MonoNPHM: Dynamic Head Reconstruction from Monocular Videos

Supplementary Material

1. Overview

This supplementary document provides additional implementation details on our network architecture (Sec. 2.1), training (Sec. 2.3) and tracking strategy (Sec. 2.4).

Additionally, we present more qualitative results (Sec. 3) and discuss our ablation experiments (Sec. 3.2).

We kindly suggest the reviewers watch our supplementary video, for a temporally complete visualization of the tracked sequences.

2. Implementation Details

In Sec. 2.1 we provide details about the individual network components of MonoNPHM. Sec. 2.2 describes how we implement a memory efficient variant of the MLP ensemble proposed in [3].

2.1. Network Architectures

Some of mentioned details in this subsection require detailed knowledge about NPHM [3].

Expression Network To represent our backward deformation field $\mathcal{F}_{\text{exp}}$ we use a 6-layer MLP with a width of 400. The expression codes $\mathbf{z}_{\text{exp}}$ are 100 dimensional. The dependence on $\mathbf{z}_{\text{geo}}$ is bottlenecked by a linear projection to 16 dimensions, as proposed in [3].

Geometry Network Our local geometry MLPs f^k_{geo} have 4 layers and a width of 200. Out of the 65 anchors, 30 are symmetric, meaning that the ensemble consists of $64-30 = 34$ MLPs. Note, however, that the spatial input of f^k_{geo} is augmented with the predicted hyper-dimensions.

Appearance Network Our appearance MLPs f^k_{app} follow the same structure as f^k_{geo}, but receive extracted geometry features $\mathbf{h}_{\text{geo}}(x_c)$ as input. $\mathbf{h}_{\text{geo}}$ is a two-layer MLP (widths 100 and 16), that maps the hidden features of the last layers of f^k_{geo} to 16 dimensions.

Anchor Prediction Compared to the anchor layout used in NPHM [3], we increase the number of anchors from 39 to 65, and rearrange them, such that the anchors coincide with the most important facial landmarks for tracking. Fig. 1 shows our anchor layout. The anchor prediction MLP $\mathcal{A}$ consists of 3 linear layers and has a hidden dimension of 64.

2.2. Efficient Implementation

To account for the computational burden of the increased number of anchors and added appearance MLPs, we prune the computations of the local MLP ensemble.

kNN Pruning NPHM executes every MLP f^k_{geo} for each query point x_c. Instead, we use Pytorch3D [14] to compute the 8 nearest neighbors $\mathcal{N}_{x_c}$ for each query. Then, we conceptualize the execution of local MLPs as a graph convolution, implement usingPytorchGeometric [2]. The graph convolution is restricted to $\mathcal{N}_{x_c}$ (see equation 2 in the main document). In practice, this decreases the number of MLP executions for each query from 65 to 8 (the number of nearest neighbors). Hence, GPU memory demand is roughly reduced 8-fold

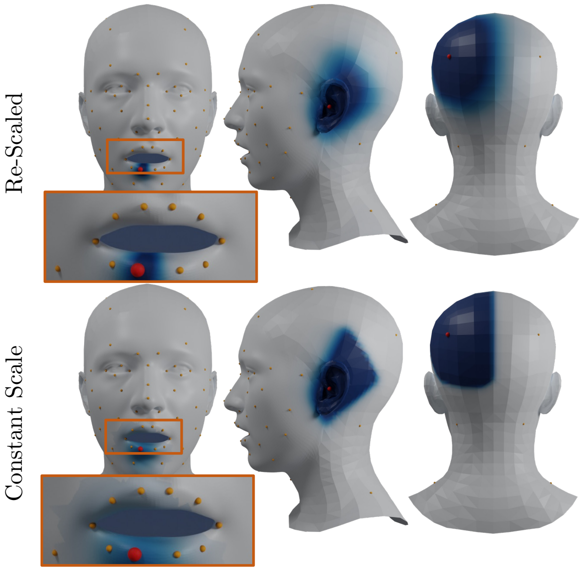

Figure 1. **Re-Scaling w_k:** We show weights w_k for three different anchors (red) among all the 65 anchors (orange). The mesh surface are colored according to w_k where white corresponds to a low value and blue to a large value. The top row shows our re-scaled weights compared to a constant scale (bottom row). Note the discontinuities on the bottom left, and the sharp decay on the bottom right.

