International Journal of Computer Vision (2020) 128:2534–2551
https://doi.org/10.1007/s11263-020-01329-8

3DFaceGAN: Adversarial Nets for 3D Face Representation, Generation, and Translation

Stylianos Moschoglou[1] · Stylianos Ploumpis[1] · Mihalis A. Nicolaou[2] · Athanasios Papaioannou[1] · Stefanos Zafeiriou[1]

Received: 30 April 2019 / Accepted: 7 April 2020 / Published online: 6 May 2020

Abstract

Over the past few years, Generative Adversarial Networks (GANs) have garnered increased interest among researchers in Computer Vision, with applications including, but not limited to, image generation, translation, imputation, and super-resolution. Nevertheless, no GAN-based method has been proposed in the literature that can successfully represent, generate or translate 3D facial shapes (meshes). This can be primarily attributed to two facts, namely that (a) publicly available 3D face databases are scarce as well as limited in terms of sample size and variability (e.g., few subjects, little diversity in race and gender), and (b) mesh convolutions for deep networks present several challenges that are not entirely tackled in the literature, leading to operator approximations and model instability, often failing to preserve high-frequency components of the distribution. As a result, linear methods such as Principal Component Analysis (PCA) have been mainly utilized towards 3D shape analysis, despite being unable to capture non-linearities and high frequency details of the 3D face—such as eyelid and lip variations. In this work, we present 3DFaceGAN, the first GAN tailored towards modeling the distribution of 3D facial surfaces, while retaining the high frequency details of 3D face shapes. We conduct an extensive series of both qualitative and quantitative experiments, where the merits of 3DFaceGAN are clearly demonstrated against other, state-of-the-art methods in tasks such as 3D shape representation, generation, and translation.

Keywords 3D · Face · GAN · Generation · Translation · Representation

1 Introduction

GANs are a promising unsupervised machine learning methodology implemented by a system of two deep neural networks competing against each other in a zero-sum game framework (Goodfellow et al. 2014). GANs became immediately very popular due to their unprecedented capability in terms of implicitly modeling the distribution of visual data, thus being able to generate and synthesize novel yet realistic images and videos, by preserving high-frequency details of the data distribution and hence appearing authentic to human observers. Many different GAN architectures have been proposed over the past few years, such as the Deep Convolutional GAN (DCGAN) (Radford et al. 2016) and the Progressive GAN (PGAN) (Karras et al. 2018), which was the first to show impressive results in generation of high-resolution images (Fig. 1).

A type of GANs which has also been extensively studied in the literature is the so-called Conditional GAN (CGAN) (Mirza and Osindero 2014), where the inputs of the genera-

Communicated by Jun-Yan Zhu, Hongsheng Li, Eli Shechtman, Ming-Yu Liu, Jan Kautz, Antonio Torralba.

Stylianos Moschoglou and Stylianos Ploumpis contributed equally to this work.

✉ Stylianos Moschoglou
s.moschoglou@imperial.ac.uk

Stylianos Ploumpis
s.ploumpis@imperial.ac.uk

Mihalis A. Nicolaou
m.nicolaou@cyi.ac.cy

Athanasios Papaioannou
a.papaioannou11@imperial.ac.uk

Stefanos Zafeiriou
s.zafeiriou@imperial.ac.uk

[1] Department of Computing, Imperial College London, United Kingdom and Facesoft.io., London, UK

[2] Computation-based Science and Technology Research Centre, The Cyprus Institute, Nicosia, Cyprus

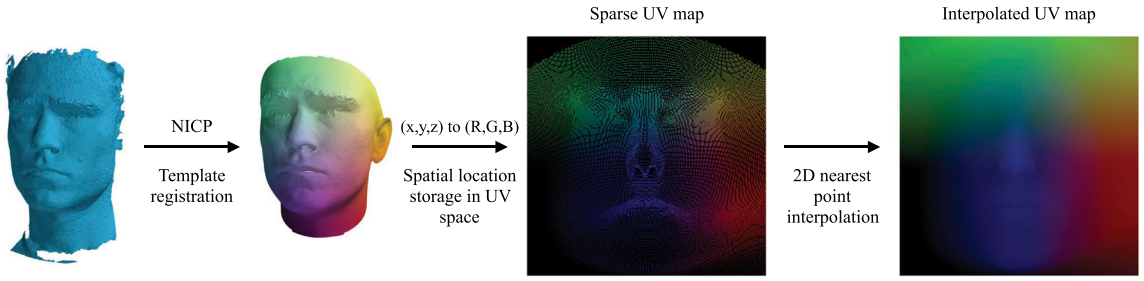

Fig. 1 A graphical representation of the data preprocessing step. We begin by applying non-rigidly a mesh template to the raw scan and we later store the spatial information of the vertices (x, y, z) into a UV space. Lastly, a 2D nearest point interpolation is performed to fill out the missing values

tor as well as the discriminator are conditioned on the class labels. Applications of CGANs include domain transfer (Kim et al. 2017; Bousmalis et al. 2017; Tzeng et al. 2017), image completion (Li et al. 2017; Yang et al. 2017; Wang et al. 2017), image super-resolution (Nguyen et al. 2018; Johnson et al. 2016; Ledig et al. 2017) and image translation (Isola et al. 2017; Zhu et al. 2017; Choi et al. 2018; Wang et al. 2018).

Despite the great success GANs have had in 2D image/video generation, representation, and translation, no GAN method tailored towards tackling the aforementioned tasks in 3D shapes has been introduced in the literature. This is primarily attributed to the lack of appropriate decoder networks for meshes that are able to retain the high frequency details (Dosovitskiy and Brox 2016; Jackson et al. 2017).

In this paper, we study the task of representation, generation, and translation of 3D facial surfaces using GANs. Examples of the applications of 3DFaceGAN in the tasks of 3D face translation as well as 3D face representation and generation are presented in Figs. 2 and 3, respectively. Due to the fact that (a) the use of volumetric representation leads to very low-quality representation of faces (Fan et al. 2017; Qi et al. 2017), and (b) the current geometric deep learning approaches (Bronstein et al. 2017), and especially spectral convolution, preserve only the low-frequency details of the 3D faces, we study approaches that use 2D convolutions in a UV unwrapping of the 3D face. The process of unwrapping a 3D face in the UV domain is shown in Fig. 1. Overall, the contributions of this work can be summarized as follows.

- We introduce a novel autoencoder-like network architecture for GANs, which achieves state-of-the-art results in tasks such as 3D face representation, generation, and translation.
- We introduce a novel training framework for GANs, especially tailored for 3D facial data.
- We introduce a novel process for generating realistic 3D facial data, retaining the high frequency details of the 3D face.

The rest of the paper is structured as follows. In Sect. 2, we succinctly present the various methodologies that can be utilized in order to feed 3D facial data into a deep network and argue why the UV unwrapping of the 3D face was the method of choice. In Sect. 3, we present all the details with respect to 3DFaceGAN training process, losses, and model architectures. Finally, in Sect. 4, we provide information about the database we collected, the preprocessing we carried out in the databases we utilized for the experiments and lastly we present extensive quantitative and qualitative experiments of 3DFaceGAN against other state-of-the-art deep networks.

2 3D Face Representations for Deep Nets

The most common representation of a 3D face is through a 3D mesh. Adopting a 3D mesh representation requires application of mesh convolutions defined on non-Euclidean domains (i.e., geometric deep learning methodologies[1]). Over the past few years, the field of geometric deep learning has received significant attention (Maron et al. 2017; Litany et al. 2017; Lei et al. 2017). Methods relevant to this paper are auto-encoder structures such as Ranjan et al. (2018); Litany et al. (2018). Nevertheless, such auto-encoders, due to the type of convolutions applied, mainly preserve low-frequency details of the meshes. Furthermore, architectures that could potentially preserve high-frequency details, such as skip connections, have not yet been attempted in geometric deep learning. Therefore, geometric deep learning methods are not yet suitable for the problem we study in this paper.

Another way to work with 3D meshes is to concatenate the coordinates of the 3D points in an 1D vector and utilize fully connected layers to decode correctly the structure of the point cloud (Fan et al. 2017; Qi et al. 2017). Nevertheless, in this way the triangulation and spatial adjacent information is lost and the number of the parameters describing this formulation is extremely large which makes the network hard to train.

[1] A thorough overview describing the first attempts towards geometric deep learning can be found in Bronstein et al. (2017).

