3D Guided Fine-Grained Face Manipulation

Zhenglin Geng[1,2], Chen Cao[2], and Sergey Tulyakov[2]

[1]Stanford University, [2]Snap Inc.

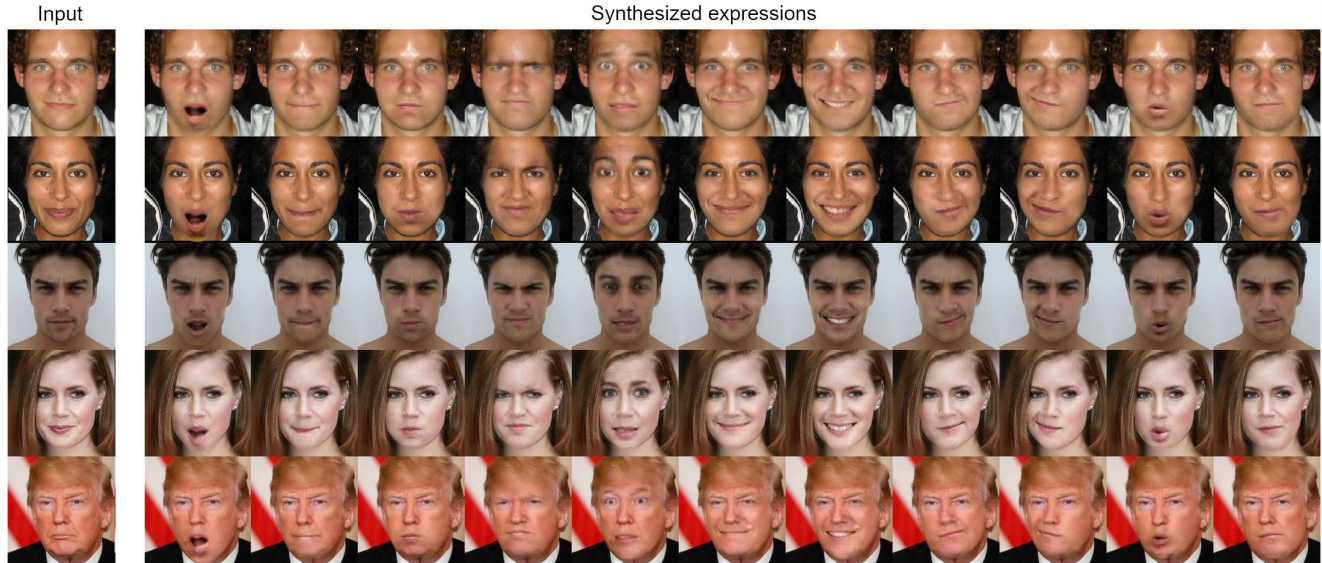zhenglin@stanford.edu,{chen.cao,stulyakov}@snapchat.com

Figure 1: **Qualitative samples.** Given an image, our method can generate multiple realistic expressions of the same subject.

Abstract

We present a method for fine-grained face manipulation. Given a face image with an arbitrary expression, our method can synthesize another arbitrary expression by the same person. This is achieved by first fitting a 3D face model and then disentangling the face into a texture and a shape. We then learn different networks in these two spaces. In the texture space, we use a conditional generative network to change the appearance, and carefully design input formats and loss functions to achieve the best results. In the shape space, we use a fully connected network to predict the accurate shapes and use the available depth data for supervision. Both networks are conditioned on expression coefficients rather than discrete labels, allowing us to generate an unlimited amount of expressions. We show the superiority of this disentangling approach through both quantitative and qualitative studies. In a user study, our method is preferred in 85% of cases when compared to the most recent work. When compared to the ground truth, annotators cannot reliably distinguish between our synthesized images and real images, preferring our method in 53% of the cases.

1. Introduction

Face manipulation, a problem involving changing the facial expressions in images enables many creative applications. Until very recently, this problem was mainly addressed from a graphical perspective in which a 3D Morphable Model (3DMM) was first fitted to the image and then re-rendered with a different facial expression. Such techniques jointly model both the shape and the appearance and are typically trained using spatially aligned 3D scans of people [1]. A desired facial expression can then be generated by combining graphical primitives called blendshapes [16]. The blendshapes often correspond to the Facial Action Coding System (FACS) [7] which defines a set of anatomically related muscle activations. Unfortunately, due to the Gaussian assumption, 3DMMs often produce blurry shapes and appearances, preventing realistic face rendering.

Deep generative techniques offer a different way of solving the face manipulation problem. In contrast to 3DMMs, they learn an internal representation that jointly models the shape and the appearance of the faces. Manipulation is then performed by conditioning the decoder on expression labels

[6, 14] or latent vectors [17]. This solution is sub-optimal in several respects. First, neural networks have been recently shown to have difficulties in generating simple geometric transformations [18], whereas face manipulation involves many such transformations, such as mouths opening, eyes closing and other transformations. Second, their models require many examples of such transformations along with their intensities at the training time, which becomes even more problematic for less common expressions such as sad-smile or negative-surprise. Third, each model supports only a small set of manipulation operations, not allowing fine-grained 3D manipulation.

In this paper we present a novel method that combines 3DMMs and deep generative techniques in a single framework for fine-grained face manipulation. Randomly selected qualitative samples produced by our method are given in Fig. 1. Given a face image, we first fit a 3D face model on the image to obtain the texture and the shape. The shape is further represented as identity and expression coefficients using a bilinear model [5]. This way we disentangle the shape and the texture spaces and use separate branches in our pipeline to apply transformations in these spaces.

