

FaceTuneGAN: Face Autoencoder for Convolutional Expression Transfer Using Neural Generative Adversarial Networks

Nicolas Olivier^{a,b,*}, Kelian Baert^{a,c,*}, Fabien Danieau^a, Franck Multon^{b,d} and Quentin Avril^a

^aInterDigital Inc., France

^bInria, Univ Rennes, CNRS, IRISA, France

^cIMT Atlantique, France

^dM2S, France

ARTICLE INFO

Keywords:

Digital double, 3D face, Expression transfer, Deep learning

ABSTRACT

In this paper, we present FaceTuneGAN, a new 3D face model representation decomposing and encoding separately facial identity and facial expression. We propose a first adaptation of image-to-image translation networks, that have successfully been used in the 2D domain, to 3D face geometry. Leveraging recently released large face scan databases, a neural network has been trained to decouple factors of variations with a better knowledge of the face, enabling facial expressions transfer and neutralization of expressive faces. Specifically, we design an adversarial architecture adapting the base architecture of FUNIT and using SpiralNet++ for our convolutional and sampling operations. Using two publicly available datasets (FaceScape and CoMA), FaceTuneGAN has a better identity decomposition and face neutralization than state-of-the-art techniques. It also outperforms classical deformation transfer approach by predicting blendshapes closer to ground-truth data and with less of undesired artifacts due to too different facial morphologies between source and target.

1. Introduction

Acquiring, modeling, simulating and rendering the 3D geometry of the human face have remained key challenges in computer graphics for decades. Lately, 3D digital characters have sparked more and more interest in numerous domains, with recent research enabling higher fidelity and realism. While the modeling of actors was made available years ago [2], 3D realistic faces can now be generated from scratch [30], and thus ease the population of digital worlds. With capture technology becoming available to the mass-market, realistic digital doubles enable seamless telecommunication between individuals within virtual worlds [51]. They also find use outside of the entertainment and telecommunication industries, e.g. sports, education, health and security. For instance, they have been used for the pre-visualization of plastic surgeries [5].

However, achieving a convincing representation of the 3D human face is still a difficult task. Countless attempts have resulted in appearances that, while technically impressive, fall just short of fooling the audience and end up in the so-called *uncanny valley*. This is due to our considerable ability to effortlessly infer social cues and gather information when looking at human faces [33]. Variations of the 3D face shape are based on several factors, including identity, pose, expression and age. An approach that is successfully able to decompose faces into these factors of variations would bring a better understanding of the face semantic, and eventually improve many applications such as expression transfer, expression extrapolation, performance retargeting,

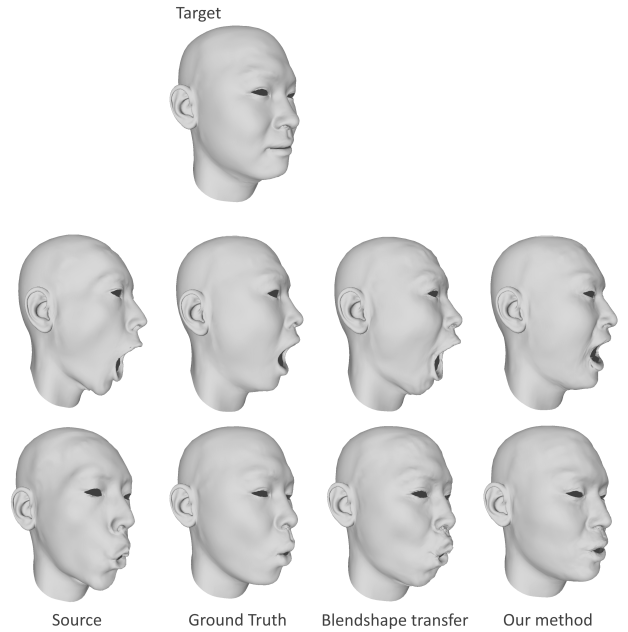


Figure 1: Blendshape transfer compared to our method. Results for the expressions *mouth_stretch* (row 2) and *lip_funneler* (row 3) expressions for two subjects of the FaceScape dataset are shown.

avatar creation [10] and personalization [41], facial recognition, or aging and de-aging. This work focuses on two factors: the identity and the expression. Our goal is to add expressions to a given 3D face model while preserving its identity and respecting its morphological constraints.

Current methods for bringing facial expressions to 3D faces often involve a lot of manual artistic work. While some high-budget productions can afford to have expressions

*Contributed Equally

**contact: nicolas.olivier@interdigital.com
ORCID(s):

manually sculpted by artists, this is a limiting factor for many applications and it is hardly scalable. Tools have been developed to adapt expressions automatically from one face to another. Blendshape transfer is a typical example [47]. However, most of these automatic methods fail to appropriately take into account the specific morphology of the person, resulting in inaccurate depictions.

Recently, the performance of style transfer and image-to-image translation networks has seen major performance improvements thanks to developments in deep learning. Style transfer is the task of separating a given data into a *content* part, and a *style* part, in order to stylize it by replacing its style part with another. The advent of generative adversarial networks [16] and their many variations [37, 43, 24] have opened the door to new architectures that outperform past approaches for style transfer in various domains [20, 57, 35]. Also, 2D generative models [25, 8] have come a long way in synthesizing convincing faces from compact latent representations while keeping some control on the semantics of the generated image. In this work, we apply style transfer techniques to the 3D domain in order to decompose face geometries into structural and stylistic features. Adopting the language of image-to-image translation methods, we designate *content* as the identity dependent features of the face, and *style* as the elements that vary with facial expressions. We apply this approach to decompose identity and expression using only expression labels, allowing morphology aware expression transfer between a face with a given identity (content), and another with a given expression (style). We present an architecture that is able to decompose a given 3D face into decoupled content (identity) and style (expression) latent codes, and map these latent representations back into a face. Using the SpiralNet++ convolution operator [15], we adapt successful image-to-image translations techniques, such as the multitask adversarial discriminator introduced by Liu et al. [35] to the 3D domain. Limiting ourselves to geometry (i.e. no texture), we achieve state-of-the-art results for decomposition and reconstruction of expressive 3D faces with a flexible approach that could be applied to other domains. This allows us to accurately transfer expressions in a morphology aware manner.

Following a review of related work in Section 2, our approach is presented in Section 3. Results and comparison against state-of-the-art methods are then detailed and discussed in Section 4. Finally, limitations and conclusions are provided in Section 5 and 6 respectively. Additional technical details are available in the appendices.

2. Related Work

Due to the prominence of face modeling as a computer graphics problem, there exists many methods for manipulating 3D geometry that are specifically tailored for the human face. These approaches aim to map raw geometries to more semantic spaces, in order to manipulate them in a more meaningful manner than directly displacing vertices. In this section, we first review the literature for 3D facial expression

modeling to assess the limitations of current methods. We then explore the state of image-to-image translation networks. Next, we introduce common 3D mesh convolution operators used in learning-based methods. Finally, we investigate how similar approaches have been developed in the 3D domain.

2.1. 3D Face Modeling

Methods for modeling expressions of 3D human face have historically been largely linear or multilinear. The reader interested in this topic can refer to the survey published by Egger et al. [12]. We first consider these two categories of models, before reviewing recent nonlinear approaches.

2.1.1. Linear and Multilinear 3D Face Models

Blanz and Vetter originally presented an approach for creating a 3D Morphable Model of the human face [3]. They used principal component analysis (PCA) to infer the distribution of facial geometry for a finite set of face scans. This method allows for manipulation of face features by manually mapping attributes to vectors in the parameter space. Rudimentary expression transfer is achieved by applying the deformation of one subject's expressive mesh onto another. One major limitation of this first naive approach is that the expression is not adapted to the target face. For instance, when applying a mouth-opening expression from one subject to another with a larger jaw, the vertex displacements should not be the same (example on Figure 1).

To alleviate the issue of PCA components mixing identity and expression, Vlasic et al. extend the PCA into a multilinear model which can decorrelate shape variations caused by identity and expression [49]. Li et al. introduce the FLAME model, which combines a linear shape spaces, articulated parts (jaw, neck, eyeballs) and blendshapes for expressions and pose [31]. This allows to further disentangle identity, pose, and expression. Despite several improvements over time (e.g. [50, 4]), these approaches fall short when it comes to modelling subtle facial details. Because variations of the face shape in the real world are nonlinear in nature, we now look at architectures that introduce nonlinearities for more accurate representations of the face shape.

2.1.2. Nonlinear 3D Face Models

In recent years, nonlinear approaches to modeling the human face have outperformed previous methods. In 2017, Li et al. present the linear FLAME model. Since then, new models that make use of recent developments in deep learning have outperformed FLAME. Ranjan et al. manage to represent 3D faces with 75% fewer model parameters than previous attempts, and outperform previous approaches for reconstruction and interpolation of expressions, in addition to demonstrating the ability to synthesize new faces [44]. Adopting a multi-scale autoencoder approach, they use graph convolutions on face meshes and introduce new efficient down-sampling and up-sampling operators. Abrevaya et al. use an Auxiliary Classifier GAN [40] to model non-linear variations of 3D face geometry while decoupling

identity and expression factors [1]. Given an identity vector, an expression vector and noise, the generator outputs 3D coordinates of the mesh. These coordinates are mapped to a two-dimensional image, which is fed to a discriminator that classifies it into an identity class, an expression class and real/fake classes using standard 2D Convolutional Neural Networks (CNN) techniques. Although these models achieve state-of-the-art reconstruction results and a decoupled face representation, they require optimizing an input noise vector to fit a given face. Jiang et al. address this by adopting an autoencoder architecture, similar to CoMA's [44], with the added ability to decompose the input face into identity and expression latent representations [21]. A fusion module then reconstructs these representations into a face. Zhang et al. extend this decomposition approach with an architecture that enforces distributional independence between identity and expression attributes by design [54]. They obtain great performance on the task of neutralizing the expression or identity of a given mesh. Their method consists in extracting both an identity mesh and an expression mesh from the given face, and adding them together for reconstruction. However, the decoded identity and expression mesh are respectively expression-agnostic and identity-agnostic by design. Thus, their sum cannot accurately capture the coupling between the two factors. This limits the performance of expression transfer for expressions that differ substantially from the neutral expression.

