EUROGRAPHICS 2022 / R. Chaine and M. H. Kim
(Guest Editors)

Volume 41 (2022), Number 2

Shape Transformers: Topology-Independent 3D Shape Models Using Transformers

Prashanth Chandran[1,2] Gaspard Zoss[2] Markus Gross [1,2] Paulo Gotardo[2] Derek Bradley[2]

[1]ETH Zurich [2]DisneyResearch|Studios

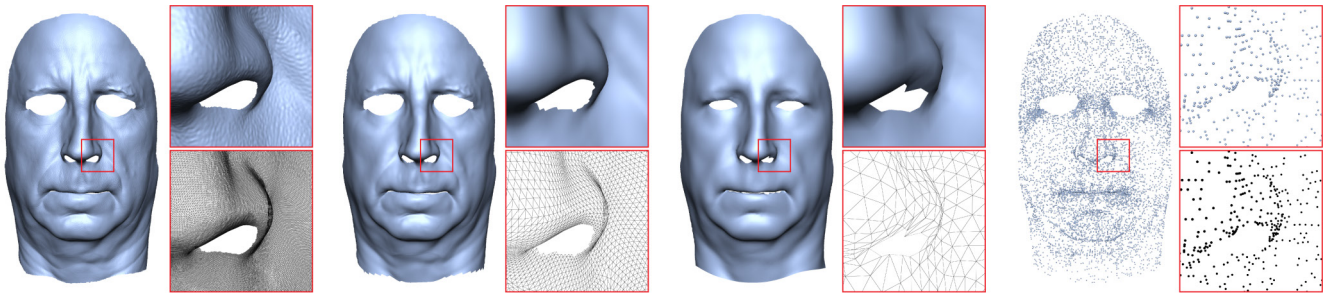

Figure 1: *Our transformer-based 3D shape autoencoder leverages the transformer's self-attention mechanism to effectively capture nonlinear spatial correlation of arbitrary extent, without the need to dictate the size of the receptive field* a priori. *It can be trained on a mixture of 3D datasets of different topologies and spatial resolutions. At test time, the same model can be evaluated to output different topologies and arbitrary spatial resolutions for different application scenarios. Here we show the output of one model evaluated with 4 different topologies, highlighting differences in the same region for each variation.*

Abstract

Parametric 3D shape models are heavily utilized in computer graphics and vision applications to provide priors on the observed variability of an object's geometry (e.g., *for faces). Original models were linear and operated on the entire shape at once. They were later enhanced to provide localized control on different shape parts separately. In deep shape models, nonlinearity was introduced via a sequence of fully-connected layers and activation functions, and locality was introduced in recent models that use mesh convolution networks. As common limitations, these models often dictate, in one way or another, the allowed extent of spatial correlations and also require that a fixed mesh topology be specified ahead of time. To overcome these limitations, we present* Shape Transformers, *a new nonlinear parametric 3D shape model based on transformer architectures. A key benefit of this new model comes from using the transformer's self-attention mechanism to automatically learn nonlinear spatial correlations for a class of 3D shapes. This is in contrast to global models that correlate everything and local models that dictate the correlation extent. Our transformer 3D shape autoencoder is a better alternative to mesh convolution models, which require specially-crafted convolution, and down/up-sampling operators that can be difficult to design. Our model is also topologically independent: it can be trained once and then evaluated on any mesh topology, unlike most previous methods. We demonstrate the application of our model to different datasets, including 3D faces, 3D hand shapes and full human bodies. Our experiments demonstrate the strong potential of our Shape Transformer model in several applications in computer graphics and vision.*

CCS Concepts

• ***Computing methodologies*** → ***Shape modeling; Modeling methodologies;***

1. Introduction

Parametric 3D shape models are ubiquitous in computer graphics and computer vision since they provide a prior on the space of observable geometric variation and deformation, helping constrain many problems such as 3D animation and performance capture. For instance, face and full body models are usually built from datasets of 3D scans in spatial correspondence, and the aim is to represent the domain of statistically-plausible 3D shapes in a compact and controllable way. The most common model is the linear 3D morphable model (3DMM, [BV99]), which expresses new shapes as linear combinations of prototypical basis shapes. Over the past two decades, researchers have investigated face and body models and

improved upon 3DMMs in several ways, including the semantic separation of identity versus expression/pose parameters, and the creation of localized linear models in an attempt to decouple unwanted spatial correlations at distant points across the 3D shape, to achieve better generalization. Notably, localized models can be much more expressive than global ones, and can be built from smaller datasets. But their expressive power comes at the cost of decreased robustness; i.e., it can be easier to generate implausible shapes with local models than global ones.

A major drawback of 3DMMs is their linear nature, which leads to less compact models that cannot represent continuous, nonlinear deformation of familiar shapes such as human faces and bodies. For this reason, the explosion of deep learning has already been leveraged to build more powerful, nonlinear models. Initial deep-learning models of 3D geometry were particularly limited to fully-connected network architectures [CBGB20] that have a much larger number of free parameters and, thus, require more training data. Fully-connected networks represent global shape models, and are thus susceptible to the same limitations of global 3DMMs in terms of generalization and modeling spurious global correlations across the shape. In contrast, convolutional neural networks (CNNs), with fewer free parameters and more localized receptive fields, have also been leveraged for building new models. The building blocks for these models were initially limited to familiar convolutions on the simple 2D image grid. More recently, new definitions of convolutions on 3D point neighborhoods were proposed to allow the creation of geometric models in the form of CNNs that can operate on the surface of a 3D mesh [HHF*19, GCBZ19]. While mesh-convolutional architectures yield compact 3D models that represent local surface deformations, such convolutional models capture global correlations very poorly, requiring careful handcrafting of down/up-sampling operators on meshes.

Whether linear or nonlinear, the choice between a fully global versus a strictly local model is quite limiting. In both cases, the extent of spatial correlations between different points on the shape is dictated *a priori*. In the case of global models, all points on the surface are naïvely correlated with each other. Although correlations are ultimately learned from data, the limited sampling of the true underlying shape distribution inevitably leads to learning spurious global correlations (*e.g.*, a model that only sees both eyes blinking together during training cannot express a face with a single eye closed). Local models alleviate this problem to some extent by introducing explicit independence assumptions that remove distant correlations, but the specific local region structure is dictated ahead of time, instead of being learned from data. For the complex geometric domain of human faces and bodies, it can be very difficult (or impossible) to manually specify which 3D points should be correlated with each other in the creation of realistic 3D shapes.

When building shape models, in addition to the choices between linear versus nonlinear and global versus local models, another important aspect is the mesh topology used to create the model. Notably, previous parametric shape models are designed for fixed mesh topologies. In other words, the topology used to create the model must be the topology used to evaluate the model. This issue is often overlooked, but in fact introduces an important limitation when attempting to use the same pre-trained model in different application scenarios (*e.g.*, high-quality animation, video games, or other visualization in realtime). Often the parametric model does not adapt naturally to the application and constraints at hand, without having to be re-trained at different resolutions.

This paper directly addresses these main issues concerning parametric 3D shape models, namely: (*i*) can the nonlinear model automatically learn the important spatial correlations without being dictated *a priori*; and (*ii*) can the model be topology independent, allowing pre-trained models to be applied at different resolutions depending on the use case. To address these issues, we present a new parametric 3D shape autoencoder based on transformer architectures, which we call *Shape Transformers*. A transformer [VSP*17] is a neural network designed to carry out inference over sequences of input tokens (data) of arbitrary length, as well as translation into another type of token sequence. Transformers have been used very successfully in natural language processing and are increasingly being utilized in many computer vision applications. Our new shape model is a neural network that exploits the transformer's self-attention mechanism to automatically and dynamically capture spatial correlations across the entire shape. In contrast to other models, our transformer autoencoder is both a global and a local neural shape model and can represent geometric shape detail at an arbitrary spatial resolution, without being tied to a specific mesh topology. As a result, our method does not require careful, hand-crafted pre-computation of down/up-sampling operators like mesh-convolutional models, and can be both trained and applied with meshes of multiple topologies and resolutions. In addition to these new benefits, our transformer autoencoder provides all the usual advantages of recent shape models, including nonlinear deformation characteristics, and the ability to disentangle semantic attributes like identity and expression, to sample from the model parameters to generate new shapes, and to optimize over the set of parameters to generate specific shapes in reconstruction scenarios. We demonstrate our new shape model on various datasets including 3D human faces, hands and full bodies. We also show the strong potential for our transformer-based shape models in different applications in computer graphics and vision.

2. Related Work

The seminal linear 3DMM was originally proposed by Blanz and Vetter [BV99] to model faces. Over the years, linear models were improved by semantically separating the parameters that correspond to identity from those that encode expression, giving birth to so-called Multi-Linear Models (MLMs) [VBPP05]. Localized linear models were shown to improve expressibility by splitting the shape into a subset of distinct regions [TDlTM11, NVW*13], and incorporating anatomical constraints [WBGB16]. Wang et al. [WBZB20] showed the best of both worlds by developing a global-local multilinear framework for facial modeling. There is a multitude of applications that make use of such linear shape models. We refer to the recent survey of Egger et al. [EST*20] for a detailed account of 3DMMs and the challenges in building and applying them. The obvious drawbacks of all linear face models are their lesser degree of compactness and inability to model the continuous, nonlinear deformation of human faces. Some linear models also dictate the allowed extent of spatial correlations *a priori*, and are created on fixed topologies. Linear models of shape defor-

mation also include techniques like Linear Blend Skinning (LBS) [KCZO08] which is widely used tool in the industry for character animation. Kavan et al.[Kv05] also introduced a dual quaternion extension to LBS to allow for nonlinear joint articulations. Such techniques were also utilized in the building of full body morphable models like the popular SMPL model [LMR*15].