The texture branch consists of a convolutional neural network and assumes the texture and the desired expression as inputs, producing a new texture which corresponds to the desired expression. Due to the difficulties of the convolutional networks in generating geometric transformations, we propose conditioning the texture branch on the UV maps that describes target geometry information instead of directly concatenating the labels as in [6] or coefficients as in [21]. To better preserve texture-expression consistency and the identities in the generated images, we design corresponding loss functions for improved results.

The shape branch is implemented using a fully connected neural network taking the identity and the expression coefficients as inputs and outputting *shape deformation* necessary to accurately match the desired expression. Notably, a common problem in fitting a morphable model to the face is its inability to fully capture the face shape given only a 2D RGB input image sparsely labeled with 2D landmarks. This is often called face shape hallucination [25]. At training time, to improve 3D reconstruction, we additionally supervise the shape branch using the available depth data in the FaceWarehouse dataset [5].

The proposed approach has a number of benefits. First, we disentangle the texture and shape shapes to make it easier to learn for each branch. In the texture space, faces tend to be more similar despite significant variance in the image space caused by different poses and expressions. Therefore, the texture branch only focuses on the appearance details such as wrinkles, shadows and shading. Similarly, the shape branch focuses on the geometric details only. Second, since we represent expressions as a combination of Face

Action Unit coefficients [7], rather than discrete labels, our approach can generate infinite number of target expressions. Third, we further distinguish identity and expression coefficients, to better preserve subject-specific features by only changing the expression components in the shape space.

We compare the proposed method to the most recent face manipulation methods [6, 21] and show that our approach is superior in all the experiments. In the user studies that we conducted, the presented method is preferred more than 85% of the time when compared with the existing works. When compared to the ground truth testing images, our method is preferred in 53% of cases, supporting that it is difficult for a human to distinguish real images from those generated by our method.

2. Related Work

We review relevant geometry based methods and deep generative methods for face manipulation.

Geometry-based methods. A pioneering work of Blanz and Vetter [1] presented the first public 3D Morphable Model (3DMM). They densely captured surface geometry and color data of 200 identities and created a linear model to represent the face variations of different subjects using principal component analysis (PCA). Vlasic *et al*. [27] proposed a multilinear model of facial expressions for tracking and re-targeting. Cao *et al*. [5] proposed FaceWarehouse, an extensive facial expression database, which contains 47 different facial expressions for each of the 150 subjects. This dataset later became one of the most adopted datasets for 3D face fitting and animation [4, 33].

In [5], they first fitted a 3D face shape to match the input image, and then changed the expression coefficients to perform animation by warping the image to a new expression. Thies *et al*. [23] presented Face2Face for real-time video-to-video facial expression re-targeting. They first fit a 3DMM together with lighting parameters and re-render it in the target video. Although these geometry-based methods produce convincing results of large-scale motions, they are unable to model parts not existing in the source image, such as teeth when the mouth is closed, and resort to rendering such parts using conventional graphics approaches. Therefore, these methods often fail to achieve realistic results, as humans are especially sensitive to non-realistic artifacts in faces.

Deep generative methods. Face manipulation can be viewed as the unpaired image-to-image translation problem [18, 32] . Until very recently, one had to train a separate model, attribute-by-attribute to perform face manipulation [17]. Lample *et al*. [14] proposed to additionally control the intensity of the attribute. Their work can change two attributes at the same time, but only at the cost of reduced image quality. Choi *et al*. [6] used conditional image-to-image translation to allow multiple attributes to be trained
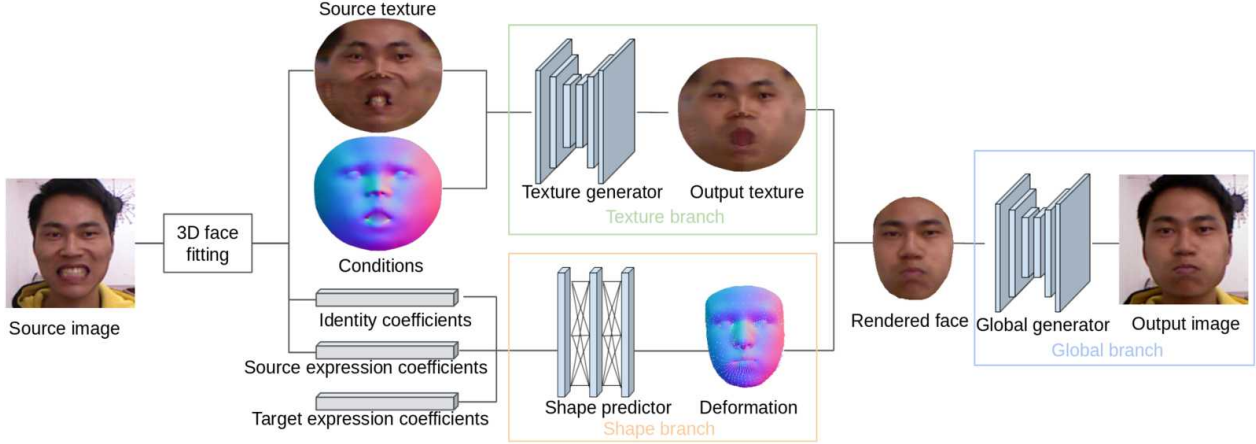
Figure 2: **Overview of our pipeline.** We first fit a 3DMM to the input and decouple it into the texture and shape coefficients. The texture branch assumes the source texture and the spatial representation of the target expression to produce the output shape. The shape branch uses the 3DDM coefficients to output a shape deformation. Finally, the global branch blends the two outputs in the image space.

together in an unsupervised fashion. These attributes can include gender, age, hair color, expression and so on. Despite the impressive results, their approach is still limited to a finite number of attributes, preventing fine-grained manipulation. Several video generation methods for face animation were proposed. Given a face image, such methods perform video prediction [26] or motion transfer [22, 28] to manipulate faces. Recently, Pumarola *et al.* [21] presented a work performing anatomically-aware face animation. Similarly to us, they animate faces according to Facial Action Units.

