HARP: Personalized Hand Reconstruction from Monocular RGB Videos **Appendix**

A. Datasets

In this section, we describe the details of each dataset used in the experiments. We reiterate that our goal is to propose a scalable and robust system that can create faithful hand avatars given a short video sequence that is captured by commodity hardwares, such as a smartphone. Such setup facilitates the utility of our method in downstream applications, e.g., personalized hand avatar creation for end-users of AR/VR devices. Unfortunately, there is no existing dataset designed and captured for this scenario. Therefore we capture the **Hand Appearance Dataset** with a smartphone in a room with common lighting conditions (e.g., light bulbs on the ceiling).

To demonstrate that HARP is robust to different capture setups, we additionally test HARP on sequences from the **InterHand2.6M [12]** and **HanCo [17]** datasets. To supplement the lack of *accurate* ground truth to evaluate the pose refinement results, we create our **Synthetic Dataset** using a ray tracing engine. The details of each dataset are as follows:

Hand Appearance Dataset. The dataset is partitioned into three parts as described in the main paper. All of the sequences are captured with a hand-held smartphone camera in different conditions. The foreground masks, which include both the hand and arm, are obtained using an off-the-shelf segmentation tool Unscreen [4]. The images are resized to 448x448 pixels, which we use as a default size in all of our experiments unless indicated otherwise.

InterHand2.6M [12]. We demonstrate HARP's ability to create an avatar from existing datasets on sequences from the Interhand2.6M dataset. The sequences in the dataset are captured in a capture dome with a multi-view camera rig and uniform lighting. As the dataset does not include segmentation masks, we obtain foreground-background masks using RVM [9]. We notice that flares from light bulbs in the capture dome often interfere with the segmentation and are sometimes categorized as foreground. We note that such artifact is difficult to remove and could degrade the optimized appearance. To avoid such artifacts, we use 500 frames from *cam400266* of *Capture0/ROM03_RT_No_Occlusion* from the 30-FPS test set. The images are cropped to 334x334 pixels with a hand at the center.

Synthetic Dataset. In order to evaluate the pose refinement results, we create the synthetic dataset with perfect ground truth pose annotations. The images are rendered using the ray tracing engine Cycles in Blender [2]. We leverage the NIMBLE model [7] to obtain the hand meshes and appearances. The appearances are manually selected to ensure diversity in size and skin color from the appearances sampled from NIMBLE. Due to the dependency on NIMBLE, the generated hands are truncated at the wrist and the arm is not visible in the images. For rendering, we use the same Blender settings as the one provided in the demo of NIMBLE. The hand motions are the same in all of the generated sequences. The main differences between each sequence are the viewpoint, the hand shape, and the hand appearance. For each identity, we generate two motions, one is a hand-flipping motion and another one with finger movement. The motions are 5 seconds long at 30 FPS. Fig. A.1 shows the images from our Synthetic dataset.

HanCo [17]. We show the results from the following sequences in Fig. 6 in the main paper: 0 (cam7), 2 (cam1), 10 (cam4), and 27 (cam4). Both sides of the hand are visible in these sequences. The provided foreground-background masks are used as input for the optimization. Nevertheless, note that the dataset is not suitable for avatar creation and should be treated only as a reference due to the low resolution of the image at 224x224 pixels and the unrealistic light stage. Samples images are shown in Fig. A.2.

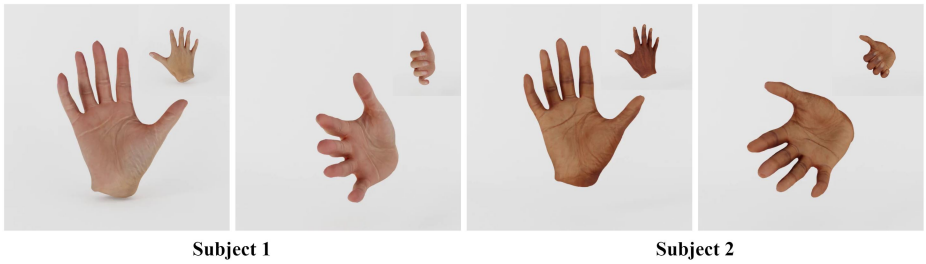

Figure A.1. **Synthetic dataset**. Each image shows the first frame and the middle frame from each sequence.

