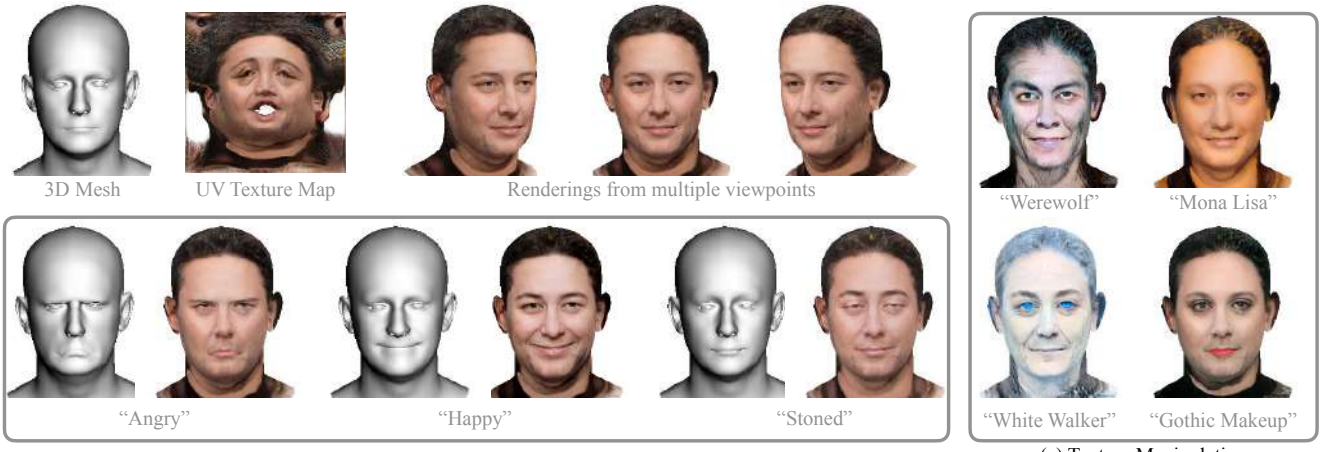
ClipFace: Text-guided Editing of Textured 3D Morphable Models

Shivangi Aneja[1] Justus Thies[2] Angela Dai[1] Matthias Nießner[1]
[1]Technical University of Munich [2]Max Planck Institute for Intelligent Systems



(b) Expression Manipulation (a) Texture Manipulation

Figure 1. ClipFace learns a self-supervised generative model for jointly synthesizing geometry and texture leveraging 3D morphable face models, that can be guided by text prompts. For a given 3D mesh with fixed topology, we can generate arbitrary face textures as UV maps (top). The textured mesh can then be manipulated with text guidance to generate diverse set of textures and geometric expressions in 3D by altering (a) only the UV texture maps for Texture Manipulation and (b) both UV maps and mesh geometry for Expression Manipulation.

Abstract

We propose ClipFace, a novel self-supervised approach for text-guided editing of textured 3D morphable model of faces. Specifically, we employ user-friendly language prompts to enable control of the expressions as well as appearance of 3D faces. We leverage the geometric expressiveness of 3D morphable models, which inherently possess limited controllability and texture expressivity, and develop a self-supervised generative model to jointly synthesize expressive, textured, and articulated faces in 3D. We enable high-quality texture generation for 3D faces by adversarial self-supervised training, guided by differentiable rendering against collections of real RGB images. Controllable editing and manipulation are given by language prompts to adapt texture and expression of the 3D morphable model. To this end, we propose a neural network that predicts both texture and expression latent codes of the morphable model. Our model is trained in a self-supervised fashion by exploiting differentiable rendering and losses based on a pre-trained CLIP model. Once trained, our model jointly predicts face textures in UV-space, along with expression parameters to capture both geometry and texture changes in facial expressions in a single forward pass. We further show the applicability of our method to generate temporally changing textures for a given animation sequence. The source code is publicly available here[1].

1. Introduction

Modeling 3D content is central to many applications in our modern digital age, including asset creation for video games and films, as well as mixed reality. In particular, modeling 3D human face avatars is a fundamental element towards digital expression. However, current content creation processes require extensive time from highly-skilled artists in creating compelling 3D face models.

3D morphable models present a promising approach for modeling animatable avatars, with popular blendshape models used for human faces (e.g., FLAME [33]) or bodies (e.g., SMPL [35]). In particular, they offer a compact, parametric representation to model an object, while maintaining a mesh representation that fits the classical graphics pipelines for editing and animation. Additionally, the

[1] https://shivangi-aneja.github.io/projects/clipface/

shared topology of the representation enables deformation and texture transfer capabilities.

Despite such morphable models' expressive capability in geometric modeling and potential practical applicability towards artist creation pipelines, they remain insufficient for augmenting artist workflows. This is due to limited controllability, as they rely on PCA models, and lack of texture expressiveness, since the models have been built from very limited quantities of 3D-captured textures; both of these aspects are crucial for content creation and visual consumption. We thus address the challenging task of creating a generative model to enable synthesis of expressive, textured, and articulated human faces in 3D.

We propose ClipFace, to enable controllable generation and editing of 3D faces. We leverage the geometric expressiveness of 3D morphable models, and introduce a self-supervised generative model to jointly synthesize textures and adapt expression parameters of the morphable model. To facilitate controllable editing and manipulation, we exploit the power of vision-language models [43] to enable user-friendly generation of diverse textures and expressions in 3D faces. This allows us to specify facial expressions as well as the appearance of the human via text while maintaining a clean 3D mesh representation that can be consumed by standard graphics applications. Such text-based editing enables intuitive control over the content creation process.

Our generative model is trained in a self-supervised fashion, leveraging the availability of large-scale high-quality face image datasets with differentiable rendering to produce a powerful texture generator that can be controlled along with the morphable model geometry by text prompts. Based on our high-quality texture generator, we learn a neural network that can edit the texture latent code as well as the expression parameters of the 3D morphable model with text prompt supervision and losses based on CLIP. Our approach further enables generating temporally varying textures for a given driving expression sequence.

To summarize, the main contributions of this paper are:

- We propose a novel approach to controllable editing of textured, parametric 3D morphable models through user-friendly text prompts, by exploiting CLIP-based supervision to jointly synthesize texture and expressions of a 3D face model.
- Our controllable 3D face model is supported by our high-quality texture generator, trained in a self-supervised fashion on 2D images only.
- Our approach additionally enables generating temporally varying textures of an animated 3D face model from a driving video sequence.

2. Related Work

Texture Generation There is a large corpus of research works in the field of generative models for UV textures [14–16, 30–32, 36, 37, 47]. These methods achieve impressive results; however, the majority is fully supervised in nature, requiring ground truth textures, which in turn necessitate collection in a controlled capture setting. Learning self-supervised texture generation is much more challenging, and only a handful of methods exist. For instance, Marriott *et al*. [37] were among the first to leverage Progressive GANs [21] and 3D Morphable Models [5] to generate textures for facial recognition; however, the textures generated are still relatively low resolution and are unable to produce high-frequency details. Textures generated by Slossberg *et al*. [47] made a significant improvement in quality by using pretrained StyleGAN [24] and StyleRig [48]. The closest inspiration to our texture generator is StyleUV [32] which also operates on a mesh. Both methods achieve stunning results but currently do not take the head and ears into account. In our work, we propose a generative model to synthesize UV textures for the full-head topology; however, our main focus lies in enabling text-guided editing and control.

Semantic Manipulation of Facial Attributes Manipulation of face images has also seen significant study following the success of StyleGAN2 [24] image generation. In particular, its disentangled latent space facilitates both texture editing as well as enables a level of control over pose and expressions of generated images. Several methods [2, 10, 17, 28, 34, 48, 49] have made significant progress to induce controllability to images by embedding 3D priors via conditioning the StyleGAN on known facial attributes extracted from synthetic face renderings. However, these methods operate in the 2D image domain, and although they achieve high-quality results on a per-frame basis, consistent and coherent rendering from varying poses and expressions remains challenging. Motivated by such impressive 2D generators, we propose to lift them to 3D and directly operate in UV space of a 3D mesh, thus producing temporally-consistent results when rendering animation sequences.

Text-Guided Image Manipulation The recent progress in 2D language models has opened up significant opportunities for text-guided image manipulation [1, 3, 4, 7, 8, 44–46]. For instance, the contrastive language-image pre-training (CLIP) [43] model has been used for text-guided editing for a variety of applications [6, 13, 19, 27, 38, 40, 42, 50]. StyleClip [40] presented seminal advances in stylizing human face images by leveraging the expressive power of CLIP in combination with the generative power of StyleGAN to produce unique manipulations for faces. This was followed by StyleGAN-Nada [13], which enables adapting image gen-

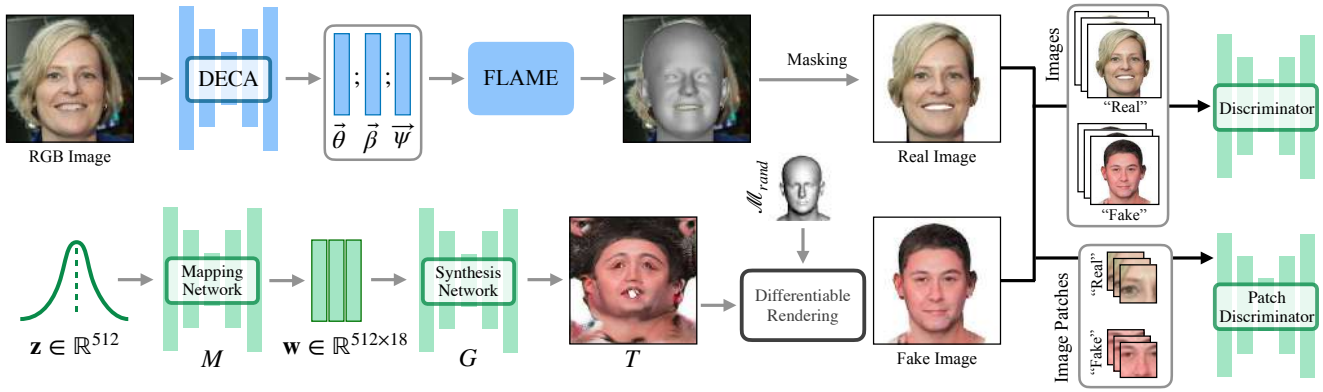


Figure 2. **Texture Generation:** We learn self-supervised texture generation from collections of 2D images. An RGB image is encoded by pretrained encoder DECA [12] to extract shape $\vec{\boldsymbol{\beta}}$, pose $\vec{\boldsymbol{\theta}}$ and expression $\vec{\boldsymbol{\psi}}$ coefficents in FLAME's latent space, which are decoded by FLAME [33] to deformed mesh vertices. The background and mouth interior are then masked out, generating the 'Real Image' for our adversarial formulation. In parallel, latent code $z \in \mathbb{R}^{512}$ is sampled from $\mathcal{N}(0, I)$ and input to mapping network M to generate intermediate latent $\mathbf{w} \in \mathbb{R}^{512\times18}$, which is used by synthesis network G to generate the UV texture image. The predicted texture is differentiably rendered on a randomly deformed FLAME mesh to generate the 'Fake Image.' Two discriminators interpret the generated and masked real image, at full resolution and at patch size 64×64. Frozen models are denoted in blue, and learnable networks in green.

eration to a remarkable diversity of styles from various domains, without requiring image examples from those domains. However, these manipulations are designed for the image space, and are not 3D consistent.