To address the linearity issue, most recent works compute deep 3D shape models in the form of neural networks. The feed-forward, fully-connected neural network provides a simple architecture to train neural 3D face models [ABWB19, CBGB20]. As in previous approaches, these models operate by flattening the list of the vertices into a vector and feeding it through the network, which includes a sequence of linear projections and nonlinear activation layers. They have strong representative power due to a large number of parameters, but they are also strongly susceptible to overfitting and poor generalization. These are also global shape models and are tied to a specific 3D mesh topology.

Graph Convolution Networks (GCNs) generalize the convolutional operator to work with arbitrary graphs [DBV16, RBSB18, HHF*19, BBP*19, GCBZ19, ZWL*20, CK21]. They make use of the graph connectivity to define convolutions over the appropriate local neighborhood of an input vertex. Generally speaking, GCNs are not tied to a specific topology as each iteration of training or evaluation can take as input a variable set of vertices of the graph and the desired adjacency information for performing convolutions. While the localized nature of graph convolutional networks make them efficient and small in size, they introduce difficulties in modeling long-range relationships across vertices of the graph. In the context of geometry processing, to overcome this problem of a limited receptive field, previous work has computed matrix operators for mesh down/up-sampling that respectively decrease and increase the resolution of the topological graph of a mesh. These matrices are often carefully pre-computed before training [RBSB18, BBP*19, GCBZ19, ZWL*20] and are used to increase/decrease the neighborhood over which a graph convolutional filter operates. We refer to this specific form of graph convolution network as a mesh convolution. Mesh-convolutional networks restrict the model to a single topology and have no guarantee to remain optimal for deformed shapes with the same topology (*e.g.*, different facial expressions). To alleviate the latter issue, previous work has also attempted to learn these down/up sampling operations [CK21] but the fixed topology issue remains.

There exists many variants of these graph convolutional operators in practice. Ranjan et al. [RBSB18] used Chebyshev convolutions [DBV16] and pre-computed down/up-sampling matrices to train a face model. Hanocka et al. [HHF*19] proposed MeshCNN, a convolutional network that operates on edge features instead. Using the known topology of a mesh, they start by defining edge properties (such as orientation, length etc.) and use convolutions to learn higher dimensional features over pairs of vertices. They also define new ways of down/up-sampling edge features to process a given shape at multiple resolutions. They show applications in learning to segment and sub-sample meshes using their network. Bouritsas et al. [BBP*19] introduced SpiralNet as an alternative to standard GCNs that use the adjacency information during convolution. They fix the topology over which the network operates and proposed pre-computing spirals of vertices of a fixed length and passing them through an MLP to extract features. SpiralNet++ [GCBZ19] extended the speed and efficiency of SpiralNet by introducing dilation, a means of increasing the receptive field of the MLP without affecting the capacity of the network, and showed reduction in training times and improved performance on reconstruction tasks. Zhou et al. [ZWL*20] also assume a fixed topology and learn a basis of convolutional filters that operate on a vertex neighbourhood. To increase representative power, their model jointly learns a per-vertex weight map that informs the network of which basis of filters the vertex would benefit from being convolved with the most. They also introduce down/up-sampling techniques that offer the user more control over the process.

Our work makes use of transformer architectures [VSP*17] to construct topology-independent 3D shape models that more effectively learn spatial correlations. Originally proposed for language models, transfomers were later naturally adopted for inference over temporal sequences and also in vision tasks [DBK*21]. Notably, recent research has investigated the use of transformers for 3D reconstruction from images. Lin et al. [LWL21a] use a vanilla transformer to reconstruct coarse posed human bodies and coarse posed hands from images and a learnable MLP to upsample the meshes to full resolution. In a follow up work [LWL21b], the authors coupled their previous vanilla transformer with graph-convolutional layers and showed better accuracy for the body and hand reconstruction tasks. In contrast to their work, our new model stands out as the first implicit parametric model designed exclusively as a transformer network and capable of outputting high resolution meshes. Our models are also topology-independent and can be trained on a mixture of arbitrary mesh topologies, and then applied on other topologies and spatial resolutions while Lin et al. [LWL21a, LWL21b] use an upsampling MLP, fixing the output mesh resolution.

Another type of topology-agnostic shape models are built using neural, continuous implicit representations, where an MLP operates on individual 3D points as input [PFS*19, YTB*21, CZ19, DLJ*20, GYH*20]. These models represent the 3D shape as a level-0 isosurface that must be searched within the represented volume using a root finding algorithm. In contrast, our method employs an explicit canonical shape and outputs a continuous field of deformation offsets, making its evaluation trivial and significantly more efficient when sampling shapes at very high resolution (*e.g.* Fig. 15). Also, the architecture we propose is transformer-based and operates simultaneously on an arbitrary number of points, using its self-attention mechanism to exploit both local and global context. Our new architecture enables a number of new applications, which we demonstrate in Section 4.

3. Modeling 3D Shapes Using Transformers

This section introduces our novel, topology-independent 3D shape model, which we call a Shape Transformer. Our model leverages a transformer's self-attention mechanism to more effectively capture nonlinear spatial correlation of arbitrary extent, while overcoming some of the limitations of recent neural architectures used in shape modeling, such as GCNs. Instead of carefully hand-crafting the network's receptive field, as in a GCN, our key idea is to exploit self-attention to automatically learn, in a data-driven manner, the actual

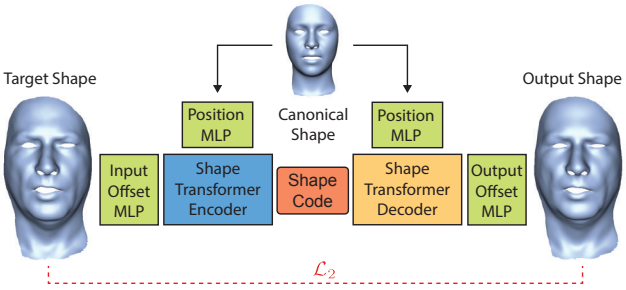

Figure 2: *Overview of our transformer-based autoencoder for topology-independent, nonlinear 3D shape modeling. At the center of the model lies a canonical 3D shape that is deformed based on a shape code associated with the target 3D shape. Shape codes can be generated randomly, interpolated, or computed from an input 3D shape via the encoder. This autoencoder is trained end-to-end, in a fully-supervised fashion using datasets of registered 3D shapes, possibly varying in topology and spatial resolution.*

extent of the receptive field at each point in a 3D shape. As a result, our model is able to capture both short- and long-distance spatial correlation on the target 3D geometry, without explicit definitions of neighborhood, connectivity, or convolution. To avoid GPU memory bottlenecks, our model computes a smaller cross-covariance self-attention matrix, following El-Nouby et al. [ENTC*21] (see Section 3.5). While we use faces as example when introducing our model, the proposed architecture generalizes to other shapes including human hands and full bodies, as shown in Section 4.

3.1. Overview

A schematic of Shape Transformer, our transformer-based 3D shape autoencoder architecture is shown in Fig. 2. We begin with an overview of the decoder, Fig. 2 (right), to illustrate how the model generates the output 3D shape. At the center of our architecture lies a canonical 3D shape (*e.g.*, a template body or face with standard identity, in a reference pose or expression) whose surface provides the continuous domain of canonical 3D locations used to sample the model. These 3D locations are also associated with a latent shape code: a parameter vector that represents a particular geometric deformation that is applied to the canonical 3D geometry to yield the output 3D shape (*e.g.*, a face with a different identity and expression). Together, the shape code and queried 3D locations form the input to our transformer-based decoder, while the output comprises 3D offset vectors that correspond to each queried 3D location. The generated offsets are added to the canonical shape to produce the output 3D shape, Fig. 2 (right). Similarly, the inputs to the shape encoder, Fig. 2 (left), are also represented as 3D offsets from the corresponding canonical points. In contrast to these offsets, points on the canonical shape itself are always queried (sampled) as absolute 3D coordinates in world space. Our transformer-based 3D shape autoencoder can be trained end-to-end, in a fully supervised way using datasets of registered 3D shapes. Note a first interesting result of our transformer-based architecture: both at training and test times, the input to the encoder and decoder can comprise an arbitrary number of queried 3D locations that are near and/or far away from each other, and in arbitrary ordering. Each of these input 3D locations is processed separately, to a large extent, but is also embedded (transformed) with information coming from the other inputs, as captured by the transformer's dynamic self-attention mechanism. This property enables our shape model to be trained on a mixture of 3D datasets of different topologies and spatial resolutions (*e.g.*, high-quality dense 3D scans, marker-based motion capture data). At test time, the same model can be evaluated on other topologies with arbitrarily-varying spatial resolutions, depending on the user's need and application.

