Im2Haircut: Single-view Strand-based Hair Reconstruction for Human Avatars (Supplementary)

Vanessa Sklyarova[1,2] Egor Zakharov[2] Malte Prinzler[1,2] Giorgio Becherini[1]
Michael J. Black[1] Justus Thies[1,3]

[1]Max Planck Institute for Intelligent Systems [2]ETH Zürich [3]Technical University of Darmstadt

1. Background

To use 3D Gaussian Splatting for soft-rasterization of hair, we force Gaussians to lie on line segments. We define the mean and covariance matrix of each individual Gaussian as:

$$\mu_{ij} = \frac{1}{2}\big(p_{ij} + p_{i,j+1}\big), \quad C_{ij} = E_{ij} D_{ij} \big(E_{ij} D_{ij}\big)^T. \quad (1)$$

Here, $E_{ij} = \{b_{ij}, t_{ij}, n_{ij}\}$ is a TBN basis associated with the strand curve, $b_{ij} = v_{ij}/|v_{ij}|, v_{ij} = p_{i,j+1} - p_{ij}$, where v_{ij} denotes the segment vector and b_{ij} its normalized direction vector. D_{ij} is defined as $D_{ij} = \text{diag}(d_{ij}, \epsilon, \epsilon)$, where d_{ij} is set to be proportional to the length of v_{ij} and ϵ denotes a small value. Such parametrization allows effective propagation of photometric information into hairstyle geometry.

2. Training details

2.1. Strands parametrization

For basis calculation, we launch the Incremental PCA method on all hairstyles from the PERM [4] dataset. We use 200 points for each strand to provide more degrees of freedom for hairstyles. We found that PCA method can effectively compress each strand of size 200×3 into 64 dimensions.

2.2. Optimization details

We use 4 NVIDIA A100 GPUs to train our entire method, which takes a total of 5 days and 19 hours.

For the **coarse stage**, we optimize the model for $420,000$ iterations (around 73 hours) with an effective batch size of 32 using AdamW [6] with a weight decay of 0.001 and a learning rate of 0.0001. We use the following weights: $\lambda_{\text{PCA}} = 0.1$, $\lambda_{\text{dir}} = 0.1$, $\lambda_{\text{curv}} = 1$, and $\lambda_{\text{mask}} = 0.0001$.

For the **fine stage**, we first fine-tune the fine branch with the ground-truth PCA map for the first 10 components for 200,000 iterations or around 45 hours. We initialize the Encoder and Decoder architectures using the pretrained weights from the coarse branch. For optimization, we use the following weights: $\lambda_{\text{PCA}} = 10$, $\lambda_{\text{dir}} = 0.1$, $\lambda_{\text{curv}} = 0.1$, and $\lambda_{\text{mask}} = 0.0001$. We also calculate visibility weights for points and apply a $3\times$ weight for points, direction, and curvature losses.

Then, we finetune the coarse and fine branches together for 80,000 iterations on synthetic data (around 8-9 hours). For optimization, we use the following weights: $\lambda_{\text{PCA}} = 1$, $\lambda_{\text{dir}} = 0.1$, $\lambda_{\text{curv}} = 0.1$, and $\lambda_{\text{mask}} = 0.0001$.

Finally, we optimize the model **on mixed dataset** with real and synthetic data for 16,000 iterations (around 12 hours). To balance the gradient propagation from real and synthetic data, we weight the photometric losses with weight $r = 0.5$. We use L_1 distance to calculate the depth loss. We use the following parameters: $\lambda_{\text{PCA}} = 0.1$, $\lambda_{\text{dir}} = 0.1$, $\lambda_{\text{curv}} = 0.1$, $\lambda_{\text{mask}} = 0.0001$, width for each gaussian = 0.005, $\lambda_{depth} = 0.01, \lambda_{pen} = 0.1, \lambda_{seg} = 10, \lambda_{dirmap} = 5$.

During the **inversion stage**, we optimize the hairstyle for 400 iterations on a single A100, which takes around 10 minutes. We use the following parameters: width for each gaussian is set to 0.00035, upsampling from the texture size 64×64 to 256×256, $\lambda_{seg} = 1, \lambda_{dirmap} = 0.8, \lambda_{pen} = 0.3, \lambda_{depth} = 0.01$.

2.3. Preprocessing

We align all synthetic hairstyles to the same bust model. To simplify the training, we ensure that real and synthetic data share the same scale range. This is achieved by applying an affine transformation to the input image, using facial keypoints estimated from both the real image and the rendered bust model. For real and synthetic data, we extract depth using Depth Pro [1]. We normalize the depth map values within the hair silhouette by following these steps: (i) we erode the hair segmentation mask to reduce boundary artifacts; (ii) we clip the depth values using the 2nd and 98th quantiles; and (iii) perform min-max normalization.

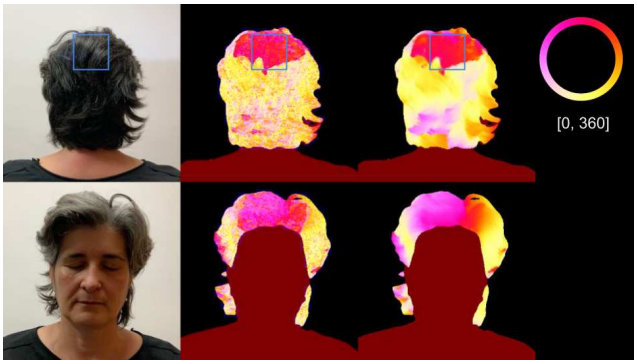

Image Gabor+Dirmap Dirmap

Figure 1. Combination of Gabor map with Direction map from Hairstep [13]. Although the model predicts correct directions for frontal views, it produces incorrect results for back views.