2.2. Style Transfer and Image-to-Image Translation

Style transfer methods traditionally transform an input image so that it preserves its content while adopting stylistic features of another input [14]. Image-to-image translation networks are a generalization of these procedures. Such networks have been able to successfully map between the specific local attributes of multiple sets of images [57, 35]. In other words, they can transfer an image from one class to another while keeping the content intact. Typical examples include translating between animal classes, changing the season of a photographed landscape or altering specific features of a human face (e.g. hair color, morphology, age). Isola et al. released the *pix2pix* framework, outperforming previous methods [29, 28, 56] by applying Conditional GANs to the image translation problem [20]. Furthermore, they have demonstrated outstanding performance in a large variety of domains, even compared to domain-specific methods. Unsupervised image-to-image translation methods have obtained competitive results without requiring pair supervision [48, 34, 35, 57]. This opens the door to wider applications, since less time needs to be spent manually annotating the data. These methods use specific constraints to preserve features of the input data in the absence of a target ground truth while training. If we consider a face's identity to be the content, and its expression to be a style, they appear to be good candidates to solve our problem. They are although limited to 2D domains.

2.3. Mesh Convolution Operators

Adapting successful image-to-image translation techniques to 3D geometry could enable better accuracy in transferring attributes from one shape to another. To tackle this ill-posed problem, one of the challenges is to extract style features from the shape structure. Analogous to the convolutional layers that are used to extract style information from images, several approaches have been developed for performing convolutions on 3D meshes over the past few years. Many papers apply spectral graph convolutions [6] to 3D faces [21, 54, 44]. Others map geometries to the 2D domain and apply standard 2D convolution operators [46, 1, 38].

In 2018, Lim et al. introduced SpiralNet, a new convolution operator specialized for 3D meshes [32]. Gong et al. later released SpiralNet++ to refine the approach [15]. This new operator captures local geometric features by using pre-computed spiral sequences on the mesh surface. Their results are competitive with previous state-of-the-art methods for reconstruction on 3D face datasets. With the same number of parameters, they outperform previous methods while running several times faster.

2.4. 3D Style Transfer

While there is a growth in approaches to apply convolutions to the 3D domain, there currently exist few attempts at 3D shape-to-shape style transfer. Recently, Segu et al. developed a 3D style transfer architecture that uses a PointNet [42] encoder and a decoder with Adaptive Instance normalization (AdaNorm) [52] to perform shape translation on static objects [45]. In 2019, Moschoglou et al. introduced 3DFaceGAN for representation, generation and translation of 3D facial surfaces allowing to transfer expression and resolution levels between facial scans.[38]. While they obtained state-of-the-art results at the time for the task of reconstruction, the expression transfer capabilities are limited. Their approach to multi-label face translation considers mapping neutral faces to a limited amount of discrete expressions. However, the identity and expression factors are not explicitly decoupled, as they share the same latent space.

Our aim is to create a model that can take arbitrary faces as input, and can disentangle identity and expression to perform expression transfer. This analysis of the literature suggests that techniques from recent image-to-image translation networks could be adapted to the 3D domain.

3. Proposed Method

We propose an architecture that adapts successful image-to-image translation techniques to 3D geometry in order to map local features of a 3D face shape (content) from one facial expression to another (style). Specifically, we adapt the base architecture of FUNIT [35], using SpiralNet++ [15] for our convolution and sampling operators. First, we define several terms and notations used in our approach. We then present our architecture and the objectives that are optimized during training.

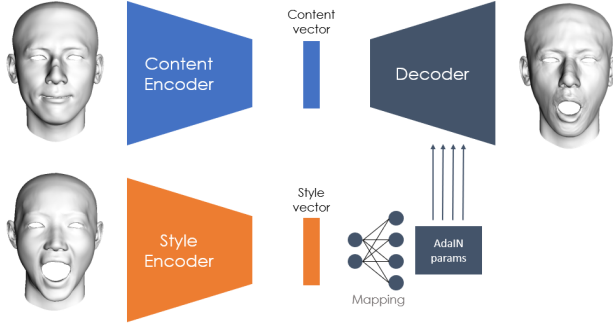


Figure 2: Our network architecture. The encoders extract and compress the features of their input into low-dimensional content and style vectors. A decoder reconstructs a mesh from these compact representations. The style information is passed to the decoder through AdaIN normalization layers. More details of the architecture are given in the Appendix (Section A).

3.1. Notations and Definitions

In our approach, we adopt language analogous to style transfer and image-to-image translations papers. We designate the *content* of the faces as their identity: these are the structural features of the face that do not depend on the facial expressions and should be preserved when performing style transfer. The term *style* corresponds to the features that vary with facial expressions.

Let χ denote a domain of 3D shape, defined as triangular meshes. All shapes within these domains have previously been fitted to the same topology. Thus, any sample $x \in \chi$ is represented by a matrix of 3D coordinates of shape $(V, 3)$, V being the number of vertices in our topology.

We propose to learn encoding functions E_c and E_s that map a sample $x \in \chi$ to its respective content and style latent codes $E_c(x)$ and $E_s(x)$. We also learn a corresponding decoding function Dec that reconstructs a 3D geometry from the latent codes. These functions should ideally satisfy the reconstruction constraint $Dec(E_c(x), E_s(x)) = x$, meaning that we are able to encode and decode our geometry in a lossless manner. To achieve expression transfer, we use the common method of encoding a source mesh, swapping its style code with one that corresponds to a different expression, then decode to reconstruct a mesh.

Average vertex distance: we use average vertex distance to measure distances between two geometries. Given two triangle meshes x and y with V vertices under the same topology, the average vertex distance is defined as:

$$AVD(x, y) = \frac{1}{V} \sum_{i=1}^V \|x_i - y_i\|_2 \quad (1)$$

where x_i and y_i respectively denote the i th vertex of x and y and $\|\cdot\|_2$ is the L2 Euclidean distance.

3.2. Architecture

We introduce an autoencoder that is able to separate the content (identity) and style (expression) features of the

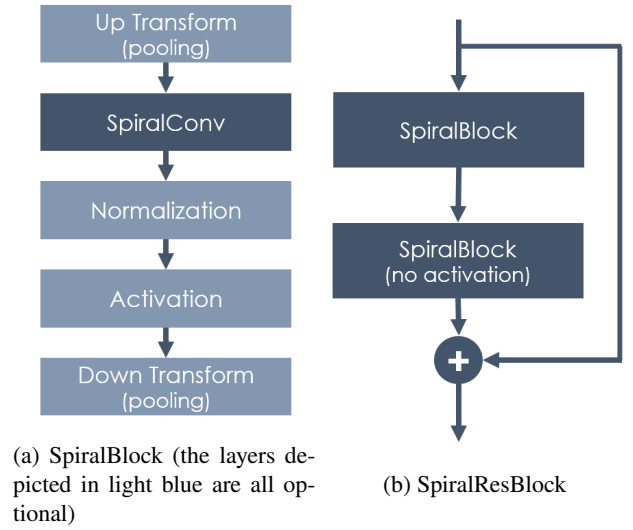


Figure 3: The building blocks of the network.

input facial shapes. The complete architecture is presented in Figure 2. The specific composition of the modules is given in the appendix (Section A, Figures 9 and 10), with tweaks depending on the dataset used for training. We adopt SpiralNet++ [15] for all of our convolutions and pooling [13] layers and use SpiralBlock as the main building block of our mapping functions. A SpiralBlock (Figure 3a) is composed of an optional up-sampling pooling layer, followed by a convolutional layer, normalization, activation and an optional down-sampling layer. Additionally, we use the SpiralResBlock variant described in Figure 3b.

3.2.1. SpiralNet-based Autoencoder for 3D Faces

The autoencoder is composed of 3 parts: The content encoder, that takes a face as input, and extract its content (identity), the style encoder, that takes another face and extract its style (expression), and finally the decoder, which generates a face from the outputs of the content and style encoders (see Figure 2).

Content Encoder: the content encoder is composed of several downscaling SpiralBlocks (the exact number depends on the dataset, see appendix A), two SpiralResBlocks and a Multi-Layer Perceptron (MLP). The SpiralBlocks capture local information in the geometry. We then flatten the spatial and feature dimensions together and use the MLP to compress the information into the content representation.

Style Encoder: similarly, the style encoder is made up of a series of downscaling SpiralBlocks. However, as style information must be global in nature, instead of flattening the spatial and feature dimensions together, we compute the mean along the spatial dimension. Similarly to the content encoder, the resulting vector is then fed to a MLP in order to control the size of the latent space.

Decoder: the decoder upscales the content code into a 3D geometry using SpiralBlocks that are conditioned by the style code using Adaptive Instance normalization (AdaIN) [18]. Given a sample x that is passing through the network,

AdaIN first normalizes the activations in each channel of x to a zero mean and unit variance. The activations are then scaled on a per-channel basis. We use a mapping function M that maps a style code y into (μ, σ) parameters for every channel of each AdaIN layer. Hence the following equation:

$$\text{AdaIN}(x, y) = M_\sigma(y) \frac{x - \mu(x)}{\sigma(x)} + M_\mu(y) \quad (2)$$

M is a learned affine function composed of multiple fully connected layers, taking the style latent code as input. Since the AdaIN transformation operates on whole channels, the style code alters global appearance information while the local features (e.g. the shape of the chin) are determined by the content code. The mapping function and decoder are not conditioned by a discrete class label. As a result, style codes are not domain-specific, in contrast with several existing methods for image translation [19], image generation [8] and shape transfer [45]. Having a single style space allows us to interpolate between style codes.