The goal of the decoder, during training, is thus to learn to estimate non-linear spatial correlations across the arbitrary sets of canonical 3D locations and use this contextual information to predict the corresponding 3D offsets that will form the output shape, as guided by the input shape code. In contrast, the encoder is given an arbitrarily-sized sequence of input 3D point offsets (relative to the canonical shape), from which the goal is to estimate an associated shape code. The corresponding 3D points on the canonical shape are also provided as input to the encoder. The architectures of both the encoder and decoder comprise a sequence of 4 transformer blocks, as detailed in Section 3.2 and Section 3.3. While our network could operate without the encoder, by following the traditional alternative approach of computing the latent (shape) codes via iterative optimization, having an encoder is often beneficial: it provides faster estimation of shape codes at test time, while also learning a better structure for the latent space of shape codes during training. Additionally, there are also particular advantages from having a *transformer-based* encoder: (*i*) the encoder naturally operates on arbitrarily-sized subsets of input 3D points (and offsets), thus facilitating the handling of cases of missing data as when encoding shapes that are partially occluded; and (*ii*) the transformer can also be applied to separately encode smaller regions of the input 3D shape, yielding distinct per-region codes that can be mixed and used together to extrapolate novel, unseen 3D shapes, thus increasing the expressiveness of the model. These advantages, along with other applications of our method are demonstrated in our experimental results in Section 4.

3.2. Transformer-based 3D Shape Encoder

A typical transformer operates on a sequence of input tokens, transforming them into a desired output sequence by modeling token-to-token interaction via *self-attention* [VSP*17], while also individually modifying the tokens with residual MLPs. In our transformer-based encoder, the input tokens are given as (*i*) an arbitrary set of 3D locations on the canonical shape, and (*ii*) a set of corresponding 3D offsets that lead to a target shape. As described next, the canonical points are used to position-encode the target offsets, hence the need for them to be in correspondence. However, the number of sampled points and the sampling strategy remain arbitrary.

As shown in Fig. 3, each canonical 3D point is first individually fed through a 4-layer *Position MLP* that maps the point onto a high-dimensional latent position. In parallel, each target 3D offset is also fed through a similar *Input Offset MLP* that maps the offset onto another high-dimensional latent code. Applying these two initial MLPs before passing the tokens over to the transformer, allows the model to represent the 3D shape more generically, without being limited to the spatial distribution in the 3D world space. This operation enables the network to stretch and squeeze the distribution

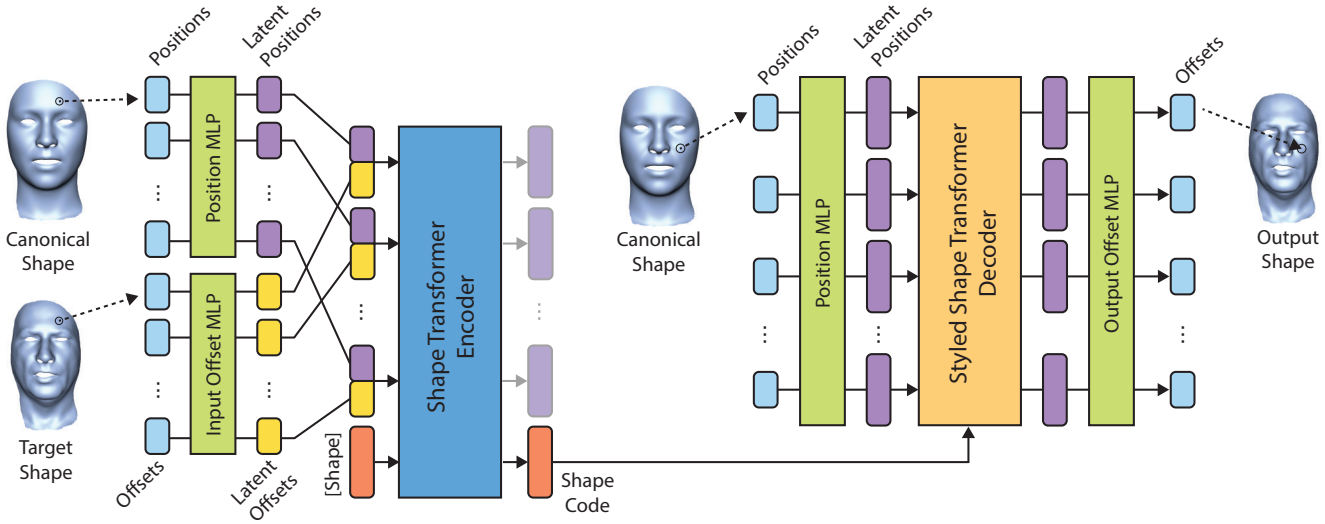

Figure 3: *The detailed architecture of the proposed transformer-based 3D shape autoencoder: (left) transformer-based 3D shape encoder, including a special, fixed input token for shape queries that is trained with the network; and (right) transformer-based 3D shape decoder.*

of input tokens such that they are optimally distributed for the task at hand. The input to the transformer encoder is then obtained by concatenating the corresponding canonical and offset tokens into a single sequence of tokens. Position encoding via this concatenation step effectively tags each offset token with features related to a specific position in the canonical shape, thus helping the transformer in learning spatial correlations. This type of position encoding has been found to be beneficial in previous work [LWL21a] compared to regular sinusoidal position encoding. Our choice of position encoding method is continuous and naturally allows for arbitrary resolution in the input space. An ablation study showing its effectiveness is presented in the supplemental material.

Our transformer encoder is then designed as a sequence of four standard XCiT blocks [ENTC*21]. The goal for this transformer is to produce a single shape code from the collection of input tokens. This is however in contrast with a key property of transformers, which are sequence-to-sequence architectures that provide as many outputs as the number of input tokens. To address this issue, an elegant solution to implementing our goal using a transformer is to introduce an extra input token, similar to the `[CLS]` token in BERT [DCLT19], that is learned (optimized for) with the network parameters during training. This single, new input *shape query token* is itself a part of the encoder that is used to encode all input shapes and remains constant after training. Unlike the other tokens, this special token is not position encoded. It is processed by the transformer encoder and is embedded with information from the other input tokens, leading to the desired output shape code. The other tokens output by the encoder are not used (grayed out tokens in Fig. 3). Thus, this mechanism allows us to retrieve a shape code from the encoder irrespective of the number of input tokens.

3.3. Transformer-based 3D Shape Decoder

For our transformer-based decoder, the input tokens are given as (i) an arbitrary set of 3D locations on the canonical shape (as for the encoder), and (ii) a shape code vector. Note that the queried canonical points do not have to match those seen by the encoder. As when encoding, the sampled canonical 3D points are separately fed through a *Position MLP* to provide high-dimensional tokens for the decoder. The decoder also consists of a sequence of 4 transformer blocks with standard residual MLPs, but *with modulated input tokens*: instead of concatenating each token with the shape code, the shape code is used to modulate the intermediate activations that serve as input to each decoder block. This novel *style-modulated* transformer block is described in detail in Section 3.4. The decoder outputs a different sequence of tokens (as in a standard trasformer), one for each queried canonical point. Finally, each of these output token is independently processed by the *Output Offset MLP* to produce final 3D offsets that are added to the canonical points, resulting in the decoded 3D points of the ouput shape.

3.4. Style-Modulated Transformer Block in Shape Decoder

The basic idea motivating the design of our new style-modulated transformer block is to inject information derived from the shape code directly into each one of the transformer blocks. As a result, the network does not have to waste capacity in memorizing this information while carrying it forward to subsequent blocks. The architecture of our transformer block is illustrated in Fig. 4. We build on top of the recently proposed XCiT block [ENTC*21] due to its memory efficiency when operating on very long sequences of tokens (see discussion in Section 3.5). In each transformer block, a 5-layer *Style MLP* maps the input shape code into a style code of the same size as each input token (these per-block MLPs have *ReLU* activations and shared weights for the first four layers). Each input is then individually modulated by the style token via multiplication of corresponding feature positions. The intended effect is to emphasize or suppress a subset of features of the input token, as guided by the style code at the particular layer of the transformer block. As shown in Fig. 4, the set of modulated tokens is then fed through a standard XCiT transformer block. This block is then responsible for processing each token and exchanging information

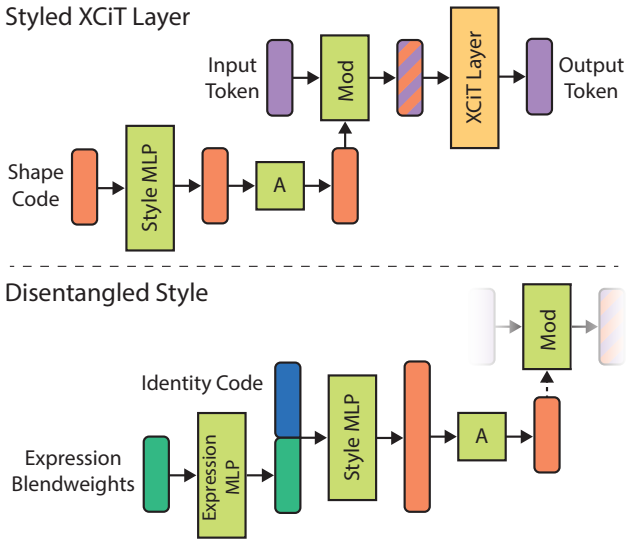

Figure 4: *Our new style-modulated XCiT layer (top) allows the shape code to suppress or emphasize different features of the per-point tokens at different layers of our transformer-based decoder. In our identity-expression disentangled version (bottom), we explicitly split the Identity Code from the expression: the blendweights first go through an Expression MLP and its output is concatenated with the Identity Code before it follows a path similar to the Shape Code, modulating the XCiT Layer.*

across all tokens via self-attention. As modulation happens just before (outside) each transformer block, it is also possible to use different shape codes to modulate different input tokens, as a way of introducing localized shape deformation. This more advanced feature of our model is demonstrated empirically in Section 4.8 and in our supplemental material.

