High-fidelity 3D Face Generation from Natural Language Descriptions —— Supplementary Material ——

Menghua Wu, Hao Zhu✉, Linjia Huang, Yiyu Zhuang, Yuanxun Lu, Xun Cao

Nanjing University, Nanjing, China

A. Overview

In the supplementary material, we first explain the details of our DESCRIBE3D dataset in Section B, including all the annotated attributes and all descriptive options. Then more implementation details and details about the evaluation metrics are explained in Section C and Section D. Finally, more results regarding qualitative revaluation and comparison experiments are presented in Section E.

B. Dataset

B.1. 3D Models.

The 3D models in our DESCRIBE3D dataset are collected from HeadSpace [4] and FaceScape [10]. The collected model contains a 3D triangle mesh and a UV texture map, and is transformed into the topologically uniform models with Procrustes analysis [7] and non-rigid iterative closest point (NICP) algorithm [2]. The processed topologically uniform mesh model consists of 26369 vertices and 52536 triangle faces, attached with a UV texture map at a resolution of 1024×1024. The UV texture map is downsampled into 512×512 resolution to train the texture generation network.

B.2. Descriptive Texts

We established a questionnaire containing 25 single-choice questions and one short-answer question, and request a professional labeling institute to complete the labeling task with this questionnaire. The high-quality rendered images of the faces in the front view and side view are shown to the annotators. As shown in Figure 1, each question is about a facial attribute (left column), and each choice represents a description of this attribute (right column). Illustrations about these descriptions are inserted to help annotators understand the meaning of these descriptions. The short-answer question is "*Please observe the main view and side view of the face picture, describe the facial features of the face of each group of pictures in detail, as far as possible through your description, you can reproduce all the details of the face, and your facial feature description can distinguish the face and other faces.*".

For the convenience of the annotation, we visualize the models as shown in Figure 2. We first collect the 3D face models from FaceScape and HeadSpace datasets, normalize the scale and position, then render the models at the front and side views. The mean face and the extracted facial landmarks are visualized for judgment.The annotators are required to complete a questionnaire containing 25 multiple-choice questions and a free description according to the rendered images. The secondary labeling and sampling inspection are conducted to ensure the accuracy of the labeling.

B.3. Text Composer

According to the annotated attributes, we use pre-defined sentence patterns to generate text descriptions for each 3D face model. It is worth noting that we use the composed texts but not the answer texts of the short-answer question to train and test our model because the answer texts are not comprehensive enough.

The composed descriptive texts contain 7 sentences to describe eyebrows, eyes, nose, mouth, face shape, race, gender and age, and beard respectively. We pre-defined two sentence patterns. Taking eyes as an example, we use "His eyes are ..." or "He has ... eyes" to form our sentences. The order and the number of sentences can be adjusted to augment the texts for training, which will be detailed in Section C.1.

C. Implement Details

C.1. Training of Text Parser

To train the text parser, we generate an augmented dataset containing an input text and corresponding descriptive code. Since the input text can be generated from the descriptive code as explained in Section B, we first generate $1,000,000$ random and non-redundant descriptive codes by combining random 24 attributes (ears are not included), then generate corresponding descriptive sentences. Each

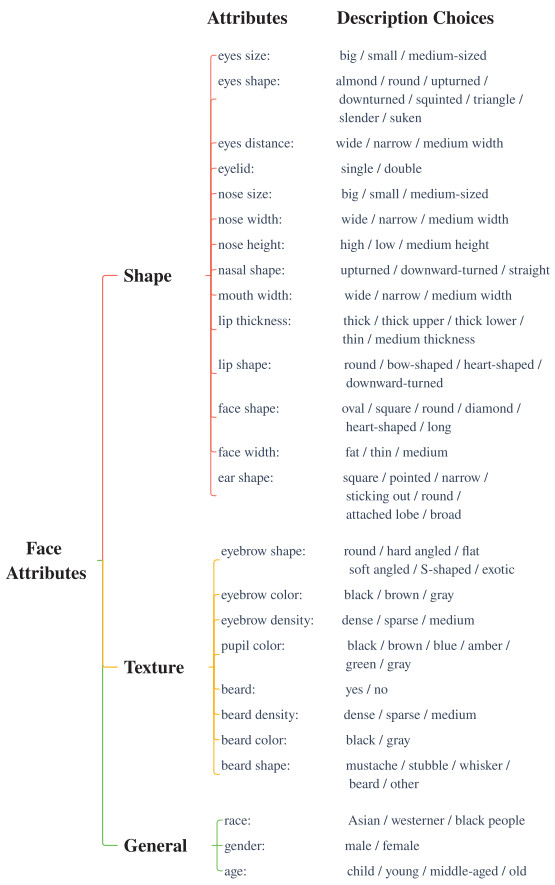

Figure 1. Attributes and descriptive choices in our annotation questionnaire.

