Supplementary Material:
Learning Neural Parametric Head Models

Simon Giebenhain[1] Tobias Kirschstein[1] Markos Georgopoulos[2] Martin Rünz[2]
Lourdes Agapito[3] Matthias Nießner[1]

[1]Technical University of Munich [2]Synthesia [3]University College London

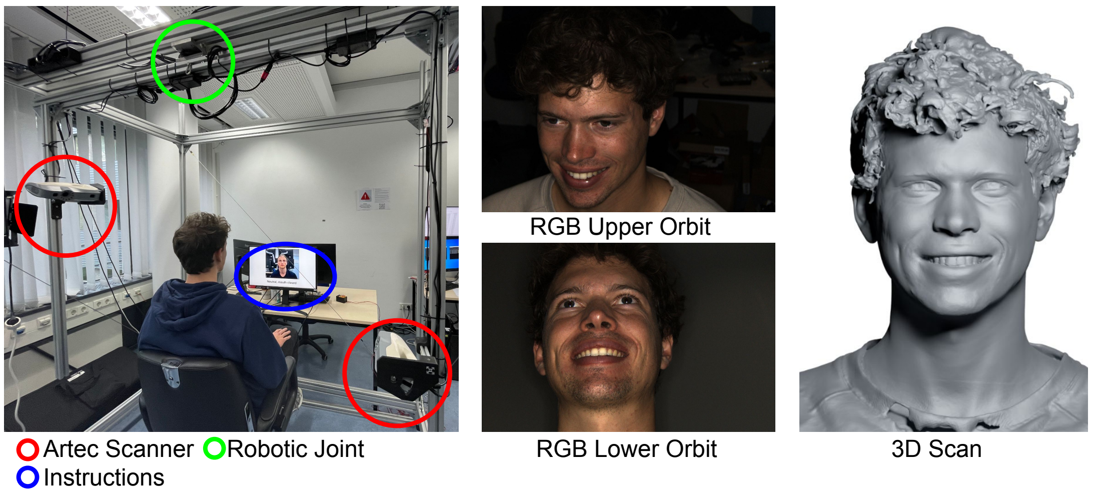

Figure 1. Our custom capture set-up. Participants are seated on a height-adjustable chair. A screen presents instructions for the 23 different expression to perform. Next to the 3D scans (right), the scanners also capture 1.3 mega pixel RGB images. This examples shows the fidelity of the laser scanners.

1. Overview

Our supplementary material is structured as follows: In section 2 we provide additional details about our capture set-up and dataset.
We provide details on the different approaches to fit point clouds for our model and all baselines in Section 3. Additionally, we provide results on reconstructing an unknown identity and expression from a single view depth map in Section 3.3 and provide a proof-of-concept tracking algorithm for a commodity depth sensor in Section 3.4.
Furthermore, we provide implementation details in Section 4. Finally, we evaluate the robustness of our model w.r.t. noise and sparsity in section 5.
For additional visual results, we refer to our supplemental video. All of our code and data will be available for research purposes[1].

[1]https://simongiebenhain.github.io/NPHM

2. Dataset

High quality data is of fundamental importance for every learning algorithm. We therefore decided to capture a high quality dataset of 3D head scans. In the following, we provide details about our custom capture set-up and the dataset.

For more samples of our dataset, we refer to Figure 12.

2.1. Capture Set-Up

Figure 1 shows our custom capture set-up, which is built inside of an aluminium cube with an edge length of two meters. We use a robotic actuator[2] to rotate an inverted U-shape around a participant's head.

We place two Artec Eva scanners opposite of each other, with complementary viewing angles on the ends of the inverted U-shape. The height and angles of the scanners are

[2]We use an acuator of the TUAKA series of Sumitomot Drive Technologies: https://us.sumitomodrive.com/en-us/actuators

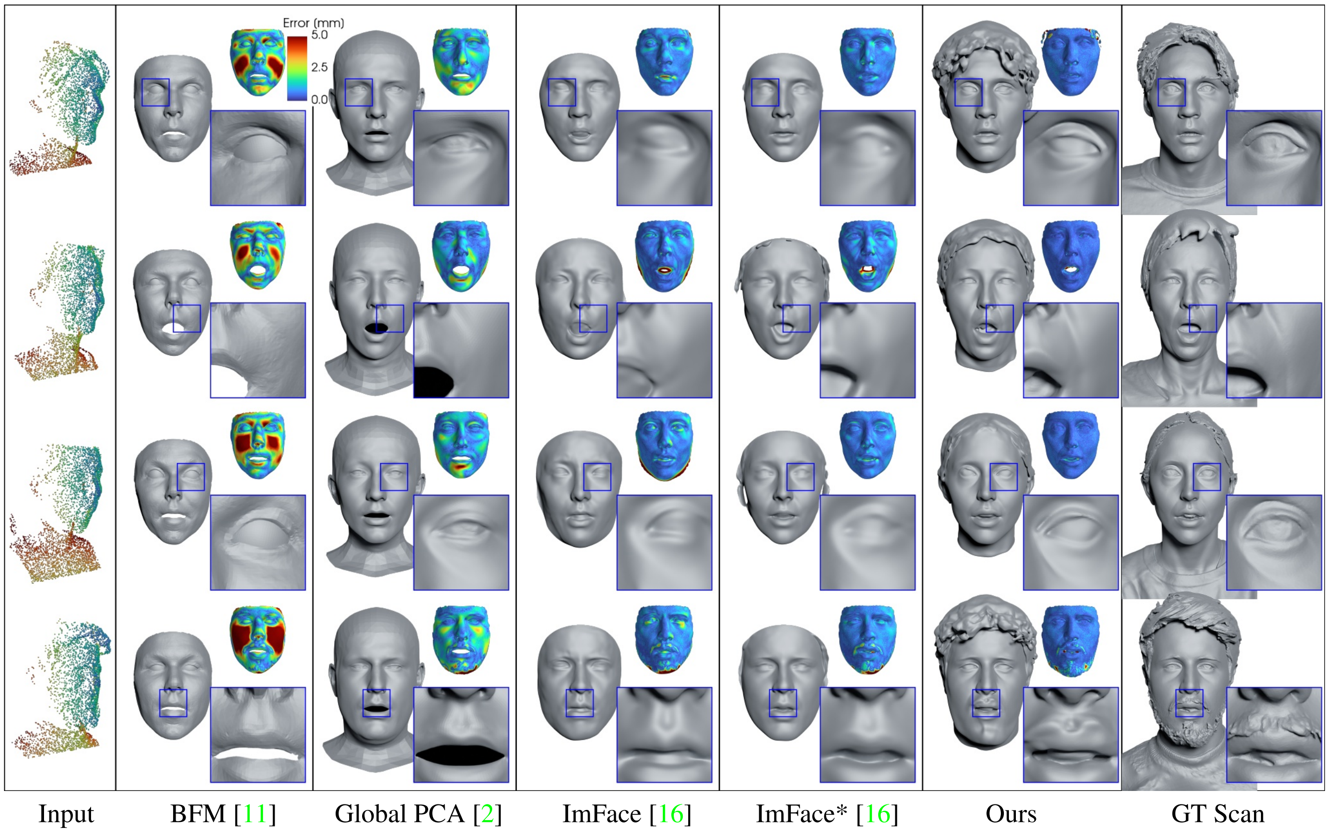

Figure 2. More identity fitting comparisons with Basel Face Model [11], a global PCA [2], ImFace [16] and an ImFace that is trained on our data (marked with *). These are the remaining baselines that are missing in Figure 5 of the main paper.

adjusted to obtain an optimal coverage, while avoiding extreme step angles which decrease scanning accuracy.

