StrokeFaceNeRF: Stroke-based Facial Appearance Editing in Neural Radiance Field

Supplementary Material

In this supplement, we provide a comprehensive overview of the color stroke image generation methods (Section 1) and additional visual results (Section 2). We also introduce our user interface in Section 3. Subsequently, we delve into the specifics of our implementation (Section 4), including network architecture details and mask padding strategy. Further insights into our experiments are discussed in Section 5, encompassing additional ablation studies on the two-stage training and further explanations for experiments. Additionally, we address potential artifacts (Section 6) that serve as subjects for future research. We encourage readers to explore the accompanying supplemental video, which features live demonstrations of our appearance editing process alongside additional visual results.

1. Color Stroke Images Generation

During the dataset construction, we synthesize facial images and corresponding stroke maps. Typically, users draw simple monochromatic strokes to simplify interaction. Therefore, synthesized color stroke images should preserve the original image's color and geometric layout while eliminating high-frequency texture features and local details.

Directly applying methods like Gaussian filtering to remove high-frequency information often results in edge blurring, leading to the generation of unreasonable color stroke images. To address this issue, the bilateral filtering method [11] is employed. Bilateral filtering, an extension of Gaussian filtering, incorporates weights related to pixel values. When the pixel values within the neighborhood differ significantly from the central pixel, the corresponding weight decreases. This approach effectively maintains the edge features of the facial image while eliminating high-frequency details. Specifically, for a filtering window centered at position q, consisting of a set of pixels S, the filtered result I_{BF} is computed as follows:

$$I_{BF} = \frac{1}{W_q}\sum_{p\in S} exp(-\frac{||p-q||^2}{2\sigma_s^2})exp(-\frac{||I_p-I_q||^2}{2\sigma_r^2})I_p \tag{1}$$

Where, σ_s and σ_r are user-specified parameters, I_p and I_q represent the pixel values corresponding to positions p and q, respectively. W_q represents the sum of weights within the filtering window:

$$W_q = \sum_{p\in S} exp(-\frac{||p-q||^2}{2\sigma_s^2})exp(-\frac{||I_p-I_q||^2}{2\sigma_r^2}) \tag{2}$$

To generate the color stroke image, the image is first processed using median filtering with a window size of 5. Then, bilateral filtering is applied iteratively for 20 rounds with σ_s and σ_r values set to 7 and 25, respectively.

Nevertheless, some regions exhibit smooth transitions between different colored areas after filtering, posing challenges for users aiming to replicate specific effects. To tackle this issue, we employ a pixel value clustering technique based on semantic information. Initially, the facial image undergoes semantic segmentation. Within the hair area, clustering allows all pixel colors to be replaced by the nearest cluster center color, thereby eliminating smooth transition regions. The same methodology is applied to the facial area, yielding the final color stroke image.

2. Additional Results

Synthetic Image Appearance Editing Results. Figure 1 presents additional appearance editing results produced by our methods. We randomly sampled latent codes from EG3D and performed diverse color stroke editing. Our model demonstrates its remarkable capability in flexibly modifying various appearance aspects from different angles, including lightning, hair styles, skin attributes, glasses, and facial patterns.

Real Image Appearance Editing Results. Figure 2 provides more results of real image appearance editing through Pivotal Tuning Inversion (PTI) [8] and our approach. This pipeline allows diverse and high quality editing over real facial image and generates corresponding 3D reconstruction. It holds significant promise for a variety of applications such as personalized user-driven 3D avatar editing.

Inaccurate masks and random strokes Editing Results. Figure 3 demonstrates robustness of our method in handling inaccurate inputs of masks and color strokes. For instance, in Figure 3(b) where the mask overlaps the forehead, the generated image still has reasonable edited effects. Despite the inaccurate mask, the stroke input primarily affects the hair, maintaining the color in the forehead. Additionally, in Figure 3(d) where users draw random or tousle strokes, our method adapts to generate intended edited results. More importantly, our method provides an interactive stroke-based editing system, which enable users to easily modify the mask and stroke input until achieving their desired results. In real drawing scenarios with possible inaccurate inputs, our method can still produces satisfactory results, or facilitates convenient user-driven adjustments.

