Authentic Hand Avatar from a Phone Scan via Universal Hand Model

Supplementary Material

In this supplementary material, we provide more experiments, discussions, and other details that could not be included in the main text due to the lack of pages. The contents are summarized below:

- Sec. A: More qualitative results
- Sec. B: More ablation studies
- Sec. C: UHM architectures and loss functions
- Sec. D: Details of adaptation to a phone scan
- Sec. E: Our datasets
- Sec. F: Experiment details
- Sec. G: Failure cases

A. More qualitative results

A.1. ID code interpolation

Fig. A, B, and C show that our UHM produces smoothly changing 3D meshes from the linearly interpolated ID codes, where the two ID codes are from the unseen test set. Our 3D meshes from the interpolated ID codes have a natural and realistic surface. On the other hand, Fig. B and C show that Handy [22] fails to disentangle 3D pose and ID. As each row of all figures is from the same pose but from different ID codes, only ID-related information (*i.e.*, thickness) should change while preserving the 3D pose. However, Fig. B and C show that only changing the ID code of Handy produces 3D meshes with different 3D poses. This is evident in Fig. C as the rightmost result of Handy has a totally different 3D pose from the leftmost one.

A.2. ID code random sampling

Fig. D shows 3D meshes from our UHM with randomly sampled ID codes from the Gaussian distribution and zero 3D poses. Our ID space spans a wide range of ID space, including diverse bone lengths and 3D hand shapes.

A.3. Effectiveness of the image matching loss

Fig. E shows that using our image matching loss of Sec. 4 of the main manuscript produces consistent unwrapped textures compared to the reference texture. (a) has consistent fingernail tips (yellow circles), while (c) produces inconsistent ones compared to those of the (b) reference texture.

A.4. Low-resolution UHM

Fig. F demonstrate that even with a half number of vertices (3K), ours achieves better fidelity than NIMBLE (6K) and Handy (7K). For example, the low-resolution UHM has natural muscle bulging around the thumb and wrinkles around the pinky finger.

A.5. 3D hand avatars

Fig. G shows that our 3D hand avatar achieves sharper textures than HARP [9]. Handy [22] fails to produce authentic results, consistent with Fig. 10 and 11 of the main manuscript. Fig. H additionally shows our adapted 3D hand avatar, rendered with Phong reflection model and environment maps, as in Fig. 1 (b) of the main manuscript. To this end, given an environment map, we first do preconvolution to map the illumination in the environment map to diffuse and specular lighting representation similar to [20]. Then, the final texture is obtained by combining the diffuse and specular representation with our adapted texture (optimized texture of Sec. 5.4 of the main manuscript) according to the normal map from 3D mesh and view direction. The 3D poses of (b) are from the tracked results from a different subject of our studio data, which shows that our hand avatars can be driven with novel poses. The results are not photorealistic due to the limitation of the Phong reflection model, but they show the potential of our hand avatar, which can be combined with future relightable hand models [1].

B. More ablation studies

B.1. Effectiveness of the texture optimization

Fig. I shows that our texture optimization, described in Sec. 5.4 of the main manuscript, further enhances the photorealism of the texture.

B.2. Effectiveness of the TV regularizer during the adaptation

Fig. J shows the effectiveness of the total variation (TV) regularizer to our ShadowNet. Without the TV regularizer, ShadowNet tried to consider all darkness differences between 1) albedo+shadow and 2) captured images as shadows. As a result, local sharp textures, including wrinkles are considered shadows. As described in Sec. 5.3 of the main manuscript, by applying the TV regularizer to the ShadowNet, we can prevent such undesired shadows.

B.3. Extension of DHM to the universal case vs. UHM.

Fig. K shows that when performing the tracking and modeling at the same time, special considerations are necessary for universal hand modeling. We choose DHM [15], a high-fidelity personalized 3D hand model, as a comparison target because it has a similar training pipeline that performs the tracking and modeling at the same time as ours. One critical difference between DHM and our UHM is that DHM

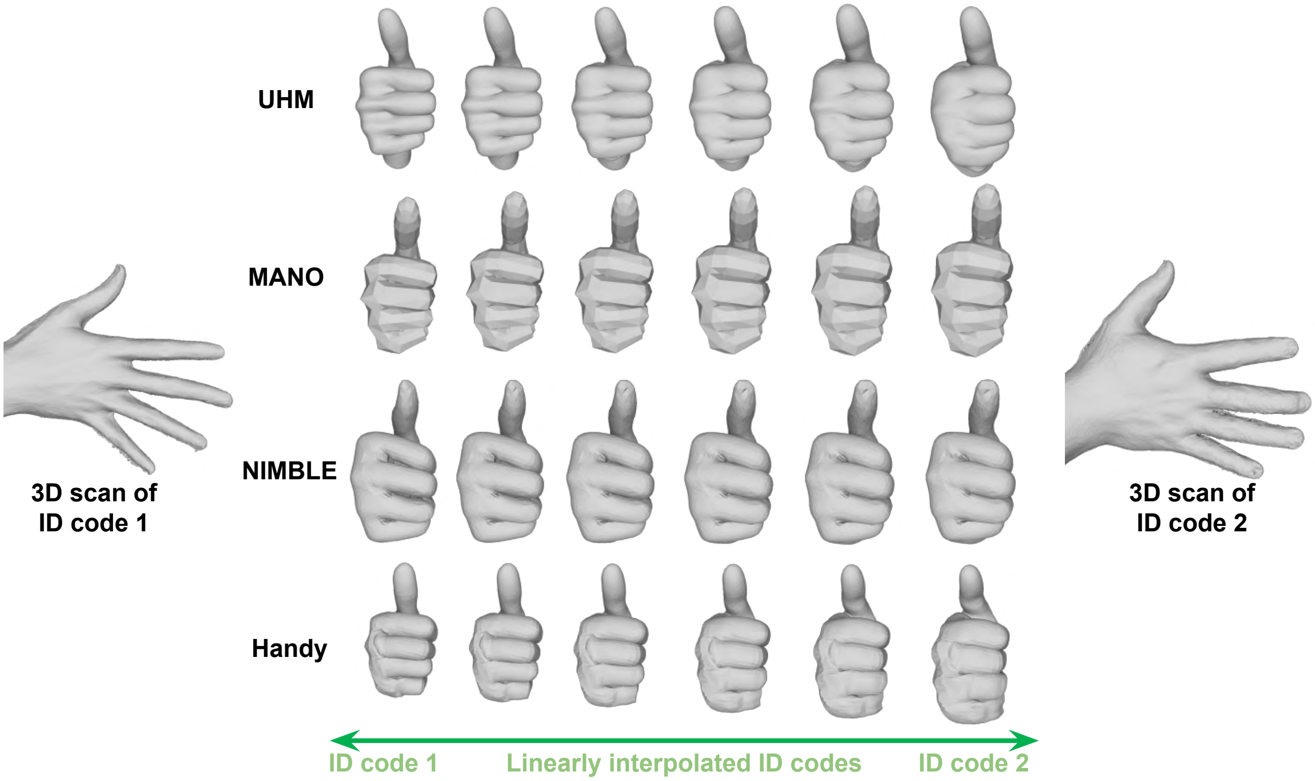


Figure A. Comparison of 3D meshes from linearly interpolated ID codes. The leftmost and rightmost 3D scans show examples of the ID codes 1 and 2. For each row, 3D meshes have the same 3D pose and only ID code changes by a linear interpolation.