Re-Scaling w_k For a given query point x_c and local MLP associated to the anchor point $\mathbf{a}_k$, NPHM uses weights

$$w^*_k(x_c, \mathbf{a}_k) = \exp\left(\frac{-\|x_c - \mathbf{a}_k\|_2}{2\sigma}\right), \qquad (1)$$

and normalizes them to w_k in order to blend the predictions of the individual MLPs. However, when restricting the computations to the set of nearest neighbors, such a constant-scale Gaussian weighting results in discontinuous for points on the boundary of Voronoi cells, i.e. when the set of nearest neighbors changes.

As demonstrated in the bottom left of Fig. 1, the influence of the highlighted anchor point exhibits a sharp boundary. This effect can be mitigated by reducing σ to be significantly smaller than the size of the Voronoi cells. However, due to the non-uniform spatial arrangement of anchors, finding a single σ that ensures smooth boundaries for all anchors is impossible.

Consequently, we vary

$$\sigma(x_c) = \frac{1}{4} \max_{x \in \mathcal{N}_{x_c}} \|x_c - x\|_2 \tag{2}$$

according to the set of nearest neighbors of $\mathcal{N}_{x_c}$. Doing so ensures that $w_k^*(x_c, \mathbf{a}_k)$ decays quickly enough to zero when approaching the boundaries of its Voronoi cell.

2.3. Training Details

2.3.1 Data Preparation

We use the 3D textured scans of the NPHM dataset [3] for training. To this and we sample points on the surface $\mathcal{S}_{\text{surf}}$ and near the surface $\mathcal{S}_{\text{near}}$, and define $\mathcal{S}_{\text{all}} = \mathcal{S}_{\text{surf}} \cup \mathcal{S}_{\text{near}}$. For $x_p \in \mathcal{S}_{\text{all}}$ we precompute its normal $n(x_p)$ and color $\text{RGB}(x_p)$. Additionally, we precompute samples $(x_p, x_c) \in \mathcal{S}_{\text{corr}}$ of corresponding points in posed and canonical space following [12] and using the provided registered meshes in the NPHM dataset.

2.3.2 Loss Functions

We train MonoNPHM in an end-to-end fashion, similar to ImFace [18] which jointly trains geometry and expression networks.

Geometry Supervision The employed losses are similar to [5], however, adopted to dynamic objects similarly to [18]. Hence, the main losses for the geometry and expression supervision put constraints on the zero-level set through

$$\mathcal{L}_{\text{level-set}} = \sum_{x_p \in \mathcal{S}_{\text{surf}}} \|\mathcal{F}_{\texttt{geo}}(\mathcal{F}_{\texttt{exp}}(x_p))\|_1 \tag{3}$$

and on the surface normals through

$$\mathcal{L}_{\text{n}} = \sum_{x_p \in \mathcal{S}_{\text{surf}}} \|\nabla_{x_p} \mathcal{F}_{\texttt{geo}}(\mathcal{F}_{\texttt{exp}}(x_p)) - n(x_p)\|_2, \tag{4}$$

where we omit the dependence on latent codes for brevity. Additionally, we enforce the eikonal constraint

$$\mathcal{L}_{\text{eik}} = \sum_{x_p \in \mathcal{S}_{\text{all}}} \|\nabla_{x_p} \mathcal{F}_{\texttt{geo}}(\mathcal{F}_{\texttt{exp}}(x_p)) - 1\|_2. \tag{5}$$

To guide $\mathcal{F}_{\texttt{exp}}$ during the first half of training we include a correspondence loss

$$\mathcal{L}_{\text{corr}} = \sum_{(x_p, x_c) \in \mathcal{S}_{\text{corr}}} \|\mathcal{F}_{\texttt{exp}}(x_p) - x_c\|_1. \tag{6}$$

On one side this provides direct expression supervision. On the other side $\mathcal{L}_{\text{corr}}$ also enforces the first 3 dimensions of the canonical space to behave as Euclidean as possible. This is not only desirable but also extremely important for the landmark loss $\mathcal{L}_{\texttt{lm}}$ to work. For the same reason, we regularize predicted hyper-dimensions $\omega = [\mathcal{F}_{\texttt{exp}}(x_p)]_\omega$ to be small using

$$\mathcal{L}_{\text{hyper}} = \sum_{x_p \in \mathcal{S}_{\text{all}}} \| \, [\mathcal{F}_{\texttt{exp}_\omega}(x_p)]_\omega \, \|_2. \tag{7}$$

In a similar fashion, we regularize predicted deformations to be small

$$\mathcal{L}_{\text{def}} = \sum_{x_p \in \mathcal{S}_{\text{surf}}} \|\mathcal{F}_{\texttt{exp}}(x_p) - x_p\|_2. \tag{8}$$

Finally, we include the same regularization terms as [3], i.e. we constrain the norm of $\mathbf{z}_{\texttt{geo}}$ and $\mathbf{z}_{\texttt{exp}}$ and apply a symmetry loss on the symmetric parts of $\mathbf{z}_{\texttt{geo}}$.