Figure 2. Example of the visualization.

sentence contains 1 or more attributes of a specific region, such as "His face is fat" (1 attribute) and "His face is fat and round" (2 attributes). In each training iteration for a certain region, we randomly select $2-7$ sentences from the composed sentences to generate a training tuple, and the order of the sentences is shuffled.

C.2. Region-Specific Triplet Loss

We propose Region-Specific Triplet Loss (RST Loss) to train our shape generator, which is formulated as:

$$L_{RST} = \max(\|\hat{v_i} - v_i\|_1 - \|\hat{v_i} - v_i^*\|_1 + m_i, 0) \cdot \lambda_i, \tag{1}$$

where $\hat{v_i}$ is the predicted vertices of i-th region, v_i is the corresponding ground-truth, and v_i^* is its counter example. m_i and λ_i represent corresponding region margin and weight respectively.

Concretely, we choose four regions and randomly select one region for training in each iteration. We pre-establish a mapping list between positive and negative samples, for example, "big and high nose" is the negative sample for "small and low nose", and "fat and round face" is the negative sample for "long and thin face". In the training phase, we randomly select a sample as the positive example, then select the corresponding negative example according to the mapping list and train the model with the RST loss. In all our experiments, r is set to the average value of the RST loss between all positive and negative examples, and the threshold m_i is equal to r. The weight λ_i is set to eliminate the influence of different scales in different regions.

C.3. Post-process

We use MetaHuman Creator [1] to automatically register the generated 3D face into a riggable model, then manually add hair and skin texture. The 3D shape of the generated model and the post-processed model is highly consistent (mean error distance $< 0.3mm$). MetaHuman Creator cannot fit hair and skin textures, so hair and skin textures are manually assigned from the assets library, and we think this is the reason for visual inconsistencies between before and after the post-processed. The post-processed 3D faces can be directly used in rigging and animation pipelines.

D. Evaluation Metric

We use three metrics for quantitative evaluation: Chamfer Distance (CD), Complete Rate (CR), and Relative Face Recognition Rate (RFRR). We will explain how they are calculated below.

• **Chamfer Distance(CD)**: CD measures the overall error distance. Given the processed predicted mesh $\mathcal{M}_p$ and the ground-truth mesh $\mathcal{M}_g$, chamfer distance is formulated as:

$$\begin{aligned} CD(\mathcal{M}_p, \mathcal{M}_g) = & \frac{1}{N_p} \sum_{x \in \mathcal{M}_p}^{N_p} \sum_{y \in \mathcal{M}_g}^{N_g} \min \|x - y\|_2 + \\ & \frac{1}{N_g} \sum_{y \in \mathcal{M}_g}^{N_g} \sum_{x \in \mathcal{M}_p}^{N_p} \min \|x - y\|_2, \end{aligned} \tag{2}$$

where N_p, N_g are the numbers of the vertices of the predicted mesh and the ground-truth mesh respectively. Since Latent3D [3] only reconstructs the front face, we extract the front face from our predicted mesh and then calculate the CD.