2.2. Details

During the six seconds of a 360° rotation, each scanner roughly produces 95 frames. Each frame captures range measurements obtained by analyzing a structured light projection using a stereo camera pair. Additionally, a third camera captures RGB images every fifth frame, as depicted in Figure 1. Note that we currently do not use the captured RGB input, except for facial landmark detection.

We process the individual 3D measurements of each frame using the provided software of Artec. First, we align the individual frames of the upper and lower scanner using a global registration algorithm. The individual frames are then fused into a single 3D mesh. Subsequently, we use a hole-filling algorithm and remove disconnected parts, for simplicity.

2.3. Expressions

As mentioned in the main paper, our 23 facial expressions are adapted from FaceWarehouse [3]. We illustrate the different expressions that we capture in figure 13. As mentioned before, the neutral, open-mouthed expression is of special importance since it serves as our canonical expression.

2.4. GDPR

Due to the sensitivity of the captured data, all participants in our dataset signed an agreement form compliant with GDPR. Please note that GDPR compliance includes the right for every participant to request the timely deletion of their data. We will enforce these rights in the distribution of our dataset.

3. Fitting

The following provides a description of how we used the learned prior of our model and all baselines to fit the single view depth maps.
Additionally, we show qualitative results of the remaining baselines for the identity and expression fitting experiment in Figure 2 and 3, respectively.
Furthermore, we present quantitative and qualitative results for joint identity and expression reconstruction based on a single depth map in Section 3.3.

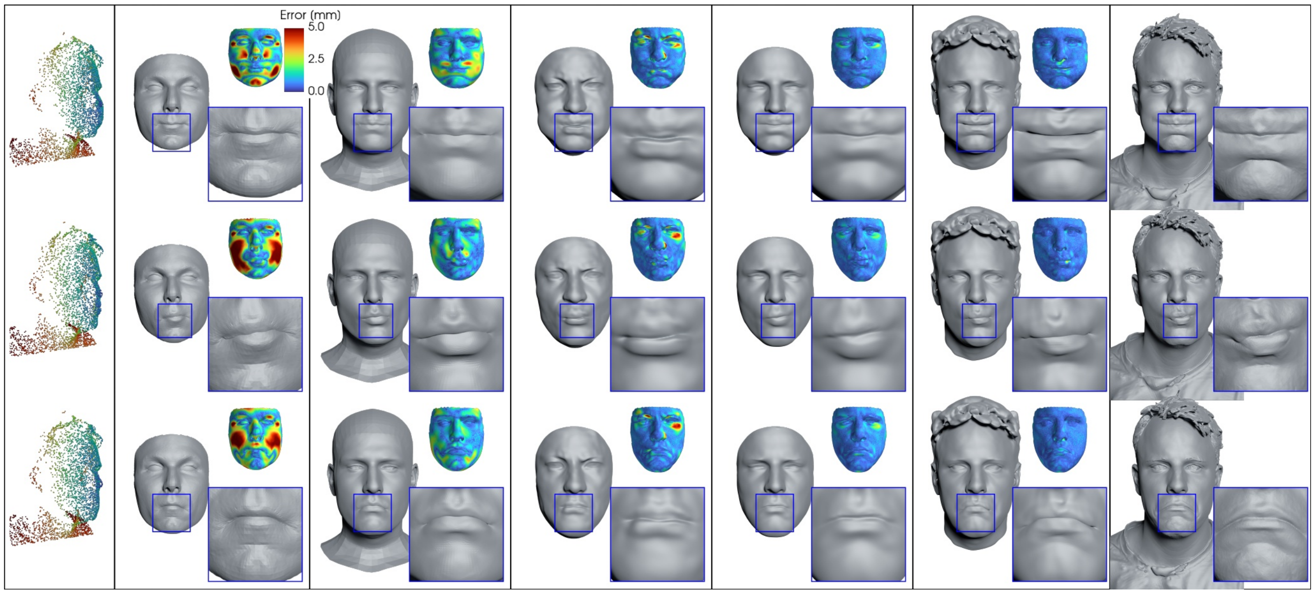

Figure 3. More expression fitting comparisons with Basel Face Model [11], a global PCA [2], ImFace [16] and an ImFace that is trained on our data (marked with *). These are the remaining baselines that are missing in Figure 6 of the main paper.

Pretrained Models Due to the difference in neutral expressions between our data and the pretrained baselines, *i.e.* BFM [11], FLAME [8] and ImFace [16], we cannot fit the identity in an isolated fashion, since that would be unfair. To mitigate this, we fit all these models jointly to all expressions of a person. Additionally, we provide facial landmarks and optimize for equation (1) of the main paper. The results are then used to evaluate both the identity and expression fitting experiments.

3.1. Identity Fitting

Given a single view depth map $X_p \subset \mathbb{R}^3$ of an unknown person in neutral facial expression, we optimize for an identity code $\mathbf{z}^{\text{id}}$, as well as an expression code $\mathbf{z}^{\text{ex}}$. We include the latter in the optimization, in order to account for minor deviations from a perfect canonical facial expression.

PCA-Based Models For this purpose, we again optimize for equation (1) but only provide the neutral depth map and corresponding landmarks.

ImFace Since ImFace utilizes backward deformations, the observed points X_p in posed space can be backward-warped into canonical space, where $\mathcal{F}_{\text{id}}$ can act on them. Therefore, the fitting task can be formulated naturally to minimize:

$$\sum_{x_p \in X} |\mathcal{F}_{\text{id}}\left(\mathcal{F}_{\text{ex}}^{\leftarrow}(x_p, \mathbf{z}^{\text{ex}}), \mathbf{z}^{\text{id}}\right)| + \lambda \mathcal{R}_1(\mathbf{z}^{\text{id}}, \mathbf{z}^{\text{ex}}), \quad (1)$$

where $\mathcal{R}_1$ includes the same regularization terms used in ImFace [16]. We write $\mathcal{F}_{\text{ex}}^{\leftarrow}$ to denote the backward deformation field of ImFace and $\mathcal{F}_{\text{id}}$ for its SDF in canonical space. (Please note that for the sake of this discussion, we ignore the fact that their $\mathcal{F}_{\text{id}}$ is composed of another deformation field and a template SDF.) We use their official code and hyperparameters.