The method presented in this paper is different than geometry-based and deep generative methods in that it combines the benefits of both lines of work in a single end-to-end trainable framework. As opposed to purely 3DMM-based methods and similarly to deep generative works, our framework features high quality face texture synthesis. In contrast to deep generative works, and similarly to 3DMMs-based methods, our approach can generate arbitrary number of facial expressions. A key difference with Pumarola *et al.* [21] is that we learn to explicitly disentangle shape and appearance into different branches. This enables learning a rich face prior from our shape branch, and allows the texture branch to focus on synthesizing realistic images.

3. Method

Our pipeline is shown in Fig. 2. The approach requires a face image and the desired expression encoded by coefficients. We first fit the 3D face shape and camera projection matrix from the image, with which we extract textures (Sec. 3.1). Then, we input the texture and the target expression to the texture branch and generate the target texture containing the details of the desired expression (Sec. 3.2). As the 3DDM-based shape representations are often inaccu-

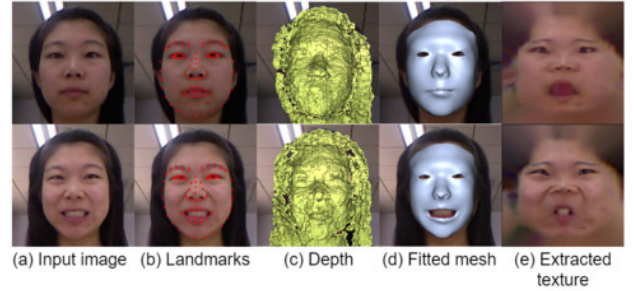(a) Input image (b) Landmarks (c) Depth (d) Fitted mesh (e) Extracted texture

Figure 3: Examples of fitting a 3DMM to an RGB-D image.

rate, we use a fully connected network in the shape branch to predict a more accurate shape for improved synthesis quality (Sec. 3.3). The predicted texture and shape are then combined and rendered to obtain a target image. We then use the global branch network on the target image to further improve the quality (Sec. 3.4).

3.1. 3D Face Fitting

Face fitting is the process of estimating the 3D face shape and the camera projection matrix given an input face image. Following [5], we represent the 3D face shape using a bilinear model as:

$$\mathbf{S} = C_r \times_2 \mathbf{a} \times_3 \mathbf{e}, \quad (1)$$

where $\mathbf{S} \in \mathbb{R}^{3N}$ is the face shape, N is the number of vertices, $C_r \in \mathbb{R}^{3N \times N_a \times N_e}$ is the weight tensor, $\mathbf{a} \in \mathbb{R}^{N_a}$ are the identity coefficients, $\mathbf{e} \in \mathbb{R}^{N_e}$ are the expression coefficients, $\times_i$ is the tensor contraction operation along the ith mode of the bilinear model. In our experiments, $N_e = 46, N_a = 50$ and $N = 1220$.

Given a face image (Fig. 3a), we first detect the 96 2D landmarks using [12] (Fig. 3b). Then, we jointly estimate
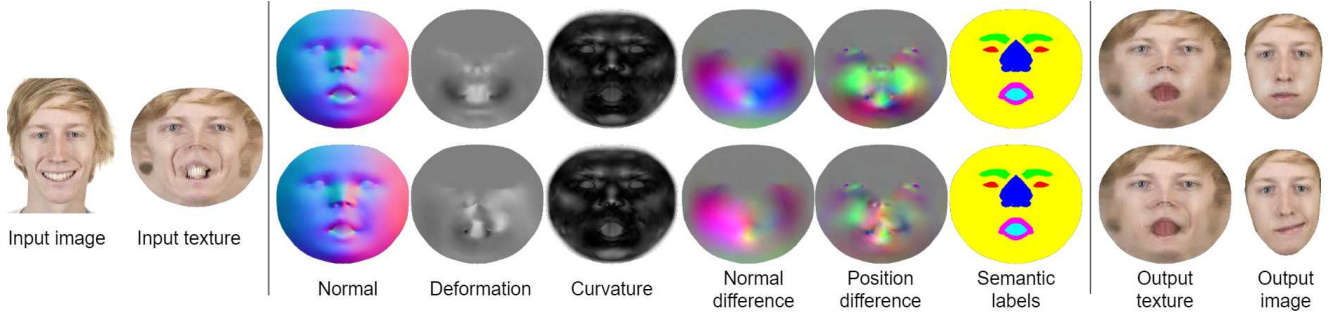
Figure 4: **Texture branch inputs and outputs.** The input image is mapped to the texture space. The texture branch uses important geometry information represented spatially using UV-maps. The output of the texture branch is a texture containing the desired target expression.

the camera projection transformation $\mathbf{M} : \mathbb{R}^3 \rightarrow \mathbb{R}^2$, as well as identity and expression coefficients, by minimizing the L2 distance between the projected landmarks and detected landmarks. Note that we fix the identity coefficients for multiple images of the same person during optimization.

The inaccuracy in the fitting process causes the extracted textures to be misaligned and thus introduces additional variance for neural network to learn. To tackle this, we make use of the depth data when it is available. For the input image with a depth map (Fig. 3c), we minimize the L2 distance between the shape vertices and its closest 3D depth points and then refine the shape using [8] (Fig. 3d). When the depth is not available, we deform the shape to further reduce the landmark errors as in [5].

We define a 2D UV coordinate for each 3D shape vertex, consistent across the dataset. The textures are extracted with the UV coordinates, camera projection and fitted 3D shape using the standard rasterization pipeline. (Fig. 3e).