Figure 2. **Synthetic dataset** is generated by rendering ground-truth hairstyles in Blender.

2.4. Interpolation

To obtain interpolated hairstyles from our guiding strands during the inversion stage, we utilize an upsampling method with a weighted combination of nearest and bilinear interpolation following HAAR [9], adapted to operate in 3D space. This results in around 10,000 strands during optimization. For visualization purposes, we repeat this procedure and increase up to 30,000 strands.

3. Evaluation details

3.1. Baselines

We compare our method with Hairstep [13] using the original publicly available code. Results for NeuralHDHair [11] and PERM [4] were provided by the respective authors. For Hairnet [14], we use images extracted from the Hairstep [13] paper, and do not compare on more samples, as the performance of this model is significantly worse than other existing methods. Finally, in Figure 3, we compare with Hairmony [7] on several images and use the corresponding results extracted from their paper.

All baselines use augmented versions of the USC-HairSalon [5] dataset. The PERM dataset is constructed similarly, so the synthetic datasets used here and in prior work are comparable. Retraining all models under identical conditions is not feasible since prior works did not release training code.

Quantitative comparison. For quantitative comparison of our method with the baselines, we render synthetic hairstyles using Blender [2] (see images in Figure 2).

Inference time. Perm [4] is an optimization-based method based on a retrieval procedure that could take around 4 hours. Hairstep [13] takes around 3-5 minutes to reconstruct 30,000 strands on an NVIDIA RTX3090. Our method employs 400 optimization iterations, requiring approximately 10 minutes on an A100, yet it achieves superior strand quality compared to Hairstep [13] within the same time frame, see "Ours$^{\text{same cost}}$" in Figure 6 for comparison under the same computation budget.

Visualization. To show rendering results, we use Unreal Engine [3]. For NeuralHDHair [11] we have the 60,000 strands, while for Hairstep [13] and our around 30,000. For PERM [4] original number of strands from the authors was used, as additional interpolation may lead to worse results.

4. Applications

4.1. Multi-view optimization

In this section, we show more results of integrating our prior model in the multi-view reconstruction scenario. Rather than relying solely on the direction map, we enhance it with Gabor orientation maps to incorporate finer details. Since the direction map estimator from Hairstep [13] struggles with accurate predictions from side and back views (compare Figure 1, first row) as well as loose high-frequency details from frontal, we propose using an augmented Gabor direction map (see "Gabor+Dirmap" in Figure 1) for near-frontal views and an undirected Gabor map for other viewpoints. The optimization process follows the same weighting scheme as for the single-view scenario, with the only difference that for the orientation loss we compute either $\mathcal{L}_{\text{dir}}$ or $\mathcal{L}_{\text{undir}}$ depending on the view.

During optimization, we input a **near-frontal** image into the encoder to extract more accurate features, while supervising from other views using the same Gaussian Splatting-based procedure with soft-rasterization of hair strands. At each iteration, we randomly sample a single view for supervision. We also experimented with rendering and applying weighted loss from all views simultaneously, but this slowed down the training.

In Figure 11, we show reconstruction results of our method using different numbers of views, such as 1, 3, 8, and 32 on H3DS dataset [8]. We compare our approach to Gaussian Haircut (GH) [12] for the 8- and 32-view cases, as GH fails to produce reasonable results with only 1 or 3

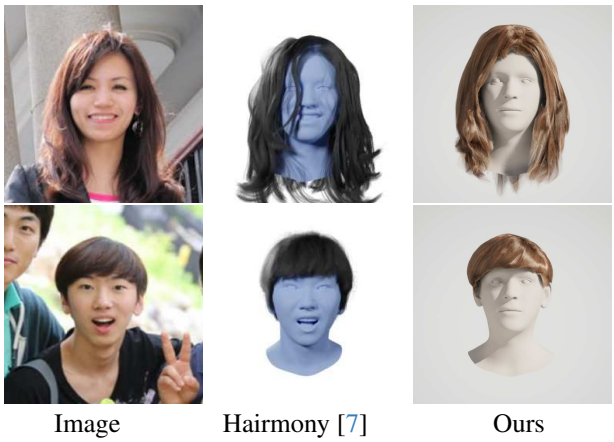

Figure 3. **Comparison** of our model with retrieval-based method Hairmony [7] .

views. For a fair comparison, we lunch GH using its official repository and default configuration settings. Our optimization process involves 400 steps for the 1 view case, 800 steps for 3 views, and 1600 steps for 8 and 32 views. Running all 1600 steps takes approximately 45 minutes, significantly faster than GH, which takes around 10 hours.

In Figure 11, GH has some freezing issues with strands that happen because of direct optimization for directions in 3D space. Also, it has some issues with the scalp mask, which may be resolved with an improved scalp estimator algorithm. Our method has some smoothing in the results because of the use of interpolation. This could be potentially resolved by using a learnable neural interpolation scheme during the optimization stage. We notice an improved curly geometry for our method (see Figure 11, last 3 rows). While the hairstyle in GH appears more detailed, some strands exhibit noisy structures. Potentially, both approaches could be combined to achieve more detailed hair while staying within the hairstyle prior to prevent unrealistic structures.