3.5. Side-Stepping GPU Memory Bottlenecks

A transformer dynamically builds one or more attention matrices that capture correlations and propagate information across a potentially very large number of input tokens. Each attention matrix is a dense matrix whose number of entries depend on the squared length of the input/output sequence. In our model, the number of elements in the attention matrix is the squared number of simultaneously queried canonical 3D points. For high-quality 3D shapes such as the ones used to train our network, the total number of observed 3D points can range from a few thousands to hundreds of thousands. Building the standard self-attention matrix for sequences of such length is most often infeasible due to GPU memory constraints. To circumvent this well-known problem with transformers, we rely on the recently proposed XCiT cross-covariance attention [ENTC*21] as a more compact replacement for the standard self-attention. Since our goals are different than that in El-Nouby et al. [ENTC*21], we replace their 3×3 (LPI) convolutional layer with a 1×1 convolution and keep our model invariant to permutation on the input tokens. In addition, our continuous shape model naturally makes it possible to apply different strategies for sampling separate subsets of 3D points, sequentially, in order to progressively evaluate complete and dense 3D shapes with a virtually unlimited number of points.

Training. We train our model using a dataset of registered 3D shapes, with known spatial correspondence (but not necessarily of the same topology), from which the canonical shape is defined as the mean shape or any particular, most representative shape (*e.g.*, the most dense 3D shape, when training using different topologies). We first compute 3D offsets between points in all training shapes, relative to those in the canonical shape. Then, the weights of our 3D shape autoencoder and the query shape code shown in Fig. 3 are jointly optimized during training in a normal supervised fashion; the objective is to minimize the L2-loss between the ground-truth per-point 3D offsets, which are provided as inputs to the encoder, and the corresponding 3D offsets that are output by the decoder (Fig. 2). Once trained, our model allows for a variety of topology-independent applications in 3D shape modeling, reconstruction, and shape deformation, as shown in the following.

4. Results

We now demonstrate how our topology-independent transformer shape model performs on datasets with different classes of 3D shapes, including faces, hands, and full human bodies (Section 4.1), while also analyzing and evaluating our design choices. We begin by showcasing the representative power of our model, specifically in the scenarios of reconstructing and interpolating 3D shapes (Section 4.2). We then highlight one of the main benefits of our approach, which is the topology independence of the underlying shape model (Section 4.3), and illustrate several new applications enabled by this property including shape completion (Section 4.4), accurate shape fitting and generalization (Section 4.5), and the creation of high-quality faces (Section 4.6). In Section 4.7, we present several experiments that demonstrate how the model reacts to scenarios never seen during training. We additionally present an extension of the original architecture that allows for disentangling facial identity and expression (Section 4.8), for better application in parametric face modeling. Kindly refer to our supplementary material for additional experiments.

4.1. Datasets

Our 3D shape autoencoder is generic and can be trained on a wide variety of 3D shapes. In this section, we briefly introduce the datasets we use to validate our approach. For 3D faces, we consider the datasets of COMA [RBSB18] and Semantic Deep Face Models (SDFM) [CBGB20]. Besides faces, we show application on the MANO hand dataset [RTB17], and the DFAUST body dataset [BRPMB17]. The characteristics of these datasets and specific information on how we trained our model are detailed next.

COMA: Introduced by Ranjan et al. [RBSB18], this dataset comprises registered 3D human faces consisting of 20,465 meshes spanning 12 extreme facial expressions. Each mesh contains 5,023 vertices. COMA has been successfully use in previous work [BBP*19, GCBZ19] for building parametric face models. For a fair comparison of the representative power of our model in the reconstruction task, we partition the dataset into training and testing subsets following an identical 9:1 ratio as in these previous works.

SDFM: This is another dataset with human faces, originally used by Chandran et al. [CBGB20] to build their semantic deep face model. This dataset currently includes 9,000 meshes of 375 distinct identities in 24 facial expressions. Each mesh includes 5,257 vertices on the front of the face and neck. The SDFM dataset allows us to demonstrate the application of our shape model in the task of disentangling facial identity and expression (see Section 4.8).

MANO: Introduced by Romero et al. [RTB17], this dataset comprises in total 1514 unique, registered 3D meshes of human hands, with a pair of left and right hands for 777 subjects. There are also mirrored counterparts for the left hand meshes leading to a final count of 2,291 meshes. The hands in this dataset span a considerable range of 3D shapes and consist of complex pose and shape deformations. Each mesh is database consists of 778 vertices. For our experiments, we considered only the 777 right hand meshes from this dataset and randomly left out 70 shapes for validation.

DFAUST: This human body dataset was proposed by Bogo et al. [BRPMB17] and contains 140 sequences of registered human body meshes. The dataset includes 10 subjects performing various movements and even includes dynamics. Each mesh in this dataset consists of 6,890 vertices. As in Zhou et al. [ZWL*20], we used 103 sequences for training, 13 sequences for validation, and the remaining 13 sequences for testing.

Architectural Details and Hyper-Parameters.
On COMA, SDFM, and MANO, our model as set up as follows:
Canonical shape: average of all shapes in the training dataset
Sampling strategy: all 3D vertices on both encoder and decoder
Code/token sizes: 64 dimensions in shape code, style code, and in all per-point input tokens.

On DFAUST, the model setup was:
Canonical shape: average of rest shapes in the training dataset
Sampling strategy: all 3D vertices on both encoder and decoder
Code/token sizes: 128 dimensions in shape code, style code, and in all per-point input tokens.

4.2. Shape Reconstruction and Interpolation

A basic task of any data-driven deformable model is shape reconstruction and its ability interpolate smoothly between shapes. Fig. 5 demonstrates the reconstruction accuracy for three different instances of our model, trained on COMA, MANO and DFAUST. In all three cases, our transformer-based autoencoder was able to accurately model the different classes of 3D shapes with very small errors of only a few millimeters. A quantitative evaluation of our Shape Transformer's performance on COMA, in comparison to other models in the related work, is shown in Table 1. As the table shows, our model provides the lowest average modeling error (and standard deviation) on this face dataset. The performance of our method on DFAUST is summarized in Table 2. Shape Transformer produces competitive results on the DFAUST benchmark while only being trained for 170 epochs as compared to the 300 epochs training for the other methods.On MANO, our Shape Transformer achieves a reconstruction error of 1.628 mm and 2.657 mm on the training and validation sets, respectively.

The ability of our nonlinear model to interpolate between two different 3D shapes (i.e., two shape codes) is illustrated in Fig. 6. Please refer to the supplemental video for more examples, and to

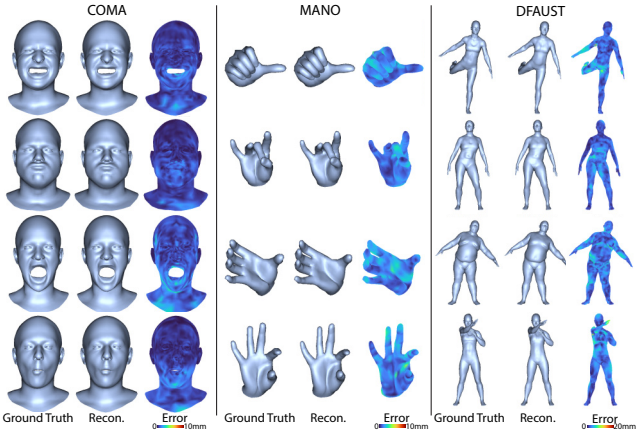

Figure 5: *Modeling errors of our transformer-based autoencoder on test samples of COMA, MANO, and DFAUST datasets.*

Table 1: *Quantitative comparison of face reconstruction (modeling) performance versus related work on the COMA dataset.*

Method	mean error (mm)	median error (mm)	time per epoch (s)
FeaStNet	0.523 ± 0.643	0.297	133.183
MoNet	0.526 ± 0.605	0.353	97.009
COMA [RBSB18]	0.470 ± 0.598	0.263	77.943
ChebyConv (K=9)	0.436 ± 0.562	0.242	86.627
Neural3DMM [BBP*19]	0.443 ± 0.560	0.245	107.137
SpiralNet++ [GCBZ19]	0.426 ± 0.538	0.238	30.417
DilatedSpiralNet++ [GCBZ19]	0.423 ± 0.534	0.236	29.181
Ours	0.413 ± 0.313	0.323	400

