OMG-Avatar: One-shot Multi-LOD Gaussian Head Avatar

Supplementary Material

6. More Visualization Results

We present additional cross-reenactment results on the VFHQ and HDTF datasets in Fig. 10, along with further qualitative evaluations on in-the-wild images in Fig. 11 and Fig. 12. Novel view results are shown in Fig. 9. Notably, Fig. 12 demonstrates our model's strong generalization capability—even to highly stylized or non-photorealistic inputs such as cartoon portraits (4th row) and statue-like sculptures (5th row). Our method faithfully preserves fine structural details in the shoulder region while accurately transferring the target pose and expression. This is exemplified by the clear rendering of the bloodstain on the collar in the second-to-last row of Fig. 12. Furthermore, our approach exhibits remarkable robustness to facial occlusions—such as sunglasses in the second-to-last row of Fig. 11—without degrading reenactment quality or introducing visible artifacts.

To the best of our knowledge, GAGAvatar [6], LAM [21] and ours are the only state-of-the-art methods that support one-shot, feed-forward 3D avatar reconstruction with real-time facial reenactment. We compare our approach against both and present the results in Fig. 8. Additional video reenactment results on in-the-wild imagery are provided in the supplementary video (OMG-Avatar.mp4).

7. Ethical Discussion

Our method enables high-fidelity, animatable 3D head avatar generation with potential applications in video production, digital communication, and other domains. However, like other advanced generative models, it could be misused to create deceptive or non-consensual synthetic content (commonly known as "deepfakes") that may mislead, manipulate, or infringe on personal privacy. We firmly oppose such misuse and emphasize that our work is intended solely for legitimate, consent-based applications. To mitigate potential risks, we propose the following safeguards:

- **Visible and invisible watermarking.** We will integrate visible watermarking mechanisms into our released code, as shown in Figure 7. Visible watermarks will be embedded in all generated images and videos, clearly indicating that the content is AI-generated, enabling viewers to easily distinguish synthetic media from authentic recordings. In addition, we plan to adopt robust invisible watermarking techniques [45] that embed and reliably decode arbitrary data (*e.g.*, hyperlinks) in a perceptually invisible manner, while remaining resilient to real-world distortions such as compression, printing, and re-photography. These watermarks are designed to be difficult to remove without degrading visual quality.
- **Strict licensing.** Our code and models will be released under a restrictive license that prohibits the creation of avatars based on real individuals without explicit consent, particularly for commercial purposes. The license further restricts usage to ethical and non-deceptive applications, and any violation can be traced through the embedded watermarking system.

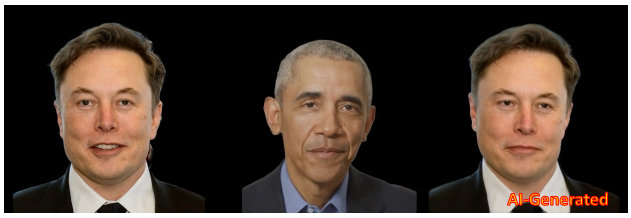

Figure 7. Visible watermarks will be embedded in all generated images and videos, clearly indicating that the content is AI-generated.

While our method advances the state of animatable head generation, we acknowledge its dual-use potential. Through technical measures like watermarking and policy-level controls via licensing, we aim to minimize the risk of abuse. As technology developers, we have the responsibility to build safeguards into our systems. However, preventing misuse requires broader efforts, from platform policies, legal frameworks, and user awareness. We call on researchers, developers, and content creators to exercise ethical judgment and social responsibility when deploying generative avatar systems. With appropriate oversight and responsible use, our work can contribute positively to immersive communication and other beneficial applications.

8. Reproducibility Details

8.1. Dataset Processing

We construct our training and testing datasets by uniformly sampling frames from the videos in the VFHQ dataset. Specifically, we use 15, 204 video clips for training and 50 clips for testing. For the training set, we sample a number of frames N per clip based on the video length, following the strategy proposed in Chu and Harada [6]:

- $N = 25$ if the video length is less than 200 frames,
- $N = 50$ if the video length is between 200 and 300 frames,
- $N = 75$ if the video length exceeds 300 frames.