Latent codes interpolation. Figure 4 provides linear

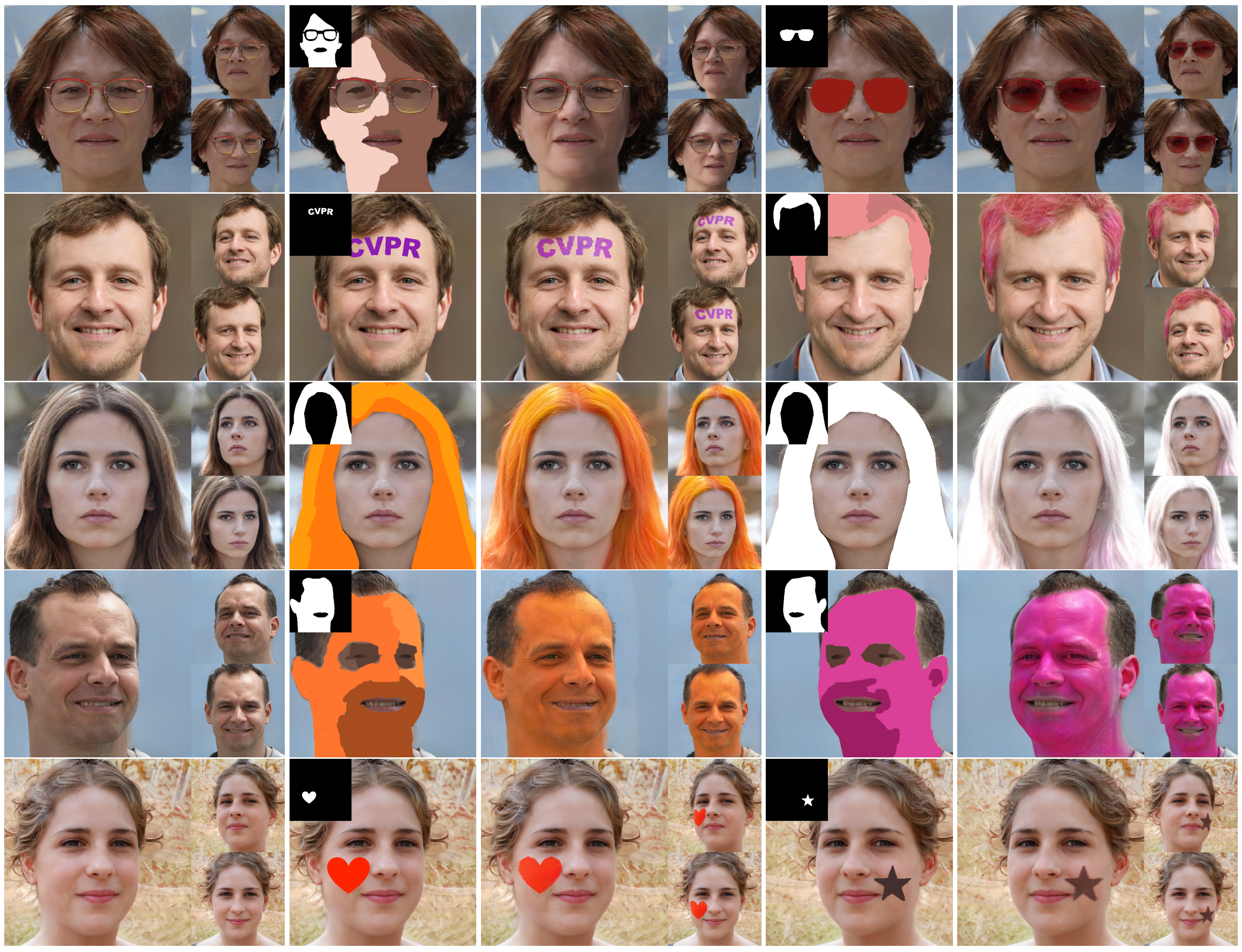


Figure 1. Additional color stroke-based facial NeRF appearance editing results. Given the original faces (a), users drawn the editing color strokes (b) (d). Our method generates the corresponding editing results (c)(e). Additional masks are shown in top left corner, which is the same as stroke's shape in the first four rows, and drawn by users in the last row.

interpolations between latent codes of distinct appearances produced by our method. The results validate that our 3D GAN inherits the well-behaved latent space of the StyleGAN2 [5] backbone and 3D-consistent editing results.

