Learning to Stabilize Faces

J. Bednarik [ID] and E. Wood [ID] and V. Choutas [ID] and T. Bolkart [ID] and D. Wang [ID] and C. Wu [ID] and T. Beeler [ID]
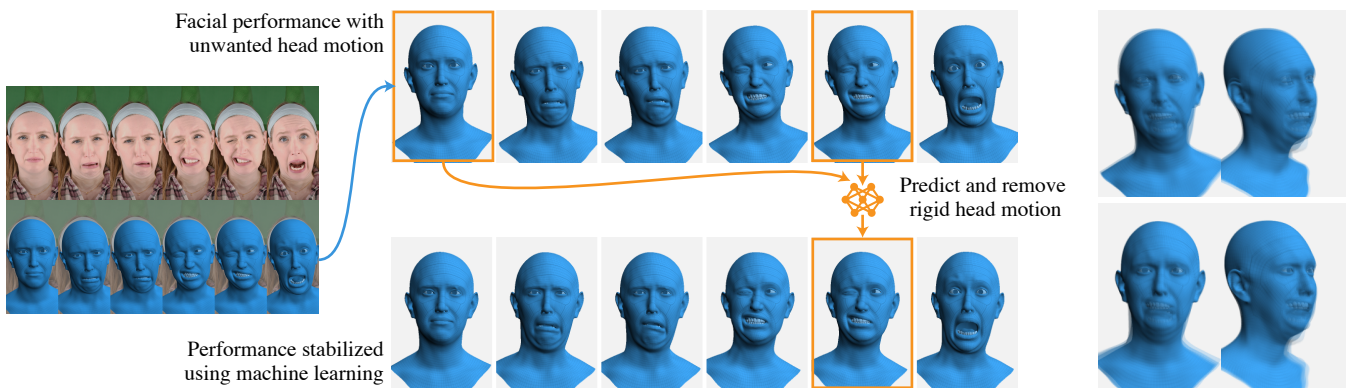
Google

Figure 1: *Our method stabilizes face meshes using machine learning. Given an input pair of face meshes, we predict the rigid transform between the two, and remove it. Shown on the right are all face meshes overlaid. After stabilization, the rigid head parts appear aligned.*

Abstract

Nowadays, it is possible to scan faces and automatically register them with high quality. However, the resulting face meshes often need further processing: we need to stabilize *them to remove unwanted head movement. Stabilization is important for tasks like game development or movie making which require facial expressions to be cleanly separated from rigid head motion. Since manual stabilization is labor-intensive, there have been attempts to automate it. However, previous methods remain impractical: they either still require some manual input, produce imprecise alignments, rely on dubious heuristics and slow optimization, or assume a temporally ordered input. Instead, we present a new learning-based approach that is simple and fully automatic. We treat stabilization as a regression problem: given two face meshes, our network directly predicts the rigid transform between them that brings their skulls into alignment. We generate synthetic training data using a 3D Morphable Model (3DMM), exploiting the fact that 3DMM parameters separate skull motion from facial skin motion. Through extensive experiments we show that our approach outperforms the state-of-the-art both quantitatively and qualitatively on the tasks of stabilizing discrete sets of facial expressions as well as dynamic facial performances. Furthermore, we provide an ablation study detailing the design choices and best practices to help others adopt our approach for their own uses.*

1. Introduction

High-fidelity face data is generally captured using multiple cameras, where a 3D mesh of fixed topology is non-rigidly registered to a collection of images. This is either done using traditional multi-view stereo and model-based optimization [EST*20], or more modern deep learning techniques [BLB23, LLB*21, LCC*22].

Often, a capture session will involve a single subject performing multiple expressions. When comparing meshes corresponding to different expressions, one observes inter-sample vertex motion caused by two phenomena. First, the skin deforms as the subject

contracts underlying facial muscles to perform expressions, and second, the whole head rigidly translates and rotates as the subject moves their neck and body. This rigid motion can be small, e.g. accidental shifts of the head in a sequence of static expressions, or large, e.g. dramatic head shakes in an expressive performance. For many use cases, we are only interested in the former, with the latter representing undesirable residual motion that pollutes the data.

The goal of *stabilization* is hence to remove unwanted rigid head motion, such as the one shown in Fig. 1. This is an essential step for building personalized face rigs for anima-

tion [ARL*09, SEL17], facial deformation transfer and retargeting [BWP13, LYYB13, CCGB22], and creating data-driven linear bases for 3D Morphable Models (3DMM) [EST*20].

Stabilization can be done manually by a skilled artist, but this is highly labor-intensive, so attempts have been made to automate it. Automatic stabilization would be easy if some parts of the face never moved during expressions, but unfortunately, this is not the case. The ultimate solution might track and align the skull itself, e.g. using X-Ray, but since this is impractical we must do our best with the visible parts of the face. Alas, high-quality fully automatic solutions are not yet readily available. Existing solutions either rely on imprecise Procrustes alignment of the skin vertices [WBLP11, BWP13, WSS18, CCWL18], require manually annotated landmarks [BB14, WBGB16], have prohibitive constraints on the input format [FNH*17, LLD18], or only produce approximate results by unposing a 3DMM [LBB*17].

We present the first learning-based method for stabilization. We treat stabilization as a regression problem: given an input pair of meshes, we use a neural network to predict the unwanted rigid transform between the two underlying skulls. We train our network using a large and diverse synthetic dataset, generated by randomly sampling both a 3DMM and a database of registered faces. Crucially, we exploit the fact that 3DMMs have a stable skull by design, which lets us freely recombine identity and expression into novel training samples. We show that such a model trained on synthetic data works well on real face meshes and achieves state-of-the-art accuracy both visually and quantitatively.

2. Related Work

Being an essential step of many face synthesis and analysis tasks, stabilization has been studied both on its own and as a part of a larger problem. Traditional approaches rely on rigid Procrustes alignment [Gow75] either directly, or combined with ICP [AHB87] when mesh correspondences are not known [WBLP11, BWP13] To prevent bad alignments often caused by jaw movement, these approaches typically only use the upper face region for alignment.

A different body of work achieves higher accuracy than Procrustes by considering the real physiology of the human skull and skin [ZBGB19]. [BB14, WBGB16] first estimate the shape of the underlying skull and optimize its pose via anatomically motivated heuristics involving skin thickness and skin-bone sliding. While accurate, the methods are not fully automatic and require per-subject one-time skull shape initialization, which involves costly manual landmark annotation.

A fully automated method was proposed in [LLD18], which assumes that given a reference coordinate frame of the skull, every facial surface point moves around its rest position. The authors rely on a heuristic linking the sharpness of the per-point position histogram to the stabilization quality and design a corresponding optimization scheme. The method requires an input in the form of a smooth facial performance and thus cannot be used to stabilize arbitrary expression pairs.