Text-Guided 3D Manipulation Following the success of text-guided image manipulation, recent works have adopted powerful vision-language models to enable text guidance for 3D object manipulation. Text2Mesh [38] was one of the first pioneering methods to leverage a pre-trained 2D CLIP model as guidance to generate language-conditioned 3D mesh textures and geometric offsets. Here, edits are realized as part of a test-time optimization that aims to solve for the texture and mesh offsets in a neural field representation, such that their re-renderings minimizes a 2D CLIP loss from different viewpoints. Similar to Text2Mesh, CLIP-Mesh [25] produces textured meshes by jointly estimating texture and deformation of a template mesh, based on text inputs using CLIP. Recently, Canfes *et al*. [6] adapt TB-GAN [14], an expression-conditioned generative model to produce UV-texture maps, with a CLIP loss to produce facial expressions in 3D, although the quality of textures and expressions is limited, due to relying on 3D scans for TB-GAN training. Our method is inspired by these lines of research; however, our focus lies on leveraging the parametric representations of 3D morphable face models with high-fidelity textures, which can enable content creation for direct use in many applications such as games or movies.

3. Method

ClipFace targets text-guided synthesis of textured 3D face models. It consists of two fundamental components: (i) an expressive generative texture space for facial appearances (Sec 3.1), and (ii) a text-guided prediction of the latent code of the texture and the expression parameters of the underlying statistical morphable model (Sec 3.2). In the following, we will detail these contributions and further demonstrate how they enable producing temporally changing expressions for a given animation sequence Sec. 3.3.

3.1. Generative Synthesis of Face Appearance

Since there do not exist any large-scale datasets for UV textures, we propose a self-supervised method to learn the appearance manifold of human faces, as depicted in Figure 2. Rather than learn texturing from ground truth UV textures, we instead leverage large-scale RGB image datasets of faces, which we use in an adversarial formulation through differentiable rendering. For our experiments, we use the FFHQ dataset [23] which consists of 70,000 diverse, high-quality images. As we focus on the texture of the human head, we remove images that contain headwear (caps, scarfs, etc.) and eyewear (sunglasses and spectacles) using face parsing [51], resulting in a subset of 45,000 images. Based on this data, we train a StyleGAN [22] generator to produce UV textures that when rendered on top of the FLAME mesh [33] results in realistic imagery.

More specifically, we use the FLAME model as our shape prior to produce different geometric shapes and facial expressions. It can be defined as:

$$\mathcal{F}(\vec{\boldsymbol{\beta}}, \vec{\boldsymbol{\theta}}, \vec{\boldsymbol{\psi}}) : \mathbb{R}^{|\vec{\boldsymbol{\beta}}|\times|\vec{\boldsymbol{\theta}}|\times|\vec{\boldsymbol{\psi}}|} \rightarrow \mathbb{R}^{3N}, \tag{1}$$

where $\vec{\boldsymbol{\beta}} \in \mathbb{R}^{100}$ are the shape parameters, $\vec{\boldsymbol{\theta}} \in \mathbb{R}^{9}$ the yaw pose, and $\vec{\boldsymbol{\psi}} \in \mathbb{R}^{50}$ the linear expression coefficients.

To recover the distribution of face shapes and expressions from the training dataset, we employ DECA [12], a

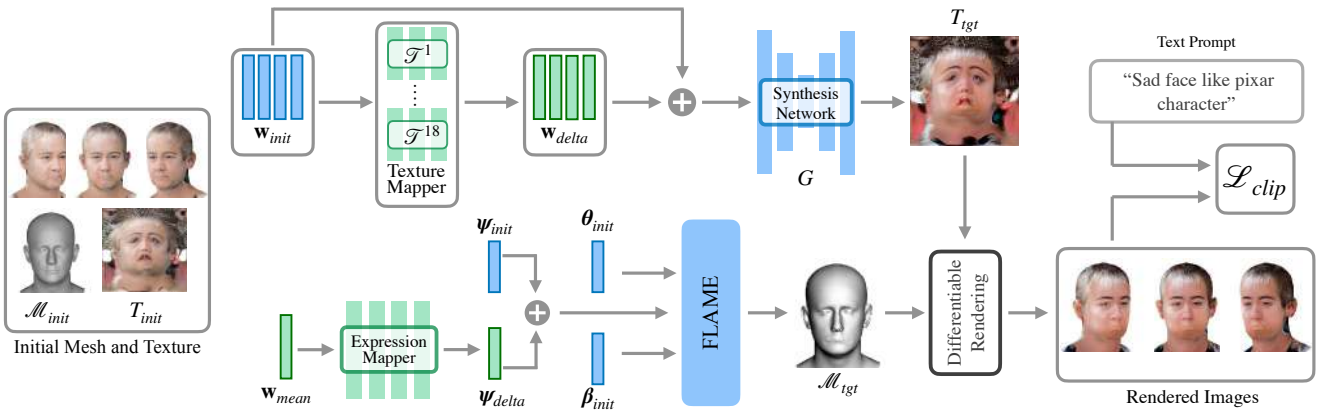


Figure 3. **Text-guided Synthesis of Textured 3D Face Models:** From a given textured mesh with texture code $\mathbf{w}_{\text{init}}$, we synthesize various styles by adapting both texture and expression to an input text prompt. $\mathbf{w}_{\text{init}}$ is input to the texture mappers $\mathcal{T} = [\mathcal{T}^1, ..., \mathcal{T}^{18}]$ to obtain texture offsets $\mathbf{w}_{\text{delta}} \in \mathbb{R}^{512\times18}$ for 18 different levels of $\mathbf{w}_{\text{init}}$. The expression mapper $\mathcal{E}$ takes mean latent code $\mathbf{w}_{\text{mean}} = \|\mathbf{w}_{\text{init}}\|_2$ as input, and predicts expression offset ψ_{delta} to obtain deformed mesh geometry $\mathcal{M}_{\text{tgt}}$. The generated UV map T_{tgt} and deformed mesh $\mathcal{M}_{\text{tgt}}$ are differentiably rendered to generate styles that fit the text prompt.

pretrained encoder that takes an RGB image as input and outputs the corresponding FLAME parameters $\vec{\beta}, \vec{\theta}, \vec{\psi}$, including orthographic camera parameters $\mathbf{c}$. We use the recovered parameters to remove the backgrounds from the original images, and only keep the image region that is covered by the corresponding face model. Using this distribution of face geometries and camera parameters $\mathcal{D} \sim [\vec{\beta}, \vec{\theta}, \vec{\psi}, \mathbf{c}]$, along with the masked real samples, we train the StyleGAN network using differentiable rendering [29]. We sample a latent code $\boldsymbol{z} \in \mathbb{R}^{512}$ from Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I})$ to generate the intermediate latent code $\mathbf{w} \in \mathbb{R}^{512\times18}$ using the a mapping network M: $\mathbf{w} = M(\mathbf{z})$. This latent code $\mathbf{w}$ is passed to the synthesis network G to generate UV texture map $T \in \mathbb{R}^{512\times512\times3}$: $T = G(\mathbf{w})$. This predicted texture T is then rendered on a randomly sampled deformed mesh from our discrete distribution of face geometries $\mathcal{D}$. We use an image resolution of 512×512.

Both the generated image and masked real image are then passed to the discriminator during training. To generate high-fidelity details in the UV maps, we use a patch discriminator alongside a full-image discriminator to critique the generator. We apply image augmentations (e.g., color jitter, image flipping, hue/saturation changes) to both full-image and image patches before feeding them to the discriminator. The patch size is set to 64×64 for all of our experiments. Note that the patch discriminator is critical to producing high-frequency texture details; see Section 4.

3.2. Text-guided Synthesis of Textured 3D Models

For a given textured mesh with texture code $\mathbf{w}_{\text{init}} = \{\mathbf{w}^1_{\text{init}}, \mathbf{w}^2_{\text{init}}, ...\mathbf{w}^{18}_{\text{init}}\} \in \mathbb{R}^{512\times18}$ in neutral pose $\boldsymbol{\theta}_{\text{init}}$ and neutral expression $\boldsymbol{\psi}_{\text{init}}$, our goal is to learn optimal offsets $\mathbf{w}_{\text{delta}}, \boldsymbol{\psi}_{\text{delta}}$ for texture and expression respectively defined through text prompts. As a source of supervision, we use a pretrained CLIP [43] model due to its high expressiveness, and formulate the offsets as:

$$\mathbf{w}^*_{\text{delta}}, \boldsymbol{\psi}^*_{\text{delta}} = \arg\min_{\mathbf{w}_{\text{delta}}, \boldsymbol{\psi}_{\text{delta}}} \mathcal{L}_{\text{total}}, \tag{2}$$

where $\mathcal{L}_{\text{total}}$ formulates CLIP guidance and expression regularization, as defined in Eq. 9.