3.2. Texture Branch

Our texture branch learns a function $G(\mathbf{T}^{\mathrm{src}}, \mathbf{e}^{\mathrm{src}}, \mathbf{e}^{\mathrm{tgt}})$ which transfers a texture $\mathbf{T}^{\mathrm{src}}$ extracted from the source image with the expression $\mathbf{e}^{\mathrm{src}}$, to texture $\mathbf{T}^{\mathrm{tgt}}$, containing the target expression $\mathbf{e}^{\mathrm{tgt}}$. Inspired by recent advances in image-to-image translation [9, 32], we adopt conditional generative adversarial networks (cGAN) to learn the function G.

Input format. Typically the generator G is modeled as a convolutional neural network. In our case, the generator needs to take both the texture image $\mathbf{T}$ and the expression coefficients $\mathbf{e}$ as input. A straightforward approach to combine these different formats is to concatenate each element of $\mathbf{e}$ as a separate feature map to the input image $\mathbf{T}$ as in [6, 21]. We argue that converting the geometry information of $\mathbf{e}$ into a spatial representation, such as a UV-map, helps better utilize local convolutional operations learned by the texture branch.

In our implementation, this information includes object space normals, deformation, curvature, position difference,

normal difference and semantic labels. We show examples in Fig. 4. Normal determines the local surface orientation which is considered important in shading. Deformation is determined by the ratio of the one-ring area near each vertex in the target and neutral expressions, where a small deformation value means compression and can be associated with wrinkles. Curvature differentiates bumped regions from flat regions. Position and normal differences imply similarities between source and target expressions near each vertex, indicating the likelihood of the output pixel resembling the input pixel at the same location. Furthermore, to address the translational equivariance issue of convolutions [18], semantic labels are used to indicate different facial components which should be synthesized differently. These labels include eyes, eyebrows, nose, lips and inner mouth and others. As all the shapes have the fixed layout in the UV space, we manually define the labels on the 3D mesh and rasterize them to get the semantic map. We then use this semantic map for all the samples. We evaluate the effectiveness of our input format in Sec. 5.1.

Loss functions. Let $\mathbf{T}^{\mathrm{real}}_{i,p}, \mathbf{T}^{\mathrm{fake}}_{i,p}$ be the real and fake textures of identity $\mathbf{a}_i$ under the expression $\mathbf{e}_p$. We design three discriminator terms to improve the synthesis quality:

- D_{real} is the standard discriminator to distinguish between real textures $\mathbf{T}^{\mathrm{real}}_{i,p}$ and synthesized fake textures $\mathbf{T}^{\mathrm{fake}}_{i,p}$.

- D_{pair} is used to ensure pair consistency between the texture and the expression coefficients as [2]. Our discriminator D_{pair} learns to differentiate matched pairs of real texture and expressions $(\mathbf{T}^{\mathrm{real}}_{i,p}, \mathbf{e}_p)$ from matched pairs of fake texture and expressions $(\mathbf{T}^{\mathrm{fake}}_{i,p}, \mathbf{e}_p)$ and mismatched pairs of real texture and expressions $(\mathbf{T}^{\mathrm{real}}_{i,p}, \mathbf{e}_r)$, where $\mathbf{e}_r$ is a random expression.

- D_{iden} is designed to preserve identities. It is used to differentiate real textures with the same identity $(\mathbf{T}^{\mathrm{real}}_{i,p}, \mathbf{T}^{\mathrm{real}}_{i,q})$, from real and fake textures with the
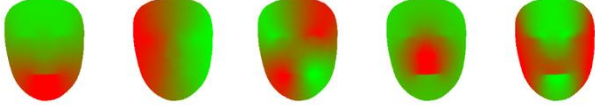
Figure 5: The first 5 eigenvectors of an average face model. In each eigenvector, the vertices with similar colors have similar deformation.

Figure 6: Demonstration of the global branch.

same identity $(\mathbf{T}_{i,p}^{\text{real}}, \mathbf{T}_{i,q}^{\text{fake}})$, and real textures with different identities $(\mathbf{T}_{i,p}^{\text{real}}, \mathbf{T}_{j,q}^{\text{real}})$, where p, q index random expressions and i, j index different identities.

We use LSGAN [20] to calculate the respective loss terms $\mathcal{L}_{\text{real}}, \mathcal{L}_{\text{pair}}, \mathcal{L}_{\text{iden}}$. The combined GAN objective writes as:

$$\mathcal{L}_{\text{GAN}} = \mathcal{L}_{\text{real}} + \mathcal{L}_{\text{pair}} + \mathcal{L}_{\text{iden}}. \tag{2}$$

The objective for our discriminators is

$$\max_{D_{\text{real}}, D_{\text{pair}}, D_{\text{iden}}} \mathcal{L}_{\text{GAN}}. \tag{3}$$

The generator G minimizes $\mathcal{L}_{\text{GAN}}$ and is supervised by $\mathcal{L}_1$ loss and perceptual loss [10] $\mathcal{L}_{\text{perc}}$. Though the original perceptual loss is proposed in image space, we find it effective in texture space as well (Sec. 5.1). Thus our generator objective is

$$\min_{G} \mathcal{L}_{\text{GAN}} + \lambda_{\mathcal{L}_1} \mathcal{L}_1 + \lambda_{\text{perc}} \mathcal{L}_{\text{perc}}. \tag{4}$$

In our experiments, we empirically set $\lambda_{\mathcal{L}_1} = 10, \lambda_{\text{perc}} = 10$. For more details of our loss terms, please see our supplementary materials.