3. User Interface Design.

As shown in Figure 5, we design a user interface on top of the proposed pipeline to facilitate color stroke-based free view facial editing using [1]. Users can edit facial images randomly sampled from EG3D or upload real face images. Our system then synthesizes and displays photo-realistic 3D faces in the right panel. Users can change the views with the sliders on the control panel, and the 3D faces are rotated simultaneously. After editing from a single viewpoint, users can also rotate the face, engaging in continuous editing from alternative views, thereby supporting detailed and expressive facial appearance manipulation. Throughout the editing process, users have the flexibility to erase or reset strokes and draw new strokes to depict their desired appearance. Our system defaults to inferthe mask area, denoted as M, based on users' edits over the stroke images. Additionally, for users who prefer a more localized editing, we have included a tab for drawing masks to specify particular areas of interest. This offers users greater control over the editing process, allowing for both global and local modifications to achieve the desired facial appearance.

4. Implementation Details

We implemented our framework by building upon the official PyTorch implementation of EG3D [2] avail-

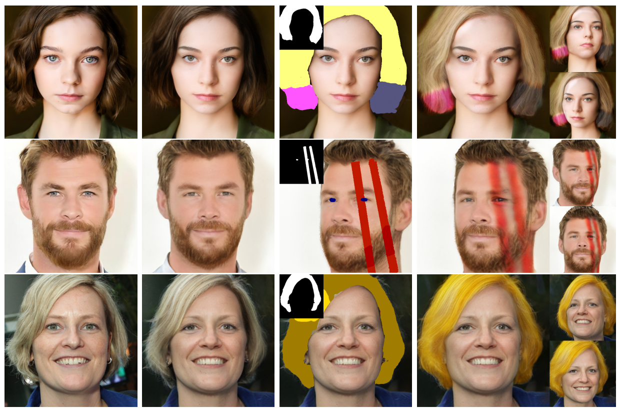


Figure 2. Additional real image appearance editing results.

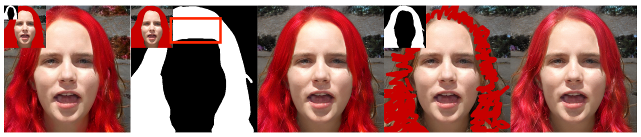


Figure 3. Inaccurate masks and random strokes Editing Results

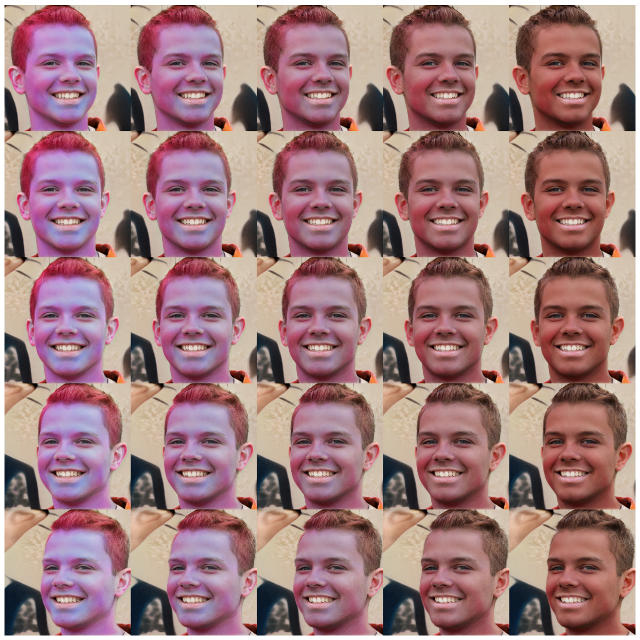
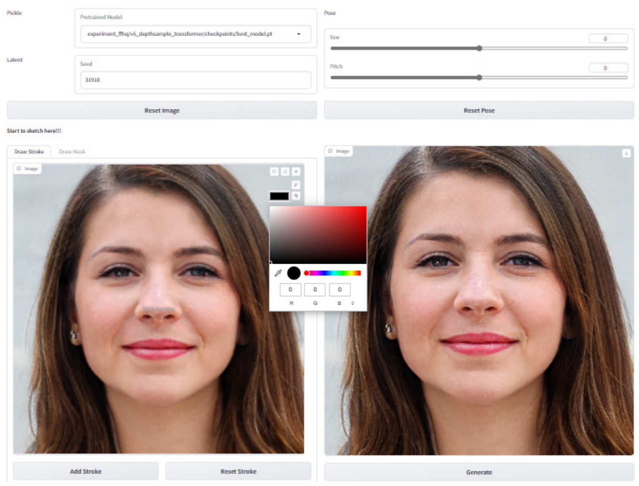


