# 3D Face Cartoonizer: Generating Personalized 3D Cartoon Faces from 2D Real Photos with a Hybrid Dataset - Supplementary Material -

Ming Guo[1], Shunfei Wang[2], Zhibo Wang[1], Ming Lu[3], Xiufen Cui[2], Xiao Ling[2], and Feng Xu[1]

[1] BNRist and school of software, Tsinghua University
[2] OPPO, China
[3] Intel Labs China

## 1 Statistics on Our Dataset

As mentioned in Section 3.1 in the main text, our dataset consists of various samples in different races and gender groups. Statistics on our dataset are given in Table 1.

## 2 Network Architecture

For the geometry module, as mentioned in Section 4.1 in the main paper, the two encoders $E_{recon}$ and $E_{gen}$ are fed in with 2D cartoon images and 2D real images respectively and output feature vectors containing a geometry part and a pose part. Both of them use ResNet-34 [3] as the backbone and are enhanced with an attention mechanism as shown in Figure 1a. The Channel Attention Layer and the Spatial Attention Layer [11] are inserted both before and after the "Resblocks" of ResNet-34. The result of $E_{recon}$ or $E_{gen}$ is a vector $\boldsymbol{f} \in \mathbb{R}^{100}$. The 8 independent discriminators $\mathcal{D}_i$, $i \in \{1, \cdots, 8\}$ try to distinguish the generated 3D face shapes from the artist-made ones by different local regions of a mesh. Assuming that there are $N$ vertex in the $ith$ local region, the discriminator $\mathcal{D}_i$ with architecture in Figure 1b will be fed in with a $N \times 3$ length vector representing the vertex positions of the local region and output a 1 element score measuring the probability to be "a real sample".

**Table 1.** Statistics of our hybrid cartoon dataset.

| | **Race** | | | | **Gender** | |
|---|---|---|---|---|---|---|
| | White | Asian | Black | Indian | Male | Female |
| Low-Q | 4391 | 1292 | 657 | 502 | 2928 | 3914 |
| High-Q | 33 | 44 | 35 | 18 | 62 | 68 |

For the texture module, as mentioned in Section 4.2 in the main paper, it has totally three encoders, namely $E_{img}$, $E_{normal}$ and $E_{tex}$. The $E_{img}$ and $E_{normal}$ share the same architecture as shown in Figure 1c. The $E_{tex}$ adopts the same architecture as pSp but removes the input layer. Please refer to [7] for more details.

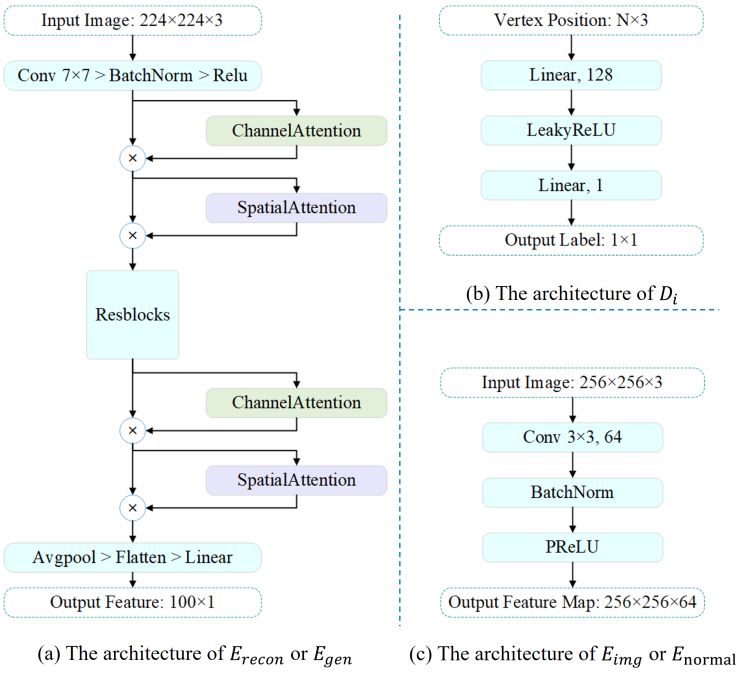


**Fig. 1.** The detailed network architecture of our 3D Face Cartoonizer.

## 3 Training Details

### 3.1 Geometry module

For the geometry module, we use Adam optimizer and adopt different learning rates for different sub-networks in different training stages. In the reconstruction stage, the learning rate for the encoder $E_{recon}$ is set to $1 \times 10^{-4}$. The architecture and learning rate setting of the decoder $D_{share}$ is the same as the decoder in [1]. For the first two fully connected layers and the last fully connected layer of $D_{share}$, we set the learning rates to $1 \times 10^{-5}$ and $1 \times 10^{-8}$ respectively. In the generation stage, we fix the parameters of the decoder $D_{share}$ and the learning rate for the encoder $E_{gen}$ is set to $1 \times 10^{-4}$. In the adversarial finetuning stage, as we only have 90 triplets for training, we freeze $E_{gen}$ except the last