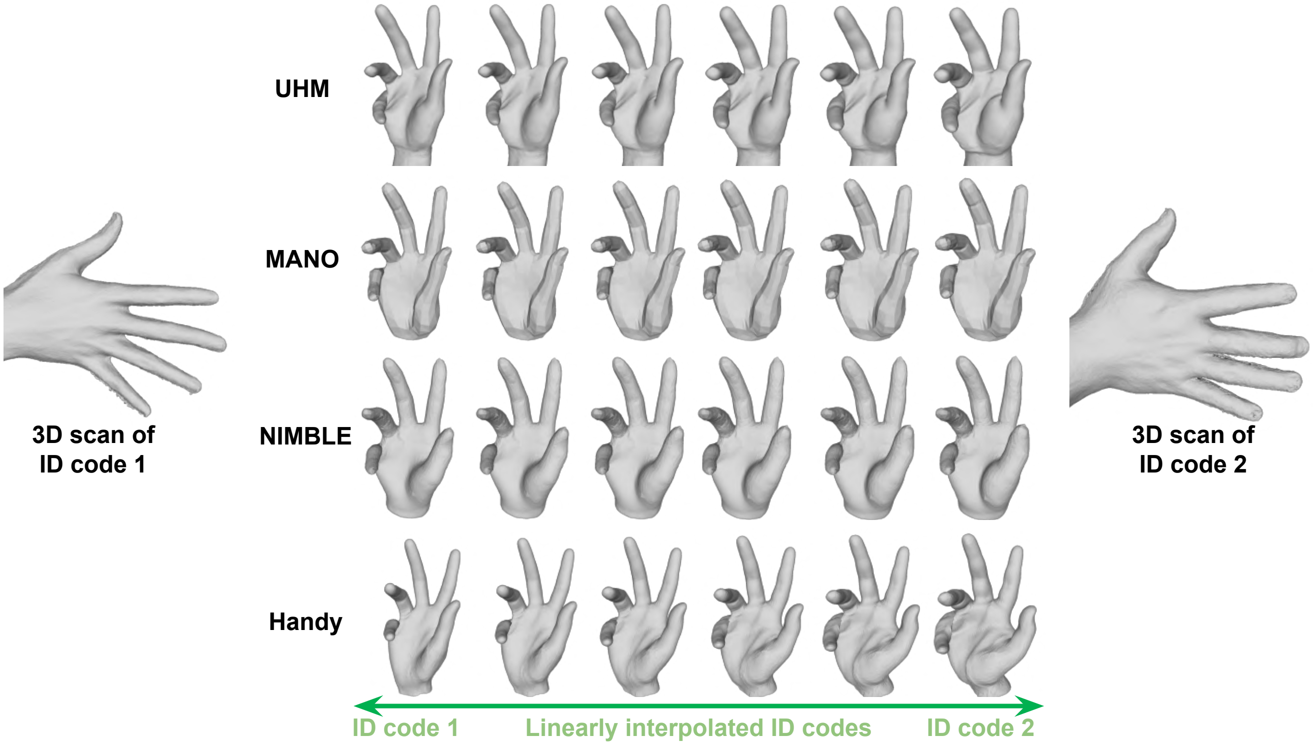


Figure B. Comparison of 3D meshes from linearly interpolated ID codes. The leftmost and rightmost 3D scans show examples of the ID codes 1 and 2. For each row, 3D meshes have the same 3D pose and only ID code changes by a linear interpolation.

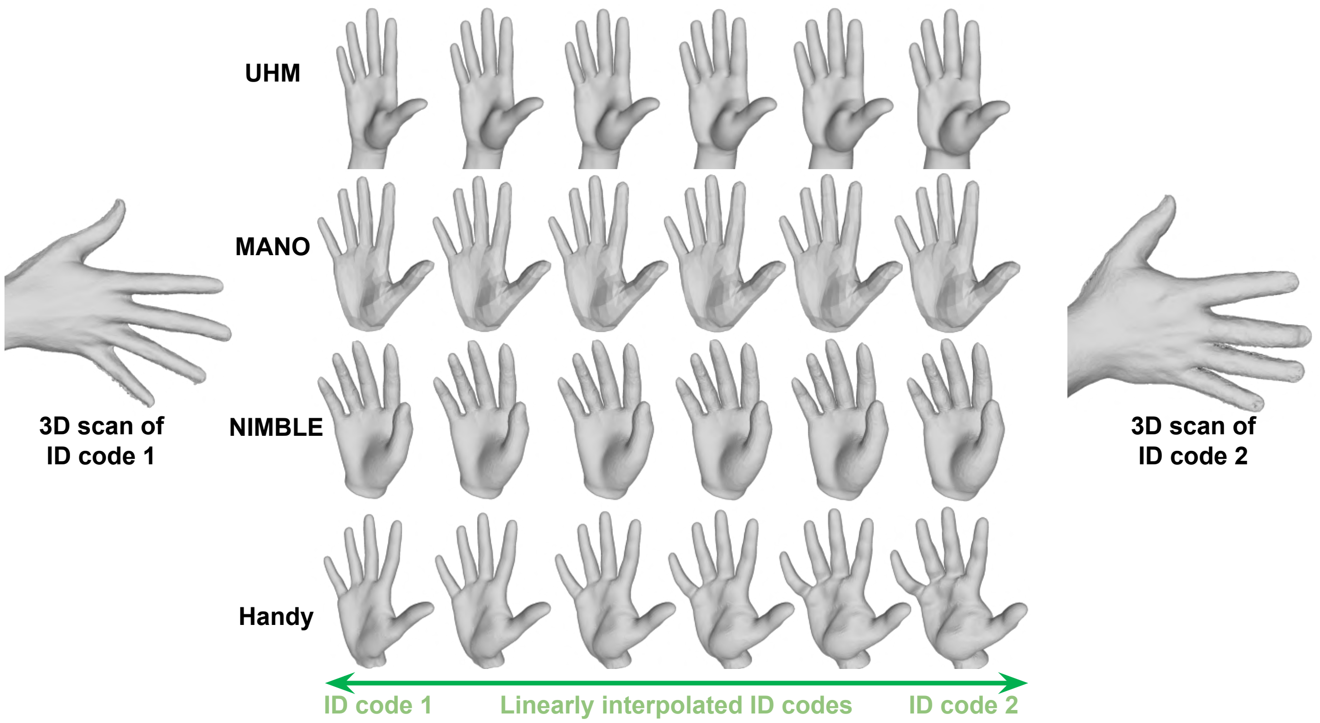


Figure C. Comparison of 3D meshes from linearly interpolated ID codes. The leftmost and rightmost 3D scans show examples of the ID codes 1 and 2. For each row, 3D meshes have the **zero pose (*i.e.*, 3D pose of the template space)**, and only ID code changes by a linear interpolation.

is a personalized 3D hand model, which does not learn the ID space and cannot generalize to novel IDs. For systems that perform the tracking and modeling at the same time, one major difficulty of universal hand modeling is disentangling ID and pose information as all supervision targets, such as 3D joint coordinates, 3D scans, masks, and images, are entangled representations of ID and pose. We effectively achieve the disentanglement by calculating loss functions using two types of 3D meshes: one from both correctives and the other only from the ID-dependent correctives, as described in Sec. 4 of the main manuscript. During the training, the ID-dependent correctives of all frames that belong to the same ID are from the same inputs (*i.e.*, 3D joint coordinates and depth maps of the neutral pose, as described in **IDEncoder and IDDecoder.** of Sec. 3.2 of the main manuscript). Therefore, supervising 3D meshes that are only from the ID-dependent correctives can make IDDecoder formulate meaningful ID space (Fig. K (b)) without being affected by the pose-and-ID-dependent correctives, which naturally achieves the disentanglement of the ID and pose. On the other hand, without the supervision of the 3D meshes that are only from the ID-dependent correctives like DHM, the model cannot disentangle ID and pose, which results in meaningless ID space (Fig. K (a)). Such disentanglement is especially challenging for systems that perform the tracking and modeling at the same time because previous separate pipeline [12, 22, 23] can perform tracking for each ID, which can naturally provide assets that only have ID information without pose by canceling pose from the tracked meshes.

C. UHM network architectures and regularizers

C.1. Network architectures

We describe detailed network architectures of UHM, briefly described in Sec. 3.2 of the main manuscript.

IDEncoder. IDEncoder outputs ID code $\mathbf{z}^{\text{id}} \in \mathbb{R}^{32}$ from a pair of a depth map and 3D joint coordinates of each training subject. To prepare the inputs of the IDEncoder, we first select a single pair of a 3D scan and 3D joint coordinates for each subject. Hence, there are subjects number of (3D scan and 3D joint) pairs. As the IDEncoder should capture only ID-related information, the poses of the inputs of the IDEncoder should be normalized. To this end, we take the pairs from the first frame of the captures as the poses at the first frames are close to zero poses, which we call *neutral poses*. Then, we rigidly align the selected 3D scans and 3D joint coordinates to a reference coordinate system and render depth maps from the aligned 3D scans from the front and back views. In this way, we can further normalize views, which exist and are hard to be normalized in images.