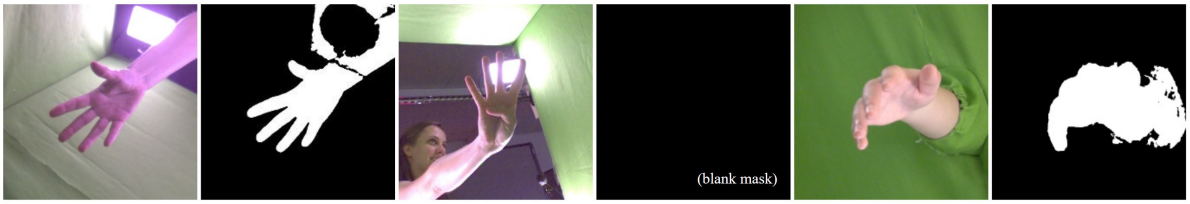

Figure A.2. **Sample images and masks from the HanCo [17] dataset**. The dataset is not suitable for avatar creation because of the low resolution and unrealistic light stage.

B. Baselines

To ensure a fair comparison between HARP and the baseline methods, all of the baselines are optimized at test time, with an equivalent number of epochs when possible. We use the officially released code from each baseline. As some of the baselines are not designed for hand avatar creation via optimization, we need to make adjustments and modifications, which we describe for each baseline in the followings. In the case of HTML [14] and NIMBLE [7], we use the same optimization pipeline as in HARP, replacing only the relevant parts with their models. All the methods take the same images and pose initialization as input.

HTML [14]. We replace the UV texture and normal map of HARP with the texture produced by the HTML model. The HTML texture vector is optimized instead of the HARP texture. As the HTML texture is defined on the surface of MANO [16] hand mesh, we use the MANO template instead of our HARP template. In addition, we allow vertex displacement along normals in the same manner as in HARP. The same optimizers and loss terms are used in the optimization.

NIMBLE [7]. As the NIMBLE model provides both shape and appearance space, we replace the HARP hand geometry and appearance with NIMBLE. The NIMBLE pose, shape, and appearance parameters are updated during the optimization. The same optimizers and loss terms are applied. For initialization, we fit NIMBLE to the METRO prediction instead of our template, using the same formulation as described in Sec. C.2.

S2Hand [1]. We modified the original S2hand code which predicts the appearance from the input image to allow the optimization with photometric losses used in HARP. Because the S2Hand model requires ground truth camera parameters for projecting the predicted mesh onto the image frame, during evaluation we optimize the camera parameters by minimizing the photometric loss with respect to the input image. This optimization is done separately to obtain the best match for each frame. We acknowledge that the optimized texture quality might be affected by the less accurate pose estimation from S2Hand. Nevertheless, due to the fact that S2hand requires ground truth camera intrinsics for image projection but METRO estimates camera extrinsics for a fixed set of intrinsics, it is not possible to optimize S2Hand with our coarse initialization.

Neural Head Avatar [3]. As this method is designed specifically for reconstructing a human head avatar, we make several modifications to adapt it to the hand avatar creation task. Notably, we drop the dependency on face segmentation, landmark, and predicted normals from the model. For segmentation, there are only foreground and background, which are the same as the ones used in other baselines. For landmarks, the landmark locations are replaced with hand key point locations. The predicted normal input is discarded from the model. In terms of implementation, a fixed identity is used in place of the unavailable input. For the geometry, we replace the FLAME [6] model with our hand template with arm (details in Sec. C.1). We follow the official code and instructions for training and evaluating the results.

C. Implementation Details

C.1. Hand Templates

In order to create a hand avatar, we observe that the truncation at the wrist in the MANO [16] model is problematic to the appearance-creation process and does not reflect the reality that a hand is always attached to an arm. Thus, we implement a version of the hand model with an arm, which we derive from SMPLX [13] by truncating the mesh at the elbow, moving the root joint to the right-hand wrist, and linearly subdividing the mesh once. As a result, our hand model has two modes: hand-only and hand-with-arm, which can be used interchangeably depending on the available mask. The comparison between the template meshes is shown in Fig. C.1.

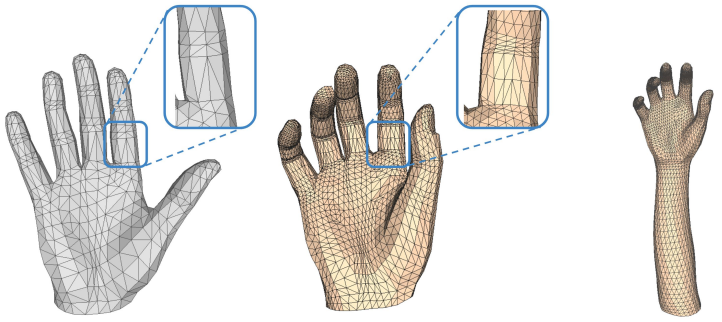

Figure C.1. Template Meshes.