4.2. Simulations

For simulation results, we use Unreal Engine [3]. To do that, we convert hairstyles into Alembic format and import as a groom into Unreal. For simulation, we use around $30,000$ strands.

5. Additional experiments

5.1. Reconstruction results

More baselines. In Figure 3, we show a comparison with the retrieval-based method Hairmony [7]. While retrieval-based methods could predict general hair style, they do not contain any personalized details and are restricted by the diversity and quality of the dataset. We show an extended comparison with Hairstep [13] and NeuralHDHair [11] in

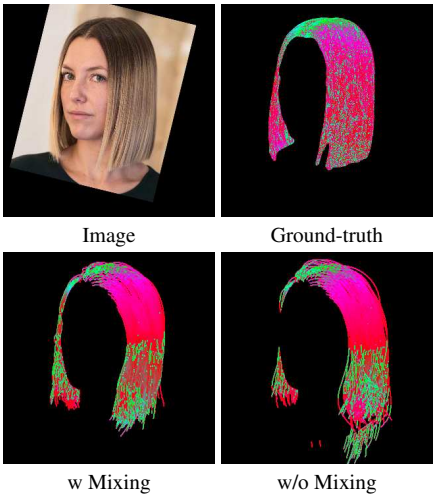

Figure 4. **Mixing strategy.** The performance of the Hairstyle prior model with and without mixing strategy. The model trained on real images can better regress the hair silhouette and orientations. The color in the image corresponds to the direction in the orientation maps.

Figures 19, 20.

Back views. We show additional visualization results for back views of our method, Hairstep [13], and NeuralHDHair [11] (see Figure 8). While NeuralHDHair [11] produces accurate and smooth results for back views, the method lacks details from the frontal view. Hairstep [13] reconstructs unrealistic back views, especially for short hairstyles (see Figure 13). Our method generates more realistic results for all types of hair (see Figures 12, 13).

Wavy hairstyles. In Figure 12, we show a comparison with Hairstep [13] on wavy hairstyles. While our method has problems in the accurate estimation of curls from a single image, it outperforms Hairstep [13] in terms of quality.

Out-of-distribution samples. We present results of our method on out-of-distribution samples (Figures 13, 14, 15), demonstrating its ability to reconstruct hairstyles for portraits, movie and cartoon characters, as well as drawings.

More results. Finally, we show our reconstruction results on more images (see Figures 17, 16, 18). Note that baldness artifacts or inaccuracies in hairstyles arise from failures in the camera or segmentation estimators. Additional camera fine-tuning and holistic reconstruction may resolve this problem.

Figure 5. **Hair reconstruction stage.** We demonstrate the importance of optimizing in Hairstyle prior space of the model for single-view inversion (compare "Ours" to "w/o $prior_{3D}$" and "w/o $prior_{pca}$"). Also, we show a scenario when only the decoder is optimized while the encoder is kept fixed (see "w/o Enc").

Figure 6. **Comparison with Hairstep** under the same computational cost.

	chamfer_pts ↓	chamfer_angle ↓	angle error ↓	mask ↓	$\mathcal{L}_{\text{undir}}$ ↓
coarse branch	0.00026	0.110	15.81	0.517	0.735
w/o depth input	0.00031	0.114	16.33	0.548	0.737
w/o dir	0.00028	0.112	15.95	0.553	0.737

Table 1. **Extended ablation on losses and usage of depth representation** as input during the training of the coarse branch.

	chamfer_pts ↓	chamfer_angle ↓	angle error ↓	mask ↓	$\mathcal{L}_{\text{undir}}$ ↓
hybrid training	0.00030	0.143	18.03	0.405	0.695
Prior$^{\text{w/o dir}}$	0.00028	0.140	17.84	0.418	0.723
Prior$^{\text{w/o penetr}}$	0.00030	0.139	17.60	0.412	0.689
Prior$^{\text{w/o silh}}$	0.00033	0.141	17.96	0.571	0.710

Table 2. **Contribution of losses** during training prior model.

5.2. Extended ablation

We present an extended version of our ablation study in Figures 5, 7, 9, 10 and Tables 1, 2. First, we show the ablation study on losses and input representation used during training the coarse branch, the Hybrid model, and during the inversion stage. Then, we show more reconstruction results obtained using a Hairstyle prior that is trained on a mixture of synthetic and real images, and without it. To disentangle the contribution of rendering loss from the mixture strategy, we train our prior model with rendering loss computed only on synthetic data. Also, we clarify the importance of optimization in the learned Hairstyle prior space compared to direct optimization of directions in 3D space or in PCA hair map. Finally, we conduct an experiment with and without the optimization of the encoder during inversion.

Coarse stage. In Table 1, we show an extended ablation of using depth as input and the usage of direction loss during training the coarse branch model. Without depth input or direction loss, we see the degradation of quality across all metrics.

Importance of losses. In Figure 7, we extend an ablation study on losses for (1) the optimization phase (opt* with our final Hybrid model), and (2) Hybrid model training (Prior*; results are shown w/ opt which uses *all* losses). During both training and optimization, the orientation loss (dir) plays a critical role in improving fine-grained strand details (see close-ups). The silhouette (silh) improves pixel alignment. Removing depth maps worsens the results (see "opt$^{\text{w/o depth}}$"). Our method is also robust to depth corruption (see "opt$^{\text{w/ blur depth}}$", where we blurred the depth maps). Lastly, penetration loss leads to better internal geometry.

