Multilinear Autoencoder for 3D Face Model Learning

Victoria Fernández Abrevaya, Stefanie Wuhrer, Edmond Boyer

Multilinear Autoencoder for 3D Face Model Learning

Victoria Fernández Abrevaya Stefanie Wuhrer Edmond Boyer
Inria, Université Grenoble Alpes (LJK), France
{victoria.fernandez-abrevaya, stefanie.wuhrer, edmond.boyer}@inria.fr

Abstract

Generative models have proved to be useful tools to represent 3D human faces and their statistical variations. With the increase of 3D scan databases available for training, a growing challenge lies in the ability to learn generative face models that effectively encode shape variations with respect to desired attributes, such as identity and expression, given datasets that can be diverse. This paper addresses this challenge by proposing a framework that learns a generative 3D face model using an autoencoder architecture, allowing hence for weakly supervised training. The main contribution is to combine a convolutional neural network-based encoder with a multilinear model-based decoder, taking therefore advantage of both the convolutional network robustness to corrupted and incomplete data, and of the multilinear model capacity to effectively model and decouple shape variations. Given a set of 3D face scans with annotation labels for the desired attributes, e.g. *identities and expressions, our method learns an expressive multilinear model that decouples shape changes due to the different factors. Experimental results demonstrate that the proposed method outperforms recent approaches when learning multilinear face models from incomplete training data, particularly in terms of space decoupling, and that it is capable of learning from an order of magnitude more data than previous methods.*

1. Introduction

Generative models of 3D faces are commonly used as priors when solving under-constrained problems such as reconstructing 3D face shape from partial or noisy data, recognition of faces or expressions, and face or expression transfer among others, *e.g.* [6]. They proved to be beneficial in these tasks as they provide parametric representations for sampling 3D face models which can differentiate changes due to natural factors including identity, expression or even age, *e.g.* [26, 1]. An important challenge is then to learn these models from datasets that can be diverse and so that they effectively encode shape variations with respect to desired attributes. This is especially true with the recent increase of available databases of 3D face scans that can be used as training data, *e.g.* [18].

Methods that build generative face models should therefore ideally present the following characteristics. First, the methods should leverage all available training data, which prohibits strategies assuming specific factor representativity in the data, *e.g.* a complete training data tensor that captures all combinations of identities and expressions. Second, the methods should handle data corrupted by noise including both geometric noise and erroneous labels for the factors. Third, the methods should require little or no preprocessing, and hence avoid the need for accurate registrations of the training data. Fourth, the resulting generative models should encode the rich shape variation of 3D faces while decoupling the effects of the different factors considered.

In this paper, we take a step towards achieving these goals by proposing a novel framework that learns a generative 3D face model using an autoencoder architecture. The main innovation is to combine a convolutional neural network (CNN)-based encoder with a multilinear model decoder. By leveraging that classical autoencoders are unsupervised, our modified autoencoder framework allows for weakly supervised model learning. It further combines the advantages of convolutional networks of being robust to corrupted and incomplete data, as assured by the encoder, with the advantages of multilinear models of effectively modeling and decoupling shape variations, over data attributes, in the decoder. Our approach inherits the advantage of autoencoders of being scalable for large datasets. Moreover, using a multilinear model as decoder rather than a CNN allows our approach to explicitly take advantage of redundant training data showing the same factor, and to effectively decouple shape variations in the learned representation. Note that while we choose a multilinear model for the decoder, our architecture easily generalizes to other generative models with similar properties.

Our approach builds on recent works that use deep neural networks for 3D face modeling. In particular, two of them [16, 24] have successfully explored the combination of a CNN-based encoder with a linear generative model as decoder for the 3D reconstruction of faces from 2D photos and videos. We follow a similar strategy however, unlike Tewari *et al.* [24] our decoder is learned with the rest of the network, and unlike Laine *et al.* [16], our learned model generalizes to various factors captured for different subjects.

Our method takes as input a set of 3D face scans annotated with labels for each factor, *e.g.* identities and expressions are given, and provides: (i) A multilinear model, which is able to accurately reconstruct the training data and decouples shape changes due to different factors; (ii) Improved registrations of the training data; (iii) A trained autoencoder with a CNN-based encoder and a multilinear model decoder capable of regressing from any 3D face scan to the registered model, thus allowing to efficiently compute correspondences for new data.

Our model performs favorably against other recent approaches that learn multilinear face models from incomplete training data tensors [4, 27]. Especially, we show experimentally that our method is capable of building rich models which achieve a better decoupling of factors. This is demonstrated by a classification rate of synthetically transferred expressions that is over 13% higher than for the competing methods. While experiments in this paper focus on identity and expression attributes present in the training data, our formalism readily generalizes to other factors as well such as age. Our code and models are available at `http://mae.gforge.inria.fr/`.

2. Related Work

There is an extensive amount of work on 3D human face modeling and recognition, and a full review is beyond the scope of this work. In the following, we focus on the works most closely related to the proposed method.

Generative modeling of 3D faces Linear models have first been introduced to model face shape in neutral expression along with appearance information [3] and later been extended to include expression change as a linear factor [1]. These linear models are often called 3D morphable models (3DMM), and such models have recently been learned from large training sets [5] and with high-quality appearance information [9]. These models do not account for correlations of expression and identity spaces.