This results in a total of 766, 263 training frames. For test-

Subdivision Times	Head Only	Head+Shoulder on VFHQ	Head+Shoulder on HDTF
#0	5, 023	13, 883	14, 466
#1	20, 018	28, 878	29, 461
#2	79, 936	88, 796	89, 379
Shoulder Points		+8, 860	+9, 443

Table 5. The number of Gaussian points for different subdivision level. Integrating the shoulder region leads to an average increase of 8, 860 Gaussian points on the VFHQ dataset and 9, 443 on the HDTF dataset.

	# Transformer Layers	# Attention Heads Per Layer	Attention Map Size	Feature Dimension	Training GPU Hours
LAM	10	16	$80\text{k} \times N_{\text{DINO}}$	1024	$\sim 2600\,\text{h}$ (2 weeks $\times$ 8 GPUs)
Ours	2	8	$5\text{k} \times N_{\text{DINO}}$	256	$\sim 200\,\text{h}$

Table 6. Comparison of training configurations for LAM and ours.

ing, we sample $N = 50$ frames per video, yielding 2, 500 test frames in total.

To evaluate the generalization ability of our model, we directly test it on the HDTF dataset using weights trained on the VFHQ dataset. We follow the dataset split setting from Ma et al. [36], and uniformly sample 100 frames per video, resulting in a total of 1, 900 frames for evaluation.

8.2. More Implementation Details

We utilize a frozen DINOv2 model to extract both local and identity features from an input image.The local feature has a size of $256 \times 296 \times 296$, while the identity feature is of size 1369×768. The original FLAME mesh contains 5, 023 vertices, and its positional encoding has a size of $5, 023 \times 256$. After passing through a two-layer Transformer, we obtain a global feature map with dimensions $5, 023 \times 256$. Subdividing the global feature results in hierarchical representations at level 1 and level 2, with sizes of $20, 018 \times 256$ and $79, 936 \times 256$, respectively. The first dimension of these features corresponds to the number of vertices at each subdivision level.

During training, the number of subdivisions is progressively increased according to the training progress. Specifically, no subdivision is applied in the early phase ($\leq 10\%$ of total iterations), one level is used in the intermediate phase (10%–30%), and a random strategy with a bias towards 2 (70% for 2, 20% for 1, and 10% for 0) in the later stage. The model is trained for 6 epochs on a single NVIDIA A100 GPU using the Adam optimizer and a linear learning rate decay schedule, with an initial learning rate of 1×10^{-4} and a batch size of 8.

9. Mesh Subdivision

We apply the Loop subdivision algorithm [34] to perform mesh subdivision on the FLAME model, utilizing its implementation in PyTorch3D. The Loop subdivision algorithm refines a triangle mesh by introducing a new vertex at the midpoint of each edge and dividing each triangular face into four smaller triangles. Additionally, vertex attribute vectors (*e.g.*, F^{GS}_{global} in Eq. (3)) are subdivided by averaging the attribute values of the two vertices that form each edge.

In Table 5, we present the number of Gaussian points for different subdivision levels, and the shoulder region approximately adds 9K additional points.

10. Computational Cost Analysis

Similar to our approach, the LAM method also utilizes mesh subdivision to increase the number of points, thereby improving the Gaussian's capability to capture fine-grained details. Both methods set the number of subdivision iterations to 2. However, the computational complexity differs significantly. Let k denote the subdivision level. After k iterations, the number of vertices in the mesh is approximately $4^k V_0$, where V_0 represents the number of vertices in the original FLAME mesh. Consequently, the computational complexity of the most expensive module, the cross attention, is:

$$O(l \cdot h \cdot 4^k \cdot V_0 \cdot N_{\text{DINO}} \cdot d_{\text{head}}) \tag{13}$$

where l denotes the number of transformer layers, h is the number of attention heads, N_{DINO} is the number of DINO features, and d_{head} is the feature dimension. In our method, cross-attention is computed at the 0-th level (before any subdivision), while LAM performs it on the finest-level subdivided mesh, where the computational complexity grows exponentially with the number of subdivisions k. As a result, the reconstruction cost of our method is only $1/640$ that of LAM. As shown in Tab. 6, our training GPU-hours are more than 90% lower than LAM's.