In Table 2, we calculate the metrics on synthetic and real data with omitted losses during training the Hybrid model. Surprisingly, omitting penetration loss improves metrics on synthetic data, but results in increased mask loss on real data. Excluding direction or silhouette loss produces bad results on real data.

Figure 7. **Extended ablation on losses.** We show the importance of mixing strategy by training a prior model using rendering loss computed only on synthetic data (see Prior$^{\text{syn only}}$). Also, we provide an ablation on losses during hybrid model training with post optimization using *all* losses, and their contribution during optimization with our final Hybrid model. Lastly, we analyze robustness results of our method with blurred depth as input.

Importance of Mixing strategy. In Figure 4, we show results of our Hairstyle prior before ("w/o Mixing") and after training ("w Mixing") on a mix of synthetic and real data. Model "w Mixing" can provide better hairstyle initialization in terms of hair silhouette and orientations for the inversion stage. Note, here we rasterize orientations of obtained hairstyles using OpenGL [10] and color them for visualization purposes.

In Figures 9, 10, we show more results of inversion in the hairstyle prior space trained with (see "Ours") and without mixing strategy (see "w/o Mixing"). The model that is not trained on real images results in more unrealistic structures and penetrations. To disentangle the contribution of mixing strategy from rendering loss, in Figure 7 (see "Prior$^{\text{syn only}}$"), we show inversion results in the space of a prior model trained with rendering loss computed only on synthetic data.

Importance of optimization in prior space. In the proposed "Ours" configuration, we jointly optimize both the encoder and decoder architectures. The result of the optimization hairstyle in the 3D space after retrieving the coarse structure from the Hairstyle prior noted as "w/o $prior_{3D}$". Compared to our optimization setup, we decrease the learning rate to 0.00001 while doing 400 steps for inversion. Using more steps does not improve the quality of results. While this approach fails for wavy hairstyles, it produces realistic results for simpler, straight hairstyles, see Figure 5. Additionally, instead of optimizing the directions of the strands in the 3D space, as done in Gaussian Haircut [12], we optimize a PCA texture map, initialized from the hairstyle prior (see "w/o $prior_{pca}$"). Interestingly, this method introduces noisy artifacts in the generated strands.

Optimization of Encoder. We examine optimization within our prior space while keeping the encoders frozen (see Figure 5, "w/o Enc"). We find that this configuration underperforms compared to jointly optimizing both the encoder and decoder architectures.

