UniFaceGAN: High-Quality 3D face Editing with a Unified Latent Space

Jinfu Wei[1], Zheng Zhang[2], Ran Liao[1*], Duan Gao[2*]
[1]Shenzhen International Graduate School, Tsinghua University, Shenzhen, China
[2]Huawei Cloud Computing Technologies Co., Ltd., Shenzhen, China

***Abstract*—Recent advancements in 3D face generation have explored various representation and generative models. However, these methods often offer limited 3D face editing capabilities. In this paper, we introduce UniFaceGAN, a novel framework for 3D facial editing, leveraging a unified latent space to facilitate diverse and user-friendly 3D facial manipulation. The key to efficient 3D facial editing lies in establishing a representation space that offers essential facial priors. To achieve this, we propose encoding high-dimensional 3D faces into a compact, disentangled latent space which is learned through conditional 3D GANs guided by text descriptions. With the help of the GAN inversion techniques, UniFaceGAN allows us to edit existing 3D faces, accompanied by a residual editing strategy to mitigate inversion errors efficiently. We demonstrate UniFaceGAN can generate high-quality 3D faces and supports various 3D face editing applications, including CLIP-based stylizations, multiple-point-based drag manipulation, and local blending among multiple faces.**

***Index Terms*—3D Digital Avatar, Multimodal Deep Generative Model, 3D Face Editing**

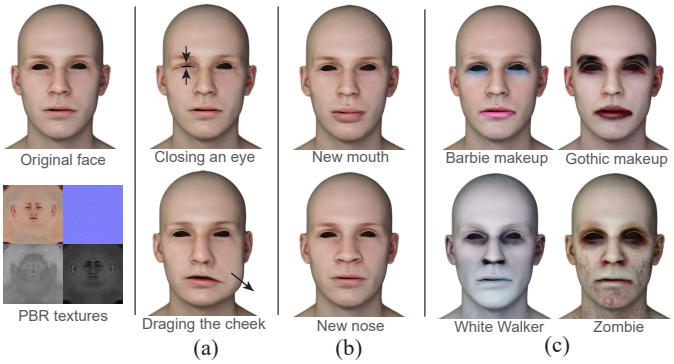

Fig. 1. Editing operations of UniFaceGAN. We can drag points on his face (a), replace his facial organs with another one (b), and put fancy makeups on his face by simple descriptions (c).

I. Introduction

Creating high-quality, customized digital face models is a challenging research problem in computer graphics and computer vision. While early methods [1]–[4] can generate remarkable digital humans, they often require specialized hardware and extensive manual post-processing by skilled artists. Parametric facial models [5]–[7] enabled lightweight face reconstruction from a few photographs. However, these methods cannot produce high-frequency details due to their linear nature. Recent advances in 2D image generation [8]–[12] enable highly various and high-quality face generation. Moreover, implicit 3D representation such as the Neural Radiance Fileds(NeRF) [13] and 3D Gaussian Splatting [14] are introduced to generative models, that enhance 3D avatars generation [15]–[21]. However, implicit representation-based generative models had limitations in relighting and appearance editing since the surface reflectance properties and illumination are not disentangled. Furthermore, implicit representation is not fully compatible with the current CG pipeline.

For explicit 3D faces, the main challenges are from limited 3D facial data. Existing diffusion-based methods [22], [23] leverage the face priors from pre-trained diffusion models to generate diverse 3D faces, while producing face geometry by selection or reconstruction under the constrain of parametric facial models, which restricts their freedom to edit geometry. Moreover, SDS-based texture optimization is time-consuming along with multiple denoising processes. Besides, there are several 3D GAN-based generative model trained on a certain dataset [24]–[28] for fast generation. However, constrained by the dataset, they usually suffer from limited diversity.

To address these limitations in 3D face editing, we introduce UniFaceGAN, a novel framework designed for efficient 3D face editing with a wide range of controls including stylization, point-based drag manipulation, and local blending among multiple assets. Our key idea is to encode high-dimensional face data into a unified latent space, from which various 3D face editing operators can be performed. The main motivation is to establish a GAN space with a high degree of freedom for face editing, distinct from the commonly employed raw 3D space or parametric space. Besides, the CLIP space in pre-trained CLIP [29] shows sufficient 2D face priors, which can be joined with our GAN space to enhance texture editing without needing a large amount of 3D face.