ResNet-18 [7] takes two-view depth maps of neutral pose for each subject. Please note that IDEncoder always

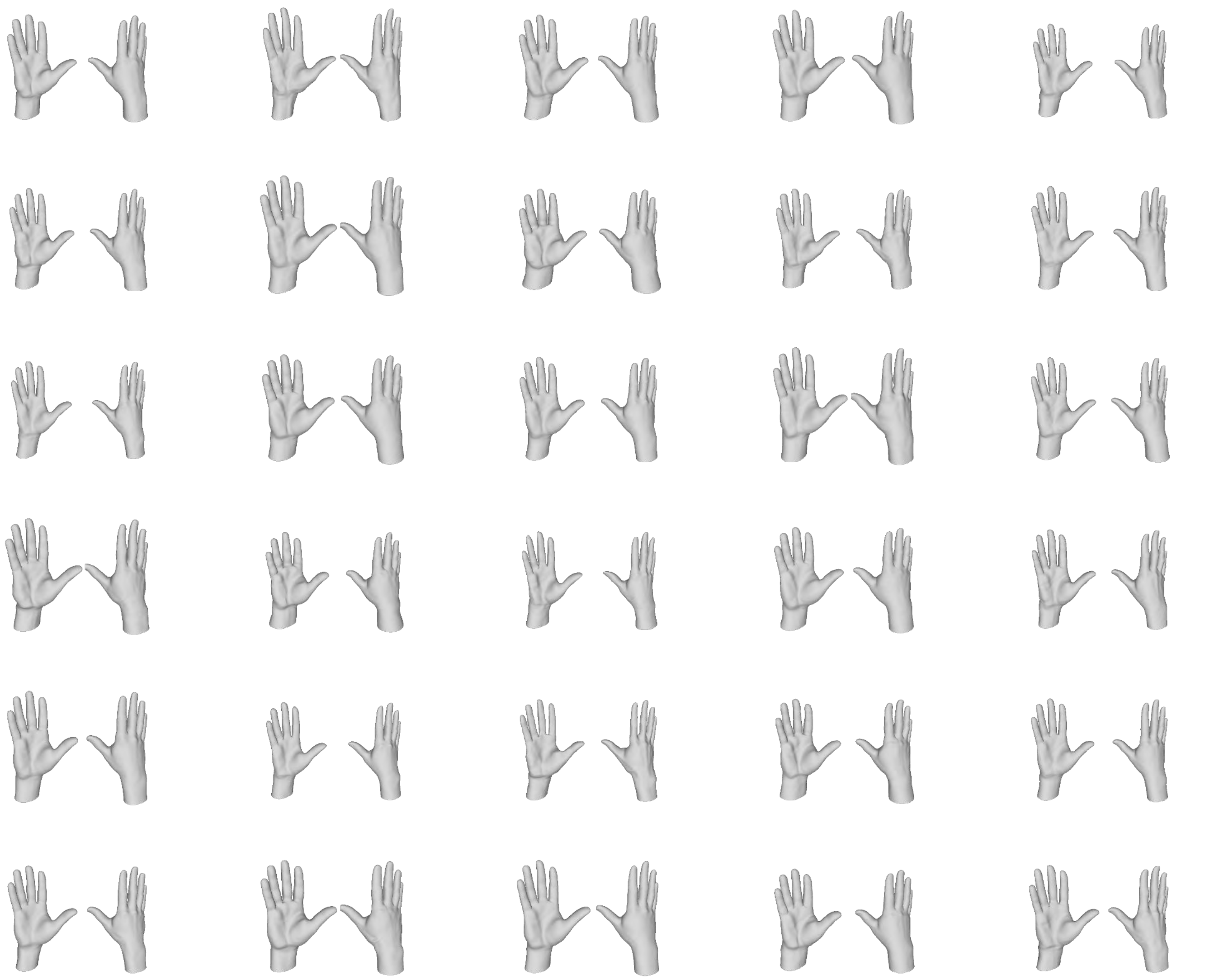
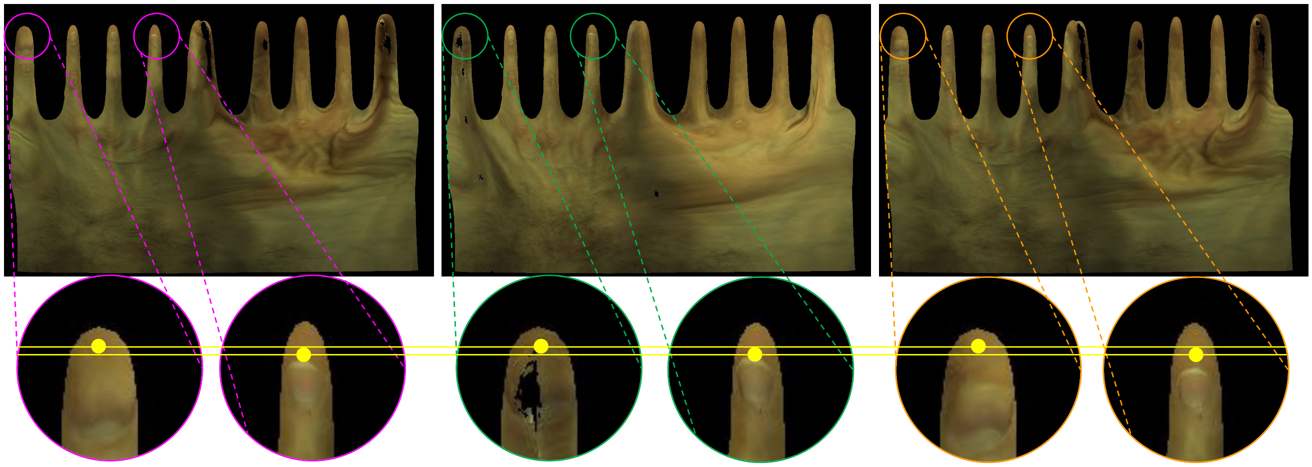


Figure D. 3D meshes from randomly sampled ID codes from the Gaussian distribution with zero 3D poses.

takes the same inputs for the same subject during the training. Hence, the size of the mini-batch is $2N_\mathrm{s}$, where N_s is the number of unique subjects in the mini-batch of PoseEncoder. The ResNet-18 is initialized with ImageNet [24] classification, and we discard fully connected layers. The output of ResNet-18 is a 512-dimensional feature vector. We reshape the feature vector to a 1024-dimensional one, which represents a multi-view feature for each subject. The multi-view feature is concatenated with the 3D joint coordinate of a neutral pose and passed to two fully connected layers, which produce the id code $\mathbf{z}^{\mathrm{id}}$ using the reparameterization trick [10]. The two fully connected layers consist of 512 hidden units and an intermediate ReLU activation function.

IDDecoder. IDDecoder takes the ID code $\mathbf{z}^{\mathrm{id}}$ and outputs ID-dependent skeleton correctives $\Delta\bar{\boldsymbol{J}}^{\mathrm{id}}$ and ID-dependent vertex correctives $\Delta\bar{\boldsymbol{V}}^{\mathrm{id}}$. The IDDecoder consists of two fully connected layers with a ReLU activation function for the non-linearity. The hidden size of the fully connected layers is set to 512. $\Delta\bar{\boldsymbol{J}}^{\mathrm{id}}$ should not replicate any changes, which can be replicated by 3D joint rotations. In other words, 3D hands should be in the "zero pose" after applying $\Delta\bar{\boldsymbol{J}}^{\mathrm{id}}$ to the template mesh. Hence, except for child joints of the wrist, we enable only 1 degree of freedom (DoF) of $\Delta\bar{\boldsymbol{J}}^{\mathrm{id}}$ to restrict it to only affect the lengths of fingers. In this way, the learned ID space is not mixed with the pose.