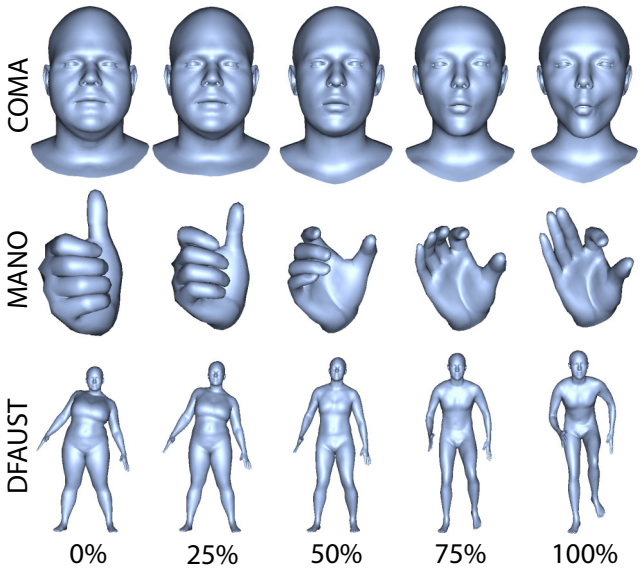

Figure 6: *We can interpolate between shape codes to create realistic in-between shapes. Here, the shapes at 0% and 100% are training data samples, the others are new interpolated shapes.*

Table 2: *Quantitative comparison on DFAUST*

Method	Training error (mm)	Validation error (mm)	Model size
Neural3DMM [BBP*19]	6.42	7.39	2.0m
MeshCNN* [HHF*19]	83.3	101.7	2.2m
Zhou et al.[ZWL*20]	3.73	5.01	1.9m
Ours	4.58	5.31	2.1m

visualize the smooth transition during interpolation. As the interpolation weight changes, the resulting shape progressively changes identity, pose, and expression as the shape code moves within the model's latent space. Note how the interpolations effectively capture nonlinear deformations and always correspond to plausible shapes, what would be difficult to obtain with a linear model.

Comparison to Implicit Models. Here we compare our Shape Transformer's performance against two implicit models. The first is that of DeepSDF [PFS*19] on the SDFM dataset. For this experiment, we exported each SDFM shape as a signed distance field, following Park et al. [PFS*19] by sampling 250K points in the normalized bounding volume of the dataset. DeepSDF uses a latent shape code of 512 dimensions and an 8-layer MLP with skip connections. Since DeepSDF is trained in an auto-decoder fashion (without an encoder), validation shapes must be fit (reconstructed) via an iterative optimization step. As a second comparison baseline, we take the implicit MLP architecture used by DeepSDF and train it to output per-point offsets from our canonical shape (i.e., no signed distance field, no self-attention). We trained these two baselines and a facial Shape Transformer for 200 epochs each. As shown in Table 3, our transformer-based model outperforms the two baselines, which showed similarly inferior generalization. Fig. 7 compares the generalization errors on a validation example and also shows the faster convergence of our Shape Transformer compared to a simple MLP that predicts per-point offsets from our canonical shape.

Decoding Context. As the attention mechanism in the Shape Transformer naturally depends on the set of vertices sampled for decoding, we provide an evaluation of the reconstruction performance of the COMA Shape Transformer while varying the number of simultaneously sampled locations on the canonical face surface. The result of this evaluation, in Fig. 9, shows that the attention produces reasonable deformations for different decoding densities, but that accuracy is best at higher resolutions. In a related experiment, we kept the number of sampled points constant but varied their spatial extent, to measure the effect of local and global correlations while decoding vertex sets. Intuitively, we reconstruct a vertex while forcing the attention weight to be zero for all vertices further away than a given distance. We repeat this experiment for 100 vertices on the face and for 8 different distance thresholds (from 10mm to 200mm), allowing the model to capture increasing global context. Fig. 10 plots the average reconstruction error as a function of the neighbourhood distance. Again, leveraging global context via self-attention gives the best results, showing that Shape Transformers can effectively balance both local and global spatial correlations. This helps our model outperform mesh convolutional (local) models (Table 1, Table 2) and also other implicit, single-point models based on MLPs without self-attention (Table 3).

Table 3: *Comparison* vs *implicit shape models on SDFM data.*

Method	Validation error (mm)
Implicit MLP	1.76
DeepSDF [PFS*19]	1.35
Ours	0.768

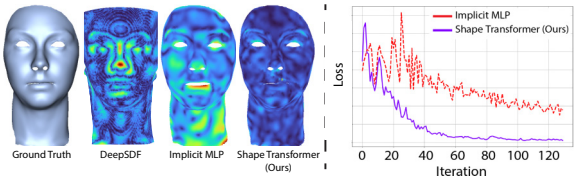

Figure 7: *Shape Transformer outperforms implicit shape models in face reconstruction (left), while also converging faster to a better solution when compared to per-point MLPs without self-attention.*

4.3. Topology Independence

One of the key benefits of the Shape Transformer model is its topology independence, where the model can be trained on one or many different topologies, and then evaluated on any new topologies without re-training. We illustrate the topological independence for the SDFM Shape Transformer in Fig. 1, and the MANO Shape Transformer in Fig. 8. Note that the model produces accurate overall geometry for all topologies, and also learns to increase the surface details when the mesh topology permits (*e.g.*, the Catmull-Clark subdivision result in Fig. 8 has long thin triangles in the middle of the palm, and the Shape Transformer is able to create more detailed wrinkles). Shape Transformer could thus perform as a valuable tool for mesh super-resolution/upsampling. Note that in order to support a new topology at test time, we require a one-time mapping of the new topology to the canonical manifold (*e.g.*, by registering a single new mesh onto the canonical shape). This is easily achieved using traditional mesh deformation tools that are readily available in 3D modeling packages [Com18] and in literature [WAT*11, SCOL*04].

4.4. Shape Completion

Face models occasionally have to deal with incomplete data, for example in image-based reconstruction tasks when parts of the face are occluded. It is often the task of the face model to complete missing regions in a plausible way. Here we show that Shape Transformers are very well suited to this task. Recall that the shape code in the Shape Transformer is obtained from an encoder which is also implemented as a transformer, and thus is also topology independent. This means that it is sufficient to supply only partial shapes to the encoder, in order to obtain the shape code for decoding. We illustrate this case of missing data on the SDFM Shape Transformer in Fig. 11 by manually cropping away large regions of vertices from shapes before encoding them, and evaluating the reconstruction errors. The Shape Transformer naturally handles the missing region and reconstructs plausible deformations even when only trained with complete information during training. An extreme case of missing input data is to sample only a subset of the vertices of the target mesh. Our shape encoder is able to still obtain plausible shape codes under such extreme cases, which we show by slowly reducing the number of encoded vertices in our COMA

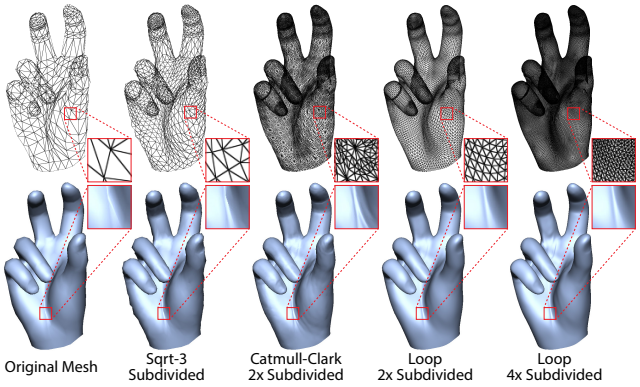

Figure 8: *Here we show the MANO Shape Transformer evaluated on several different topologies, even though the model was trained only once on the original topology.*

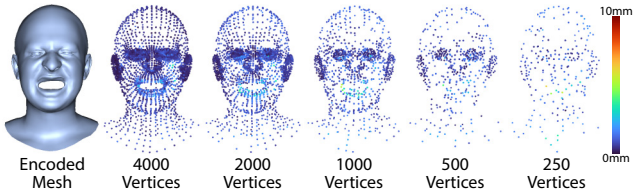

Figure 9: *We show reconstruction errors while varying the number of queried vertices in the decoder of the COMA Shape Transformer. Even with very few vertices, the sampled offsets remain accurate, demonstrating the robustness of self-attention in our decoder.*

Shape Transformer in Fig. 12. This experiment is valuable as it supports a real-world scenario of sparse facial tracking, *e.g.*, from detected landmarks or motion-capture dots. We demonstrate this use case in Fig. 13, where we show reconstruction errors for two different frames of a captured performance, evaluating two different marker layouts — 64 landmarks corresponding to a landmark detector, and 160 evenly placed markers corresponding to a motion-capture layout. Note that in this experiment we demonstrate an actor-specific use case, and thus we train a special actor-specific Shape Transformer such that the encoded shape codes represent expressions only. Please refer to the supplemental video for several complete sparse performance reconstructions.