NPM and NPHM For forward deformation models, formulating a loss to jointly optimize for $\mathbf{z}^{\text{id}}$ and $\mathbf{z}^{\text{ex}}$ is less straght forward. The authors of NPM [9] proposed a slightly convoluted formulation using a TSDF grid estimated from the depth observations. Instead, we resort to the iterative root finding scheme proposed in SNARF [4], that inverts the forward deformation. Given a point $x_p \in X$ in posed space, its corresponding points in canonical space is its preimage under $\mathcal{F}_{\text{ex}}^{\rightarrow}$. The authors of [4] propose to solve for

$$x_c = \arg\min_x |x_p - \mathcal{F}_{\text{ex}}^{\rightarrow}(x, \mathbf{z}^{\text{ex}})| \quad (2)$$

iteratively to establish a corresponding point x_c in canonical space. In order to avoid backpropagation through this iterative procedure, they utilize analytical gradients instead, which can be derived as described in [1]. Using these cor-

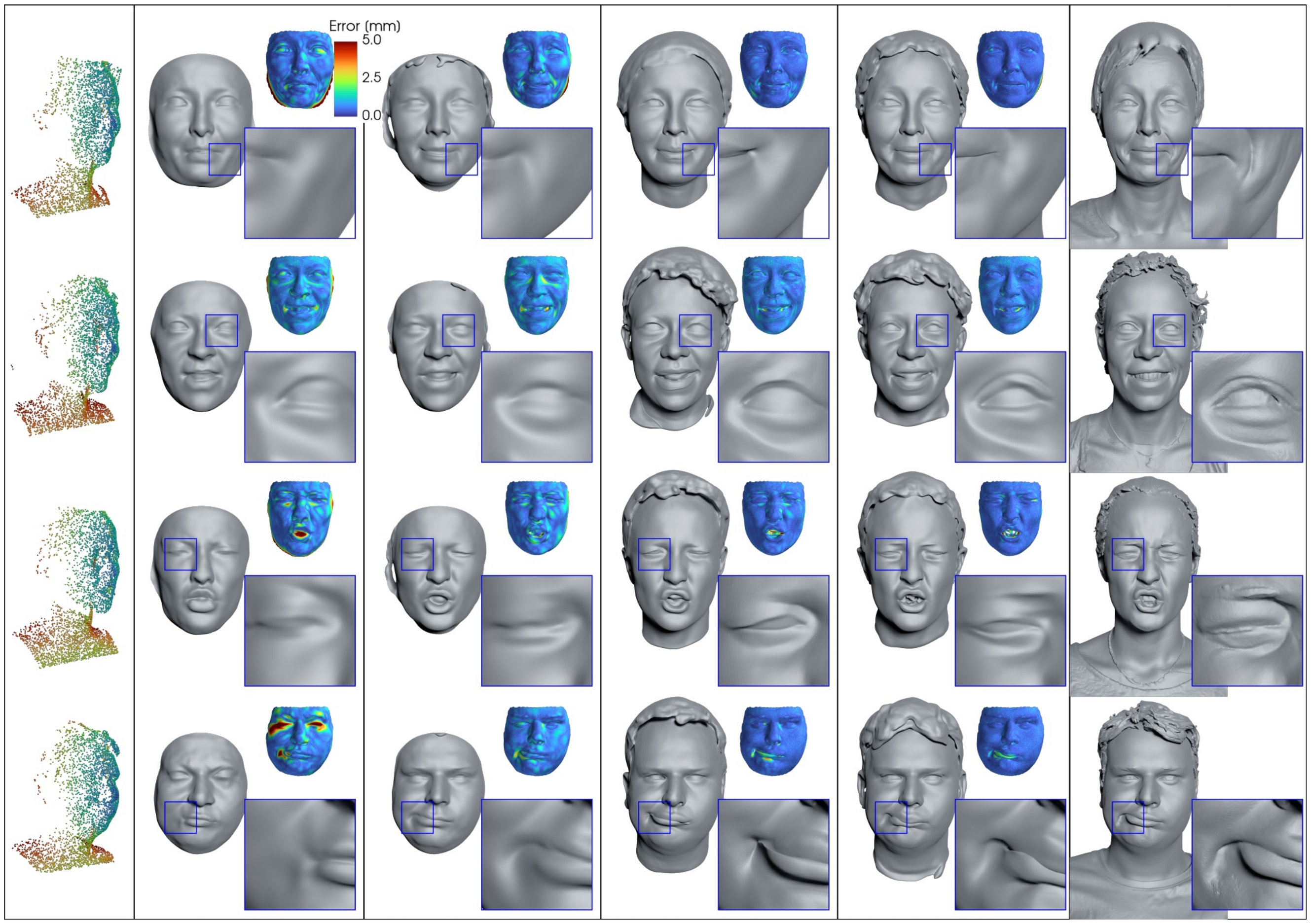

Figure 4. Results when fitting both identity and expression codes jointly on a single depth map. We compare against ImFace [16], an ImFace that is trained on our data (marked with *), and NPM [9].

respondences we can then resort to the loss in equation 1

$$\sum_{x_p \in X} \left| \mathcal{F}_{\text{id}}\left(x_c, \mathbf{z}^{\text{id}}\right) \right| + \\ \lambda^{\text{fit}}_{\text{glob}} \|\mathbf{z}^{\text{id}}_{\text{glob}}\|_2^2 + \lambda^{\text{fit}}_{\text{ex}} \|\mathbf{z}^{\text{ex}}\|_2^2 + \lambda^{\text{fit}}_{\text{loc}} \sum_{k=1}^{K} \|\mathbf{z}^{\text{id}}_k\|_2^2 + \lambda^{\text{fit}}_{\text{sy}} \mathcal{L}_{\text{sy}}, \tag{3}$$

where x_c replaces the result of the backward deformation. The second line regularizes all latent codes, as well as the difference between symmetric facial regions. For NPM we simply omit the local latent code and symmetry regularization terms. Furthermore, we did not observe topological issues and therefore stick with a single initialization $x_{\text{init}} = x_p$ for the iterative root finding.

For our ablation in Section 5.3 of the main paper, as well as, Section 5, we isolate the expression component completely and replace x_c with x_p, assuming that the observed pose is perfectly neutral.

3.2. Expression Fitting

In our expression fitting experiment, we investigate the models' performance to obtain $\mathbf{z}^{\text{id}}$ and $\{\mathbf{z}^{\text{ex}}_s\}_{s=1}^{S}$, given S observations $\{X_p^s\}_{s=1}^{S}$ in posed space, where S is the total number of scans per person.

For our PCA-based baselines, as well as both variants of ImFace, we jointly optimize for the parameters of interest using the same losses as in the previous section.

For the forward deformation models, we find the $\mathbf{z}^{\text{id}}$ from the identity fitting already provides a good estimate. For simplicity, we then keep $\mathbf{z}^{\text{id}}$ fixed and only optimize for $\{\mathbf{z}^{\text{ex}}_s\}_{s=1}^{S}$ using equation 3.

3.3. Single-Expression Fitting

The expression fitting task in the main paper attempts to evaluate the expressiveness of each model's expression space by constraining the identity codes to remain the same over all scans of one person.

we present another experiment that aims to reconstruct $\mathbf{z}^{\text{id}}$ and $\mathbf{z}^{\text{ex}}$ jointly given only one depth map of an unknown person in arbitrary expression.

Table 1 reports quantitative numbers that further support the effectiveness of the proposed model.

