DEFORMTOON3D: Deformable Neural Radiance Fields for 3D Toonification – Supplementary Material –

Junzhe Zhang[1,3*] Yushi Lan[1*] Shuai Yang[1] Fangzhou Hong[1]
Quan Wang[3] Chai Kiat Yeo[2] Ziwei Liu[1] Chen Change Loy[1]
[1]S-Lab, Nanyang Technological University
[2]Nanyang Technological University [3]SenseTime Research
{shuai.yang,asckyeo,ziwei.liu,ccloy}@ntu.edu.sg
{junzhe001,yushi001, fangzhou001}@e.ntu.edu.sg {wangquan}@sensetime.com

1. Background

Since recent 3D-aware image generative models are all based on neural implicit representations, especially NeRF [10], here we briefly introduce the NeRF-based 3D representation and more StyleSDF details for clarification.

NeRF-based 3D Representation. NeRF [10] proposed an implicit 3D representation for novel view synthesis. Specifically, NeRF defines a scene as $\{\boldsymbol{c}, \sigma\} = F_\Phi(\boldsymbol{x}, \boldsymbol{v})$, where $\boldsymbol{x}$ is the query point, $\boldsymbol{v}$ is the viewing direction from camera origin to $\boldsymbol{x}$, $\boldsymbol{c}$ is the emitted radiance (RGB value), σ is the volume density. To query the RGB value $C(\boldsymbol{r})$ of a point on a ray $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{v}$ shoot from the 3D coordinate origin $\boldsymbol{o}$, we have the volume rendering formulation,

$$C(\boldsymbol{r}) = \int_{t_n}^{t_f} T(t)\sigma(\boldsymbol{r}(t))\boldsymbol{c}(\boldsymbol{r}(t), \boldsymbol{v})dt, \tag{1}$$

where $T(t) = \exp(-\int_{t_n}^{t} \sigma(\boldsymbol{r}(s))ds)$ is the accumulated transmittance along the ray $\boldsymbol{r}$ from t_n to t. t_n and t_f denote the near and far bounds.

Hybrid 3D Generation. In hybrid 3D generation [11, 1, 6], the intermediate feature map is calculated by replacing the color $\boldsymbol{c}$ with feature $\mathbf{f}$, namely $\mathbf{F}(\mathbf{r}) = \int_{t_n}^{t_f} T(t)\sigma(\mathbf{r}(t))\mathbf{f}(\mathbf{r}(t), \boldsymbol{v})dt$. Then, a StyleGAN [7, 8]-based decoder upsamples $\mathbf{F}$ into high-resolution images with high-frequency details.

SDF and Radiance-based Geometry Representation. The intermediate geometry representation of $G_\mathbf{0}$ diversifies the characteristics of different 3D GANs. Specifically, StyleSDF [11] uses $G_\mathbf{0}$ to predict the signed distance $d(\boldsymbol{x}) = G_\mathbf{0}(\mathbf{w}, \boldsymbol{x})$ between the query point $\boldsymbol{x}$ and the shape surface, where the density function $\sigma(\boldsymbol{x})$ can be transformed from $d(\boldsymbol{x})$ [11, 15, 16] for volume rendering [10]. The incorporation of SDF leads to higher-quality geometry in terms of expressiveness view consistency and clear definition of the surface.

In this paper, we mainly adopt StyleSDF [11] due to its high-quality geometry surface and high-fidelity texture. In StyleSDF, the Sigmoid activation function σ is replaced by $\sigma(\boldsymbol{x}) = K_\alpha\left(d(\boldsymbol{x})\right) = \text{Sigmoid}\left(-d(\boldsymbol{x})/\alpha\right)/\alpha$, where α is a learned parameter that controls the tightness of the density around the surface boundary.

2. Implementation Details

CIPS-3D Baseline. Following CLIPS-3D [18], we fine-tune $G_\mathbf{1}$ of StyleSDF on the toonified images with identical optimization parameters in the official implementation. The fine-tuning time for one style costs 10 V100 minutes.