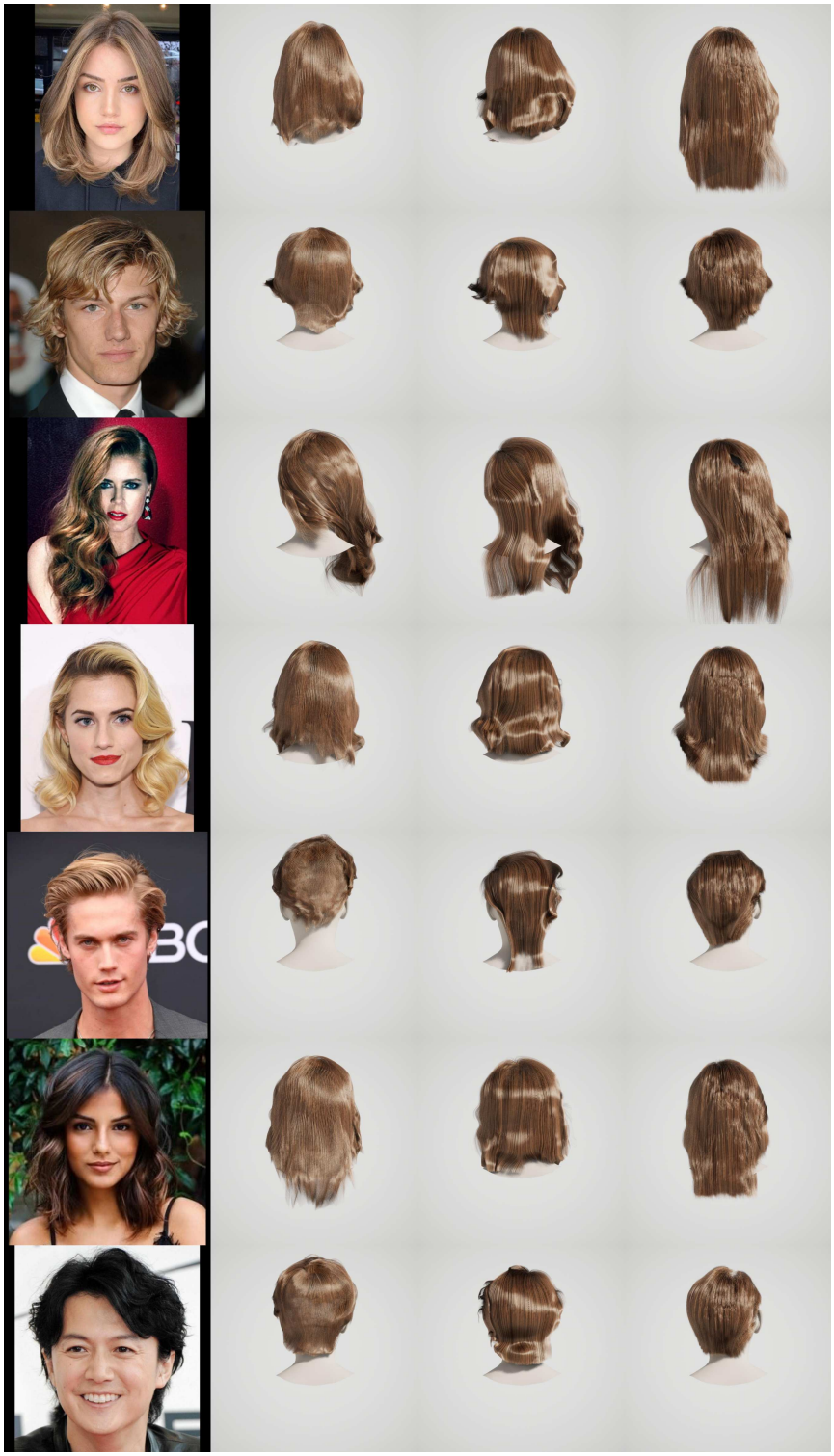

Figure 8. **Back view comparison.** Comparison of back view geometry of reconstructions obtained by our method, Hairstep [13] and NeuralHDHair [11].

Figure 9. **Extended ablation on importance of training on synthetic and real data.** Results of "Ours" correspond to columns 2-4, while "w/o Mixing" to 5-7.

Figure 10. **Extended ablation on importance of training on synthetic and real data.** Results of "Ours" correspond to columns 2-4, while "w/o Mixing" to 5-7.

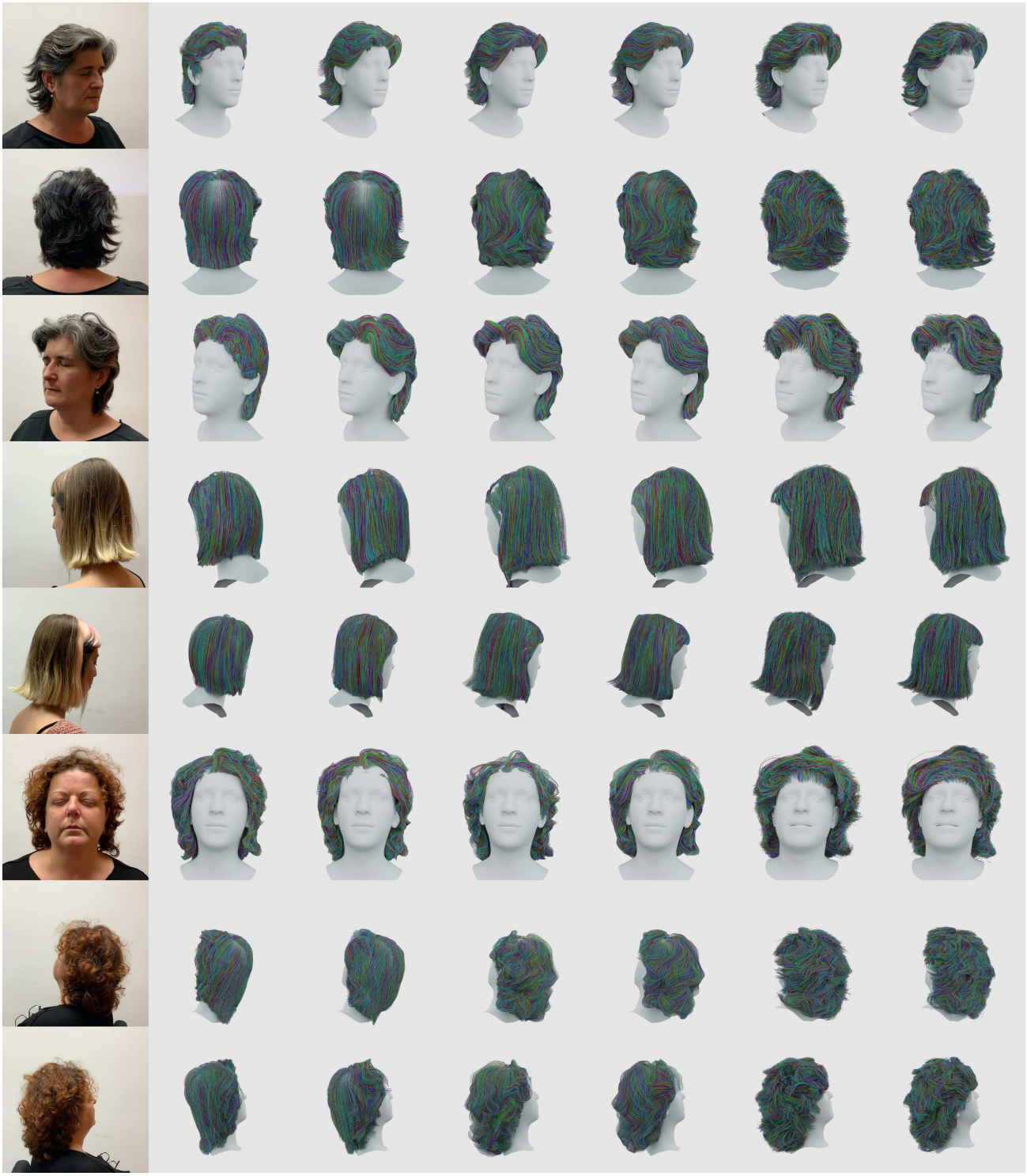

Figure 11. **Extended qualitative comparison using real-world multi-view scenes [8] with Gaussian Haircut (GH) [12].** We compared in a scenario with 1, 3, 8, and 32 views available. Note, GH fails in scenarios with 1 and 3 views. Digital zoom-in is recommended.