We introduce the following strategies to overcome the challenges in constructing a unified latent space suitable for 3D facial generation and editing: 1) Efficient facial representation: We utilize a neural network-friendly 3D facial representation that efficiently handles the inherent complexity of 3D representation, characterized by high dimensionality and irregularities. 2) Optimization strategies joint with raw 3D space and CLIP space: the raw 3D space allows directly manipulation, which can be used to modify face geometries; the CLIP space can extend optimization space with CLIP prior, ahcieving the synchronously editing of physically-based rendering (PBR)

*Corresponding authors

appearances. 3) Reisdue-based editing strategy: With the GAN inversion technique, we can editing arbitrary 3D faces in the latent space, and minimize the embedding errors by incorporating a residue-based editing strategy.

UniFaceGAN is capable of generating high-quality facial models within 60 milliseconds and supports various 3D face editing applications as shown in Figure 1, for both geometries and PBR textures. In summary, our main contributions are: 1) A 3D face generation framework, which bridges a unified latent space for geometry and PBR materials of faces, enabling high-quality 3D face generation and editing; 2) A stylization method to synchronously manipulate PBR appearances with only a simple text description; 3) A novel way to manipulate face geometries with prior of GANs, which preserves better facial consistency than previous methods.

II. Method

A. Neural-network-friendly 3D facial representation

We present 3D faces, using parameter maps in the UV texture space defined by the fixed-topology geometry. More specifically, for each point x on the face surface, we characterize its facial representation by a surface position $p(x)$, and physically-based material parameters $k_d(x), k_s(x), n(x), r(x)$. Texture-space parameter maps are a natural 2D representation, which is highly compatible with convolutional-based neural networks. A position map $p(x)$ acts similarly to other texture maps but instead stores the 3D position coordinates (x, y, z). This texture-space representation is equivalent to the fixed-topology mesh, meaning that they can be converted back and forth losslessly with sufficient texture resolution.

B. Unified Generator

As shown in Figure 2, our unified generator consists of a unified projector and a multi-branch generator. The unified projector and multi-branch generator work collaboratively. Specifically, the latent code for each generator is generated by a unified projector $\mathcal{P}$, four two-layer MLPs:

$$\mathcal{P}(w) : \mathbb{R}^{|w|} \rightarrow \mathbb{R}^{|w_p|\times|w_{k_d}|\times|w_n|\times|w_s|}, \tag{1}$$

where w is the unified latent code, w_p, w_{k_d}, w_n, and w_s correspond to the position, diffuse, normal, and specular(specular albedo and specular roughness) latent code, respectively. The multi-branch generator consists of four individual generators to reconstruct our 3D facial representation from the latent space:

$$\mathcal{G} = \{\mathcal{G}_p(w_p), \mathcal{G}_{k_d}(w_{k_d}), \mathcal{G}_n(w_n), \mathcal{G}_s(w_s)\}, \tag{2}$$

where $\mathcal{G}_p, \mathcal{G}_{k_d}, \mathcal{G}_n$ generate 3-channel position map, diffuse albedo, and normal map respectively, $\mathcal{G}_s(w_s)$ produces a 2-channel specular map including specular albedo and roughness.

We opt for a multi-branch generator architecture instead of directly using a monolithic generator to generate 11-channel textures. This choice is motivated by two key considerations. Firstly, while there are correlations between different properties, the details tend to vary significantly. The misalignment between different properties can affect both the training stability. Secondly, training a large monolithic network requires much larger memory requirements and computational resources. In contrast, our multi-branch generators can be trained separately and subsequently fine-tuned for a few iterations, resulting in more efficient resource utilization.

C. Residual Editing

In a typical editing scenario, given an initial 3D face $\mathcal{F}_0$, we first employ the inversion algorithm to find its latent representation w_0 and the corresponding inverted 3D face $\mathcal{F}_0^* = \mathcal{G}(w_0)$. Next, certain latent manipulation function $\mathcal{M}$ is applied to modify the latent code, resulting in the edited latent code $w_1 = \mathcal{M}(w_0)$. Finally, the final edited 3D face is generated using the generator $\mathcal{F}_1^* = \mathcal{G}(\mathcal{M}(w_0))$. The differences $\mathcal{F}_1^* - \mathcal{F}_0$ between the input 3D face and the edited result contains two terms: the first term is the actual editing $\Delta = \mathcal{F}_1^* - \mathcal{F}_0^*$ that we intend to achieve and the second term is the inversion errors $\delta = \mathcal{F}_0^* - \mathcal{F}_0$ that we aim to eliminate. Although the inversion error is relatively small, it mainly contains high-frequency details that are crucial for capturing unique characteristics, and thus cannot be ignored.