Anchor Supervision Anchor positions are directly supervised using

$$\mathcal{L}_{\mathcal{A}} = \|\mathbf{a}_{\text{gt}} - \mathcal{A}(\mathbf{z}_{\texttt{geo}})\|_F \tag{9}$$

where the ground truth anchor positions $\mathbf{a}_{\text{gt}}$ are extracted from the registered meshes in FLAME [9] topology, as provided by the NPHM dataset. Therefore, the anchors are supervised to follow the Euclidean coordinate system of the FLAME model. While this seems obvious, we note that without the necessary precautions imposed by $\mathcal{L}_{\text{corr}}$, $\mathcal{L}_{\text{def}}$, and $\mathcal{L}_{\text{hyper}}$, our canonical space becomes non-euclidean, similarly to [8, 11, 20].

Appearance Supervision The appearance codes $\mathbf{z}_{\texttt{app}}$ and network $\mathcal{F}_{\texttt{app}}$ are jointly optimized alongside the geometry, by including

$$\mathcal{L}_{\text{app}} = \sum_{x_p \in \mathcal{S}_{\text{all}}} \|\mathcal{F}_{\texttt{app}}(\mathbf{h}_{\texttt{geo}}(\mathcal{F}_{\texttt{exp}}(x_p)) - \text{RGB}(x_p)\|_1 \tag{10}$$

into our training. Similarly as before, we also regularize the norm of $\mathbf{z}_{\texttt{app}}$. We do not include a perceptual loss during training, as done in [10, 17], since we are focused on geometry reconstruction via inverse rendering, instead of photorealistic appearance.

2.3.3 Training Strategy

Using the above-mentioned losses, we train all networks and latent codes jointly in an auto-decoder fashion [13]. We use the Adam optimizer [7], and periodically divide the learning rates by half every 500 epochs, for a total of 2500 epochs and use a batch size of 64. We start with $lr_{\text{networks}} = 0.0005$, $lr_{\text{lat-can}} = 0.002$ and $lr_{\text{lat-exp}} = 0.01$, for the network parameters, latent codes for canonical space and latent expression codes, respectively.

2.4. Tracking Details

We perform iterative root finding using 5 random samples normally distributed around the canonical anchor $\mathbf{a}_k$ of interest, as we experience similar convergence issues to [1] that are dependent on the initial position.

Since the inside of the mouth is subject to extreme shadows, far beyond what our simple lighting assumptions can explain, we use the predicted facial segmentation masks [19] to down-weigh the color loss $\mathcal{L}_{\texttt{RGB}}$ by a factor of 25 for that region.

Furthermore, we employ several mechanisms to encourage a *coarse-to-fine* optimization. First, we decay all learning rates of the employed Adam optimizer periodically throughout the optimization. The learning rate for the head pose and spherical harmonics parameters ζ start larger and decay faster compared to the learning rate of the latent codes. Second, we increase the inverse standard deviation from the NeuS [15] volume rendering formulation from 0.3 to 0.8. Therefore, the rendering densities are initially distributed widely around the surface, allowing for a large volume that receives gradients in the coarser stages of optimization. Third, the influence of the landmark loss $\mathcal{L}_{\texttt{lm}}$ is strongly decayed throughout the optimization progress. Initial epochs strongly rely on landmark guidance, while later ones are barely affected by it anymore. Additionally, we weigh the landmarks of the eyes, mouth and chin 100 more then the remaining ones.

3. Additional Qualitative Results

3.1. Additional Comparisons

Next to the results in the main paper and our supplementary video, we show additional qualitative comparisons against our baselines in Fig. 2. Note that each row shows a frame from a different sequence, which are reconstructed separately.

Note that we due not show additional results for NHA [4] and HeadNeRF [6], since both methods do not have accurate geometry as their main focus.