4.5. Projecting New Shapes by Optimizing Shape Codes

As mentioned in Section 3, we have two options to compute a shape code for our Shape Transformers. One option, which have explored so far, relies on shape codes output by our transformer-based encoder. The second option is to fit the pre-trained Shape Transformer to a particular new 3D shape by iteratively optimizing for a new shape code that minimizes an L2-loss between the decoded 3D points and the 3D points on the new target shape. This second option can be beneficial for particular shapes that differ from those in the training data and for which the shape code output by the encoder is suboptimal. Thus, it can serve to initialize a subsequent optimization step. Furthermore, this optimization is now free to compute different shape codes for different regions of the face, further improving quality of fit and expressibility of the

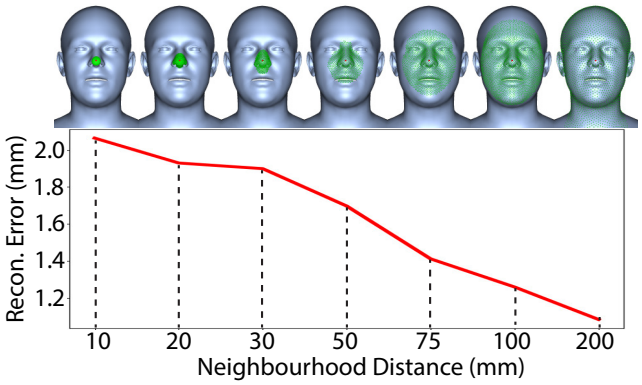

Figure 10: *We show the effect of local and global correlations by decoding a vertex with increasing neighborhood sizes (green). Higher accuracy is achieved when leveraging more global context.*

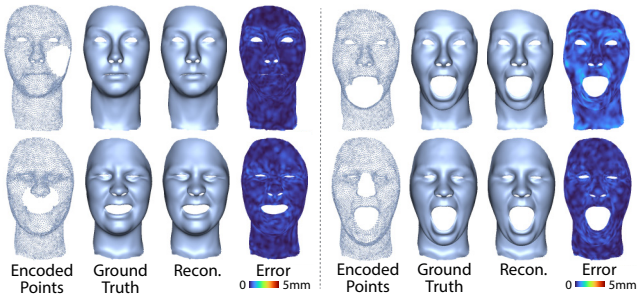

Figure 11: *Shape Transformers can be used for shape completion when large regions of the target are missing,* e.g.*, the cheek, chin, upper lip, and nose shown above on the SDFM Shape Transformer.*

model. We note that such a projection into the Shape Transformer's latent space assumes that correspondences are known between the target and canonical shapes. Fig. 14 shows an experiment on which we fit the Shape Transformer to two unseen subjects, with one local shape code per vertex. This projection takes 15 seconds per shape and is guided by gradient descent. Naturally, this corresponds to an over-parameterization of the model, leading to perfect fits with zero error. The purpose of this experiment however is to demonstrate how the Shape Transformer allows for the number of local shapes to be arbitrarily adjusted to match the desired accuracy at inference time. Interesting avenues for future work include extracting optimal patch layouts from the Shape Transformer such that a target face can be represented with as few codes as possible.

4.6. High-Quality Face Modeling

We now demonstrate the application of building a high-quality, identity-specific Shape Transformer from 24 facial scans of a single person. Each scan has over 500,000 vertices capturing fine detail down to the skin pores and fine wrinkles. This high-quality dataset was captured with a multiview stereo setup in controlled studio-like conditions [BHB*11]. This highly detailed dataset comprises about 36 million coordinates in total, while our trained Shape Transformer learns to represent it with about 1 million weights. We used all 24 shapes to train this Shape Transformer, with the goal of generating a high-quality model that can be interpolated or driven

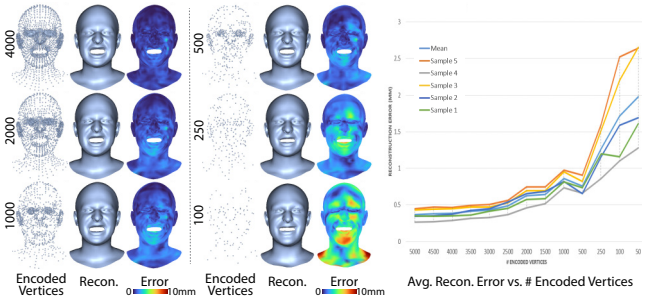

Figure 12: *Slowly decreasing the number of encoded vertices for the COMA Shape Transformer degrades the reconstruction naturally. Left: reconstructions for 6 different encodings from 4000 to 100 vertices. Right: avg. reconstruction errors for 5 different training samples increase as the number of encoded vertices decreases.*

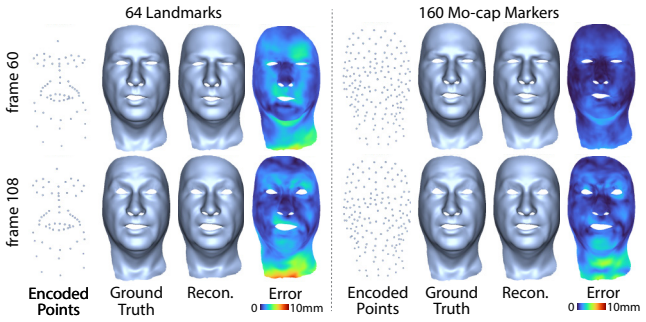

Figure 13: *We illustrate actor-specific Shape Transformer reconstruction from sparse points on 2 frames of a performance,* e.g.*, from 64 landmarks or 160 hand-placed motion-capture markers.*

by data in order to generate high-quality facial animation. As the shapes in this case are of extremely high resolution, they cannot be queried all at once like in our previous experiments due to GPU memory limitations. Thus, we opt for a simple training strategy where the encoder always encodes a fixed set of 20,000 canonical positions on the target scan. These points are sampled once using dart throwing such that they cover the entire face. On the decoder side, we randomly sample 20,000 points at each iteration and reconstruct the corresponding target offsets. This strategy allows us to consistently learn a single shape code per training sample on the encoder side while still allowing us to supervise the decoder with rich ground truth. At inference time, we also split the queried points into equally sized chunks that fit in memory and query them in sequence. The reconstruction accuracy for a few 3D shapes in the training data is shown in Fig. 15(a). Both the figure insets and the very small modeling errors (mostly below 0.25 mm) show that the model is able to effectively capture the fine geometric detail in the training shapes. Fig. 15(b) shows the novel, high-quality 3D shapes generated by the Shape Transformer as we interpolate between two of the original shapes. Besides generating dense, realistic geometry, this same Shape Transformer can be sampled at different topologies and spatial resolution, for different application scenarios (Fig. 1).

4.7. Model Analysis

Canonical Shape Changes. Since the Shape Transformer makes use of a canonical shape to sample queries during training and inference, a natural question to ask is about its reaction to off-surface queries? To understand this behavior, we modified the canonical shape *after* the Shape Transformer was trained and visualized the change in reconstructing a particular shape. Fig. 16 shows the effect of varying the canonical shape in 3 different ways: scale, non-rigid deformations and pose. The results seem to indicate the our model is more or less invariant to the scale of the canonical shape, while failing understandably for large changes in pose. Kindly refer to the supplemental video for more results of this experiment.

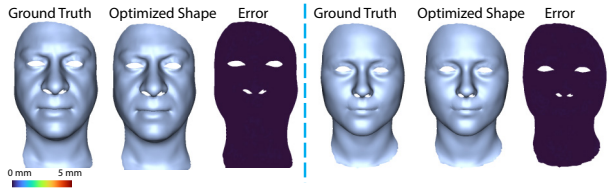

Figure 14: *We illustrate the arbitrary representative power of the Shape Transformer with an over-parameterized projection resulting in the perfect reconstruction of the ground truth shape.*

Style Mixing. One more application that is allowed by our style modulated decoder is style mixing. By style mixing, we refer to the use of different shape codes to modulate each layer of the decoder. We first obtain a set of candidate shape codes to style mix by encoding a set of shapes through our encoder. The different shape codes can then be mix and matched to modulate the decoder at different layers. Mixing 2 different shapes in the 4 layer COMA and MANO Shape Transformer decoder is demonstrated in Fig. 17.

4.8. Disentangling Identity and Expression

Previous face models showed several benefits of being able to disentangle the deformation caused by identity changes from the deformation caused by expression changes [VBPP05, CBGB20]. Our new model is also able to disentangle identity and expression, by splitting the shape code into two separate parts. During training, we then constrain the identity part to have the same code for all expressions of the same subject. In addition, the two parts in the code allows us to modulate the two semantic aspects separately, opening up several applications as illustrated in the following.

Identity Synthesis. In Fig. 18, each row shows many different sampled identities obtained by change the part of the shape code corresponding to identity, while leaving the expression code fixed. Along each column, the identity is fixed but the expression code changes.

Identity/Expression Interpolation. A disentangled Shape Transformer allows to interpolate only the identity or expression part of the shape code, as illustrated on SDFM data in Fig. 19 (left). Furthermore, the interpolation can be applied only locally (*e.g.*, to only half of the face as shown in Fig. 19 (right)) by using a different shape code with a subset of the queried 3D points given to the transformer. Note that different identities and expressions were used for the two different experiments (global versus local interpolation).