Multilinear models were introduced to model the influence on face shape of different factors as independent, which allows for expression transfer [26]. They were later used to edit 2D images and videos with the help of 3D face reconstructions [10, 6]. FaceWarehouse [6] is a popular publicly available multilinear 3D face model. While multilinear models effectively decouple shape variations due to different factors, they require carefully acquired training data where each subject is captured in every factor.

Li *et al.* [18] recently introduced a generative model learned from a large collection of 3D motion sequences of faces. Pose changes due to skeletal motion is modeled using a skinning approach, while shape changes due to identity, expression, and pose correction are modeled as linear factors similar to 3DMM. Interestingly, they note that it is an open problem to extend tensor-based multilinear models to handle dynamic training data.

We take a step in this direction by deriving an efficient method to learn a multilinear model, from an incomplete tensor of training data, that effectively decouples factor effects.

Learning a multilinear model from partial or noisy data Traditionally, multilinear models are learned by assembling a set of training data into a tensor and performing a tensor decomposition [17]. This requires each training face to be present in all factors to be modeled. Furthermore, noise in the data, registration or labeling affect the quality of the model. While tensor completion methods can be used to solve the problem of incomplete data, they do not scale well in practice, especially if the tensor is dense as in our case.

Two recent methods were proposed to address these problems. A groupwise optimization was proposed to handle both missing and noisy data and was shown to outperform tensor completion methods [4]. However, this method is computationally costly and hence does not scale to large datasets with high dimensionality in two or more factors. Another work proposed an unsupervised method to compute a multilinear model from partial data [27]. While this method is computationally more efficient, it uses a non-standard tensor decomposition that leads to a generative model that does not fully decouple the modes.

We present in this paper a scalable solution to this problem.

Deep neural networks for 3D face modeling Deep neural networks have experimentally been shown to summarize large groups of data and automatically extract only the relevant features for a large variety of problems. They provide an efficient structure for the optimization of large datasets. This motivates the use of deep learning as a more scalable and robust approach with such datasets.

Recent works use CNN frameworks to recover detailed 3D models from a single input photograph [30, 20, 21, 25]. To represent the solution space a linear 3DMM is used, which restricts the solution from being very detailed. Richardson *et al.* [21] improve the initial 3DMM estimate with the help of a fine-scale network that allows to recover mid-scale facial detail. CNNs have also been used for shape regression of 3D faces in-the-wild [13].

A similar recent line of works have explored combining a CNN-based encoder with a generative model as decoder for the problem of 3D face reconstruction from 2D photos and videos [16, 24]. Unlike our method, these works use linear models to represent 3D faces, which captures limited expression variation w.r.t. tensor-based models.

In most of these works the linear solution space is learned a-priori and fixed during training. A notable exception is Laine *et al.* [16], in which the linear 3DMM is initialized with principal component analysis, and refined during fine-tuning of the network. The model trained by Laine *et al.* is person-specific and does not generalize to new subjects.

Sela *et al.* [23] propose a model-free network that directly regresses from an input image to a depth image and a correspondence map, thereby allowing to recover fine-scale geometric detail. However, as no model is used, noise, accessories and facial hair not present in the training data cannot be handled by the network.

Inspired by these works, our approach takes a middle-ground between the use of a linear model and a model-free approach to represent 3D faces. We choose multilinear models as they offer rich representations for various factors across different subjects [6].

3. Overview

The goal of our method is to learn a generative model of faces from a set of labeled 3D scans that are possibly corrupted by both geometric noise and label errors. To achieve this goal, we propose an autoencoder architecture with a CNN-based encoder and a multilinear model-based decoder, as illustrated in Figure 1 and detailed in the following section.

Input Data To train the autoencoder we consider 3D face scans showing variations in different factors, *e.g.* identity and expression, along with the corresponding labels. Not all combinations of factors are required in the input scans, and part of the training data can come without labels. The input scans are registered using a non-rigid approach, *e.g.* [2, 18], which enables reconstruction errors between the output meshes and the input scans to be estimated in a consistent way. These registrations need not be precise since they only serve as initialization and will be refined.

Encoder The CNN encoder maps each 3D face scan into a low-dimensional representation that decouples the influence of the different factors on the final shape. Extending CNNs to unorganized 3D geometric data is an active field of research (*e.g.* [19]) and beyond the scope of this work. Instead, we take advantage of the fact that 3D faces can be mapped onto 2D depth images for which regular CNN apply. Hence, the first step of the encoder is to project input 3D scans into grayscale images that contain depth information. The remainder of the encoder consists of a ResNet-18 architecture [14] followed by three fully connected layers, which transform depth images into d-dimensional vectors with the concatenated coefficients for each mode.

Decoder The multilinear decoder splits the encoder's output according to the factors, applies mode-n multiplication between latent vectors and the core tensor, and adds a previously computed mean face, as usually done with multilinear models (see Section 4.1). The output of the decoder are 3D vertex coordinates that, combined with the mean face connectivity, define 3D face meshes.