In order to optimize this loss, we learn a texture mapper $\mathcal{T} = [\mathcal{T}^1, ..., \mathcal{T}^{18}]$ and an expression mapper $\mathcal{E}$. The texture mapper predicts the latent variable offsets across the different levels $\{1, 2, ...18\}$ of the StyleGAN generator:

$$\mathbf{w}_{\text{delta}} = \begin{cases} \mathcal{T}^1(\mathbf{w}^1_{\text{init}}) \\ \mathcal{T}^2(\mathbf{w}^2_{\text{init}}) \\ \vdots \\ \mathcal{T}^{18}(\mathbf{w}^{18}_{\text{init}}). \end{cases} \tag{3}$$

The expression mapper $\mathcal{E}$ learns the expression offsets and takes as input $\mathbf{w}_{\text{mean}} \in \mathbb{R}^{512}$, the mean of 18-different levels of the latent space, and outputs the expression offsets $\boldsymbol{\psi}_{\text{delta}}$:

$$\boldsymbol{\psi}_{\text{delta}} = \mathcal{E}(\mathbf{w}_{\text{mean}}). \tag{4}$$

We use a 4-layer MLP architecture with ReLU activations for the mappers. The method is shown in Figure 3.

Naively using a CLIP loss as in StyleClip [40] to train the mappers tends to result in unwanted identity and/or illumination changes in texture. Thus, we draw inspiration from Gal *et al.* [13], and leverage the CLIP-space direction between the initial style and the to-be-performed manipulation in order to perform consistent and identity-preserving manipulation. We compute the 'text-delta' direction $\Delta\mathbf{t}$ in CLIP-space between the initial text prompt $\mathbf{t}_{\text{init}}$ and the tar-

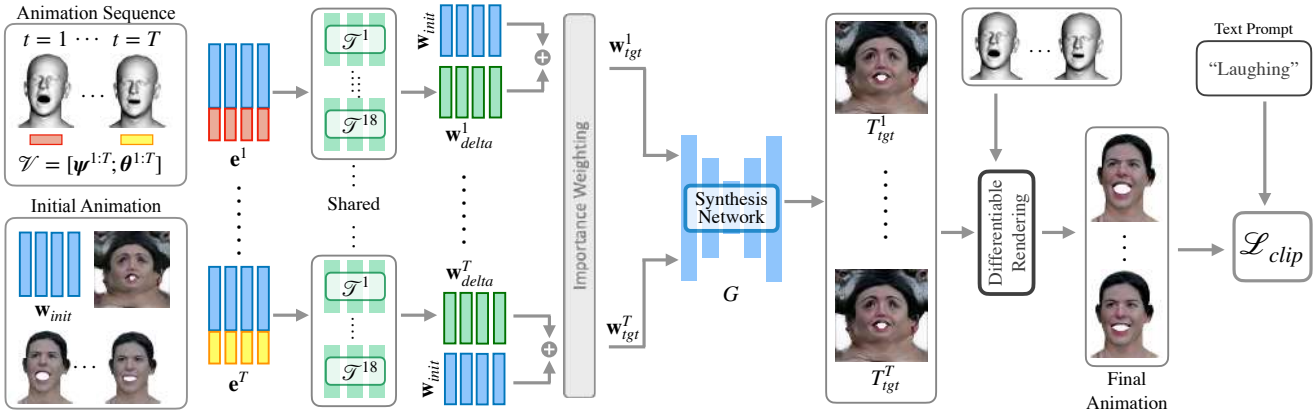


Figure 4. **Texture Manipulation for Animation Sequences:** Given a video sequence $\mathcal{V} = [\psi^{1:T}; \theta^{1:T}]$ of T frames, an initial texture code $\mathbf{w}_{\text{init}}$, and a text prompt, we synthesize a 3D textured animation to match the text. We concatenate $\mathcal{V}$ to $\mathbf{w}_{\text{init}}$ across different timestamps to obtain $\mathbf{e}^t$, which is input to the time-shared texture mappers $\mathcal{T}$ to obtain time-dependent textures offsets $\mathbf{w}^{1:T}_{\text{delta}}$ for all frames. The new texture codes $\mathbf{w}^{1:T}_{\text{tgt}}$ generated using importance weighting, are then passed to our texture generator G to obtain time-dependent UV textures $T^{1:T}_{\text{tgt}}$, which are then differentiably rendered to generate the final animation, guided by the CLIP loss across all frames.

get text prompt $\mathbf{t}_{\text{tgt}}$, indicating which attributes from the initial style should be changed:

$$\Delta\mathbf{t} = E_T(\mathbf{t}_{\text{tgt}}) - E_T(\mathbf{t}_{\text{init}}). \tag{5}$$

Guided via the CLIP-space image direction between the initial image $\mathbf{i}_{\text{init}}$ and the target image $\mathbf{i}_{\text{tgt}}$ generated using our textured model, we train the mapping networks to predict style, guided by a given text prompt:

$$\Delta\mathbf{i} = E_I(\mathbf{i}_{\text{tgt}}) - E_I(\mathbf{i}_{\text{init}}), \tag{6}$$

where $\mathbf{i}_{\text{init}}$ is the image rendered with initial parameters $(\mathbf{w}_{\text{init}}, \boldsymbol{\beta}_{\text{init}}, \boldsymbol{\theta}_{\text{init}}, \boldsymbol{\psi}_{\text{init}})$ and $\mathbf{i}_{\text{tgt}}$ the image with the target parameters $(\mathbf{w}_{\text{tgt}}, \boldsymbol{\beta}_{\text{tgt}}, \boldsymbol{\theta}_{\text{tgt}}, \boldsymbol{\psi}_{\text{tgt}})$. Note that we do not alter the pose $\boldsymbol{\theta}$ and shape code $\boldsymbol{\beta}$ of the FLAME model. The CLIP loss $\mathcal{L}_{\text{clip}}$ is then computed as:

$$\mathcal{L}_{\text{clip}} = 1 - \frac{\Delta\mathbf{i}.\Delta\mathbf{t}}{|\Delta\mathbf{i}||\Delta\mathbf{t}|}. \tag{7}$$

In order to prevent the mesh from taking unrealistic expressions, we further regularize the expressions using the Mahalanobis prior as:

$$\mathcal{L}_{\text{reg}} = \boldsymbol{\psi}^T \Sigma_{\psi}^{-1} \boldsymbol{\psi}, \tag{8}$$

where Σ_{ψ}^{-1} is the diagonal expression covariance matrix of FLAME model.

The full training loss can then be written as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{clip}} + \lambda_{\text{reg}}\mathcal{L}_{\text{reg}}. \tag{9}$$

Note that we can also only alter the texture without changing expressions by keeping the expression mapper frozen and not fine-tuning it.

3.3. Texture Manipulation for Video Sequences

Given an expression video sequence, we propose a novel technique to manipulate the textures for every frame of the video guided by a CLIP loss (see Figure 4). That is, for a given animation sequence $\mathcal{V} = [\boldsymbol{\theta}^{1:T}; \boldsymbol{\psi}^{1:T}]$ of T frames, with expression codes $\boldsymbol{\psi}^{1:T} = [\boldsymbol{\psi}^1, \boldsymbol{\psi}^2, ...\boldsymbol{\psi}^T]$, pose codes $\boldsymbol{\theta}^{1:T} = [\boldsymbol{\theta}^1, \boldsymbol{\theta}^2, ...\boldsymbol{\theta}^T]$, and a given texture code $\mathbf{w}_{\text{init}}$, we use a multi-layer perception as our texture mapper $\mathcal{T} = [\mathcal{T}^1, ..., \mathcal{T}^{18}]$ to generate time-dependent texture offsets $\mathbf{w}^{1:T}_{\text{delta}}$ for different levels of the texture latent space. This mapper receives as input $\mathbf{e}^{1:T}$, the concatenation of the initial texture code $\mathbf{w}_{\text{init}}$ with the time-dependent expression and pose code $[\boldsymbol{\psi}^t; \boldsymbol{\theta}^t]$. Mathematically, we have:

$$\mathbf{e}^{1:T} = [\mathbf{e}^1, ...\mathbf{e}^T] \tag{10}$$

$$\mathbf{e}^t = [\mathbf{w}_{\text{init}}; \boldsymbol{\psi}^t; \boldsymbol{\theta}^t], \tag{11}$$

where $\boldsymbol{\psi}^t$ and $\boldsymbol{\theta}^t$ refer to the expression and pose code at timestamp t extracted from sequence $\mathcal{V}$. Next, we pass $\mathbf{e}^{1:T}$ to the time-shared texture mapper $\mathcal{T}$ to obtain texture offsets $\mathbf{w}^{1:T}_{\text{delta}}$. To ensure a coherent animation and smooth transition across frames, we weight the predicted offsets $\mathbf{w}^{1:T}_{\text{delta}}$ using importance weights $\mathcal{I} = [i_1, ...i_T]$ extracted from video sequence $\mathcal{V}$, before adding them to $\mathbf{w}_{\text{init}}$:

$$\mathbf{w}^t_{\text{tgt}} = \mathbf{w}_{\text{init}} + i_t \cdot \mathbf{w}^t_{\text{delta}}. \tag{12}$$

We compute importance weights by measuring the deviation between the neutral shape $[\boldsymbol{\theta}_{\text{neutral}}; \boldsymbol{\psi}_{\text{neutral}}]$ and per-frame face shape $[\boldsymbol{\theta}^t; \boldsymbol{\psi}^t]$, with by min-max normalization:

$$i_t = \frac{\delta^t - \min(\boldsymbol{\delta}^{1:T})}{\max(\boldsymbol{\delta}^{1:T}) - \min(\boldsymbol{\delta}^{1:T})}, \tag{13}$$

with $\delta^t = \left|\left|[\boldsymbol{\theta}_{\text{neutral}}; \boldsymbol{\psi}_{\text{neutral}}] - [\boldsymbol{\theta}^t; \boldsymbol{\psi}^t]\right|\right|_2$. The importance weighting ensures that key frames with strong expressions

are emphasized. The predicted target latent codes $\mathbf{w}_{\text{tgt}}^{1:T}$ are then used to generate the UV maps $T_{\text{tgt}}^{1:T}$, which are differentiably rendered onto the given animation sequence $\mathcal{V}$.