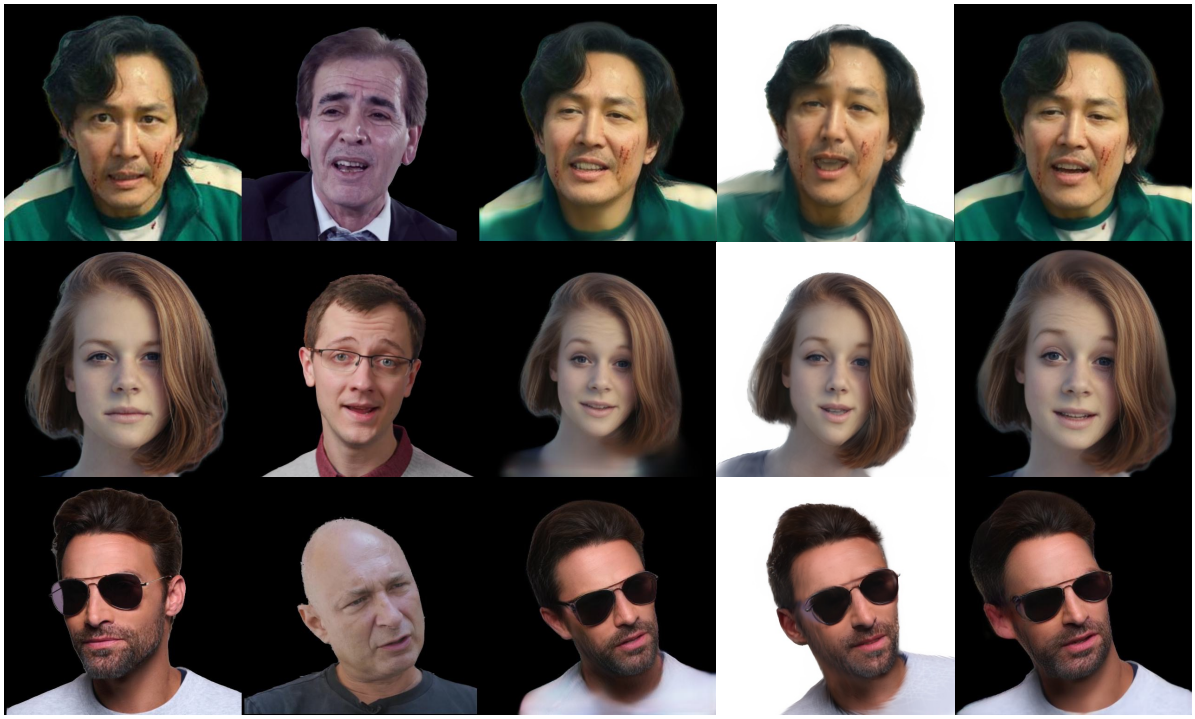

Figure 8. Comparision with state-of-the-art methods that support one-shot, feed-forward 3D avatar reconstruction with real-time facial reenactment.

Figure 9. Reenactment and multi-view results of OMG-Avatar on in-the-wild images.