**Fig. 2.** A gallery of our generated 3D cartoon faces.

"Resblock" to reduce the number of trainable parameters in order to avoid overfitting and adopt the same learning rate setting as the generation stage. We train the reconstruction stage for 250,000 iterations, the generation stage for 100,000 iterations, and the adversarial finetuning stage for 50,000 iterations with a mini-batch size of 128.

### 3.2 Texture module

**Training Losses** When training $E_{img}$, $E_{normal}$ and $E_{tex}$, we adopt the loss settings inspired by pSp [7].

$$L = w_2 L_2 + w_{LPIPS} L_{LPIPS} + w_{reg} L_{reg}, \tag{1}$$

We set the loss weights $w_2$ to 1, $w_{LPIPS}$ to 0.8 and $w_{reg}$ to 0.005 in all our experiments.

The $L_2$ measures the pixel level difference between the generated UV texture $\hat{T}_c$ and the ground truth UV texture $T_c$.

$$L_2 = \|\hat{T}_c - T_c\|_2. \tag{2}$$

The $L_{LPIPS}$ measures the perceptual similarities of $\hat{T}_c$ and $T_c$.

$$L_{LPIPS} = \|\mathcal{F}(\hat{T}_c) - \mathcal{F}(T_c)\|_2. \tag{3}$$

where $\mathcal{F}$ represents the operator to get the perceptual image patch score [15].

The $L_{reg}$ is the regularization loss. With the input image $I_r$ and the normal map of the 3D cartoon $N_c$,

$$L_{reg} = \|E_{tex}(E_{img}(I_r) + E_{normal}(N_c)) - \bar{w}\|_2. \tag{4}$$

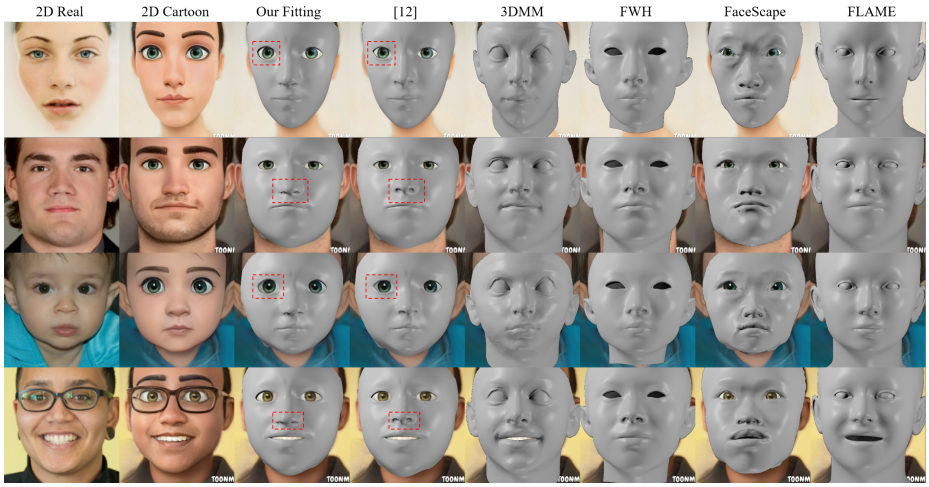


**Fig. 3.** Qualitative comparison of our low-quality data preparation method (cartoon face fitting) with the state-of-art face fitting methods.

where $\bar{w}$ is the average style vector of the pretrained StyleGAN[4].

**Training Process** For the texture module, we first pretrain StyleGAN using our texture dataset with the learning rate setting the same as [4] and the mini-batch size of 2 for 500,000 iterations. Then we train $E_{img}$, $E_{normal}$ and $E_{tex}$ while fixing the parameters of StyleGAN. Here, we use Ranger optimizer, a synergistic optimizer combining Rectified Adam [6] and LookAhead [14], with the learning rate of $1 \times 10^{-4}$. We train these encoders for another 500,000 iterations with a mini-batch size of 2.