Figure 4. Latent codes interpolation. We linearly interpolate between latent codes representing distinct appearances of an individual. In each row, we keep the camera pose constant and interpolate the latent codes. In each column, we fix the latent code and show the rendering results of different camera angles.

able at https://github.com/NVlabs/eg3d and the official implementation of the pSp [7] network, accessible at



Pickle
Pretrained Model: experiment_ffhq/v5_depthsample_transformer/checkpoints/best_model.pt
Latent
Seed: 31918
Pose
Yaw: 0
Pitch: 0
Start to sketch here!!!

Figure 5. The user interface of StrokeFaceNeRF. The interface empowers users to apply color strokes to generated images or upload their own images. The resulting edited images are displayed in the right panel. A control panel positioned at the top of the interface facilitates various key operations, encompassing the model weights, manipulation of random seed, and control over viewpoint rotation.

https://github.com/eladrich/pixel2style2pixel. We adopt the hierarchical encoder network architecture from pSp as our color encoder, and the EG3D generator is employed to generate the final facial NeRF based on the fused latent codes and mask inputs.

The processing time. For data preprocessing, given facial images, the generation of color stroke images through bilateral filtering, requires approximately 3 seconds. Subsequently, during stroke-based editing, the average time for both model inference and image rendering is about 1.1 seconds, facilitating nearly real-time interactive stroke edits.

Latent Fusion Network. Our latent fusion network is a cross-attention transformer block, customized for improved fusion between the color stroke latent code w_c and the original facial latent feature w. Specifically, the input query tokens Q are derived as $w_c W_Q$, while the key tokens K and value tokens V are projected from w as wW_K and wW_V respectively. To mitigate the potential influence of the original latent code, we employ a routing scheme inspired by [3] to align the value features with queries uniquely. More precisely, the Softmax layer $Softmax_Q$ normalizes over queries instead of keys and subsequently re-normalizes the matrix over keys using $Norm_K$. The final edited latent code w_{edit} is calculated as follows:

$$w_{edit} = Norm_K(Softmax_Q(\frac{QK^T}{\sqrt{d}}))V \tag{3}$$

Mask Padding. Our volumetric mask fusion method effectively facilitates local editing and preserves original fea-

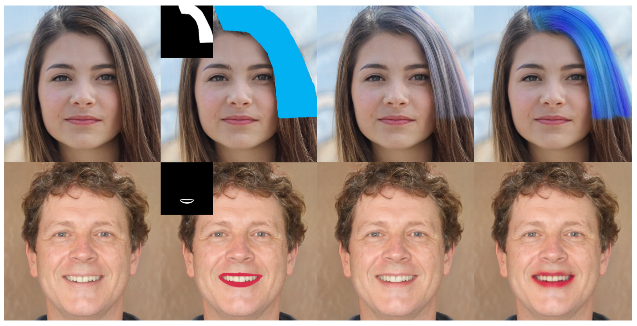


Figure 6. Additional ablation study results.

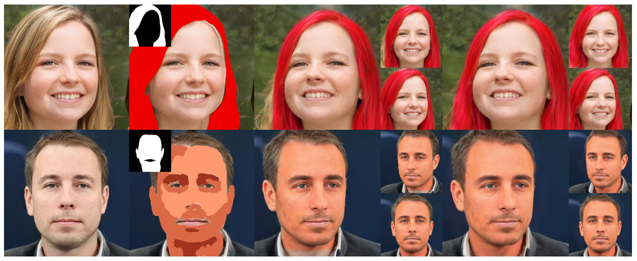


Figure 7. Mask padding editing results.