Training In addition to a generative loss that accounts for reconstruction errors, the training phase optimizes also a latent loss that measures whether input faces with the same labels are mapped onto close-by points in the parameter space, hence enforcing shape variations to be decoupled with respect to their factors in the latent space. The space that models face variations is large compared to the available training data and a good initialization is required to learn it. To this aim, both encoder and decoder are pre-trained, as will be detailed in Section 4.4.

Once the autoencoder has been trained, it can be used to regress from any 3D face scan to the model, thereby allowing to efficiently register new data.

4. Multilinear Autoencoder

We now describe the proposed autoencoder architecture that allows to learn k modes of variation in the input face data through a multilinear model.

4.1. Multilinear Model

In a multilinear model a face is represented by a set of vectors $\{\mathbf{w}_2^T, \mathbf{w}_3^T, \ldots, \mathbf{w}_{k+1}^T\}$, $\mathbf{w}_j \in \mathbb{R}^{m_j}$, where k is the number of linear modes attached to faces in the model. Let $\mathbf{x}$ be the vector of 3D coordinates associated to the n vertices of a face mesh, then the multilinear model relates the latent k factors $\mathbf{w}_i$ with the 3D face $\mathbf{x}$ by:

$$\mathbf{x} = \bar{\mathbf{x}} + \mathcal{M} \times_2 \mathbf{w}_2^T \times_3 \mathbf{w}_3^T \ldots \times_{k+1} \mathbf{w}_{k+1}^T, \qquad (1)$$

where $\bar{\mathbf{x}}$ is the mean face, $\mathcal{M} \in \mathbb{R}^{3n \times m_2 \times m_3 \times \ldots m_{k+1}}$ is a tensor that combines the linear modes $\mathbf{w}_j$ called the *core tensor* of the multilinear model, and $\times_j$ is the product of $\mathcal{M}$ and a vector along mode j [15]. The full model is therefore represented by the entries of $\mathcal{M}$ in addition to the set of coefficients $\mathbf{w}_j^{(i)}$ for the i-th face and the j-th factor.

The training process seeks to obtain good reconstructions of the data, while at the same time decoupling the latent representation with respect to the factors of variation. Hence, we will use two loss functions: a geometric loss that measures the reconstruction error, and a latent loss that

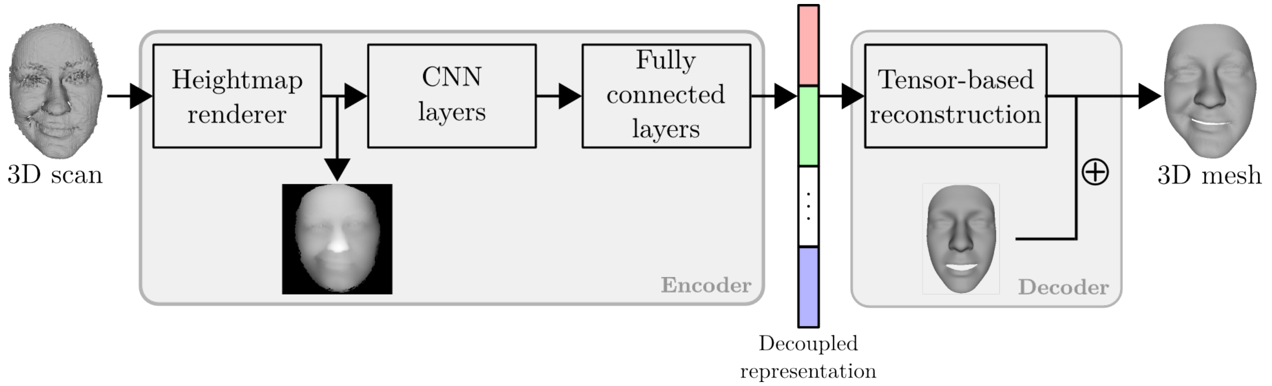


Figure 1: Architecture for our multilinear autoencoder. The encoder takes as input a 3D mesh, which is rendered into a heightmap, processed by a deep CNN architecture, and transformed into a latent representation by the fully-connected layers. The decoder splits the latent representation according to the specified factors and performs a multilinear transformation in order to get the output mesh. Both encoder and decoder are optimized during training.

softly evaluates how decoupled the latent space is, by measuring how close two embeddings with the same label are.

Generative loss The loss of a generative multilinear model over a dataset $\mathbf{X}$ of faces is measured as the error between the reconstructions by the generative model (see Equation 1) and the observed faces $\mathbf{x}_i$:

$$\mathcal{L}_G = \sum_{\mathbf{x}_i \in \mathbf{X}} \left\| \mathbf{x}_i - \left(\bar{\mathbf{x}} + \mathcal{M} \times_2 \mathbf{w}_2^{(i)} \dots \times_{k+1} \mathbf{w}_{k+1}^{(i)} \right) \right\|_2^2, \tag{2}$$

As shown by Wang *et al*. [27], this equation can be written in matrix form as:

$$\mathcal{L}_G = \sum_{\mathbf{x}_i \in \mathbf{X}} \left\| \mathbf{x}_i - \left(\bar{\mathbf{x}} + \mathbf{M}_{(1)} \left(\bigotimes_{j=2}^{k+1} \mathbf{w}_j^{(i)} \right) \right) \right\|_2^2, \tag{3}$$

where $\otimes$ represents the Kronecker product of vectors $\mathbf{w}_j$, and $\mathbf{M}_{(1)}$ is the matricized version of $\mathcal{M}$ containing the mode-1 fibers of $\mathcal{M}$ as columns [15]. Writing the transformation $\mathbf{M}_{(1)} \left(\otimes_{j=2}^{k+1} \mathbf{w}_j^{(i)} \right)$ as layers of a neural network allows to learn the multilinear model $\mathcal{M}$ while training the autoencoder. Note also that Equation 3 allows to represent a given label in mode j by different coefficients $\mathbf{w}_j$ for different faces. This can be an advantage when the labeling is not trust-worthy, allowing for flexibility in the factor separation.