3.2. Ablations

While our main document only reported quantitative results of our ablation experiments, due to space reasons, Fig. 3 and our supplementary video show qualitative results. In the following we highlight some key insights from our ablation experiments:

Effect of $\mathcal{L}_{\texttt{lm}}$ Generally, our tracking performs well even when the landmark loss is disabled. However, some extreme expressions are completely missed without it, see the second column in Fig. 3.
Additionally, utilizing a landmark detector trained on large image collections of in-the-wild images provides some robustness against lighting and shadow effects.

Volume Rendering vs. Sphere Tracing Utilizing sphere tracing [16], instead of a volumetric formulation [15], for differentiable SDF-based rendering results in reconstructions that are perceptively dissimilar to the subject. Additionally, we note that the sphere tracing sometimes gets stuck in local minima, where it is not able to remove hair geometry in front of the forehead, see columns four and five.

Spherical Harmonics Since our model is trained on 3D scans, with albedo-like texture, accounting for lighting effects is important. Removing the spherical harmonics term, makes the task slightly ill-posed and generally results in worse reconstruction quality.

Deformation Formulation We ablate our deformation module, consisting of backward deformations and hyper dimensions, against the forward deformation utilized in NPHM [3]. To this end we extend NPHM's canonical space using our proposed approach to include color prediction. We denote this model as NPHM$_{\texttt{app}}$. Due to its invert deformation direction iterative-root-finding is required during rendering and not for the landmark loss. Another difference is that it needs to be trained in two stages according to [3]. Otherwise, the same losses and hyperparamters are used for tracking.

Fig. 3 indicates that the forward deformation module mainly has problems in the mouth region, e.g. with folded lips.

Anchor Layout Additionally, we ablate the proposed anchor layout against the version used in NPHM, which uses 39 anchors instead of our proposed 65 anchors. This mainly results in a slightly less dense landmark loss, and slightly reduced capacity, due to a lower number of local MLPs.

Color Communication Conditioning the color MLP $\mathcal{F}_{\texttt{app}}$ directly on canonical spatial coordinates x_c instead of geometry features $\mathbf{h}_{\texttt{geo}}(x_c)$, gives the model extra freedom since both outputs are less correlated. For example in column 3 this results in a failure to separate the hair and cheek. Additionally, such a communication bottleneck was found to be beneficial for disentangling the geometry and appearance latent spaces [17].

Local vs. Global MLPs Our MLPs modeling the SDF and texture field follow the local structure proposed in [3], i.e. we use an ensemble of local MLPs, each centered around its specific facial anchor points. Additionally, symmetric face regions are represented using the same MLP, but with mirrored coordinates. Our main motivation for choosing such an architecture are the facial anchors, which we exploit to formulate our landmark loss. We realized that it is also possible to use the same landmark loss while using global MLPs for both SDF and texture field. To this end, it is necessary to add the anchor prediction network $\mathcal{A}$ to the architecture, although the predicted anchors are not used anywhere else in that architecture. We find that training such a model is still capable of successfully associating the geometry code $\mathbf{z}_{\texttt{geo}}$ with plausible facial anchors. Nevertheless, the local MLP ensemble still learns a more detailed latent representation, which, for example, shows in the slightly blurry eye reconstructions in columns three and five.

References

[1] 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

[2] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. 1

[3] Simon Giebenhain, Tobias Kirschstein, Markos Georgopoulos, Martin Rünz, Lourdes Agapito, and Matthias Nießner. Learning neural parametric head models. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2023. 1, 2, 3, 4

[4] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18653–18664, 2022. 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. 2

[6] Yang Hong, Bo Peng, Haiyao Xiao, Ligang Liu, and Juyong Zhang. Headnerf: A real-time nerf-based parametric head model. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 3

[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. 3

[8] Jiahui Lei and Kostas Daniilidis. Cadex: Learning canonical deformation coordinate space for dynamic surface representation via neural homeomorphism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022. 2

[9] 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. 2

[10] Connor Z. Lin, Koki Nagano, Jan Kautz, Eric R. Chan, Umar Iqbal, Leonidas Guibas, Gordon Wetzstein, and Sameh Khamis. Single-shot implicit morphable faces with consistent texture parameterization. In *ACM SIGGRAPH 2023 Conference Proceedings*, 2023. 2

[11] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. *ACM Trans. Graph.*, 38(4):65:1–65:14, 2019. 2

[12] 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, 2021. 2

[13] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation.