We introduce a residual-based strategy to mitigate the inversion errors during editing by simply subtracting the inversion error from the edited mesh. Specifically, we compute the final edited mesh as follows:

$$\mathcal{F}_1 = \mathcal{F}_1^* - \delta = \mathcal{F}_0 + \mathcal{G}(\mathcal{M}(w_0)) - \mathcal{G}(w_0). \tag{3}$$

D. Latent Manipulation

1) CLIP-based Stylization: With the pre-trained CLIP model, we can expand the capability to edit different textures by its priors of images and texts, enhancing diverse stylizations. To do so, we adopt the directional CLIP loss [30] operation in CLIP space and introduce a rendering process like ClipFace [31], while we allow changing PBR appearances not just on a single texture. Specifically, a frozen generator G_{frozen} is to generate original faces, and different branches in another trainable generator G_{train} are optimized to minimize the directional CLIP loss (Equation (4)) under random positions of the camera and convert the original face to the text_{tgt} style. We found that the geometry is unstable when optimizing, so we keep $\mathcal{G}_p$ frozen in stylization.

$$\begin{aligned}
&\Delta T = E_T(\text{text}_{\text{tgt}}) - E_T(\text{text}_{\text{tgt}}),\\
&\Delta I = E_I(R(\mathcal{G}_{\text{frozen}}(w), p)) - E_I(R(\mathcal{G}_{\text{train}}(w), p)),\\
&\mathcal{L}_{\text{clip}} = \mathbb{E}_{p,w}(1 - \frac{\Delta T \Delta I}{|\Delta T||\Delta I|}),
\end{aligned} \tag{4}$$

where p is the camera position, R is the rendering process, and $\text{text}_{\text{init}}$ is set as "a face".

2) Geometry manipulation: Initially, a GAN model is trained with high-quality and artifact-free data, establishing a relationship between latent codes and faces. Within a certain latent space range, any latent code will produce a reasonable and artifact-free face. Based on this observation, we can convert a coarse edited face geometry $\mathcal{F}_p^{\text{coarse}}$ with only the

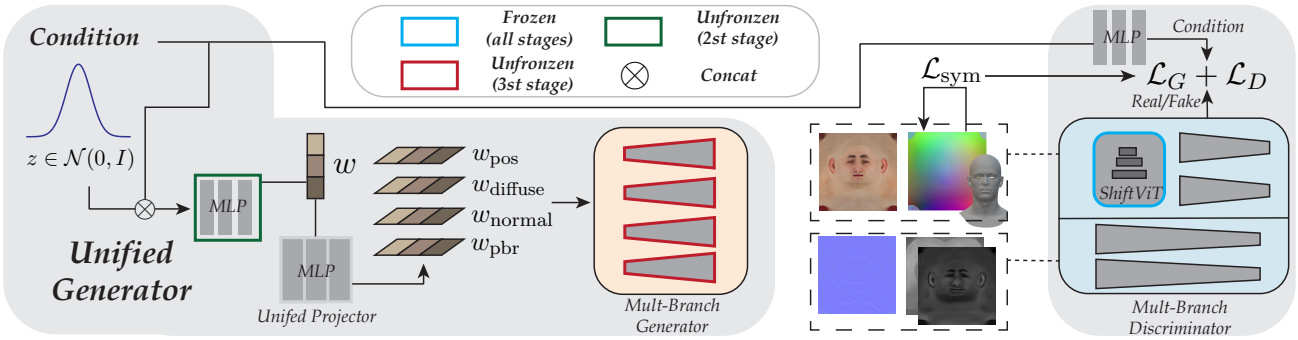

Fig. 2. Architecture of Unified Generator (Section II-B) and Training Pipeline (Section II-E).

dragged points moved, to an artifact-free face by optimizing w_p by

$$\mathcal{L} = \|W \odot (\mathcal{G}_p(w_p) - \mathcal{F}_p^{\text{coarse}})\|, \tag{5}$$

where W satisfies: high weight on dragged points, zero weight around dragged points for adaptive change, and one weight on other regions for keeping identity.