Method	L_1-Chamfer ↓	N. C. ↑	F-Score@1.5 ↑
ImFace [16]	0.375e−2	0.966	0.825
ImFace* [16]	0.320e−2	0.972	0.879
NPM [9]	0.243e−2	0.969	0.928
Ours	**0.207**e−2	**0.974**	**0.947**

* trained on our data

Table 1. Fitting performance from a single depth map of unknown identity and unknown expression.

Hyperparameters for NPM and NPHM We optimize Equation 3 using the Adam optimizer for 700 iterations. The optimization procedure starts with a learning rate of 0.01 and is decayed by a factor of 10 after epochs 200, 350, and 500. For our model we use $\lambda^{\text{fit}}_{\text{glob}} = 0.05$, $\lambda^{\text{fit}}_{\text{loc}} = 0.05$ and $\lambda^{\text{fit}}_{\text{ex}} = 0.003$ to regularize the global and local identity and expression components, respectively. Additionally, we encourage symmetry $\lambda^{\text{fit}}_{\text{sy}} = 1.0$ for the first half of iterations and then set $\lambda^{\text{fit}}_{\text{sy}} = 0.0$. Additionally, we divide $\lambda^{\text{fit}}_{\text{loc}}$ and $\lambda^{\text{fit}}_{\text{glob}}$ by a factor of 5 at epochs 200 and 500, such that the model first learns the coarse facial expression before focusing on the details of the identity.
For NPM we use the exact same hyperparameters as for our model. However, the local regularization and symmetry prior have no effect.

3.4. Real-World Tracking

Additionally, we evaluate our model in a real-world face tracking scenario. For this purpose, we fit our model against a depth video captured with a Kinect Azure, a commodity depth sensor. Figure 5 shows our results of a single frame and a comparison to the FLAME model. For the full tracking results, we refer to our supplemental video.

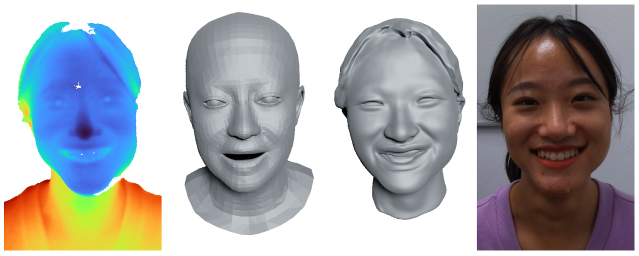

Figure 5. Real-world tracking. For a single frame, we show from left to right: the depth map obtained from a commodity depth sensor, FLAME, and our reconstructions, and an image as reference.

For this proof of concept, we simply optimize for $\mathbf{z}^{\text{id}}$ using a single frame and subsequently optimize for head pose and expression parameters for each frame. Additionally, we include a total variation prior along the temporal axis over estimated head pose and expression parameters. More specifically, we add

$$\mathcal{L}_{TV}(\phi) = \sum_{t=1}^{T} \|\phi(t+1) - \phi(t)\| \tag{4}$$

to optimization problem, where $\phi(t)$ denotes any of the time dependent optimization parameters, *i.e.* expression and pose.

To coarsely align the coordinates system of the back-projected depth map into our canonical coordinate system, we calculate the similarity transform using [14] from detected landmarks to the landmarks of the average FLAME face (note that our model shares the coordinate system of FLAME).

To further guide the optimization, we also include landmarks at the mouth and eye corners, as well as on the top and bottom of the lips, which we denote as $\mathbf{a}_t \in \mathbb{R}^{8\times3}$ for each time step.

First, we utilize the detected landmarks for the initial identity fitting on a chosen frame t_{can}. Here the landmarks serve as additional supervision for $\mathbf{z}^{\text{id}}_{\text{glob}}$, by including the term

$$\|\text{MLP}_{\text{pos}}(\mathbf{z}^{\text{id}}_{\text{glob}}) - \mathbf{a}_{t_{\text{can}}}\|_1. \tag{5}$$

In this stage, we additionally estimate normals using a Sobel filter and use them as additional supervision signal, as in Equation 10.

During expression fitting, we incorporated the eight facial landmarks as direct supervision for the forward deformation network:

$$\sum_{t=1}^{T} \|\mathcal{F}_{\text{ex}}(\text{MLP}_{\text{pos}}(\mathbf{z}^{\text{id}}_{\text{glob}}), \mathbf{z}^{\text{ex}}_t, \mathbf{z}^{\text{id}}_{\text{glob}}) - \mathbf{a}_t\|_1. \tag{6}$$

4. Implementation Details

We implement our approach – including registration, training, and inference – in PyTorch and, unless otherwise mentioned, run all heavy computations on the GPU, for which we use an Nvidia GTX 3090.

4.1. Non-Rigid Registration

In Equations (1) and (2) of the main paper, we use the point-to-plane distance $d(v, \mathcal{S})$ from a point $v \in \mathbb{R}^3$ to a surface $\mathcal{S} \subset \mathbb{R}^3$. To make our energy terms more robust, we filter this distance based on a distance δ_d and normal threshold δ_n, such that

$$d^*(v,\mathcal{S}) = \begin{cases} 0, & \text{if } d(v,\mathcal{S}) > \delta_d, \\ 0, & \text{if } \langle n(v), n(s)\rangle > \delta_n, \\ d(v,\mathcal{S}), & \text{otherwise,} \end{cases} \tag{7}$$

where

$$d(v,\mathcal{S}) = \min_{s\in\mathcal{S}} |\langle v-s, n(s)\rangle| \tag{8}$$

is the unfiltered point to plane distance and $n(v)$ and $n(s)$ denote the vertex normals of v in the template mesh and the normals of its nearest neighbor in the target $\mathcal{S}$, respectively.

FLAME Fitting We regularize our optimization in FLAME parameter space using

$$\mathcal{R}(\Phi_j) = \lambda_{\text{id}}\frac{\|\mathbf{z}^{\text{id}}\|_2^2}{20} + \lambda_{\text{ex}}\|\mathbf{z}^{\text{ex}_j}\|_2^2 + \lambda_{\text{jaw}}\|\theta_j\|_2^2 + \lambda_{\text{rigid}}(\|R_j\|_2^2 + \|t_j\|_2^2). \tag{9}$$

We use $\lambda_{\text{id}} = 1/5000$, $\lambda_{\text{ex}} = 1/3000$ to regularize the identity and expression parameters respectively. For the jaw angle and the rigid parameters, we regularize with $\lambda_{\text{jaw}} = 1/10$ and $\lambda_{\text{rigid}} = 1/10$. Since the point-to-plane distance initially gives an unreliable signal, despite our filtering we down-weight the point-to-plane distance with $\lambda_d = 1/15$ for the first 300 iterations. For all remaining iterations of the 2000 iterations, we set $\lambda_d = 1$. We solve Equation (1) using the Adam [7] optimizer with a learning rate of $4e^{-3}$, which is decayed by a factor of 5 for the final 500 iterations.

Finetuning We exponentially decay the weight λ_{ARAP} of the ARAP [13] term with a factor of 0.99. We start with $\lambda_{\text{ARAP}} = 10.0$, but do not decay below $\lambda_{\text{ARAP}} = 0.1$. On average our implementation converges after 400-500 iterations of the L-BFGS optimizer and takes roughly 4 minutes on a single Nvidia 1080 GPU.

