fewer Gaussians, reducing training time and enabling faster rendering, as shown in Table 1, with higher FPS values.

Neural Network Design for Color and Opacity Prediction. To ensure efficiency, we designed a fully connected neural network with two hidden layers of 64 neurons each, a common practice balancing complexity and performance. For spatial encoding, we employed hash encoding with a grid search of configurations on two subjects, testing 4 or 8 features per level, 4 or 8 levels, and a per-level scale of 1.5. Minimal differences were observed, and the optimal configuration of 4 levels and 8 features per level was selected. Feature vector sizes of 32 and 64 were also tested, with 64 providing slightly better results.

Our network uses ReLU activation between layers and sigmoid activation for opacity output, consistent with the original Gaussian Splatting opacity model. No activation function is applied to the RGB output. This network operates on updated Gaussian properties, including position, rotation, and scaling, immediately prior to rendering. Additionally, we removed the opacity reset step in the original Gaussian Splatting pipeline since opacity is now predicted directly by the neural network.

Strategy Comparison and Results. We performed extensive comparisons to evaluate the impact of our neural network on rendering quality, as well as its influence on training time and performance. Four subjects were used for testing, with configurations varying by:

Input Variations:

- Position only (similar to NeRF).
- Position, rotation, and scale combined.

Prediction Targets:

- Direct prediction of RGB color and opacity.
- Modifiers for SH coefficients (for color) and opacity values.

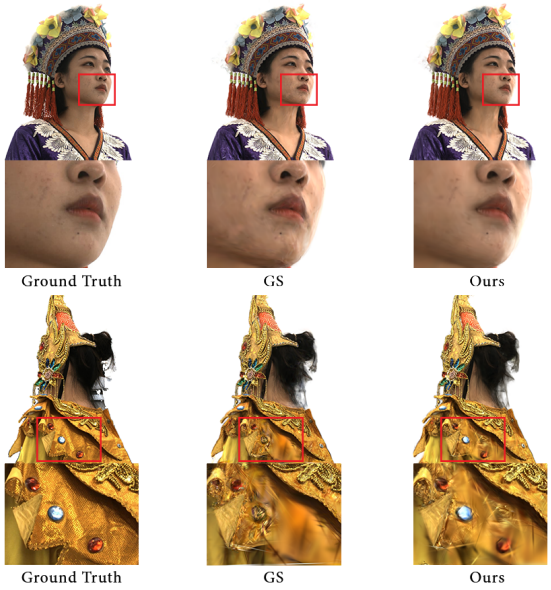

Fig. 7. Comparison of color and opacity predictions without using random point initialization. Top: Our method produces smoother predictions compared to 3DGS in unseen views. Bottom: Our approach captures finer details in unseen views, such as the gems, missed by 3DGS.

Prediction Scope:

- Predict color only.
- Predict opacity only.
- Predict both color and opacity.

The detailed comparison results are shown in Table 2. Using SfM-based initialization, direct prediction of RGB color and opacity with position, rotation, and scale as input yielded slightly better rendering quality than other configurations. However, with random initialization, predicting opacity only appeared to achieve better results compared to predicting color or both color and opacity. This could be due to the enhancement of opacity values for points that are not accurately positioned on the surface geometry, which improves the overall quality by reducing the visual impact of poorly placed points.

The detailed comparison results for four subjects are shown in Table 1, evaluating four methods: the original 3DGS, 3DGS+MLP (using full input of position, rotation, and scale to directly predict color and opacity), random initialization, and random initialization + MLP prediction. Additionally, the visual comparison for these methods across the four subjects is illustrated in Fig. 8, highlighting qualitative differences in rendering quality, particularly in unseen views.