3.3. Shape Branch

The 3D face shape $\mathbf{S}$ is a non-linear function of the expression coefficients due to the complex interaction of muscles, flesh and bones. Previous works [4, 5] model this complex interaction linearly. Although this method is simple and widely adopted, we argue that these limited expression models can only represent the large-scale motion, and struggle to capture the fine-grained details.

To further increase the accuracy of the shape branch, we deform the face shape either through depth or landmarks as mentioned in Sec 3.1. To fully capture these geometric details, we formulate the shape function as a linear part using Eqn. 1 and and a non-linear part $D(\mathbf{a}, \mathbf{e}^{\text{src}}, \mathbf{e}^{\text{tgt}})$, which is an additional deformation field. Similarly to [24], we train a neural network to learn only the non-linear deformation D to reduce variance.

The output of $D(\mathbf{a}, \mathbf{e}^{\text{src}}, \mathbf{e}^{\text{tgt}})$ represents the per vertext displacement vectors. These vectors can be very high dimensional. To reduce dimensionality, we model the displacements with a spectral representation as in [3]. More
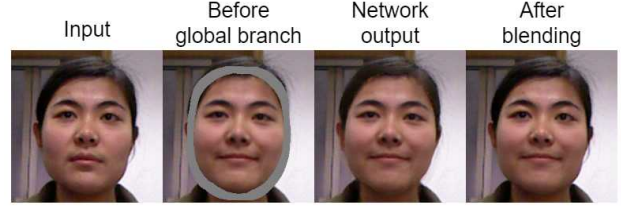
specifically, we compute eigenvectors of the k smallest non-zero eigenvalues of the graph Laplacian matrix of a generic 3D face shape [15] and use them as the basis of vertex displacements. We use a fully connected network with 2 hidden layers to predict the basis coefficients. Fig. 5 shows the first 5 eigenvectors. In our experiments we set $k = 100$.

3.4. Global Branch

We use the predicted texture $\hat{\mathbf{T}}$ and shape $\hat{\mathbf{S}}$ to render the predicted face on the image. The goal of the global branch is to blend this face into the background seamlessly. We show the process in Fig. 6. We first make the artificial margin between the rendered face and the background and train a network to hallucinate in between. The margin is computed using a dilation approach with kernel size 12. To fill in the margin, one could use image inpainting techniques [29, 30]. We have a simpler problem since the input image is usually similar to the background image. Therefore, we use the global network that takes the input image, the rendered face and the region outside of the margin as input. The network then learns to blend the generated face and the background together. Occasionally this still produces artifacts near the boundary. Therefore at test time, we apply image blending with the input image as a post-processing step. We describe this step in more details in the supplemental materials.

4. Implementation Details

Datasets. Our datasets include FaceWarehouse [5] and Chicago Face Dataset (CFD) [19]. For the training set, we use 493 identities from FaceWarehouse, each with at least 20 different expressions and 152 identities from CFD, each with at least 5 expressions. Among this data, 140 identities in Facewarehouse have depth. For the test set, we use 87 identities from Facewarehouse and 5 identities from CFD. Our datasets span different genders and skin colors. We use 256×256 for our image and texture resolution. To further increase the resolution multistage generative models can be employed [11, 31].

Network architecture. The texture and global branch generators adopt pix2pix [9] architecture with attention maps [21]. We change the transposed convolutions to upsampling layers followed by 3x3 convolutions. Similarly our discriminators adopt the pix2pix discriminator architectures. See the supplement for more details.
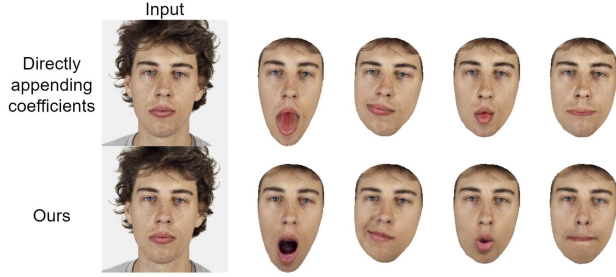
Figure 7: Qualitative samples of different inputs of the texture branch evaluated on rare facial expressions. Note that the proposed approach generalizes better compared to the standard method of directly appending expression coefficients.

Training. We use Adam [13] optimizer with a learning rate of 0.0001, $\beta_1 = 0.5$, $\beta_2 = 0.9$. We first train the texture branch and the shape branch. Then we fix their weights and train the global branch. We use a single NVIDIA Tesla V100 GPU and we train for 5 days to get the best results.

5. Experiments

In this section, we first conduct an ablation study to evaluate the design choices in our system. Next, we compare our approach with other approaches both qualitatively and quantitatively. Finally, we show additional qualitative results.

5.1. Ablation Study

Texture branch input format. We compare our proposed input format with directly concatenating expression coefficients to the input of the neural network as in [6, 21]. We show that our approach generalizes better by transferring an image from CFD to a rare expression that CFD rarely covers in Fig. 7. The model (top row) which appends expression coefficients directly as input fails to generate the correct appearance for regions like the inner mouth, cheeks near mouth corners and lips. This occurs since the generator has rarely seen the combination of this face skin color with these specific coefficients in the training dataset. The proposed approach, which conditions on the texture branch on the spatial representation of geometry information, generalizes better. We believe our approach better uses the local convoluational structure of the neural network.