PoseEncoder. PoseEncoder outputs 6D rotation [29] of joints $\boldsymbol{\Theta}$ from a pair of a single RGB image and 3D joint coordinates of arbitrary poses and identities. The 3D global rotation and translation are obtained by rigidly aligning wrist and four finger root joints (except the thumb root joint) to the target 3D joint coordinates. Unlike IDEncoder's inputs consist of a single pair of each subject, PoseEncoder's inputs are from any poses and subjects. Our PoseEncoder


(a) Unwrapped texture with image matching loss (Ours)

(b) Reference texture

(c) Unwrapped texture without image matching loss

Figure E. Comparison of 3D hand avatars on HARP dataset [9].

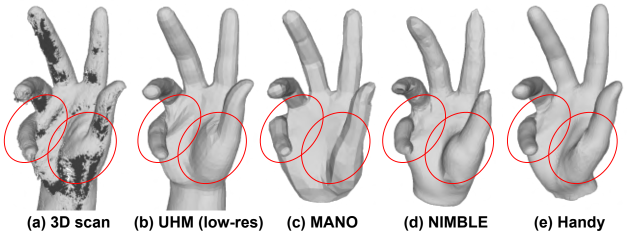


Figure F. Comparison of the low-resolution UHM and previous 3D hand models [12, 22, 23] on our test set.

has a similar network architecture as Pose2Pose [17]. The ResNet-50 of Pose2Pose is initialized with ImageNet [24] classification, and the remaining parts are randomly initialized.

PoseDecoder. PoseDecoder outputs pose-and-ID-dependent vertex corrective $\Delta\bar{\boldsymbol{V}}^{\text{pose}}$ in a sparse way using local joint clusters for better generalization to unseen poses following STAR [19]. To this end, we make J number of local joint clusters, where each cluster consists of 6D rotations of a joint, its parent joint, and a child joint. J denotes the number of joints. We additionally concatenate the ID code for each cluster. Hence, each local joint cluster has the dimension of $\mathbb{R}^{18+32}$, where 18 and 32 represent three 6D rotations and the dimension of the ID code, respectively. Please note that we pass 6D rotations after masking invalid DoFs and root rotation to zero. Then, the local joint clusters are passed to two separable convolutions with an intermediate ReLU activation function for a non-linearity. The hidden size of the separable convolution is set to 256. The output of the separable convolutions of each local joint cluster has the dimension of $\mathbb{R}^{V\times 3}$. V denotes the number of vertices of our template mesh. Formally, we denote the above process by $\boldsymbol{F}_j = f(\theta_j, \theta_{p(j)}, \theta_{c(j)}, \mathbf{z}^{\text{id}})$, where $\boldsymbol{F}_j$ denotes the output of the separable convolution of jth local joint cluster. $p(j)$ and $c(j)$ denote parent child joint of jth joint, respectively. The final pose-and-ID-dependent vertex corrective $\Delta\bar{\boldsymbol{V}}^{\text{pose}}$ is obtained by $\Delta\bar{\boldsymbol{V}}^{\text{pose}} = \sum_j \boldsymbol{\Phi}_j(f(\theta_j, \theta_{p(j)}, \theta_{c(j)}, \mathbf{z}^{\text{id}}) - f(\mathbf{0}, \mathbf{0}, \mathbf{0}, \mathbf{z}^{\text{id}}))$. $\boldsymbol{\Phi} \in \mathbb{R}^{V\times J}$ is a mask, which introduces sparsity. It is initialized with a geodesic distance between the vth vertex and jth joint in the template mesh. We subtract the output of the separable convolution from the zero pose to prevent pose-and-ID-dependent vertex corrective $\Delta\bar{\boldsymbol{V}}^{\text{pose}}$ from replicating only ID-dependent geometry, which should be replicated by ID-dependent vertex corrective $\Delta\bar{\boldsymbol{V}}^{\text{id}}$.

C.2. Loss functions for the tracking and modeling

Our UHM is trained in an end-to-end manner by minimizing L, defined as below:

$$\begin{aligned} L =& L_{\text{pose}} + 10L_{\text{p2p}} + 0.1L_{\text{mask}} + 0.1L_{\text{img}} \\ &+ 0.01L_{\boldsymbol{\Theta}} + 0.001L_{\mathbf{z}^{\text{id}}} + 1000L_{\Delta\bar{\boldsymbol{V}}^{\text{id}}} + 10L_{\Delta\bar{\boldsymbol{V}}^{\text{pose}}} \\ &+ 75000L_{\text{lap}} + 0.001L_{\boldsymbol{\Phi}} + 0.1L_{\text{vol}} + 0.1L_{\text{cut}}, \end{aligned} \tag{1}$$

where L_{pose}, L_{p2p}, L_{mask}, and L_{img} are described in the Sec. 4 of the main manuscript. The remaining loss functions are regularizers, described below.

First, we minimize $L_{\boldsymbol{\Theta}}$, a squared $L2$ norm of $\boldsymbol{\Theta}$ after converting it to an axis-angle representation, to prevent extreme rotations. Second, we minimize $L_{\mathbf{z}^{\text{id}}}$, a KL divergence between $\mathbf{z}^{\text{id}}$ and the normal Gaussian distribution. In this way, we can make the ID latent space follow the Gaussian distribution, necessary for sampling novel ID from a known (*e.g.*, Gaussian) distribution. Third, we minimize $L_{\Delta\bar{\boldsymbol{V}}^{\text{id}}}$ and $L_{\Delta\bar{\boldsymbol{V}}^{\text{pose}}}$, a squared $L2$ norm of the tangential component of $\Delta\bar{\boldsymbol{V}}^{\text{id}}$ and $\Delta\bar{\boldsymbol{V}}^{\text{pose}}$, respectively. They

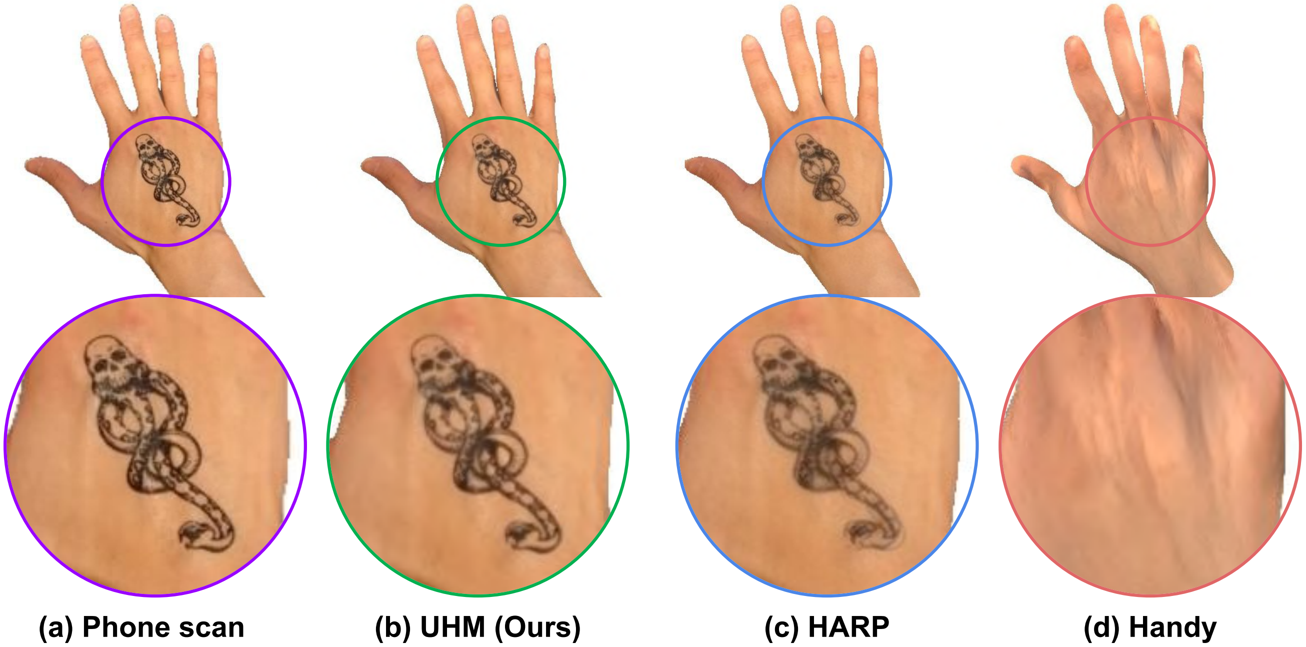


Figure G. Comparison of 3D hand avatars on HARP dataset [9].