Since both the FLAME fitting and finetuning require a large number of nearest neighbor queries between vertices of the optimized mesh and the target mesh, we utilize FAISS [6], which provides efficient, GPU-optimized search indices for approximate similarity search.

4.2. Data Preparation and Training

Identity Training To train $\mathcal{F}_{\text{id}}$, we use the loss

$$\begin{aligned}\mathcal{L}_{\text{IGR}} = &\sum_{x\in\delta X} \lambda_s|\mathcal{F}_{\text{id}}(x)| + \lambda_s\left(1-\langle\nabla\mathcal{F}_{\text{id}}(x), n(x)\rangle\right) \\ &+ \sum_{x\in X\cup\delta X} \lambda_{\text{eik}}(\|\nabla\mathcal{F}_{\text{id}(x)}\|_2-1) \\ &+ \sum_{x\in X}\lambda_0\exp(-\alpha|\mathcal{F}_{\text{id}}(x)|)\end{aligned} \tag{10}$$

introduced in [5] and [12], where we omit the conditioning of $\mathcal{F}_{\text{id}}$ for simplicity. Here δX denotes samples on the surface and X denotes samples in space. We choose $\lambda_s = 2$, $\lambda_n = 0.3$, $\lambda_{\text{eik}} = 0.1$ and $\lambda_0 = 0.01$. For the additional hyperparameters mentioned in Equation (11)we set $\lambda_{\text{reg}}^{\text{id}} = 0.005$, $\lambda_a = 7.5$ and $\lambda_{\text{sy}} = 0.005$.

Furthermore, we train for $15,000$ epochs with a learning rate of 0.0005 and 0.001 for the network parameters and latent codes, respectively. Both learning rates are decayed by a factor of 0.5 every $3,000$ epochs. We use a batch size of 16 and $|\delta X| = 500$ points sampled on the surface. Samples X are obtained by adding Gaussian noise with $\sigma = 0.01$ to surface points and adding some points sampled uniformly in a bounding box. Additionally, we use gradient clipping with a cut-off value of 0.1 and weight decay with a factor of 0.01.

Since this loss only requires samples on the surface directly, we precompute $2,000,000$ points sampled uniformly on the surface of the 3D scans, after removing the lower part of the scan, which we determine using a plane spanned by three vertices on the neck of our registered template mesh. Since our focus lies on the front part of the face, 80% of these points are sampled on the front and 20% on the back and neck. The frontal area is determined by a region on our registered meshes, which covers the face, ears, and forehead. We additionally sample surface normals.

Training the identity network takes about 12 hours until convergence on a single GPU.

Expression Training For the training of $\mathcal{F}_{\text{ex}}$, we follow NPMs [9] and precompute samples of the deformation field, which can be used for direct supervision of $\mathcal{F}_{\text{ex}}$.

More specifically, let $\mathcal{M}$ and $\mathcal{M}'$ be a neutral and expression scan. For a point $x \in \mathcal{M}$, we determine the corresponding point $x' \in \mathcal{M}'$ using barycentric coordinates and construct samples of the deformation field $\delta(x) = x' - x$. While strictly speaking the deformation is only defined for points on the surface, we compute field values close to the surface by offsetting along the normal direction, *i.e.* $\delta(x + \alpha n(x)) = x' + \alpha n(x') - (x + \alpha n(x))$, where we sample $\alpha \sim \mathcal{N}(0, \tau_i\mathbb{I}_3)$ twice with standard deviations $\tau_1 = 0.02$ and $\tau_2 = 0.004$. Overall, we sample $2,000,000$ points per expression.

For the expression training we use $\lambda_{\text{reg}}^{\text{ex}} = 5e^{-5}$ and a learning rate of $5e^{-4}$ and $1e^{-3}$ for the network and latent codes, respectively. We train for $2,000$ epochs with a learning rate decay of 0.5 every 600 epochs, gradient clipping at 0.025 and weight decay strength $5e^{-4}$. We use 1000 samples to compute $\mathcal{L}_{\text{ex}}$ and a batch size of 32.

Training the expression network until convergence takes about 8 hours on a single GPU.

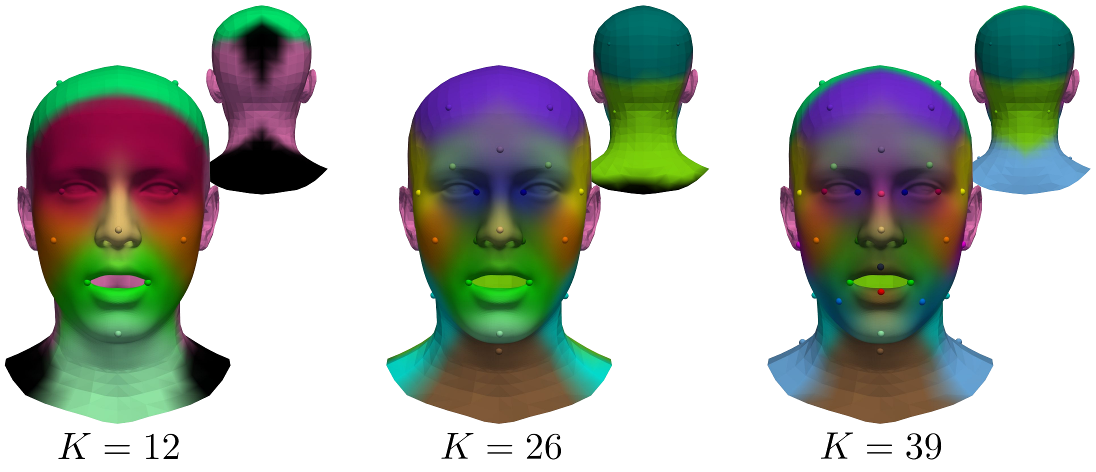

Figure 6. Anchor Layout: Each anchor is assigned a unique color, except for symmetric pairs which share colors. We calculate vertex colors by blending in the same fashion, as for the ensemble of local MLPs. Consequently, the colors show the influence that each local MLP has on its surrounding. Black denotes the color of f_0. Anchor points were chosen as vertices of the average over all registrations.

4.3. Architectural Details

4.3.1 NPMs

In the main paper, we compare our proposed method against our implementation of NPMs [9]. Instead of the proposed ensemble of local MLPs, NPMs use the original architecture of DeepSDF [10] with 8 layers, a hidden dimensionality of 1024, and $\mathbf{Z}_{\text{id}} = 512$ dimensions for the latent vector for $\mathcal{F}_{\text{id}}$.

The expression latent dimension is $d_{\text{ex}} = 200$ and the MLP has 6 hidden layers with 512 hidden units. We use identical settings for NPHM.

4.3.2 NPHMs

Our default choice for the number of anchor points is $K = 39$, of which $K_{\text{symm}} = 16$ are symmetric. This leads to 7 anchor points lying directly on the symmetry axis, and hence parameters of $16 + 7 = 23$ local DeepSDFs have to be optimized. Figure 6 depicts the arrangement of the anchor points.