## 4 Additional Results

To further demonstrate the generalization of our method, Figure 2 shows that our method can successfully generate high-quality cartoon faces from real facial images with different genders, ages and races.

### 4.1 Cartoon Image Fitting Comparison

To demonstrate the advantages of our low-quality data preparation method, we qualitatively compare our method on cartoon image fitting with several existing landmark fitting frameworks including 3DMM [16], FaceWareHouse [2], FaceScape [13], FLAME [5] and [12] in Figure 3. It shows that directly applying real face fitting methods to cartoon face fitting will lead to obvious errors. Although roughly fitting to the landmarks on the cartoon images, these methods fail to produce the cartoon style. [12] is designed to fit stylized facial images and

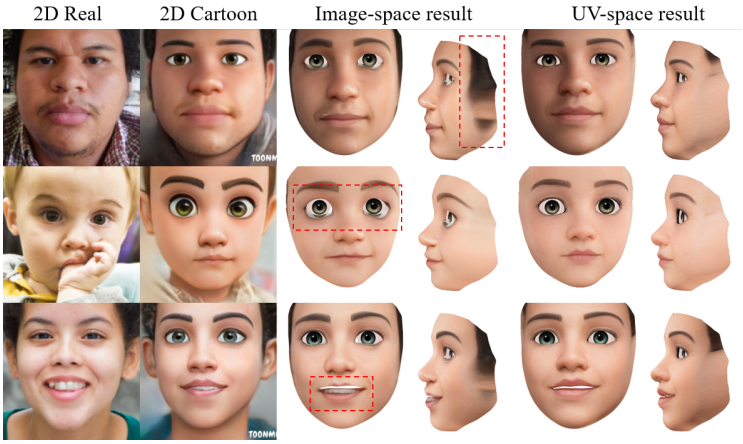


**Fig. 4.** Qualitative comparison between image-space texture generation and UV-space texture generation.

achieves much better fitting results compared with the above real faces fitting methods. Thanks to our extra landmarks around the eyelids and nostrils, our low-quality data preparation method better captures the cartoon styles, especially around eye contours and nostrils compared with [12].

### 4.2 Ablation Study of the Texture Module

We further evaluate the design of our cartoon texture synthesis module. Our method uses the network architecture in pSp [7] to synthesize a complete cartoon facial texture in UV space. We compare this design with a naive image-space solution. Specifically, we retrain the same network architecture on the proposed dataset to translate a real facial image into a cartoon facial image, and then map the cartoon image to the 3D cartoon face. As shown in the last two rows in Figure 4, this naive solution suffers from severe misalignment and generates obvious artifacts around eyes and mouths. In addition, since the image-space method can only synthesize the frontal color of the face, there is a lot of blurring and stretching of the texture in the side face (Figure 4, 1st row).

### 4.3 Applications

The proposed method can also be extended to various applications, *i.e.*, style editing and cartoon face animation.

**Style Editing**. We implement a simple but effective method for typical cartoon style editing of the generated 3D cartoon faces. Specifically, we use K-means algorithm to find several typical styles of 3D cartoon faces represented by the cluster centers of the low-quality data of our hybrid dataset. Our geometry module generates a cartoon face by first estimating a suitable deformation and then applying it to the mean cartoon face. To enable the generation of different styles,

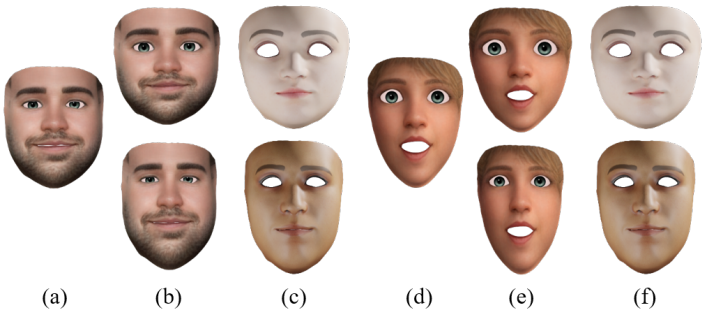


**Fig. 5.** Some examples of our 3D cartoon face editing. (a,d) the original generated 3D cartoon faces, (b,e) the edited results of style transfer, (c,f) the target styles.