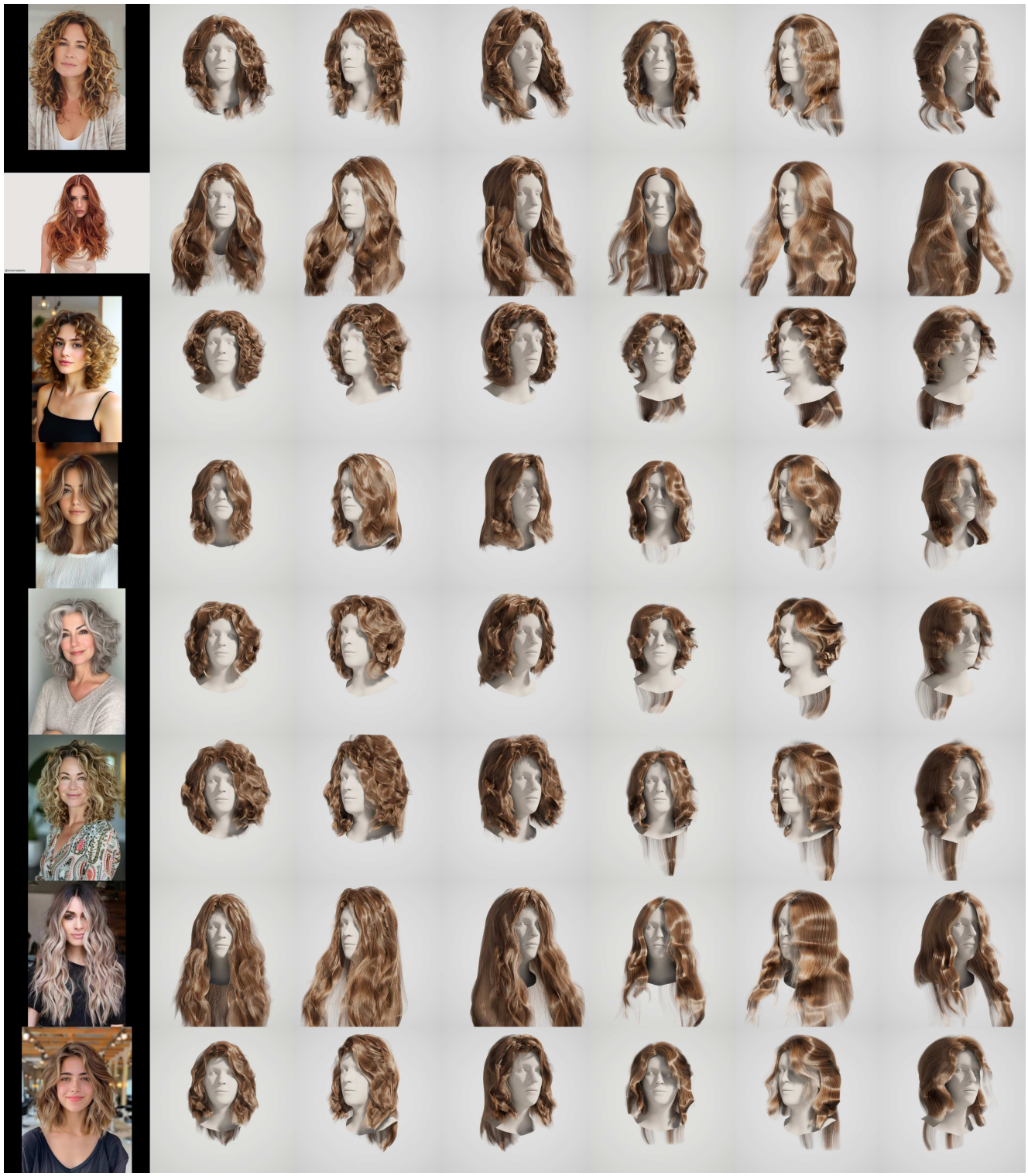

Figure 12. **Extended qualitative comparison of our method (columns 2–4) with Hairstep [13] (last three columns) on wavy samples.** Our method can reconstruct curlier structures with more realistic back geometry.

Figure 13. **Extended qualitative comparison of our method (columns 2–4) with Hairstep [13] (last three columns)** on out-of-distribution samples.