We can also locally replace some regions of faces. Different from dragging, when replacing the region M in source face $\mathcal{F}_p^{\text{src}}$ by target face $\mathcal{F}_p^{\text{tgt}}$, we cannot blend those two faces directly because of difference of heights. Thus, we introduce a slack displacement vector $\epsilon_{\text{offset}} \in \mathbb{R}^3$ to adaptively fix the difference of heights and we can change the loss function as:

$$\mathcal{L} = \|\mathcal{G}_p(w_p) - (1 - M)\mathcal{F}_p^{\text{src}} + M(\mathcal{F}_p^{\text{tgt}} + \epsilon_{\text{offset}})\|. \tag{6}$$

E. Training

1) Discriminators: In our implementation, for the diffuse discriminator $\mathcal{D}_{k_d}$ and position discriminator $\mathcal{D}_p$, we adopt the architecture of ShiftViT [32]. We found that the ShiftViT discriminator more accurately captures the facial features of the diffuse albedo and position map compared to the StyleGAN's discriminator as shown in Table I.

2) Training Loss: The training loss of our conditional generative model consists of four terms:

$$\mathcal{L} = \mathcal{L}_{\mathcal{G}} + \mathcal{L}_{\mathcal{D}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{sym}}, \tag{7}$$

where $\mathcal{L}_{\mathcal{G}}$ is the conditional GAN loss for the generator, $\mathcal{L}_{\mathcal{D}}$ is the conditional GAN loss for the discriminator, $\mathcal{L}_{\text{reg}}$ is the regularization term used in the training of StyleGAN including r1 regularization, and $\mathcal{L}_{\text{sym}}$ is the symmetry regularization term.

We have observed that the position map is more sensitive to small numerical variations. This means that even a small difference can lead to visually noticeable artifacts in the reconstructed mesh. To ensure smooth facial geometry, we introduce an additional symmetry regularization term $\mathcal{L}_{\text{sym}}$, aiming to minimize the error between symmetrical texels in the position map that lie on each side of vertical midlines.

$$\mathcal{L}_{\text{sym}} = \sum_{i,j} (|a_{\text{pos}}^{0,i,j} + a_{\text{flip}}^{0,i,j}| + \sum_{1 \le c \le 2} |a_{\text{pos}}^{c,i,j} - a_{\text{flip}}^{c,i,j}|) \tag{8}$$

where a_{pos} is the position map (symmetrical to the Y-O-Z plane) and a_{flip} is the horizontally fliped position map.

3) Pipeline: We propose a multi-stage training scheme to accelerate the training process: (1) Train each generator and its corresponding discriminator separately at a resolution of 512×512 as warmup and then fine-tune $1,024$ models using $1k$ resolution data. (2) Integrate independent generators for different facial properties into a unified generative model by training the unified projector while keeping other modules frozen. (3) Unlock the mapping MLP, allowing for adaptive improvements to the distribution of the unified latent space. (4) Unlock all trainable weights in the final fine-tuned stage to achieve a more unified and consistent generative model.

III. Experiment

A. Dataset

We collect the face assets of 126 identities from [33], with labels of races, genders, and ages, and manually label them with extra labels like face shape, nose shape, and mouth shape. To improve the diversity of geometry, we construct a facial geometry dataset derived from the CelebA-HQ dataset with a single-image 3D face reconstruction technique and group them with the textures by the above labels.

B. Implementation Details

Our structure of generators follows StyleGAN2 [34] and CLIP is ViT-B/32. We use Adam as our optimizer, batch size 32, and learning rates of 3e-4 and 5e-4 for generators and discriminators using four Nvidia 3090 GPUs. Firstly we train each submodel for 512×512 images in 4 million iterations until they converge and finetune them in 1024×1024 images in 2 million iterations. The three stages of finally unified training are 50, 150, and 650 iterations.

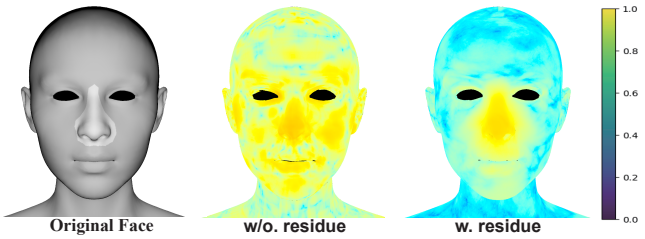

Fig. 3. Comparison of residual editing.