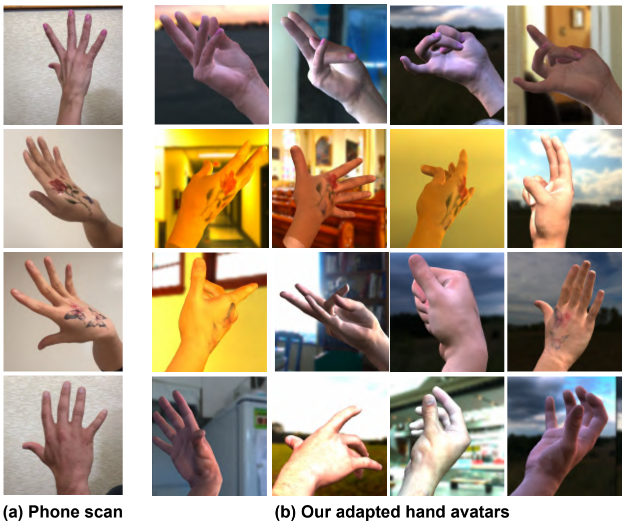


Figure H. Our adapted 3D hand avatar with the Phong reflection model and environment maps [4, 8].

prevent the vertex correctives from overwhelming the ID-dependent skeleton corrective $\Delta\bar{\boldsymbol{J}}^{\text{id}}$. To be more specific, we encourage the finger lengths to be adjusted mainly by $\Delta\bar{\boldsymbol{J}}^{\text{id}}$, not by the vertex correctives. Fourth, we minimize L_{lap}, the Laplacian regularizer for smooth surface. Like L_{p2p} and L_{mask}, we compute two types of this regularizer from 1) both correctives and 2) only ID-dependent corrective to learn meaningful ID space. Fifth, we minimize $L_{\boldsymbol{\Phi}}$, a $L1$ norm of $\text{ReLU}(\boldsymbol{\Phi})$, following STAR [19]. In this way, we can encourage sparsity of the $\Delta\bar{\boldsymbol{V}}^{\text{pose}}$, beneficial for the generalizability to unseen 3D poses. Sixth, we minimize

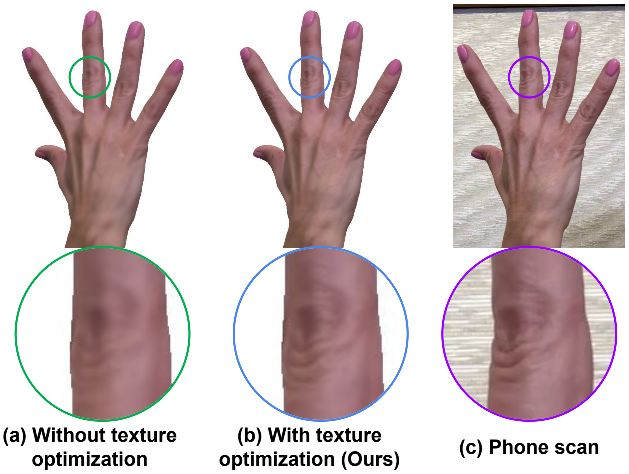
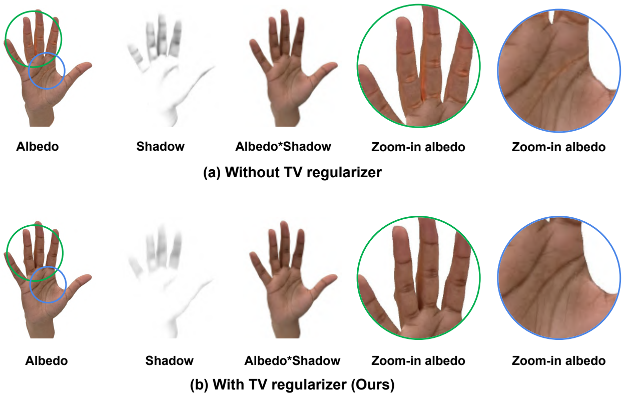

(a) Without texture optimization
(b) With texture optimization (Ours)
(c) Phone scan

Figure I. The effectiveness of the texture optimization during our phone adaptation.

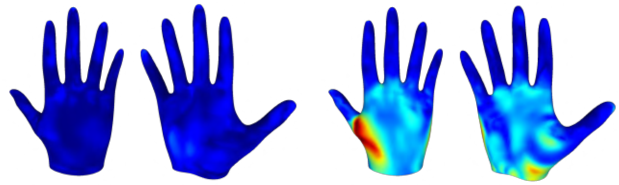
L_{vol}, a volume-preserving regularizer. It first pre-calculates the radius of spheres for each finger in the zero pose space only with $\Delta\bar{\boldsymbol{V}}^{\text{id}}$ without $\Delta\bar{\boldsymbol{V}}^{\text{pose}}$. Then, L_{vol} is the difference between 1) the distance from vertices to sphere radius and 2) the radius of the sphere if the distance shorter than the radius. It encourages our UHM to preserve the minimal volume of each finger, where the minimal values are calculated in the zero pose space with $\Delta\bar{\boldsymbol{V}}^{\text{id}}$. Finally, we minimize L_{cut} for a flat cut around the forearm. To this end, we make a virtual vertex at the center of the cut and make virtual triangles using the virtual vertex and pairs of two connected vertices at the cut. L_{cut} is a $L1$ distance between dot products of all those virtual triangles. In this way, we



(a) Without TV regularizer

(b) With TV regularizer (Ours)

Figure J. The effectiveness of the total variation (TV) regularizer to the ShadowNet.



(a) Extension of DHM to the universal case

(b) UHM (Ours)

Figure K. Comparison of the standard deviation of ID-dependent vertex corrective $\Delta\bar{V}^{\text{id}}$. The correctives for the standard deviation computation are obtained by 1) randomly sampling 512 ID code $\mathbf{z}^{\text{id}}$ from the normal Gaussian and 2) passing them to pre-trained IDDecoder.

can encourage all vertices at the cut to be on the same plane, which results in a flat cut.

D. Details of adaptation to a phone scan

We provide detailed descriptions of our adaptation stage, described in Sec. 5 of the main manuscript.

D.1. Geometry fitting

For the geometry fitting, we designed PoseNet, which has a similar network architecture to Pose2Pose [17] with minor modifications. The PoseNet outputs 3D global rotation, 3D pose $\boldsymbol{\Theta}$, pose code, and 3D global translation of UHM from an image, a depth map, a mask, and 2D joint coordinates, where the inputs are obtained from Sec. 5.1 of the main manuscript. The pose code is a latent vector of a pre-trained VAE, which embeds plausible hand pose space similar to V-Poser [21]. The VAE is pre-trained on our capture studio dataset and fixed during the adaptation stage.

We randomly initialize PoseNet before the training. The PoseNet is used for the pose tracking, a similar spirit of neural annotators [18]. In addition to the outputs of the network, we optimize ID code $\mathbf{z}^{\text{id}}$, shared across all frames as all frames are from the same person. With the outputs of the network and the ID code, we obtain 3D mesh using pre-trained decoders of UHM, used to unwrap images to the UV space.

The PoseNet is trained in a self-supervised way by being trained with the inputs of the network (*i.e.*, 2D joint coordinates, a depth map, and a mask). During the training, we fixed pre-trained decoders of UHM. For the kinematic-level personalization (*e.g.*, bone lengths), we minimize $L1$ distance between projected 2D hand joint coordinates and the target. Also, for the surface-level personalization (*e.g.*, thickness of hand surface), we minimize $L1$ distance between the rendered mask and the target. We additionally minimize the $L1$ distance between the rendered depth map and the target to address the depth and scale ambiguity. Finally, we minimize the $L1$ distance between 3D joint coordinates from the pose code and MANO parameters, where the MANO parameters are from an off-the-shelf regressor [14]. In this way, we can address the depth ambiguity of the 2D keypoints. Then, 3D joint angles and 3D mesh from the pose code are used to supervise those from the 3D pose $\boldsymbol{\Theta}$.