Latent loss The previous formulation does not evaluate the coefficients $\mathbf{w}_j$ directly but the reconstruction they yield. Hence, a given mode of variation j might be affected not only by its mode coefficients $\mathbf{w}_j$, but possibly also by the others. In other words, a simple geometric loss can lose the ability to decouple the different modes of variation, which impacts the expressiveness of the model. For instance, expression transfer can then not be performed by simply exchanging the expression coefficients.

To overcome this, we define a loss function that softly constrains latent parameters. Considering the subset $\mathbf{X}_l^{(j)}$ of training faces with provided labels in mode j, and the set $\mathbf{W}_j^{(i)} = \left\{ \mathbf{w}_j^{(i_1)}, \mathbf{w}_j^{(i_2)}, \dots \right\}$ of all mode-j coefficients in the training data that share the same *label in mode* j as $\mathbf{x}_i$, the function writes:

$$\mathcal{L}_L = \sum_{j=2}^{k+1} \sum_{\mathbf{x}_i \in \mathbf{X}_l^{(j)}} \frac{1}{\left| \mathbf{W}_j^{(i)} \right|} \sum_{\mathbf{w}_j^{(p)} \in \mathbf{W}_j^{(i)}} \left\| \mathbf{w}_j^{(i)} - \mathbf{w}_j^{(p)} \right\|_2^2, \tag{4}$$

where the average over coefficients accounts for very different sizes of the sets $\mathbf{W}_j^{(i)}$. We choose this soft constraint to preserve some flexibility over the labels.

4.2. CNN Encoder

The encoder transforms the 3D face input data into a vector $\mathbf{w}$ which contains the concatenated model coefficients, *i.e*. the latent parameters of the face. In order to do this both robustly and efficiently, we leverage recent advances achieved by convolutional neural networks.

The first layer of the network takes as input a 3D scan and converts it into a 2D depth image that encodes heights from a fixed plane, computed by casting rays in the direction normal to the plane. The regression from the 2D heigthmap to the model coefficients is implemented using ResNet-18 [14], a state-of-the-art architecture which has recently shown very good performances in face-related problems [20, 25]. The CNN reduces the image to a 256-dimensional vector, after which three fully-connected layers perform the regression towards the coefficient vector $\mathbf{w}^T$ of the specified dimensions.

4.3. Multilinear Decoder

The multilinear decoder takes as input the vector $\mathbf{w}^T$, which is seen as a concatenation of mode coefficients $\mathbf{w}^T = \left\{\mathbf{w}_2^T, \mathbf{w}_3^T, \dots, \mathbf{w}_{k+1}^T\right\}$, and transforms it into 3D vertex coordinates by performing mode multiplications with the core tensor. As explained in Section 4.1, this operation can be written as the product between the matricized version of the tensor $\mathbf{M}_{(1)}$ and the Kronecker product of each mode coefficient (see Equation 3). Thus, in order to learn the core tensor $\mathcal{M}$ parameters, we implement each of these operations as a layer in the global network, and allow the linear module represented by $\mathbf{M}_{(1)}$ to be learned with the rest of the parameters. This way, we benefit from the capacity of CNNs to robustly summarize the representative aspects of an entire dataset, and from the associated optimization machinery to find the model in a scalable manner.

4.4. Estimation

The multilinear autoencoder estimation proceeds in two stages. First, we initialize both CNN encoder and multilinear decoder since our training data is limited with respect to the number of parameters involved in the multilinear autoencoder. Initializing the multilinear decoder with random values does not yield good results in our experiments. Thus, we initialize it by performing Higher Order Singular Value Decomposition (HOSVD) [17] on a complete subset of the data, *i.e.* a subset in which all the factors of variation are present for all elements. To subsequently pre-train the CNN encoder, we optimize it separately using the generative loss in Equation 2 with the fixed initial multilinear model, and with both registered and unregistered scans to augment training data.

Second, the full network is optimized with all available face data. This is achieved by minimizing the following combined generative and latent loss:

$$\underset{\mathbf{M}_{(1)},\{\mathbf{w}_j^{(i)}\}}{\arg\min} \quad \mathcal{L}_G + \lambda \mathcal{L}_L, \tag{5}$$

where λ weighs the contribution of the latent loss.

5. Evaluation

This section starts by presenting implementation details and an evaluation protocol (Section 5.1). As the final goal of this work is to obtain a generative model of 3D faces, we subsequently present in Section 5.2 evaluations of the multilinear model that can be extracted after training, *i.e.* of the multilinear decoder independently of the rest of the network. We measure both the quality of the model and the decoupling of the latent space, and compare to state-of-the-art methods that learn multilinear 3D face models from incomplete data. Finally, Section 5.3 evaluates the multilinear autoencoder and its ability to register raw scans into the new model.