Head stabilization is often addressed as a step within face reconstruction and tracking pipelines [FNH*17, WSS18, CCWL18].
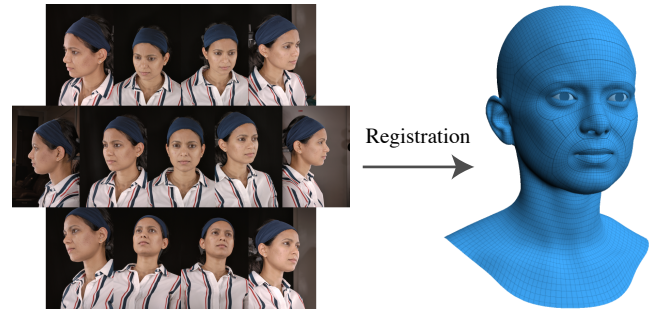

Figure 2: *Our dataset includes* 38 360 *face meshes which were registered to multi-view images. These registrations are used for building our 3DMM and sampling random expressions.*

The method of [FNH*17] initializes the stabilization using Procrustes alignment, followed by finding axis-aligned rotations of the reconstructed mesh that best explain the observation. This method relies on an optimization scheme and requires the availability of reconstructed eyeballs. [WSS18] convert the meshes to geometry images [GGH02] and show that Procrustes alignment can perform well as long as it is applied only on a proper region of the head which is learned in a data-driven fashion. Similarly, [CCWL18] divide the template face mesh into segments contributing to the rigid stabilization optimization with different weights, which are a function of the input. Similarly to [LLD18], the approach only works for facial performances and thus cannot be used to stabilize expression sets.

Other approaches use 3DMMs to stabilize faces. A common practice is to fit the model to the observed data and then use the model parameters to undo any undesirable global motion due to rigid transformation or neck rotation [EST*20, LBB*17]. The main drawback is the sensitivity to the quality of the 3DMM fits. Any residual fitting error is arbitrarily distributed between the model parameters and the global rigid motion. Furthermore, 3DMM parameters underlying the registered meshes are not always available, which is typically the case for modern deep-learning-based solutions [BLB23, LLB*21, LCC*22].

In contrast to prior work, our approach is fully automatic. We do not require a temporally consistent sequence of frames; our method works on just two meshes at a time. Finally, we only use a 3DMM to train our method; at inference time we require only 3D meshes.

3. Methodology

In Sec. 3.1 we first formalize the problem and our desired solution. Then, in Sec. 3.2, we describe our 3DMM which plays a key role in our approach. Next, in Sec. 3.3, we explain how to predict the rigid transform between two heads using a neural network. Finally, we provide details regarding synthesizing training data and training our network in Secs. 3.3 and 3.4.
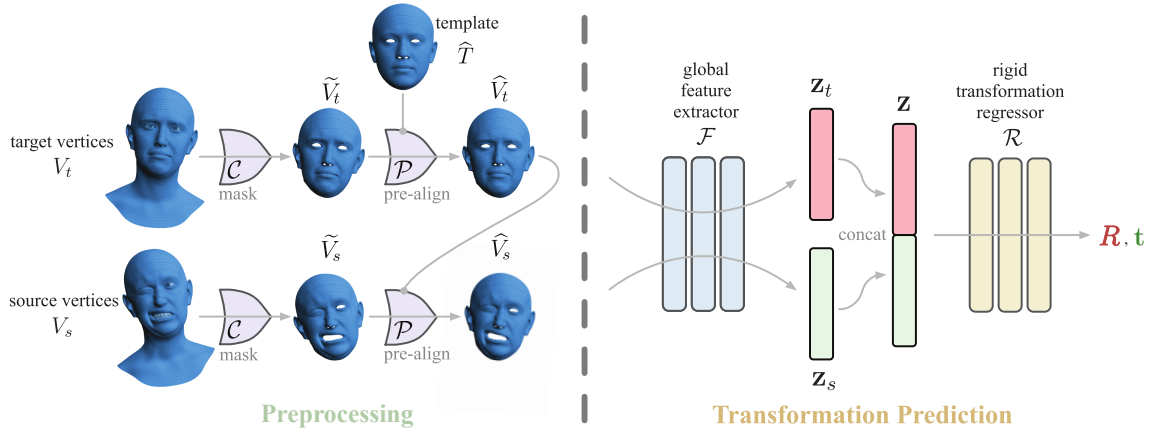
Figure 3: *The architecture of the rigid transformation predictor. The network takes a pair (V_s, V_t) of source and target skin vertices of the same subject on the input and predicts the rotation R and translation **t** which best aligns the input pair on the output.*

3.1. Problem Formulation

Our goal is to remove unwanted rigid transformation between any two head meshes from a single subject so that the underlying skulls are aligned.

Formally, let $H = (V, W)$ be a head consisting of N_V observable exterior points $V \in \mathbb{R}^{4 \times N_V}$ and N_W unobservable skull points $W \in \mathbb{R}^{4 \times N_W}$ in homogeneous coordinates. Given two misaligned heads $H_s = (V_s, W_s)$ and $H_t = (V_t, W_t)$, representing a *source* and a *target* facial expression of the same subject, our goal is to find a rigid transformation $S^* \in \mathbb{R}^{4 \times 4}$ which best aligns the source and target skulls, i.e. $S^* = \arg\min_S \mathcal{E}(W_s, W_t, S)$, where

$$\mathcal{E}(W_s, W_t, S) = \|SW_s - W_t\|_F. \tag{1}$$

Note, that in the ideal noiseless scenario, the skulls align perfectly and $\mathcal{E}(W_s, W_t, S^*) = 0$.

Since we only observe the facial exterior points V_i rather than the skull points W_i, we cannot directly minimize the energy $\mathcal{E}$ of Eq. 1. Instead, our aim is to find a function

$$\mathcal{S}(V_s, V_t) : \mathbb{R}^{2 \times 4 \times N_V} \to \mathbb{R}^{4 \times 4}, \tag{2}$$

which infers the transformation S^* by only considering observable vertices V_s, V_t.

We propose modeling $\mathcal{S}$ as a neural network trained in a supervised way to minimize the energy $\mathcal{E}$ of Eq. 1 for any pair of exterior vertices (V_s, V_t). However, for an observed V_i, the mapping $V_i \to W_i$ is generally unknown and thus the energy $\mathcal{E}$ cannot be readily evaluated during the training.

To address this problem, we propose using a 3DMM to sidestep the missing link between the facial exterior and the skull V, W, thus evaluating the energy $\mathcal{E}$ of Eq. 1 even when W is unknown. The details are described in the following sections.

3.2. 3DMMs to the Rescue

Following the literature [LBB∗17, PKA∗09, WBH∗21], we formulate our 3DMM as a function $\mathcal{M}_\Psi(\Theta) : \mathbb{R}^{|\Theta|} \to \mathbb{R}^{4 \times N_V}$ which,

given model data Ψ, takes parameters Θ and generates $N_V = 17\,821$ vertices in homogeneous coordinates. Our 3DMM model is that of [WBH∗21] with the following differences: (i) we define a custom topology, (ii) we employ 3rd party expression blendshapes [pol], and (iii) we use custom artist-painted skinning weights, see Fig. 5 and Sec. 3.2 for more details. Like [WBH∗21], our 3DMM includes the eyeballs and teeth.