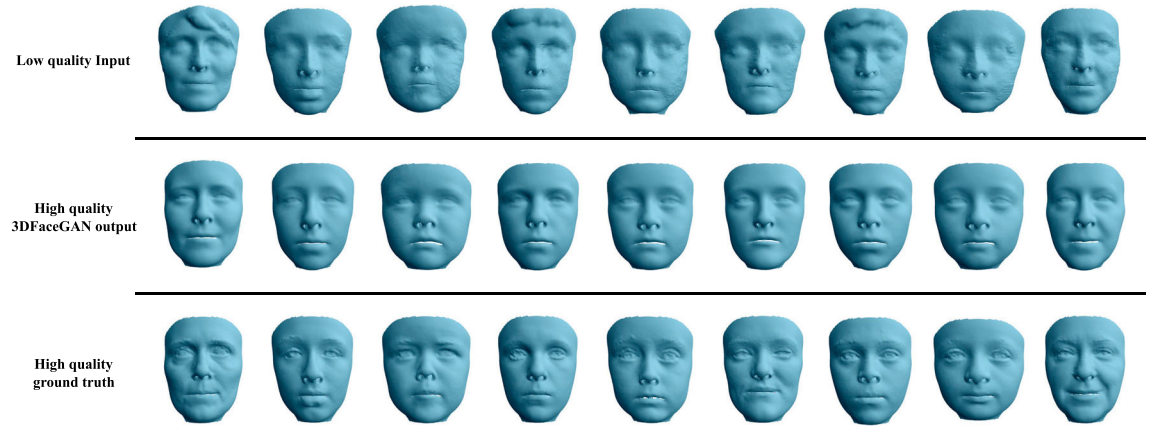

Fig. 2 Results of 3DFaceGAN in the shape translation task on test data of the proposed Hi-Lo database. The first row of shapes shows the low quality facial meshes captured by a low cost sensor, whereas the bottom row depicts the same subjects captured in high quality by an expensive high-end apparatus. The middle row shows our shape translation output results when the network takes as inputs the low quality 3D facial scans

Fig. 3 3D face representation and generation utilizing the proposed 3DFaceGAN. In **a** we demonstrate the 3D face representation capability of 3DFaceGAN. The first row shows the reconstructed 3D faces whereas the second row shows the corresponding real 3D faces. As evidenced, 3DFaceGAN is able to capture and reconstruct non-linear details of the 3D face such as lips, eyelids, etc. In **b** we present the generative nature of 3DFaceGAN. The left and right hand side show the real 3D face targets. The generated samples in between show the reconstructions and the interpolations of the targets in the latent space

Recently, many approaches aim at regressing directly on the latent parameters of a learned model space, e.g., PCA, rather than the 3D coordinates of points (Richardson et al. 2017; Tran et al. 2017; Dou et al. 2017; Genova et al. 2018). This formulation limits the geometrical details of the 3D representations and is restricted to their latent model space. Some approaches try to alleviate this problem by combining a regression methodology with a multi-level face model that induces an out-of-space generalization of the learned subspace (Tewari et al. 2018). Additionally, Tewari et al. (2019) propose to learn a face identity model from a multi-frame video-based self-supervised deep network that jointly learns to reconstruct 3D faces in-the-wild. Furthermore, various model based extensions try to capture the high frequency

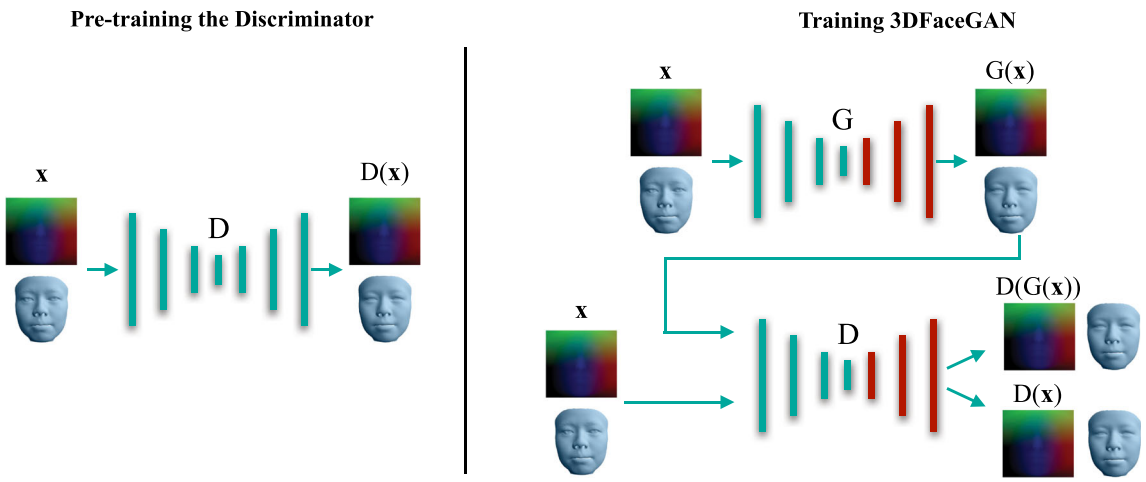

Fig. 4 3DFaceGAN training process in a nutshell. The networks receive (extract) 2D facial UVs as inputs (outputs). The corresponding 3D faces are shown below or next to them. We firstly pre-train D (left figure). We then use the learned weights/biases to initialize D and G and subsequently start the adversarial training (right figure). The decoder parts of D and G are depicted in red color as we freeze the weights/biases updates during the training phase of 3DFaceGAN

facial characteristics by introducing a nonlinear subspace that models both the shape and the appearance (Liu et al. 2019; Tran et al. 2018). Although these methods produce realistic facial characteristics, they are difficult to train and are very sensitive to initialization. In contrast, a 3D volumetric space is introduced in Jackson et al. (2017) as a representation of a 3D structure and exploits a Volumetric Regression Network which outputs a discretized version of the 3D structure. Due to discretization, the predicted 3D shape has low quality and corresponds to non-surface points that are difficult to handle (Fig. 4).

Lastly, in Feng et al. (2018), a UV spatial map framework is utilized where the 3D coordinates of the points are stored in a UV space instead of the texture values of the mesh. This formulation exhibits a very good representation for 3D meshes where there are no overlapping regions and the mesh is optimally unwrapped. Since the 3D mesh is transferred in a 2D UV domain, we are then able to use 2D convolutions, with the whole range of capabilities they offer. As a result, this is our preferred methodology for preprocessing the 3D face scans, as further explained in Sect. 4.2.

3 3DFaceGAN

In this Section we describe the training process, network architectures, and loss functions we utilized for 3DFaceGAN. Moreover, we discuss the framework we utilized for 3D face generation as well as present an extension of 3DFaceGAN which is able to handle data annotated with multiple labels. Finally, we should point out that while most GANs use discriminator architectures with logit outputs, in 3DFaceGAN we use an autoencoder as a discriminator for reasons that are thoroughly explained in Sect. 3.2 .

3.1 Objective Function

The main objective of the generator G is to retrieve a facial UV map x as input and generate a *fake* one, $G(x)$, which in turn should be as close as possible to the *real* target facial UV map y. For example, in the case of 3D face translation, the input can be a neutral face and the output a certain expression (e.g., *happiness*) or in the case of 3D face reconstruction the input can be a 3D facial UV map and the output a reconstruction of the particular 3D facial UV map. The goal of the discriminator D is to distinguish between the *real* (y) and *fake* ($G(x)$) facial UV maps. Throughout the training process, D and G compete against each other until they reach an equilibrium, i.e., until D can no longer differentiate between the *fake* and the *real* facial UV maps.

Adversarial loss. To achieve the 3DFaceGAN objective, we propose to utilize the following loss for the adversarial part. That is,

$$\begin{aligned} \mathcal{L}_D &= \mathbb{E}_y\left[\mathcal{L}(y)\right] - \lambda_{adv} \cdot \mathbb{E}_x\left[\mathcal{L}(G(x))\right], \\ \mathcal{L}_G &= \mathbb{E}_x\left[\mathcal{L}(G(x))\right], \end{aligned} \tag{1}$$

where $D(\cdot)$ refers to the output of the discriminator D, $\mathcal{L}(x) \doteq \|x - D(x)\|_1$, and λ_{adv} is the hyper-parameter which controls how much weight should be put on $\mathcal{L}(G(x))$. The higher the λ_{adv}, the more emphasis D puts on the task of differentiating between the real and fake data. The lower the λ_{adv}, the more emphasis D puts on reconstructing the actual real data. There is a fine line between which task D

should primarily focus on by adjusting λ_{adv}. In our experiments we deduced that for relatively low values of λ_{adv} we retrieve optimal performance as then D is able to influence the updates of G in such a way that the generated facial UV maps are more realistic. During the adversarial training, D tries to minimize $\mathcal{L}_D$ whereas G tries to minimize $\mathcal{L}_G$. Similar to recent works such as Zhao et al. (2017); Berthelot et al. (2017), the discriminator D has the structure of an autoencoder. Nevertheless, the main differences are that (a) we do not make use of the margin m as in Zhao et al. (2017) or the equilibrium constraint as in Berthelot et al. (2017), and (b) we use the autoencoder structure of the discriminator and pre-train it with the *real* UV targets prior to the adversarial training. Further details about the training procedure are presented in Sect. 3.2.