Texture branch loss functions. We apply ablation study on our loss terms. We show results in Fig. 8. We first remove $\mathcal{L}_{\text{pair}}$ and $\mathcal{L}_{\text{iden}}$. The $\mathcal{L}_{\text{pair}}$ term is designed to enforce texture-expression pair consistency. We can observe that expression specific features such as wrinkles are less observable after the removal. $\mathcal{L}_{\text{iden}}$ is designed to preserve identities, which helps direct appearance details from the source texture to the synthesized texture. We can see that
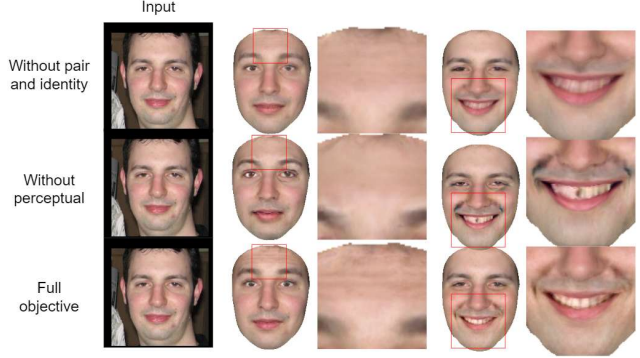

Figure 8: Qualitative evaluation of different loss terms. The model trained with the full objects generates images with higher fidelity.

Table 1: RMSE of vertices with/without shape branch. Lower number is better.

	RMSE (mm)
Without shape branch	2.2158
With shape branch	**1.7619**

the synthesized images, especially near the teeth region, are more blurry after the removal. We then remove the perceptual loss while keeping the rest unchanged. Similar to $\mathcal{L}_1$ loss [9], the perceptual loss helps avoid artifacts near the mouth region. It also allows the network to capture more subtle details such as wrinkles on the forehead compared to $\mathcal{L}_1$ loss alone.

Shape branch. We demonstrate that our shape branch generates more realistic shapes than the linear blendshapes both quantitatively and qualitatively. We first compute the root mean square error (RMSE) between the generated face mesh and our ground truth fitted face mesh in Table 1. After being deformed by our shape branch, the predicted mesh is closer to the ground truth. We also show an example demonstrating the change of the mesh in Fig. 9. Without the shape branch, the fitted linear blendshapes tend to open the jaw more widely, which looks less natural, while our shape branch learns to close the jaw, such that the shape gets closer to the ground truth.

Note that despite the obvious benefits of the texture shape decoupling, our carefully designed input format, loss functions and shape branch are necessary for best results.

5.2. Comparisons

We compare our face manipulation results to the direct texture mapping approach [5], StarGAN [6] and GANimation [21]. The method in [5] is a linear model combined with a computer graphics rendering approach, which also separates the texture and the shape but does not alter the
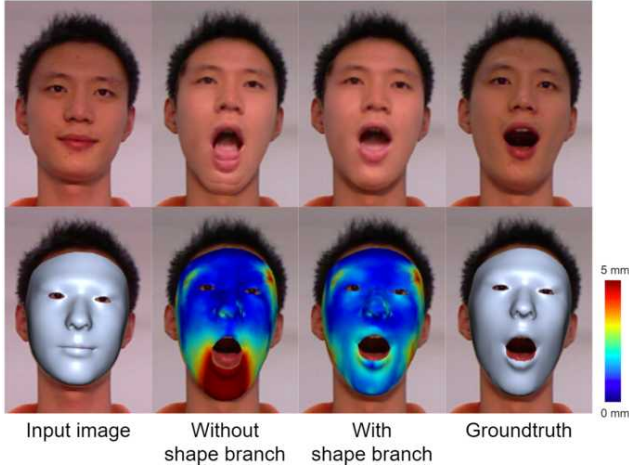
Figure 9: Qualitative evaluation of the shape branch. The top row shows the input image, the generated images without and with the shape branch and the ground truth target image. The bottom row shows the fitted mesh with the color coded depth error in mm. Lower depth error makes the generated images more realistic.

Table 2: Quantitative comparisons and user studies results of different methods. We report the Average Content Distance (ACD, lower is better) and the user preference score (higher is beter). Best results in bold.

Methods	ACD	User Preference Ours / Others
Texture mapping [5]	0.6194	**69.8** / 30.2
StarGAN [6]	0.5981	**86.8** / 13.2
GANimation [21]	0.5595	**86.2** / 13.8
Ours	**0.5107**	N/A
Ground truth	0.4608	**53.4** / 46.6

texture. The latter two train on image space only concatenating the attributes or action units directly to the input. To evaluate the effectiveness of different methods at handling wide ranges of extreme expressions, we choose FaceWarehouse [5] as our training set because it contains many challenging expressions other datasets do not normally cover. We use the 87 identities in our test set as mentioned in Sec. 4 and we do not include CFD for easier comparisons. We trained StarGAN using 20 different expressions as attributes. We implemented GANimation with all the attention mechanisms and loss terms, except that we replaced the regressor with a classifer in their discriminator, which tends to give better results on our dataset. For all the comparison experiments, we transfer neutral expressions to different expressions. We use the real captured data from [5] as the ground truth.

Qualitative study. We show several examples in Fig. 10. Direct texture mapping is not able to generate wrinkles in the smiling expression, teeth in mouth opening expression or correct shading details in mouth blowing expression. For StarGAN and GANimation, we observe that they tend to produce more artifacts in expressions that have larger scale facial movements like mouth opening and mouth sloping. We hypothesize that this is because the competing approaches need to learn a complex model with all the rigid pose, shape and appearance variance together, while our fitting process and shape branch take the first two away, leaving a simpler function for the texture branch to learn. We also find that GANimation sometimes leaves the details from the input image in the output. Interested readers can

magnify the lips region on the 4th column and eyebrows region in the 5th column to see the artifacts. We hypothesize that this is a problem caused by the attention mechanism in the image space. Our approach has a fixed texture layout and thus does not have this problem.