Model parameters $\Theta = (\beta, \phi, \theta, \tau)$ include identity $\beta \in \mathbb{R}^{|\beta|}$, expression $\phi \in \mathbb{R}^{|\phi|}$, rotations of $K = 4$ joints $\theta \in \mathbb{R}^{K \times 3}$ and global translation $\tau \in \mathbb{R}^3$. Model data $\Psi = (T, J, \mathbf{I}, \mathbf{E}, \mathbf{Q}, \mathbf{W}, P)$ includes the template face vertices $T \in \mathbb{R}^{4 \times N_V}$, template joint locations $J \in \mathbb{R}^{4 \times K}$, linear vertex identity basis $\mathbf{I} \in \mathbb{R}^{|\beta| \times 4 \times N_V}$, linear expression basis $\mathbf{E} \in \mathbb{R}^{|\phi| \times 4 \times N_V}$, linear joint identity basis $\mathbf{Q} \in \mathbb{R}^{|\beta| \times 4 \times K}$, skinning weights $\mathbf{W} \in \mathbb{R}^{K \times N_V}$, and skeletal hierarchy $P \in \mathbb{Z}^K$.

Formally, vertices are generated as follows:

$$\mathcal{M}_\Psi(\Theta) = \mathcal{L}(V_{\text{bind}}, \mathbf{X}, \mathbf{W})$$

$\mathcal{L}$ is a standard Linear Blend Skinning function that transforms bind-pose vertices V_{bind} by skinning transforms $\mathbf{X}$, with weights $\mathbf{W}$ controlling how each vertex is affected by each joint.

$$\mathbf{X} = \mathcal{X}(J_{\text{bind}}, \theta, \tau; P)$$

$\mathcal{X}$ builds skinning transforms by propagating per-joint rotations θ down the kinematic tree defined by P, taking bind-pose joint locations J_{bind} and root joint translation τ into account. Vertices and joints in the bind pose are determined using linear bases:

$$V_{\text{bind}} = T + \sum_{i=1}^{|\beta|} \beta_i \mathbf{I}_i + \sum_{i=1}^{|\phi|} \phi_i \mathbf{E}_i \quad \text{and} \quad J_{\text{bind}} = J + \sum_{i=1}^{|\beta|} \beta_i \mathbf{Q}_i$$

Data provenance. Our vertex identity basis $\mathbf{I}$ is computed by performing PCA on a dataset of registered 3D heads obtained using a multi-camera capture studio. We use a custom optimization-based registration pipeline with standard steps of detecting facial landmarks, enforcing photometric consistency and regularizing surface deformation as done before [FNH∗17, LBB∗17, CCWL18]. The
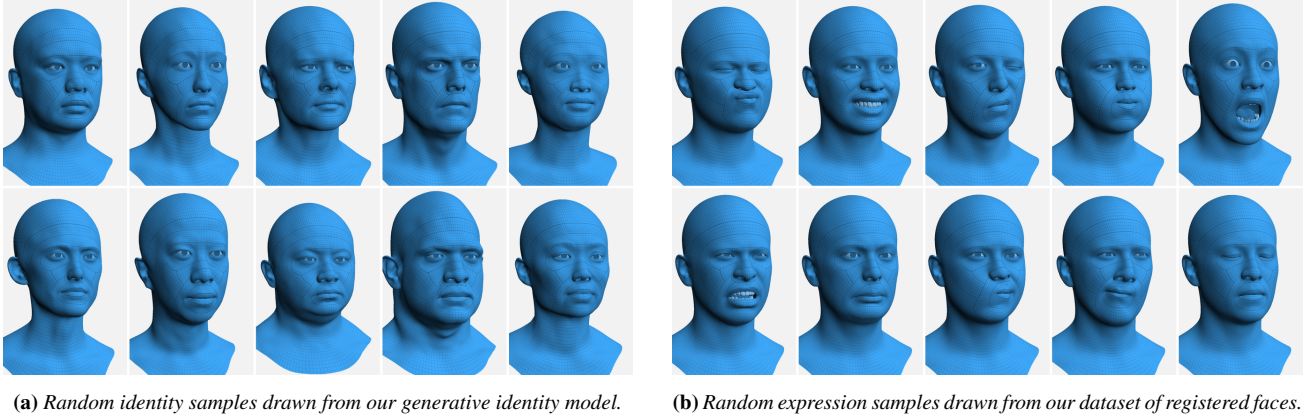
(a) *Random identity samples drawn from our generative identity model.*

(b) *Random expression samples drawn from our dataset of registered faces.*

Figure 4: *Our stabilization neural network is trained with synthetic data only. We synthesize realistic and diverse faces by mixing random identities (Fig. 4a) with random expressions (Fig. 4b).*

dataset contains 38 360 frames of 2 519 subjects performing a variety of different expressions. Each registration contains a mesh with N_V vertices and estimated 3DMM parameters Θ. See Fig. 2 for an example. Our expression basis **E** is FACS-like [EF78] and was authored by an artist, specifically, we purchased a set of blendshapes Polywink [pol]. Each basis controls a localized area and guarantees a stable skull. Template joint positions J and skinning weights **W**, too, were designed by an artist, see Fig. 5. Joint identity basis **Q** is computed as an average of the vertices corresponding to **I** weighted by **W**.
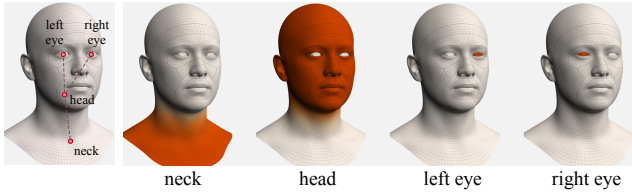

Figure 5: *Schematic view of the kinematic chain of our 3DMM (left) and skinning weights corresponding to the joints (right).*

3DMMs have a stable skull. Generally, 3DMMs do not explicitly model the skull. But, by design, changing expression parameters ϕ should only deform the exterior vertices V, without affecting the hypothetical underlying skull W. Thus, for any pair of source and target expressions where $V_s = \mathcal{M}(\beta, \phi_s, \theta, \tau)$ and $V_t = \mathcal{M}(\beta, \phi_t, \theta, \tau)$, the corresponding skulls should align, i.e. $W_s = W_t$.

Suppose we rigidly transform the target skin vertices V_t by any rigid transformation Z, the skull follows and $W_t = ZW_s$. By plugging this in the Eq. 1 we get

$$\mathcal{E}(W_s, W_t, S) = \|SW_s - ZW_s\|_{\mathrm{F}} = \|(S - Z)W_s\|_{\mathrm{F}}, \qquad (3)$$

and thus $Z = \arg\min_S \mathcal{E}(W_s, W_t, S)$.

In other words, our 3DMM lets us generate facial exterior vertices V of various expressions and arbitrary rigid transformations Z which can be used to evaluate the energy $\mathcal{E}$ even without having access to the actual skull W. We use this convenient property to

generate expressions V_i perturbed by rigid transformations Z_i (see Sec. 3.3), and train a neural network to predict Z_i which, as we showed above, is the sought-after minimum of energy $\mathcal{E}$ of Eq. 1.

3.3. Data-driven Transformation Predictor

We model the function $\mathcal{S}$ as a neural network, which takes a pair of misaligned face vertices and produces a rigid transformation S.