Together, the content encoder, style encoder, style mapping and decoder make up the generator G of our adversarial network.

3.2.2. Discriminator

Similar to FUNIT [35], we implement a multi-task adversarial discriminator D . Its role is to both enforce that the output mesh belongs to the distribution of the target style class, and that its geometry cannot be distinguished from a real scan. D solves as many binary classification tasks as there are style classes in the dataset. For each of these style classes, D outputs a classification of whether the geometry is a real sample of that class, or a translation output from the generator.

Let s denote a style class.

- When updating the discriminator with a translation output of class s from the generator, we penalize D if and only if its s -th output is positive and ignore the predictions for other classes. Given a real geometry of style class s , we penalize it if its s -th output is negative. This way, the discriminator learns to distinguish real from generated meshes.
- When updating the generator, we perform a style translation using a sample of style class s . We then penalize G if the s -th prediction output from D is negative. This encourages the generator to output realistic meshes.

Similarly to our encoders, the discriminator is composed of SpiralBlocks that gradually downscale its input mesh in the vertex dimension while adding features, followed by a linear layer for classification. More details can be found in Appendix A.

3.3. Loss Functions

In this section, we define the loss functions that are used for training our network. Let x and s denote two samples

respectively taken from our content and style sets. Let x_r denote the reconstructed mesh obtained by encoding and decoding x :

$$x_r = \text{Dec}(E_c(x), E_s(x)) \quad (3)$$

We also define x_t , the mesh obtained after translating x into the style class of s :

$$x_t = \text{Dec}(E_c(x), E_s(s)) \quad (4)$$

Reconstruction loss: we define the reconstruction loss for our generator as the following:

$$L_{\text{rec}}(G) = ||x_r - x||_2 \quad (5)$$

By minimizing this loss, we force the encoders and decoder to extract relevant information in order to compress to and from our latent spaces with the least possible loss.

Cycle consistency loss: we adopt the cycle consistency loss [55] [57], defined as:

$$L_{\text{cycle}}(G) = ||\text{Dec}(E_c(x_t), E_s(x)) - x||_2 \quad (6)$$

Its purpose is to ensure that the generator is able to translate x to the style class of s and back to its original style class with minimal content information loss.

Style reconstruction loss: we introduce a novel style reconstruction loss to encourage G to preserve the specific style features of the input style mesh in the style latent space:

$$L_{\text{srec}}(G) = ||E_s(x_t) - E_s(x)||_1 \quad (7)$$

Adversarial loss: our adversarial loss is a conditional loss given by:

$$L_{\text{adv}}(G, D) = \mathbb{E} \left[-\log(D_s(s)) \right] + \mathbb{E} \left[\log(1 - D_s(x_t)) \right] \quad (8)$$

where $D_s(\cdot)$ denotes the discriminator output for the style class of s and $\mathbb{E}[\cdot]$ the mean over the current batch.

Feature matching loss: we use a feature matching loss that leverages our multi-task adversarial discriminator to encourage our generator to output meshes that belong to the correct style class:

$$L_{\text{feat}}(G) = \mathbb{E} \left[||D^f(x_r) - D^f(x)||_1 \right] + \mathbb{E} \left[||D^f(x_t) - D^f(s)||_1 \right] \quad (9)$$

where $D^f(\cdot)$ denotes the last feature layer of the discriminator, prior to classification. By minimizing this loss, we enforce that the generator preserves style features when reconstructing the input and includes style features of the target when translating.

Discriminator regularization loss: we regularize the training by adopting the gradient penalty regularization loss L_{reg} introduced by Mescheder et al. [36] and used in FUNIT [35].

Laplacian Smoothing loss: finally, we use a Laplacian Smoothing loss on the translated mesh [11] [39]. Specifically, we use Pytorch3D's implementation of uniform mesh

Laplacian Smoothing, which consists in minimizing the distance between every vertex and the centroid of its neighbors [22]. Empirically, we have determined that this constraint makes the generator perform better at reconstruction at the cost of some high frequency resolution.

Full objective: our network is trained by solving the following minimax optimization problem:

$$\min_D \max_G L_{\text{reg}}(D) + \lambda_{\text{adv}} L_{\text{adv}}(G, D) + \lambda_{\text{feat}} L_{\text{feat}}(G) + \lambda_{\text{srec}} L_{\text{srec}}(G) + L_{\text{rec}}(G) + L_{\text{cycle}}(G) \quad (10)$$

where λ_{adv} , λ_{feat} and λ_{srec} , are hyper-parameters. The discriminator is trained for one iteration every time we train the generator.

4. Results

This section reports the evaluation of our proposed method. We first describe the datasets that were used. Then, we give details for the implementation with which the evaluation was conducted. Finally, our results are evaluated in a series of experiments and compared to existing methods.

4.1. Datasets

We evaluate the capabilities of our approach on two publicly available datasets:

FaceScape [53]: this dataset includes 16,940 topologically uniformed 3D face models, captured from 847 subjects performing 20 facial expressions. Displacement and texture maps are also available, though they are not used in this work. Out of the 847 subjects, we discard 40 due to issues with some of the scans. We select 10% of the remaining scans as our test set.

CoMA [44]: this dataset contains dynamic sequences of 12 subjects, each performing 12 facial expressions. In total, it comprises 144 sequences which add up to more than 20k face scans. While the small number of subjects does not allow for great generalization of identity features, the facial expressions are more extreme and asymmetrical than those of FaceScape. In our case, we need discrete expression labels for training. First, we select the first frame of each sequence as samples of neutral expression for the subject. Then, on each sequence, we select the frame with the largest average vertex distance to the first (neutral) frame. Since it is not always the best match to represent the expression, we manually verify all sequences and adjust this selection. We also split up the *mouth_up*, *mouth_middle* and *mouth_down* expressions into their left and right variants, adding up to a total of 17 expressions, though some do not exist for all subjects (see Figure 11). Finally, we sample 10 frames before and after the selected one in each sequence to add diversity and noise. We obtain a total of 3,720 scans, which we randomly split into train and test sets by a 9:1 ratio.

For fair comparison, we limit the dimensionality of our latent space. On the CoMA dataset, both our content and style spaces have 4 dimensions, adding up to a total dimensionality of 8. On the FaceScape dataset, Kacem et al.

[23], use a single latent space of 25 dimensions. However, this latent space is only used to represent neutral faces. Nonetheless, we limit ourselves to 20 content dimensions and 5 style dimensions for these comparisons.

4.2. Implementation Details

We set the weights in Equation (10) to $\lambda_{\text{adv}} = 1.0$, $\lambda_{\text{feat}} = 1.0$, $\lambda_{\text{srec}} = 0.4$. All of our spiral convolutions use a sequence length of 9 with no dilation (see [15] for more information). We train our generator and discriminator using ADAM optimizers [26], with a learning rate of 1×10^{-4} and a weight decay of 5×10^{-5} . All weights are initialized using the Kaiming method [17]. We train with a batch size of 8, until no significant improvements are seen on our reconstruction, neutralization and style transfer metrics. The training duration depends on the dataset. On FaceScape, we train for 70 epochs over a duration of approximately 36 hours. On CoMA, the network is trained for 480 epochs over 8 hours. Fine-tuning was done on a NVIDIA GeForce RTX 2080 Ti, a Tesla P100 GPU and a Tesla V100. The final training runs were done on the 2080 Ti for CoMA, Tesla V100 for FaceScape.

4.3. Experiments

First, we assess our autoencoder's ability to reconstruct an input mesh and compare it with state-of-the-art methods. Second, we conduct the expression neutralization task and compare our results with several baselines. Third, we evaluate our performance on the more general expression transfer task, and we compare the results to a blendshape transfer method.

4.3.1. Reconstruction

In order to ensure that our autoencoder is able to compress the information with as little information loss as possible, we calculate the reconstruction error as follows:

$$E_{\text{rec}} = \frac{1}{|S|} \sum_{x \in S} \text{AVD}(x, \text{Dec}(E_c(x), E_s(x))) \quad (11)$$

where S is a testing set of face geometries.

Figure 5 shows error maps of reconstruction samples on the CoMA dataset. In Table 1, we list quantitative results compared with other methods for both reconstruction and disentanglement of identity and expression. All methods use a total latent space size of 8. On CoMA, we attain reconstruction results in the range of other disentanglement methods: we perform better than FLAME [31] and Jiang et al. [21] but worse than Zhang et al. [54]. Note that the latter benefits from being able to use the entire CoMA database, while we only select a few frames per sequence to fit discrete expression labels. We also do not require each mesh to be paired with its neutral ground truth for training. For comparison, we include the results of the original SpiralNet++ [15] architecture, which obtains better performance on the reconstruction but does not disentangle identity and expression.