As shown in Table 1, random initialization with masks achieves the highest FPS, significantly outperforming the original 3DGS, and also demonstrates the fastest training times. This method results in fewer Gaussians—nearly half compared to the original 3DGS. This reduction is due to our initialization points evenly covering the geometry of the face, unlike SfM-based initialization, which fails to extract points on smooth surfaces such as facial regions. Consequently, the original 3DGS relies on densification (cloning

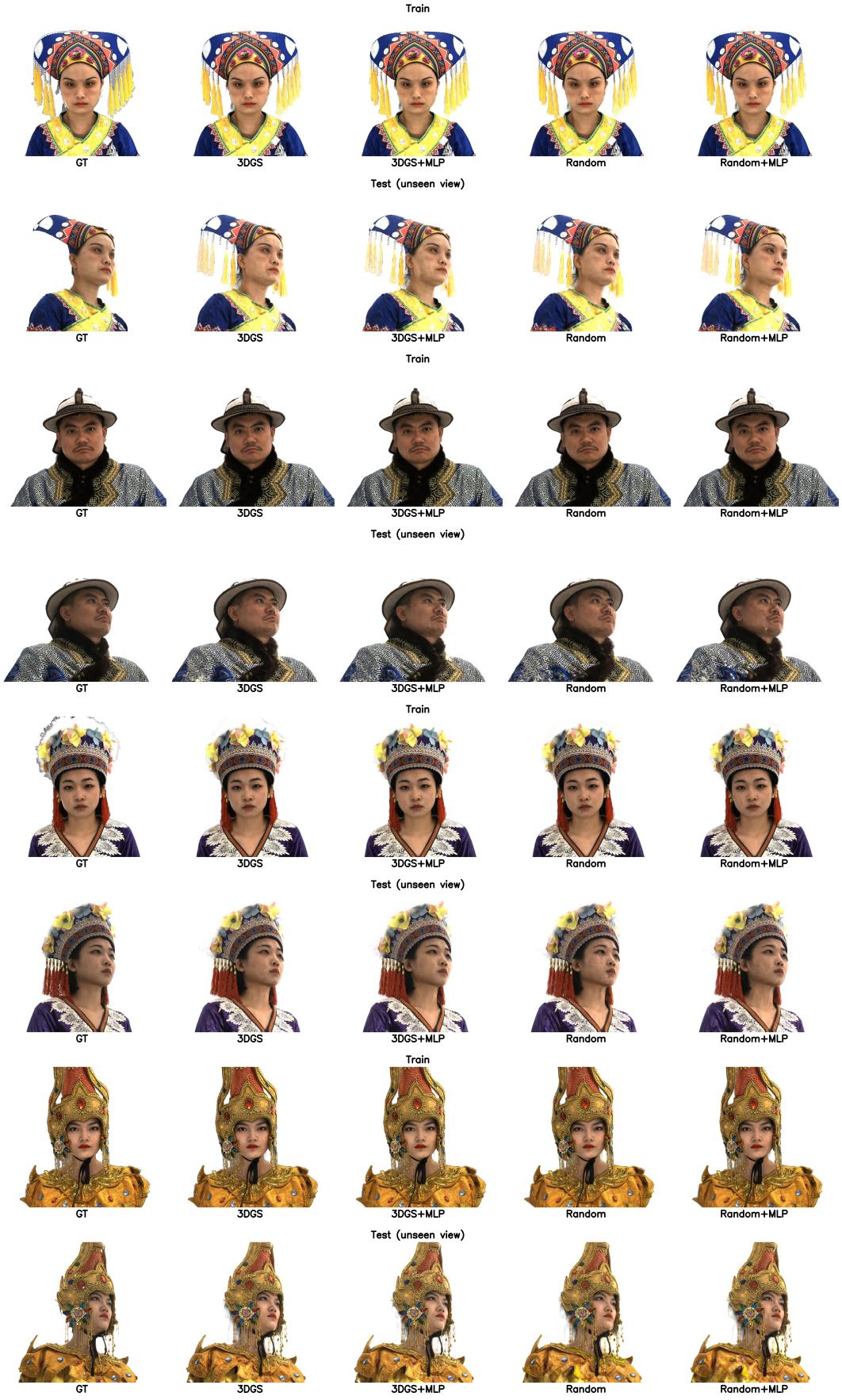

Fig. 8. Comparison of ground truth images from training views and test views across different methods: original 3DGS, 3DGS with our MLP prediction, random point initialization, and random point initialization with MLP prediction.