E3DGE Baseline. Following E3DGE [9], we first fine-tune $G_\mathbf{0}$ for 400 iterations with batch size 24, and further fine-tune $G_\mathbf{1}$ for 400 iterations with batch size 8. All hyper-parameters are left unchanged with the official StyleSDF [11] implementation. The overall fine-tuning time for one style costs around 30 minutes on a single V100 GPU.

StyleGAN-NADA Baseline. We reproduce StyleGAN-NADA [5] on StyleSDF with the following modifications. For $G_\mathbf{0}$ optimization, we fix the pre-trained mapping network, affine code transformations, view-direction MLP, color-prediction MLP, and density-prediction MLP. For $G_\mathbf{1}$ optimization, we follow the original implementation and fine-tune all weights except toRGB layers, affine code transformations, and mapping network. The k layers to optimize are also selected adaptively using StyleCLIP global loss. Other hyper-parameters and training procedures are left unchanged. The whole optimization costs around 5 minutes on a single V100 GPU.

*Equal contribution.

2.1. Additional Method Details

StyleField. Given the instance code $\mathbf{w}$, style code $\boldsymbol{z}_S$, we concatenate them along the channel dimension and send them into a 4-layer mapping network [7]. The mapping network first maps $\mathbf{w} \oplus \boldsymbol{z}_S$ to a set of modulation signals $\{\boldsymbol{\beta}, \boldsymbol{\gamma}\}$, where $\boldsymbol{\beta} = \{\beta_i\}, \boldsymbol{\gamma} = \{\gamma_i\}$. To associate the given codes to the corresponding deformation, the modulation signals will be injected into the MLP network, serving as FiLM conditions [12, 4, 14] to modulate its features at different layers as $\boldsymbol{f}_{i+1} = \sin(\gamma_i \cdot (\boldsymbol{W}_i \boldsymbol{f}_i + \boldsymbol{b}_i) + \beta_i)$. To support multi-style code, we associate each style index with a learnable embedding. During inference, we pass in the corresponding style index to retrieve the style embedding for conditional deformation.

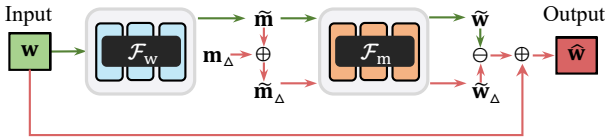

Figure 1: Inference pipeline of the proposed animation pipeline.

Animatable Stylized Portrait Training Details. We show the overall 3DMM animation inference pipeline in Fig. 1. Specifically, we train the whole framework in a self-supervised manner. In each iteration, we synthesize a batch of pose images $\mathbf{I} = G(\mathbf{w})$, where $G = G_\mathbf{1} \circ G_\mathbf{0}$. For 3DMM supervision, we leverage the state-of-the-art 3DMM predictor [3] to infer the pseudo ground-truth 3DMM parameter $\mathbf{m}_{\text{GT}}$. With the synthesized training corpus, we reconstruct the input codes $\hat{\mathbf{m}} = \mathcal{F}_\mathbf{w}(\mathbf{w})$ and $\hat{\mathbf{w}} = \mathcal{F}_\mathbf{m}(\mathbf{m}_{\text{GT}})$ and impose MSE reconstruction loss. We further render the reconstructed code $\tilde{\mathbf{I}} = G(\hat{\mathbf{w}})$ and $\tilde{\mathbf{I}}_\mathcal{M} = \text{DFR}(\hat{\mathbf{m}})$, where DFR is a differentiable render [13] that renders the reconstructed 3DMM mesh to image. The rendered images are supervised with corresponding loss [3], which yields better performance in our observations.

This training objective shall guarantee plausible 3DMM editing using the procedure stated in the main context. However, in practice, we find the training is unstable and predicted codes are not disentangled well during inference. We make the following modifications to the overall training pipeline and improve the editing performance:

First, we observe that the 3DMM head pose parameters, including a head rotation $\mathbf{R} \in SO(3)$ and translation $\mathbf{t} \in \mathbb{R}^3$ parameters, are in contradict with the pose control of 3D GAN. This deteriorates the disentanglement of learned codes and destabilizes the training since the predicted 3DMM codes $\tilde{\mathbf{m}}$ must contain accurate head pose to minimize the reconstruction loss with $\text{DFR}(\mathbf{m})$. To address this issue, we mask out the head rotation $\mathbf{R}$ and translation $\mathbf{t} \in \mathbb{R}^3$ dimension in all 3DMM parameters with the binary mask. This enforces all the 3DMM images $\mathbf{I}_\mathcal{M}$ to be rendered from frontal pose and encourages the networks to focus on facial expression $\boldsymbol{\delta} \in \mathbb{R}^{64}$ alignment between $\mathcal{W}$ and $\mathcal{M}$.