Preprocessing. To make the problem easier for $\mathcal{S}$ we preprocess input meshes by (i) removing less-relevant parts of the mesh that may distract the network (see Sec. 4.5), and (ii) pre-aligning each mesh using a naive method. Our preprocessing function $(V_s, V_t) \rightarrow (\widehat{V}_s, \widehat{V}_t)$ is defined as follows. Let $\mathcal{C} : \mathbb{R}^{4 \times N_V} \rightarrow \mathbb{R}^{4 \times N_C}$ be a mask extracting $N_C < N_V$ vertices corresponding to the frontal face area (see Fig. 3), $\mathcal{P}(X, Y)$ a Procrustes alignment of points X to Y, and $\widehat{T} \in \mathbb{R}^{4 \times N_C}$ a centered and axis aligned neutral face mesh. We obtain the $(\widehat{V}_s, \widehat{V}_t)$ as (see also Fig. 3):

$$\widetilde{V}_j = \mathcal{C}\left(V_j\right), j \in \{s, t\}$$
$$\widehat{V}_t = \mathcal{P}\left(\widetilde{V}_t, \widehat{T}\right),$$
$$\widehat{V}_s = \mathcal{P}\left(\widetilde{V}_s, \widehat{V}_t\right).$$

Although our 3DMM contains teeth, we do not include them in masked faces $\widetilde{V}_i$. This is because the teeth are rarely observed in practice, so we cannot rely on them being accurately registered. After extracting the frontal face region via $\mathcal{C}$, the meshes $\widetilde{V}_i$ have 6 663 vertices.

Transformation predictor. We model the function $\mathcal{S}$ as neural network with trainable parameters Ω as

$$\mathcal{S}_{\Omega}(V_s, V_t) = \mathcal{R}\left(\mathcal{F}(V_s) \oplus \mathcal{F}(V_t)\right), \qquad (4)$$

where $\mathcal{F} : \mathbb{R}^{4N_V} \rightarrow \mathbb{R}^L$ is a global feature extractor mapping flattened skin vertices V_i into a L-dimensional latent code $\mathbf{z}_i$, the operator $\oplus$ represents vector concatenation and $\mathcal{R} : \mathbb{R}^{2L} \rightarrow \mathbb{R}^{4 \times 4}$ is a rigid transformation predictor producing the transformation S consisting of rotation R and translation $\mathbf{t}$ best aligning the input pair

(V_s, V_t). See Fig. 3 for the architecture overview. In practice, both $\mathcal{F}$ and $\mathcal{R}$ are modeled as multi-layer perceptrons (MLP) accepting a flattened vector of mesh vertices and the combined latent code respectively.

Synthesizing training data. We synthesize training data for $\mathcal{S}_\Omega$ using our 3DMM. Each training sample is a random pair of faces $(\widehat{V}_s, \widehat{V}_t)_i$ along with the corresponding ground-truth (GT) transformation $\overline{S}_i$. Generating realistic and diverse random samples using parametric models is a long-standing open problem, as supported by the body of relevant literature [BKL*16, PCG*19, ZBX*20, DRC*22, TAL*22].

Our strategy is simple. We start with a training dataset of N registered meshes and their estimated 3DMM parameters $\mathbf{B} = \{\beta_i | 1 \leq i \leq N\}$, $\mathbf{\Phi} = \{\phi_i | 1 \leq i \leq N\}$. We then fit a normal distribution $\mathcal{N}(\beta_\mu, \mathrm{Diag}(\beta_\sigma))$ to $\mathbf{B}$. For a single training pair, we draw one random identity vector from the identity distribution (see Fig. 4a), and draw two random expression vectors directly from $\mathbf{\Phi}$ (see Fig. 4b). We slightly perturb the expression vectors with a small amount of noise $\mathcal{N}(\mathbf{0}, \mathrm{Diag}(\epsilon_\phi \mathbf{1}))$, and pose the meshes with $\mathcal{M}$. Sampled meshes are preprocessed the same way as shown in Fig. 3 but each Procrustes pre-alignment $\mathcal{P}$ is complemented by a small random rigid transformation to mimic the noise encountered when aligning real-world meshes.

Let $\mathcal{Q}(X, Y) \rightarrow Q \in \mathbb{R}^{4 \times 4}$ compute a rigid Procrustes alignment of points X to Y (i.e. $\mathcal{Q}$ is equivalent to $\mathcal{P}$ but retrieves the transformation), and let $\mathcal{G}(\epsilon_R, \epsilon_T)$ sample a random rigid transformation given limits $\epsilon_R, \epsilon_T \in \mathbb{R}$ (see Appendix). The process of generating one data sample $(\widehat{V}_s, \widehat{V}_t, \overline{S})$ is described in Alg. 1.

One might worry: is synthetic data good enough for $\mathcal{S}_\Omega$ to generalize to real-world registered meshes? What about the dreaded domain gap? Fortunately, our results in Sec. 4.4 indicate that our dataset synthesis scheme works, and $\mathcal{S}_\Omega$ does indeed generalize to real-world data.

Algorithm 1: Generating one training sample.

Input : $\mathcal{M}, \widehat{T}, \beta_\mu, \beta_\sigma, \mathbf{\Phi}, \epsilon_\phi, \epsilon_R, \epsilon_T$
Output: $\widehat{V}_s, \widehat{V}_t, \overline{S}$

//Sample identity, expression parameters and tf. noise.
1 $\beta \sim \mathcal{U}(\beta_\mu - 3\beta_\sigma, \beta_\mu + 3\beta_\sigma)$
2 $\phi_s, \phi_t \sim \mathcal{U}(\mathbf{\Phi}) + \mathcal{N}(\mathbf{0}, \mathrm{Diag}(\epsilon_\phi \mathbf{1}))$
3 $S_{\epsilon_s} = \mathcal{G}(\epsilon_R, \epsilon_T), S_{\epsilon_t} = \mathcal{G}(\epsilon_R, \epsilon_T)$
   //Sample source and target vertices.
4 $\widetilde{V}_s = \mathcal{C}(\mathcal{M}(\beta, \phi_s))$
5 $\widetilde{V}_t = \mathcal{C}(\mathcal{M}(\beta, \phi_t))$
   //Noisily align target to template.
6 $S_{t \rightarrow \widehat{T}} = \mathcal{Q}\left(\widetilde{V}_t, \widehat{T}\right)$
7 $\widehat{V}_t = S_{\epsilon_t} S_{t \rightarrow \widehat{T}} \widetilde{V}_t$
   //Noisily align source to target.
8 $S_{s \rightarrow t} = \mathcal{Q}\left(\widetilde{V}_s, \widehat{V}_t\right)$
9 $\widehat{V}_s = S_{\epsilon_s} S_{s \rightarrow t} \widetilde{V}_s$
   //Compute the GT transformation.
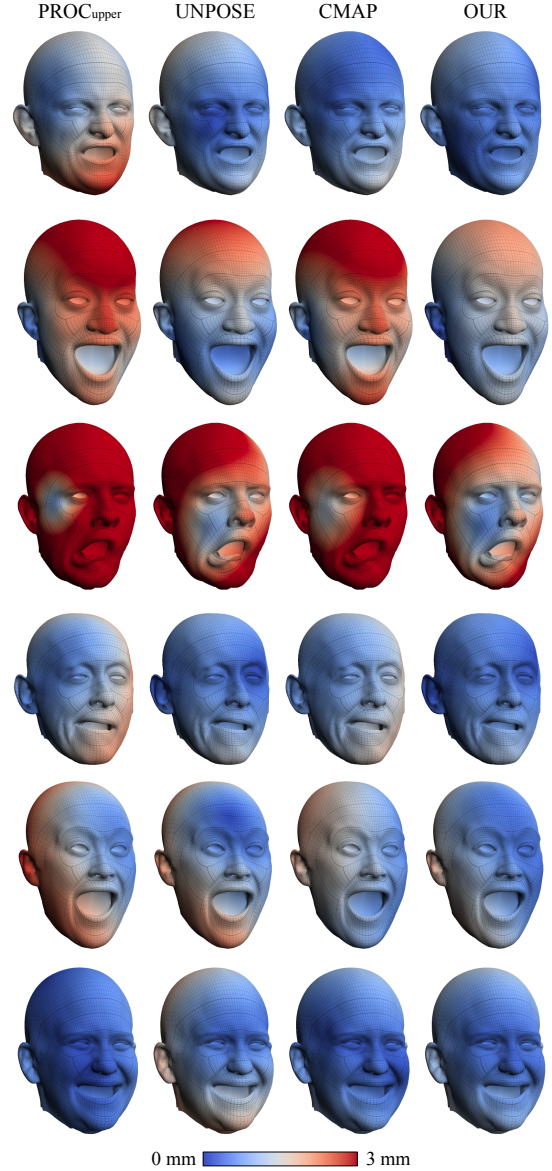10 $\overline{S} = S_{\epsilon_t} S_{t \rightarrow \widehat{T}} S_{s \rightarrow t}^{-1} S_{\epsilon_s}^{-1}$