Reconstruction loss. With the utilization of the adversarial loss (1), the generator G is trying to "fool" the discriminator D. Nevertheless, this does *not* guarantee that the *fake* facial UV will be close to the corresponding *real*, target one. To impose this, we use an $L1$ loss between the *fake* sample $G(x)$ and the corresponding *real* one, y, so that they are as similar as possible, as in Isola et al. (2017). Namely, the reconstruction loss is the following.

$$\mathcal{L}_{rec} = \mathbb{E}_x \left\| G(x) - y \right\|_1 . \tag{2}$$

Full objective. In sum, taking into account (1) and (2), the full objective becomes

$$\begin{aligned} \mathcal{L}_D &= \mathbb{E}_y \left[\mathcal{L}(y) \right] - \lambda_{adv} \cdot \mathbb{E}_x \left[\mathcal{L}(G(x)) \right], \\ \mathcal{L}_G &= \mathbb{E}_x \left[\mathcal{L}(G(x)) \right] + \lambda_{rec} \cdot \mathcal{L}_{rec}, \end{aligned} \tag{3}$$

where λ_{rec} is the hyper-parameter that controls how much emphasis should be put on the reconstruction loss. Overall, the discriminator D tries to minimize $\mathcal{L}_D$ while the generator G tries to minimize $\mathcal{L}_G$.

3.2 Training Procedure

In this Section, we first describe how we pre-train the discriminator (autoencoder) D and then provide details with respect to the adversarial training of 3DFaceGAN.

Pre-training the discriminator. The majority of GANs in the literature utilize discriminator architectures with logit outputs that correspond to a prediction on whether the input fed into the discriminator is *real* or *fake*. Recently proposed GAN variations have nevertheless taken a different approach, namely by utilizing autoencoder structures as discriminators (Zhao et al. 2017; Berthelot et al. 2017). Using an autoencoder structure in the discriminator D is of paramount importance in the proposed 3DFaceGAN. The benefit is twofold: (a) we can pre-train the autoencoder D acting as discriminator prior to the adversarial training, which leads to better quantitative as well as more compelling visual results[2], and (b) we are able to compute the actual UV space dense loss, as compared to simply deciding on whether the input is real or fake. As we empirically show in our experiments and ablation studies, this approach encourages the generator to produce more realistic results than other, state-of-the-art methodologies.

Adversarial training. Before starting the adversarial training, we initialize the weights and biases[3] for both the generator G and the discriminator D utilizing the learned parameters estimated after the pre-training of D (the architecture of G is identical to the architecture of D). During the training phase of 3DFaceGAN, we freeze the parameter updates in the decoder parts for both the generator G and the discriminator D. Furthermore, we utilize a low learning rate on the encoder and bottleneck parts of G and D so that overall the parameter updates are relatively close to the ones found during the pre-training of D.

Network architectures. The network architectures for both the discriminator D and the generator G are the same. In particular, each network is consisted of 2D convolutional blocks with kernel size of three, stride and padding size of one. Down-sampling is achieved by average 2D pooling with kernel and stride size of two. The convolution filters grow linearly in each down-sampling step. Up-sampling is implemented by nearest-neighbor with scale factor of two. The activation function that is primarily used is ELU (Clevert et al. 2016), apart from the last layer of both D and G where Tanh is utilized instead. At the bottleneck we utilize fully connected layers and thus project the tensors to a latent vector $b \in \mathbb{R}^{N_b}$. To generate more compelling visual results, we utilized skip connections (He et al. 2016; Huang et al. 2017) in the first layers of the decoder part of both the generator and the discriminator. Further details about the network architectures are provided in Table 1.

3.3 3D Face Generation

Variational autoencoders (VAEs) (Kingma and Welling 2014) are widely used for generating new data using autoencoder-like structures. In this setting, VAEs add a constraint on the latent embeddings of the autoencoders that forces them to roughly follow a normal distribution. We can then generate new data by sampling a latent embedding from the normal distribution and pass it to the decoder. Neverthe-

[2] note that pre-training D is not possible when the outputs are logits since there are no fake data to compare against prior to the adversarial training.

[3] for brevity in the text, we will use the term parameters to refer to the weights and the biases from this point onwards.

Table 1 Generator/discriminator network architectures of 3DFaceGAN. As far as the notation is concerned, C denotes the number of input/output channels, K denotes the kernel size, S denotes the stride size, P denotes the padding size, AvgPool2D denotes average 2D pooling, UpNN denotes nearest-neighbor upsampling, and SF refers to the scaling factor size of the nearest-neighbor upsampling. CONV-BLOCK(C1, C2, K, S, P) and DECONV-BLOCK(C1, C2, K, S, P) refer to a block of two convolutions where the first is CONV(C1, C2, K, S, P) followed by an ELU (Clevert et al. 2016) activation function and the second is CONV(C2, C2, K, S, P), also followed by an ELU (Clevert et al. 2016) activation function

Part	Input → Output shape	Layer information
Encoder	(h, w, 3) → (h, w, n)	CONV-(Cn, K3x3, S1, P1), ELU
	(h, w, n) → ($\frac{h}{2}$, $\frac{w}{2}$, 2n)	CONV-BLOCK-(Cn, 2n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{2}$, $\frac{w}{2}$, 2n) → ($\frac{h}{4}$, $\frac{w}{4}$, 3n)	CONV-BLOCK-(C2n, C3n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{4}$, $\frac{w}{4}$, 3n) → ($\frac{h}{8}$, $\frac{w}{8}$, 4n)	CONV-BLOCK-(C3n, C4n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{8}$, $\frac{w}{8}$, 4n) → ($\frac{h}{16}$, $\frac{w}{16}$, 5n)	CONV-BLOCK-(C4n, C5n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{16}$, $\frac{w}{16}$, 5n) → ($\frac{h}{32}$, $\frac{w}{32}$, 6n)	CONV-BLOCK-(C5n, C6n, K3x3, S1, P1), AvgPool2D(K2x2, S2)
	($\frac{h}{32}$, $\frac{w}{32}$, 6n) → ($\frac{h}{32}$, $\frac{w}{32}$, 6n)	CONV-BLOCK-(C6n, C6n, K3x3, S1, P1)
Bottleneck_1	($\frac{h}{32} \times \frac{w}{32} \times$6n) → n	Fully connected
Bottleneck_2	n → ($\frac{h}{32} \times \frac{w}{32} \times$n)	Fully connected
Decoder	($\frac{h}{32}$, $\frac{w}{32}$, n) → ($\frac{h}{16}$, $\frac{w}{16}$, n)	DECONV-BLOCK(Cn, Cn K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{16}$, $\frac{w}{16}$, n) → ($\frac{h}{8}$, $\frac{w}{8}$, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{8}$, $\frac{w}{8}$, n) → ($\frac{h}{4}$, $\frac{w}{4}$, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{8}$, $\frac{w}{8}$, n) → ($\frac{h}{4}$, $\frac{w}{4}$, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	($\frac{h}{2}$, $\frac{w}{2}$, n) → (h, w, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1), UpNN(SF2)
	(h, w, n) → (h, w, n)	DECONV-BLOCK(Cn, Cn, K3x3, S1, P1)
	(h, w, n) → (h, w, 3)	DECONV(Cn, C3, K3x3, S1, P1), Tanh

less, it was empirically shown that enforcing the embeddings in the training process to follow a normal distribution leads to generators that are unable to capture high frequency details (Litany et al. 2018). To alleviate this, we propose to generate data using Algorithm 1, which better retains the generated data fidelity, as shown in Sect. 4.

3.4 3DFaceGAN for Multi-Label 3D Data

Over the last few years, databases annotated with regards to multiple labels are becoming available in the scientific community. For instance, 4DFAB (Cheng et al. 2018) is a publicly available 3D facial database containing data annotated with respect to multiple expressions.

We can extend 3DFaceGAN to handle data annotated with regards to multiple labels as follows. Without any loss of generality, suppose there are three labels in the database (e.g., expressions *neutral*, *happiness* and *surprise*). We adopt the so-called one-hot representation and thus denote the existence of a particular label in a datum by 1 and the absence by 0. For example, a 3D face datum annotated with the label *happiness* will have the following label representation: $l = [0, 1, 0]$, where the first entry corresponds to the label *neutral*, the second to the label *happiness* and the third to the label *surprise*. We then choose the desired l we want to generate (e.g., if we want to translate a neutral face to a surprised one, we would choose $l = [0, 0, 1]$) and then spatially replicate it and concatenate it in the input that is then fed to the generator. The real target is the actual expression (in this case *surprise*) with the corresponding l spatially replicated and concatenated. Apart from this change, the rest of the training process is exactly the same as the one described in Sect. 3.2.