5.1. Implementation and Evaluation Protocol

Implementation details To pre-train the encoder and to learn the generative model during fine-tuning we use the *AdaDelta* algorithm [29], with parameters as provided in the paper. We use a mini-batch size of 32, a learning rate of 0.1 and no weight decay. The encoder was pre-trained for 17 epochs. For model learning, the full autoencoder is fine-tuned for 100 epochs. The fully connected layers are initialized to random Gaussian weights. Unless specified, we set the dimensions of identity and expression spaces to 65 and 20, respectively, and use $\lambda = 1e^{-1}$. The framework was implemented using Torch7 [7].

Training data for initialization As explained earlier, random initializations are not satisfying and we perform therefore HOSVD on a complete tensor; particularly on the BU-3DFE [28] dataset, which provides 100 identities performing 25 expressions. To pre-train the CNN encoder we use the BU-3DFE and Bosphorus [22] datasets, with the registrations provided by Bolkart and Wuhrer [4], which gives ~ 5000 registered scans. To augment the training data, we sample from the initial multilinear model, randomly rotate each face by an angle $\theta \in [-30°; 30°]$ in yaw, pitch or roll axes, and apply a random scale in $[0.7; 1.1]$. Furthermore, we use both the registered data and the corresponding raw 3D scans, for which the registered versions allow to recover ground truth vertex correspondences for training. This augmentation allows the CNN encoder to learn richer feature extractors, as the raw scans contain larger geometric errors, holes and extra parts such as hair and the neck. This results in a training set of about $500,000$ depth images.

Training data for model optimization We demonstrate the capabilities of the multilinear autoencoder (MAE) trained on two different datasets. A first MAE is learned from static data, using the combined Bosphorus and BU-3DFE databases, with registrations provided by Bolkart and Wuhrer, for a total of 5194 meshes. We will refer to this MAE as *Bu+Bosph*. The second MAE is learned by combining the previous with a subset of the dynamic database D3DFACS [8] using the publicly available registrations of Li *et al.* [18], which allows to build a considerably larger training set. We will refer to this MAE as *D3DFACS*. For this data, we provide sparse expression labels by considering the first 5 frames of each sequence as the neutral expression, and frames located in the middle as peak frames, which are assigned the indicated facial action unit. For testing we leave all sequences of one subject out of the training set, as well as two sequences for each of the other subjects. In total, *D3DFACS* is trained from 49169 scans. Note that the *D3DFACS* training set is an order of magnitude larger than the training sets used in previous methods [4, 27]. Using these training sets shows that MAE can be learned from diverse data and handle inaccurate registrations obtained using fully automatic methods as well as missing labels.

Test data We test both *Bu+Bosph* and *D3DFACS* on parts of the sequences of the D3DFACS database that were left out. In particular, we manually subsample these sequences to keep the most relevant key-frames for testing and to avoid evaluations on very similar scans. In total, there are 270 test frames covering ten subjects and a large expression range.

Evaluation metrics We evaluate the quality of generative models using the metrics *generalization* and *specificity* [11]. *Generalization* measures the ability of the model to adapt to unseen data, and is evaluated by projecting test data into the model space and calculating the reconstruction error. To provide a common framework for comparisons, this is implemented by iteratively fixing one space and finding the optimal coefficients for the other one [26]. *Specificity* measures whether only valid members of the shape class are modeled, or in other words, the model's suitability for generating synthetic data. To evaluate specificity, we assume the data to follow independent normal distributions in identity and expression spaces and sample 1000 faces. To compute the normal distribution using a maximum likelihood estimation while accounting for an imbalanced number of training samples for different labels, we group the coefficients by label and summarize each group by its medoid. For each randomly drawn sample we measure its mean vertex distance to all elements in the training data and keep the minimum value; specificity is defined as the average of this process over all synthetically generated faces.

In Section 5.2, to objectively evaluate the capacity of the model to decouple the spaces, we transfer a recent protocol proposed for body poses [12] to faces. For this, we train a classifier to recognize expressions from a given depth image. We then transfer expressions to each of the identities in the test set and let the classifier measure whether the transferred expression label is preserved. To perform the transfer, we replace the expression weight $\mathbf{w}_3$ of the test face by the medoid of all expression weights with a fixed label over the training data. To train the classifier, we fine-tune the encoder of our architecture, only this time for a classification task. The expression classifier is trained to distinguish 7 prototypical expressions (anger, happiness, disgust, sadness, fear, surprise and neutral) by using the Bosphorus and BU3DFE databases. Note that the goal here is to objectively compare the expression transfer capabilities of different models and not to build an accurate classifier.

5.2. Generative Model Evaluation

This section shows evaluations on the quality of the learned generative models, as well as comparisons to two state-of-the-art methods on multilinear model learning of 3D faces from incomplete data.