On the FaceScape dataset, we obtain a mean reconstruction error of 0.81 mm, with a standard deviation of 0.25 mm

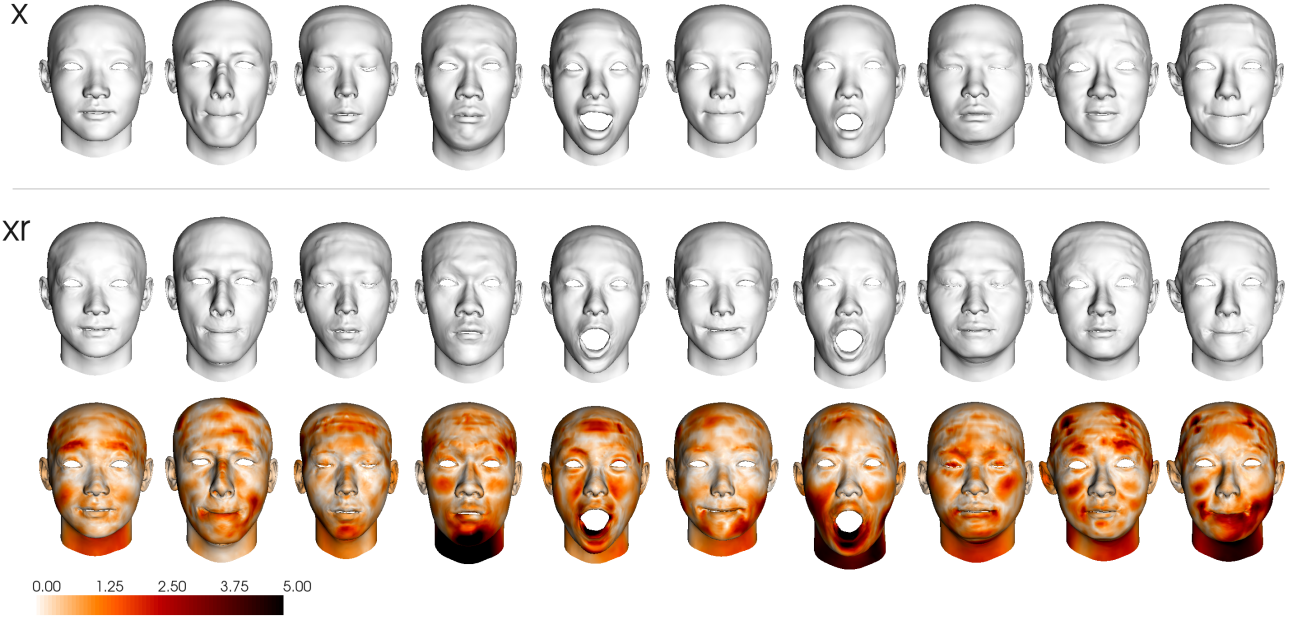


Figure 4: Reconstruction examples on FaceScape, with latent dimensions (20,5). The first row shows the input mesh. The reconstruction is displayed in the second row, while the third row shows an error map relative to the input. Values are in millimeters.

Model	Mean	Median
SpiralNet++ [15]	0.54 ± 0.66	0.32
FLAME [31]	1.45 ± 1.65	0.87
Jiang et al. [21]	1.41 ± 1.64	1.02
Zhang et al. [54]	0.67 ± 0.75	0.43
Our method	0.83 ± 0.21	0.77

Table 1

Reconstruction error on the CoMA dataset (mm). The SpiralNet++ method does not disentangle identity and expression.

and a median of 0.76 mm. Metrics from other work are not available for comparison. Examples of reconstructions are given in Figure 4. We can observe that the error is spread over the face, with notable patches on the neck, eyebrows, lips and cheeks. On the *mouth_stretch* expression (columns 5 and 7), the network visibly struggles to capture the geometry and position of the lower lip.

4.3.2. Expression Neutralization

We evaluate our method on the expression neutralization task using two metrics. We first adapt the identity decomposition error introduced by Jiang et al. [21] (and adopted by Zhang et al. [54]) to our approach. Let $x_{c,s}$ denote a mesh of content class c and style class s . We compute the identity decomposition as follows: for each content class c , we randomly select another content class c' and apply its neutral style onto each $(x_{c,s})_{s \in S}$. Then, we calculate the standard deviation of the resulting meshes using AVD :

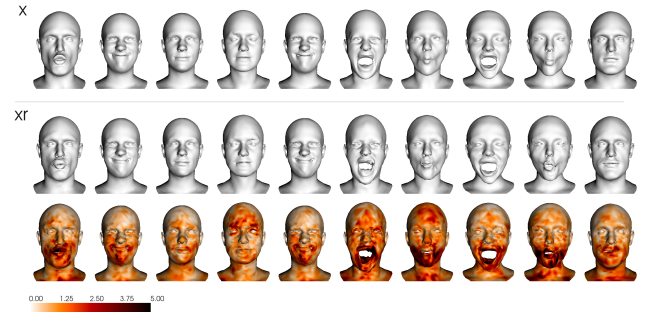


Figure 5: Reconstruction examples on CoMA. The first row shows the input mesh. The reconstruction is displayed in the second row, while the third row shows an error map relative to the input. Values are in millimeters.

$\sigma_c = \text{std}_{s \in S} \text{Dec}(E_c(x_{c,s}), E_s(x_{c',\text{neutral}}))$. Since we are effectively applying the same neutral style on meshes $(x_{c,s})_{s \in S}$, which only differ in style, a successfully trained network is supposed to yield identical outputs. A visualization for a subject of the CoMA dataset is given in Figure 6. We report the mean of these deviations as our identity decomposition metric (Equation (12)). For quantitative results, see Table 2.

$$E_{\text{id_decomp}} = \frac{1}{|C|} \sum_{c \in C} \sigma_c \quad (12)$$

In addition, we compare our results to existing methods on the FaceScape dataset by reporting the mean error between neutralized faces and the corresponding ground truth neutral. More precisely, we apply the following process: we

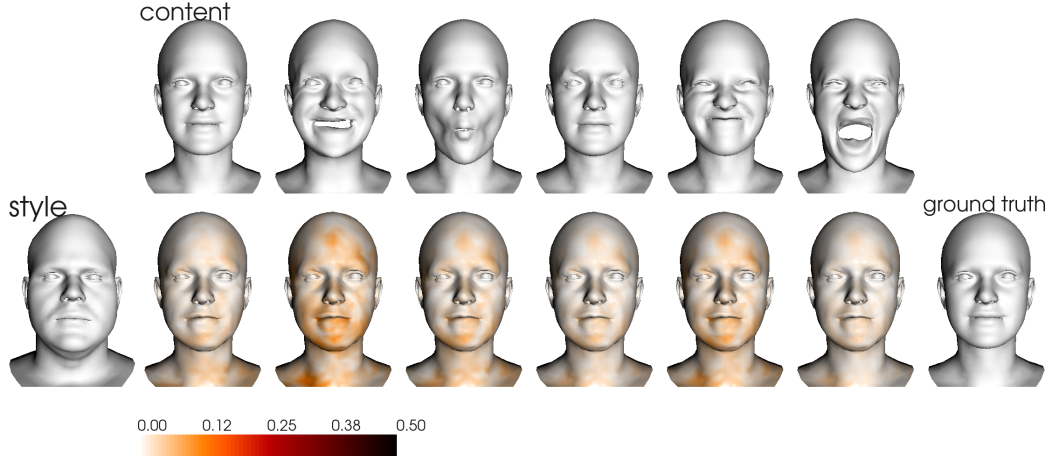


Figure 6: Variations of neutralized meshes for one subject of CoMA. The error map is relative to the mean of the outputs. Values are in millimeters. The identity decomposition error is the mean of this standard deviation computed for each subject.

Model	Mean	Median
FLAME [31]	0.599	0.591
Jiang et al. [21]	0.102	0.096
Zhang et al. [54]	0.019	0.020
Our method	0.018	0.018

Table 2
Identity decomposition error on the CoMA dataset (mm).

Model	Dataset	mean \pm std (median)
Ranjan et al. [44]	FaceScape	2.88
Kacem et al. [23]	FaceScape	2.02
Our method	FaceScape	1.47 \pm 1.43 (1.31)
Ranjan et al. [44]	CoMA	3.28
Kacem et al. [23]	CoMA	2.73
Our method	CoMA	0.98 \pm 0.46 (0.81)

Table 3
Neutralization error (mm). Standard deviations and medians for other methods are not provided.

randomly draw n triplets $(c, c', s)_{c \in C, c' \in C \setminus \{c\}, s \in S \setminus \{\text{neutral}\}}$, i.e. two different content classes and a non-neutral style class. For each triplet (c, c', s) , we apply the style of $x_{c', \text{neutral}}$ onto $x_{c, s}$ and compare the output with ground truth $x_{c, \text{neutral}}$ (Equation (13)). Results are presented in Table 3.

$$E_{\text{neu}} = \frac{1}{n} \sum_{(c, c', s)} \text{AVD}(x_{c, \text{neutral}}, \text{Dec}(E_c(x_{c, s}), E_s(x_{c', \text{neutral}}))) \quad (13)$$

Note that our model is trained in the unpaired setting. We do not explicitly pair expressive faces with their neutral

counterpart, contrary to the compared methods. These metrics allow us to evaluate our method against the state of the art for expression neutralization. However, this task is only one particular case of our network's capabilities.

4.3.3. Expression Transfer

We now evaluate our model on the more general expression transfer task. We introduce a metric similar to the neutralization error above and provide a baseline. We randomly draw n triplets exactly as described above: each triplet (c, c', s) contains two different content classes and a non-neutral style class. This time, the style of $x_{c', s}$ is applied onto $x_{c, \text{neutral}}$: $x_t = \text{Dec}(E_c(x_{c, \text{neutral}}), E_s(x_{c', s}))$. The output x_t is compared to the ground truth $x_{c, s}$.

$$E_{\text{transfer}} = \frac{1}{n} \sum_{(c, c', s)} \text{AVD}(x_{c, s}, x_t) \quad (14)$$

In Table 4, we report our expression transfer error on the FaceScape dataset for each style class s . It can be observed that some expressions are associated with much higher transfer errors than others. The most difficult expressions seem to be the ones farthest apart from neutral. Figure 7 shows visual examples for *smile* and *mouth_stretch*, the expressions with the lowest and highest error. It illustrates one of the challenges of the task particularly well. The first subject in the *mouth_stretch* expression opens his jaw less than most subjects in the dataset. In the transfer output, the lower part of the face is lower than it should be, because the corresponding $x_{c', s}$ (not shown) whose style is transferred has a more open jaw. During training, this would penalize the network. However, this difference could perhaps be attributed to a different interpretation of the expression by the subject (i.e. for the same expression, two scans can be different due to factors other than identity).