To train texture mapper $\mathcal{T}$, we minimize Eq. 7 for the given text prompt $\mathbf{t}_{\text{tgt}}$ and the rendered frames $\mathbf{i}_{\text{tgt}}^{t}$ aggregated over T timesteps for all the frames from the video.

$$\mathbf{w}_{\text{delta}}^{1:T} = \arg\min_{\mathbf{w}_{\text{delta}}^{1:T}} \sum_{t=1}^{T} \mathcal{L}_{\text{clip}}(\mathbf{t}_{\text{tgt}}, \mathbf{i}_{\text{tgt}}^{t}). \tag{14}$$

4. Results

We evaluate ClipFace on the tasks of texture generation, text-guided synthesis of textured 3D face models, and text-guided manipulation of animation sequences. For texture generation, we evaluate on standard GAN metrics FID and KID. For text-guided manipulation, we evaluate perceptual quality using KID in addition to CLIP score, which is evaluated as the cosine similarity to the text prompt using pre-trained CLIP models. We use two different CLIP variants, 'ViT-B/16' and 'ViT-L/14', each on 224×224 pixels as input. We report average scores for these pre-trained variants.

Implementation Details: For our texture generator, we produce 512×512 texture maps. We use an Adam optimizer with a learning rate of 2e-3, batch size 8, gradient penalty 10, and path length regularization 2 for all our experiments. We use a learning rate of 0.005 and 0.0001 for the expression and texture mappers, also using Adam. For differentiable rendering, we use NvDiffrast [29]. For the patch discriminator, we use a patch size of 64×64. We train for 300,000 iterations until convergence. For the text-guided manipulation experiments, we use the same model architecture for expression and texture mappers, a 4-layer MLP with ReLU activations. For CLIP supervision, we use the pretrained 'ViT-B/32' variant. For text manipulation tasks, we train for 20,000 iterations.

Texture Generation We evaluate the quality of our generated textures and compare with existing unsupervised texture generation methods in Table 1 and Figure 5. ClipFace outperforms other baselines in perceptual quality. Although Slossberg *et al*. [47] can obtain good textures for the interior face region, it does not synthesize head and ears.

Method	FID ↓	KID ↓
FlameTex [18]	76.627	0.063
Slossberg *et al*. [47]	32.794	0.021
Ours (w/o Patch)	14.622	0.010
Ours (w/ Patch)	**8.859**	**0.003**

Table 1. Quantitative evaluation of texture quality. Our approach significantly outperforms baselines in both FID and KID scores.

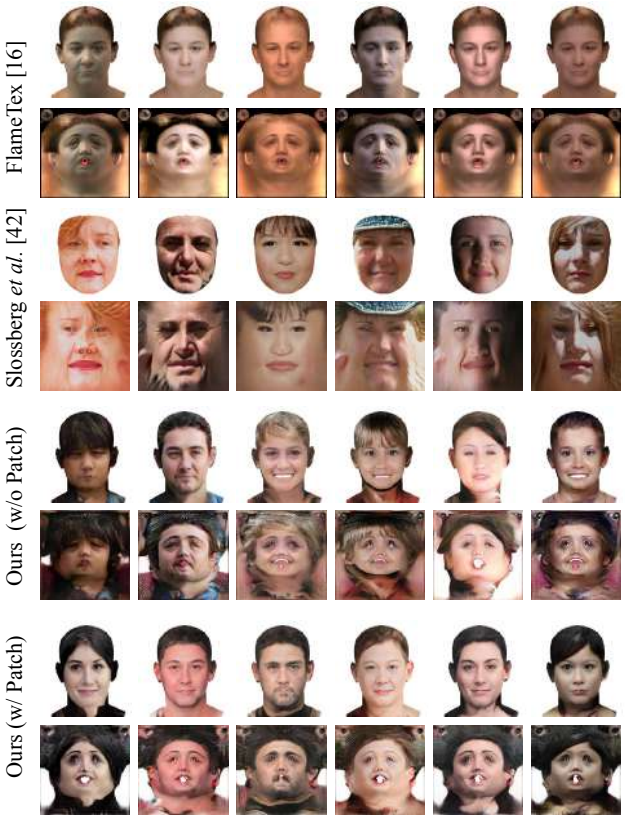


Figure 5. Comparison of texturing quality. Our approach is able to synthesize diverse texture styles ranging across different skin colors, ethnicities, and demographics.

Texture & Expression Manipulation We compare with CLIP-based texturing techniques for texture manipulation in Figure 6 and Table 2. Note that for comparisons with Text2Mesh [38], we follow the authors' suggestion to first perform remeshing to increase vertices from 5023 to 60,000 before optimization. Our approach generates consistently high-quality textures for various prompts, in comparison to baselines. In particular, our texture generator enables high-quality editing even in small face regions (e.g., lips and eyes). Note that Text2Mesh yields a high CLIP score, while producing semantically implausible results, as the specified text prompts highly match rendered colors irrespective of

Method	KID ↓	CLIP Score ↑
Latent3d [6]	0.221	0.227 ±0.041
FlameTex [18]	0.014	0.235 ±0.053
ClipMatrix [20]	0.138	0.243 ±0.049
Text2Mesh [38]	0.146	**0.264** ±0.044
Ours	**0.004**	0.251 ±0.059

Table 2. Evaluation of text manipulation. ClipFace effectively matches text prompts while maintaining high perceptual fidelity.

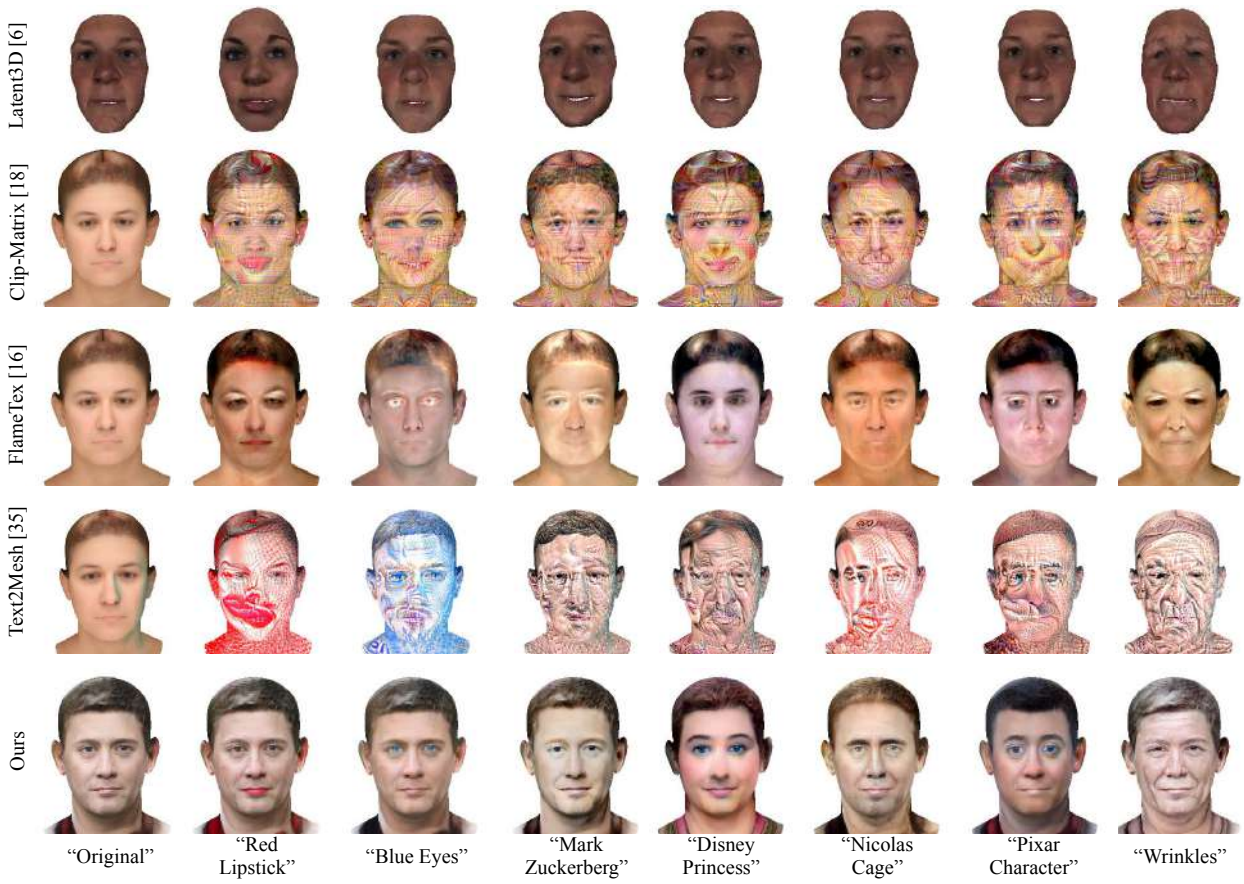


Figure 6. Qualitative comparison for texture manipulation: We compare our method against several 3D texturing methods. Our method obtains consistently high-quality textures, even capable deftly adapting identity when guided by text prompt.

the global face context (i.e., which region should be edited). In contrast, our method also generates high-quality face texture, evident in the perceptual KID metric.

We show additional ClipFace texturing results on a wide variety of prompts, including on fictional characters, in Figure 7, demonstrating our expressive power. Furthermore, we show results for expression manipulation in Figure 9. ClipFace faithfully deforms face geometry and texture to match a variety of text prompts, where expression regularization maintains plausible geometry and directional loss enables balanced adaptation of geometry and texture. We refer to the supplemental for more visuals.

Texture Manipulation for Video Sequences Finally, we show results for texture manipulation for given animation sequences in Fig. 8. ClipFace can produce more expressive animation compared to a constant texture that looks monotonic. We show results for only 3 frames; however, we refer readers to the supplemental video for more detailed results.