Note that our synthesized images have different camera poses than the ground truth. This is because FaceWarehouse is captured with different head poses and our images are cropped differently based on the face sizes. Also note that the synthesized images look different from the ground truth. This is because there are numerous ways that a person can perform an expression and our method only generates a possible realization of that expression.

Quantitative study. We adopt Average Content Distance (ACD) from [26] to evaluate how well identities are preserved using different methods. We extract feature vectors from each synthesized image and compute the $\mathcal{L}_2$ distance to the feature vector of the input image. We show the results in Table 2. Our method gives the best results besides ground truth. Note that we do not optimize with respect to any pretrained face recognition networks at training time. We attribute our lower ACD to our disentanglement representation of texture and shape, which makes it easier to preserve identities.

User study. We perform a user study on Amazon Mechanical Turk (AMT), where each worker is presented with the reference image, an image synthesized by our method and an image synthesized by a competing method. We ask the turkers to evaluate the synthesized images based on their quality, realism of the expression, and similarity to the reference image. Since faces in ground truth images have different poses, for comparison with ground truth we only ask the subjects to evaluate based on the quality of the image and expression, eliminating other irrelevant factors as much as possible. For each comparison, we have $1,740$ pairs of images and each pair is evaluated by 3 workers. We only accepted turkers with a lifetime HIT approval rate $\geq 95\%$. We show the results in Table 2. Users prefer our methods over all other methods. We get a slightly higher preference score than ground truth. This proves that it is difficult for
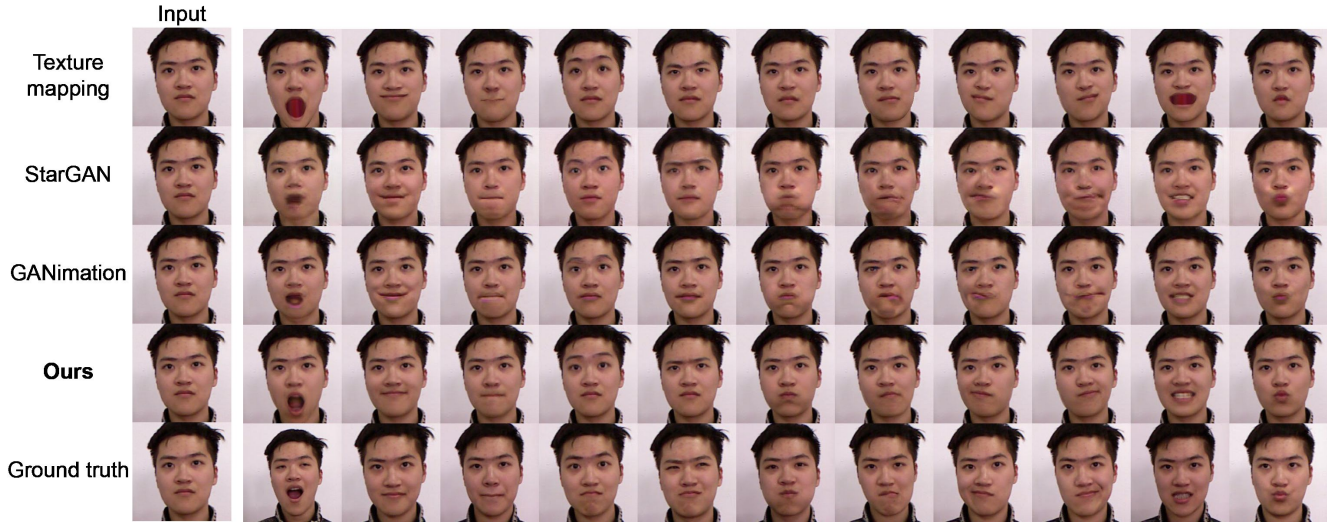
Figure 10: **Comparison of face synthesis methods.** Given the same input image each method generated 11 different facial expressions. Our approach produced less artifacts and renders more realistically looking images than the competing methods.
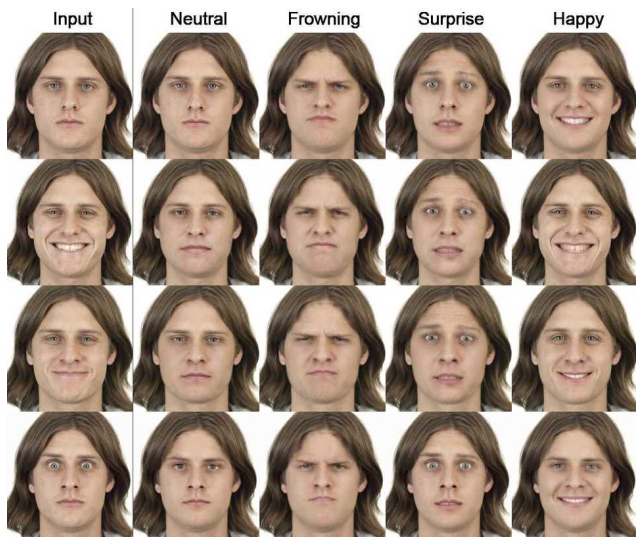

Figure 11: Manipulating images with different source expressions. Although the input images are different, each manipulated image looks plausible.

humans to distinguish between the images generated by our method and the ground truth.

5.3. More Results

Different input expressions. Our method can handle input expressions that are not neutral. We show synthesized images using the same person with different expressions as input in Fig. 11. Although our input images are different, our synthesized images with the same target expressions still look similar. We also note that the method can generate a different version of each expression for each subject.

Images in the wild. We show examples of our method applied to images in-the-wild in Fig. 1 and refer the reader to the supplementary materials for more in-the-wild results. Due to the decoupled face representation and separate texture and shape branches, our method is robust to different identities, expressions, head poses or lighting.