TABLE 1
Comparison of methods on rendering quality, training time, and performance across four test subjects.

Method	Subject 1						Subject 2					
	SSIM↑	PSNR↑	LPIPS↓	FPS	Train	# Gauss.	SSIM↑	PSNR↑	LPIPS↓	FPS	Train	# Gauss.
3DGS	0.9211	22.6394	0.1035	182.3	0:56:50	453,940	0.9314	24.2927	0.0782	116.5	1:05:45	807,494
3DGS+MLP	0.9212	22.7136	0.1062	21.4	1:46:52	687,393	0.9306	24.1268	0.0831	17.3	2:18:37	1,078,950
Random	0.9156	22.2716	0.1124	270.4	0:51:35	247,139	0.9223	23.7712	0.0936	205.7	0:56:20	377,845
Random+MLP	0.9092	21.6873	0.1227	51.1	1:06:18	243,852	0.9159	23.3672	0.1007	40.5	1:13:45	312,940

Method	Subject 3						Subject 4					
	SSIM↑	PSNR↑	LPIPS↓	FPS	Train	# Gauss.	SSIM↑	PSNR↑	LPIPS↓	FPS	Train	# Gauss.
3DGS	0.9344	24.6257	0.0897	170.4	0:56:24	495,993	0.8727	22.5021	0.1127	159.3	0:59:43	554,340
3DGS+MLP	0.9340	24.3393	0.0948	18.5	1:48:16	824,544	0.8837	22.6146	0.1155	20.0	1:53:24	815,818
Random	0.9298	24.3231	0.0995	263.0	0:50:39	243,983	0.8598	22.1335	0.1295	232.2	0:58:41	309,872
Random+MLP	0.9304	24.1596	0.1017	53.9	1:03:47	222,757	0.8601	22.1613	0.1337	40.7	1:18:45	314,444

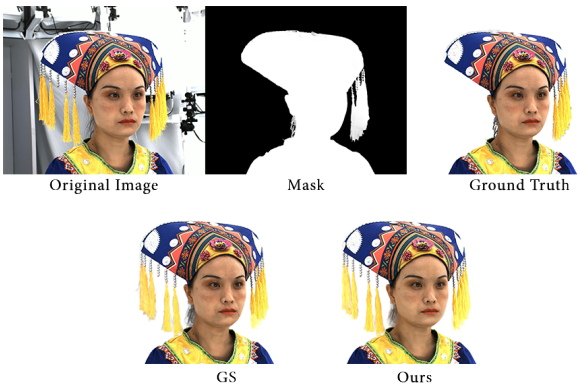

Fig. 9. Illustration of masked images and their impact on generated results. The ground truth image misses deformable accessories due to mask inaccuracies, as shown in the top column. However, the generated images correctly include these accessories because they are present in other images, though the inaccuracies in this image negatively impact training and lower the calculated metric values.

and splitting existing points) to fill these gaps, increasing the total number of Gaussians. However, the random nature of our initialization means that points are not always correctly aligned with the surface geometry, leading to lower overall quality compared to SfM-based initialization.

It is also important to highlight the characteristics of the four subjects in our dataset. Subject 1 contains deformable accessories, while Subject 4 features shiny surfaces and accessories, such as gems. In these cases, our MLP-enhanced methods performed better than the original 3DGS, achieving higher fidelity in rendering fine details. Conversely, for Subject 2 and Subject 3, which lack complex or reflective features, the original 3DGS outperformed our method slightly. This highlights that while the MLP method shows potential for high-detail scenarios, its performance varies depending on the subject's characteristics.

The MLP-enhanced methods show slight improvements in rendering fine details, particularly in unseen views, but come at the cost of longer training times and reduced rendering speed. As shown in the Table 1, FPS is significantly lower and training times are much higher due to the involvement of neural networks. Additionally, the MLP methods result in a higher number of Gaussians because the neural network influences the original Gaussian Splatting densification step to create more points, ensuring fine details are represented more accurately.