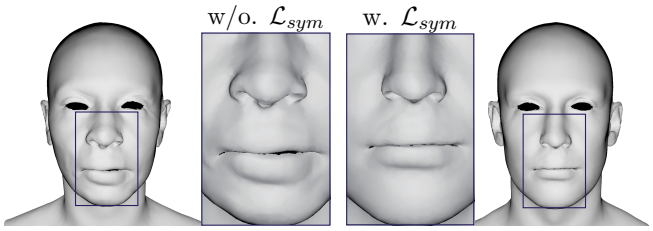

Fig. 4. Comparison of Symmetry Loss.

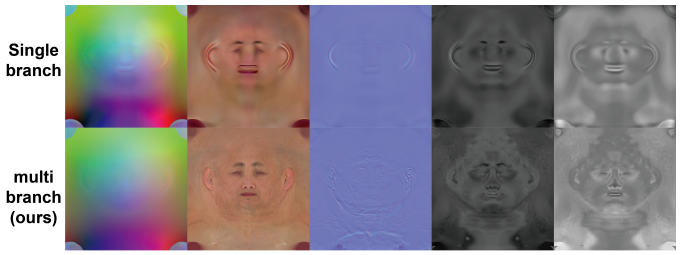

Fig. 5. Comparison of different architecture.

C. Ablation Study

1) Residual Editing: To evaluate the effectiveness of residual editing, we apply a drag operation on the nose and compare the maintenance of unedited areas in the faces. The normalized logarithm of L1 error maps are shown in Figure 3. It is clear that the edited face with the residual editing method maintains more details in the unedited area while allowing partial changes for the adaptation edition.

2) Symmetry Loss: We trained our generator of shapes with and without symmetry loss respectively and compared the faces from those two generators. Figure 4 shows the different results of generated faces based on the use of symmetry loss. It is clear that the symmetry loss benefits neater geometries.

3) Multi-Branch Architecture: Compared with our multi-branch architecture, we double the base channels of the generator and discriminator of StyleGAN so that its parameters are almost the same as our model. As shown in Figure 5, the textures from single-branch architecture have similar structures but deviate from the original distributions. With separate architecture, different properties fit each distribution well.

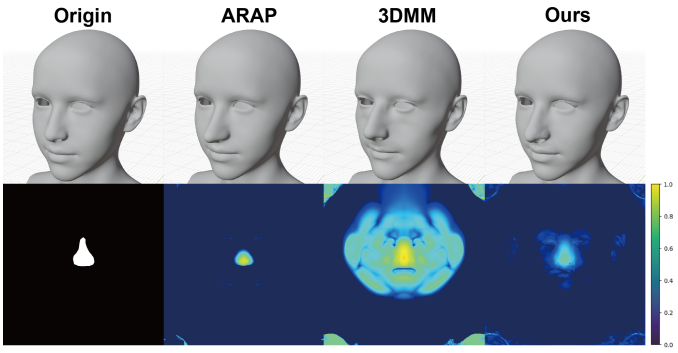

Fig. 6. Comparison of geometry manipulation.

4) ShiftViT-Based Discriminator: Table I digitally shows the effect of different discriminator settings for submodels' training. By using ShiftViT, it keeps lower FIDs for generating diffuse maps and meshes but a higher FID for normal maps. So we introduce ShiftViT in the discriminator of diffuse and mesh while keeping the original discriminator in other assets.

TABLE I
THE COMPARISON OF SHIFTVIT-BASED DISCRIMINATOR

Discriminator setting	FID↓		
	Diffuse	Normal	Mesh
w/o. ShiftViT	74.13743	**29.59193**	11.26426
w. ShiftViT	**36.00264**	44.23839	**3.970599**

D. Comparison

1) Geometry manipulation: We try different methods to manipulate the geometries in Figure 6. ARAP [35] is one of the most usual methods to drag geometries, while works as a general method without priors of faces. Using the same optimization target, we can also optimize 3DMM coefficients instead of GAN's latent code. However, the linearity makes it hard to keep identities. In comparison, the GAN space has better properties for the geometry manipulation task.