Figure 6: *The stabilization error visualized as an error map in the range* $[0, 3]$ *mm.*

Loss functions. Since we know the GT transformation $\overline{S}_i$ for each source-target pair, we train the transformation predictor in a supervised way with the following loss function:

$$\mathcal{L} = \mathcal{L}_R + \alpha_T \mathcal{L}_T, \tag{5}$$
$$\mathcal{L}_R = \|\overline{R} - R\|_F,$$
$$\mathcal{L}_T = \|\overline{\mathbf{t}} - \mathbf{t}\|,$$

where $\overline{R}, \overline{\mathbf{t}}$ are the GT rotation and translation respectively, and α_T is a scalar weighing the two loss terms.
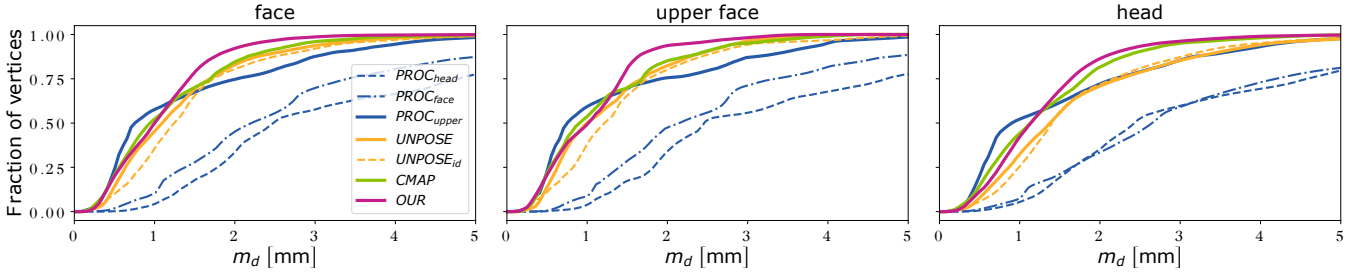
Figure 7: *Percentage of correct keypoints (PCK) computed on three different head regions, Head, Face and Upper shown for all the tested methods.*

3.4. Training Details

The feature extractor $\mathcal{F}$ is modeled as a 3-layered Multilayer Perceptron (MLP) with sizes $(1024, 512, 512)$, the dimension of the latent space $L = 256$ and the transformation regressor $\mathcal{R}$ is a 3-layered MLP with sizes $(512, 512, 512)$. All MLP layers use ReLU except the last linear layer. The network is trained with the Adam optimizer and learning rate $5e^{-5}$ for $125\,000$ iterations. The weight parameter α_T of Eq. 5 is empirically set to 1 in all our experiments. The output translation $\mathbf{t}$ is represented as a 3D vector and the output rotation R as a 6D representation [ZBJ*19].

4. Experiments

In the following text we describe the test data and metrics we use to evaluate our method (OUR), as well as previous methods we compare against. Furthermore, we present an ablation study and analysis, thus shedding light on the inner workings of our method.

4.1. Training and Evaluation Data

We use a real-world dataset of registered meshes introduced in Sec. 3.2. The GT stabilization of the samples is unknown, however, and hard to obtain [BB14, WSS18, CCWL18]. We follow [BB14] and manually stabilize 45 expressions across 15 randomly selected subjects.

We only select those 45 expressions with visible upper teeth, as it is the only visible head part, which does not deform with changing facial expressions. For one expression per subject, we manually annotate 2D keypoints on the 6 upper frontal teeth in 5 camera views and triangulate them to obtain a 3D polyline.

Finally, to stabilize an annotated expression to the remaining ones, an operator manually transforms the source mesh until (i) the 3D teeth polyline projected to the camera view visually aligns with the visible teeth, and (ii) until the two meshes appear visually aligned in a 3D viewer too, see Fig. 8 for examples of annotated expressions. We obtain 30 source frames with GT transformations to their corresponding target frames.

All the expressions of the 15 subjects are removed from the training dataset, and the annotated set is split into 5 validation and 10 test subjects with corresponding 10 validation and 20 test expressions.

4.2. Metrics

Let S and $\bar{S}$ be the predicted and GT transformation respectively, and $V = SV_s$ and $\bar{V} = \bar{S}V_s$ be the predicted and GT stabilized source vertices respectively. Furthermore, let $V_j^{(i)}$ denote j-th vertex of the i-th dataset sample, where $1 \leq j \leq M$ and $1 \leq i \leq N$. We use the following metrics to quantitatively evaluate OUR and the competing methods.

Mean vertex distance (m_d). The metric is computed as $m_d = \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \|\bar{V}_j^{(i)} - V_j^{(i)}\|$.

Maximum vertex distance (m_x). Following [BB14], we complement m_d with the maximum vertex distance averaged over the N samples: $m_x = \frac{1}{N} \sum_{i=1}^{N} \max_j \|\bar{V}_j^{(i)} - V_j^{(i)}\|$.

Area Under Curve (m_{AUC}). We also compute the percentage of correct keypoints (PCK) metric popular in human pose estimation domain [ZWC*23, CFW*22, KAB20], evaluate it in the range of $[0, 5]$ mm and report the area under curve (AUC).

In order to gain further insight into which facial parts are the most challenging for the methods to align, we evaluate all the metrics on various head mesh masks as explained in Sec. 4.4.

4.3. Competing Methods

Existing approaches are often subject to prohibitive requirements, such as manual keypoint annotation [BB14, WBGB16], temporally consistent sequence of input meshes [LLD18, CCWL18] or reconstructed eyeballs [FNH*17], none of which are needed by OUR. For fair comparison, we thus select the following methods which have looser restrictions.