Finally, to generate 3D facial data with respect to a particular label, we follow the same process as the one presented in Algorithm 1, with the only difference being that we extract different pairs of $(\boldsymbol{\mu}_Z, \boldsymbol{\Sigma}_Z)$ for every subset of the data, each corresponding to a particular label in the database. We then choose the pair $(\boldsymbol{\mu}_Z, \boldsymbol{\Sigma}_Z)$ corresponding to the desired label and sample from this multi-variate Gaussian distribution.

4 Experiments

In this Section we (a) describe the databases which we used to carry out the experiments utilizing 3DFaceGAN, (b) provide information with respect to the data preprocessing we conducted prior to feeding the 3D data into the network, (c) succinctly describe the baseline state-of-the-art algorithms we employed for comparisons and (d) provide quantitative as well as qualitative results on a series of experiments that demonstrate the superiority of 3DFaceGAN.

Algorithm 1: 3D face generation algorithm.

Step 1: Train 3DFaceGAN utilizing (3).
Step 2: Extract the trained G, and for all N training facial UV maps:
for $i = 1 : N$ **do**
  Input UV map $\mathbf{x}_i$ in G.
  Extract the corresponding bottleneck $\mathbf{z}_i \in \mathbb{R}^{N_b \times 1}$.
end
Step 3: Concatenate column-wise all of the bottlenecks, i.e., $\mathbf{Z} = [\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_N]$.
Step 4: Extract the mean $\boldsymbol{\mu}_Z$ of $\mathbf{Z}$ and the covariance $\boldsymbol{\Sigma}_Z$ of the zero-mean $\mathbf{Z}$.
Step 5: To generate new data, retain only the trained Bottleneck$_2$ and the Decoder part of G (see Table 1 for the network structures) and sample a new $\mathbf{z}_i$ (i.e., Bottleneck$_2$ input) from the multivariate Gaussian $\mathcal{N}(\boldsymbol{\mu}_Z, \boldsymbol{\Sigma}_Z)$.

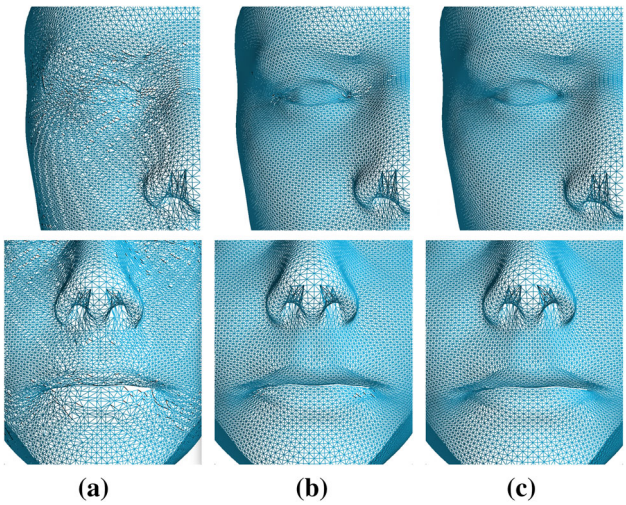

Fig. 5 Re-sampling errors of different UV sizes. The quantitative errors are 11.013 mm, 0.164 mm, 0.013 mm for the UV sizes of 128, 256, and 512, respectively, illustrated from left to right

Table 2 Generalization metric for the meshes of the test set for the 3D face representation task. The table reports the mean error (Mean), the standard deviation (std), the Area Under the Curve (AUC), and the Failure Rate (FR) of the Cumulative Error Distributions of Fig. 6a

Method	Mean	std	AUC	FR (%)
3DFaceGAN	0.0031	± 0.0028	0.741	1.42e−7
CoMA	0.0038	± 0.0037	0.716	3.66e−7
PCA	0.0040	± 0.0040	0.711	0.91e−6
PGAN	0.0041	± 0.0041	0.705	1.22e−6

4.1 Databases

4.1.1 The Hi-Lo database

Hi-Lo database contains approximately 6000 3D facial scans captured during a special exhibition in the Science Museum, London. It is divided into the high quality data (*Hi*) recorded with a 3dMD face capturing system and the low quality (*Lo*) data captured with a V1 Kinect sensor. All the subjects were recorded in neutral expression. The overlapping subjects that were recorded in both frameworks were approximately 3000.

The 3dMD apparatus utilizes a 4 camera structured light stereo system which can create 3D triangular surface meshes composed of approximately 60, 000 vertices joined into approximately 120, 000 triangles. Moreover, the low quality database was captured with a KinectFusion framework (Newcombe et al. 2011). In contrast to the 3dMD system, multiple frames are required to build a single 3D representation of the subject's face. The fused meshes were built by employing a 6083 voxel grid. In order to accurately reconstruct the entire surface of the faces, a circular motion scanning pattern was carried out. Each subject was instructed to stay still in a fixed pose during the entire scanning process with a neutral facial expression. The frame rate for every subject was constant at 8 frames per second.

Furthermore, all 3000 subjects provided metadata about themselves, including their gender, age, and ethnicity. The database covers a wide variety of age, gender (48% male, 52% female), and ethnicity (82% White, 9% Asian, 5% Mixed Heritage, 3% Black and 1% other).

Hi-Lo database was utilized for the experiments of 3D face representation and generation, where we utilized the high quality data to train 3DFaceGAN. Moreover, *Hi-Lo* database was used for demonstrating the capabilities of 3DFaceGAN in a 3D face translation setting, where the low quality data are translated into high quality ones. In all of the training tasks, 85% of the data were used for training and the rest were used for testing. The subjects in the training and testing sets are disjoint.

4.1.2 4DFAB Database

4DFAB database (Cheng et al. 2018) contains 3D facial data from 180 subjects (60 females, 120 males), aged from 5 to 75 years old. The subjects vary in their ethnicity background, coming from more than 30 different ethnic groups. For the capturing process, the DI4D dynamic capturing system[4] was used.

4DFAB (Cheng et al. 2018) contains data varying in expressions, such as *neutral*, *happiness*, and *surprise*. As a result, we utilized it to showcase 3DFaceGAN's capability in successfully handling data annotated with multiple labels in the task of 3D face translation as well as generation. In all of the training tasks, 85% of the data were used for training and the rest were used for testing. The subjects in the training and testing sets are disjoint.

[4] http://www.di4d.com.

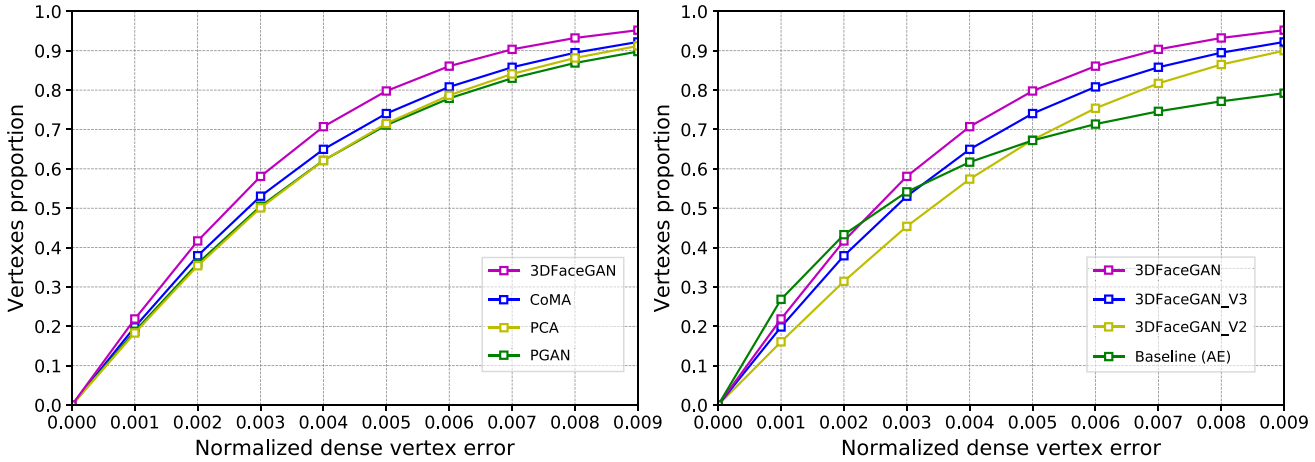

Fig. 6 **a** Generalization results on the test set for the 3D face representation task. The results are presented as cumulative error distributions of the normalized dense vertex errors. 3DFaceGAN outperforms all of the compared methods by a large margin. **b** Ablation study generalization results for the 3D face representation task. The results are presented as cumulative error distributions of the normalized dense vertex errors

Fig. 7 Qualitative results of 3DFaceGAN compared to CoMA (Ranjan et al. 2018) in the 3D representation task. Moreover, heatmaps are provided, visualizing the errors of both approaches against the ground truth test data. As evidenced, 3DFaceGAN is able to better capture the variation in the test data, especially in the eye and nose regions, where most of the non-linearities are present

4.2 Data Preprocessing

In order to feed the 3D data into a deep network several steps need to be carried out. Since we employ various databases, the representation of the facial topology is not consistent in terms of vertex number and triangulation. To this end, we need to find a suitable template T that can easily retain the information of all raw scans across all databases and describe them with the same triangulation/topology. We utilized the mean face mesh of the LSFM model proposed by Booth et al.

