My3DGen: A Scalable Personalized 3D Generative Model

Supplementary Material

A. Overview of Supplementary Materials

The supplementary materials include the following additional details:

- Sec. B describes the details of our convolutional LoRA decomposition, where we show the difference between the original implementation and ours. We compare the results visually in Fig. 10, where facial texture is accompanied by checkerboard artifacts using the original LoRA paper's code implementation [33].
- Sec. C provides details on the celebrity dataset and additional information regarding our ablation studies on the effect of dataset size in Sec. 4.4.
- Sec. D provides additional results for Sec. 4.4, where in-the-wild cat images are used to personalize pretrained EG3D model following the same My3DGen procedures.
- Sec. E shows image inversion results without PTI [62] in Fig. 12. We re-project latent code into personal convex hull following Mystyle [53] for inversion where model weights remain unchanged.
- Sec. F further discusses the interpolation results shown in Sec. 4.2, particularly why the interpolation curve is different from that in Mystyle [53].
- Sec. G describes the detailed hardware configurations and training time for our experiments.
- In Sec. H, we display failure cases for our experiments in Fig. 13.

B. LoRA for Convolutional Layer

LoRA [33] is originally defined for matrix multiplication for fully-connected layers. However, convolution operation with C_1 output channels, C_2 input channels, and kernel size of $k \times k$ is often implemented as matrix multiplication with a matrix W under "im2col" [14] transform on the image X. The matrix W has dimension $W \in \mathbb{R}^{C_1 \times C_2 kk}$.

$$\text{Conv}_\theta(X) = W\,\text{im2col}(X) \tag{3}$$

Therefore, we can decompose matrix M for convolution layers similar to LoRA. With a rank r LoRA decomposition, let $B \in \mathbb{R}^{C_1 \times r}$ and $A \in \mathbb{R}^{r \times C_2 kk}$, we have the following equation.

$$W = BA \tag{4}$$

We found official LoRA [33] implementation performs the following decomposition. Matrix W is assumed to have dimension $W \in \mathbb{R}^{C_1 k \times C_2 k}$, while B, A have dimension $B \in \mathbb{R}^{C_1 k \times r}$, $A \in \mathbb{R}^{r \times C_2 k}$. Ours differs from the original LoRA implementation in two ways:

- LoRA showed weight matrix $W \in \mathbb{R}^{C_1 \times C_2 kk}$ that maps from input space $C_2 kk$ to output space C_1 of a layer can have low rank structure. It is unclear if matrix $W \in \mathbb{R}^{C_1 k \times C_2 k}$ has low rank structure.

Figure 10. Following the same personalization pipeline, we compare reconstructed results using the original LoRA code (middle) and our own LoRA implementation (right). The original algorithm introduces idiosyncratic artifacts, such as diagonal stripe patterns. It is recommended to zoom in for finer details, especially around the cheeks and forehead.

- We are surprised to find that the official implementation of LoRA directly interprets the memory content of the matrix $W \in \mathbb{R}^{C_1 k \times C_2 k}$ as $W \in \mathbb{R}^{C_1 \times C_2 kk}$ and perform convolution operation. We suspect this is a bug. Even though the matrix W is trainable, we suspect that such implementation has important consequences on performance, as now we are equivalently trying to find a low rank decomposition for a matrix that has no clear meaning, and might not have a low rank structure.

We compare ours with the official implementation of LoRA in Fig. 10, where the official implementation introduces checkerboard artifacts while our implementation is better at keeping the original image content.

C. Dataset Size

Using the same dataset in Mystyle [53], we further process the images following the preprocessing pipeline in EG3D [12]. We show the number of images in the reference and test sets in Tab. 3. In Sec. 4.4, we conduct ablation studies to investigate the effect of the size of the training set. When tuning on 100 images, if the reference set size is below 100, we use all the images in the reference set as the training set, such as 97 for *Dwayne Johnson* and 92 for *Xi Jinping*. Unless otherwise specified, we tune on 50 images for the majority of our personalization experiments.

D. AFHQv2 Cats

We also extend our personalization method to cat faces. Leveraging a photo album consisting of 22 in-the-wild images of one individual cat, we detect poses following [9] and apply the same procedure used for human faces to personalize the pretrained EG3D-AFHQ model, which was pretrained on a dataset of 15000 animal images, including 5000 cat images across different identities and breeds.