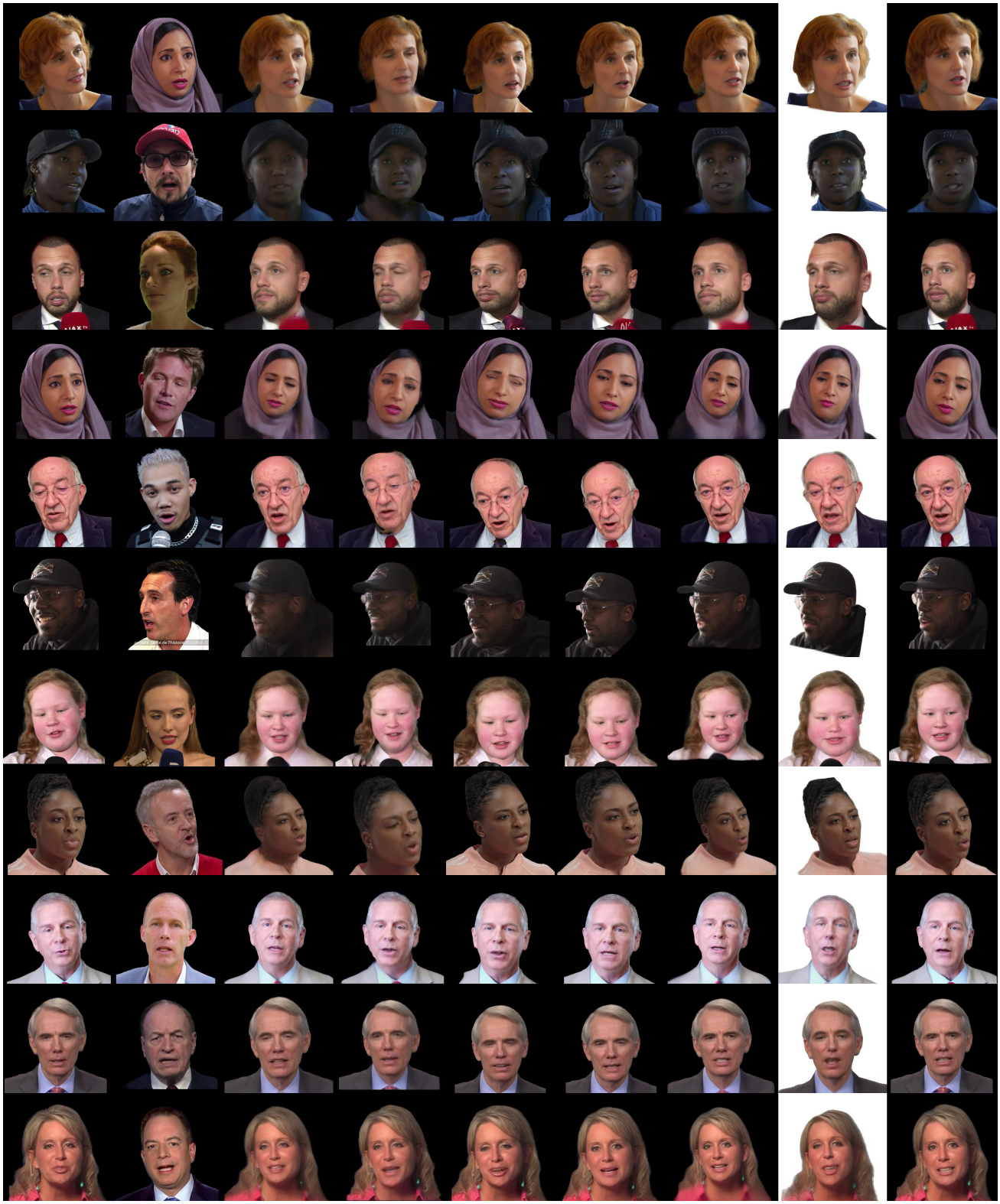

Figure 10. More cross-identity reenactment results on VFHQ and HDTF datasets.