Procrustes alignment. Since registered face meshes are in correspondence, we can use Procrustes [Gow75] to rigidly align them [VBPP05]. To limit the impact of potentially irrelevant facial areas, we evaluate Procrustes on three vertex subsets corresponding to the full *head*, *face* and *upper face*, the last one being suggested in [WBLP11, BWP13], which we refer to as PROC$_{head}$, PROC$_{face}$ and PROC$_{upper}$.

3DMM unposing. Each registered face comes with estimated 3DMM parameters so we can stabilize the pair by unposing both

Table 1: *The comparison of the methods on three different head regions. The metrics m_d and m_x are measured in milimeters, m_{AUC} in percent.*

	face			upper face			head		
method	$m_d \downarrow$	$m_x \downarrow$	$m_{AUC} \uparrow$	$m_d \downarrow$	$m_x \downarrow$	$m_{AUC} \uparrow$	$m_d \downarrow$	$m_x \downarrow$	$m_{AUC} \uparrow$
$PROC_{head}$	3.46 ± 2.37	14.76	40.30	3.48 ± 2.32	11.85	40.15	3.34 ± 2.32	14.76	42.03
$PROC_{face}$	2.84 ± 2.21	14.22	49.67	2.71 ± 1.86	10.85	50.08	3.39 ± 2.51	14.22	42.82
$PROC_{upper}$	1.40 ± 1.28	8.36	72.15	1.38 ± 1.24	5.92	72.32	1.53 ± 1.33	8.36	69.63
$UNPOSE$	1.32 ± 0.94	7.75	73.50	1.25 ± 0.80	4.44	74.73	1.72 ± 1.21	7.99	65.85
$UNPOSE_{id}$	1.46 ± 0.98	8.08	70.82	1.42 ± 0.93	5.59	71.40	1.71 ± 1.10	8.08	65.86
$CMAP$	1.20 ± 0.86	6.15	75.71	1.19 ± 0.85	4.52	75.96	1.33 ± 0.89	**6.15**	73.14
OUR	$\mathbf{1.08 \pm 0.64}$	**5.37**	**78.03**	$\mathbf{1.07 \pm 0.63}$	**3.85**	**78.39**	$\mathbf{1.29 \pm 0.78}$	6.36	**74.05**

the source and the target to the bind pose, as done in [LBB*17]. Using our 3DMM introduced in Sec. 3.2, and given source and target parameters $\Theta_x = (\beta_x, \phi_x, \theta_x, \tau_x)$, where $x \in \{s, t\}$, the unposed mesh vertices are obtained as $\mathcal{M}(\beta_x, \phi_x, \mathbf{0}_\theta, \mathbf{0}_\tau)$ with $\mathbf{0}_\theta, \mathbf{0}_\tau$ being zero rotation and translation parameters. The predicted transformation S is thus fully defined by $\theta_s, \tau_s, \theta_t, \tau_t$. Since only expressions of the same subject are stabilized, we also consider the cases where we refit the 3DMM model to the source and target meshes so that the identity parameters β are the same, i.e. $\beta_s = \beta_t$. We refer to these two flavours as UNPOSE and UNPOSE$_{id}$. Also note that unposing a 3DMM comes with a particular drawback. When fitting the 3DMM to the observed meshes, the non-zero fitting error is arbitrarily distributed between the global pose and the 3DMM parameters, which precludes perfect stabilization. Furthermore, the necessary step of fitting the 3DMM to the observed meshes renders the method slower than the other approaches.

Learned confidence map. We reimplemented the confidence-map-based rigid stabilization module of [WSS18], and trained it on our own data. We found the original formulation produced unsatisfactory results, on par with the basic PROC$_{head}$. Therefore, we modified the method to encourage high global contrast and local spatial consistency, and found the best set of hyper-parameters on the validation set, please see the Appendix for more details. We only experiment with the modified variant and refer to it as CMAP.

4.4. Results

We now compare our method to existing work quantitatively and qualitatively using the annotated test set described in Sec. 4.1. Furthermore, an ablation study motivates our design choices.

Quantitative results. We evaluate all methods on the metrics defined in Sec. 4.2. To reveal any biases towards specific parts of the head, we measure error across three regions. *Head* discards the neck which is not relevant to stabilization quality, *Face* considers the frontal face area, and *Upper* considers the forehead and nose only, which is typically the most robust to changing expressions.

As can be seen in Tab. 1, OUR yields the best performance improving upon the best performing baseline CMAP by 10%. Among all the variants of Procrustes alignment, PROC$_{upper}$ performs the best, which is in line with the assumptions made by prior work [BWP13, WBLP11], but still falls short of the remaining methods.

While UNPOSE improves on the Procrustes alignment, it suffers from the underlying non-zero fitting error, as discussed in Sec. 4.3, manifesting in imprecise stabilization. Also note, that despite guaranteeing consistent identity parameters, UNPOSE$_{id}$ underperforms UNPOSE which we attribute to worse 3DMM fitting caused by decreased model flexibility.

CMAP yields the best result among the prior work. Remarkably, the method is based on Procrustes alignment, but shows that seemingly rudimentary rigid alignment can yield decent results, if one learns alignment mask from the data. Despite that, the linear nature of the Procrustes alignment limits the performance when compared to the ML-based solution of OUR.

To demonstrate the robustness of the methods, we further show curves evaluating the PCK metric in the range $[0,5]$ mm in Fig. 7. It is evident that OUR generally outperforms the other methods. Interestingly, PROC$_{upper}$ produces stabilization with a much higher fraction of vertices below the error of 1 mm, but deteriorates above this mark. This phenomenon is due to the fact that PROC$_{upper}$ focuses solely on the forehead while ignoring the rest of the face contributing to the high overall error.

Qualitative results. Figures 6 and 8 contain a visual comparison of the methods on the task of stabilizing arbitrary expression pairs from the test set. In Fig. 6 we show the spatial error of stabilizing an expression pair using GT and predicted transformation. In Fig. 8 we transform the GT 3D teeth poly-line (see Sec. 4.1) from a source to a target expression using transformations predicted by each method, project the poly-line to two camera views of the target, and overlay it with the GT one. We only show the best performing variants of Procrustes (PROC$_{upper}$) and 3DMM unposing (UNPOSE).

The qualitative results are in line with the aforementioned observations. PROC$_{upper}$ produces the least precise stabilization, while UNPOSE and CMAP generally yield high-quality alignment, where the errors start appearing for more extreme and/or asymmetric facial expressions such as the top-right subject moving their jaw sideways or the top left subject snarling in Fig. 8. OUR, too, suffers from misalignments on more complex expressions but is generally more robust.