Celebrity	Reference set size	Test set size
Barack Obama	192	13
Dwayne Johnson	97	12
Joe Biden	200	13
Kamala Harris	110	7
Michelle Obama	279	9
Oprah Winfrey	135	9
Scarlett Johansson	260	13
Taylor Swift	158	9
Xi Jinping	92	15

Table 3. The sizes of the reference and test sets of our dataset.

The results, showcased in Fig. 11, demonstrate that our personalization technique significantly enhances the quality of the pretrained triplane representations for cat faces. This successful extension of our approach demonstrates the versatility and effectiveness of our method across different domains, paving the way for personalized 3D generative modeling of full human bodies, other animals, or objects.

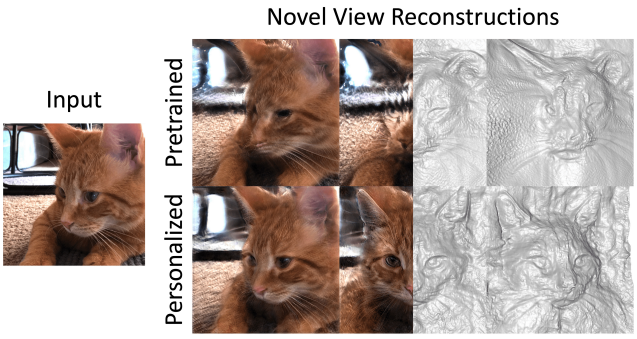

Figure 11. Comparison between a pre-trained model and a personalized model for inverting an in-the-wild cat photo.

E. Image Inversion without PTI

In Sec. 4.2, we perform image inversion tasks using PTI [62] to align with previous works [12, 71], where PTI requires changing the model weights for the best inversion quality. Nevertheless, we provide inversion results following Mystyle [53] where the model weights remain unchanged and only the latent code is re-projected into the convex hull. As shown in Fig. 12, although personalization helps maintain identity in the inversion tasks, it still lacks facial details for both full fine-tuning and ours, compared to PTI. Further works may design an encoder for EG3D inversion similar to TriPlaneNet [6].

F. ID_{sim} Curve Shape in Interpolation Tasks

Interestingly, unlike the previous findings of Mystyle [53], there is no significant difference in ID_{sim} scores between the interpolated latent codes and the anchors. The interpolation ID_{sim} curve in Mystyle follows a reserved U-shape, while our curve is flatter, as shown in Fig. 4. It is hypothesized that this lack of difference may be due to both our ID_{sim} metric design and EG3D's 3D advantage, where the extreme properties of anchors, such as pose, have a smaller impact on ID_{sim} compared to 2D-GANs.

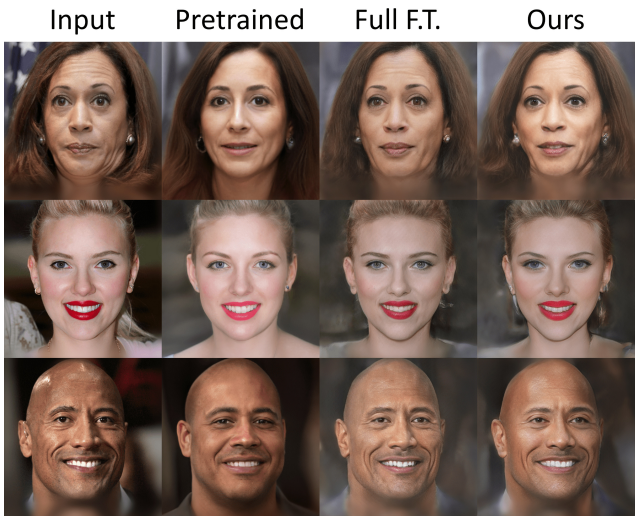

Figure 12. Image inversion results without optimizing network weights. F.T. indicates fine-tuning and ours is My3DGen.

G. Training Time

We perform our personalization experiments on 4 NVIDIA RTX A6000 GPUs. Our total training time is 5 hours with LoRA, compared to 6 hours without LoRA.

H. Failure Cases

My3DGen struggles to reconstruct objects that obscure the face, such as hands and phones, even with PTI. The cause for this is a deficiency in corresponding images of objects in the pre-training facial dataset, FFHQ. Further works may design an EG3D-specific encoder that can encode objects into the latent space similar to Live3DPortrait [71].

Figure 13. Cases where the inversion method fails to reconstruct objects.