6. Conclusion

We presented a 3D guided fine grained face manipulation approach to transfer from one arbitrary expression to another arbitrary expression. The method decomposes an image into shape and texture spaces, followed by processing of these spaces with separate branches. We showed the benefits of such a scheme. Conditioning the pipeline on the spatial representation of important geometry information is advantageous over the straightforward approach of directly appending expression coefficients. To further boost the quality, we introduced several of the loss functions accounting for the pairwise consistency and identity. Our ablation studies supported the proposed framework. Furthermore, our method showed a significantly better ACD score as well as a preference by human annotators when compared to the competing approaches. Finally, when compared to the real images, the annotators were not able to distinguish our generated images from the real images, fully supporting the benefits of the presented method.

References

[1] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. SIGGRAPH '99, pages 187–194, New York, NY, USA, 1999. ACM Press/Addison-Wesley Publishing Co. 1, 2

[2] C. Bodnar. Text to image synthesis using generative adversarial networks. *CoRR*, abs/1805.00676, 2018. 4

[3] S. Bouaziz, Y. Wang, and M. Pauly. Online modeling for realtime facial animation. *ACM Trans. Graph.*, 32(4):40:1–40:10, July 2013. 5

[4] C. Cao, Q. Hou, and K. Zhou. Displaced dynamic expression regression for real-time facial tracking and animation. *ACM Trans. Graph.*, 33(4):43:1–43:10, July 2014. 2, 5

[5] C. Cao, Y. Weng, S. Zhou, Y. Tong, and K. Zhou. Facewarehouse: A 3d facial expression database for visual computing. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):413–425, 2014. 2, 3, 4, 5, 6, 7

[6] Y. Choi, M. Choi, M. Kim, J. Ha, S. Kim, and J. Choo. Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8789–8797, 2018. 2, 4, 6, 7

[7] E. Friesen and P. Ekman. Facial action coding system: a technique for the measurement of facial movement. *Consulting Psychologists Press*, 1978. 1, 2

[8] J. Huang, X. Shi, X. Liu, K. Zhou, L.-Y. Wei, S.-H. Teng, H. Bao, B. Guo, and H.-Y. Shum. Subspace gradient domain mesh deformation. *ACM Trans. Graph.*, 25(3):1126–1134, July 2006. 4

[9] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5967–5976, 2017. 4, 5, 6

[10] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European Conference on Computer Vision*, pages 694–711, 2016. 5

[11] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *CoRR*, abs/1710.10196, 2017. 5

[12] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, 2014. 3

[13] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 6

[14] G. Lample, N. Zeghidour, N. Usunier, A. Bordes, L. Denoyer, and M. Ranzato. Fader networks: Manipulating images by sliding attributes. In *Proceedings of the Neural Information Processing Systems Conference*, pages 5969–5978, 2017. 2

[15] B. Lévy and R. H. Zhang. Spectral Geometry Processing, 2009. ACM SIGGRAPH ASIA Course Notes. 5

[16] H. Li, T. Weise, and M. Pauly. Example-based facial rigging. *ACM Trans. Graph.*, 29(4):32:1–32:6, 2010. 1

[17] M. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *Proceedings of the Neural Information Processing Systems Conference*, pages 700–708, 2017. 2

[18] R. Liu, J. Lehman, P. Molino, F. P. Such, E. Frank, A. Sergeev, and J. Yosinski. An intriguing failing of convolutional neural networks and the coordconv solution. *CoRR*, abs/1807.03247, 2018. 2, 4

[19] D. S. Ma, J. Correll, and B. Wittenbrink. The chicago face database: A free stimulus set of faces and norming data. *Behavior research methods*, 47(4):1122–1135, 2015. 5

[20] X. Mao, Q. Li, H. Xie, R. Y. K. Lau, and Z. Wang. Multi-class generative adversarial networks with the L2 loss function. *CoRR*, abs/1611.04076, 2016. 5

[21] A. Pumarola, A. Agudo, A. M. Martinez, A. Sanfeliu, and F. Moreno-Noguer. Ganimation: Anatomically-aware facial animation from a single image. In *Proceedings of the European Conference on Computer Vision*, pages 835–851, 2018. 2, 3, 4, 5, 6, 7

[22] A. Siarohin, S. Lathuilire, S. Tulyakov, E. Ricci, and N. Sebe. Animating arbitrary objects via deep motion transfer. *arXiv:1812.08861*, 2018. 3

[23] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner. Face2face: Real-time face capture and reenactment of RGB videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2387–2395, 2016. 2

[24] L. Tran and X. Liu. Nonlinear 3d face morphable model. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7346–7355, 2018. 5

[25] S. Tulyakov, L. A. Jeni, J. F. Cohn, and N. Sebe. Viewpoint-consistent 3d face alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(9):2250–2264, 2018. 2

[26] S. Tulyakov, M. Liu, X. Yang, and J. Kautz. Mocogan: Decomposing motion and content for video generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1526–1535, 2018. 3, 7

[27] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. SIGGRAPH '05, pages 426–433, New York, NY, USA, 2005. ACM. 2

[28] O. Wiles, A. Sophia Koepke, and A. Zisserman. X2face: A network for controlling face generation using images, audio, and pose codes. In *Proceedings of the European Conference on Computer Vision*, pages 670–686, 2018. 3

[29] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Free-form image inpainting with gated convolution. *CoRR*, abs/1806.03589, 2018. 5

[30] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5505–5514, 2018. 5

[31] H. Zhang, T. Xu, and H. Li. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5908–5916, 2017. 5

[32] J. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2242–2251, 2017. 2, 4

[33] X. Zhu, Z. Lei, X. Liu, H. Shi, and S. Z. Li. Face alignment across large poses: A 3d solution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 146–155, 2016. 2