Blendshape-Based Performance Retargeting. A Shape Transformer with disentangled identity and expressions codes naturally facilitates application in the retargetting of facial performances. Fig. 20 shows the modeled sequence of facial expressions captured

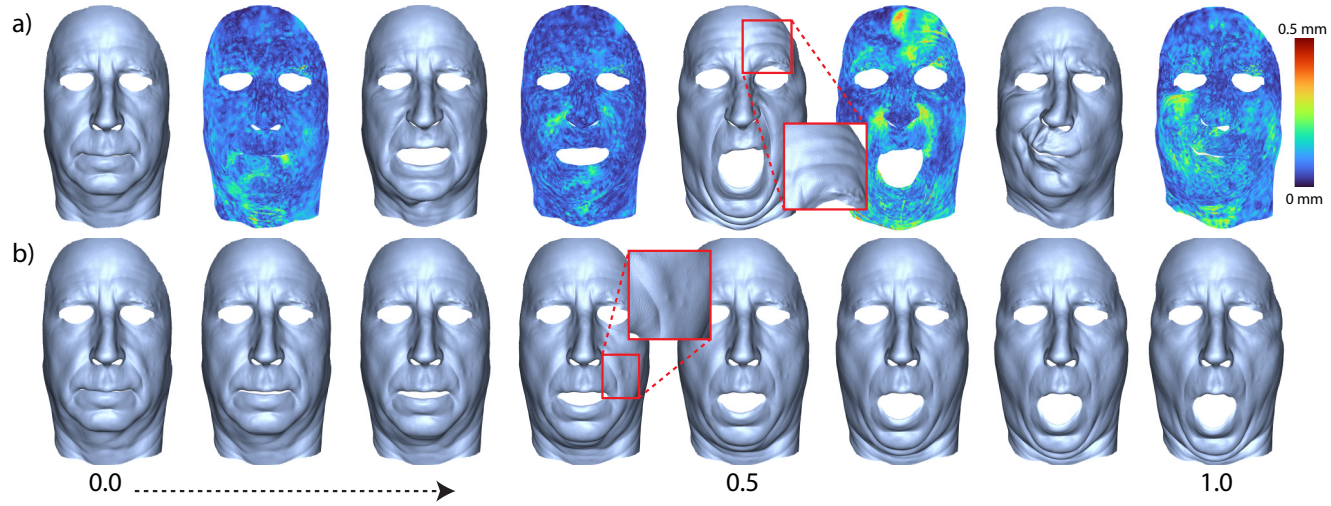

Figure 15: *High-quality, person-specific Shape Transformer trained on 24 facial scans with fine geometry detail capturing skin pores and wrinkles: (a) modeling error on 4 training shapes; (b) novel, high-quality shapes generated via interpolation of two training shapes.*

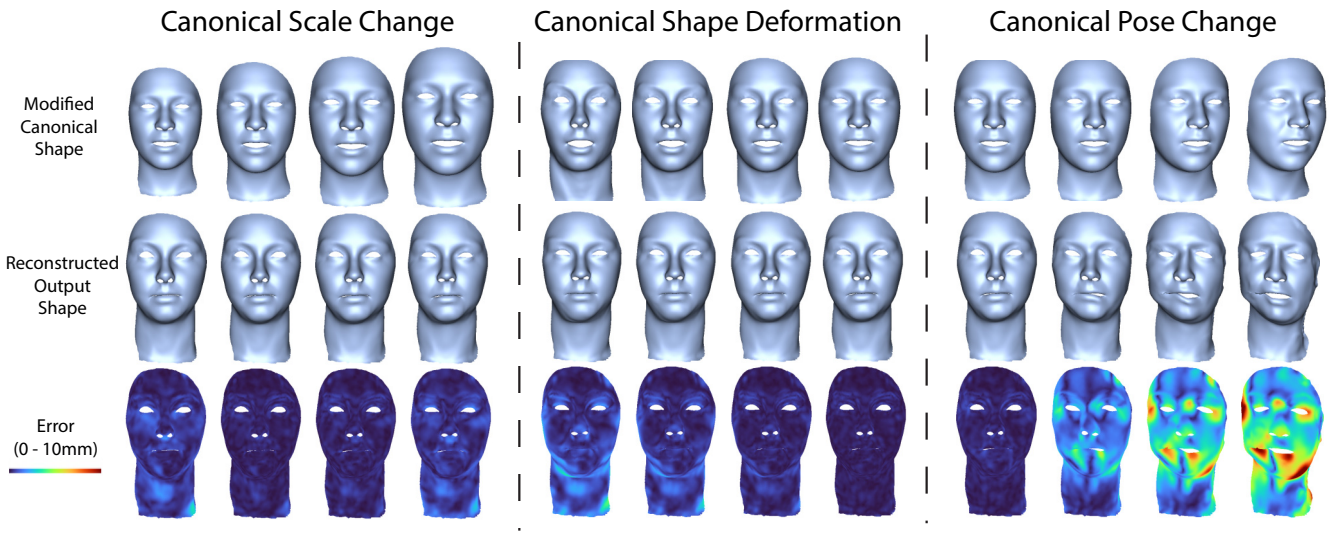

Figure 16: *Here we show the effect of modifying the canonical shape at inference after our transformer was trained. We show the effects that scale, non-rigid deformations and pose have on the trained model. These results indicate that the model is more or less robust to changes in scale and small non-rigid deformations. Changes in pose seem to affect the predicted offsets to a greater extent.*

for the source actor, which are then retargetted to two other actors by transferring only the expression codes.

5. Conclusion

Parametric shape models are among the most common tools used in computer graphics applications. These data-driven priors are usually built from a corpus of 3D scans, and are often used to represent 3D faces, hands and bodies (among other items). A number of different parametric models exist, ranging from linear to nonlinear and local to global, but they all share the common limitation that they usually dictate the extent of spatial correlations that occur during deformation. Furthermore, they are all designed to work on a fixed topology. In this work we present Shape Transformers, a new nonlinear parametric 3D shape model based on transformer architectures, which uses the transformer's "self-attention" mechanism to automatically learn nonlinear spatial correlations. Additionally, our model is topologically independent: it can be trained once and then evaluated on any mesh topology. We demonstrate how our new model can be applied on various datasets, including 3D faces, hands and full bodies, and illustrate several applications like reconstruction, shape completion, performance capture and retargeting. We believe our transformer-based 3D shape model shows strong potential in computer graphics and vision applications.

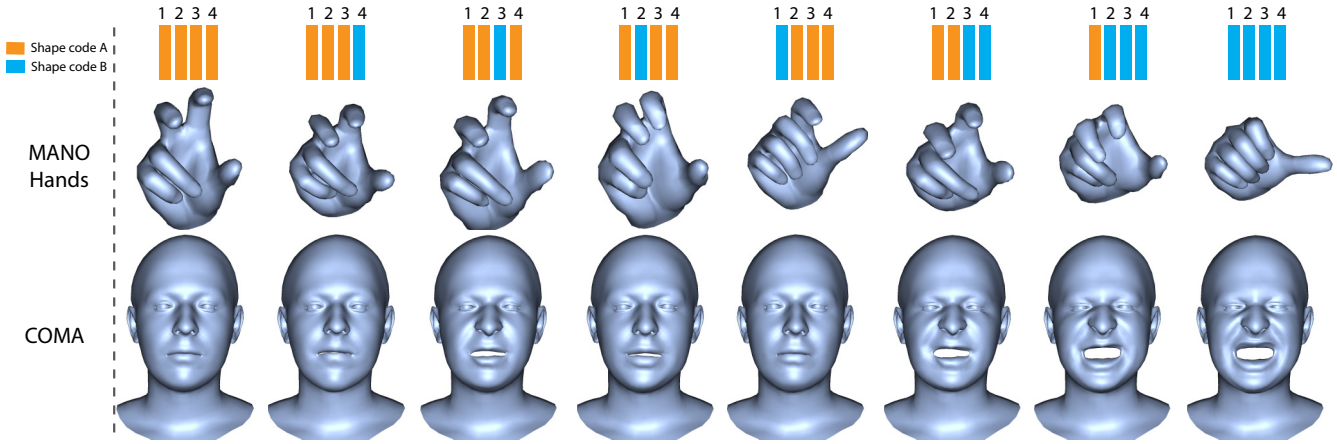

Figure 17: *Here we show the effect of modulating the different layers of our decoder with different shape codes. Though during training, we always train with a single shape code applied at all layers, our model is able to produce smoothly deforming shapes when mixing styles at different layers at inference time. The effect of such style mixing is displayed on two datasets consisting of hands and faces.*

Figure 18: *Our Shape Transformer with separate shape codes for identity and expression allows for semantically sampling novel identities, while retaining the ability to control their expression.*

References

[ABWB19] ABREVAYA V. F., BOUKHAYMA A., WUHRER S., BOYER E.: A decoupled 3d facial shape model by adversarial training. In *IEEE ICCV* (2019). 3

[BBP*19] 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 *IEEE ICCV* (2019), pp. 7212–7221. 3, 6, 7, 8

[BHB*11] BEELER T., HAHN F., BRADLEY D., BICKEL B., BEARDSLEY P., GOTSMAN C., SUMNER R. W., GROSS M.: High-quality passive facial performance capture using anchor frames. *ACM Trans. Graphics (Proc. SIGGRAPH) 30* (2011), 75:1–75:10. 9

[BRPMB17] BOGO F., ROMERO J., PONS-MOLL G., BLACK M. J.: Dynamic FAUST: Registering human bodies in motion. In *IEEE CVPR* (2017). 6, 7

[BV99] BLANZ V., VETTER T.: A morphable model for the synthesis of 3d faces. In *Siggraph* (1999), vol. 99, pp. 187–194. 1, 2