**Fig. 6.** Cartoon face animation driven by real images.

we replace the mean cartoon face with the aforementioned different typical 3D cartoon faces, and thus the geometry module is able to flexibly produce 3D cartoon faces with the corresponding typical styles. Figure 5 presents some results of the style editing method. This style editing method can synthesize 3D cartoon faces with more different style types while preserving the identity and the expression of the original faces.

**Facial Animation**. We animate the generated 3D cartoon faces by constructing user-specific expression blendshapes. Specifically, we first ask artists to make a set of expression blenshapes for the mean face of our 3D cartoon dataset. The semantic definition of the expression blendshapes is the same as that in [10]. Then, we apply the deformation transfer algorithm [9] to transfer the artist-made expression blendshapes to fit the identity of the generated 3D cartoon face. Next, we design landmark-guided blendshape correction method using Laplacian deformation [8] to automatically fix the artifacts in the results of deformation transfer [9], *e.g.*, open eyes in the "eye closing" blendshape. Finally, we adopt a face tracking algorithm [2] to extract the facial rigid motion and expression

blendshape coefficients to drive the user-specific blendshapes, generating 3D cartoon faces under the target facial poses and expressions, as shown in Figure 6. We provide a supplementary video for presenting 3D cartoon face generation, editing with results of K-means and animation using a source video.

## 5 Limitation

To the best of our knowledge, our method is the first attempt to generate a 3D cartoon face from a real facial image. However, there are still several limitations. Even though the usage of our model does not require 2D cartoon faces, the training still relays on a 2D cartoon face generator, ToonMe. Also, due to the use of ToonMe, we can only handle almost frontal face images as ToonMe cannot handle faces with extreme poses well.

## References

1. Cai, H., Guo, Y., Peng, Z., Zhang, J.: Landmark detection and 3d face reconstruction for caricature using a nonlinear parametric model. Graphical Models **115**, 101103 (2021)
2. Cao, C., Weng, Y., Zhou, S., Tong, Y., Zhou, K.: Facewarehouse: A 3d facial expression database for visual computing. IEEE Transactions on Visualization and Computer Graphics **20**(3), 413–425 (2013)
3. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
4. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of StyleGAN. In: Proc. CVPR (2020)
5. Li, T., Bolkart, T., Black, M.J., Li, H., Romero, J.: Learning a model of facial shape and expression from 4d scans. ACM Trans. Graph. **36**(6), 194–1 (2017)
6. Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Han, J.: On the variance of the adaptive learning rate and beyond. arXiv preprint arXiv:1908.03265 (2019)
7. Richardson, E., Alaluf, Y., Patashnik, O., Nitzan, Y., Azar, Y., Shapiro, S., Cohen-Or, D.: Encoding in style: a stylegan encoder for image-to-image translation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2287–2296 (2021)
8. Sorkine, O., Cohen-Or, D., Lipman, Y., Alexa, M., Rössl, C., Seidel, H.P.: Laplacian surface editing. In: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing. pp. 175–184 (2004)
9. Sumner, R.W., Popović, J.: Deformation transfer for triangle meshes. ACM Transactions on graphics (TOG) **23**(3), 399–405 (2004)
10. Wang, Z., Ling, J., Feng, C., Lu, M., Xu, F.: Emotion-preserving blendshape update with real-time face tracking. IEEE Transactions on Visualization & Computer Graphics (01), 1–1 (2020)
11. Woo, S., Park, J., Lee, J.Y., Kweon, I.S.: Cbam: Convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018)

12. Wu, Q., Zhang, J., Lai, Y.K., Zheng, J., Cai, J.: Alive caricature from 2d to 3d. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7336–7345 (2018)
13. Yang, H., Zhu, H., Wang, Y., Huang, M., Shen, Q., Yang, R., Cao, X.: Facescape: a large-scale high quality 3d face dataset and detailed riggable 3d face prediction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 601–610 (2020)
14. Zhang, M.R., Lucas, J., Hinton, G., Ba, J.: Lookahead optimizer: k steps forward, 1 step back. arXiv preprint arXiv:1907.08610 (2019)
15. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 586–595 (2018)
16. Zhu, X., Lei, Z., Yan, J., Yi, D., Li, S.Z.: High-fidelity pose and expression normalization for face recognition in the wild. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 787–796 (2015)