Limitations Although ClipFace can generate high-quality textures and expressions, it still has various limitations. For instance, we do not handle accessories like jewelry, headwear, or eyewear, due to our use of the FLAME [33] model, which does not capture accessories or complex hair. We believe that this could be further improved by augmenting the parametric 3D model with artist-designed assets for 3D hair, headwear, or eyewear. Additionally, we do not explicitly consider semantic parts, which could help to provide even finer-grained localization of manipulations and facial detail.

5. Conclusion

In this paper, we have introduced ClipFace, a novel approach to enable text-guided editing of textured 3D morphable face models. We jointly synthesize high-quality textures and adapt geometry based on the expressions of the morphable model, in a self-supervised fashion. This enables compelling 3D face generation across a variety of textures, expressions, and styles, based on user-friendly text prompts. We further demonstrate the ability of ClipFace

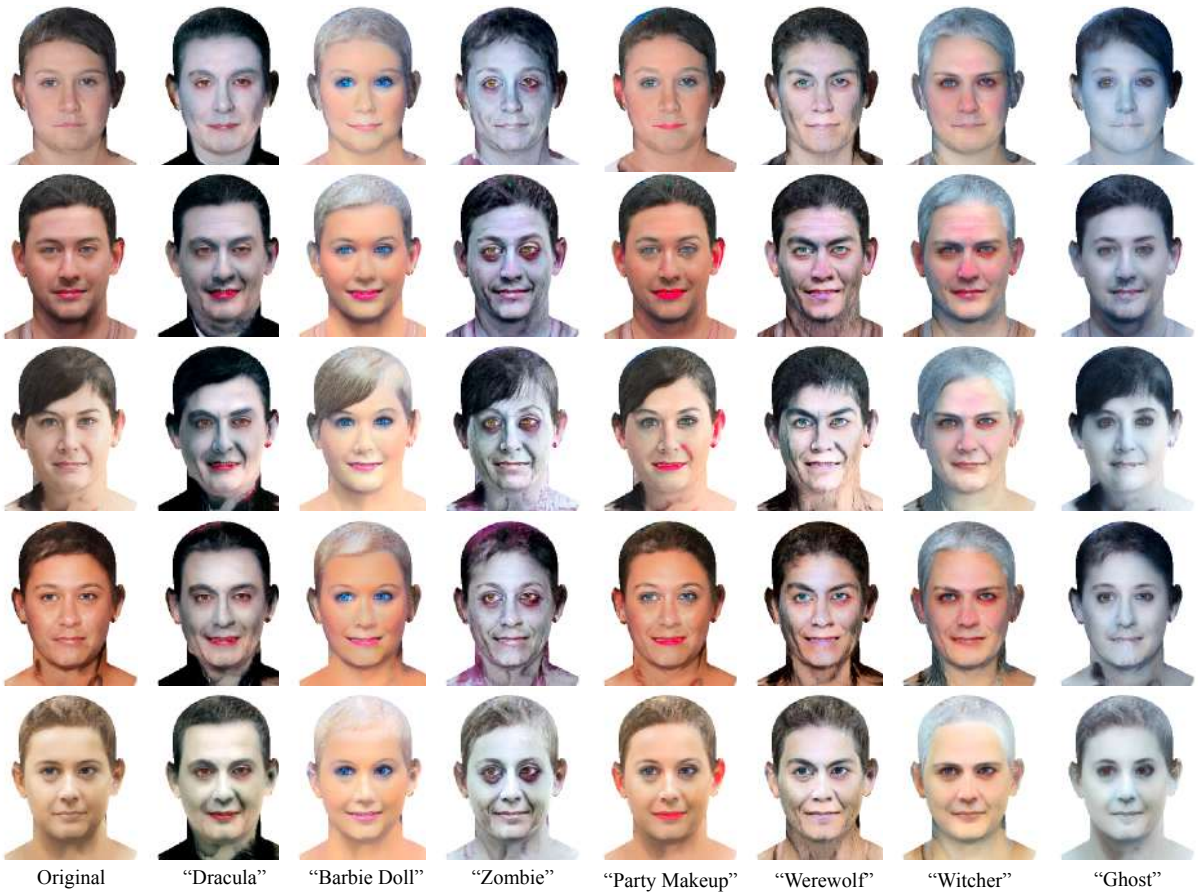


Figure 7. Texture manipulations. 'Original' (left) shows the input image, followed by ClipFace-generated textures for the text prompts.

to synthesize of animation sequences, driven by a guiding video sequence. We believe this is an important first step towards enabling controllable, realistic texture and expression modeling for 3D face models, dovetailing with conventional graphics pipelines, which will enable many new possibilities for content creation and digital avatars.

Acknowledgments

This work was supported by the ERC Starting Grant Scan2CAD (804724), the Bavarian State Ministry of Science and the Arts and coordinated by the Bavarian Research Institute for Digital Transformation (bidt), the German Research Foundation (DFG) Grant "Making Machine Learning on Static and Dynamic 3D Data Practical", the German Research Foundation (DFG) Research Unit "Learning and Simulation in Visual Computing" and Sony Semiconductor Solutions Corporation. We would like to thank Yawar Siddiqui for help with implementation of differential rendering codebase, Artem Sevastopolsky and Guy Gafni for helpful discussions, and Alexey Bokhovkin, Chandan Yeshwanth, and Yueh-Cheng Liu for help during internal review.

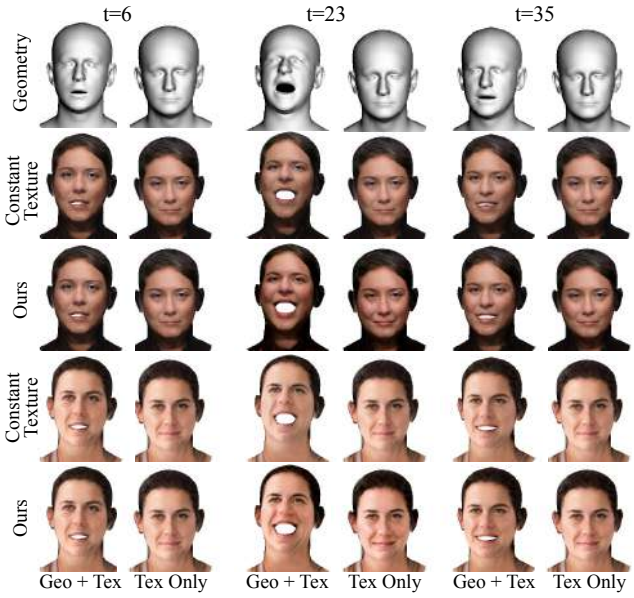


Figure 8. Expression manipulation that generates video sequences. Geo + Tex shows our textures overlaid on the animated mesh, and Tex Only shows our texture in the neutral pose. Our ability to manipulate texture enables more compelling animation, particularly in articulated expressions (e.g., t=23).

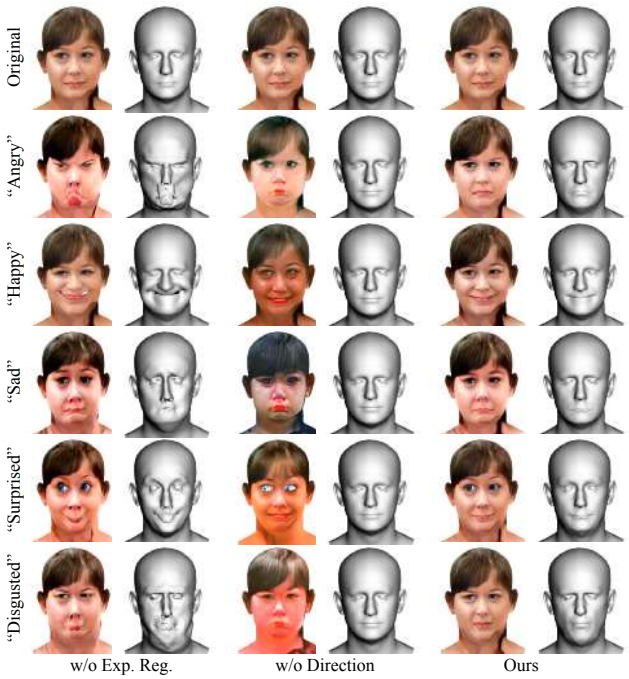


Figure 9. ClipFace generates a large variety of expressions, faithfully deforming the mesh geometry and manipulating texture for more expressiveness. Our expression regularization is important to produce realistic geometric expressions, and directional loss for balanced manipulation of both texture and geometry.

Appendix

We provide additional ablation studies and results in Section 6, network architecture and training details in Section 7, and further discussion of baseline method experimental setup in Section 8.

6. Additional Results

We provide additional results for texture and expression manipulation, as well as an additional comparison to texturing baselines. For results related to video animation, we refer readers to the supplementary video.

Additional Baseline Comparisons: We compare our method with texturing baselines for additional text prompts, and show results in Figure 10. Our method outperforms these baselines and achieves high quality manipulations.

Expression Manipulation: We show additional results for expression manipulation, and analyze the effect of our expression regularization and directional clip loss in Figure 11. Our proposed technique outperforms others and achieves realistic texture manipulation.

Texture Manipulation: We show additional texture manipulation results with a large variety of text prompts in Figure 12. As can be seen, our method is able to generate a wide variety of textures, even capable of adapting identity when implied by the text prompt.

Effect of Pre-training: Finally, we analyze the effect of pre-training the texture and expression mappers. We first pre-train the texture mapper $\mathcal{T}$ and expression mapper $\mathcal{E}$ to predict zero offsets using ℓ_2 regularization before training them with our CLIP loss. We show the effect of pre-training mapper networks to predict zero offsets in Figure 13. Without pre-training, the textures begin with unrealistic values and converge to low quality styles with visible artifacts.

7. Architecture & Training Details

ClipFace is implemented in the Pytorch Lightning framework [11, 39]. For differentiable rendering, we use the NvDiffrast [29] library.