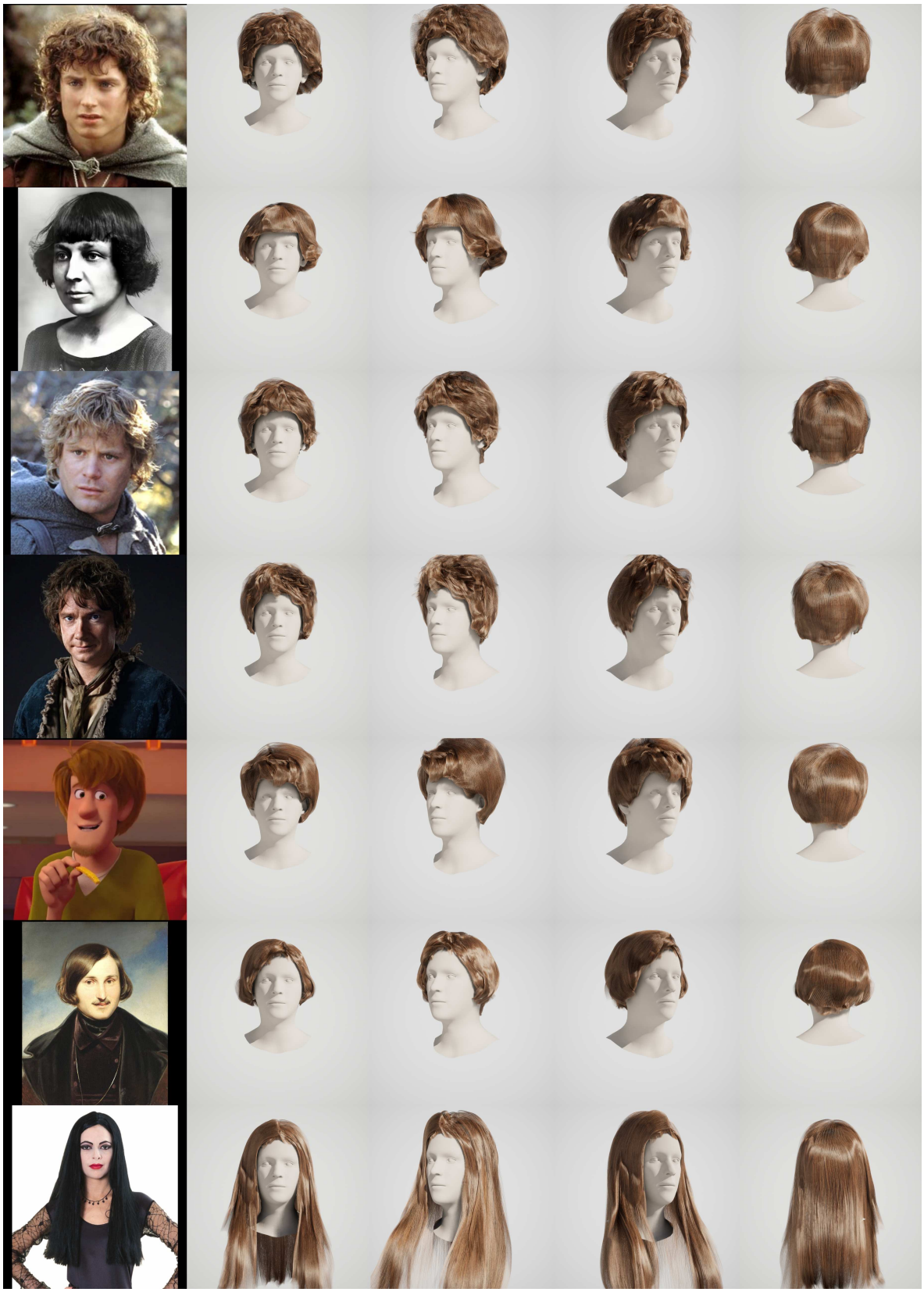

Figure 14. More results of our method on out-of-distribution data.

Figure 15. **More results** of our method on out-of-distribution data.

Figure 16. **Additional results** of our model.

Figure 17. **Additional results** of our model.

