Supplementary material for
Removing the Quality Tax in Controllable Face Generation

Yiwen Huang Zhiqiu Yu Xinjie Yi Yue Wang James Tompkin
Brown University

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 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 nvdiffrast [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:

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

$$\mathcal{L}_G = -\mathbb{E}_{p,z}[\log(D(G(\text{rep}(p),z)))] + \lambda\mathcal{L}_{\text{consistency}} \quad (2)$$

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 $k = 2\times10^6$. The learning rate of both G and D is set to 2.5×10^{-3}. We train our model until D sees 25M real images [7–9]; training took 10 days on 4 × 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 4×4 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

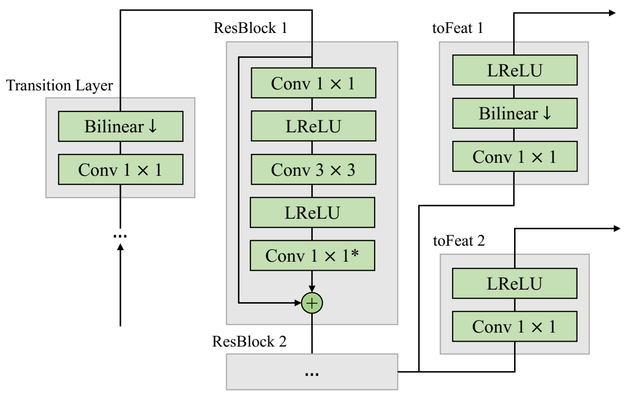

Figure 1. The detailed breakdown of a general stage of E.

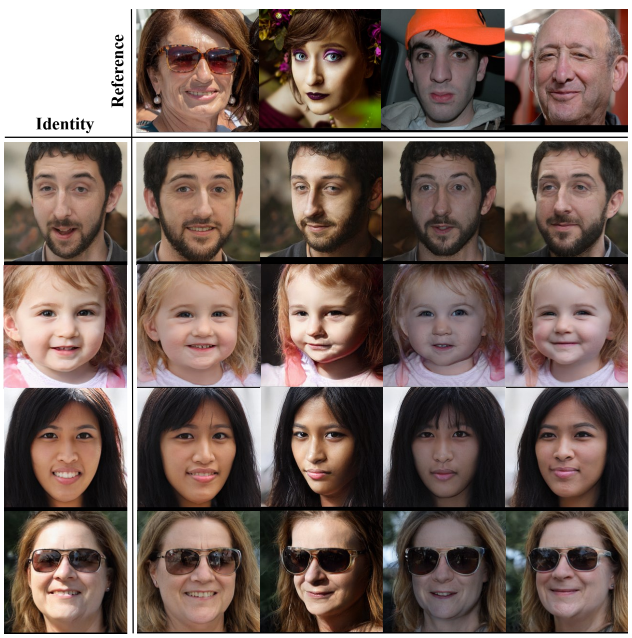

Figure 2. Reference-based generation results. We extract the expression, illumination, and pose coefficients from reference images (first row) and apply them to randomly generated images (first column).

than the 3-3 block [4]. The final convolutional layer (marked by *) in the residual block is initialized to 0 [16], and this eliminates the need for normalization or residual rescaling [9]. We apply equalized learning rate to all convolutional layers.