Finally we compare our method to traditional blendshape transfer [47]. Results are depicted in Figure 8. A visual comparison to our method is shown on Figure 1. For the two selected expressions, the blendshape transfer method applies

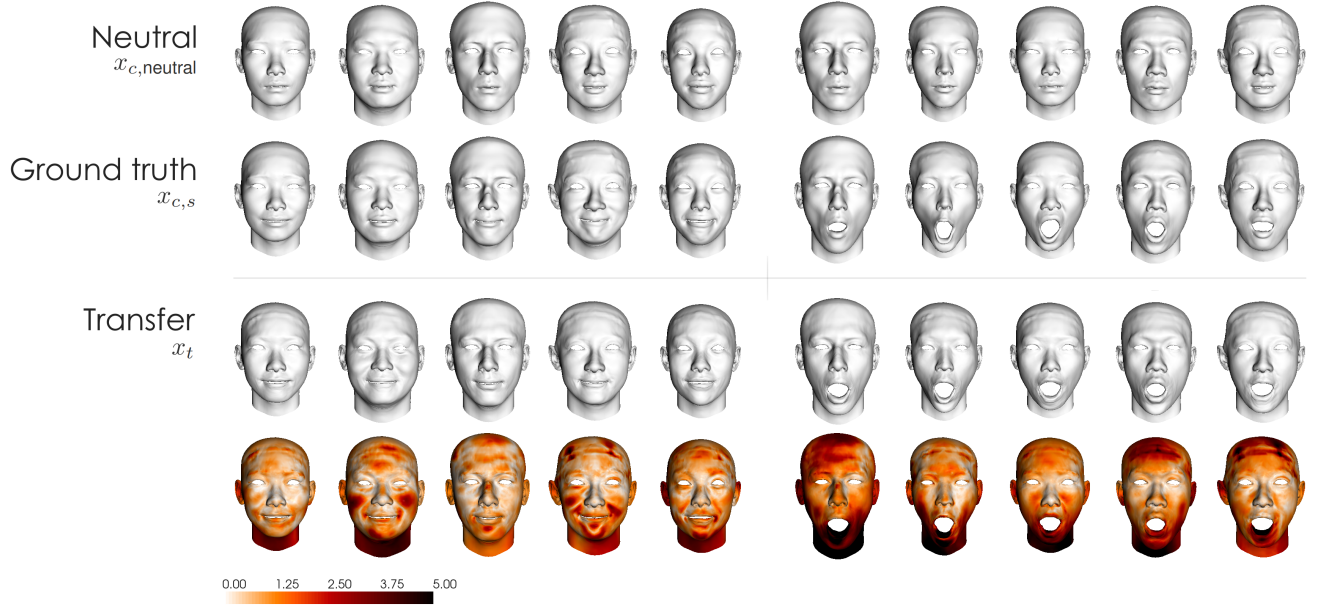


Figure 7: Expression transfer results for two expressions (smile on the left, mouth_stretch on the right). The transfer outputs are computed as described in Section 4.3.3.

Expression	Our Method	Bs transfer
smile	1.23	2.11
anger	1.32	3.25
sadness	1.45	4.10
grin	1.57	2.60
mouth_stretch	2.02	3.26
mouth_left	1.61	2.78
mouth_right	1.54	2.45
dimpler	1.42	2.20
jaw_left	1.53	2.24
jaw_right	1.63	2.59
jaw_forward	1.62	2.43
chin_raiser	1.65	2.82
lip_pucker	1.56	2.36
lip_funnel	1.80	2.64
lip_roll	1.52	2.50
cheek_blowing	1.77	3.55
eye_closed	1.23	1.86
brow_raiser	1.30	3.06
brow_lower	1.41	3.27
all	1.54	2.74

Table 4

Expression transfer errors on FaceScape (mm) with our method and the blendshape transfer.

the expression geometry to the target as it is on the source face, without taking into account the target morphology. Our method leads to more realistic outputs that look closer to the ground truth. Objective measurements shows that our method has smaller errors, as reported in Table 4.

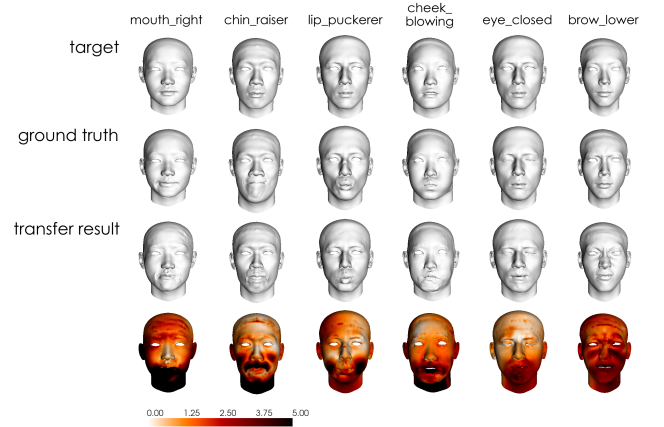


Figure 8: Blendshape transfer results. Expressions are transferred from a source (not shown) to the target

5. Limitations and Future Work

We have developed a network architecture that adapts style transfer techniques to 3D faces and demonstrated results that are competitive with the state of the art reconstruction, neutralization and expression transfer. However, there are several ways in which this work could be extended.

Interpolation : we visually investigated our model's ability to interpolate between two faces. Leveraging our low-dimensional representations, we perform linear interpolations in the content and style spaces. Let A and B denote two style codes obtained by encoding two scans of the same subject in a different facial expression. To interpolate, we move along the style vector \overline{AB} in fixed increments of $s \times \|\overline{AB}\|$ in the style space. Similarly, we can interpolate in content

space. Figure 12 shows results of interpolating between expressions of the same subject on the FaceScape dataset. In Figure 13, we interpolate between different subjects in their neutral expression.

Geometric approaches (e.g. blendshapes) have explicit control over the intensity of the expression being applied. On the contrary, our method in its current state does not have any additional constraint on the latent spaces. Hence the results of these interpolations are completely dependent on the structure of the data on which the network was trained. We find that the network learns a much more regular representation for the content space than the style space. We attribute this to the large amount of subjects in the FaceScape dataset, in comparison to the more limited variation of the 20 expressions.

Extrapolation : the previous interpolation method was extended to investigate the network’s ability to extrapolate from the training distribution. In Figure 14, we extrapolate in style space along a vector going from the neutral code of a subject to its style code for another expression. We find that while the network can sometimes succeed in amplifying the expression (e.g. *mouth_stretch* and *cheek_blowing* on Figure 14), it is also susceptible to changing some features in unexpected ways (e.g. the *dimpler* expression on Figure 14).

Figure 15 shows extrapolation in the content space. Our method is able to amplify the differences between two faces. Similarly to our findings for interpolation, we can observe that the extrapolations in content space are less prone to adding noise and warping unwanted features than those in the style space. These interpolation and extrapolation experiments are only preliminary tests. Due to the subjective nature of this task, a user study would need to be conducted to validate these results.

Increased realism : while we currently only predict geometry, one could use the data available in FaceScape to learn prediction of displacement maps and textures from the latent codes and generate much more realistic faces. Our network’s ability to decouple identity and expression factors could be well-suited to modelling the subtle expression-specific texture variations, a capability demonstrated by Chandran et al. [7].

Investigate bias : it is common for deep learning methods to display biases in the output distribution. While the gender of the subjects in the FaceScape dataset is distributed quite evenly, there is a clear bias toward 18 to 24 year-old Asians. It would be beneficial to study how this bias is reflected in the outputs and whether the network is able to generalize to more diverse faces.

Generative model : currently, sampling random content or style codes is very unlikely to provide satisfying results. In order to add this generative ability to the network, our autoencoder could be turned into a Variational Autoencoder [27] by adding a regularization term (e.g. Kullback-Leibler divergence loss) and encoding distributions instead of single points. This could also improve our ability to interpolate in latent space.

Unseen expressions : we could take a step further toward adapting the performance of image translation networks to the 3D domain by adding few-shot inference for unseen style classes. Given a few samples of a new style class, the model would be able to translate meshes to that new class. This capability is demonstrated in 2D by the FUNIT model [35].

Application to other domains : finally, while our approach was only tested on face datasets, applying it to data from other domains could yield interesting results. The image-to-image translation models from which it is inspired have been able to operate on animals, shapes of shoes and handbags, seasonal variations of landscapes, painting styles, etc. The challenge here will be the availability of large, labeled 3D datasets.

6. Conclusion

We have presented FaceTuneGAN, a novel 3D face model representation decomposing and encoding separately facial identity and facial expression. The proposed method is a first adaptation to the 3D domain of existing work for transferring features of images from one style class to another. Adapting several existing techniques to the 3D domain, this new style-based adversarial autoencoder architecture can capture identity and expression features of the input 3D face in separated low-dimensional spaces. Two encoders respectively extract the content (identity) and style (expression) information, which the decoder takes as inputs to reconstruct a 3D face. Additionally, a discriminator is used in an adversarial training scheme to regularize the training, enforcing the output to be realistic and of the correct style class. This method is shown to be better with previous state-of-the-art approaches on several tasks, outperforming them on several metrics, such as identity decomposition and neutralisation.

This method could bolster the creation of digital characters, allowing to accurately transfer expressions between various facial morphologies. The architecture of the method containing nothing specific to expressions, it also has the potential to be adapted to applied to other style transfer applications for 3D models.

7. Acknowledgments

This project has received funding from the Association Nationale de la Recherche et de la Technologie under CIFRE agreement No 2018/1656, and from the European Union’s Horizon 2020 research and innovation program under grant agreement No 952147.