[CBGB20] CHANDRAN P., BRADLEY D., GROSS M., BEELER T.: Semantic deep face models. In *Int. Conf. on 3D Vision* (2020), pp. 345–354. 2, 3, 6, 7, 10

[CK21] CHEN Z., KIM T.-K.: Learning feature aggregation for deep 3d morphable models. In *IEEE CVPR* (2021). 3

[Com18] COMMUNITY B. O.: *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL: http://www.blender.org. 8

[CZ19] CHEN Z., ZHANG H.: Learning implicit fields for generative shape modeling. *IEEE CVPR* (2019). 3

[DBK*21] DOSOVITSKIY A., BEYER L., KOLESNIKOV A., WEISSENBORN D., ZHAI X., UNTERTHINER T., DEHGHANI M., MINDERER M., HEIGOLD G., GELLY S., USZKOREIT J., HOULSBY N.: An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR* (2021). 3

[DBV16] DEFFERRARD M., BRESSON X., VANDERGHEYNST P.: Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS* (2016), p. 3844–3852. 3

[DCLT19] DEVLIN J., CHANG M.-W., LEE K., TOUTANOVA K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT* (2019), pp. 4171–4186. 5

[DLJ*20] DENG B., LEWIS J. P., JERUZALSKI T., PONS-MOLL G., HINTON G., NOROUZI M., TAGLIASACCHI A.: Nasa neural articulated shape approximation. In *ECCV* (2020). 3

[ENTC*21] EL-NOUBY A., TOUVRON H., CARON M., BOJANOWSKI P., DOUZE M., JOULIN A., LAPTEV I., NEVEROVA N., SYNNAEVE G., VERBEEK J., ET AL.: Xcit: Cross-covariance image transformers. *arXiv preprint arXiv:2106.09681* (2021). 4, 5, 6

[EST*20] EGGER B., SMITH W. A. P., TEWARI A., WUHRER S., ZOLLHOEFER M., BEELER T., BERNARD F., BOLKART T., KORTYLEWSKI A., ROMDHANI S., THEOBALT C., BLANZ V., VETTER T.: 3d morphable face models - past, present and future. *ACM Trans. Graphics 39*, 5 (2020). 2

[GCBZ19] GONG S., CHEN L., BRONSTEIN M., ZAFEIRIOU S.: Spiralnet++: A fast and highly efficient mesh convolution operator. In *IEEE ICCV Workshops* (2019), pp. 4141–4148. 2, 3, 6, 7

[GYH*20] GROPP A., YARIV L., HAIM N., ATZMON M., LIPMAN Y.: Implicit geometric regularization for learning shapes. In *Proc. Machine Learning and Systems*. 2020, pp. 3569–3579. 3

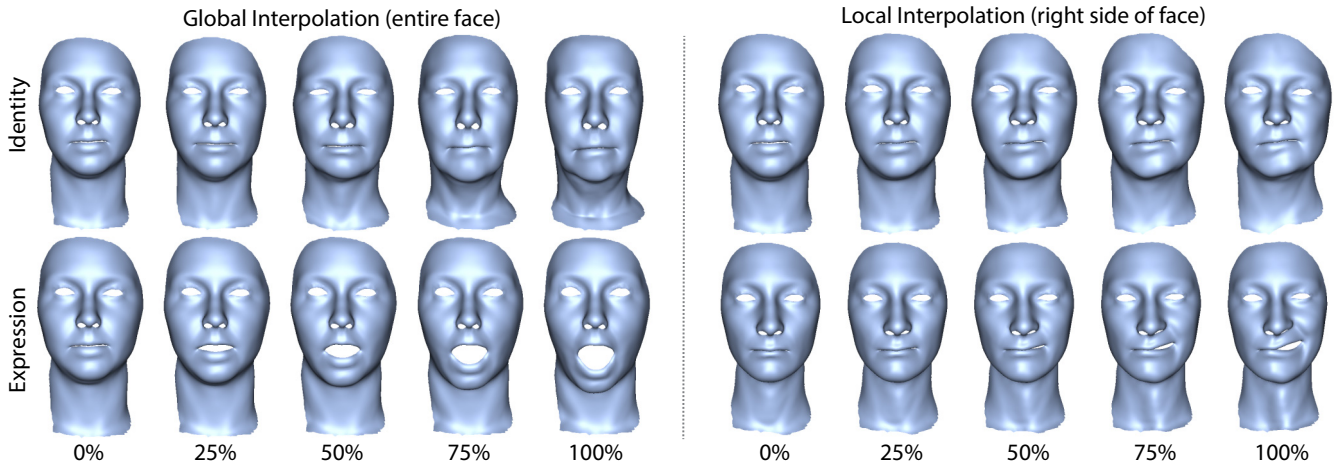

Figure 19: *A disentangled COMA Shape Transformer allows to interpolate only the identity (top row) or the expression (bottom row) while keeping the other part of the shape code fixed. This interpolation can happen by changing the shape code globally for the entire face (left) or even locally for a part of the face (right). Note that the left and right experiments are performed with different identities and expressions.*

Figure 20: *A Shape Transformer with disentangled identity/expression codes naturally supports retargetting of facial expressions from a captured performance across different identities.*

[HHF*19] HANOCKA R., HERTZ A., FISH N., GIRYES R., FLEISHMAN S., COHEN-OR D.: Meshcnn: A network with an edge. *ACM Trans. Graphics (Proc. SIGGRAPH) 38*, 4 (2019). 2, 3, 8

[KCZO08] KAVAN L., COLLINS S., ZARA J., O'SULLIVAN C.: Geometric skinning with approximate dual quaternion blending. *ACM Trans. Graphics 27*, 4 (2008). 3

[Kv05] KAVAN L., ŽÁRA J.: Spherical blend skinning: A real-time deformation of articulated models. In *Proc. I3D* (2005), p. 9–16. 3

[LMR*15] LOPER M., MAHMOOD N., ROMERO J., PONS-MOLL G., BLACK M. J.: SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia) 34*, 6 (2015), 248:1–248:16. 3

[LWL21a] LIN K., WANG L., LIU Z.: End-to-end human pose and mesh reconstruction with transformers. In *IEEE CVPR* (2021). 3, 5

[LWL21b] LIN K., WANG L., LIU Z.: Mesh graphormer. In *IEEE ICCV* (2021). 3

[NVW*13] NEUMANN T., VARANASI K., WENGER S., WACKER M., MAGNOR M., THEOBALT C.: Sparse localized deformation components. *ACM Trans. Graphics (Proc. SIGGRAPH Asia) 32*, 6 (2013). 2

[PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: Deepsdf: Learning continuous signed distance functions for shape representation. In *IEEE CVPR* (June 2019). 3, 8

[RBSB18] RANJAN A., BOLKART T., SANYAL S., BLACK M. J.: Generating 3d faces using convolutional mesh autoencoders. In *ECCV* (2018). 3, 6, 7

[RTB17] ROMERO J., TZIONAS D., BLACK M. J.: Embodied hands: Modeling and capturing hands and bodies together. *ACM Trans. Graphics (Proc. SIGGRAPH Asia) 36*, 6 (2017). 6, 7

[SCOL*04] SORKINE O., COHEN-OR D., LIPMAN Y., ALEXA M., RÖSSL C., SEIDEL H.-P.: Laplacian surface editing. In *SGP* (2004), pp. 179–188. 8

[TDITM11] TENA J. R., DE LA TORRE F., MATTHEWS I.: Interactive region-based linear 3d face models. *ACM Trans. Graphics (Proc. SIGGRAPH) 30*, 4 (2011), 76:1–76:10. 2

[VBPP05] VLASIC D., BRAND M., PFISTER H., POPOVIĆ J.: Face transfer with multilinear models. *ACM Trans. Graphics 24*, 3 (2005), 426–433. 2, 10

[VSP*17] VASWANI A., SHAZEER N., PARMAR N., USZKOREIT J., JONES L., GOMEZ A. N., KAISER L. U., POLOSUKHIN I.: Attention is all you need. In *NeurIPS* (2017), vol. 30. 2, 3, 4

[WAT*11] WILSON C. A., ALEXANDER O., TUNWATTANAPONG B., PEERS P., GHOSH A., BUSCH J., HARTHOLT A., DEBEVEC P. E.: Facial cartography: interactive scan correspondence. In *SCA* (2011). 8

[WBGB16] WU C., BRADLEY D., GROSS M., BEELER T.: An anatomically-constrained local deformation model for monocular face capture. *ACM Trans. Graphics (Proc. SIGGRAPH) 35*, 4 (2016), 115. 2

[WBZB20] WANG M., BRADLEY D., ZAFEIRIOU S., BEELER T.: Facial expression synthesis using a global-local multilinear framework. *Comp. Graph. Forum (Proc. Eurographics) 39*, 2 (2020), 235–245. 2

[YTB*21] YENAMANDRA T., TEWARI A., BERNARD F., SEIDEL H.-P., ELGHARIB M., CREMERS D., THEOBALT C.: i3dmm: Deep implicit 3d morphable model of human heads. In *IEEE CVPR* (June 2021), pp. 12803–12813. 3

[ZWL*20] ZHOU Y., WU C., LI Z., CAO C., YE Y., SARAGIH J., LI H., SHEIKH Y.: Fully convolutional mesh autoencoder using efficient spatially varying kernels. In *NeurIPS* (2020). 3, 7, 8