Influence of latent loss We first measure how different values of λ affect the quality of the generative models, both for *BU+Bosph* and *D3DFACS*. Results are shown in Table 1. As expected, greater values of λ result in better decoupling of the spaces, as well as more specific models. Higher values also result in higher errors for the training data, which can be explained by the fact that the reconstruction error takes less precedence and that the latent representation is more heavily constrained. For the same reasons, the generalization ability also decreases with higher values of λ. Qualitative examples of the results can be seen in Figure 2. All selected models produce plausible synthetic faces, but there is a clear decrease in the quality of the transfers when the value of λ is too low. For this reason, we select $\lambda = 1e^{-1}$ for the following experiments.

Comparison to initialization We found experimentally that the multilinear decoder needs to be initialized with a previously trained multilinear model. Hence, we evaluate how the autoencoder-based learning process improves this initial model. To this end, we measure generalization, specificity and expression transfer of the initial HOSVD model, and compare it to the model learned with our MAE on the same dataset (BU-3DFE). Table 2 shows the result. We can see that while the generalization error remains approximately the same, the new model becomes more specific and is better able to transfer expressions on unseen data, which implies that our multilinear decoder effectively decouples shape variations due to different factors.

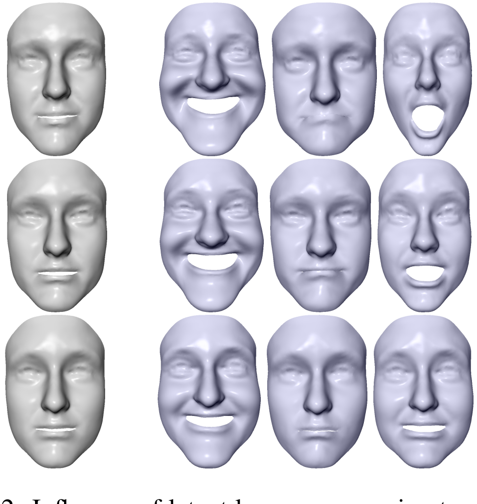


Figure 2: Influence of latent loss on expression transfer on *Bu+Bosph*. From left to right: original registration, transferred expressions: happy, sad, surprise. From top to bottom: $\lambda = 1, \lambda = 1e^{-1}, \lambda = 1e^{-2}$.

λ	BU+Bosph				D3DFACS			
	Training error	Generalization	Specificity	Expression	Training error	Generalization	Specificity	Expression
1	1.60	0.93	**0.92**	36.95	0.48	0.43	**1.39**	19.66
$1e^{-1}$	1.11	0.92	1.53	**48.87**	0.44	0.40	1.68	**20.03**
$1e^{-2}$	**1.02**	**0.90**	1.61	34.37	**0.41**	**0.40**	1.84	19.93

Table 1: Influence of latent loss. Median error in training data (mm), generalization error (mm), specificity error (mm) and percentage of correct classifications after expression transfer for our two training datasets. Best values in bold.

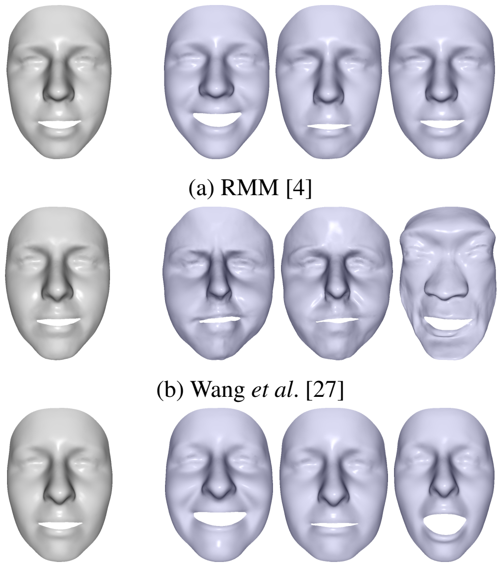
Model	Generalization	Specificity	Expression
Initial model	**0.92**	2.50	15.66
MAE	0.93	**1.43**	**53.08**

Table 2: Comparison between our MAE decoder and the initialization in terms of median generalization error (mm), specificity error (mm), and percentage of correct classifications for expression transfer.

Method	Generalization	Specificity	Expression
RMM [4]	1.34	1.83	32.99
Wang *et al*. [27]	**1.23**	2.38	18.50
MAE	1.35	**1.43**	**46.39**

Table 3: Comparison between state-of-the-art and our MAE decoder on *Bu-Bosph-subset* data, in terms of median generalization error (mm), specificity error (mm), and percentage of correct classifications for expression transfer.

Comparison to state-of-the-art We compare our model to two closely related works, which learn multilinear models of 3D faces from incomplete data: RMM [4] and Wang *et al*. [27]. For RMM we use the publicly available model provided by the authors, which was built using a subset of the databases we consider; in particular, all 205 identites from BU-3DFE and Bosphorus dataset, and 7 expressions from BU-3DFE plus 23 expressions from Bosphorus. For a fair comparison we use the same training data, which will be referred to as *Bu-Bosph-subset*, and select the same dimensions of the representations *i.e*. 23 for identity space and 6 for expression space. We build a model using this setting for the method of Wang *et al*. with code provided by the authors. Table 3 shows the results obtained. We can see that our method outperforms the other two in terms of specificity and expression transfer, while keeping a generalization error close to the others. Figure 3 shows an example of expression transfer results. Note that while RMM and MAE achieve visually plausible results, Wang *et al*. gives noisy implausible faces as their tensor decomposition does not yield a good decoupling of the different modes.