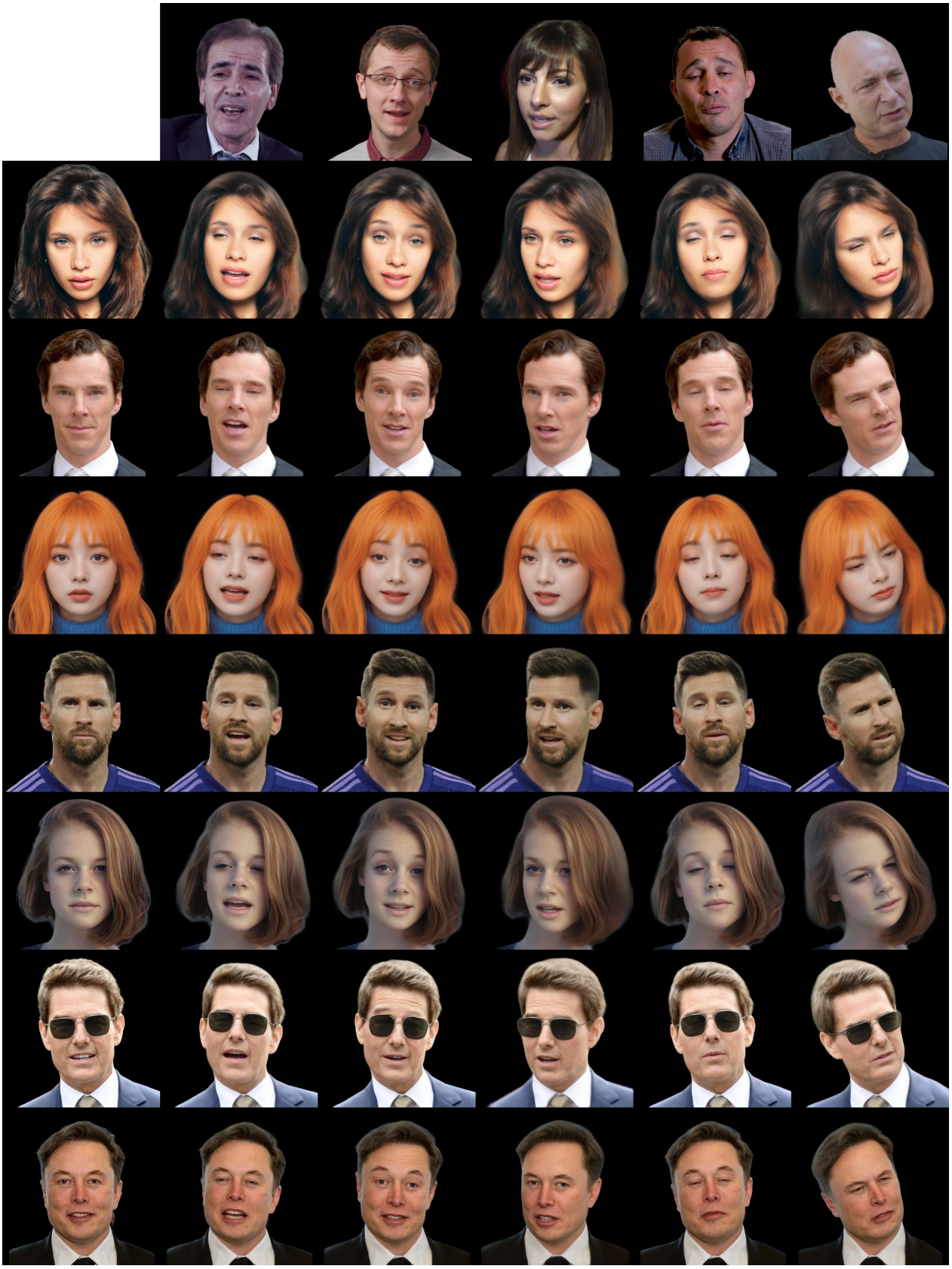

Figure 11. Visualization of cross-reenacted results on in-the-wild images.