References

- [1] Abrevaya, V.F., Boukhayma, A., Wuhler, S., Boyer, E., 2019. A decoupled 3D facial shape model by adversarial training, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE. pp. 9418–9427. URL: <https://hal.archives-ouvertes.fr/hal-02064711>, doi:10.1109/ICCV.2019.00951, arXiv:1902.03619.
- [2] Alexander, O., Rogers, M., Lambeth, W., Chiang, J.Y., Ma, W.C., Wang, C.C., Debevec, P., 2010. The Digital Emily Project: Achieving

- a photorealistic digital actor. *IEEE Computer Graphics and Applications* 30, 20–31. doi:10.1109/MCG.2010.65.
- [3] Blanz, V., Vetter, T., 1999. A morphable model for the synthesis of 3D faces, in: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1999*, ACM Press. pp. 187–194. doi:10.1145/311535.311556.
 - [4] Bolkart, T., Wuhler, S., 2016. A robust multilinear model learning framework for 3D faces, in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 4911–4919. doi:10.1109/CVPR.2016.531.
 - [5] Bottino, A., De Simone, M., Laurentini, A., Sforza, C., 2012. A new 3-D tool for planning plastic surgery. *IEEE Transactions on Biomedical Engineering* 59, 3439–3449. doi:10.1109/TBME.2012.2217496.
 - [6] Bruna, J., Zaremba, W., Szlam, A., LeCun, Y., 2014. Spectral networks and deep locally connected networks on graphs. 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings URL: <http://arxiv.org/abs/1312.6203>, arXiv:1312.6203.
 - [7] Chandran, P., Bradley, D., Gross, M., Beeler, T., 2020. Semantic Deep Face Models. *Proceedings - 2020 International Conference on 3D Vision, 3DV 2020*, 345–354. doi:10.1109/3DV50981.2020.00044.
 - [8] Choi, Y., Uh, Y., Yoo, J., Ha, J.W., 2020. StarGAN v2: Diverse Image Synthesis for Multiple Domains. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 8185–8194. URL: <http://arxiv.org/abs/1912.01865>, doi:10.1109/CVPR42600.2020.00821, arXiv:1912.01865.
 - [9] Clevert, D.A., Unterthiner, T., Hochreiter, S., 2016. Fast and accurate deep network learning by exponential linear units (ELUs), in: 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings. arXiv:1511.07289.
 - [10] Danieau, F., Gubins, I., Olivier, N., Dumas, O., Denis, B., Lopez, T., Mollet, N., Frager, B., Avril, Q., 2019. Automatic generation and stylization of 3d facial rigs. 26th IEEE Conference on Virtual Reality and 3D User Interfaces, VR 2019 - Proceedings, 784–792. URL: <https://ieeexplore.ieee.org/document/8798208/>, doi:10.1109/VR.2019.8798208.
 - [11] Desbrun, M., Meyer, M., Schröder, P., Barr, A.H., 1999. Implicit fairing of irregular meshes using diffusion and curvature flow, in: *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1999*, ACM Press. pp. 317–324. URL: <http://portal.acm.org/citation.cfm?doid=311535.311576>, doi:10.1145/311535.311576.
 - [12] Egger, B., Smith, W.A., Tewari, A., Wuhler, S., Zollhoefer, M., Beeler, T., Bernard, F., Bolkart, T., Kortylewski, A., Romdhani, S., Theobalt, C., Blanz, V., Vetter, T., 2020. 3D Morphable Face Models—Past, Present, and Future. *ACM Transactions on Graphics* 39. URL: <http://arxiv.org/abs/1909.01815>, doi:10.1145/3395208, arXiv:1909.01815.
 - [13] Garland, M., Heckbert, P.S., 1997. Surface simplification using quadric error metrics, in: *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1997*, ACM Press. pp. 209–216. doi:10.1145/258734.258849.
 - [14] Gatys, L.A., Ecker, A.S., Bethge, M., 2016. Image Style Transfer Using Convolutional Neural Networks, in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE*, pp. 2414–2423. URL: <http://ieeexplore.ieee.org/document/7780634/>, doi:10.1109/CVPR.2016.265.
 - [15] Gong, S., Chen, L., Bronstein, M., Zafeiriou, S., 2019. SpiralNet++: A fast and highly efficient mesh convolution operator. *Proceedings - 2019 International Conference on Computer Vision Workshop, ICCVW 2019*, 4141–4148. URL: <http://arxiv.org/abs/1911.05856>, doi:10.1109/ICCVW.2019.00509, arXiv:1911.05856.
 - [16] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y., 2014. Generative adversarial nets, in: Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., Weinberger, K.Q. (Eds.), *Advances in Neural Information Processing Systems*, Curran Associates, Inc. pp. 2672–2680. URL: <https://proceedings.neurips.cc/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf>, doi:10.3156/jsoft.29.5_177_2.
 - [17] He, K., Zhang, X., Ren, S., Sun, J., 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, in: *Proceedings of the IEEE International Conference on Computer Vision, IEEE*, pp. 1026–1034. URL: <http://ieeexplore.ieee.org/document/7410480/>, doi:10.1109/ICCV.2015.123, arXiv:1502.01852.
 - [18] Huang, X., Belongie, S., 2017. Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization, in: *Proceedings of the IEEE International Conference on Computer Vision, IEEE*, pp. 1510–1519. URL: <http://ieeexplore.ieee.org/document/8237429/>, doi:10.1109/ICCV.2017.167, arXiv:1703.06868.
 - [19] Hui, L., Li, X., Chen, J., He, H., Yang, J., 2018. Unsupervised Multi-Domain Image Translation with Domain-Specific Encoders/Decoders, in: *Proceedings - International Conference on Pattern Recognition*, pp. 2044–2049. doi:10.1109/ICPR.2018.8545169, arXiv:1712.02050.
 - [20] Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A., 2017. Image-to-image translation with conditional adversarial networks. *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 2017-Janua*, 5967–5976. URL: <http://arxiv.org/abs/1611.07004>, doi:10.1109/CVPR.2017.632, arXiv:1611.07004.
 - [21] Jiang, Z.H., Wu, Q., Chen, K., Zhang, J., 2019. Disentangled representation learning for 3D face shape, in: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 11949–11958. URL: <http://arxiv.org/abs/1902.09887>, doi:10.1109/CVPR.2019.01223, arXiv:1902.09887.
 - [22] Johnson, J., Ravi, N., Reizenstein, J., Novotny, D., Tulsiani, S., Lassner, C., Branson, S., 2020. Accelerating 3D deep learning with PyTorch3D. *SIGGRAPH Asia 2020 Courses*, 1–1. doi:10.1145/3415263.3419160, arXiv:2007.08501.
 - [23] Kacem, A., Cherenkova, K., Aouada, D., 2021. Disentangled Face Identity Representations for joint 3D Face Recognition and Expression Neutralisation URL: <http://arxiv.org/abs/2104.10273>, arXiv:2104.10273.
 - [24] Karras, T., Aila, T., Laine, S., Lehtinen, J., 2018. Progressive growing of GANs for improved quality, stability, and variation. 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings URL: <http://arxiv.org/abs/1710.10196>, arXiv:1710.10196.
 - [25] Karras, T., Laine, S., Aila, T., 2019. A style-based generator architecture for generative adversarial networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2019-June*, 4396–4405. URL: <http://arxiv.org/abs/1812.04948>, doi:10.1109/CVPR.2019.00453, arXiv:1812.04948.
 - [26] Kingma, D.P., Ba, J.L., 2015. Adam: A method for stochastic optimization. 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings arXiv:1412.6980.
 - [27] Kingma, D.P., Welling, M., 2014. Auto-encoding variational bayes, in: 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings. arXiv:1312.6114.
 - [28] Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A., Tejani, A., Totz, J., Wang, Z., Shi, W., 2017. Photo-realistic single image super-resolution using a generative adversarial network, in: *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 105–114. doi:10.1109/CVPR.2017.19, arXiv:1609.04802.
 - [29] Li, C., Wand, M., 2016. Precomputed real-time texture synthesis with markovian generative adversarial networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9907 LNCS*, 702–716. doi:10.1007/978-3-319-46487-9_43, arXiv:1604.04382.
 - [30] Li, R., Bladin, K., Zhao, Y., Chinara, C., Ingraham, O., Xiang, P., Ren, X., Prasad, P., Kishore, B., Xing, J., Li, H., 2020. Learning formation of physically-based face attributes. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3407–3416. URL: <http://arxiv.org/abs/2004.03458>, doi:10.1109/CVPR42600.2020.00347, arXiv:2004.03458.

