PCA Retargeting: Encoding Linear Shape Models as Convolutional Mesh Autoencoders

Eimear O' Sullivan
Imperial College London, UK
e.o-sullivan16@imperial.ac.uk

Stefanos Zafeiriou
Imperial College London, UK
s.zafeiriou@imperial.ac.uk

Abstract

3D Morphable Models have long played a key role in the construction of statistical shape models. While earlier models employed Principal Component Analysis, recent work has migrated towards mesh autoencoder models for the construction of lightweight, non-linear shape models that facilitate state-of-the-art reconstruction and the capture of high-fidelity details. Doing so results in a loss of interpretability and regularisation in the model latent space. To address this, we propose *PCA retargeting*, a method for expressing linear PCA models as mesh autoencoders and thereby retaining the gaussianity of the latent space. To encourage the capture of mesh details outside the expressive range of a PCA model, we introduce "free" latent space parameters and evaluate their impact on model performance.

1 Introduction

3D Morphable Models (3DMMs) are one of the foremost technologies for the statistical analysis and modelling of the human face and body. Earlier examples typically employed principal component analysis (PCA) for model construction [2], however recent years have seen the successful application of geometric deep learning leading to impressive gains in model reconstruction accuracy and representational power. Many of these gains comes from the introduction of convolutional mesh autoencoders [35] and increased efficiency of mesh convolution operators [5, 13, 15, 17].

As with standard autoencoder models, convolutional mesh autoencoders are subject to the loss of linear independence and regularity in the latent space that is inherent in PCA models. An example of where this might be problematic would be an attempt to create a blendshape model using a mesh autoencoder network. A blendshape model can be considered as a weighted sum of target expressions that can be combined with a neutral mesh to achieve facial animation [8, 10]. The linear independence of distinct blendshapes is paramount to their success. As autoencoder models do not offer any control over the latent space distribution, this linear independence cannot be preserved and autoencoder models of the human face typically address expressions in isolation [35].

In this paper we propose the concept of *PCA retargeting*, a means of converting a PCA model to a mesh autoencoder format. Our aim is to train an autoencoder model to accurately reconstruct a target mesh while constraining the latent space to mimic the shape parameters the target mesh would produce in a PCA model. Autoencoders are typically trained end-to-end, where the reconstruction loss is calculated based on the output of the decoder and backpropagated through the network. This encourages the model to find an optimal encoding in the bottleneck layer that will permit the decoder to accurately reconstruct an input from its latent representation. To faithfully retarget a PCA model as an autoencoder, however, the latent space of the autoencoder model must follow the PCA shape space as precisely as possible. To enforce this constraint on the latent space, we draw inspiration from variational autoencoders (VAEs). VAEs can be considered autoencoders whose encoding space has been regularised during training to ensure desirable properties for data generation [22, 37]. A Gaussian distribution is often imposed on the latent space, though von Mises-Fisher [12, 44] and

Pre-registration workshop NeurIPS (2020), Vancouver, Canada.

Dirichlet distributions [21] have also proven effective. This desired latent distribution is achieved by adding a regularisation term to the loss backpropagated through the model. By imposing a loss on the model latent space, we aim to compel the encoder to learn the desired latent representation.

The representational power of a PCA model is constrained by the number of linear eigenvectors it contains, and large models are required to achieve high resolution meshes [3]. To allow the retargeted model to represent high-fidelity mesh details, we propose to extend the length of the autoencoder latent vector beyond the number of shape parameters used in the retargeting process. As these additional parameters will not be constrained to follow the PCA shape parameters, the intention is that they will model details that are not expressed within the shape variation of the PCA model. We refer to these additional parameters as "free" latent variables.

Our analysis will address two main questions; 1) Can a PCA model be retargeted as a mesh autoencoder? 2) Can "free" latent parameters be used as an inexpensive means of capturing high-fidelity details? Experiments will be conducted using a large-scale facial dataset, and latent representations of varying sizes will be explored. We hypothesise that imposing rigid constraints on the latent space will allow for the training of an autoencoder model that operates as a neural analogue to a PCA model and that the introduction of "free" latent variables will permit the encoding of high frequency mesh details that can be lost when later PCA model eigencomponents are omitted.