Table 3 Ablation study generalization results for the 3D face representation task. The table reports the Area Under the Curve (AUC) and Failure rate (FR) of the Cumulative error distributions of Fig. 6b

Method	AUC	FR (%)
3DFaceGAN	0.741	1.42e−7
3DFaceGAN_V3	0.736	2.62e−7
3DFaceGAN_V2	0.704	3.15e−6
Baseline (AE)	0.697	4.24−6

(2016), which consists of approximately 54, 000 vertices that are sufficient to capture high frequency facial details. We then bring the raw scans in dense correspondence by morphing non-rigidly the template mesh to each one of them. For this task, we utilize an optimal-step Non-rigid Iterative Closest Point algorithm (De Smet and Van Gool 2010) in combination with a per vertex weighting scheme. We weight the vertices according to the Euclidean distance measured from the tip of the nose. The greater the distance from the nose tip, the bigger the weight that is assigned to that vertex, i.e., less flexible to deform. In that way we are able to avoid the noisy information recorded by the scanners on the outer regions of the raw scans.

Following the analysis of the various methods of feeding 3D meshes in deep networks in Sect. 2, we chose to describe the 3D shapes in the UV domain. UV maps are usually utilized to store texture information. In our case, we store the spatial location of each vertex as an RGB value in the UV space. In order to acquire the UV pixel coordinates for each vertex, we start by unwrapping our mesh template T into a 2D flat space by utilizing an optimal cylindrical unwrapping technique proposed by Booth and Zafeiriou (2014). Before storing the 3D coordinates into the UV space, all meshes are aligned in the 3D spaces by performing the General Procrustes Analysis (Gower 1975) and are normalized to be in the scale of $[1, -1]$. Afterwards, we place each 3D vertex in the image plane given the respective UV pixel coordinate. Finally, after storing the original vertex coordinates, we perform a 2D nearest point interpolation in the UV domain to fill out the missing areas in order to produce a dense representation of the originally sparse UV map. Since the number of vertices in S_T is more than $50K$, we choose a $256 \times 256 \times 3$ tensor as the UV map size, which assists in retrieving a high precision point cloud with negligible re-sampling errors. A graphical representation of the preprocessing pipeline can be seen in Fig. 1.

Higher UV resolutions than $256 \times 256 \times 3$ will introduce more convolutions, thus more parameters, without any visible increments in accuracy of our final result. Figure 5 illustrates the re-sampled mesh topology (vertex position and triangulation) of the same identity for different UV sizes. The first column (a) depicts the re-sampled mesh topology from a

Fig. 8 Generated faces utilizing 3DFaceGAN

128 UV size, while columns (b) and (c) show the re-sampled meshes from 256 and 512 UV sizes, respectively. We can easily identify the misalignment of vertexes and the mixed triangulation in the 128 resolution, whereas, in column (b), the 256 UV size handles very well the topological structure of the mesh except in some very rare cases around the edges of the lips and the eyes where a large amount of points are tightly grouped together in small areas. We measured the re-sampling error with a point-to-point Euclidean distance and the results are as follows.

- 11.013 mm for a UV size of 128,
- 0.164 mm for a UV size of 256,
- 0.013 mm for a UV size of 512.

We also experimented with different interpolation methodologies. Instead of interpolating the 3D coordinates (x, y, z) in the UV domain, we performed the interpolation based on the barycentric coordinates of the underling pixel that is projected to the 3D shape. Although the resulting UV domains are interpolated differently, we did not record any change in the accuracy of our final result, which means that our network architecture is able to learn the underling structure of any UV domain regardless of the interpolation method.

4.3 Training

We trained all 3DFaceGAN models utilizing Adam (Kingma and Ba 2015) with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The batch size we used for the pre-training of the discriminator was 32 for a total of 300 epochs. The batch size we used for 3DFaceGAN

was 16 for a total of 300 epochs. For our model we used $n = 128$ convolution filters and a bottleneck of size $b = 128$. The total number of trainable parameters was 38.5×10^6. The learning rates that we used for both the pre-training and training of the discriminator was $5e-5$ and the same was for the training of the generator. We linearly decayed the learning rate by 5% every 30 epochs during training. For the rest of the parameters, we used $\lambda_{adv} = 1e-3$, $\lambda_{rec} = 1$. Overall training time on a GV100 NVIDIA GPU was about 5 days.

4.4 3D Face Representation

In the 3D face representation (reconstruction) experiments, we utilize the high quality 3D face data from the *Hi-Lo* database to train the algorithms. In particular, we feed the high quality 3D data as inputs to the models and use the same data as target outputs. Before providing the qualitative as well as quantitative results, we briefly describe the baseline models we compared against as well as provide information about the error metric we used for the quantitative assessment.

4.4.1 Baseline Models

In this Section we briefly describe the state-of-the-art models we utilized to compare 3DFaceGAN against.

Vanilla Autoencoder (AE)

Vanilla Autoencoder follows exactly the same structure of the discriminator we used in 3DFaceGAN. We used the same values for the hyper-parameters and the same optimization process. This is the main baseline we compared against and the results are provided in the ablation study in Sect. 4.4.3.

Convolutional Mesh Autoencoder (CoMA)

In order to train CoMA (Ranjan et al. 2018), we use the authors' publicly available implementation and utilize the default parameter values, the only difference being that the bottleneck size is 128, to make a fair comparison against 3DFaceGAN, where we also used a bottleneck size of 128.

Principal Component Analysis (PCA)

We employ and train a standard PCA model (Jolliffe 2011) based on the meshes of our database we used for training. We aimed at retaining the 98% of variance of our available training data which corresponds to the first 50 principal components.

Progressive GAN (PGAN)

In order to train PGAN (Karras et al. 2018), we used the authors' publicly available implementation with the default parameter values. After the training is complete, in order to represent a test 3D datum, we *invert* the generator G as in Lucic et al. (2018) and Mahendran and Vedaldi (2015), i.e., we solve $z^* = \text{argmin} \|x - G(z)\|$ by applying gradient descent on z while retaining G fixed (Mahendran and Vedaldi 2015).

4.4.2 Error Metric

A common practice when it comes to evaluating statistical shape models is to estimate the intrinsic characteristics, such as the *generalization* of the model (Davies et al. 2008). The *generalization* metric captures the ability of a model to represent *unseen* 3D face shapes during the testing phase. Table 2 presents the generalization metric for 3DFaceGAN compared against the baseline models. The Failure Rate mentioned in Table 2 and later on in the text, represents the frequency with which a method fails to represent a 3D face for a specific amount of vertices within a threshold (the bins of our graph) divided by the total number of bins. Essentially, it is the probability of failure given a threshold. Moreover, in order to compute the generalization error for a given model, we compute the per-vertex Euclidean distance between every sample of the test set and its corresponding reconstruction. We observe that the model which holds the best error results and thus demonstrates greater generalization capabilities is the proposed 3DFaceGAN with mean error 0.0031 and standard deviation 0.0028. Additionally, as shown in Fig. 6a, which depicts the cumulative error distribution of the normalized dense vertex errors, 3DFaceGAN outperforms all of the baseline models (Fig. 7).