- [31] Li, T., Bolkart, T., Black, M.J., Li, H., Romero, J., 2017. Learning a model of facial shape and expression from 4D scans, in: ACM Transactions on Graphics, pp. 1–17. URL: <https://dl.acm.org/doi/10.1145/3130800.3130813>, doi:10.1145/3130800.3130813.
- [32] Lim, I., Dielen, A., Campen, M., Kobbelt, L., 2019. A simple approach to intrinsic correspondence learning on unstructured 3D meshes, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 349–362. URL: <http://arxiv.org/abs/1809.06664>, doi:10.1007/978-3-030-11015-4_26, arXiv:1809.06664.
- [33] Little, A.C., Jones, B.C., Debruine, L.M., 2011. The many faces of research on face perception. Philosophical Transactions of the Royal Society B: Biological Sciences 366, 1634–1637. URL: <https://royalsocietypublishing.org/doi/10.1098/rstb.2010.0386>, doi:10.1098/rstb.2010.0386.
- [34] Liu, M.Y., Breuel, T., Kautz, J., 2017. Unsupervised image-to-image translation networks. Advances in Neural Information Processing Systems 2017-Decem, 701–709. URL: <http://arxiv.org/abs/1703.00848>, arXiv:1703.00848.
- [35] Liu, M.Y., Huang, X., Mallya, A., Karras, T., Aila, T., Lehtinen, J., Kautz, J., 2019. Few-Shot Unsupervised Image-to-Image Translation. IEEE/CVF International Conference on Computer Vision (ICCV) URL: <http://arxiv.org/abs/1905.01723>.
- [36] Mescheder, L., Geiger, A., Nowozin, S., 2018. Which training methods for GANs do actually converge? 35th International Conference on Machine Learning, ICML 2018 8, 5589–5626. URL: <http://arxiv.org/abs/1801.04406>, arXiv:1801.04406.
- [37] Mirza, M., Osindero, S., 2014. Conditional Generative Adversarial Nets URL: <http://arxiv.org/abs/1411.1784>, arXiv:1411.1784.
- [38] Moschoglou, S., Ploumpis, S., Nicolaou, M.A., Papaioannou, A., Zafeiriou, S., 2020. 3DFaceGAN: Adversarial Nets for 3D Face Representation, Generation, and Translation. International Journal of Computer Vision 128, 2534–2551. URL: <http://arxiv.org/abs/1905.00307>, doi:10.1007/s11263-020-01329-8, arXiv:1905.00307.
- [39] Nealen, A., Igarashi, T., Sorkine, O., Alexa, M., 2006. Laplacian mesh optimization. Proceedings - GRAPHITE 2006: 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia , 381–389 URL: <http://portal.acm.org/citation.cfm?id=1174429>, doi:10.1145/1174429.1174494, <https://dl.acm.org/citation.cfm?id=1174494>, doi:10.1145/1174429.1174494.
- [40] Odena, A., Olah, C., Shlens, J., 2017. Conditional image synthesis with auxiliary classifier gans, in: 34th International Conference on Machine Learning, ICML 2017, pp. 4043–4055. URL: <http://arxiv.org/abs/1610.09585>, arXiv:1610.09585.
- [41] Olivier, N., Hoyet, L., Danieau, F., Argelaguet, F., Avril, Q., Lecuyer, A., Guillotel, P., Multon, F., 2020. The impact of stylization on face recognition, in: Proceedings - SAP 2020: ACM Symposium on Applied Perception, ACM, pp. 1–9. URL: <https://dl.acm.org/doi/10.1145/3385955.3407930>, doi:10.1145/3385955.3407930.
- [42] Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017. PointNet: Deep learning on point sets for 3D classification and segmentation. Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017 2017-Janua, 77–85. URL: <http://arxiv.org/abs/1612.00593>, doi:10.1109/CVPR.2017.16, arXiv:1612.00593.
- [43] Radford, A., Metz, L., Chintala, S., 2016. Unsupervised representation learning with deep convolutional generative adversarial networks. 4th International Conference on Learning Representations, ICLR 2016 - Conference Track Proceedings URL: <http://arxiv.org/abs/1511.06434>, arXiv:1511.06434.
- [44] Ranjan, A., Bolkart, T., Sanyal, S., Black, M.J., 2018. Generating 3D Faces Using Convolutional Mesh Autoencoders. URL: <http://arxiv.org/abs/1807.10267>, http://link.springer.com/10.1007/978-3-030-01219-9_43, doi:10.1007/978-3-030-01219-9_43, arXiv:1807.10267.
- [45] Segu, M., Grinvald, M., Siegwart, R., Tombari, F., 2020. 3DSNet: Unsupervised Shape-to-Shape 3D Style Transfer URL: <http://arxiv.org/abs/2011.13388>, arXiv:2011.13388.
- [46] Sinha, A., Bai, J., Ramani, K., 2016. Deep learning 3D shape surfaces using geometry images, in: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pp. 223–240. doi:10.1007/978-3-319-46466-4_14.
- [47] Sumner, R.W., Popović, J., 2004. Deformation transfer for triangle meshes. ACM Transactions on Graphics 23, 399–405. URL: <https://doi.org/10.1145/1015706.1015736>, doi:10.1145/1015706.1015736.
- [48] Taigman, Y., Polyak, A., Wolf, L., 2017. Unsupervised Cross-Domain Image Generation. ICLR (Poster) URL: <http://arxiv.org/abs/1611.02200>.
- [49] Vlasic, D., Brand, M., Pfister, H., Popović, J., 2005. Face transfer with multilinear models, in: ACM Transactions on Graphics, pp. 426–433. doi:10.1145/1073204.1073209.
- [50] Wang, M., Panagakis, Y., Snape, P., Zafeiriou, S., 2017. Learning the multilinear structure of visual data, in: Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, pp. 6053–6061. doi:10.1109/CVPR.2017.641.
- [51] Wei, S.E., Saragih, J., Simon, T., Harley, A.W., Lombardi, S., Perdoch, M., Hypes, A., Wang, D., Badino, H., Sheikh, Y., 2019. VR facial animation via multiview image translation. ACM Transactions on Graphics 38, 1–16. doi:10.1145/3306346.3323030.
- [52] Xu, J., Sun, X., Zhang, Z., Zhao, G., Lin, J., 2019. Understanding and improving layer normalization. Advances in Neural Information Processing Systems 32. URL: <http://arxiv.org/abs/1911.07013>, arXiv:1911.07013.
- [53] Yang, H., Zhu, H., Wang, Y., Huang, M., Shen, Q., Yang, R., Cao, X., 2020. FaceScape: A large-scale high quality 3D face dataset and detailed riggable 3D face prediction. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition , 598–607 doi:10.1109/CVPR42600.2020.00068, arXiv:2003.13989.
- [54] Zhang, Z., Yu, C., Li, H., Sun, J., Liu, F., 2020. Learning Distribution Independent Latent Representation for 3D Face Disentanglement, in: Proceedings - 2020 International Conference on 3D Vision, 3DV 2020, pp. 848–857. doi:10.1109/3DV50981.2020.00095.
- [55] Zhou, T., Krähenbühl, P., Aubry, M., Huang, Q., Efros, A.A., 2016. Learning dense correspondence via 3D-guided cycle consistency. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 2016-Decem, 117–126. URL: <http://arxiv.org/abs/1604.05383>, doi:10.1109/CVPR.2016.20, arXiv:1604.05383.
- [56] Zhu, J.Y., Krähenbühl, P., Shechtman, E., Efros, A.A., 2016. Generative visual manipulation on the natural image manifold. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9909 LNCS, 597–613. URL: <http://arxiv.org/abs/1609.03552>, doi:10.1007/978-3-319-46454-1_36, arXiv:1609.03552.
- [57] Zhu, J.Y., Park, T., Isola, P., Efros, A.A., 2017. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. Proceedings of the IEEE International Conference on Computer Vision 2017-Octob, 2242–2251. URL: <http://arxiv.org/abs/1703.10593>, doi:10.1109/ICCV.2017.244, arXiv:1703.10593.

A. Architecture Details

This section details the specific layers of our network. Due to the different amount of vertices in the datasets (27k in FaceScape, 5k in CoMA), the architecture is tweaked differently for each. Figure 9 and 10 respectively depict the composition of our network modules for training on CoMA and FaceScape. All linear layers use ReLU activations, except for the final discriminator layer which uses a Softmax activation. All SpiralBlocks and SpiralResBlocks use Exponential Linear Unit (ELU) activations [9]. The style encoder and discriminator use no normalization. The content

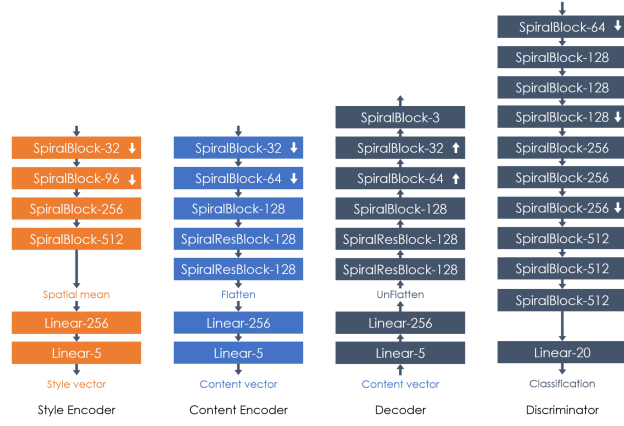


Figure 9: Composition of each module for training on FaceScape. The down/up arrows respectively indicate down-sampling and up-sampling layers, always by a factor of 4.

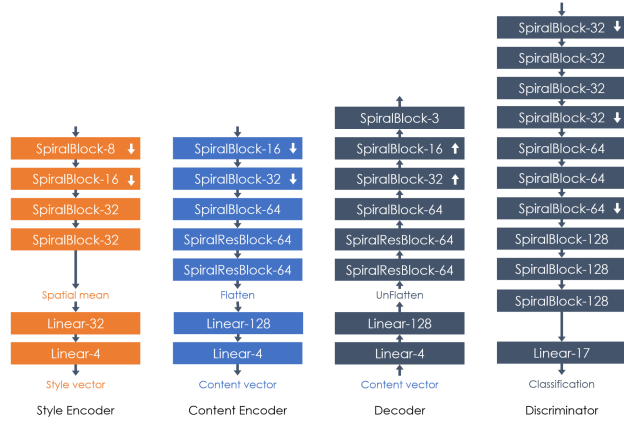


Figure 10: Network modules for training on CoMA. The down/up arrows respectively indicate down-sampling and up-sampling layers, always by a factor of 4.

encoder uses Instance Normalization layers on all SpiralBlocks and SpiralResBlocks. The decoder uses Adaptive Instance Normalization on its two first SpiralResBlocks and first SpiralBlock to inject the style information.

The mapping function M that converts the style codes into parameters for the AdaIN normalization layers of the decoder is composed of fully-connected layers. No normalization is used, and all layers use ReLU activations except for the last one. On CoMA, we use the following configuration: $4 \rightarrow 128 \rightarrow 128 \rightarrow 16 \rightarrow N_{\text{AdaIN}}$ where N_{AdaIN} is the number of AdaIN parameters in the decoder. On FaceScape, we use more layers and a larger size: $5 \rightarrow 256 \rightarrow 256 \rightarrow 256 \rightarrow 256 \rightarrow 256 \rightarrow 16 \rightarrow N_{\text{AdaIN}}$.