2 Related Work

3D Morphable Models (3DMMs) facilitate the continuous parameterisation of shape variation for a given object class by performing dimensionality reduction on training dataset of meshes [2]. Gaussian Processes [28] and linear blend-skinning [27, 38] have both been used for the construction of 3DMMs, though Principal Component Analysis (PCA) is perhaps the most prevalent approach [2, 3, 11, 28, 34]. Let X be a set of n densely registered meshes, $\{x_1, x_2, ..., x_n\}$, sampled from a given distribution $\mathcal{D}$, where each mesh has d vertices connected in a fixed topology. By making a gaussianity assumption, an arbitrary instance, $x^* \in \mathcal{D}$, can then be represented as a linear combination of the k largest eigenvectors of the covariance matrix of X:

$$x^* \approx \bar{x} + \sum_{i=1}^{k} \alpha_i U_i \tag{1}$$

where $\bar{x}$ is the mean shape, U_i is the i^{th} shape eigenvector, and α_i is the corresponding eigenvalue. Given the linear model representation, the largest shape variations are captured within the first eigenvectors. High resolution models therefore require a larger number of eigenvectors to be retained, while high finer details can be omitted when later principal components are discarded.

Advances in the field of geometric deep learning have led to the generalisation of neural networks to non-Euclidean domains, such as graphs and manifolds [6]. These emerging techniques have been applied across multiple domains, including computer graphics [4, 29], molecular prediction [43], node classification [23], and social network analysis [32, 40]. Many approaches have been applied to extend standard 2D convolutions to non-Euclidean domains, including spectral methods [7, 15, 25, 41], local charting based approaches [4, 26, 29], and convolution operators that act directly on 3D mesh topology [5, 17]. New methods for graph coarsening have also been developed [14, 45], while advances in mesh pooling and unpooling operations have led to the introduction of convolutional mesh autoencoders, an effective means of modelling 3D data [35].

Feature disentanglement permits an intuitive understanding of the latent space of deep generative models. Supervised methods typically require a-priori knowledge of the nature and quantity of generative factors [19, 24, 36]. Higgins *et.al.* presented β-VAE, a reformulation of the standard VAE framework that allowed for a disentangled representation of generative data factors by introducing a β coefficient [18]. Mathieu *et al.* argue that while β-VAE allows for control over the extent of overlap between latent data encodings, it does little to ensure that the aggregate data encodings conform to a desired structure. Instead, they present an approach that permits a desired latent encoding to be achieved by careful choice when defining the prior [30]. GANs [9], Wasserstein Autoencoders [16], different interpretations of VAEs[24, 39], and mutual information maximisation [1, 20, 42] have also been applied successfully for feature disentanglement. In this work, as the desired latent encoding is known in advance, this problem is explored from a supervised angle.

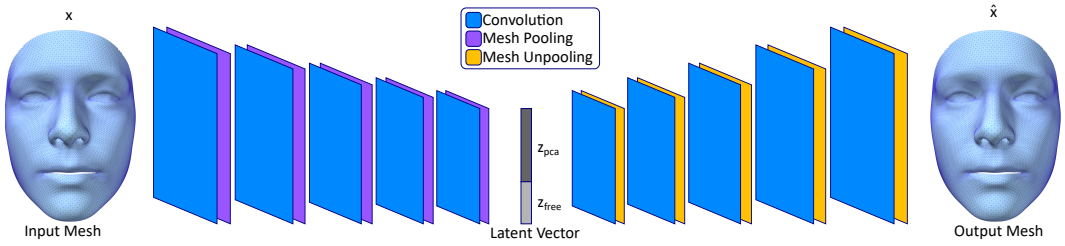

Figure 1: Network Architecture.