4.4.3 Ablation Study

In this ablation study we investigate the importance of pre-training the discriminator D prior to the adversarial training of 3DFaceGAN as well as the freezing of the weights in the decoder parts of both D and G. More specifically, we compare 3DFaceGAN against the Vanilla Autoencoder (AE) and another two 3DFaceGAN possible variations, namely (a) the simplest case, where the discriminator and generator structures are retained as is, but *no* pre-training takes place prior to the adversarial training (we refer to this methodology as *3DFaceGAN_V2*), (b) the case where (i) the discriminator and generator structures are retained as is, (ii) we pre-train the discriminator and initialize both the generator and the discriminator with the learned weights with *no* parameters frozen during the adversarial training (we refer to this methodology as *3DFaceGAN_V3*). As shown in Fig. 6b and

Fig. 9 The qualitative results of our approach compared to state-of-the-art baseline GAN methods in the 3D face translation task. The first column depicts the low quality input mesh whereas the second column represent the high quality ground truth meshes. We depict the raw results of pix2pixHD (Wang et al. 2018) and pix2pix (Isola et al. 2017) along with their smoothed versions. As a smoothing technique we utilized a standard Laplacian smoothing operator

Table 3, 3DFaceGAN outperforms Vanilla AE and 3DFaceGAN_V2 by a large margin. Moreover, 3DFaceGAN also outperforms 3DFaceGAN_V3. As a result, not only does 3DFaceGAN have the best performance among the compared 3DFaceGAN variants, but it also requires less training time compared to 3DFaceGAN_V3, as the parameters in the

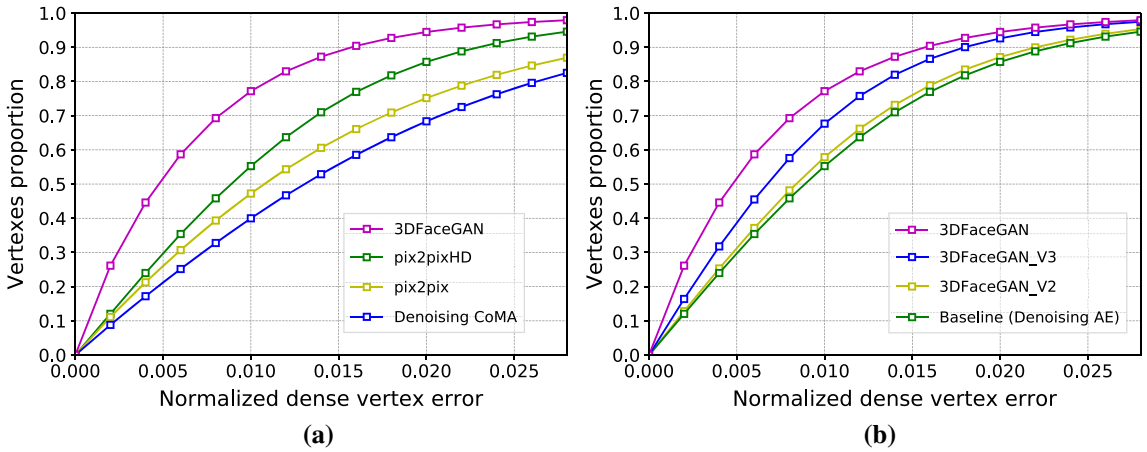

Fig. 10 **a** High quality estimation results for the 3D face translation task. The results are presented as cumulative error distributions of the normalized dense vertex errors. 3DFaceGAN outperforms all of the compared methods by a large margin. **b** Ablation study with respect to the 3D face translation task. The results are presented as cumulative error distributions of the normalized dense vertex errors

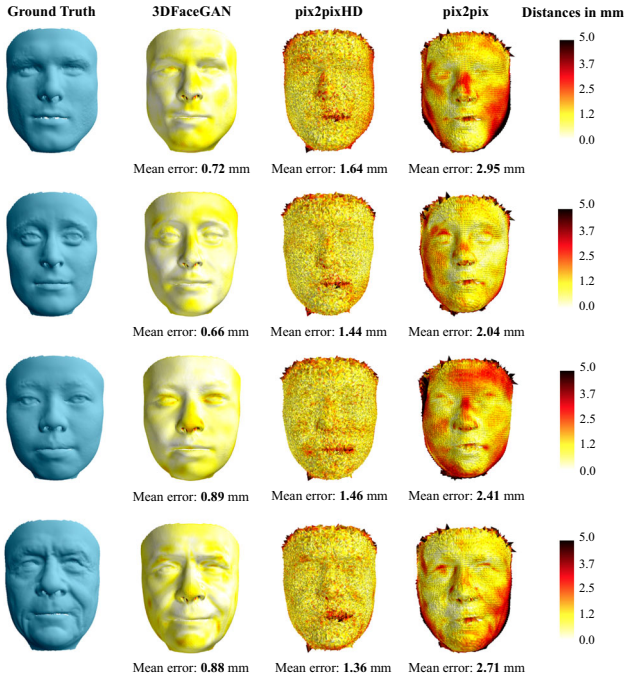

Fig. 11 Reconstruction quality of our proposed GAN network along with pix2pixHD (Wang et al. 2018) and pix2pix (Isola et al. 2017) in the 3D face translation task. As it can be seen, the mean error of 3DFaceGAN is considerably less than the other two approaches

decoder parts of both the generator and the discriminator are not updated during the training phase and thus need not be computed (Fig. 8).

4.5 3D Face Translation

In the 3D face translation experiments, we utilize the low and high quality 3D face data from the *Hi-Lo* database to train the algorithms. In particular, we feed the low quality 3D data as inputs to the models and use the high quality data as target outputs.

Table 4 High quality 3DRMSE results for the 3D face translation task. The table reports the Area Under the Curve (AUC) and failure rate of the cumulative error distributions of Fig. 10a

Method	AUC	Failure rate (%)
3DFaceGAN	0.827	5.49e−6
pix2pixHD	0.760	5.18e−5
pix2pix	0.757	1.81e−5
Denoising CoMA	0.742	2.41e−4

Table 5 Ablation study 3DRMSE results for the 3D face translation task. The table reports the Area Under the Curve (AUC) and failure rate of the cumulative error distributions of Fig. 10b

Method	AUC	Failure rate (%)
3DFaceGAN	0.827	5.49e−6
3DFaceGAN_V3	0.819	8.70e−6
3DFaceGAN_V2	0.794	1.38e−5
Baseline (Denoising AE)	0.758	1.95e−5

Before providing the qualitative as well as quantitative results, we briefly describe the baseline models we compared against as well as provide information about the error metric we used for the quantitative assessment.

4.5.1 Baseline Models

In this Section we briefly describe the state-of-the-art deep models we utilized to compare 3DFaceGAN against.

Fig. 12 Qualitative results of our approach compared to state-of-the-art baseline GAN methods in the multi-label 3D face translation task in various expressions (e.g., happiness, surprise) trained with the 4DFAB (Cheng et al. 2018) database. The first column depicts the neutral input mesh whereas the rest of the columns represent the translated meshes of the respective state-of-the art methods compared to our approach. As can be seen, 3DFaceGAN is able to retain the high-frequency details in a higher level compared to CoMA (Ranjan et al. 2018), the second best method, which produces more smoothed outputs

Denoising Vanilla Autoencoder (Denoising AE)

Denoising Vanilla Autoencoder follows exactly the same structure as the Vanilla AE in Sect. 4.4, the only difference being the inputs fed to the network (i.e., we feed the low quality 3D data as inputs to the model and use the high quality data as target outputs). This is the main baseline we compared against and the results are provided in the ablation study in Sect. 4.5.3.

Denoising Convolutional Mesh Autoencoder (Denoising CoMA)

Denoising CoMA (Ranjan et al. 2018), follows exactly the same structure as CoMA (Ranjan et al. 2018) in Sect. 4.4, the only difference being again the inputs fed to the network (i.e., we feed the low quality 3D data as inputs to the model and use the high quality data as target outputs).

pix2pix

pix2pix (Isola et al. 2017) is amongst the most widely utilized GANs for image to image translation applications. We used the official implementation and hyper-parameter initializations provided by the authors in (Isola et al. 2017).

pix2pixHD

More recently pix2pixHD (Wang et al. 2018) was proposed, which can be considered as an extension of pix2pix (Isola