The quality of our results is best assessed in videos, so please refer to our supplementary webpage.
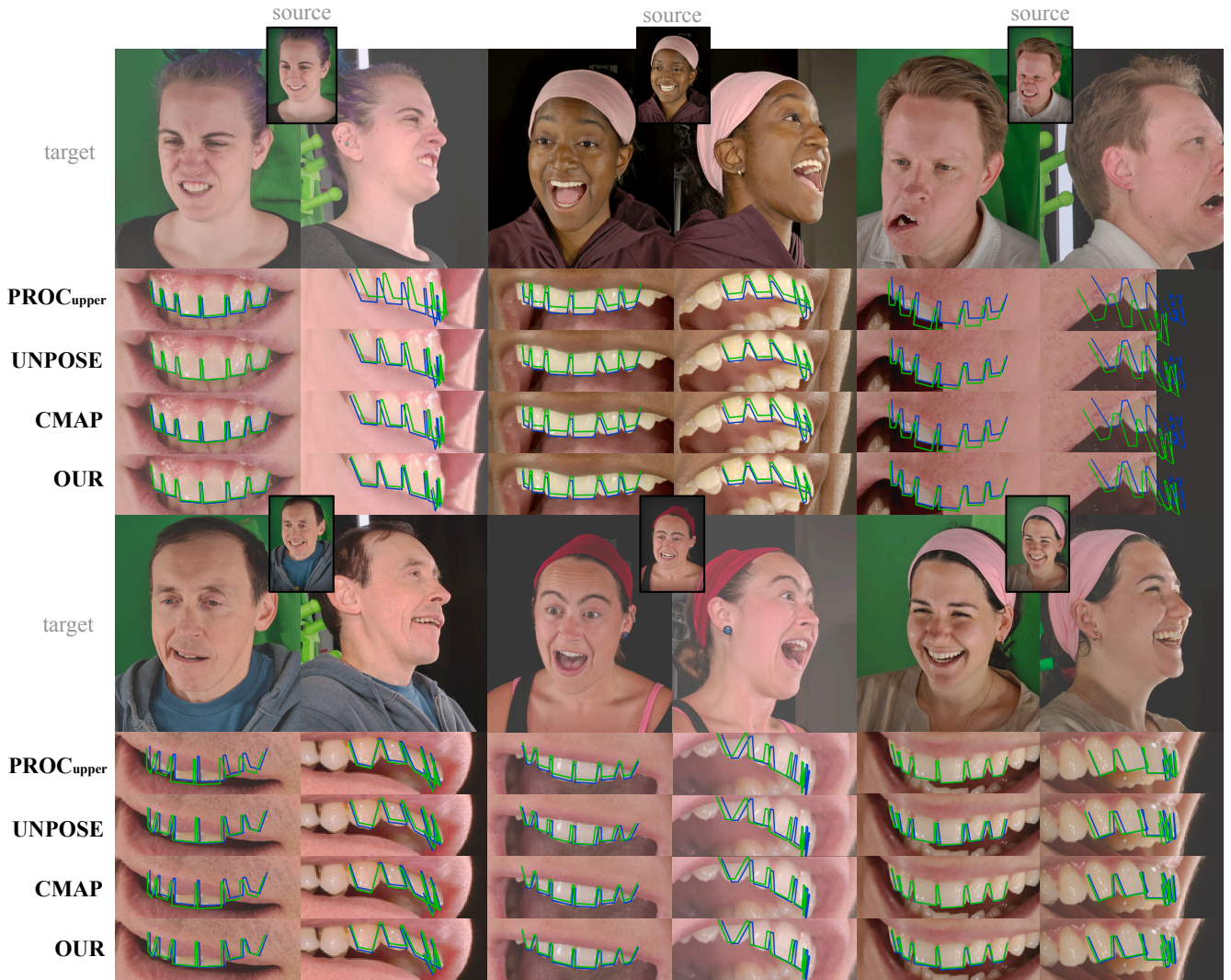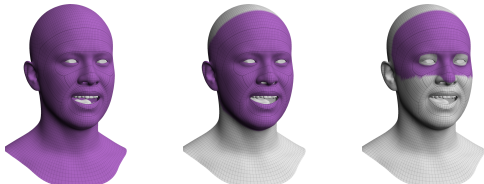
Figure 8: *Visual comparison of the methods showing predicted (green) and GT (blue) 3D teeth line projected to the front and side views.*

Table 2: *Choice of the head mesh region on the input to OUR matters, Face&Neck performs the best on the validation set.*

metrics	Full	Face&Neck	Superhero
$m_d \downarrow$	1.29 ± 0.89	$\mathbf{1.15 \pm 0.77}$	1.23 ± 0.88
$m_x \downarrow$	5.90	4.85	$\mathbf{4.47}$
$m_{\mathbf{AUC}} \uparrow$	73.98	$\mathbf{76.76}$	75.05

4.5. Ablation Study and Method Analysis

Here we discuss the choices taken to design and train the model.

Head coverage. The registered head meshes in our dataset contain the full head and neck. However, the goal is to align the underlying skulls and furthermore, as shown in the literature [WBLP11, BWP13], not all parts of the face are equally relevant for stabilization. Theoretically, an MLP-based model, which takes flattened meshes on the input, should be able to learn to ignore the irrelevant parts. However, we experimentally show this not to be the case.

We choose three regions of the input meshes shown in Tab. 2 and train OUR only on the corresponding vertices. The result of the evaluation on the validation dataset is summarized in Tab. 2. It is clear that both feeding the model irrelevant mesh regions (neck and back of the head of *Full*), and pruning the parts that potentially

carry a useful signal (cheeks and jaw of *Superhero*) are detrimental to the performance.
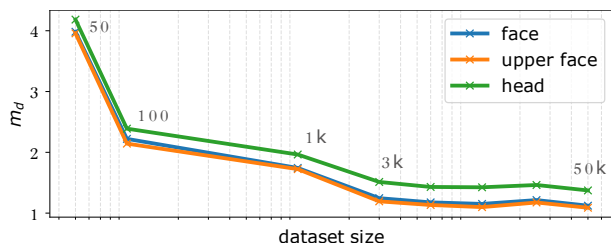

Figure 9: *Impact of the size of the training set on the performance of our method.*

Training dataset size. As discussed in Sec. 3.3, a 3DMM allows us to generate a dataset of unlimited size. How big a dataset is necessary to reach a good performance? We trained OUR on the randomly generated datasets of various sizes and evaluated them on the validation dataset, the results are presented in Fig. 9. It can be seen that training our model on datasets larger than 3000 samples yields diminishing returns. While OUR was trained on a dynamically generated (thus de facto infinite) dataset, in scenarios with limited computation resource, much smaller datasets will suffice.

5. Conclusion

We presented a novel learning-based approach for rigidly stabilizing face meshes with arbitrary expressions. Synthetic data played a key part. We designed a simple but effective scheme for synthesizing training pairs of misaligned expressive faces using a 3DMM. We used the resulting dataset to train a neural network that directly predicts the rigid transform between any two input meshes so the underlying skulls are aligned.

Our method does not require the input meshes to be temporally consistent. That is, any pair of arbitrarily differing expressions can be stabilized. This makes our approach generally useful for practitioners seeking to stabilize a continuous facial performance or random expression sets alike, where the typical downstream applications include character deformation transfer or building a custom human head parametric model, where spurious global transformations degrade the expressive capacity of the model.

As our method operates on independent mesh pairs, it can be heavily parallelized allowing for fast processing of large datasets. For example, a performance of 1000 frames is stabilized in ∼ 6 seconds on an Nvidia A100 GPU. Finally, we show through quantitative and qualitative experiments that our approach outperforms prior work.