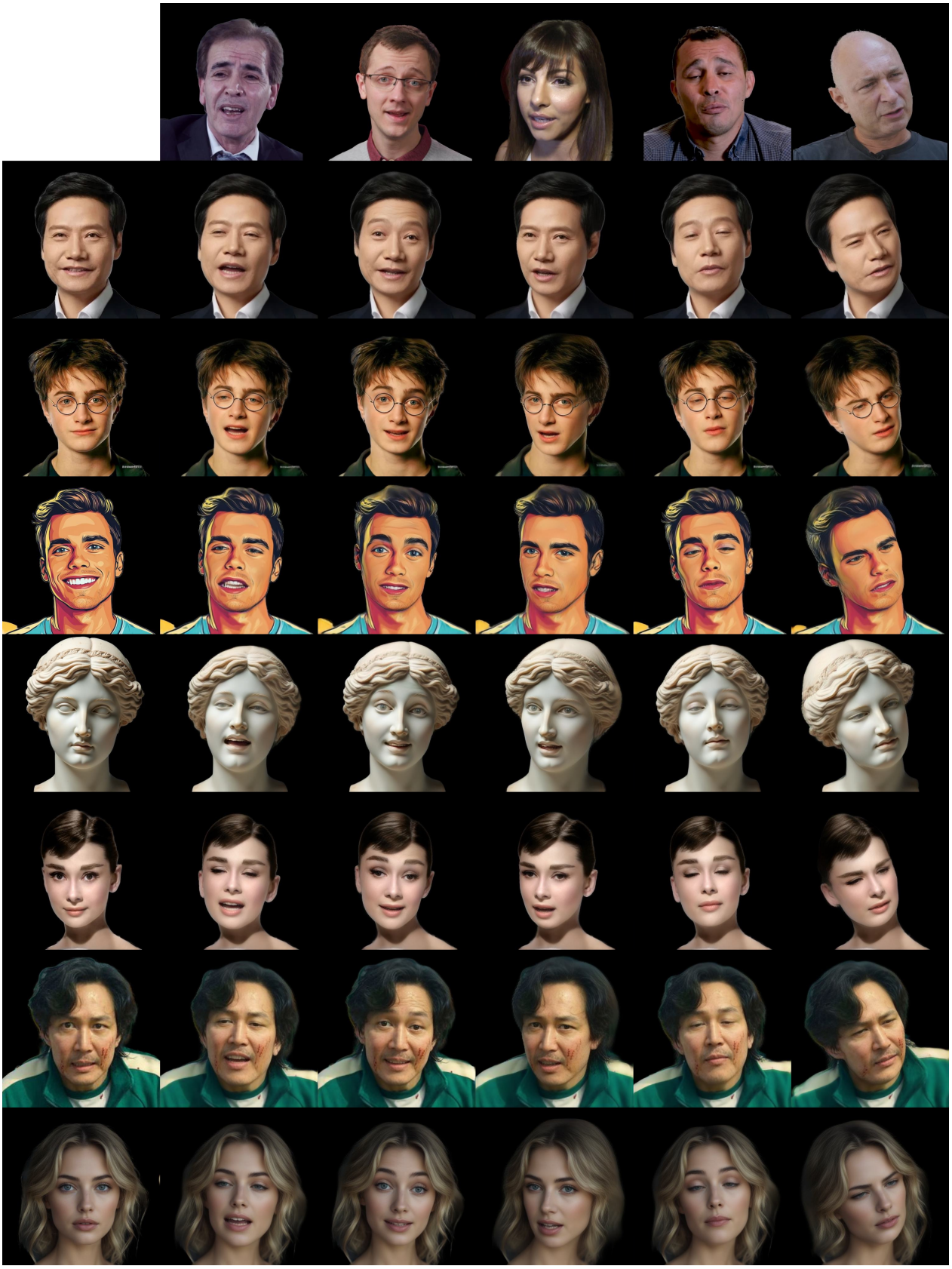

Figure 12. Visualization of cross-reenacted results on in-the-wild images.