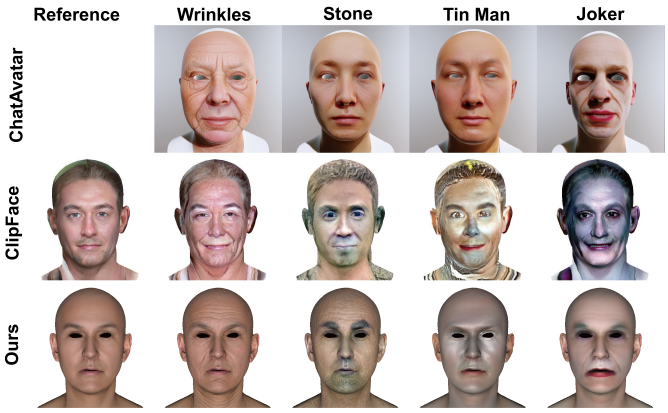

Fig. 7. Comparison of textures stylization.

2) Texture stylization: We compared our stylization method with DreamAvatar [36] and ClipFace [31] in Figure 7. To create wrinkles, we can only optimize the submodel of the normal map, which keeps more on identities. Because of our decoupling of PBR appearances, we can transform a face into another material(stone and metal) more easily. The target style texts are "a face with many wrinkles", "a face made of stone", "a tin man", and "a face that looks like the Joker", respectively.

IV. CONCLUSIONS

UniFaceGAN builds a latent space of StyleGAN for high-quality textures and geometries, succeeding in editing high-quality 3D faces. In summary, our proposed approach addresses the challenges in constructing a unified latent space for 3D facial generation and editing through a multifaceted strategy. Furthermore, residual editing allows us to manipulate in-the-wild faces based on the inversion technique. Collectively, these strategies contribute to an advanced and efficient framework for 3D facial generation and editing, with improved interpretability and fine-grained control.

REFERENCES

[1] P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar, "Acquiring the reflectance field of a human face," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '00. USA: ACM Press/Addison-Wesley Publishing Co., Jul. 2000, pp. 145–156.

[2] W.-C. Ma, T. Hawkins, P. Peers, C.-F. Chabert, M. Weiss, and P. Debevec, "Rapid acquisition of specular and diffuse normal maps from polarized spherical gradient illumination," in *Proceedings of the 18th Eurographics Conference on Rendering Techniques*, ser. EGSR'07. Goslar, DEU: Eurographics Association, Jun. 2007, pp. 183–194.

[3] A. Ghosh, G. Fyffe, B. Tunwattanapong, J. Busch, X. Yu, and P. Debevec, "Multiview face capture using polarized spherical gradient illumination," *ACM Transactions on Graphics*, vol. 30, no. 6, pp. 1–10, Dec. 2011.

[4] G. Fyffe, P. Graham, B. Tunwattanapong, A. Ghosh, and P. Debevec, "Near-instant capture of high-resolution facial geometry and reflectance," *Computer Graphics Forum*, vol. 35, no. 2, pp. 353–363, 2016.

[5] V. Blanz and T. Vetter, "A morphable model for the synthesis of 3d faces," in *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, ser. SIGGRAPH '99. USA: ACM Press/Addison-Wesley Publishing Co., 1999, p. 187–194. [Online]. Available: https://doi.org/10.1145/311535.311556

[6] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou, "Facewarehouse: A 3d facial expression database for visual computing," *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 3, pp. 413–425, Mar. 2014.

[7] P. Paysan, R. Knothe, B. Amberg, S. Romdhani, and T. Vetter, "A 3d face model for pose and illumination invariant face recognition," in *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, Sep. 2009, pp. 296–301.

[8] T. Karras, S. Laine, and T. Aila, "A style-based generator architecture for generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 4401–4410.

[9] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, Oct. 2020.

[10] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851.

[11] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 684–10 695.

[12] K. Preechakul, N. Chatthee, S. Wizadwongsa, and S. Suwajanakorn, "Diffusion autoencoders: Toward a meaningful and decodable representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 10 619–10 629.

[13] 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," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, Dec. 2021.

[14] 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/

[15] E. R. Chan, C. Z. Lin, M. A. Chan, K. Nagano, B. Pan, S. De Mello, O. Gallo, L. J. Guibas, J. Tremblay, S. Khamis, T. Karras, and G. Wetzstein, "Efficient geometry-aware 3d generative adversarial networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 123–16 133.

[16] X. Shen, J. Ma, C. Zhou, and Z. Yang, "Controllable 3d face generation with conditional style code diffusion," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 5, 2024, pp. 4811–4819.

[17] T. Wang, B. Zhang, T. Zhang, S. Gu, J. Bao, T. Baltrusaitis, J. Shen, D. Chen, F. Wen, Q. Chen, and B. Guo, "Rodin: A generative model for sculpting 3d digital avatars using diffusion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4563–4573.