The identity latent space is composed of the shared global part $\mathbf{z}^{\text{id}}_{\text{glob}} \in \mathbb{R}^{d_{\text{glob}}}$ with $d_{\text{glob}} = 64$ and local latent vectors $\mathbf{z}^{\text{id}}_{k} \in \mathbb{R}^{d_{\text{loc}}}$ with $d_{\text{loc}} = 32$. Our local MLPs have 4 hidden layers with 200 hidden units each and follow the DeepSDF [10] architecture. Note that the total number of latent identity dimensions $d_{\text{id}} = (K+1) * d_{\text{loc}} + d_{\text{glob}} = 1344$.

Furthermore, we use $\sigma = 0.1$ and $c = e^{-0.2/\sigma^2}$ to blend the ensemble of local MLPs. Figure 6 illustrates the resulting influence that the individual local MLPs have on the final prediction.

Anchor Points In the main paper, we ablated the number of face anchor points. Figure 6 shows a comparison of the different anchor layouts that we ablated. For a lower number of anchors, we increase d_{loc} such that d_{id} is roughly preserved.

For the ablation of our symmetry prior, we keep the exact same anchor layout; however, do not share network weights for symmetric anchors and do not mirror coordinates.

4.4. Metrics

Since we quantitatively compare models that represent vastly different regions of the human head, we restrict the calculations of our metrics to the face region. This also aligns with the fact, that each model only observes a single, frontal depth map, i.e. other parts of the head can only be estimated roughly.

To this end, we determine the facial area by all points which are closer than 1cm to a region defined on our registered template mesh. Within this region, we sample 1,000,000 points with their corresponding normals on the ground truth as well as on each reconstruction. Using these sampled points and normals, we compute all of our metrics.

Please note, that this evaluation does not account for the fact that reconstructions of closed-mouth expressions might contain the inner part of the mouth. The inner part of the mouth is not visible by the 3D Scanners and hence is missing in the ground truth. This especially is a disadvantage for forward deformation models, since they reconstruct large parts of the inner mouth region. To account for this one might have to exclude sampled points in the reconstructions that are not visible, *e.g.* by rendering depth images from multiple views and backprojecting them to 3D.

5. Additional Ablations

The experiments in the main paper were restricted to single view depth maps with 5000 points. Here, we present a thorough evaluation with respect to the number of input points and with respect to artificial Gaussian noise.

Number of Points: Figure 7 shows how the number of observed points effect the reconstructions quantitatively. We evaluate on 250, 500, 1000, 2500, 5000, and 10000 points, respectively. Figure 10 illustrates the effect qualitatively.

Noise: Similarly, we ablate against additive Gaussian noise with standard deviations of 0.0mm, 0.3mm, 0.75mm and 1.5mm. Quantitative and qualitative results are presented in Figures 8 and 9, respectively.

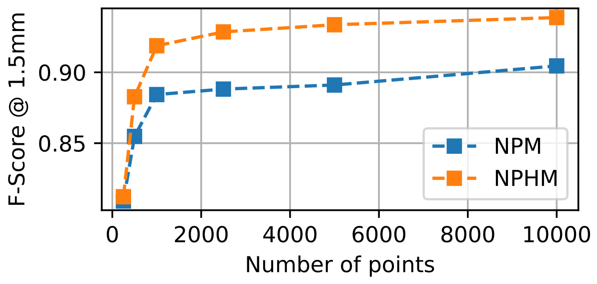

Figure 7. Ablation with respect to the number of points in the input point cloud.

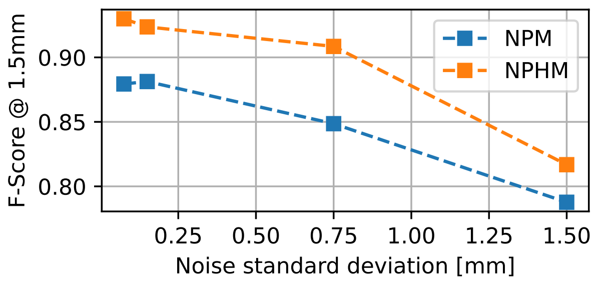

Figure 8. Robustness of our method to noise in the input point cloud.

5.1. Deformation Consistency

Furthermore, we illustrate the behaviour of our expression network $\mathcal{F}_{ex}$ in figure 11, by assigning a distinctive UV-map as colors to each vertex. To be more specific, we assign vertex colors by projecting a UV-map parallel to the ”depth-dimension”. We then fix vertex colors and deform the mesh using $\mathcal{F}_{ex}$. The results show that semantic consistency is preserved well, which is a direct consequence of our training strategy. i3DMMs [15] and ImFace [16] exhibit fewer consistent correspondences since they model backward deformations and do not rely on direct supervision from deformations.

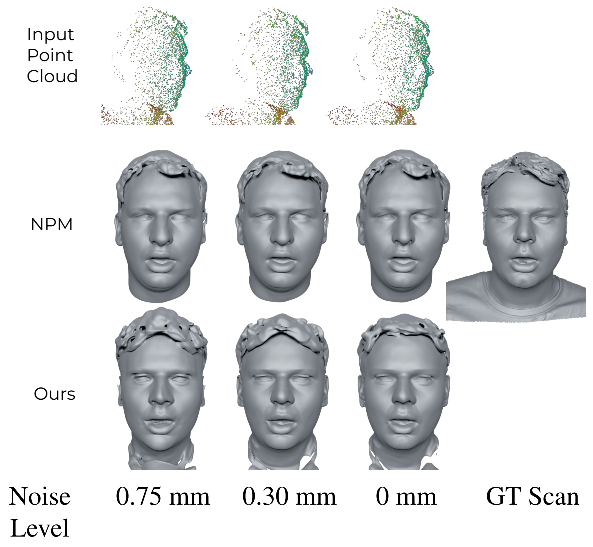

Figure 9. Qualitative comparison of NPMs [9] and our method with respect to noise in the input point cloud. We perturb the points by applying random Gaussian noise with different standard deviations.

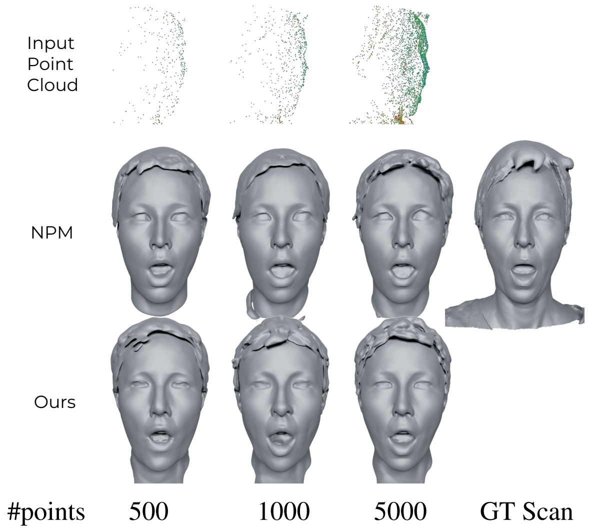

Figure 10. Qualitative comparison of NPMs [9] and our method with respect to the number of points in the input point cloud.