tures in unedited regions through a 2D mask input. However, challenges arise during significant viewpoint changes, as depicted in Figure 7. To tackle this issue, we employ a mask padding technique. When projecting the 2D mask onto the 3D space using a user-specified camera pose, we introduce directional offsets to the camera positions, specifically setting them to 0.2 for upward, downward, leftward, rightward, and diagonal directions. For mask boundaries aligned with the original image's boundary, we extend the boundary pixels into newly offset regions, synthesising an expended mask image. These new augmented masks are then projected into 3D space using the offset camera poses, and the resulting point sets are merged to construct the mask volume. This adjustment enhances the completeness of the 3D point cloud obtained from the projection of the 2D mask, thereby improving 3D consistency.

5. Experiment Details

5.1. Baselines

SC-FEGAN [4] is based on an image completion approach, using color strokes as control conditions to fill the editing area. We utilized the official code at https://github.com/run-youngjoo/SC-FEGAN and its pretrained model weights.

SDEdit [6] employs a diffusion model, guided by user-drawn color strokes to fill the editing areas based on a pre-trained generative model. We utilized the official code at https://github.com/ermongroup/SDEdit and its pretrained model weights.

pSp [7] is an efficient 2D image-to-image translation framework based on the StyleGAN generator. To establish a baseline for 3D stroke-based editing, we retrained a modified pSp network using the official codes on our datasets. This network directly projects the drawn color stroke images into EG3D's latent space, obtaining 3D facial results represented by latent codes. To better preserve the original facial features, another baseline, denoted as pSp-mixing, applies style-mixing techniques on pSp's predicted latent code by replacing layers 7-14 with the original latent code, then produces the final facial NeRF.

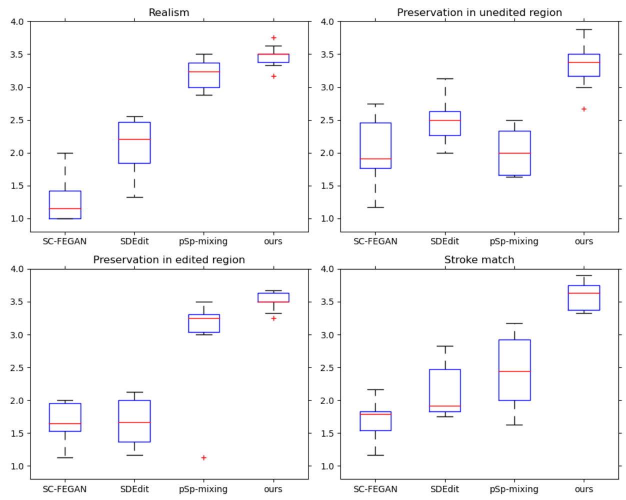


Figure 8. User study results. Our method stands out from other methods in all evaluating indicators, producing more preferable and consistent editing results.

5.2. User Study

We conducted a user study comparing our method with baselines. Ten randomly selected cases from evaluation dataset were presented to 15 participants (7 males, 8 females, aged 18-35). Participants scored edited faces from 1 to 4 based on the realism of generated faces, content preservation within unedited regions, geometry preservation within edited regions, and stroke match level. Results, presented as boxplots in Figure 8, show our method consistently outperformed others in all criteria, producing more desirable and consistent edits.

5.3. Additional Ablation Study

2 Stage training. To justify our proposed 2 stage training strategy and the architecture of latent fusion network, we conduct additional ablation experiments. Both baseline approaches are trained on the same datasets with our approaches. As depicted in Figure 6 (c), the removal of the two-stage training strategy, where both modules are trained simultaneously, poses challenges to network optimization, leading to incomplete effects in color stroke editing, and the correspondence with input strokes greatly deteriorates.

Gaussian filter radius. In accordance with Sec 3.3, our method incorporates a Gaussian filtering process to formulate the final 3D Mask, ensuring a seamless transition for edited regions. Users are provided with the flexibility to regulate the smoothness levels by adjusting the filter radius. Notably, when employing a radius value of 1, distinct boundaries in the generated graphic patterns are observed in the image below. Our method adopts a default filter radius value of 2, demonstrating optimal performance in achieving smooth fusion outcomes for appearance edits, such as modifications to hair and skin color.