[18] X. Han, Y. Cao, K. Han, X. Zhu, J. Deng, Y.-Z. Song, T. Xiang, and K.-Y. K. Wong, "Headsculpt: Crafting 3d head avatars with text," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[19] X. Huang, R. Shao, Q. Zhang, H. Zhang, Y. Feng, Y. Liu, and Q. Wang, "Humannorm: Learning normal diffusion model for high-quality and realistic 3d human generation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4568–4577.

[20] B. Lei, K. Yu, M. Feng, M. Cui, and X. Xie, "Diffusiongan3d: Boosting text-guided 3d generation and domain adaptation by combining 3d gans and diffusion priors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 10 487–10 497.

[21] H. Liu, X. Wang, Z. Wan, Y. Shen, Y. Song, J. Liao, and Q. Chen, "Headartist: Text-conditioned 3d head generation with self score distillation," in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–12.

[22] L. Zhang, Q. Qiu, H. Lin, Q. Zhang, C. Shi, W. Yang, Y. Shi, S. Yang, L. Xu, and J. Yu, "Dreamface: Progressive generation of animatable 3d faces under text guidance," *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 138:1–138:16, Jul. 2023.

[23] M. Zhou, R. Hyder, Z. Xuan, and G. Qi, "Ultravatar: A realistic animatable 3d avatar diffusion model with authenticity guided textures," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 1238–1248.

[24] R. Slossberg, G. Shamai, and R. Kimmel, "High quality facial surface and texture synthesis via generative adversarial networks," in *Computer Vision – ECCV 2018 Workshops: Munich, Germany, September 8-14, 2018, Proceedings, Part III*. Berlin, Heidelberg: Springer-Verlag, Jan. 2019, pp. 498–513.

[25] B. Gecer, A. Lattas, S. Ploumpis, J. Deng, A. Papaioannou, S. Moschoglou, and S. Zafeiriou, "Synthesizing coupled 3d face modalities by trunk-branch generative adversarial networks," in *Computer Vision – ECCV 2020*, ser. Lecture Notes in Computer Science, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds. Cham: Springer International Publishing, 2020, pp. 415–433.

[26] R. Li, K. Bladin, Y. Zhao, C. Chinara, O. Ingraham, P. Xiang, X. Ren, P. Prasad, B. Kishore, J. Xing, and H. Li, "Learning formation of physically-based face attributes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3410–3419.

[27] M. Wu, H. Zhu, L. Huang, Y. Zhuang, Y. Lu, and X. Cao, "High-fidelity 3d face generation from natural language descriptions," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4521–4530.

[28] A. Gruber, E. Collins, A. Meka, F. Mueller, K. Sarkar, S. Orts-Escolano, L. Prasso, J. Busch, M. Gross, and T. Beeler, "Gantlitz: Ultra high resolution generative model for multi-modal face textures," in *Computer Graphics Forum*, vol. 43, no. 2. Wiley Online Library, 2024, p. e15039.

[29] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 8748–8763.

[30] R. Gal, O. Patashnik, H. Maron, A. H. Bermano, G. Chechik, and D. Cohen-Or, "Stylegan-nada: Clip-guided domain adaptation of image generators," *ACM Transactions on Graphics*, vol. 41, no. 4, pp. 141:1–141:13, Jul. 2022.

[31] S. Aneja, J. Thies, A. Dai, and M. Niessner, "Clipface: Text-guided editing of textured 3d morphable models," in *ACM SIGGRAPH 2023 Conference Proceedings*, ser. SIGGRAPH '23. New York, NY, USA: Association for Computing Machinery, Jul. 2023, pp. 1–11.

[32] G. Wang, Y. Zhao, C. Tang, C. Luo, and W. Zeng, "When shift operation meets vision transformer: An extremely simple alternative to attention mechanism," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 36, no. 2, pp. 2423–2430, Jun. 2022.

[33] 3DScanStore, "3d scan store," https://www.3dscanstore.com/, 2023.

[34] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, "Analyzing and improving the image quality of stylegan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8110–8119.

[35] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling." Citeseer, 2007.

[36] Y. Cao, Y.-P. Cao, K. Han, Y. Shan, and K.-Y. K. Wong, "Dreamavatar: Text-and-shape guided 3d human avatar generation via diffusion models," 2023.