3 Model Architecture and Training

Our mesh autoencoder model follows the architecture design of [5]. The encoder is comprised of five mesh convolutional and pooling layers. The decoder architecture is the reverse of the encoder; five convolutional layers, each followed by a mesh unpooling layer. To facilitate the mesh pooling and unpooling operations, we follow the approach of [35]. In each pooling layer the number of mesh vertices will be reduced by a factor of 4. Spiral convolutions with a fixed length of 9 will be used in all convolution layers [17]. The complete network architecture is shown in Figure 1.

The framework will be implemented in PyTorch [33]. Model hyperparameters and loss weights will be determined using a grid-search strategy.

3.1 Losses

For a given input mesh, x, an L1 reconstruction loss will be applied to the decoder output, $\hat{x}$. To encourage the latent vector representation to conform to the desired encoding, z_{pca}, we add a regularisation term to the loss function. In standard VAE models, the regularisation terms typically manifests as the Kullback-Leibler (KL) divergence between the desired and achieved encoding distributions. Here, the desired latent encoding follows a standard normal distribution by design. As such, the KL regularisation term is replaced by an L1 loss on the difference between the desired latent encoding, z_{pca}, and the achieved encoding, $\hat{z}_{pca}$. Defining λ_{rec} and λ_{reg} as the weights applied to the reconstruction and regularisation loss, respectively, the updated loss function takes the form:

$$loss = \lambda_{rec}||x - \hat{x}|| + \lambda_{reg}||z_{pca} - \hat{z}_{pca}|| \tag{2}$$

3.2 "Free" Latent Variables

In addition to the latent vector components reserved to correspond with the principal components of the PCA model, we also propose to extend the latent vector size by including l additional variables. As discussed in the Introduction, PCA models capture the modes of greatest variation in first few principal components while the later components are reserved for the higher frequency details. These later components are often omitted for reasons of model size and computational speed, and this can lead to a "smoothing" of the mesh surface. By adding l "free" variables to the latent vector, we aim to capture these higher frequency details while preserving the advantage of independence among the model components provided by the orthogonal bases of the PCA models.

To conform with the distribution of the other latent variables, we would prefer that the distribution of additional "free" latent variables also take the form of a standard normal distribution with zero mean and unit variance. A KL divergence will be used to encourage the "free" variables to follow the desired Gaussian distribution. Defining the PCA latent variables as z_{pca} and the "free" latent variables as z_{free}, the complete loss function will take the form:

$$loss = \lambda_{rec}||x - \hat{x}|| + \lambda_{reg}||z_{pca} - \hat{z}_{pca}|| + \lambda_{reg}KL[\mathcal{N}(\mu_{z_{free}}, \sigma_{z_{free}}), \mathcal{N}(0, 1)] \tag{3}$$

where $\mu_{z_{free}}$ and $\sigma_{z_{free}}$, the mean and standard deviation of the "free" latent variables, will be calculated per batch during training. The full length of the latent vector is now given as the combined length of z_{pca} and z_{free}. For the "free" latent variables, no assumption of independence between variables will be made, as this criteria is not enforced by the proposed loss function.

4 Experimental Protocol

Tests will be conducted using latent vectors of size 16, 32, 64, and 128. For each latent vector size, three models will be trained and the mean and standard deviation results will be reported. Autoencoder models will be compared to baseline PCA models with the corresponding number of principal components. The number of model parameters will be recorded in each case.

4.1 Dataset

Models will be evaluated using the MeIn3D facial database [3] which consists of more than $10,000$ scans registered to a template with $28,431$ vertices. Data will be split into $9k$ training and $1k$ testing meshes. Random stratified sampling will be applied to maintain gender, age, and ethnic proportions.