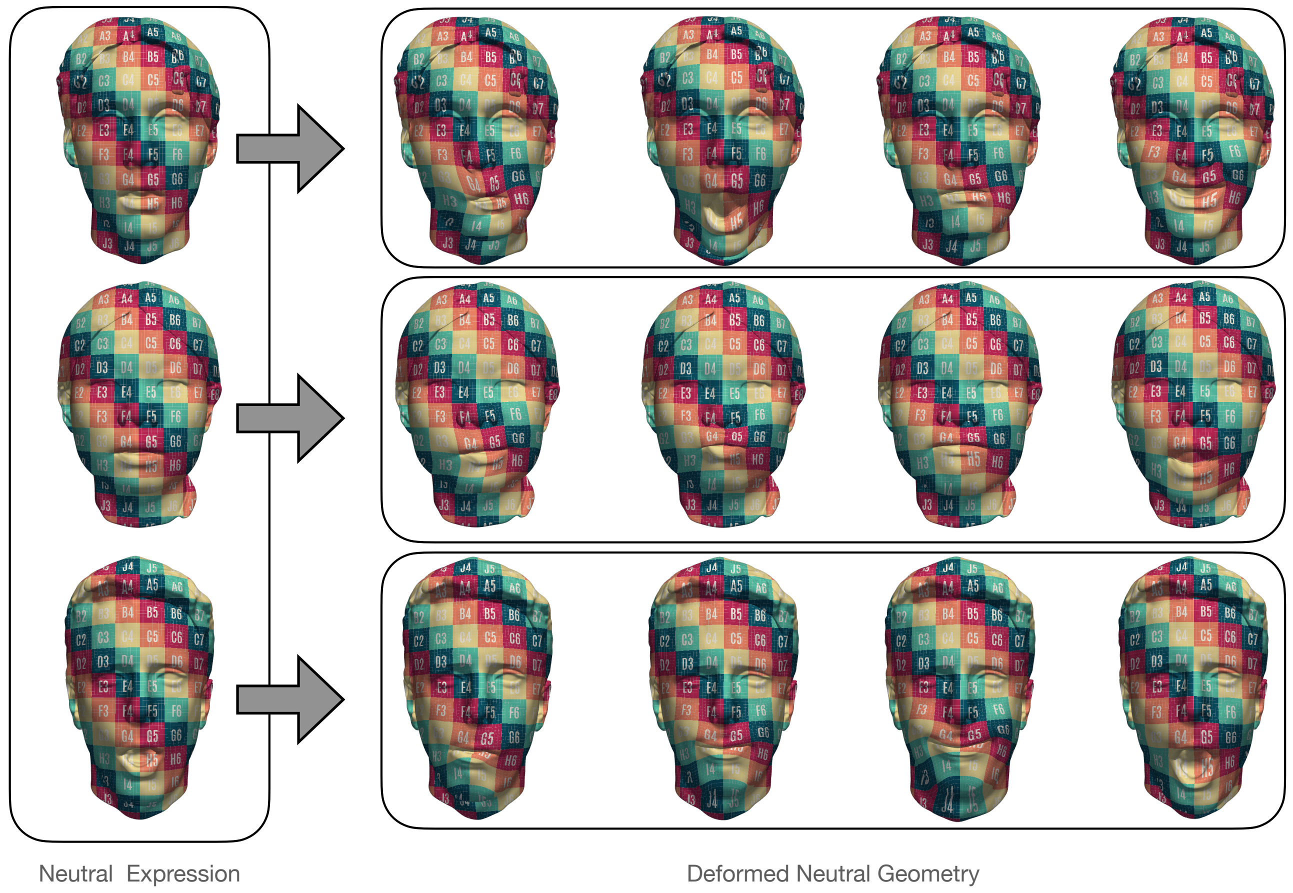

Figure 11. Deformation Consistency: We show surface correspondences between neutral and posed meshes from our test set.

References

[1] Matan Atzmon, Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron, and Yaron Lipman. Controlling neural level sets. In *Advances in Neural Information Processing Systems*, pages 2032–2041, 2019. 3

[2] Volker Blanz and Thomas Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 187–194, 1999. 2, 3

[3] Chen Cao, Yanlin Weng, Shun Zhou, Yiying Tong, and Kun Zhou. Facewarehouse: A 3d facial expression database for visual computing. 20(3):413–425, mar 2014. 2

[4] Xu Chen, Yufeng Zheng, Michael J Black, Otmar Hilliges, and Andreas Geiger. Snarf: Differentiable forward skinning for animating non-rigid neural implicit shapes. In *International Conference on Computer Vision (ICCV)*, 2021. 3

[5] Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman. Implicit geometric regularization for learning shapes. *arXiv preprint arXiv:2002.10099*, 2020. 6

[6] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. 6

[7] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. cite arxiv:1412.6980Comment: Published as a conference paper at the 3rd International Conference for Learning Representations, San Diego, 2015. 6

[8] Tianye Li, Timo Bolkart, Michael. J. Black, Hao Li, and Javier Romero. Learning a model of facial shape and expression from 4D scans. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):194:1–194:17, 2017. 3, 5

[9] Pablo Palafox, Aljaž Božič, Justus Thies, Matthias Nießner, and Angela Dai. Npms: Neural parametric models for 3d deformable shapes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12695–12705, October 2021. 3, 4, 5, 6, 7, 8

[10] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 7

[11] Pascal Paysan, Reinhard Knothe, Brian Amberg, Sami Romdhani, and Thomas Vetter. A 3d face model for pose and illumination invariant face recognition. In *2009 sixth IEEE international conference on advanced video and sig-*