C.2. Initialization

Mesh Initialization. As discussed in the main paper, before we start the optimization process, the hand pose and shape need to be initialized. Therefore, we employ the pose estimator METRO [8] which estimates the camera translations with fixed intrinsic parameters. We average the camera translations across all frames from the same sequence and fix them throughout the optimization. However, because the METRO model predicts the mesh vertex locations directly, the predicted meshes cannot be used for animation. To this end, we fit the hand template to METRO predictions by optimizing pose parameters γ and shape parameters β using the following energy term E:

$$E = \sum_{v}^{V} \|v_t - v_p\|^2$$
$$\text{where } V_t = \mathcal{M}(\gamma, \beta)$$

where V is the set of MANO vertices in the template, and v_p is the predicted location from METRO. Note that this optimization is possible because METRO prediction and MANO model share the same template mesh. To avoid local minima during fitting, we re-run the optimization process if the mean distance between METRO vertices and the optimized vertices is more than 1 cm.

C.3. Optimization

From the initialization, we first optimize using only the geometry term E_{geo} to reconstruct the hand surface. We then jointly optimize both the geometry and the appearance with the addition of E_{app}. Once the geometry is stable in the joint optimization, we then freeze the geometry optimization and continue to refine the appearance with only E_{app}. Concretely, to obtain the personalized hand pose, shape, and appearance, we employ a multi-stage optimization scheme as follows: (1) geometry optimization, (2) both geometry and appearance optimization, and (3) only appearance optimization. We use the Adam [5] for both E_{geo} and E_{app} optimization. In total, the optimization takes an average of 80 minutes on a single Nvidia 3090 GPU.

Geometry Optimization. Given the masked images, we first optimize the pose γ, shape β, vertex displacements D, translations, and rotations, with respect to the geometry objective E_{geo}. In this stage, only the geometry energy term E_{geo} is used. loss is used. We optimize using a learning rate of $1e^{-3}$ for 100 epochs.

Joint Optimization. After the coarse geometry alignment, we begin the appearance optimization with respect to the appearance objective E_{app}. In this stage, both the geometry objective E_{geo} and appearance objective E_{app} are optimized together for 50 epochs to correct geometry misalignment using appearance information on the input images. We use the learning rate of $1e^{-2}$ for the appearance optimizer.

Appearance Optimization. Lastly, we refine the appearance with only the appearance objective E_{app} for another 50 epochs. This step focuses on retrieving fine texture details which are difficult to optimize while the geometry is still changing.

Appearance Regularization Terms. To regularize the reconstruction of the UV texture and normal map, we define the appearance regularization term in the UV space. Let $\mathcal{T}$ be an albedo map and $\mathcal{G}$ be a normal map, and I is a pixel in the UV space:

$$E_{app\_reg} = E_{t\_reg} + E_{n\_reg}, \tag{1}$$

$$E_{t\_reg} = \sum_{I} \frac{1}{3} \left\| \mathcal{T}(I) - \mathcal{T}(I + \epsilon_1) \right\|_1, \tag{2}$$

$$E_{n\_reg} = \sum_{I} \frac{1}{3} (\left\| \mathcal{G}(I) - \mathcal{G}(I + \epsilon_2) \right\|_1 + \left\| \mathcal{G}(I) - u_z \right\|_2^2), \tag{3}$$

where ϵ_1, ϵ_2 are random pixel-space displacements sampled from a Gaussian with a standard deviation of 2, u_z is a unit vector pointing along z direction. Both terms ensure a smooth transition in the UV space, while the E_{n_reg} encourages the normal to be close to the surface normal.

Losses. The weights for each energy term are as defined in the table C.1:

E_{sil}	7.0
E_{init}	10.0
E_{verts}	2.0
E_{lap}	4.0
E_{norm}	0.1
E_{arap}	0.2
E_{photo}	1.0
E_{vgg}	1.0
E_{t_reg}	2.0
E_{n_reg}	0.5

Table C.1. **Weights for each energy term.**

C.4. Lighting Contribution

As we assume that the hand surface is largely non-reflective, we ignore the specular contribution in our lighting formulation. In our method, the color at each surface point is only affected by the ambient contribution, which dictates how bright each point is regardless of its position, and the diffuse contribution, which determines the brightness based on the angle between the point normal and the light direction. Diffuse lighting is also affected by the visibility term V that determines direct occlusion with respect to the light source. A higher ambient contribution will make the shadow less visible and the brightness more uniform. Figure C.2 shows the decomposition of each component in our pipeline. We show the differences between the final albedo after optimizing with and without considering the visibility term V in Fig. C.3.