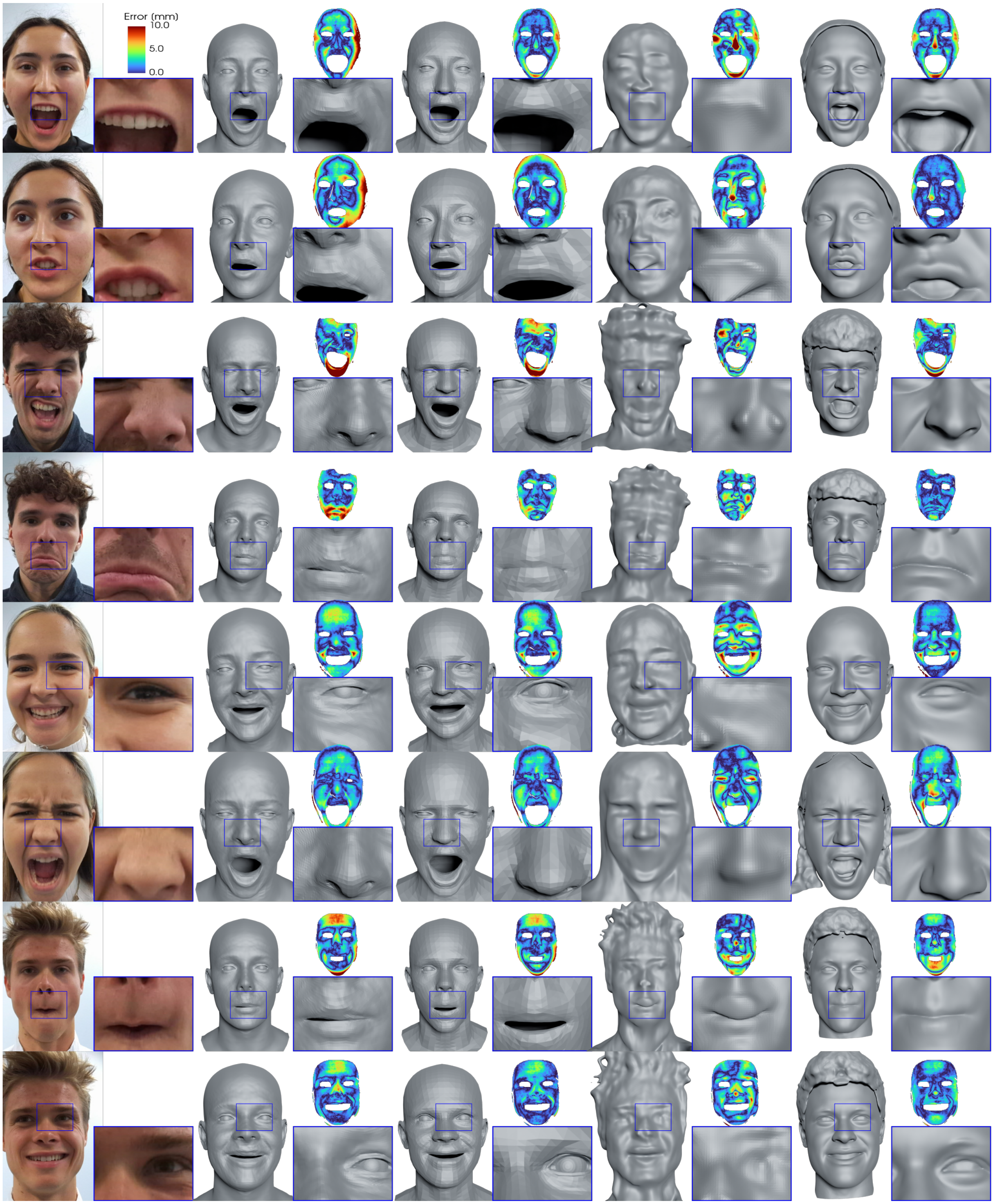

Figure 2. **Tracking Comparison:** We show additional qualitative results of the monocular 3D reconstruction task. The error maps show the color-coded point-to-mesh distance from the back-projected Kinect depth to the reconstruction.