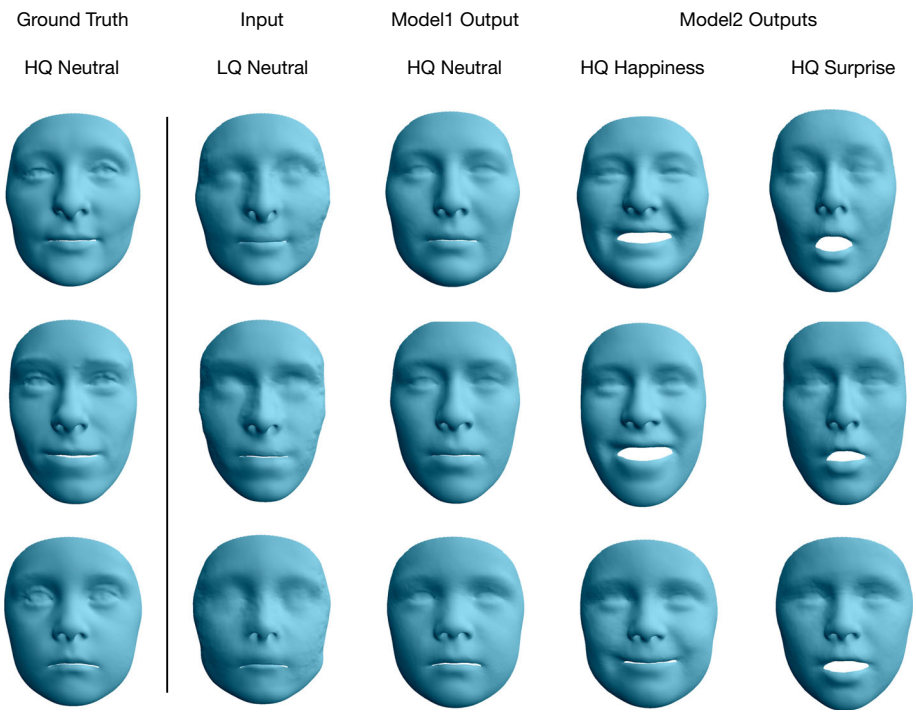

Fig. 13 Some visualizations of the cross-dataset experiment. As explained in Sect. 4.7, we utilize the Hi-Lo database to train a model that transfers the *neutral* low-quality (LQ) 3D faces to the corresponding high-quality (HQ) ones (Model1). Then, we train a model that transfers the generated HQ *neutral* 3D faces to the desired expressions, such as *happiness* or *surprise* (Model2). Please note that the 3D expression translation model is trained solely on 4DFAB (Cheng et al. 2018) as the Hi-Lo database contains subjects only in *neutral* expression. Since the depicted low-quality inputs do not give away much information about the identity of each subject, we introduce the leftmost column which shows the *neutral* HQ, ground truth 3D faces of the Hi-Lo dataset. In this way, it is easier for the reader to observe that the identity information is retained throughout the whole experiment. In sum, as it is evidenced, 3DFaceGAN can be utilized in two different datasets and then the corresponding trained models can be combined to transfer attributes between the datasets

et al. 2017) and which is able to better handle data of higher resolution. We used the official implementation and hyper-parameter initializations provided by the authors in Wang et al. (2018). As evinced in Figs. 9, 10, and 11, pix2pixHD (Wang et al. 2018) outperforms pix2pix (Isola et al. 2017), and this is expected since pix2pixHD (Wang et al. 2018) uses more intricate structures for both the generator and discriminator networks.

4.5.2 Error Metric

For each low quality test mesh we aim to estimate the high quality representation based on the 3dMD ground truth data. The error metric between the estimated and the real high quality mesh is a standard 3D Root Mean Square Error (3DRMSE) where the Euclidean distances are computed between the two meshes and normalized based on the inter-ocular distance of the test mesh. Before computing the metric error we perform dense alignment between each test mesh and its corresponding ground truth by implementing an iterative closest point (ICP) algorithm (Besl and McKay 1992). In particular, we utilized the anisotropic implementation of the ICP algorithm (Maier-Hein et al. 2011). In order to avoid any inconsistencies in the alignment we compute a point-to-plane rather than a point-to-point error. Finally, the measurements are performed in the inner part of the face, where we crop each test mesh at a radius of 150*mm* around the tip of the nose. As can be clearly seen in Fig. 10a as well as in Table 4, 3DFaceGAN outperforms all of the compared state-of-the-art methods.

4.5.3 Ablation Study

For the ablation study in this set of experiments, we use exactly the same 3DFaceGAN variants as the ones we utilized in Sect. 4.4.3. Moreover, instead of the vanilla AE in this experiment we utilize the denoising AE. As evinced in Fig. 10b and Table 5, 3DFaceGAN clearly outperforms all of the compared models.

4.6 Multi-label 3D Face Translation

In this experiment we utilize 4DFAB (Cheng et al. 2018) for the multi-label transfer of expressions. In particular, we

feed the *neutral* faces to the models and receive as outputs either the ones bearing the label *happiness* or *surprise*. It should be noted here that whereas 3DFaceGAN requires only a single model to be trained under the multi-label expression translation scenario, the rest of the compared models require different trained models for each label, i.e., a model for expression *happiness* and a model for expression *surprise*. As baseline models for comparisons, we use exactly the same as the ones in Sect. 4.5, the only difference being the inputs fed to network as well as the corresponding targets. Qualitative comparisons against the compared methods are presented in Fig. 12.

4.7 Cross-Dataset Attribute Transfer

In this experiment, we combine both of the databases we utilized so far, i.e., the Hi-Lo and the 4DFAB database (Cheng et al. 2018). Hi-Lo is a database which contains 3D face shapes of low and high quality in *neutral* expression. On the other hand, 4DFAB (Cheng et al. 2018) is a database that contains *only* high quality 3D face shapes in various expressions. As a result, we carried out the cross dataset experiment as follows: we train 3DFaceGAN as we did in Sect. 4.5 with the training data of the Hi-Lo database (Model1). We also train 3DFaceGAN as we did in Sect. 4.6 with the training data of the 4DFAB database (Cheng et al. 2018) (Model2). We then combine both of the trained models as follows. First, we pass the test data of the Hi-Lo database through the first model (Model1) and subsequently generate the high-quality *neutral* outputs. We then feed the generated, high-quality *neutral* outputs to the second model (Model2) and, by using the mechanism described in Sect. 4.6, we generate new expressions, depending on the label we desire. Some qualitative results of this procedure are presented in Fig. 13, where we generate high-quality expressions such as *surprise* or *happiness* from low quality *neutral* inputs.

4.8 3D Face Generation

In the 3D face generation experiment, we utilized the high quality data of the *Hi-Lo* database to train the algorithms. In particular, we feed the high quality 3D data as inputs to the models and use the same data as target outputs.

4.8.1 Baseline Models

The baseline models we used in this set of experiments are the same as the ones presented in Sect. 4.4.

4.8.2 Error Metric

The metric of choice to quantitatively assess the performance of the models in this set of experiments is *specificity* (Brunton et al. 2014). For a randomly generated 3D face, *specificity* metric measures the distance of this 3D face to its nearest real 3D face belonging in the test, in terms of minimum per vertex distance over all samples of the test set. To evaluate this metric, we randomly generate $n = 10,000$ face meshes from each model. Table 6 reports the specificity metric for 3DFaceGAN compared against the baseline models. In order to generate random meshes utilizing 3DFaceGAN, we sample from a multivariate Gaussian distribution, as explained in Sect. 3.3. To generate random meshes utilizing PGAN (Karras et al. 2018), we sample new latent embeddings from the multivariate normal distribution and feed them to the generator G. To generate random faces utilizing CoMA (Ranjan et al. 2018), we utilize the proposed variational convolutional mesh autoencoder structure, as described in (Ranjan et al. 2018). For the PCA model (Jolliffe 2011), we generate meshes directly from the latent eigenspace by drawing random samples from a Gaussian distribution defined by the principal eigenvalues. As shown in Table 6, 3DFaceGAN

Table 6 Specificity metric on the test set for the 3D face generation task. We generate 10,000 random faces from each model. The table reports the mean error (Mean) and the standard deviation (std)

Method	Mean	std
3DFaceGAN	1.28	±0.183
CoMA	1.40	±0.205
PCA	1.43	±0.232
PGAN	1.79	±0.189

Fig. 14 Generated faces with expression utilizing 3DFaceGAN multi-label approach

achieves the best specificity error, outperforming all compared methods by a large margin.

In Fig. 8, we present various visualizations of realistic 3D faces generated by 3DFaceGAN. As can be clearly seen, 3DFaceGAN is able to generate data varying in ethnicity, age, etc., thus capturing the whole population spectrum.

4.9 Multi-label 3D Face Generation

In this set of experiments, we utilized the 4DFAB (Cheng et al. 2018) data to generate random subjects of various expressions such as *happiness* and *surprise*, as seen in Fig. 14. The 3D faces were generated utilizing the methodology detailed in Sect. 3.4. As evinced, 3DFaceGAN is able to generate expressions of subjects varying in age and ethnicity, while retaining the high-frequency details of the 3D face.

5 Conclusion

In this paper we presented the first GAN tailored for the tasks of 3D face representation, generation, and translation. Leveraging the strengths of autoencoder-based discriminators in an adversarial framework, we propose 3DFaceGAN, a novel technique for training on large-scale 3D facial scans. As shown in an extensive series of quantitative as well as qualitative experiments against other state-of-the-art deep networks, 3DFaceGAN improves upon state-of-the-art algorithms for the tasks at-hand by a significant margin.

Acknowledgements Stylianos Moschoglou is supported by an EPSRC DTA studentship from Imperial College London, Stylianos Ploumpis by the EPSRC Project EP/N007743/1 (FACER2VM), and Stefanos Zafeiriou by the EPSRC Project EP/S010203/1 (DEFORM).

References

Berthelot, D., Schumm, T., & Metz. L. (2017). Began: Boundary equilibrium generative adversarial networks. arXiv preprint arXiv:1703.10717

Besl, P. J., & McKay, N. D. (1992). Method for registration of 3-D shapes. *Sensor Fusion IV: Control Paradigms and Data Structures*, *1611*, 586–607.

Booth, J., & Zafeiriou, S. (2014). Optimal uv spaces for facial morphable model construction. In *Proceedings of the IEEE international conference on image processing (ICIP)*, (pp. 4672–4676).