Texture Generation: For texture generation, we use the StyleGAN2 architecture with adaptive discriminator augmentation [22]. For all of our experiments, we operate at a texture and image resolution of 512×512. We apply augmentation to both the full-image discriminator as well as the patch discriminator, which operates on patches of size 64×64. For augmentations, we apply geometric transformations such as image flipping, rotation and scaling, as well as color transformations such as changing image brightness, contrast, hue, saturation, etc. For training, we used the Adam [26] optimizer with a learning rate of 0.002, batch size 8 per GPU, gradient penalty 10, and path length regularization 2. We perform multi-GPU training on 3 RTX A6000 GPUs and train for 300,000 iterations.

Texture and Expression Manipulation: For the text-guided manipulation experiments, we use a 4-layer MLP architecture with LReLU activations. For the texture mapper, we use 18 identical MLPs to predict texture offsets for different levels of the latent code $\mathbf{w}_{\text{init}} = \{\mathbf{w}^1_{\text{init}}, \mathbf{w}^2_{\text{init}}, ...\mathbf{w}^{18}_{\text{init}}\} \in \mathbb{R}^{512\times 18}$.

$$\mathbf{w}_{\text{delta}} = \begin{cases} \mathcal{T}^1(\mathbf{w}^1_{\text{init}}) \\ \mathcal{T}^2(\mathbf{w}^2_{\text{init}}) \\ \vdots \\ \mathcal{T}^{18}(\mathbf{w}^{18}_{\text{init}}). \end{cases} \tag{15}$$

Each texture MLP $\mathcal{T}^i$ takes as input 512-dimensional latent code $\mathbf{w}^i_{\text{init}}$ and outputs the 512-dimensional offset $\mathbf{w}^i_{\text{delta}}$. The expression mapper $\mathcal{E}$ takes the mean latent code as input $\mathbf{w}_{\text{mean}} \in \mathbb{R}^{512}$, and predicts the expression offset

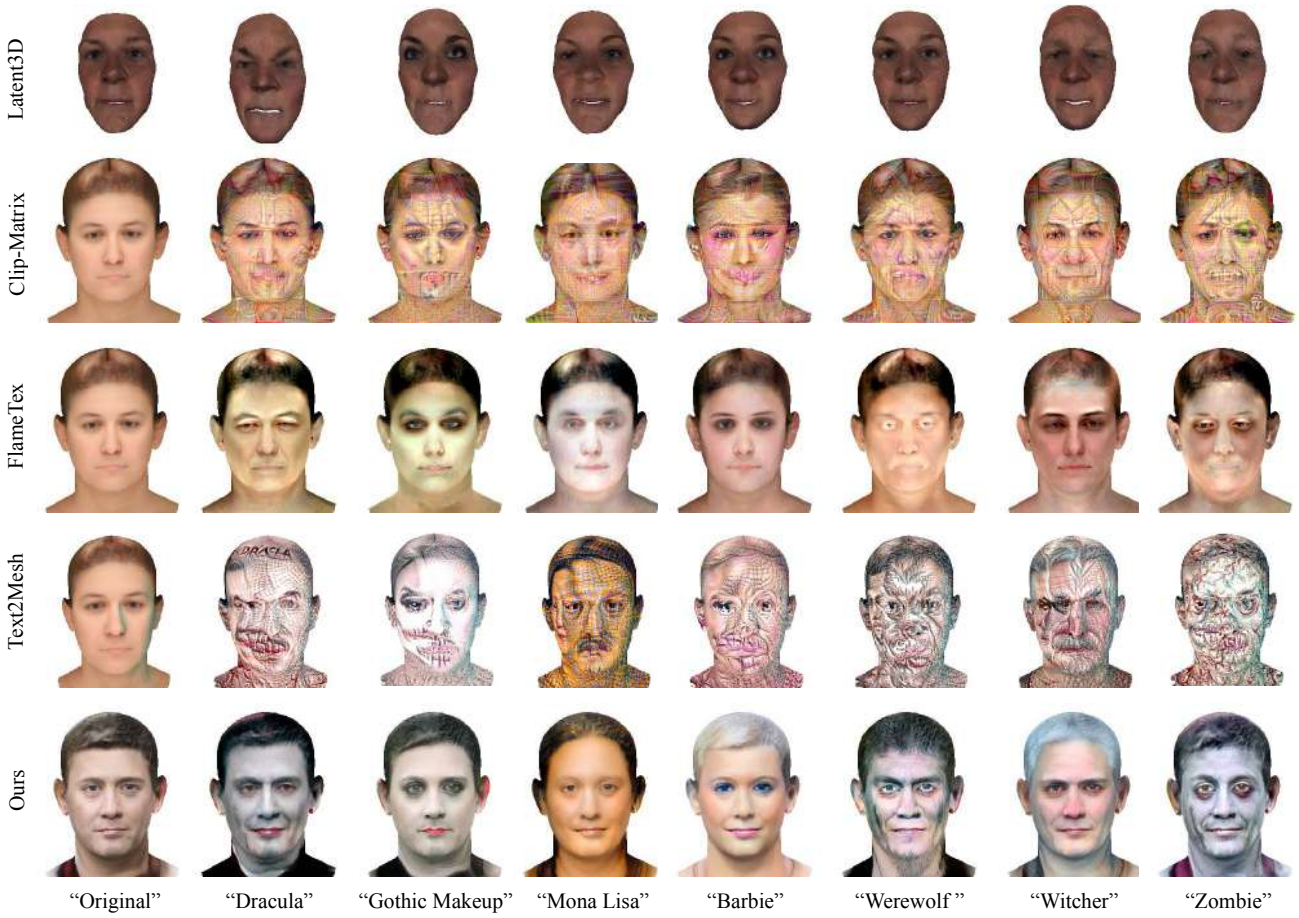


Figure 10. Additional texture manipulation results in comparison with baselines, from the text prompts shown. Our method outperforms baseline approaches in texture quality.

$\psi_{\text{delta}} \in \mathbb{R}^{50}$ as output. The network architecture for both these mappers is shown in Figure 14. For language supervision, we use the CLIP model [43]. For our experiments, we use the pre-trained 'ViT-B/32' variant for computing the CLIP loss. We use a learning rate of 0.005 for the expression mapper and 0.0001 for the texture mapper.

Texturing for Animation Sequences: For the task of texturing for animations, we learn only the texture mapper. We use the same architecture as shown in Figure 14(a). Since a given animation sequence consists of multiple frames, we share the texture mapper across different timestamps. For all text manipulation experiments, we train for 20,000 iterations.

8. Baseline Implementations

Latent3d [6]: This method builds upon TB-GAN [14], a generative model $\mathcal{G}$ that takes one-hot encoded facial expression vector $\vec{\mathbf{e}}$ and a random noise vector $\vec{\mathbf{z}} \in \mathbb{R}^d$ as input and generates shape, shape-normal and texture images. Given a pretrained generator $\mathcal{G}$, the method optimizes offset $\Delta\mathbf{c}$ for the intermediate layer $\mathbf{c}$ which is 4×4 dense layer of TB-GAN. The offset $\Delta\mathbf{c}$ gives the direction in which the target attributes specified by text prompt $\mathbf{t}$ are enhanced, while other attributes stay unchanged. The authors use a Clip-loss $\mathcal{L}_{\text{CLIP}}$, supplemented with an identity loss $\mathcal{L}_{\text{ID}}$ and L2 regularization $\mathcal{L}_{\text{L2}}$ to perform meaningful manipulation of meshes:

$$\arg\min_{\Delta\mathbf{c}\in\mathcal{C}} \mathcal{L}_{\text{CLIP}} + \lambda_{\text{ID}}\mathcal{L}_{\text{ID}} + \lambda_{\text{L2}}\mathcal{L}_{\text{L2}}, \tag{16}$$

where λ_{ID} and λ_{L2} are hyperparameters for the $\mathcal{L}_{\text{ID}}$ and $\mathcal{L}_{\text{L2}}$ respectively. The identity loss $\mathcal{L}_{\text{ID}}$ minimizes the distance between the identity of original renders and manipulated renders:

$$\mathcal{L}_{\text{ID}} = 1 - \langle R(\mathcal{G}(\mathbf{c})), R(\mathcal{G}(\mathbf{c} + \Delta\mathbf{c}))\rangle, \tag{17}$$

where R is the ArcFace [9], a facial recognition network and $\langle ., . \rangle$ computes the cosine similarity between the identi-

Figure 11. Additional expression manipulation results: our proposed method achieves realistic expression manipulation. Both our expression regularization and directional clip loss contribute notably to realistic output quality.

ties of the initial rendering and manipulated rendering. The L2 loss is used by authors to prevent artifact generation and can be written as:

$$\mathcal{L}_{\text{L2}} = \|\mathbf{c} - (\mathbf{c} + \Delta\mathbf{c})\|_2 \tag{18}$$