In summary, while the random initialization approach provides significant efficiency gains in terms of FPS and training time, it sacrifices some rendering quality due to the randomness of point placement. The MLP-enhanced methods improve detail and quality in complex scenarios, especially with shiny surfaces or deformable accessories, but at the cost of reduced performance and increased training overhead. These results underscore the trade-offs between efficiency and quality in head avatar modeling, with the choice of method depending on the specific application requirements.

Observations and Limitations. Our experiments showed that point clouds initialized with COLMAP consistently achieved better quality than those generated through random initialization. This result is expected, as random points are not accurately aligned with the subject's surface geometry. However, random initialization allows us to handle subjects where SfM fails entirely. Additionally, the significant reduction in the number of Gaussians with random initialization leads to faster training and higher FPS values, offering a practical trade-off.

For images, we use masked images with the background removed. This reduces the number of Gaussians required for the background, which could otherwise add millions of Gaussians, significantly increasing training and rendering times. Since our focus is on the head avatar, removing the background is both efficient and practical. However, this process makes the ground truth images less accurate due to mask inaccuracies, especially for deformable accessories or elements like hair, as shown in Fig. 9. These inaccuracies in the masked images, which are used to generate the 3D head avatar, may introduce errors in the 3D head avatar and affect the accuracy of evaluation metrics. This issue is more pronounced for subjects with deformable accessories.

Finally, our approach demonstrated fewer artifacts in unseen views for color and opacity prediction compared

TABLE 2
Comprehensive comparison of neural network configurations on rendering quality, training time, and performance. The table evaluates variati ons across input types (Position only vs. Position with rotation and scale), prediction strategies (Direct vs. Modifier for SH coefficients and opacity), and prediction scopes (Color only, Opacity only, and both Color and Opacity). Results are reported for one test subject, highlighting the top three values per metric: red (1st), orange (2nd), yellow (3rd). Training times are presented in hours, minutes, and seconds.

Initialization	Input Type	Prediction	Task	SSIM↑	PSNR↑	LPIPS↓	FPS	Train	# Gauss.
SfM (COLMAP)	Position only	Direct	Color Only	0.9167	22.4095	0.1091	43.2	1:12:01	586,503
			Opacity Only	0.9157	22.5662	0.1087	36.0	1:26:59	709,628
			Color + Opacity	0.9168	22.5655	0.1095	21.8	1:45:24	711,292
		Modifier	Color Only	0.9158	22.5674	0.1090	34.2	1:21:31	619,143
			Opacity Only	0.9205	22.6249	0.1066	50.5	1:05:58	461,292
			Color + Opacity	0.9153	22.5575	0.1110	21.6	1:22:14	434,688
	Position + R/S	Direct	Color Only	0.9176	22.5502	0.1068	38.2	1:14:08	613,178
			Opacity Only	0.9161	22.6530	0.1079	32.1	1:27:44	712,826
			Color + Opacity	0.9212	22.7136	0.1062	21.4	1:46:52	687,393
		Modifier	Color Only	0.9153	22.5229	0.1093	32.6	1:22:29	613,837
			Opacity Only	0.9163	22.6085	0.1124	48.4	1:06:44	450,545
			Color + Opacity	0.9170	22.6100	0.1105	28.2	1:26:05	419,684
Random (Mask)	Position only	Direct	Color Only	0.9078	21.4516	0.1227	75.6	1:00:53	300,597
			Opacity Only	0.9113	22.2256	0.1155	85.2	1:00:01	255,394
			Color + Opacity	0.9107	21.9751	0.1214	53.9	1:06:37	243,442
		Modifier	Color Only	0.9069	21.7876	0.1212	66.0	1:03:15	284,708
			Opacity Only	0.9140	22.1509	0.1158	95.6	0:55:15	217,653
			Color + Opacity	0.9079	21.9493	0.1205	50.9	1:05:23	223,044
	Position + R/S	Direct	Color Only	0.9099	21.6186	0.1210	70.3	1:00:04	308,336
			Opacity Only	0.9111	22.2250	0.1161	86.4	1:01:46	224,450
			Color + Opacity	0.9092	21.6873	0.1227	51.1	1:06:18	243,852
		Modifier	Color Only	0.9040	21.5487	0.1218	63.1	1:03:24	287,068
			Opacity Only	0.9110	21.9271	0.1154	84.0	0:58:41	235,770
			Color + Opacity	0.9067	21.9491	0.1200	50.8	1:05:34	215,934