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

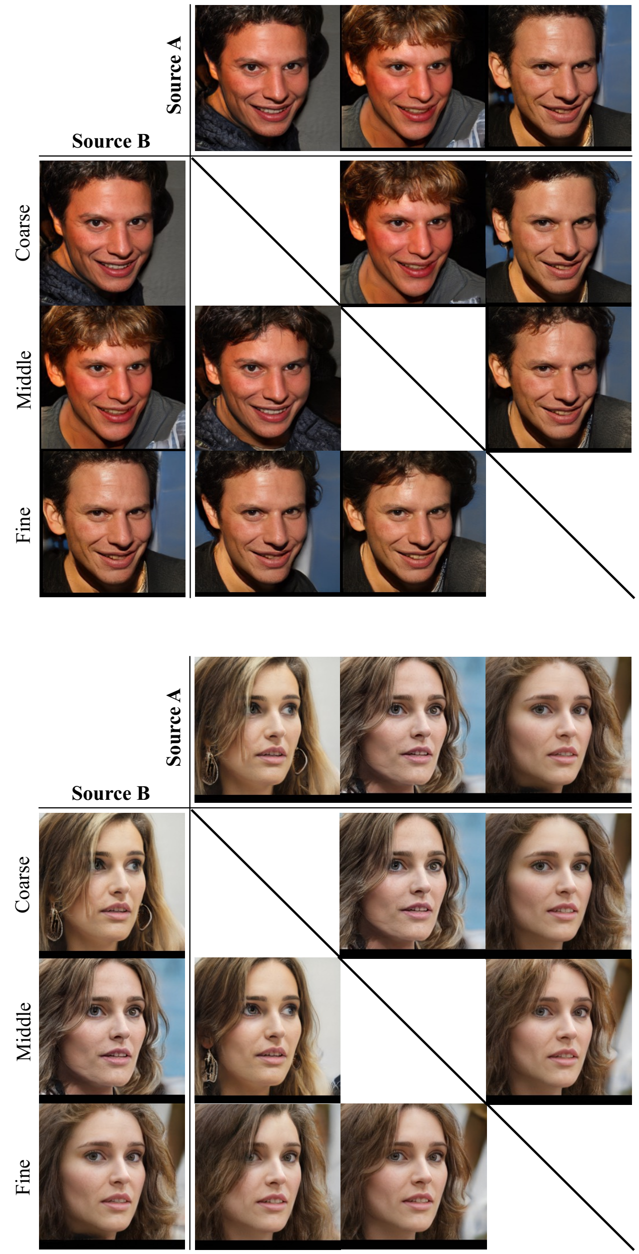

Figure 3. Style mixing results at different scales. Using the same three images for Source A and Source B, we replace the style vectors of images from Source A by the style vectors of images from Source B at coarse resolutions (4×4 - 8×8), middle resolutions (16×16 - 32×32), and fine resolutions (64×64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

C. More Results

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.

a generated image and swap its expression, illumination, and pose with those of a real image. We can see that the respective attributes from their source are all well preserved, and the image quality does not degrade. This again demonstrates the disentangled face generation from our model.

Feature granularity To inspect the impact of feature variability across the layers of the decoder, we inspect the impact of swapping features across images with the same p. In Fig. 3, we randomly pick a 3DMM coefficient vector p and randomly sample z's to generate three images (the same images for Source A and Source B). Following StyleGAN [8], we replace some of the style vectors w^+ of images from Source A by the corresponding style vectors of images from Source B at coarse, middle, and fine scales. As p is the same, the overall face region will not change significantly.

At coarse scale, there is no visible change to the images from Source A. This is expected as the high-level attributes of the image are supposed to be determined by the p vector. At middle scale, the images from Source A remain mostly unchanged except finer facial features such as the hair now resemble those in the image from Source B. At fine scale, the images from Source A undergo more significant changes where the color scheme that affects the background, clothes, hair color, and skin color now resembles those in the image from Source B. This experiment indicates that each subset of the style vectors w^+ controls a different set of features in the generated image. We also notice that attributes controlled by p remain unchanged at any scale, which means our model's p space and z space are well separated.

3DMM vector resampling with fixed noise As opposed to the experiment conducted in the main paper where we vary the noise z with fixed 3DMM vector p, we now vary p with fixed z (Fig. 4). We can see that despite the drastic change in the facial attributes from different p's, the background and clothes remain largely consistent with the same z. This is another proof that the z vector has a good control of the attributes not controlled by p.

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.

Limitations. Due to the use of a pretrained **FR** and **RDR**, our model inevitably inherits the limitations of these models. We find that Deep3DRecon [2] performs particularly poor on darker skin tone, in that it tends to predict the skin tone as the result of dim illumination. This leads to unexpected skin tone change when editing the illumination (Fig. 5). Moreover, our model does not provide explicit control over attributes not represented in $\mathcal{P}$ such as hair and eyeglasses. We believe these restrictions can be resolved in the future by an improved 3DMM.

References

[1] Yu Deng, Jiaolong Yang, Dong Chen, Fang Wen, and Xin Tong. Disentangled and controllable face image generation via 3d imitative-contrastive learning, 2020. 1

[2] Yu Deng, Jiaolong Yang, Sicheng Xu, Dong Chen, Yunde Jia, and Xin Tong. Accurate 3d face reconstruction with weakly-supervised learning: From single image to image set. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 0–0, 2019. 1, 3

[3] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020. 1

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 1, 2

[5] IEEE. *A 3D Face Model for Pose and Illumination Invariant Face Recognition*, Genova, Italy, 2009. 1

[6] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. *Advances in neural information processing systems*, 33:12104–12114, 2020. 1

[7] Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Alias-free generative adversarial networks. *Advances in Neural Information Processing Systems*, 34:852–863, 2021. 1

[8] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4401–4410, 2019. 1, 3

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

image quality of stylegan. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8110–8119, 2020. 1, 2

[10] 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. 1

[11] 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

[12] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11976–11986, 2022. 1

[13] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018. 1

[14] Axel Sauer, Katja Schwarz, and Andreas Geiger. Stylegan-xl: Scaling stylegan to large diverse datasets. In *ACM SIGGRAPH 2022 conference proceedings*, pages 1–10, 2022. 1

[15] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10819–10829, 2022. 1

[16] Hongyi Zhang, Yann N Dauphin, and Tengyu Ma. Fixup initialization: Residual learning without normalization. *arXiv preprint arXiv:1901.09321*, 2019. 2