B. CoMA dataset

Figure 11 shows one selected frame for each (subject, expression) pair of the CoMA [44] dataset.

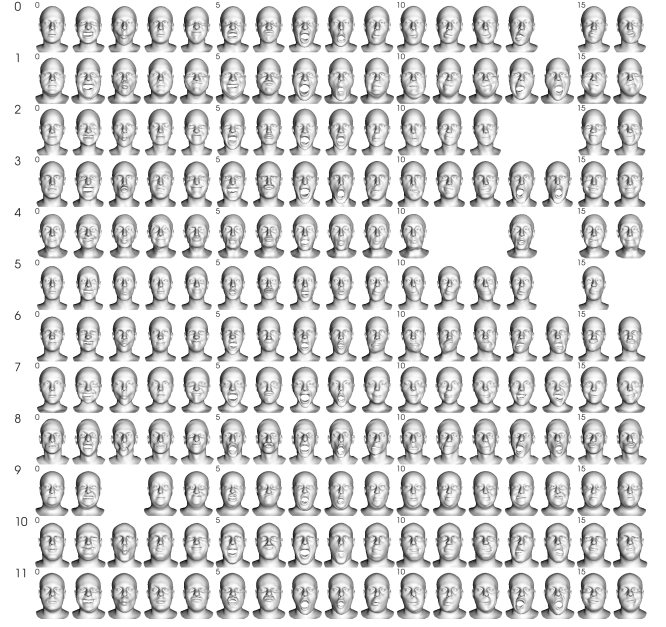


Figure 11: Selected frames on CoMA. Entries are missing when the expression performed by the subject varies significantly from the rest.

C. Interpolations

Expressions and identities interpolations are provided in Figures 12 and 13 respectively.

D. Expression Extrapolations

Expressions and identities extrapolations are provided in Figures 14 and 15 respectively.

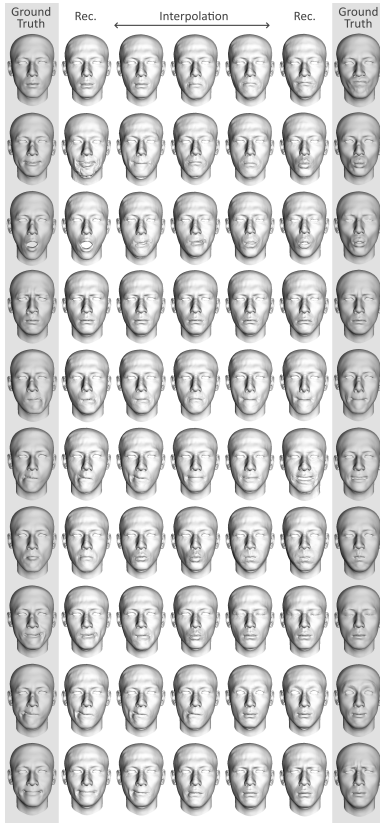


Figure 12: Expression interpolation results on FaceScape (style space). The ground truth scans are provided on the leftmost and rightmost columns. The second and second-to-last columns show their reconstruction. In the middle three columns, we interpolate along the style vector ($s = 0.25$).

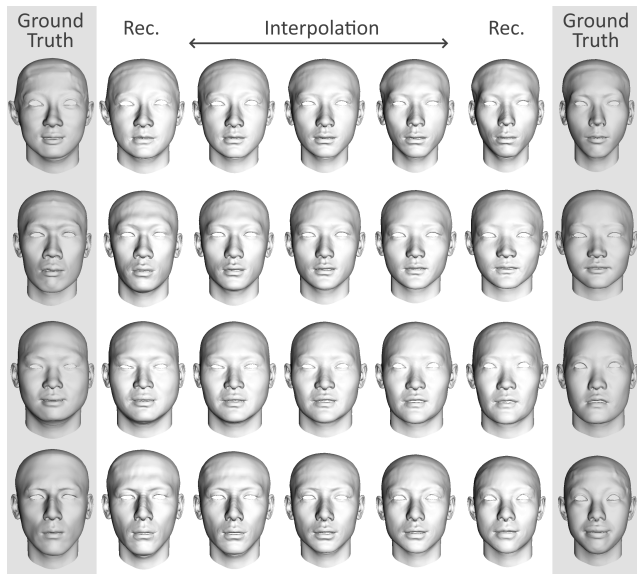


Figure 13: Identity interpolation results on FaceScape (content space). The ground truth scans are provided on the leftmost and rightmost columns. The second and second-to-last columns show their reconstruction. In the middle three columns, we interpolate along the content vector ($s = 0.25$).

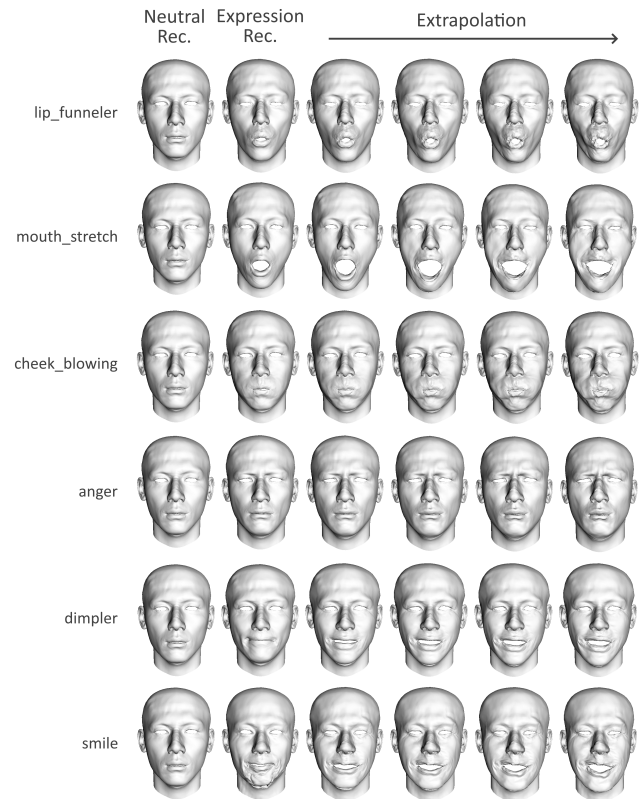


Figure 14: Extrapolation in style space. The two leftmost columns are reconstructions of the neutral and expression scans. We gradually extrapolate along the style vector ($s = 0.5$).

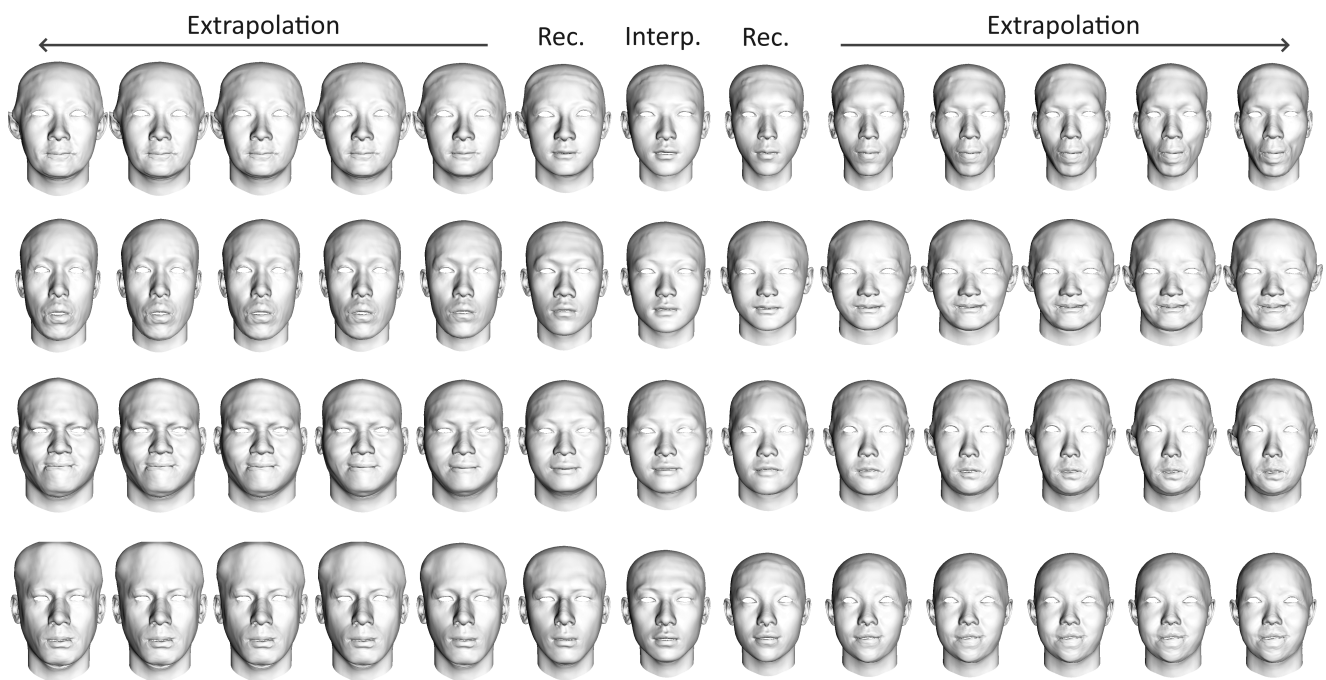


Figure 15: Extrapolation in content space. The reconstructions of two neutral scans are shown in the columns marked "Rec.". We move along their relative content vector from the middle to the outer columns ($s = 0.5$).