Second, to further impose identity preservation and bijective mapping between two spaces, we introduce cycle training [19] which regularizes $\mathbf{w} \approx \mathcal{F}_\mathbf{w}(\tilde{\mathbf{m}})$ and $\mathbf{m} \approx \mathcal{F}_\mathbf{m}(\tilde{\mathbf{w}})$. The cycle loss is also imposed on the image space.

Third, to imitate the inference pipeline, in each training iteration, we randomly shuffle the expression dimension of all the 3DMM code $\mathbf{m}_{\text{GT}}$ within a batch and generate a new set of codes $\tilde{\mathbf{m}}_{\text{GT}}$. The rendered image from $\mathcal{F}_\mathbf{m}(\tilde{\mathbf{m}}_{\text{GT}})$ shall maintain the same identity with $\mathbf{I}$ with identical pose of $\text{DFR}(\tilde{\mathbf{m}}_{\text{GT}})$. We impose the identity preservation loss [2] and landmark loss over the rendered 3DMM image [3] as supervisions. This strategy further reduces the domain gap between training and inference and further improves the final editing performance.

Fourth, we further leverage the style-based hierarchical structure within StyleSDF and reduces the attribute entanglement. Specifically, rather than using the edited $\mathcal{W}$ code $\hat{\mathbf{w}}_\Delta$ for all the style layers in G, we conduct layer-wise editing effect analysis and find that only the first 2 layers of $G_\mathbf{0}$ will handle the expression-relevant information of the synthesized image. Using the edited code for later layers will result in other attributes editing, *e.g.*, adding glasses or changing the hair structure. Therefore, we leave the remaining 7 layers of $G_\mathbf{0}$ and all 10 layers in $G_\mathbf{1}$ unchanged and only use the edited code for the top 2 $G_\mathbf{0}$ layers. This yields better disentanglement during the 3DMM-controlled style editing.

Training-wise, we adopt identical MLP architecture from PixelNeRF [17] to implement both $\mathcal{F}_*$ networks and adopt a batch size of 4 with learning rate 5×10^{-4} during the optimization. The networks are trained for $50,000$ iterations, which costs around 2 days on a single V100 GPU. Please refer to the released code for more details.

2.2. Additional Ablation Study

The robustness of the number of style codes is ablated in Table 1. Experiments are conducted with 1, 2, and 5 styles per model so that under each setting the 10 styles can be evenly divided into different runs for the sake of comparison.

Table 1: **Ablation on the number of styles.**

# styles per model	1	2	5	10
Identity similarity↑	0.795	0.776	0.784	0.781
FID↓	27.5	28.1	27.9	27.6

The effectiveness of the elastic loss is also ablated qualitatively. As shown in the visualized mesh in Fig. 2, the elas-

Table 2: **Quantitative evaluation in terms of identity similarity**↑. DEFORMTOON3D achieves the best identity consistency over all the 10 styles.

Domains	CIPS-3D	E3DGE	NADA	Ours
Pixar	0.765	0.748	0.564	**0.812**
Comic	0.643	0.614	0.496	**0.729**
Slam Dunk	0.672	0.765	0.552	**0.780**
Caricature I	0.648	0.592	0.455	**0.708**
Caricature II	0.655	0.698	0.538	**0.785**
Caricature III	0.637	0.644	0.495	**0.725**
Croods	0.796	0.831	0.626	**0.860**
Shrek	0.708	0.794	0.599	**0.835**
Rapunzel	0.603	0.696	0.564	**0.782**
Hiccup	0.684	0.688	0.464	**0.796**
Average	0.681	0.707	0.535	**0.781**

Table 3: **Quantitative evaluation in terms of FID**↓. DEFORMTOON3D achieves the best FID over 9 of the 10 styles.