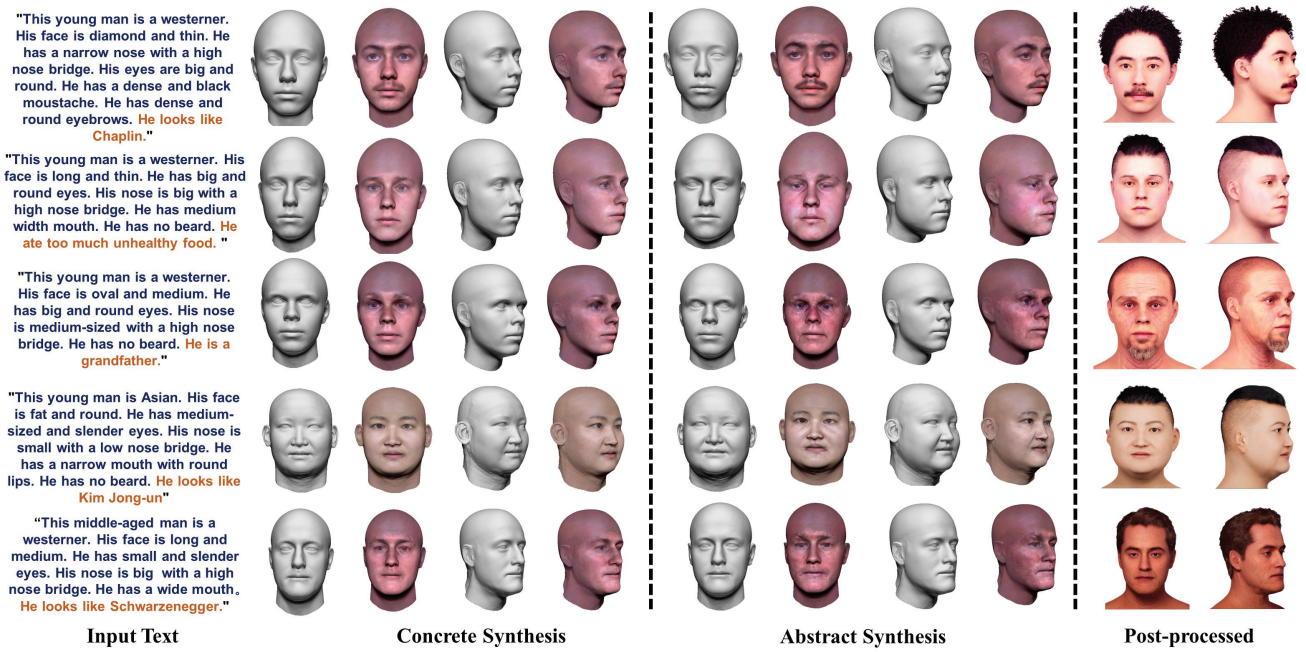

Figure 3. More qualitative results.

• **Complete Rate(CR)**: CR measures the integrity of the reconstruction results, which is formulated as:

$$\eta = \frac{P_1}{P_0}, \tag{3}$$

where P_1 is the number of points with a CD value less than $10mm$ and P_0 is the number of all points.

• **Relative Face Recognition Rate(RFRR)**: Since there is no general standard for measuring 3D face texture, we choose to use the Relative Face Recognition Rate(RFRR) similar to that in Anyface [8]. We render the predicted mesh and ground-truth mesh with the same camera parameters and use ArcFace [5] to extract features that represent facial identities. Then we calculate the cosine similarity and use it to measure the similarity between the ground-truth face and the predicted face.

We align the predicted 3D model to the ground-truth model before the computations of metrics. Specifically, we scale the predicted model to match the scale of the ground-truth model to have a consistent interpupillary distance. Then, Iterative Closest Point (ICP) algorithm is applied to align the predicted mesh to the ground-truth mesh.

E. More Results

E.1. More Visual Results and Comparisons

We show more qualitative results in Figure 3, which is the extension of Figure 6 in the main paper. We also show more comparison results in Figure 4 and Figure 5, corresponding to Figure 7 and Figure 8 in the main paper.

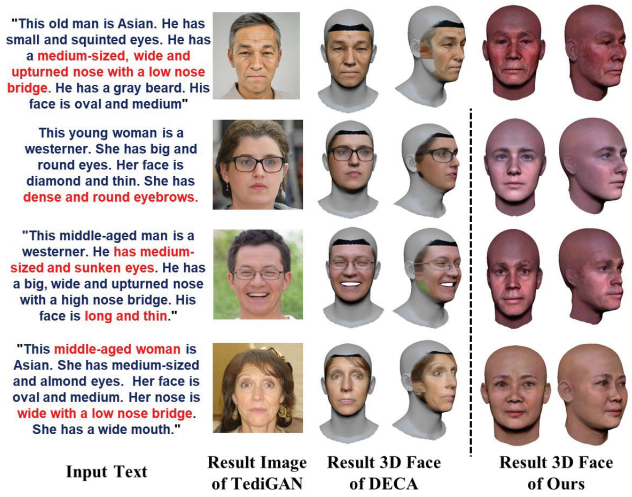

Figure 4. More comparisons to TediGAN [9]+DECA [6].

E.2. Diverse Results

We add an extra noise vector in the shape generation network. This design is based on the fact that a given descriptive text can correspond to many diverse 3D faces. Therefore, we add a noise vector as input to increase the diversity of the generation, and the effectiveness is verified in Fig 6. As we model the shape generation with the 3DMM regressing problem, the adversarial loss is not involved since it is not suitable for a parameter-regressing network.