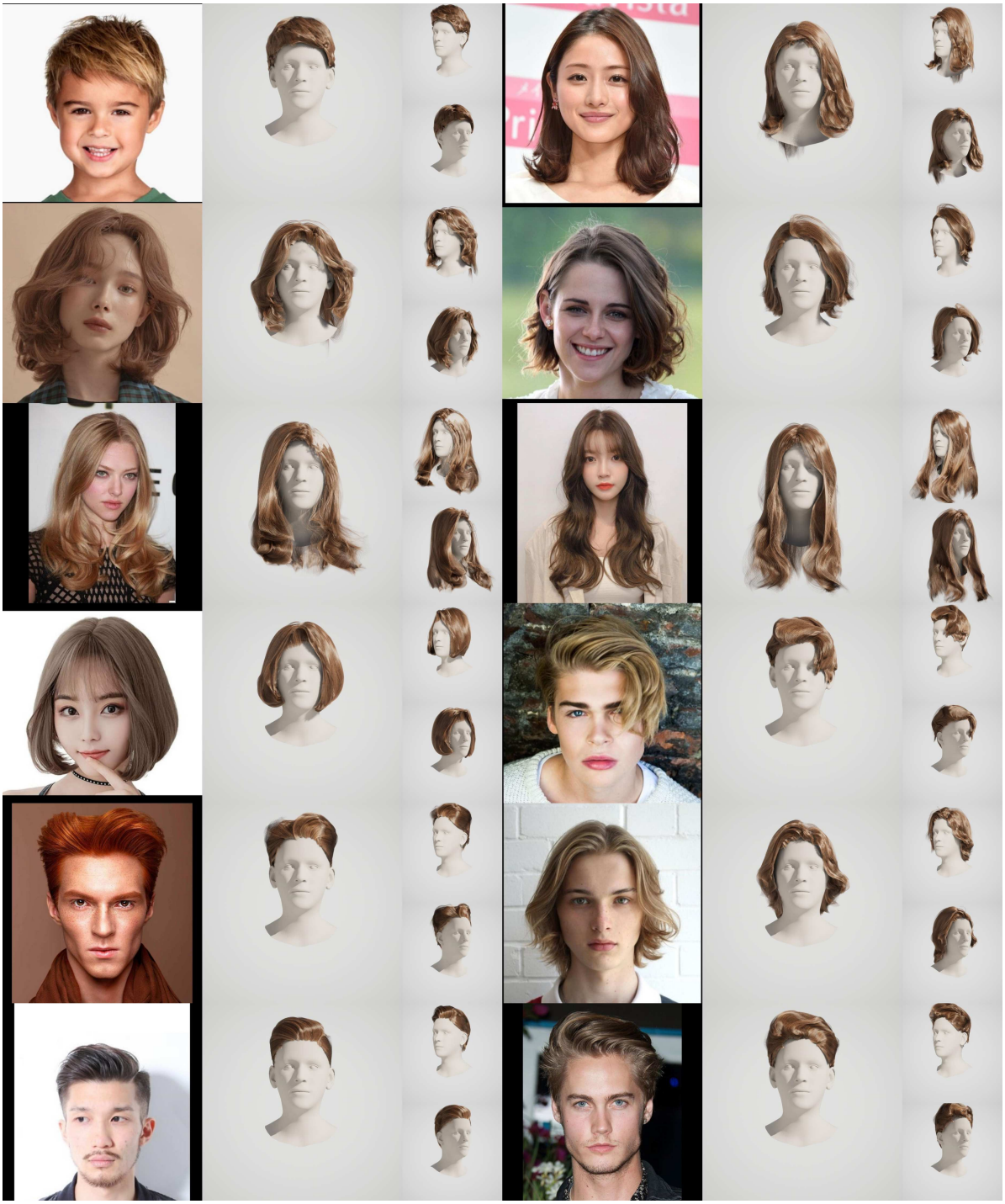

Figure 18. **Additional results** of our model.

Figure 19. **Extended qualitative comparison with Hairstep [13] and NeuralHDHair [11].** Note that in NeuralHDHair the number of rendered strands is twice as compared to our method and Hairstep.

Figure 20. **Extended qualitative comparison with Hairstep [13] and NeuralHDHair [11].**

References

[1] Aleksei Bochkovskii, Amaël Delaunoy, Hugo Germain, Marcel Santos, Yichao Zhou, Stephan R. Richter, and Vladlen Koltun. Depth pro: Sharp monocular metric depth in less than a second. *ArXiv*, abs/2410.02073, 2024. 1

[2] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 2

[3] Epic Games. Unreal engine. 2, 3

[4] 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. In *International Conference on Learning Representations*, 2025. 1, 2

[5] Liwen Hu, Chongyang Ma, Linjie Luo, and Hao Li. Single-view hair modeling using a hairstyle database. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2015)*, 34 (4):125:1–125:9, 2015. 2

[6] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 1

[7] Givi Meishvili, James Clemoes, Charlie Hewitt, Zafiirah Hosenie, Xian Xiao, Martin de La Gorce, Tibor Takacs, Tadas Baltrusaitis, Antonio Criminisi, Chyna McRae, Nina Jablonski, and Marta Wilczkowiak. Hairmony: Fairness-aware hairstyle classification. In *SIGGRAPH Asia 2024 Conference Papers*, page 1–11. ACM, 2024. 2, 3

[8] Eduard Ramon, Gil Triginer, Janna Escur, Albert Pumarola, Jaime Garcia, Xavier Giro-i Nieto, and Francesc Moreno-Noguer. H3d-net: Few-shot high-fidelity 3d head reconstruction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5620–5629, 2021. 2, 9

[9] Vanessa Sklyarova, Egor Zakharov, Otmar Hilliges, Michael J. Black, and Justus Thies. Text-conditioned generative model of 3d strand-based human hairstyles. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4703–4712, 2023. 2

[10] Mason Woo, Jackie Neider, Tom Davis, and Dave Shreiner. *OpenGL programming guide: the official guide to learning OpenGL, version 1.2*. Addison-Wesley Longman Publishing Co., Inc., 1999. 5

[11] Keyu Wu, Yifan Ye, Lingchen Yang, Hongbo Fu, Kun Zhou, and Youyi Zhengl. 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 1516–1525, 2022. 2, 3, 6, 17, 18

[12] Egor Zakharov, Vanessa Sklyarova, Michael Black, Giljoo Nam, Justus Thies, and Otmar Hilliges. Human hair reconstruction with strand-aligned 3d gaussians. In *Computer Vision – ECCV 2024: 18th European Conference, Milan, Italy, September 29–October 4, 2024, Proceedings, Part XVI*, page 409–425, Berlin, Heidelberg, 2024. Springer-Verlag. 2, 5, 9

[13] Yujian Zheng, Zirong Jin, Moran Li, Haibin Huang, Chongyang Ma, Shuguang Cui, and Xiaoguang Han. HairStep: Transfer synthetic to real using strand and depth maps for single-view 3d hair modeling. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12726–12735, 2023. 2, 3, 6, 10, 11, 17, 18

[14] Yi Zhou, Liwen Hu, Jun Xing, Weikai Chen, Han-Wei Kung, Xin Tong, and Hao Li. HairNet: Single-view hair reconstruction using convolutional neural networks. In *Computer Vision – ECCV 2018: 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part XI*, page 249–265, Berlin, Heidelberg, 2018. Springer-Verlag. 2