D.2. ShadowNet

We first tile 3D global rotation, 3D pose $\boldsymbol{\Theta}$, and ID code $\mathbf{z}^{\text{id}}$ to all texels in the UV space. In other words, all texels have the same concatenated 3D global rotation, 3D pose, and ID code. Then, we compute the dot product between 1) the normal vector of each vertex and 2) a vector from the camera to each vertex, which becomes a viewpoint feature for each vertex. We warp the per-vertex viewpoint feature to the UV space and concatenate it with the prepared tiled texels, which become the input of our ShadowNet. Given a fixed environment during the phone scan, all inputs of our ShadowNet can determine casted shadow. To distinguish each texel with its own semantic meaning, we add a learnable positional encoding to each texel and pass it to ShadowNet. To enlarge the size of the receptive field effectively, we start from 32 downsampled UV space compared to that of our UV textures.

The ShadowNet first converts the input to a 256-dimensional feature map with a convolutional layer. Then, for each resolution, we apply one convolutional layer, followed by group normalization [27] and SiLU activation function [3]. We used the nearest neighbor for the upsampling. After three times of upsampling, we apply bilinear upsampling four times and the sigmoid activation function to normalize the values of the shadow from 0 to 1.

E. Our datasets

We provide detailed descriptions of our two types of datasets.

E.1. Studio dataset

Our capture studio has 170 calibrated and synchronized cameras. All cameras lie on the front, side, and top hemispheres of the hand and are placed at a distance of about

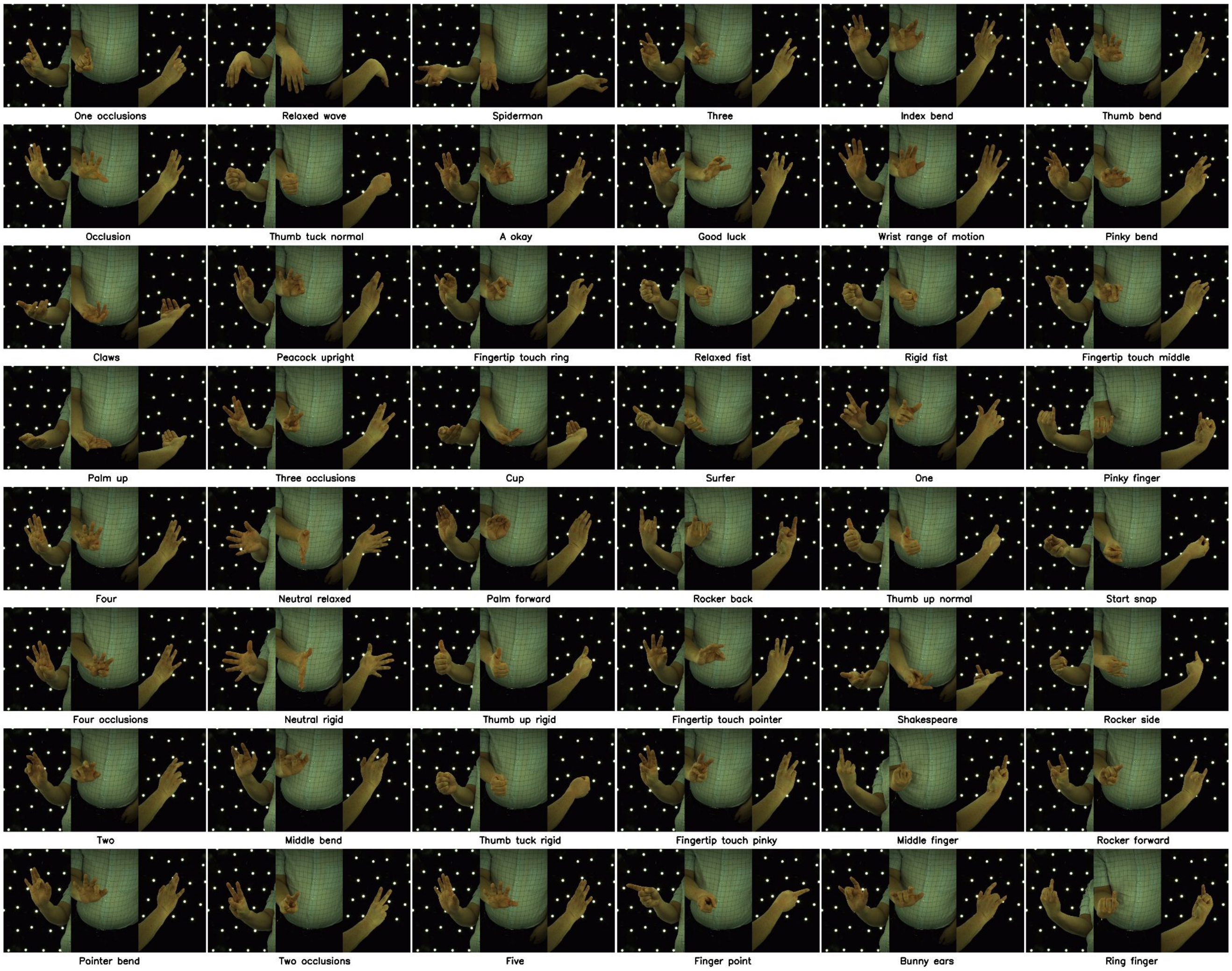

One occlusions
Relaxed wave
Spiderman
Three
Index bend
Thumb bend
Occlusion
Thumb tuck normal
A okay
Good luck
Wrist range of motion
Pinky bend
Claws
Peacock upright
Fingertip touch ring
Relaxed fist
Rigid fist
Fingertip touch middle
Palm up
Three occlusions
Cup
Surfer
One
Pinky finger
Four
Neutral relaxed
Palm forward
Rocker back
Thumb up normal
Start snap
Four occlusions
Neutral rigid
Thumb up rigid
Fingertip touch pointer
Shakespeare
Rocker side
Two
Middle bend
Thumb tuck rigid
Fingertip touch pinky
Middle finger
Rocker forward
Pointer bend
Two occlusions
Five
Finger point
Bunny ears
Ring finger

Figure L. Examples of poses of the training set of our studio dataset.

one meter from it. Images are captured with 4096 × 2668 pixels at 30 frames per second. We pre-processed the raw video data by performing 2D keypoint detection [25] and 3D scan [6]. The keypoint detector is trained on our held-out human annotation dataset, which includes 900K images with 3D rotation center coordinates of hand joints, where our manual annotation tool is similar to that of Moon *et al*. [16]. The predicted 2D keypoints of each view were triangulated with RANSAC to robustly obtain the groundtruth (GT) 3D hand joint coordinates. The combination of 2D keypoint detector and triangulation, used to obtain GT 3D hand joint coordinates, achieves a 1.71 mm error on our held-out human-annotated test set, which is quite low. Fig. L and M show pose examples of the training and testing sets of our studio dataset, respectively.

E.2. Phone scan dataset

Fig. N shows examples of our phone scan dataset. The training set mainly consists of simple poses, where the 3D global rotation of the hand mainly changes, and the 3D pose and 3D translation of the hand remain almost static. The testing set consists of diverse poses, such as a fist and thumb-up.