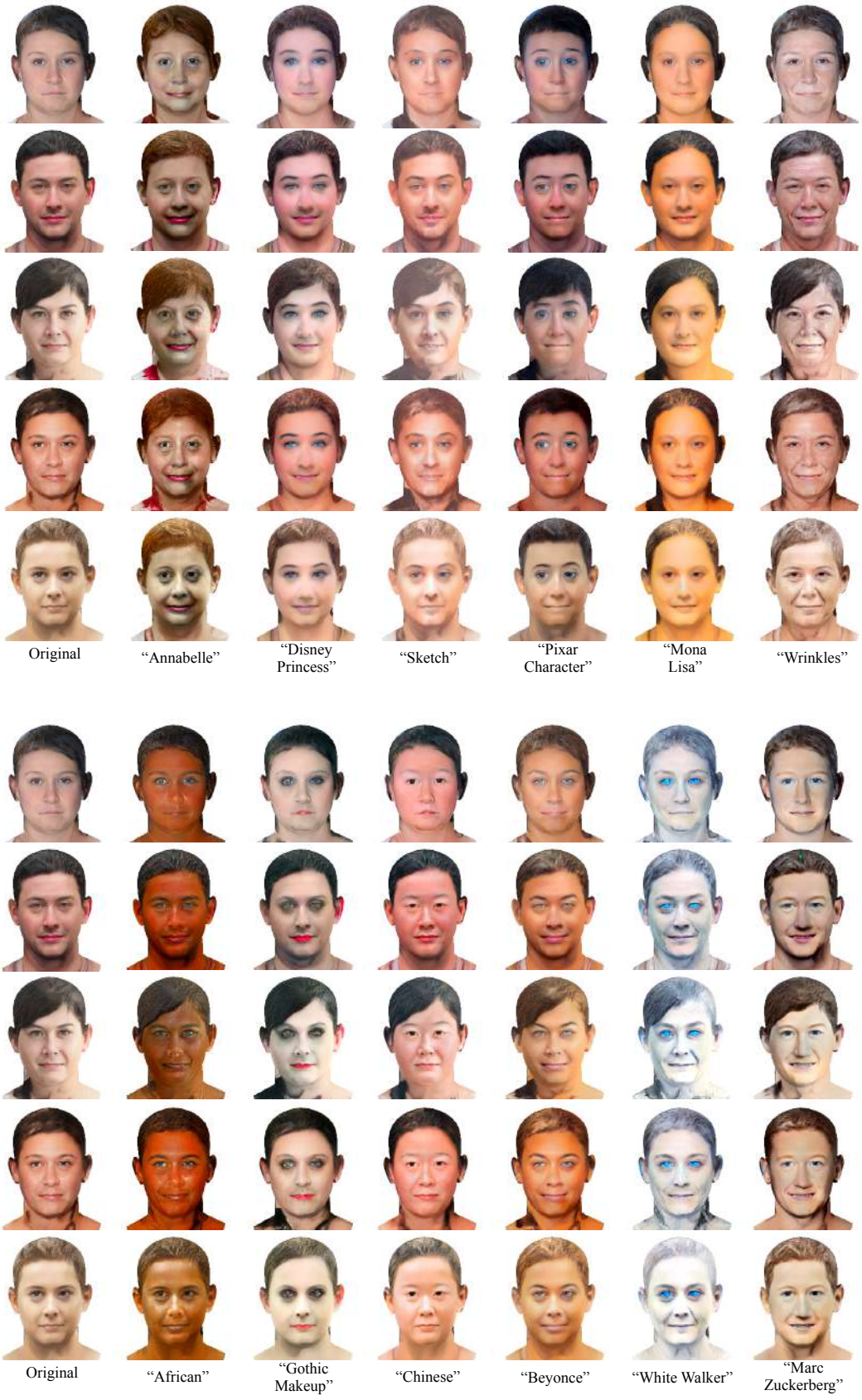


Figure 12. Additional texturing results: Our proposed method generates a diverse range of textures.

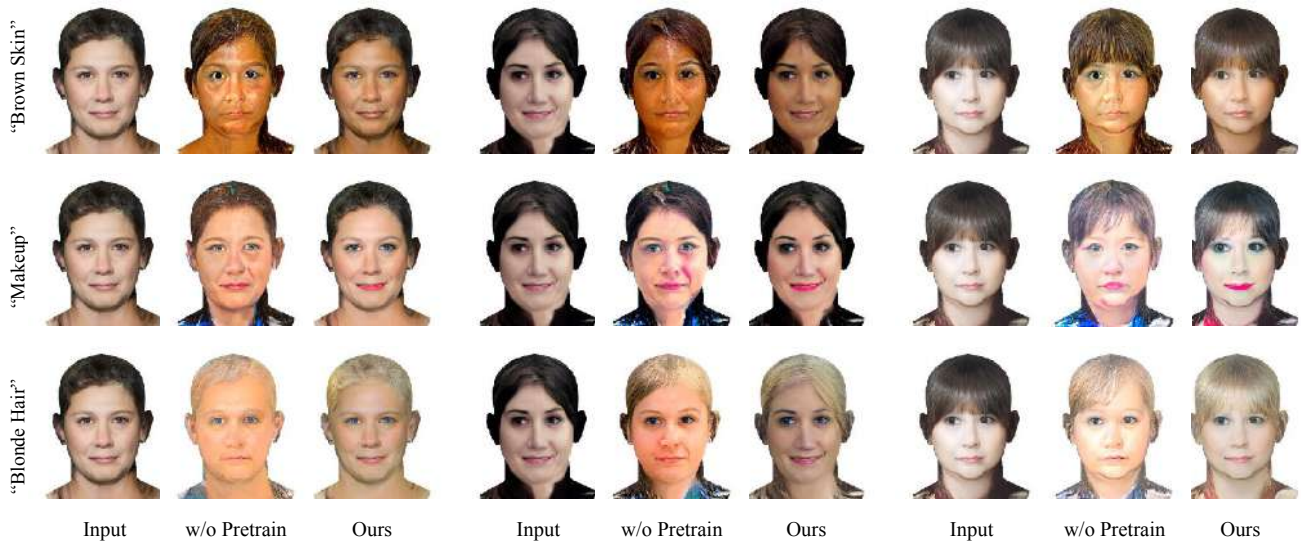
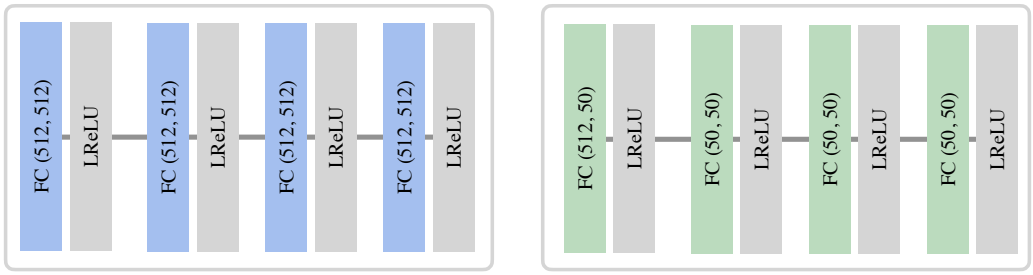


Figure 13. We evaluate the effect of pre-training texture and expression mappers. 'w/o Pretrain' refers to the case when mappers are not trained to predict zero offsets before performing text manipulation. Pre-training helps to produce realistic texture changes.



(a) Texture Mapper Architecture (b) Expression Mapper Architecture

Figure 14. Architecture Overview: Network architecture for Texture Mapper (left) and Expression mapper (right). FC(x, y) refers to a fully-connected layer, where x and y denote the input and output dimensions, respectively. LReLU refers to LeakyReLU activations.

For the Clip loss, the authors use a list of text templates like 'a photo of a ...'; 'a face of a ...', etc prefixed to the target style:

$$\mathcal{L}_{\text{CLIP}} = \frac{\Sigma_{j=1}^{K}\Sigma_{i=1}^{N} D_{\text{CLIP}}(\mathcal{I}_i, t_j)}{K \cdot N}, \tag{19}$$

where $\mathcal{I}_i$ is the rendered image from a list of N rendered images, t_j is the target text t embedded in a text template from a list of K templates. D_{CLIP} minimizes the cosine distance between CLIP embeddings of the rendered image $\mathcal{I}_i$ and the set of text prompts t_j.

ClipMatrix [20]: Given a 3D mesh and initial texture map T_{init}, ClipMatrix optimizes the texture image offset T_{delta} to match the image rendering I to the text prompt t from random camera view c:

$$\mathcal{L}(T_{delta}) = \sum_{t} \mathop{\mathbb{E}}_{c\sim\pi_c} \mathcal{L}_{\text{CLIP}}(I, t). \tag{20}$$

By sampling from random camera angles $c \sim \pi_c$ during optimization, the method ensures that output mesh shows the desired properties from different viewing angles. The image rendering can be obtained as:

$$I = \mathcal{R}(\mathcal{M}, T_{tgt}, c), \tag{21}$$

where $\mathcal{M}$ refers to the 3D mesh, $T_{tgt} = T_{init} + T_{delta}$ denotes the final UV texture map and c denotes the camera view. The clip loss $\mathcal{L}_{\text{CLIP}}(I, t)$ minimizes the negative cosine similarity in CLIP embedding space between image I and the fixed text prompt t.

$$\mathcal{L}_{\text{CLIP}}(I, t) = -cos(\phi_i(I), \phi_t(t)) \tag{22}$$

where ϕ_i and ϕ_t refer to the clip image and text encoder respectively. The texture offset T_{delta} is initialized with zero and during optimization clipping is applied to final texture image $T_{tgt} \in [-1, 1]$ to ensure that it stays in valid image range.

FlameTex [18]: FlameTex is the PCA-based texturing model designed specifically for FLAME face model [33]. The texture space for FlameTex is built using randomly selected 1500 images from the FFHQ dataset [23] and the base texture from the Basel Face Model [41]. Given a mean texture $T_{mean} \in \mathbb{R}^{512\times512\times3}$ and texture basis $T_{basis} \in \mathbb{R}^{50\times512\times512\times3}$ from the FlameTex texture model, we optimize for the texture basis coefficients $\boldsymbol{\omega} \in \mathbb{R}^{50}$ to match the target text prompt t to generate the desired texture map T_{tgt}:

$$\boldsymbol{\omega}^* = \arg\min_{\boldsymbol{\omega}} \mathcal{L}_{\text{CLIP}}(I, t) + \lambda_{\text{L2}}\mathcal{L}_{\text{L2}}(\boldsymbol{\omega}), \tag{23}$$

where $\mathcal{L}_{\text{CLIP}}$ refers to the clip loss between text prompt t and the rendered image I and $\mathcal{L}_{\text{L2}}$ refers to the L2 regularization for the texture coefficients $\boldsymbol{\omega}$ with λ_{L2} controlling the strength of the regularization. The desired texture map is given by:

$$T_{tgt} = T_{mean} + \omega * T_{basis}. \tag{24}$$

The image rendering can be obtained as:

$$I = \mathcal{R}(\mathcal{M}, T_{tgt}, c), \tag{25}$$

where $\mathcal{M}$ refers to the Flame 3D mesh, T_{tgt} denotes the final UV texture map and c denotes the camera view. The L2 regularization is applied to prevent the model from producing unrealistic texture and is given by:

$$\mathcal{L}_{\text{L2}} = \|\boldsymbol{\omega}\|_2. \tag{26}$$

We initialize $\boldsymbol{\omega}$ with zero and start from base texture T_{mean} during optimization.