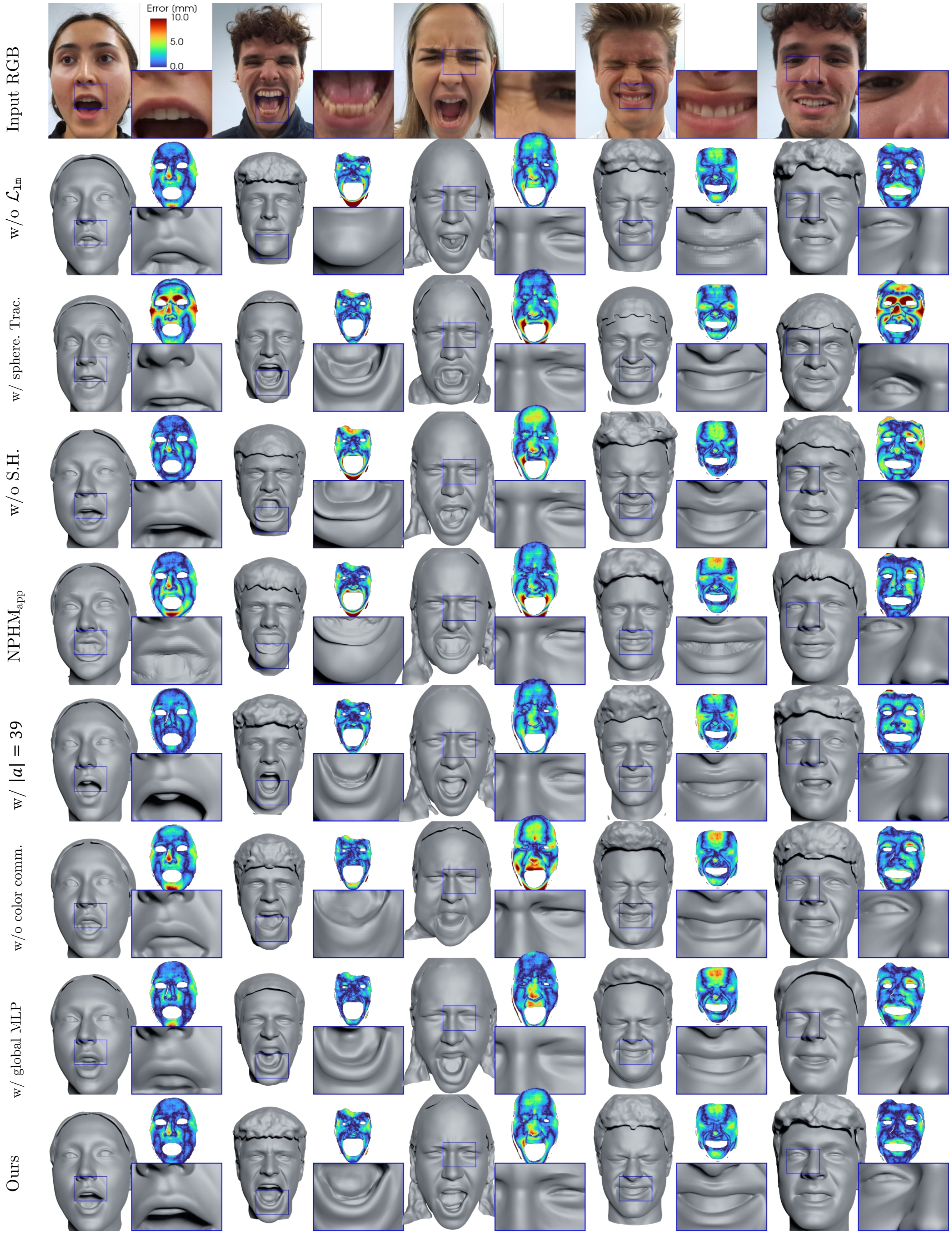

Figure 3. **Ablation Results:** Qualitative comparison of our ablation experiments, as quantitatively reported in Table 2 in the main document. Rows and columns are transposed compared to our other result figures. The error maps show the color-coded point-to-mesh distance from the back-projected Kinect depth to the reconstruction. See Sec. 3.2 for a description of our findings.

In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 3

[14] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020. 1

[15] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *NeurIPS*, 2021. 3

[16] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 3

[17] Mihai Zanfir, Thiemo Alldieck, and Cristian Sminchisescu. Phomoh: Implicit photorealistic 3d models of human heads. *CoRR*, abs/2212.07275, 2022. 2, 4

[18] 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

[19] Yinglin Zheng, Hao Yang, Ting Zhang, Jianmin Bao, Dongdong Chen, Yangyu Huang, Lu Yuan, Dong Chen, Ming Zeng, and Fang Wen. General facial representation learning in a visual-linguistic manner. *arXiv preprint arXiv:2112.03109*, 2021. 3

[20] Yufeng Zheng, Wang Yifan, Gordon Wetzstein, Michael J. Black, and Otmar Hilliges. Pointavatar: Deformable point-based head avatars from videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 2