F. Experiment details

F.1. Fitting for Sec. 6.2

For the comparisons in Fig. 9 and Tab. 1 of the main manuscript, we used 3D joint coordinates and 3D scans as target data for the fitting, the most typical setting of the tracking. For the fitting, we minimized 1) $L1$ distance between output and target 3D joint coordinates, 2) the P2S distance from 3D scans, and 3) $L2$ regularizers to the parameters. The $L2$ regularizer is introduced to prevent extreme meshes. Each loss term is weighted by 1, 10, and 0.001. As each 3D hand model has slightly different 3D joint locations despite the same semantic meaning, we do not report 3D joint error following [2, 12, 23]. For the same reason, we turned off the 3D joint loss during the fit-

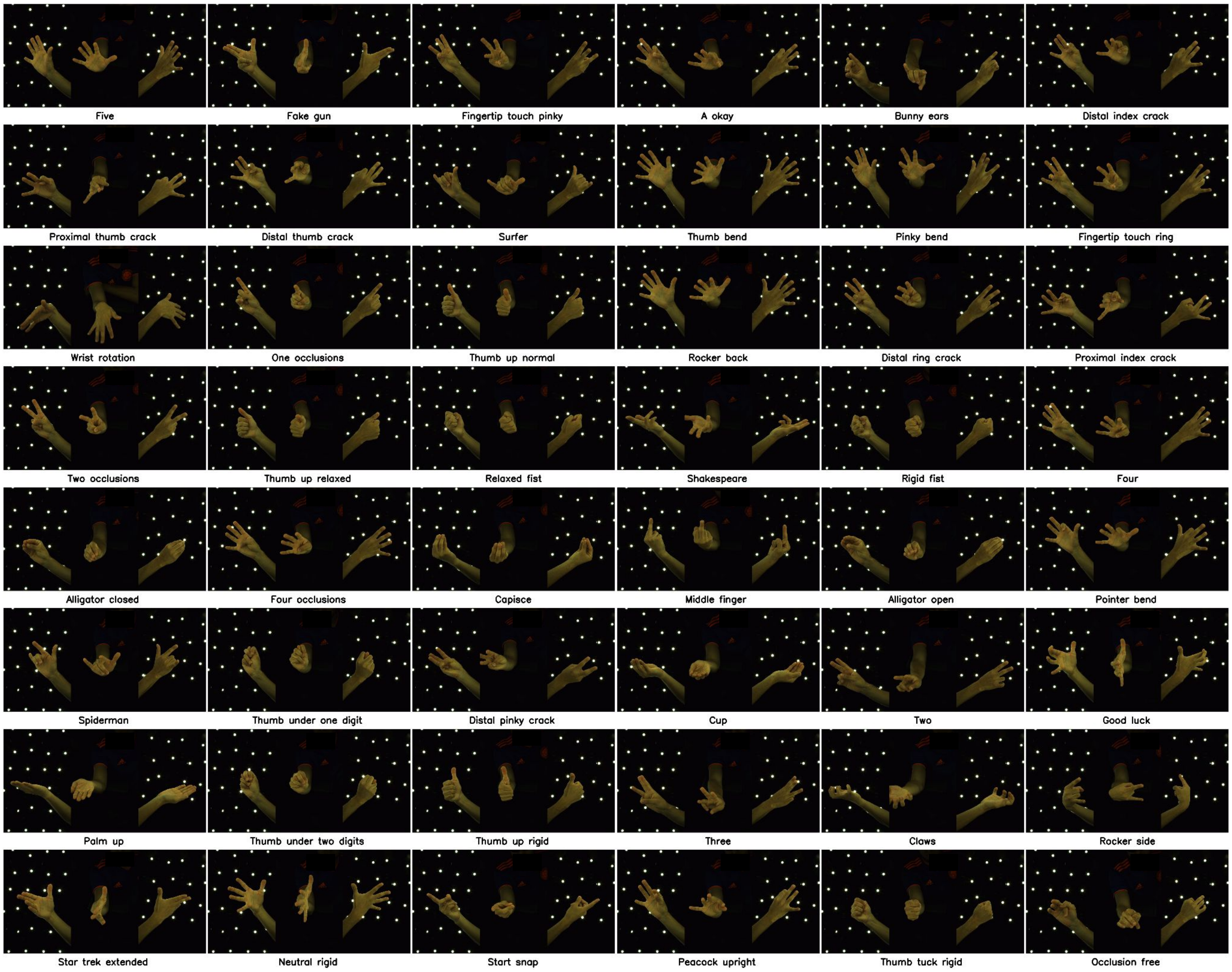
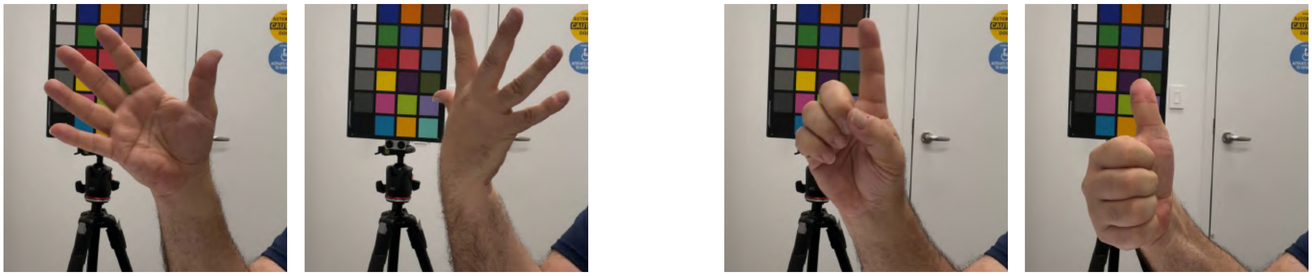


Figure M. Examples of poses of the testing set our studio dataset.



(a) Training images

(b) Testing images

Figure N. Examples of our phone scan dataset.

ting after enough iterations.

For the comparison in Tab. 2 of the main manuscript, we followed the evaluation protocol of LISA [2]. Specifically, we pre-define various numbers of available viewpoints and fit 3D pose and ID code to 2D joint coordinates of those viewpoints. Due to the depth ambiguity from 2D supervisions from a few viewpoints, we used the pose prior, used in our adaptation stage of Sec. D.1, as LISA also used geometry prior from large-scale data. As their codes are not publicly available, we brought their numbers from their paper.

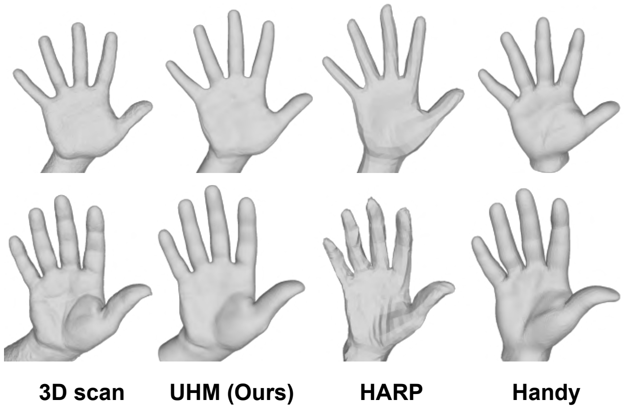


Figure O. Comparison of 3D scan and 3D meshes from various hand adaptation pipelines.

F.2. Handy fitting for Sec. 6.3

We use the same PoseNet and loss functions of ours, described in Sec. D, except for one thing: we used VGG loss function [11] on the rendered image, while we used LPIPS [28] on the rendered image for the Handy texture fitting following their paper. The latent code of the Handy's texture is shared across all frames and is optimizable.

F.3. P2S calculation of Tab. 3

We calculated the P2S errors of the adapted 3D hand avatar in Tab. 3 of the main manuscript. To this end, we first selected a frame with the neutral pose of studio capture of the four subjects, where the four subjects have co-captured studio and phone scan data. From the studio data, we have 3D joint coordinates and 3D scan of the neutral pose. Then, we optimize 3D pose and translation of each adapted avatars by minimizing $L1$ 3D joint distance and point-to-point loss function from the studio data, described in Sec. 4 of the main manuscript. During the optimization, we fix ID-related information, such as ID code $\mathbf{z}^{\text{id}}$ of ours. The P2S errors are calculated between the optimized meshes of each avatar and 3D scan from our studio data. Fig. O visualizes the optimized mesh and 3D scan. For UHM and HARP, we excluded the vertices on the forearm when calculating the 3D errors as they are too unconstrained.

F.4. HARP dataset

As the HARP dataset does not provide depth maps, we do not use the depth map loss function in our pipeline. We used Mediapipe [26] to obtain 2D hand joint coordinates and used RVM [13] to obtain foreground masks. All remaining things are the same as what is described in Sec. D for the experiments on the HARP dataset. Ours, HARP, and Handy are equally fitted to the same sequences and are evaluated with the same metrics.