Text2Mesh [38]: Given a 3D mesh, the method uses coordinate-based MLPs to predict per-vertex color and displacement conforming to the target text prompt t used for stylizing the mesh. In our experiments, we first remesh the Flame 3D mesh to increase vertices from 5K to 60K as the method works reasonably well for meshes with a higher vertex count. We used the default hyperparameters used by authors for stylizing human body meshes, as the authors did not perform experiments on human face meshes.

For every vertex point p, first the positional encoding γ is applied to obtain high frequency features, before passing them to the MLP:

$$\gamma(p) = [\cos(2\pi\mathbf{B}p), \sin(2\pi\mathbf{B}p)]^T, \tag{27}$$

where $\mathbf{B} \in \mathbb{R}^{n\times3}$ is the random Gaussian matrix. We first pretrain the MLP $f_{\boldsymbol{\theta}}$ to predict a base texture T_{init} to begin learning from a reasonable starting texture. Since we do not wish to change the geometry, we do not perform vertex displacements in our experiments. Our sanity experiments for vertex displacements produced unrealistic geometries. We train with loss function and augmentations proposed in the main paper. The loss function can be written as:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \mathcal{L}_{\text{CLIP}}(I, t), \tag{28}$$

where I refers to the image rendered from different viewpoints and t refers to the target text prompt. The image rendering I can be obtained as:

$$I = \mathcal{R}(\mathcal{M}, [f_{\boldsymbol{\theta}}(\gamma(p_i))]_{i=1}^{N}, c), \tag{29}$$

where $\mathcal{R}$ refers to the differentiable rendering, c refers to the random camera view and $p_i \in \mathbb{R}^3$ refers to the vertex coordinate of the mesh and N refers to the total number of mesh vertices.

References

[1] Rameen Abdal, Peihao Zhu, John Femiani, Niloy J. Mitra, and Peter Wonka. Clip2stylegan: Unsupervised extraction of stylegan edit directions. *CoRR*, abs/2112.05219, 2021. 2

[2] Rameen Abdal, Peihao Zhu, Niloy J. Mitra, and Peter Wonka. Styleflow: Attribute-conditioned exploration of stylegan-generated images using conditional continuous normalizing flows. *ACM Trans. Graph.*, May 2021. 2

[3] Omri Avrahami, Dani Lischinski, and Ohad Fried. Blended diffusion for text-driven editing of natural images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18208–18218, 2022. 2

[4] David Bau, Alex Andonian, Audrey Cui, YeonHwan Park, Ali Jahanian, Aude Oliva, and Antonio Torralba. Paint by word, 2021. 2

[5] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '99, page 187–194, USA, 1999. ACM Press/Addison-Wesley Publishing Co. 2

[6] Zehranaz Canfes, M. Furkan Atasoy, Alara Dirik, and Pinar Yanardag. Text and image guided 3d avatar generation and manipulation, 2022. 2, 3, 6, 10

[7] Katherine Crowson. Vqgan-clip, 2021. 2

[8] Boris Dayma, Suraj Patil, Pedro Cuenca, Khalid Saifullah, Tanishq Abraham, Phuc Le Khac, Luke Melas, and Ritobrata Ghosh. Dall·e mini, 7 2021. 2

[9] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4685–4694, 2019. 10

[10] Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. Disentangled and controllable face image generation via 3d imitative-contrastive learning. In *IEEE Computer Vision and Pattern Recognition*, 2020. 2

[11] William Falcon et al. Pytorch lightning. *GitHub. Note: https://github. com/PyTorchLightning/pytorch-lightning*, 3(6), 2019. 9

[12] Yao Feng, Haiwen Feng, Michael J. Black, and Timo Bolkart. Learning an animatable detailed 3D face model from in-the-wild images. *ACM Transactions on Graphics (ToG), Proc. SIGGRAPH*, 40(4):88:1–88:13, Aug. 2021. 3

[13] Rinon Gal, Or Patashnik, Haggai Maron, Gal Chechik, and Daniel Cohen-Or. Stylegan-nada: Clip-guided domain adaptation of image generators, 2021. 2, 4

[14] Baris Gecer, Alexander Lattas, Stylianos Ploumpis, Jiankang Deng, Athanasios Papaioannou, Stylianos Moschoglou, and Stefanos Zafeiriou. Synthesizing coupled 3d face modalities by trunk-branch generative adversarial networks. In *Proceedings of the European conference on computer vision (ECCV)*. Springer, 2020. 2, 3, 10

[15] Baris Gecer, Stylianos Ploumpis, Irene Kotsia, and Stefanos Zafeiriou. Ganfit: Generative adversarial network fitting for high fidelity 3d face reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2

[16] Baris Gecer, Stylianos Ploumpis, Irene Kotsia, and Stefanos P Zafeiriou. Fast-ganfit: Generative adversarial network for high fidelity 3d face reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[17] Partha Ghosh, Pravir Singh Gupta, Roy Uziel, Anurag Ranjan, Michael J. Black, and Timo Bolkart. GIF: Generative interpretable faces. In *International Conference on 3D Vision (3DV)*, pages 868–878, 2020. 2

[18] HavenFeng. Photometric flame fitting. https://github.com/HavenFeng/photometric_optimization, 2019. 6, 14

[19] Fangzhou Hong, Mingyuan Zhang, Liang Pan, Zhongang Cai, Lei Yang, and Ziwei Liu. Avatarclip: Zero-shot text-driven generation and animation of 3d avatars. *ACM Transactions on Graphics (TOG)*, 41(4):1–19, 2022. 2

[20] Nikolay Jetchev. Clipmatrix: Text-controlled creation of 3d textured meshes, 2021. 6, 13

[21] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of GANs for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 2

[22] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Proc. NeurIPS*, 2020. 3, 9

[23] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4396–4405, 2019. 3, 14

[24] 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

[25] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. Clip-mesh: Generating textured meshes from text using pretrained image-text models. December 2022. 3

[26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. 9

[27] Umut Kocasari, Alara Dirik, Mert Tiftikci, and Pinar Yanardag. Stylemc: Multi-channel based fast text-guided image generation and manipulation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 895–904, January 2022. 2

[28] Marek Kowalski, Stephan J. Garbin, Virginia Estellers, Tadas Baltrušaitis, Matthew Johnson, and Jamie Shotton. Config: Controllable neural face image generation. In *European Conference on Computer Vision (ECCV)*, 2020. 2

[29] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular primitives for high-performance differentiable rendering. *ACM Transactions on Graphics*, 39(6), 2020. 4, 6, 9

[30] Alexandros Lattas, Stylianos Moschoglou, Baris Gecer, Stylianos Ploumpis, Vasileios Triantafyllou, Abhijeet Ghosh, and Stefanos Zafeiriou. Avatarme: Realistically renderable 3d facial reconstruction "in-the-wild". In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[31] Alexandros Lattas, Stylianos Moschoglou, Stylianos Ploumpis, Baris Gecer, Abhijeet Ghosh, and Stefanos P Zafeiriou. Avatarme++: Facial shape and brdf inference with photorealistic rendering-aware gans. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2

[32] Myunggi Lee, Wonwoong Cho, Moonheum Kim, David I. Inouye, and Nojun Kwak. Styleuv: Diverse and high-fidelity uv map generative model. *ArXiv*, abs/2011.12893, 2020. 2

[33] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 1, 3, 7, 14

[34] Yuchen Liu, Zhixin Shu, Yijun Li, Zhe Lin, Richard Zhang, and S. Y. Kung. 3d-fm gan: Towards 3d-controllable face manipulation. *ArXiv*, abs/2208.11257, 2022. 2

[35] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, Oct. 2015. 1

[36] Huiwen Luo, Koki Nagano, Han-Wei Kung, Qingguo Xu, Zejian Wang, Lingyu Wei, Liwen Hu, and Hao Li. Normalized avatar synthesis using stylegan and perceptual refinement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11662–11672, June 2021. 2

[37] Richard T. Marriott, Sami Romdhani, and Liming Chen. A 3d gan for improved large-pose facial recognition. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13440–13450, 2021. 2

[38] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2mesh: Text-driven neural stylization for meshes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13492–13502, June 2022. 2, 3, 6, 14

[39] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 9

[40] Or Patashnik, Zongze Wu, Eli Shechtman, Daniel Cohen-Or, and Dani Lischinski. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2085–2094, October 2021. 2, 4

[41] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. *2009 Sixth IEEE International Conference on Advanced Video and Signal Based Surveillance*, pages 296–301, 2009. 14

[42] Mathis Petrovich, Michael J. Black, and Gül Varol. TEMOS: Generating diverse human motions from textual descriptions. In *European Conference on Computer Vision (ECCV)*, 2022. 2

[43] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021. 2, 4, 10

[44] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents, 2022. 2

[45] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation, 2021. 2

[46] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. 2022. 2

[47] Ron Slossberg, Ibrahim Jubran, and Ron Kimmel. Unsupervised high-fidelity facial texture generation and reconstruction. *arXiv preprint arXiv:2110.04760*, 2021. 2, 6

[48] Ayush Tewari, Mohamed Elgharib, Gaurav Bharaj, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zöllhofer, and Christian Theobalt. Stylerig: Rigging stylegan for 3d control over portrait images, cvpr 2020. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, june 2020. 2

[49] Ayush Tewari, Mohamed Elgharib, Mallikarjun B R, Florian Bernard, Hans-Peter Seidel, Patrick Pérez, Michael Zollhöfer, and Christian Theobalt. Pie: Portrait image embedding for semantic control. *ACM Trans. Graph.*, 2020. 2

[50] Kim Youwang, Kim Ji-Yeon, and Tae-Hyun Oh. Clip-actor: Text-driven recommendation and stylization for animating human meshes. In *ECCV*, 2022. 2

[51] zllrunning. face-parsing.pytorch. https://github.com/zllrunning/face-parsing.PyTorch, 2018. 3