Domains	CIPS-3D	E3DGE	NADA	Ours
Pixar	33.6	36.8	39.9	**21.5**
Comic	61.9	44.8	70.9	**33.3**
Slam Dunk	78.1	41.8	75.9	**37.3**
Caricature I	28.7	30.1	52.9	**16.0**
Caricature II	76.7	58.1	102.6	**56.4**
Caricature III	47.1	**25.8**	54.8	27.2
Croods	36.9	30.9	58.5	**22.5**
Shrek	36.0	32.1	47.2	**20.3**
Rapunzel	65.0	30.5	44.1	**17.2**
Hiccup	42.3	32.8	46.6	**24.6**
Average	50.6	34.0	59.3	**27.6**

tic loss is effective in preventing discontinuous deformation and leads to smoother geometry in the styled space.

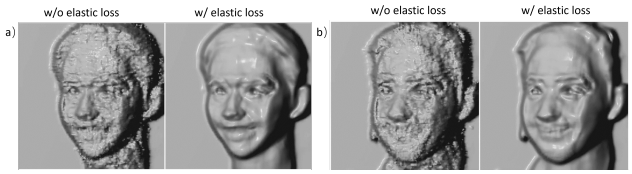

Figure 2: **Ablation on elastic loss.** Without *v.s.* with elastic loss for a) female and b) male, respectively.

2.3. Additional Quantitative and Qualitative Results

The detailed breakdown of toonification fidelity and quality are shown in Tab. 2 and Tab. 3 respectively. We include more qualitative experiment results here. In Fig. 3 we include more comparisons with the baseline methods, which demonstrates that DEFORMTOON3D produces better quality against existing methods. In Fig. 4 we show more toonification results over real images. The proposed methods yield plausible results with consistent identity preservations. We further include the stylized texture and shape pair in Fig. 5 and validate that our method produces high-quality stylization over both texture and shape.

Figure 3: **Additional qualitative comparisons with baseline methods.** DEFORMTOON3D produces better performance against all baselines regarding toonification fidelity, diversity and identity preservation. Better zoom in.

Figure 4: **Additional results of DEFORMTOON3D on the real images.** Our method enables multiple styles toonification with a single model, where both the texture and the geometry matches the target domain.