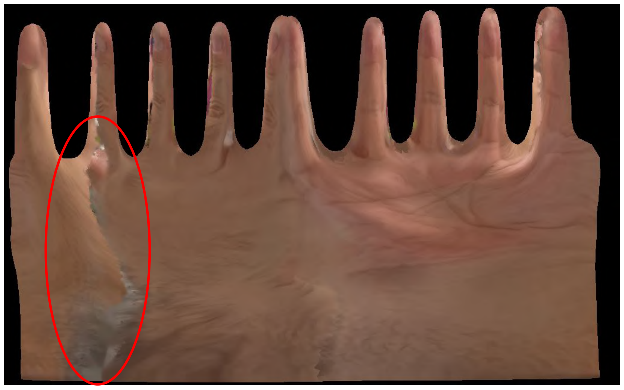


Figure P. Our optimized texture after removing shadow with the ShadowNet. The highlighted area has an evident artifact.

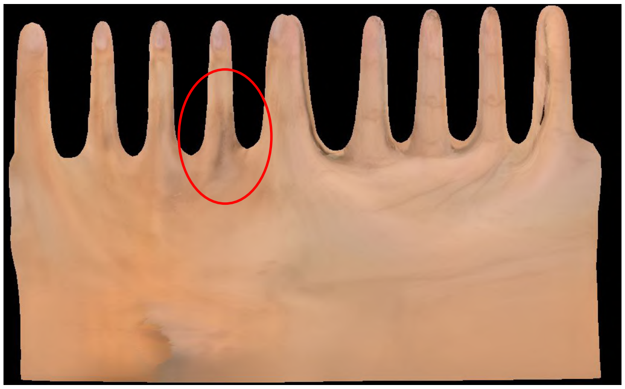


Figure Q. Our optimized texture after removing shadow with the ShadowNet. The highlighted area has an evident artifact.

G. Failure cases

Geometry fitting. We found that our geometry fitting pipeline (Sec. 5.2 of the main manuscript) sometimes suffers from a surface-level misalignment. In the geometry fitting stage, dense supervisions, such as DensePose [5] of the 3D human body, are not available. Such a lack of dense supervision makes our 3D geometry suffer from surface-level misalignment despite the accurate keypoint-level alignment. Although the image loss during the texture optimization (Sec. 5.2 of the main manuscript) provides the dense supervision, its initial texture is from the geometry fitting (Sec. 5.2 of the main manuscript), which can suffer from the surface-level misalignment.

Texture unwrapping. Fig. P shows a failure case happens in the texture unwrapping. There is an evident vertical artifact along the left part of the figure. The reason for such artifacts is that during the phone capture, the subject exposes the left and right parts of the vertical line with very different poses at different time steps. Hence, pose-dependent skin color changes and view-dependent shading of those left and right parts become very different, which results in different colors and an evident vertical line between the left and right parts. We tried to smooth such a region; however, it was not enough as the color difference is too big.

ShadowNet. Fig. Q shows a failure case of our ShadowNet. Although most of the shadow is removed, the highlighted area still has a small amount of shadow. The remaining shadow is especially evident as the skin color of this subject is bright. We think the reason for the remaining shadow is the regularizers to the ShadowNet to prevent it from considering black tattoos as shadows. Also, its capability is not guaranteed for *smooth* black tattoos and black fingernail polish. Due to the ambiguity of the intrinsic decomposition, it might perform badly in low-light conditions; we think this limitation applies to all current methods.

References

[1] Zhaoxi Chen, Gyeongsik Moon, Kaiwen Guo, Chen Cao, Stanislav Pidhorskyi, Tomas Simon, Rohan Joshi, Yuan Dong, Yichen Xu, Bernardo Pires, et al. URHand: Universal relightable hands. In *CVPR*, 2024. 1

[2] Enric Corona, Tomas Hodan, Minh Vo, Francesc Moreno-Noguer, Chris Sweeney, Richard Newcombe, and Lingni Ma. LISA: Learning implicit shape and appearance of hands. In *CVPR*, 2022. 8, 9

[3] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural networks*, 2018. 7

[4] Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. Learning to predict indoor illumination from a single image. *ACM TOG*, 2017. 6

[5] Rıza Alp Güler, Natalia Neverova, and Iasonas Kokkinos. DensePose: Dense human pose estimation in the wild. In *CVPR*, 2018. 10

[6] Kaiwen Guo, Peter Lincoln, Philip Davidson, Jay Busch, Xueming Yu, Matt Whalen, Geoff Harvey, Sergio Orts-Escolano, Rohit Pandey, Jason Dourgarian, et al. The relightables: Volumetric performance capture of humans with realistic relighting. *ACM TOG*, 2019. 8

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3

[8] Yannick Hold-Geoffroy, Akshaya Athawale, and Jean-François Lalonde. Deep sky modeling for single image outdoor lighting estimation. In *CVPR*, 2019. 6

[9] Korrawe Karunratanakul, Sergey Prokudin, Otmar Hilliges, and Siyu Tang. HARP: Personalized hand reconstruction from a monocular RGB video. In *CVPR*, 2023. 1, 5, 6

[10] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014. 4

[11] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *CVPR*, 2017. 10

[12] Yuwei Li, Longwen Zhang, Zesong Qiu, Yingwenqi Jiang, Nianyi Li, Yuexin Ma, Yuyao Zhang, Lan Xu, and Jingyi Yu. NIMBLE: a non-rigid hand model with bones and muscles. *ACM TOG*, 2022. 3, 5, 8

[13] Shanchuan Lin, Linjie Yang, Imran Saleemi, and Soumyadip Sengupta. Robust high-resolution video matting with temporal guidance. In *WACV*, 2022. 10

[14] Gyeongsik Moon. Bringing inputs to shared domains for 3D interacting hands recovery in the wild. In *CVPR*, 2023. 7

[15] Gyeongsik Moon, Takaaki Shiratori, and Kyoung Mu Lee. DeepHandMesh: A weakly-supervised deep encoder-decoder framework for high-fidelity hand mesh modeling. In *ECCV*, 2020. 1

[16] Gyeongsik Moon, Shoou-I Yu, He Wen, Takaaki Shiratori, and Kyoung Mu Lee. InterHand2.6M: A dataset and baseline for 3D interacting hand pose estimation from a single rgb image. In *ECCV*, 2020. 8

[17] Gyeongsik Moon, Hongsuk Choi, and Kyoung Mu Lee. Accurate 3D hand pose estimation for whole-body 3D human mesh estimation. In *CVPRW*, 2022. 5, 7

[18] Gyeongsik Moon, Hongsuk Choi, and Kyoung Mu Lee. NeuralAnnot: Neural annotator for 3D human mesh training sets. In *CVPRW*, 2022. 7

[19] Ahmed AA Osman, Timo Bolkart, and Michael J Black. STAR: Sparse trained articulated human body regressor. In *ECCV*, 2020. 5, 6

[20] Rohit Pandey, Sergio Orts Escolano, Chloe Legendre, Christian Haene, Sofien Bouaziz, Christoph Rhemann, Paul Debevec, and Sean Fanello. Total Relighting: learning to relight portraits for background replacement. *ACM TOG*, 2021. 1

[21] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive body capture: 3D hands, face, and body from a single image. In *CVPR*, 2019. 7

[22] Rolandos Alexandros Potamias, Stylianos Ploumpis, Stylianos Moschoglou, Vasileios Triantafyllou, and Stefanos Zafeiriou. Handy: Towards a high fidelity 3D hand shape and appearance model. In *CVPR*, 2023. 1, 3, 5

[23] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied Hands: Modeling and capturing hands and bodies together. *ACM TOG*, 2017. 3, 5, 8

[24] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 4, 5

[25] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019. 8

[26] Andrey Vakunov, Chuo-Ling Chang, Fan Zhang, George Sung, Matthias Grundmann, and Valentin Bazarevsky. MediaPipe Hands: On-device real-time hand tracking. In *CVPRW*, 2020. 10

[27] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, 2018. 7

[28] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 10

[29] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *CVPR*, 2019. 4