Booth, J., Roussos, A., Zafeiriou, S., Ponniah, A., & Dunaway, D. (2016). A 3D morphable model learnt from 10,000 faces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 5543–5552).

Bousmalis, K., Silberman, N., Dohan, D., Erhan, D., & Krishnan, D. (2017). Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (vol. 1, p. 7).

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., & Vandergheynst, P. (2017). Geometric deep learning: Going beyond Euclidean data. *IEEE Signal Processing Magazine*, *34*(4), 18–42.

Brunton, A., Salazar, A., Bolkart, T., & Wuhrer, S. (2014). Review of statistical shape spaces for 3D data with comparative analysis for human faces. *Computer Vision and Image Understanding*, *128*, 1–17.

Cheng, S., Kotsia, I., Pantic, M., & Zafeiriou, S. (2018). 4DFAB: A large scale 4D database for facial expression analysis and biometric applications. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (pp. 5117–5126).

Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., & Choo, J. (2018). Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 8789–8797).

Clevert, D.A., Unterthiner, T., & Hochreiter, S. (2016). Fast and accurate deep network learning by exponential linear units (elus). In *Proceedings of the international conference for learning representations (ICLR)*.

Davies, R., Twining, C., & Taylor, C. (2008). *Statistical models of shape: Optimization and evaluation*. Berlin: Springer.

De Smet, M., & Van Gool, L. (2010) .Optimal regions for linear model-based 3D face reconstruction. In *Proceedings of the Asian conference on computer vision*, (pp. 276–289).

Dosovitskiy, A., & Brox, T. (2016). Generating images with perceptual similarity metrics based on deep networks. In *Proceedings of the advances in neural information processing systems (NIPS)*, (pp. 658–666).

Dou, P., Shah, S.K., & Kakadiaris, I.A. (2017). End-to-end 3D face reconstruction with deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (pp. 21–26).

Fan, H., Su, H., & Guibas, L.J. (2017). A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (vol. 2, p. 6).

Feng, Y., Wu, F., Shao, X., Wang, Y., & Zhou, X. (2018). Joint 3D face reconstruction and dense alignment with position map regression network. In *Proceedings of the European conference on computer vision (ECCV)*, (pp. 534–551).

Genova, K., Cole, F., Maschinot, A., Sarna, A., Vlasic, D., & Freeman, W.T. (2018). Unsupervised training for 3D morphable model regression. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (pp. 8377–8386).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In *Proceedings of the advances in neural information processing systems*, (pp. 2672–2680).

Gower, J. C. (1975). Generalized procrustes analysis. *Psychometrika*, *40*(1), 33–51.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference*

on computer vision and pattern recognition (CVPR), (pp. 770–778).

Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K.Q. (2017). Densely connected convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (vol. 1, p. 3).

Isola, P., Zhu, J.Y., Zhou, T., & Efros, A.A. (2017). Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 1125–1134).

Jackson, A.S., Bulat, A., Argyriou, V., & Tzimiropoulos, G. (2017). Large pose 3D face reconstruction from a single image via direct volumetric cnn regression. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, (pp. 1031–1039).

Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual losses for real-time style transfer and super-resolution. In *Proceedings of the European conference on computer vision*, Springer, (pp. 694–711).

Jolliffe, I. (2011). Principal component analysis. In M. Lovric (Ed.), *International encyclopedia of statistical science* (pp. 1094–1096). Berlin: Springer.

Karras, T., Aila, T., Laine, S., & Lehtinen, J. (2018). Progressive growing of gans for improved quality, stability, and variation. In *Proceedings of the International Conference for Learning Representations (ICLR)*.

Kim, T., Cha, M., Kim, H., Lee, J.K., & Kim, J. (2017). Learning to discover cross-domain relations with generative adversarial networks. In *Proceedings of the 34th international conference on machine learning*, (vol. 70, pp. 1857–1865).

Kingma, D.P., & Ba, J. (2015). Adam: A method for stochastic optimization. In *Proceedings of the 3rd international conference for learning representations (ICLR)*.

Kingma, D.P., & Welling, M. (2014). Auto-encoding variational Bayes. In *Proceedings of the International Conference for Learning Representations (ICLR)*.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A.P., Tejani, A., Totz, J. &, Wang, Z., et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (vol. 2, p. 4).

Lei, T., Jin, W., Barzilay, R., & Jaakkola, T. (2017). Deriving neural architectures from sequence and graph kernels. In *Proceedings of the 34th international conference on machine learning*, (vol. 70, pp. 2024–2033).

Li, Y., Liu, S., Yang, J., & Yang, M.H. (2017). Generative face completion. In *Proceedings of the the IEEE conference on computer vision and pattern recognition (CVPR)*, (vol. 1, p. 3).

Litany, O., Remez, T., Rodolà, E., Bronstein, A.M., & Bronstein, M.M. (2017). Deep functional maps: Structured prediction for dense shape correspondence. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, (pp. 5660–5668).

Litany, O., Bronstein, A., Bronstein, M., & Makadia, A. (2018). Deformable shape completion with graph convolutional autoencoders. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 1886–1895).

Liu, F., Tran, L., & Liu, X. (2019). 3D face modeling from diverse raw scan data. arXiv preprint arXiv:1902.04943.

Lucic, M., Kurach, K., Michalski, M., Gelly, S., & Bousquet, O. (2018). Are gans created equal? a large-scale study. In *Proceedings of the Advances in Neural Information Processing Systems (NIPS)*.

Mahendran, A., & Vedaldi, A. (2015). Understanding deep image representations by inverting them. In*Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (pp. 5188–5196).

Maier-Hein, L., Franz, A. M., Dos Santos, T. R., Schmidt, M., Fangerau, M., Meinzer, H. P., et al. (2011). Convergent iterative closest-point algorithm to accomodate anisotropic and inhomogenous localization error. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *34*(8), 1520–1532.

Maron, H., Galun, M., Aigerman, N., Trope, M., Dym, N., Yumer, E., et al. (2017). Convolutional neural networks on surfaces via seamless toric covers. *ACM Transactions on Graphics*, *36*(4), 71.

Mirza, M., & Osindero, S. (2014). Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784.

Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohi, P., Shotton, J., Hodges, S., & Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. In *Proceedings of the IEEE international symposium on Mixed and Augmented Reality (ISMAR)*, (pp. 127–136).

Nguyen, K., Fookes, C., Sridharan, S., Tistarelli, M., & Nixon, M. (2018). Super-resolution for biometrics: A comprehensive survey. *Pattern Recognition*, *78*, 23–42.

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3D classification and segmentation. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, *1*(2), 4.

Radford, A., Metz, L., & Chintala, S. (2016). Unsupervised representation learning with deep convolutional generative adversarial networks. In *Proceedings of the international conference for learning representations (ICLR)*.

Ranjan, A., Bolkart, T., Sanyal, S., & Black, M.J. (2018). Generating 3D faces using convolutional mesh autoencoders. In *Proceedings of the European conference on computer vision (ECCV)*, (pp. 704–720).

Richardson, E., Sela, M., Or-El, R., & Kimmel, R. (2017). Learning detailed face reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (pp. 5553–5562).

Tewari, A., Zollhöfer, M., Garrido, P., Bernard, F., Kim, H., Pérez, P.,& Theobalt, C. (2018). Self-supervised multi-level face model learning for monocular reconstruction at over 250 Hz. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 2549–2559).

Tewari, A., Bernard, F., Garrido, P., Bharaj, G., Elgharib, M., Seidel, H.P., Pérez, P., Zollhofer, M., & Theobalt, C. (2019) Fml: face model learning from videos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 10812–10822).

Tran, A.T., Hassner, T., Masi, I., & Medioni, G. (2017). Regressing robust and discriminative 3D morphable models with a very deep neural network. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (pp. 1493–1502).

Tran, L., & Liu, X. (2018). Nonlinear 3D face morphable model. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 7346–7355).

Tzeng, E., Hoffman, J., Saenko, K., & Darrell, T. (2017). Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference computer vision and pattern recognition (CVPR)*, (vol. 1, p. 4).

Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., & Catanzaro, B. (2018). High-resolution image synthesis and semantic manipulation with conditional gans. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (vol. 1, p. 5).

Wang, W., Huang, Q., You, S., Yang, C., & Neumann, U. (2017). Shape inpainting using 3d generative adversarial network and recurrent convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, (pp. 2298–2306).

Yang, C., Lu, X., Lin, Z., Shechtman, E., Wang, O., & Li, H. (2017). High-resolution image inpainting using multi-scale neural patch synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, (vol. 1, p. 3).

Zhao, J., Mathieu, M., & LeCun, Y. (2017). Energy-based generative adversarial network. In *Proceedings of the international conference for learning representations (ICLR)*.

Zhu, J.Y., Park, T., Isola, P., & Efros, A.A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, (pp. 2223–2232).