(a) RMM [4]

(b) Wang *et al*. [27]

(c) MAE

Figure 3: Comparison between state-of-the-art and our MAE decoder. From left to right: original scan, transferred expressions: happy, sad, surprise.

5.3. Multilinear Autoencoder Evaluation

Finally we evaluate the full multilinear autoencoder by its ability to reconstruct the original training data, and its ability to register new, unseen data. We start by discussing the computation times of the method.

Computation times Building the initial model using HOSVD requires on average 20 seconds. The pre-training of the encoder takes about $100ms$ per mini-batch on a Nvidia Titan X GPU, which amounts to about 25 minutes per epoch for our training data. Fine-tuning the model takes about $300ms$ per mini-batch, which implies $40s$ per epoch for *Bu+Bosph* and about 8 minutes for *D3DFACS*. Generating each depth image takes around $30ms$ for the registered data. Once the training is finished, regressing from a single raw scan to 3D vertices requires around 0.5s, and this timing can be improved by batch processing.

Improvement of initial registrations The MAE learning process performs a simultaneous optimization over all training data. We observe that this allows to overcome geometric and registration errors that might be present in part of the data. This can be quantitatively assessed by observing the *compactness* of the training data, before and after model learning [11]. The main idea of this quantitative evaluation is that better registrations lead to more compact models as drift in the registration is (erroneous) variation that needs to be encoded. Compactness is computed by measuring the percentage of variability explained by a fixed number of principal components. Figure 4 shows the compactness of the initial registration and the registrations after *Bu+Bosph* training. Note that compactness improved significantly: after training, 99% of the variability can be explained with less than 15 principal components, whereas the initial registrations require 90 principal components to explain this variability. This increase in compactness is achieved while keeping similar model generalization and slightly improving specificity, as shown in Table 2.

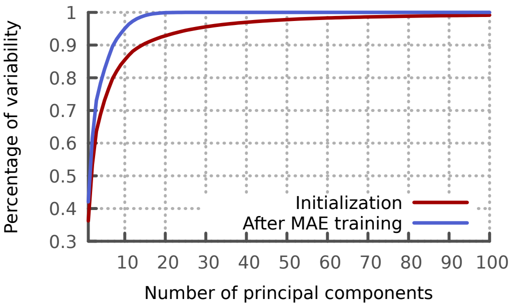


Figure 4: Improvement of initial registrations shown by compactness of the *Bu+Bosph* initialization and registration after our MAE training.

Registration of raw scans To test on real data, we evaluate the reconstructions of the test set obtained by regressing with the multilinear autoencoder when the input is the *original raw scan*. For quantitative evaluation, we consider initial registered versions of the scans as ground-truth even though this might not be exact, since the registrations were manually verified to be globally correct. The MAE reconstruction obtains a median per-vertex Euclidean error of $3.1mm$ for *Bu+Bosph*, and a median per-vertex Euclidean error of $3.6mm$ for *D3DFACS*. Figure 5 shows examples of the registrations obtained. Even though in both cases the errors are relatively high, we observe that the registrations are in general visually close to the expected identity and expression, and could be used as initializations for optimization-based refinements. Figure 6 presents further examples of the registrations obtained with *Bu+Bosph* for scans with different types of occlusion of a subject of the Bosphorus dataset that were not used for training. This shows that the MAE is robust to geometric noise and occlusion.

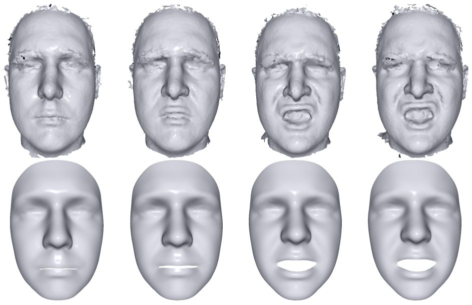


Figure 5: Registration of raw scans using *Bu+Bosph*. Top: input scans. Bottom: registered results.

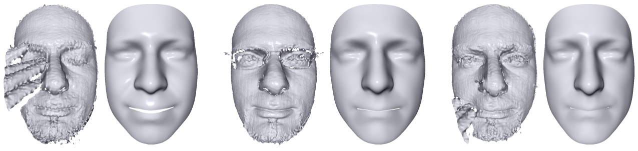


Figure 6: Registration of raw scans presenting different types of occlusion using *Bu+Bosph*. For each pair, left: input scan; right: registered result.

6. Conclusions and Future Work

We presented a multilinear autoencoder architecture for 3D faces that is capable of learning a generative model from incomplete and varied datasets, as well as regressing into this model from raw scans. Experimental evaluation showed that our generative model outperforms current state-of-the-art methods that learn from incomplete data, and that the architecture can be used for fast registration into this model.

The proposed method has limitations, among which the most notable is the need to trade-off between detailed reconstructions and decoupling the latent spaces. For future work we will investigate different loss functions that could remove the need for this trade-off, and improve both the quality of the registrations and the decoupling of the spaces. We believe this work opens possibilities for learning rich generative 3D face models from large training sets; exploring this direction is also a line of future work.