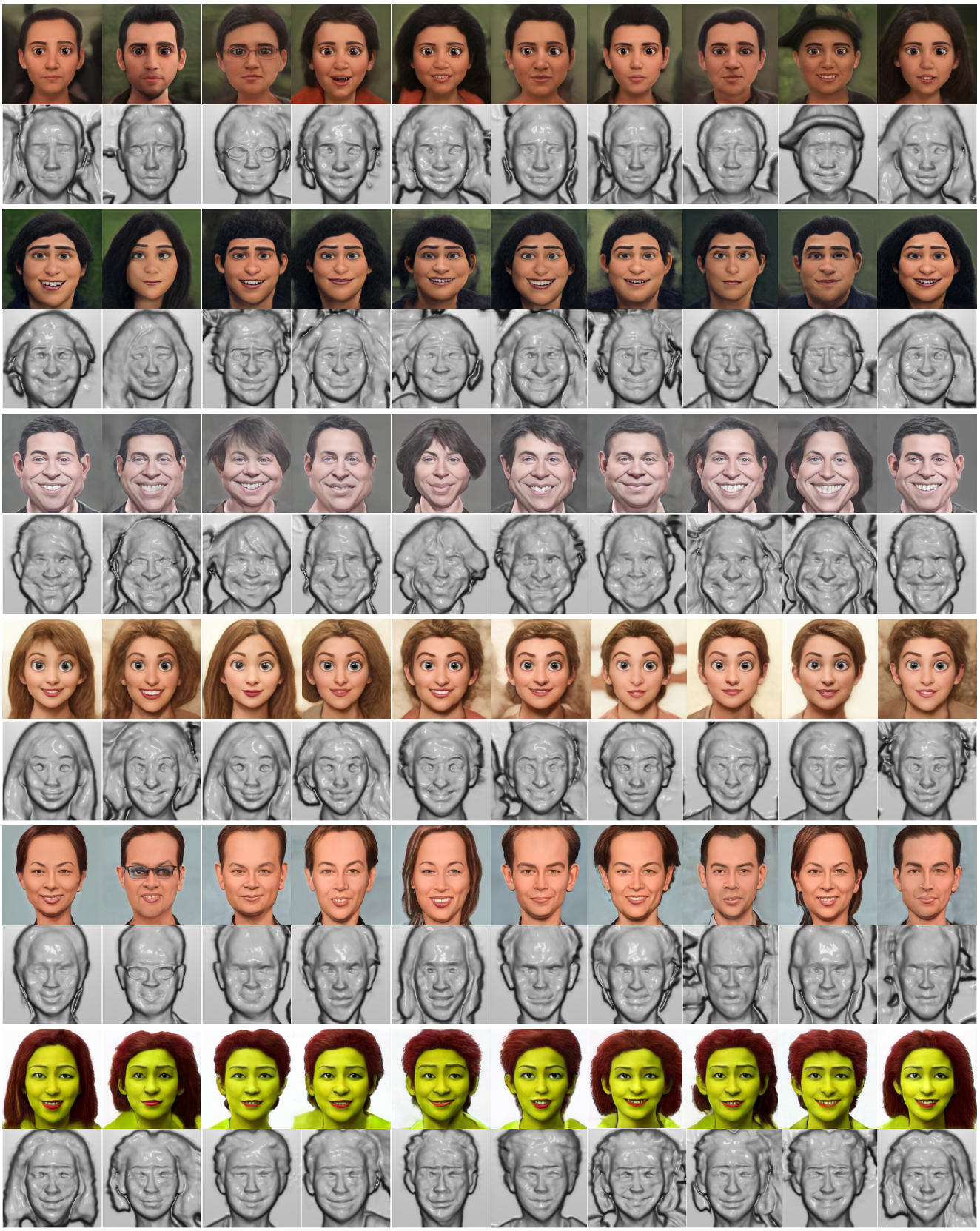

Figure 5: **Additional results of DEFORMTOON3D with stylized texture and shape.**

References

[1] 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. 1

[2] Jiankang Deng, Jia Guo, Xue Niannan, and Stefanos Zafeiriou. ArcFace: Additive angular margin loss for deep face recognition. In *CVPR*, 2019. 2

[3] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, and Xin Tong. Accurate 3D face reconstruction with weakly-supervised learning: From single image to image set. In *CVPR*, 2019. 2

[4] Vincent Dumoulin, Ethan Perez, Nathan Schucher, Florian Strub, Harm de Vries, Aaron Courville, and Yoshua Bengio. Feature-wise transformations. *Distill*, 2018. https://distill.pub/2018/feature-wise-transformations. 2

[5] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. StyleGAN-NADA: Clip-guided domain adaptation of image generators. *arXiv*, abs/2108.00946, 2021. 1

[6] Jiatao Gu, Lingjie Liu, Peng Wang, and Christian Theobalt. StyleNeRF: A style-based 3D-aware generator for high-resolution image synthesis. In *ICLR*, 2021. 1

[7] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1, 2

[8] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. In *CVPR*, 2020. 1

[9] Yushi Lan, Xuyi Meng, Shuai Yang, Chen Change Loy, and Bo Dai. E3DGE: Self-supervised geometry-aware encoder for style-based 3D gan inversion. In *CVPR*, 2022. 1

[10] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *ECCV*. Springer, 2020. 1

[11] Roy Or-El, Xuan Luo, Mengyi Shan, Eli Shechtman, Jeong Joon Park, and Ira Kemelmacher-Shlizerman. StyleSDF: High-Resolution 3D-Consistent Image and Geometry Generation. In *CVPR*, 2021. 1

[12] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *AAAI*, volume 32, 2018. 2

[13] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 2

[14] Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *NIPS*, 2020. 2

[15] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. In *NeurIPS*, 2021. 1

[16] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *NIPS*, 2021. 1

[17] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. PixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021. 2

[18] Peng Zhou, Lingxi Xie, Bingbing Ni, and Qi Tian. CIPS-3D: A 3D-Aware Generator of GANs Based on Conditionally-Independent Pixel Synthesis. *arXiv*, 2021. 1

[19] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2