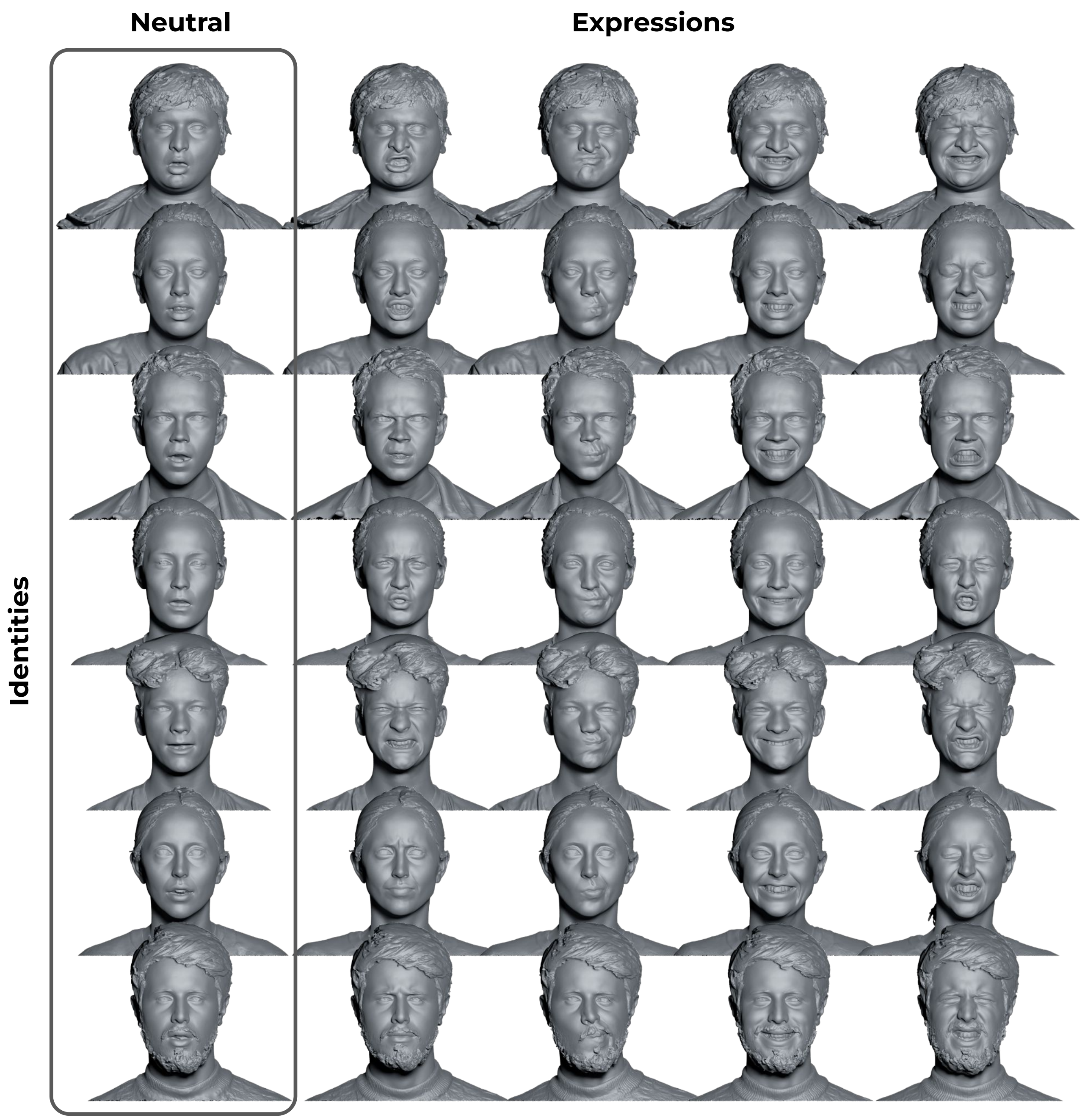

Figure 12. More 3D head scans from our newly-captured dataset.

nal based surveillance, pages 296–301. Ieee, 2009. 2, 3

[12] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 6

[13] Olga Sorkine and Marc Alexa. As-Rigid-As-Possible Surface Modeling. In Alexander Belyaev and Michael Garland, editors, *Geometry Processing*. The Eurographics Association, 2007. 6

[14] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 13(04):376–380, 1991. 5

[15] Tarun Yenamandra, Ayush Tewari, Florian Bernard, Hans-Peter Seidel, Mohamed Elgharib, Daniel Cremers, and Christian Theobalt. i3dmm: Deep implicit 3d morphable model of human heads. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*,

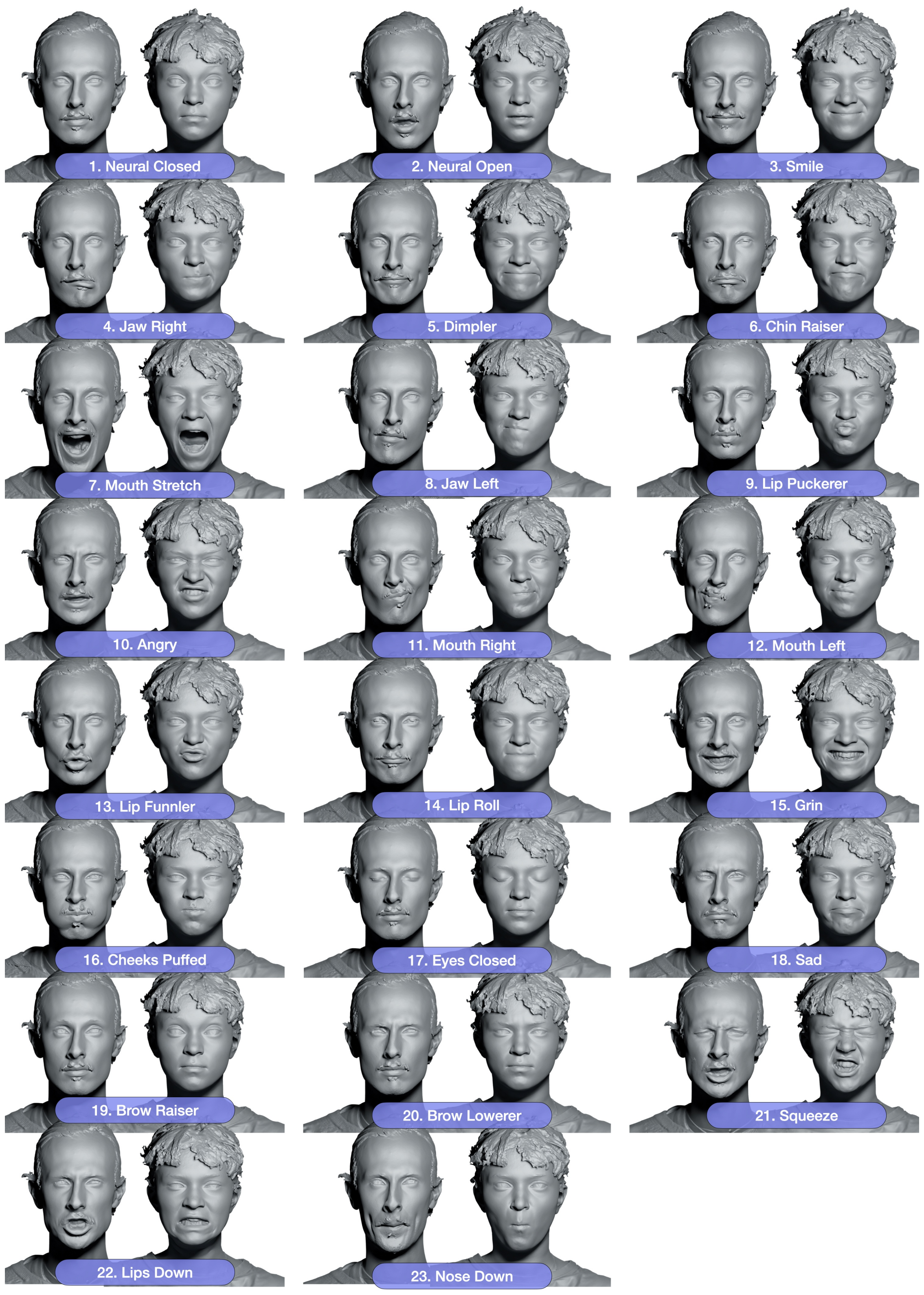

Figure 13. We capture 23 expressions for each participant. Here we show two subjects performing all expressions.

pages 12803–12813, 2021. 8

[16] Mingwu Zheng, Hongyu Yang, Di Huang, and Liming Chen. Imface: A nonlinear 3d morphable face model with implicit neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3, 4, 5, 8