Acknowledgements We would like to thank Timo Bolkart for providing the registrations for D3DFACS data [18], and Mengjiao Wang for providing the code of [27].

References

[1] B. Amberg, R. Knothe, and T. Vetter. Expression invariant 3d face recognition with a morphable model. In *Conference on Automatic Face and Gesture Recognition*, pages 1–6, 2008.

[2] B. Amberg, S. Romdhani, and T. Vetter. Optimal step nonrigid icp algorithms for surface registration. In *Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[3] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *SIGGRAPH*, pages 187–194, 1999.

[4] T. Bolkart and S. Wuhrer. A robust multilinear model learning framework for 3d faces. In *Conference on Computer Vision and Pattern Recognition*, pages 4911–4919, 2016.

[5] J. Booth, A. Roussos, S. Zafeiriou, A. Ponniah, and D. Dunaway. A 3d morphable model learnt from 10,000 faces. In *Conference on Computer Vision and Pattern Recognition*, pages 5543–5552, 2016.

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

[7] R. Collobert, K. Kavukcuoglu, and C. Farabet. Implementing neural networks efficiently. In *Neural Networks: Tricks of the Trade*, pages 537–557. Springer, 2012.

[8] D. Cosker, E. Krumhuber, and A. Hilton. A facs valid 3d dynamic action unit database with applications to 3d dynamic morphable facial modeling. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2296–2303. IEEE, 2011.

[9] H. Dai, N. Pears, W. Smith, and C. Duncan. A 3d morphable model of craniofacial shape and texture variation. In *International Conference on Computer Vision*, pages 3085–3093, 2017.

[10] K. Dale, K. Sunkavalli, M. K. Johnson, D. Vlasic, W. Matusik, and H. Pfister. Video face replacement. *ACM Transactions on Graphics*, 30(6):#130:1–10, 2011.

[11] R. Davies, C. Twining, and C. Taylor. *Statistical models of shape: Optimisation and evaluation*. Springer Science & Business Media, 2008.

[12] P. Ghosh, J. Song, E. Aksan, and O. Hilliges. Learning human motion models for long-term predictions. In *Conference on 3D Vision*, pages 1–9, 2017.

[13] R. A. Güler, G. Trigeorgis, E. Antonakos, P. Snape, S. Zafeiriou, and I. Kokkinos. DenseReg: fully convolutional dense shape regression in-the-wild. In *Conference on Computer Vision and Pattern Recognition*, pages 6799–6808, 2017.

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[15] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[16] S. Laine, T. Karras, T. Aila, A. Herva, S. Saito, R. Yu, H. Li, and J. Lehtinen. Production-level facial performance capture using deep convolutional neural networks. In *Symposium on Computer Animation*, pages #10:1–10, 2017.

[17] L. D. Lathauwer, B. D. Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal of Matrix Analysis and Applications*, 21:1253–1278, 2000.

[18] T. Li, T. Bolkart, M. J. Black, H. Li, and J. Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics*, 36(6):194:1–194:17, 2017.

[19] J. Masci, D. Boscaini, M. Bronstein, and P. Vandergheynst. Geodesic convolutional neural networks on riemannian manifolds. In *International Conference on Computer Vision*, pages 37–45, 2015.

[20] E. Richardson, M. Sela, and R. Kimmel. 3d face reconstruction by learning from synthetic data. In *Conference on 3D Vision*, pages 460–469, 2016.

[21] E. Richardson, M. Sela, R. Or-El, and R. Kimmel. Learning detailed face reconstruction from a single image. *Conference on Computer Vision and Pattern Recognition*, pages 1259–1268, 2017.

[22] A. Savran, N. Alyüz, H. Dibeklioğlu, O. Çeliktutan, B. Gökberk, B. Sankur, and L. Akarun. Bosphorus database for 3d face analysis. *Biometrics and identity management*, pages 47–56, 2008.

[23] M. Sela, E. Richardson, and R. Kimmel. Unrestricted facial geometry reconstruction using image-to-image translation. *International Conference on Computer Vision*, pages 1576–1585, 2017.

[24] A. Tewari, M. Zollhöfer, H. Kin, P. G. G. Bernard, P. Pérez, and C. Theobalt. MoFa: Model-based deep convolutional face autoencoder for unsupervised monocular reconstruction. In *International Conference on Computer Vision*, pages 3715–3724, 2017.

[25] A. T. Tran, T. Hassner, I. Masi, and G. Medioni. Regressing robust and discriminative 3d morphable models with a very deep neural network. *Conference on Computer Vision and Pattern Recognition*, pages 5163–5172, 2017.

[26] D. Vlasic, M. Brand, H. Pfister, and J. Popović. Face transfer with multilinear models. *ACM Transactions on Graphics*, 24(3):426–433, 2005.

[27] M. Wang, Y. Panagakis, P. Snape, and S. Zafeiriou. Learning the multilinear structure of visual data. In *Conference on Computer Vision and Pattern Recognition*, pages 4592–4600, 2017.

[28] L. Yin, X. Wei, Y. Sun, J. Wang, and M. J. Rosato. A 3d facial expression database for facial behavior research. In *Conference on Automatic Face and Gesture Recognition*, pages 211–216, 2006.

[29] M. D. Zeiler. Adadelta: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

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