Limitations remain. First, our method relies on sampling a 3DMM, which first needs to be built. However, such a task is well understood as evidenced by a rich body of literature [EST*20, LBB*17, PKA*09]. Second, the method operates on registered meshes with a common topology and thus it is intended to be used with production studio capture pipelines rather than in-the-wild raw 3D scans for which the method would not work out-of-the-box. Finally, to sample diverse yet realistic faces at training time, our method needs access to a large dataset of 3DMM parameters fitted to scan meshes.

References

[AHB87] ARUN K. S., HUANG T. S., BLOSTEIN S. D.: Least-squares fitting of two 3-d point sets. *IEEE Transactions on pattern analysis and machine intelligence*, 5 (1987), 698–700. 2

[ARL*09] ALEXANDER O., ROGERS M., LAMBETH W., CHIANG M. J., DEBEVEC P. E.: The digital Emily project: photoreal facial modeling and animation. In *SIGGRAPH Courses* (2009), pp. 12:1–12:15. 2

[BB14] BEELER T., BRADLEY D.: Rigid stabilization of facial expressions. *SIGGRAPH* (2014). 2, 6

[BKL*16] BOGO F., KANAZAWA A., LASSNER C., GEHLER P., ROMERO J., BLACK M. J.: Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *European Conference on Computer Vision* (2016). 5

[BLB23] BOLKART T., LI T., BLACK M. J.: Instant multi-view head capture through learnable registration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2023). 1, 2

[BWP13] BOUAZIZ S., WANG Y., PAULY M.: Online modeling for real-time facial animation. *ACM Transactions on Graphics* (2013). 2, 6, 7, 8

[CCGB22] CHANDRAN P., CICCONE L., GROSS M., BRADLEY D.: Local anatomically-constrained facial performance retargeting. *ACM Transactions on Graphics* (2022). 2

[CCWL18] CAO C., CHAI M., WOODFORD O., LUO L.: Stabilized real-time face tracking via a learned dynamic rigidity prior. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia* (2018). 2, 3, 6

[CFW*22] CHEN H., FENG R., WU S., XU H., ZHOU F., LIU Z.: 2D human pose estimation: a survey. *Multimedia Systems* (2022). 6

[DRC*22] DAVYDOV A., REMIZOVA A., CONSTANTIN V., HONARI S., SALZMANN M., FUA P.: Adversarial parametric pose prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022). 5

[EF78] EKMAN P., FRIESEN W.: *Facial Action Coding System A Technique for the Measurement of Facial Movement*. 1978. 4

[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 Transactions on Graphics* (2020). 1, 2, 9

[FNH*17] FYFFE G., NAGANO K., HUYNH L., SAITO S., BUSCH J., JONES A., LI H., DEBEVEC P.: Multi-view stereo on consistent face topology. *Computer Graphics Forum* (2017). 2, 3, 6

[GGH02] GU X., GORTLER S. J., HOPPE H.: Geometry images. *ACM Transactions on Graphics* (2002). 2

[Gow75] GOWER J. C.: Face transfer with multilinear models. *Psychometrika* (1975). 2, 6

[KAB20] KOCABAS M., ATHANASIOU N., BLACK M. J.: VIBE: Video inference for human body pose and shape estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2020). 6

[LBB*17] LI T., BOLKART T., BLACK M. J., LI H., ROMERO J.: Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia* (2017). 2, 3, 7, 9

[LCC*22] LIU S., CAI Y., CHEN H., ZHOU Y., ZHAO Y.: Rapid face asset acquisition with recurrent feature alignment. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia 41*, 6 (2022), 214:1–214:17. 1, 2

[LLB*21] LI T., LIU S., BOLKART T., LIU J., LI H., ZHAO Y.: Topologically consistent multi-view face inference using volumetric sampling. In *International Conference on Computer Vision* (2021), pp. 3824–3834. 1, 2

[LLD18] LAMARRE M., LEWIS J., DANVOYE E.: Face stabilization by mode pursuit for avatar construction. In *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)* (2018). 2, 6

[LYYB13] LI H., YU J., YE Y., BREGLER C.: Realtime facial animation with on-the-fly correctives. *ACM Transactions on Graphics* (2013). 2

[PCG*19] PAVLAKOS G., CHOUTAS V., GHORBANI N., BOLKART T., OSMAN A. A. A., TZIONAS D., BLACK M. J.: Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019). 5

[PKA*09] PAYSAN P., KNOTHE R., AMBERG B., ROMDHANI S., VETTER T.: A 3D face model for pose and illumination invariant face recognition. In *Proceedings of the 6th IEEE International Conference on Advanced Video and Signal based Surveillance (AVSS) for Security, Safety and Monitoring in Smar Environments* (2009). 3, 9

[pol] Polywink blendshapes. https://polywink.com/en/9-automatic-expressions-blendshapes-on-demand.html. Accessed: 2024-01-25. 3, 4

[SEL17] SEYMOUR M., EVANS C., LIBRERI K.: Meet Mike: Epic avatars. In *SIGGRAPH* (2017). 2

[TAL*22] TIWARI G., ANTIC D., LENSSEN J. E., SARAFIANOS N., TUNG T., PONS-MOLL G.: Pose-NDF: Modeling human pose manifolds with neural distance fields. In *European Conference on Computer Vision* (2022). 5

[VBPP05] VLASIC D., BRAND M., PFISTER H., POPOVIĆ J.: Face transfer with multilinear models. *ACM Transactions on Graphics* (2005). 6

[WBGB16] WU C., BRADLEY D., GROSS M., BEELER T.: An anatomically-constrained local deformation model for monocular face capture. *ACM Transactions on Graphics* (2016). 2, 6

[WBH*21] WOOD E., BALTRUŠAITIS T., HEWITT C., DZIADZIO S., JOHNSON M., ESTELLERS V., CASHMAN T. J., SHOTTON J.: Fake it till you make it: Face analysis in the wild using synthetic data alone. In *International Conference on Computer Vision* (2021). 3

[WBLP11] WEISE T., BOUAZIZ S., LI H., PAULY M.: Realtime performance-based facial animation. *ACM Transactions on Graphics* (2011). 2, 6, 7, 8

[WSS18] WU C., SHIRATORI T., SHEIKH Y.: Deep incremental learning for efficient high-fidelity face tracking. *ACM Transactions on Graphics* (2018). 2, 6, 7

[ZBGB19] ZOSS G., BEELER T., GROSS M., BRADLEY D.: Accurate markerless jaw tracking for facial performance capture. *ACM Transactions on Graphics* (2019). 2

[ZBJ*19] ZHOU Y., BARNES C., JINGWAN L., JIMEI Y., HAO L.: On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019). 6

[ZBX*20] ZANFIR A., BAZAVAN E. G., XU H., FREEMAN B., SUKTHANKAR R., SMINCHISESCU C.: Weakly supervised 3D human pose and shape reconstruction with normalizing flows. In *European Conference on Computer Vision* (2020). 5

[ZWC*23] ZHENG C., WU W., CHEN C., YANG T., ZHU S., SHEN J., KEHTARNAVAZ N., SHAH M.: Deep learning-based human pose estimation: A survey. *ACM Computing Surveys* (2023). 6