To obtain the desired latent vector encoding for all meshes, a PCA model will be constructed from the training dataset. The first 128 principal components will be retained. Shape parameter vectors, α, for each instance will be obtained by projecting the mesh instance onto the model. Following the probabilistic interpretation of PCA models, each parameter, α_i, is an independent random variable. Each α_i follows a Gaussian distribution with zero mean and a variance of λ_i, where λ_i is the i-th PCA eigenvalue. A shape vector with k components can therefore be normalised as follows:

$$z_{pca} = \frac{\alpha_1}{\sqrt{\lambda_1}}, \frac{\alpha_2}{\sqrt{\lambda_2}}, \cdots, \frac{\alpha_k}{\sqrt{\lambda_k}} \tag{4}$$

The normalised vector, z_{pca}, now follows a Gaussian distribution with zero mean and unit variance.

4.2 Autoencoder Accuracy

Encoder-Decoder Accuracy To evaluate the success of the PCA retargeting, we will first independently assess the encoder and decoder networks. To assess the encoder accuracy, all meshes in the test set will be passed through the encoder network and their latent space embeddings, $\hat{z}_{pca}$, retrieved. These will be compared to the ground-truth normalised shape parameters, z_{pca}, for the corresponding meshes. The decoder reconstruction accuracy for a given mesh from the normalised shape parameters will be determined by passing these parameters through the decoder network. The reconstruction error will be calculated as the per-vertex euclidean distance between the ground truth and reconstructed meshes and compared to the reconstruction error for the PCA model.

End-to-end Reconstruction Accuracy To evaluate the model in a holistic manner, each mesh in the training set will be fed through the autoencoder model and the end-to-end reconstruction error will be calculated. This error will be compared to the reconstruction accuracy of the PCA model under the same circumstances; each mesh in the test set will be projected into the shape space of the PCA model and the retrieved shape parameters will be used to obtain the model reconstruction.

4.3 "Free" latent variables

Experiments will be conducted using 16 and 64 principal components to better understand how the number of PCA components impacts the presence of high-fidelity mesh details. "Free" variable lengths of 4, 8, and 16 will be assessed and their conformity to a Gaussian distribution will be evaluated. The reconstruction error for all meshes in the test set will be calculated and compared to that of the standard autoencoder model and the PCA model.

High Frequency Details The presence of high-fidelity mesh details will be evaluated via the Gaussian curvature of the mesh. The Gaussian curvature, $\mathcal{K}$, for a given vertex on a mesh surface is give as a product of the principal curvatures, κ_1 and κ_2, at that point [31]. By calculating the Gaussian curvature at each point for all meshes reconstructed using the autoencoder, the autoencoder with "free" latent variables, and the PCA model, we can ascertain whether the introduction of the the "free" latent variables permits an increase in reconstruction detail and, if so, in which mesh regions.

References

[1] Bachman, P., Hjelm, R.D., Buchwalter, W.: Learning representations by maximizing mutual information across views (2019), https://arxiv.org/abs/1906.00910

[2] Blanz, V., Vetter, T.: A morphable model for the synthesis of 3d faces. In: Proceedings of the 26th annual conference on computer graphics and interactive techniques. pp. 187–194. SIGGRAPH '99 (Jul 1, 1999)

[3] Booth, J., Roussos, A., Ponniah, A., Dunaway, D., Zafeiriou, S.: Large scale 3d morphable models. International Journal of Computer Vision **126**(2), 233–254 (Apr 2018)

[4] Boscaini, D., Masci, J., Rodolà, E., Bronstein, M.M.: Learning shape correspondence with anisotropic convolutional neural networks. In: Neural Information Processing Systems (2016)

[5] Bouritsas, G., Bokhnyak, S., Ploumpis, S., Zafeiriou, S., Bronstein, M.: Neural 3D morphable models: Spiral convolutional networks for 3d shape representation learning and generation. In: International Conference on Computer Vision (ICCV). pp. 7212–7221. IEEE (2019)

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

[7] Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs. ICLR 2014 (2014)

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

[9] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In: International Conference on Neural Information Processing Systems. p. 2180–2188 (2016)