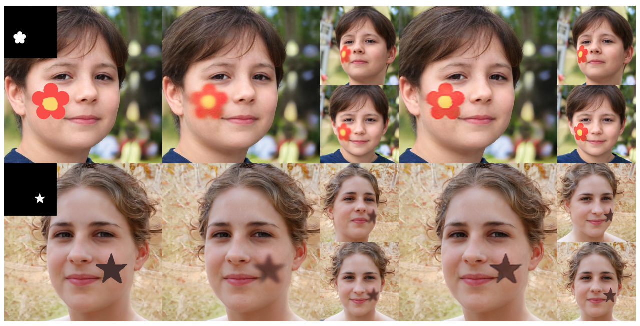


Figure 9. Filter radius ablation study results.

6. Discussion

Despite the success of our method in achieving high-quality color editing through color strokes, it is essential to address certain limitations that can impact its performance. Firstly, when it comes to highly intricate regions such as eyeballs, our method sometimes encounters challenges in producing optimal editing results. To overcome this limitation, implementing specialized processing for these detailed areas, possibly through segmentation, could significantly enhance the editing accuracy and effectiveness.

Additionally, the constraints imposed by our training dataset present difficulties in handling exceedingly complex color strokes, such as rainbow-colored hair. In such cases, our method tends to average these strokes, resulting in a potential loss of intricacies. A potential avenue for improvement involves the generation of distinct colors independently, coupled with subsequent mask creation and blending, to achieve more nuanced and refined editing results.

7. Ethical Discussion

While our work primarily focuses on positive applications such as virtual avatar design and user customized photo retouching, we acknowledge the importance of addressing ethical implications. We are committed to promoting positive usage of our technology and advocate for ethical considerations in its application. we do not condone the potential misuse of the facial editing tools. Many works [9, 10, 12] in the fake detection community could discriminate between the synthesized and real faces, and partly prevent the misuse.

References

[1] Abubakar Abid, Ali Abdalla, Ali Abid, Dawood Khan, Abdulrahman Alfozan, and James Zou. Gradio: Hassle-free sharing and testing of ml models in the wild. *arXiv preprint arXiv:1906.02569*, 2019. 2

[2] Eric R. Chan, Connor Z. Lin, Matthew A. Chan, Koki Nagano, Boxiao Pan, Shalini De Mello, Orazio Gallo, Leonidas Guibas, Jonathan Tremblay, Sameh Khamis, Tero Karras, and Gordon Wetzstein. Efficient geometry-aware 3D generative adversarial networks. In *CVPR*, 2022. 2

[3] Xueqi Hu, Qiusheng Huang, Zhengyi Shi, Siyuan Li, Changxin Gao, Li Sun, and Qingli Li. Style transformer for image inversion and editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11337–11346, 2022. 3

[4] Youngjoo Jo and Jongyoul Park. Sc-fegan: Face editing generative adversarial network with user's sketch and color. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1745–1753, 2019. 4

[5] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *Proc. CVPR*, 2020. 2

[6] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021. 4

[7] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: a stylegan encoder for image-to-image translation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2287–2296, 2021. 3, 4

[8] Daniel Roich, Ron Mokady, Amit H Bermano, and Daniel Cohen-Or. Pivotal tuning for latent-based editing of real images. *ACM Trans. Graph.*, 2021. 1

[9] Andreas Rossler, Davide Cozzolino, Luisa Verdoliva, Christian Riess, Justus Thies, and Matthias Nießner. Faceforensics++: Learning to detect manipulated facial images. In *International Conference on Computer Vision*, pages 1–11, 2019. 5

[10] Ruben Tolosana, Ruben Vera-Rodriguez, Julian Fierrez, Aythami Morales, and Javier Ortega-Garcia. Deepfakes and beyond: A survey of face manipulation and fake detection. *Information Fusion*, 64:131–148, 2020. 5

[11] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*, pages 839–846. IEEE, 1998. 1

[12] Hanqing Zhao, Wenbo Zhou, Dongdong Chen, Tianyi Wei, Weiming Zhang, and Nenghai Yu. Multi-attentional deepfake detection. In *Conference on Computer Vision and Pattern Recognition*, pages 2185–2194, 2021. 5