A. Implementation Details

We implement our model on top of the official StyleGAN2 [9] and the PyTorch release of Deep3DFaceRecon [2]. FR and RDR are both part of Deep3DFaceRecon [2] and G and D are part of StyleGAN2 [9]. We use the dataset tool provided in Deep3DFaceRecon [2] to realign FFHQ [8] so that image $x$ aligns with 3DMM representation $\text{rep}$.

StyleGAN2 backbone. We follow the latest findings in StyleGAN3 [7] and omit several insignificant details to simplify StyleGAN2 [9]. We remove mixing regularization and path length regularization. The depth of the mapping network is decreased to 2, as recommended by Karras et al. It is also noticed that decreasing the dimensionality of $z$ while maintaining the dimensions of $w$ is beneficial [14]. Therefore, we reduce the dimensions of $z$ to 64. All details are otherwise unchanged, including the network architecture, equalized learning rate, minibatch standard deviation, weight (de)modulation, lazy regularization, bilinear resampling, and exponential moving average of the generator weights. Due to the addition of an encoder, our model is slightly larger than StyleGAN2 config F [9]. Along with the encoder, our generator contains 38.6M parameters whereas the StyleGAN2 generator contains 30.0M parameters. Our discriminator, containing 28.9M parameters, is the same as in StyleGAN2.

Face reconstruction and differentiable renderer. We use the pretrained checkpoint provided by Deng et al. [2] for FR. This updated checkpoint was trained on an augmented dataset that includes FFHQ [8] and shows slight performance improvement over the TensorFlow release of Deep3DFaceRecon. We use the differentiable renderer RDR that comes with the checkpoint for FR from the same code repository. This renderer uses the Basel Face Model from 2009 [5] as the 3DMM parametric model for face modeling, and nvdiffrest [10] for rasterization. We modify RDR so it outputs $a$ and $n$ along with $r$. The renderer is otherwise unchanged. We base our model upon the Basel Face Model [5] rather than later work such as FLAME [11], as FLAME does not contain skin color information, only geometry.

Training procedure. Following the StyleGAN family [7–9], we adopt the non-saturating loss [3] and R1 gradient penalty [13] as the loss function for GAN training. We additional append our $\mathcal{L}_{\text{consistency}}$, resulting in the following objectives:

$$
\mathcal{L}_D = -\mathbb{E}_{p,z}[\log(1-D(G(\text{rep}(p),z)))] - \\
\mathbb{E}_x[\log(D(x))] + \frac{\gamma}{2}\mathbb{E}_x[\|\nabla D(x)\|_2^2]
$$

$$
\mathcal{L}_G = -\mathbb{E}_{p,z}[\log(D(G(\text{rep}(p),z)))] + \lambda\mathcal{L}_{\text{consistency}}
$$

We closely follow the training configurations of the baseline model in Karras et al. [6] and set $\gamma = 1$. The batch size is set to 64 and the group size of minibatch standard deviation is set to 8. We empirically set $\lambda = 20$ and the length of progressive blending to $k = 2 \times 10^6$. The learning rate of both $G$ and $D$ is set to $2.5 \times 10^{-5}$. We train our model until $D$ sees 25M real images [7–9]; training took 10 days on $4 \times A6000$ GPUs.

Instead of approximating the distribution $P(p)$ using a VAE [1], we simply use its empirical distribution when sampling $p \sim P(p)$ and find this to be sufficient given our 3DMM representation.

B. Encoder Architecture

Figure 1 depicts the internal structure of a general stage (every stage other than the highest resolution stage and the $4 \times 4$ stage) of our encoder $E$. Following recent advances in network architecture [12, 15], our ResNet [4] design of $E$ differs from the architecture of $D$ [9] in several ways.

General stage. We notice that the two architectural changes in [12] that lead to most performance boost are separate downsampling layers and fewer activations. Thus, we move the skip branch of the transition residual block up to the stem as a transition layer, and remove all activations in the residual block unless they are between two consecutive convolutional layers. We use leaky ReLU activation with $\alpha = 0.2$, and bilinear downsampling instead of strided convolution [8, 9]. We use the 1-3-1 bottleneck residual block as it is more efficient.
Figure 1. The detailed breakdown of a general stage of $E$.

Figure 2. Reference-based generation results. We extract the expression, illumination, and pose coefficients from reference images (first row) and apply them to randomly generated images (first column).

Figure 3. Style mixing results at different scales. Using the same three images for Source A and Source B, we replace the style vectors of images from Source A by the style vectors of images from Source B at coarse resolutions ($4 \times 4 - 8 \times 8$), middle resolutions ($16 \times 16 - 32 \times 32$), and fine resolutions ($64 \times 64 - 256 \times 256$).

Specialization. We remove bilinear downsampling from the transition layer of the highest resolution stage; it is otherwise identical to a general stage. Since the $4 \times 4$ stage of the synthesis network contains only one synthesis layer, we place one toFeat layer without leaky ReLU in the $4 \times 4$ stage of $E$ accordingly.

We show additional results in controlled generation that display the robustness of our model and explain what control exists in the non-conditioned $z$ space.

Reference-based generation In Fig. 2, we task our model with reference-based generation where we keep the identity of
Figure 4. Resampling the 3DMM coefficient vector \( p \) with the same noise vector \( z \) shows high consistency in the background and clothes while the face completely changes.

Figure 5. Reference-based generation results that show unexpected skin tone change. We see that the albedo predicted by FR does not faithfully capture the darker skin tone.

**References**


[9] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the...