[10] Cheng, S., Kotsia, I., Pantic, M., Zafeiriou, S.: 4dfab: A large scale 4D facial expression database for biometric applications. Conference on Computer Vision and Pattern Recognition pp. 5117–5126 (2018)

[11] Dai, H., Pears, N., Smith, W., Duncan, C.: A 3d morphable model of craniofacial shape and texture variation. In: IEEE International Conference on Computer Vision (ICCV). pp. 3104–3112. IEEE (2017)

[12] Davidson, T.R., Falorsi, L., De Cao, N., Kipf, T., Tomczak, J.M., Globerson, A., Silva, R.: Hyperspherical variational auto-encoders. In: Uncertainty in Artificial Intelligence. pp. 856–865. AUAI Press (2018)

[13] Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. In: Conference on Neural Information Processing Systems (NIPS) (2016)

[14] Diehl, F., Brunner, T., Truong Le, M., Knoll, A.: Towards graph pooling by edge contraction. In: ICML 2019 Workshop on Learning and Reasoning with Graph-Structured Data (2019)

[15] Fey, M., Lenssen, J.E., Weichert, F., Muller, H.: Splinecnn: Fast geometric deep learning with continuous b-spline kernels. Conference on Computer Vision and Pattern Recognition pp. 869–877 (2018)

[16] Gaujac, B., Feige, I., Barber, D.: Learning disentangled representations with wasserstein autoencoders (Oct 7, 2020), https://arxiv.org/abs/2010.03459

[17] Gong, S., Chen, L., Bronstein, M., Zafeiriou, S.: Spiralnet++: A fast and highly efficient mesh convolution operator. In: The IEEE International Conference on Computer Vision (ICCV) Workshops (2019)

[18] Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinivk, M., Mohamed, S., Lerchner, A.: beta-vae: Learning basic visual concepts with a constrained variational framework. In: International Conference on Learning Representations (2017), https://openreview.net/pdf?id=Sy2fzU9gl

[19] Hinton, G.E., Krizhevsky, A., Wang, S.D.: Transforming Auto-Encoders, Lecture Notes in Computer Science, vol. 6791. Springer Berlin Heidelberg, Berlin, Heidelberg (2011)

[20] Hjelm, R.D., Fedorov, A., Lavoie-Marchildon, S., Grewal, K., Bachman, P., Trischler, A., Bengio, Y.: Learning deep representations by mutual information estimation and maximization (2019), https://arxiv.org/abs/1808.06670

[21] Joo, W., Lee, W., Park, S., Moon, I.C.: Dirichlet variational autoencoder. Pattern recognition **107**, 107514 (Nov 2020), http://dx.doi.org/10.1016/j.patcog.2020.107514

[22] Kingma, D.P., Welling, M.: Auto-encoding variational bayes. In: International Conference on Learning Representations (ICLR) (2014)

[23] Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. International Conference on Learning Representations (ICLR) (Sep 9, 2016)

[24] Kulkarni, T.D., Whitney, W.F., Kohli, P., Tenenbaum, J.: Deep convolutional inverse graphics network. In: Advances in Neural Information Processing Systems. vol. 28, pp. 2539–2547 (2015)

[25] Levie, R., Monti, F., Bresson, X., Bronstein, M.M.: CayleyNets: Graph convolutional neural networks with complex rational spectral filters. IEEE Transactions on Signal Processing **67**(1), 97–109 (Jan 1, 2019)

[26] Lim, I., Dielen, A., Campen, M., Kobbelt, L.: A simple approach to intrinsic correspondence learning on unstructured 3d meshes. In: Proceedings of the European Conferenceon Computer Vision Workshops (ECCVW) (2018)

[27] Loper, M., Mahmood, N., Romero, J., Pons-Moll, G., Black, M.: SMPL: A skinned multi-person linear model. ACM Transactions on Graphics (TOG) **34**(6), 1–16 (2015)