to the original Gaussian Splatting pipeline, as shown in Fig. 7. Note that this comparison focuses on the prediction of color and opacity, not on random point initialization. Random point initialization generally results in lower quality compared to accurate SfM point clouds. By leveraging neural networks for color and opacity prediction, combined with efficient initialization, our method effectively balances rendering quality and computational efficiency for novel view synthesis.

5 Conclusion and Discussion

In this work, we presented an improved pipeline for static head avatar reconstruction using 3D Gaussian Splatting. By introducing a random point initialization method refined through image masks, we eliminated the reliance on Structure-from-Motion (SfM) techniques, addressing challenges related to smooth human head geometries and failed point extractions. This fast and robust approach ensures comprehensive point coverage, including complex features such as hair, shoulders, and accessories, while reducing the number of Gaussians and improving rendering efficiency. Additionally, we integrated a NeRF-inspired neural network for predicting Gaussian properties like color and opacity, utilizing hash encoding and Tiny-CUDA-NN to maintain computational efficiency.

Our experiments demonstrated that while random initialization provides a practical alternative to SfM, it produces lower-quality results due to inaccuracies in point positioning on the subject's surface. However, the reduced number of Gaussians accelerates training and rendering, offering a compelling trade-off. Moreover, our neural network configurations yielded slight improvements in rendering fidelity, particularly in unseen views, compared to the original Gaussian Splatting pipeline. The approach shows some improvement in addressing artifacts such as floating points and popping effects, though it comes at the cost of increased rendering time due to the neural network predictions.

For future work, improving the random initialization process to ensure points better conform to the subject's surface could significantly enhance rendering quality while preserving efficiency. Additionally, fine-tuning the MLP configuration and exploring optimizations for hash encoding could help mitigate bottlenecks in rendering speed, enabling a better balance between quality and performance. By addressing these areas, we aim to advance static head avatar modeling toward real-time, high-fidelity virtual representations suitable for AR/VR and related applications.

References

[1] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020.

[2] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[3] Y. Chen, L. Wang, Q. Li, H. Xiao, S. Zhang, H. Yao, and Y. Liu, "Monogaussianavatar: Monocular gaussian point-based head avatar," *arXiv*, 2023.

[4] Y. Xu, B. Chen, Z. Li, H. Zhang, L. Wang, Z. Zheng, and Y. Liu, "Gaussian head avatar: Ultra high-fidelity head avatar via dynamic gaussians," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[5] Z. Zhao, Z. Bao, Q. Li, G. Qiu, and K. Liu, "Psavatar: A point-based shape model for real-time head avatar animation with 3d gaussian splatting," *arXiv preprint arXiv:2401.12900*, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2401.12900

[6] T. Shen, J. Gao, K. Yin, M.-Y. Liu, and S. Fidler, "Deep marching tetrahedra: a hybrid representation for high-resolution 3d shape synthesis," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[7] S. Lin, A. Ryabtsev, S. Sengupta, B. Curless, S. Seitz, and I. Kemelmacher-Shlizerman, "Real-time high-resolution background matting," *arXiv*, pp. arXiv–2012, 2020.

[8] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 102:1–102:15, Jul. 2022. [Online]. Available: https://doi.org/10.1145/3528223.3530127

[9] T. Müller, "tiny-cuda-nn," 4 2021. [Online]. Available: https://github.com/NVlabs/tiny-cuda-nn

[10] D. Pan, L. Zhuo, J. Piao, H. Luo, W. Cheng, Y. Wang, S. Fan, S. Liu, L. Yang, B. Dai, Z. Liu, C. C. Loy, C. Qian, W. Wu, D. Lin, and K.-Y. Lin, "Renderme-360: A large digital asset library and benchmarks towards high-fidelity head avatars," *Advances in Neural Information Processing Systems*, vol. 36, 2024.