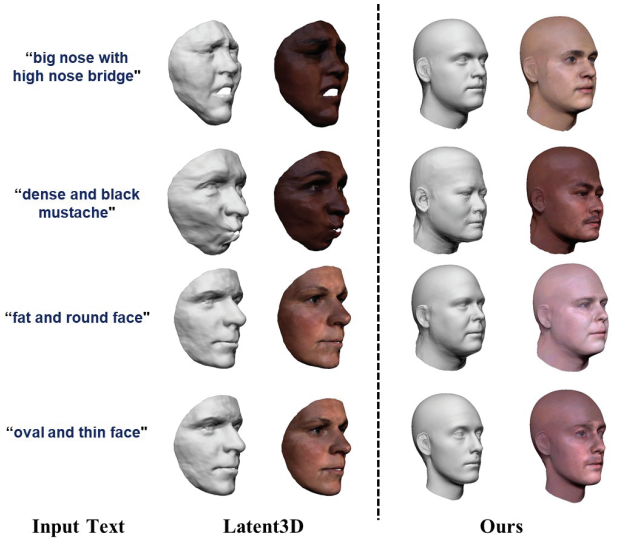

Figure 5. More comparisons to Latent3d.

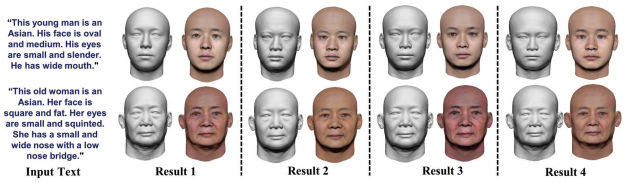

Figure 6. Diverse results generated from different noise.

E.3. Failure Cases

We found that our approach produced some failures, which can be categorized into two categories:

Casual descriptive texts. As shown in the first row of Figure 7, our model may fail with casual descriptive texts, which contains complex sentence patterns like "pointed nose embedded in ...", and figurative description like "big watery eyes". In this case, it is obvious that the result 3D face doesn't match the input description of "round face". We think the main reason is that our model is trained with relatively simple sentence patterns, and there is still room to improve the generalization of the text parser.

Special appearance. The shape and appearance of our results are strictly constrained in the S and T spaces that are built upon the training set, therefore, the abstract synthesis stage cannot generate a face with a non-human appearance. As shown in the second row of Figure 7, given "joker" as prompt, the abstract synthesis can generate a wide mouth which is a typical feature of the jokers in the films. However, the other features like the red nose and exaggerated clown makeup can not be generated, since these features are not covered in our S and T space.

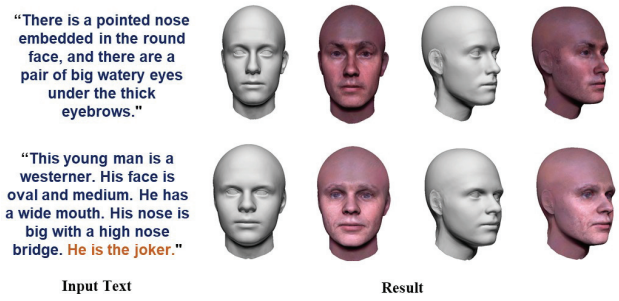

Figure 7. Failure cases.

References

[1] Metahuman creator. https://www.unrealengine.com/en-US/metahuman. Accessed: 2022-11-11. 2

[2] Brian Amberg, Sami Romdhani, and Thomas Vetter. Optimal step nonrigid icp algorithms for surface registration. In *CVPR*, pages 1–8. IEEE, 2007. 1

[3] Zehranaz Canfes, M Furkan Atasoy, Alara Dirik, and Pinar Yanardag. Text and image guided 3d avatar generation and manipulation. *arXiv preprint arXiv:2202.06079*, 2022. 2

[4] Hang Dai, Nick Pears, William Smith, and Christian Duncan. Statistical modeling of craniofacial shape and texture. *IJCV*, 128(2):547–571, 2020. 1

[5] Jiankang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. Arcface: Additive angular margin loss for deep face recognition. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4690–4699, 2019. 3

[6] Yao Feng, Haiwen Feng, Michael J Black, and Timo Bolkart. Learning an animatable detailed 3d face model from in-the-wild images. *TOG*, 40(4):1–13, 2021. 3

[7] John C Gower. Generalized procrustes analysis. *Psychometrika*, 40(1):33–51, 1975. 1

[8] Jianxin Sun, Qiyao Deng, Qi Li, Muyi Sun, Min Ren, and Zhenan Sun. Anyface: Free-style text-to-face synthesis and manipulation. In *CVPR*, pages 18687–18696, 2022. 3

[9] Weihao Xia, Yujiu Yang, Jing-Hao Xue, and Baoyuan Wu. Tedigan: Text-guided diverse face image generation and manipulation. In *CVPR*, pages 2256–2265, 2021. 3

[10] Haotian Yang, Hao Zhu, Yanru Wang, Mingkai Huang, Qiu Shen, Ruigang Yang, and Xun Cao. Facescape: a large-scale high quality 3d face dataset and detailed riggable 3d face prediction. In *CVPR*, pages 601–610, 2020. 1