[28] Lüthi, M., Jud, C., Gerig, T., Vetter, T.: Gaussian process morphable models. IEEE Transactions on Pattern Analysis and Machine Intelligence **40**(8), 1860–1873 (2018)

[29] Masci, J., Boscaini, D., Bronstein, M.M., Vandergheynst, P.: Geodesic convolutional neural networks on riemannian manifolds. In: Proceedings of the IEEE International Conference on Computer Vision Workshops (ICCVW. pp. 832–840. IEEE (2015)

[30] Mathieu, E., Rainforth, T., Siddharth, N., Teh, Y.W.: Disentangling disentanglement in variational autoencoders. Proceedings of Machine Learning Research **97** (Jun 9, 2019)

[31] Meyer, M., Desbrun, M., Schröder, P., Barr, A.H.: Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. Springer Berlin Heidelberg (2003), https://search.datacite.org/works/10.1007/978-3-662-05105-4_2

[32] Monti, F., Frasca, F., Eynard, D., Mannion, D., Bronstein, M.M.: Fake news detection on social media using geometric deep learning (Feb 10, 2019), https://arxiv.org/abs/1902.06673

[33] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. In: NIPS Autodiff Workshop (2017)

[34] Ploumpis, S., Ververas, E., O' Sullivan, E., Moschoglou, S., Wang, H., Pears, N., Smith, W., Gecer, B., Zafeiriou, S.P.: Towards a complete 3D morphable model of the human head. IEEE transactions on pattern analysis and machine intelligence **PP**, 1 (Apr 29, 2020)

[35] Ranjan, A., Bolkart, T., Sanyal, S., Black, M.J.: Generating 3d faces using convolutional mesh autoencoders. In: European Conference on Computer Vision (2018), http://arxiv.org/abs/1807.10267

[36] Reed, S., Sohn, K., Zhang, Y., Lee, H.: Learning to disentangle factors of variation with manifold interaction. In: Proceedings of the 31st International Conference on International Conference on Machine Learning. p. II–1431–II–1439 (2014)

[37] Rezende, D.J., Mohamed, S., Wierstra, D.: Stochastic backpropagation and approximate inference in deep generative models. In: International Conference on Machine Learning (ICML) (2014)

[38] Romero, J., Tzionas, D., Black, M.: Embodied hands: Modeling and capturing hands and bodies together. ACM Transactions on Graphics (TOG) **36**(6), 1–17 (2017)

[39] Shao, H., Lin, H., Yang, Q., Yao, S., Zhao, H., Abdelzaher, T.: DynamicVAE: Decoupling reconstruction error and disentangled representation learning (Sep 14, 2020), https://arxiv.org/abs/2009.06795

[40] Tan, Q., Liu, N., Hu, X.: Deep representation learning for social network analysis. Frontiers in Big Data **2** (Apr 3, 2019)

[41] Tang, S., Li, B., Yu, H.: Chebnet: Efficient and stable constructions of deep neural networks with rectified power units using chebyshev approximations (Nov 7, 2019), https://arxiv.org/abs/1911.05467

[42] Tschannen, M., Djolonga, J., Rubenstein, P.K., Gelly, S., Lucic, M.: On mutual information maximization for representation learning (2020), https://arxiv.org/abs/1907.13625

[43] Veselkov, K., Gonzalez, G., Aljifri, S., Galea, D., Mirnezami, R., Youssef, J., Bronstein, M., Laponogov, I.: Hyperfoods: Machine intelligent mapping of cancer-beating molecules in foods. Scientific reports **9**(1), 9237–12 (2019)

[44] Xu, J., Durrett, G.: Spherical latent spaces for stable variational autoencoders. In: Conference on Empirical Methods in Natural Language Processing (EMNLP) (2018), https://arxiv.org/abs/1808.10805

[45] Ying, R., You, J., Morris, C., Ren, X., Hamilton, W.L., Leskovec, J.: Hierarchical graph representation learning with differentiable pooling. In: International Conference on Neural Information Processing. pp. 4805–4815 (2018)