In our experiments, we empirically disable the self-shadowing term of HARP when we compare it with the baselines on the first part of our hand appearance dataset, where there is no hard shadow and dominant light source. In other experiments, the self-shadowing term is enabled and the ratio between the ambient and diffuse contributions is optimized together with other parameters as described in the main paper.

D. Ablation

In this section, we discuss the importance and effect of each energy term in our method.

Appearance. We observe that, without the perceptual term E_{VGG}, the resulting texture looks overly smooth as the colors are averaged over the pixels that map to a slightly different point on the hand surface. However, without the photometric L1 term E_{photo}, the result might contain noisy artifacts. The qualitative comparison is shown in Table D.1.

Geometry. Figure D.1 shows the qualitative comparison between the results from optimizing without a specific shape regularization term. The as-rigid-as-possible term E_{arap} prevents sharp edges when the mesh is deformed to fit the silhouette.

Figure C.2. **Decomposition of color contributions in our rendering pipeline.**

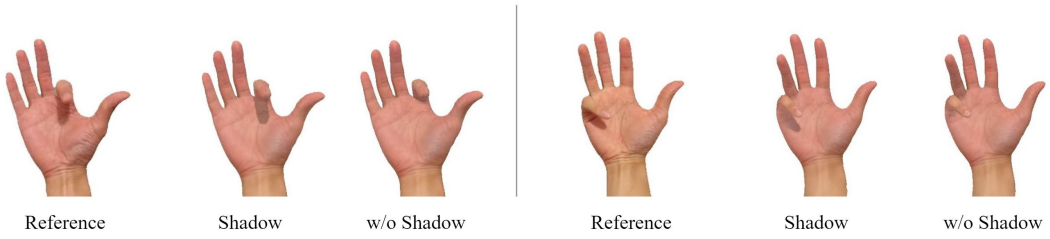

Figure C.3. **Comparison between optimization without and with self-shadowing effect.** Without considering self-shadow, the input images cannot be faithfully reconstructed by the renderer.

	L1 ↓	LPIPS ↓	MS-SSIM ↑
w/o E_{photo}	0.0171	0.0693	0.906
w/o E_{VGG}	0.0164	0.0842	0.914
HARP	0.0168	0.0712	0.908

Table D.1. **Ablation study on the appearance losses.**

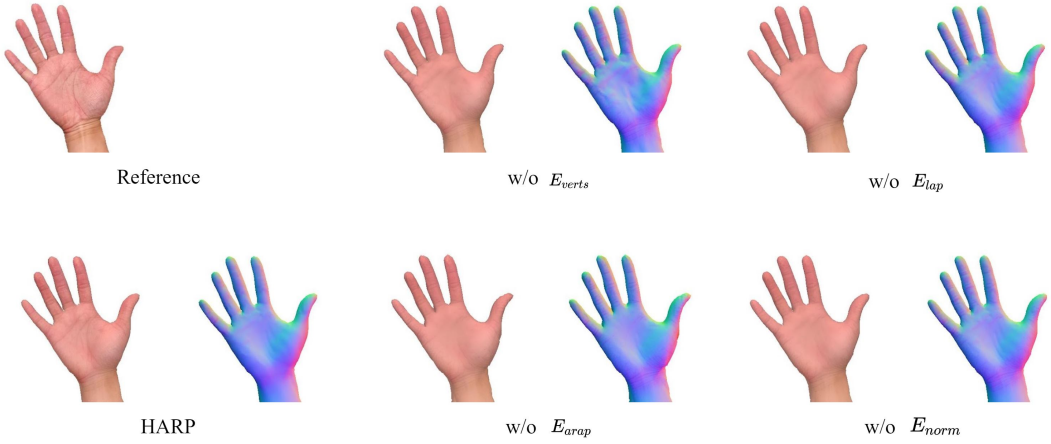

Figure D.1. **Qualitative comparison between optimization results using different geometry regularization terms.**

The vertex displacement term E_{verts} ensures that the deviation from the shape space of the underlining parametric model is minimal, such that the blendshape from the parametric model is still useful when the mesh is reposed. The other terms including the normal consistency regularization E_{norm} and the Laplacian regularization E_{lap} encourage the surface to be more smooth and less bumpy. We note that the effect of each term is less noticeable in the rendered image evaluation as the optimization can counteract the geometry change with a texture change. However, the regularization terms are necessary to ensure mesh integrity for any downstream application.

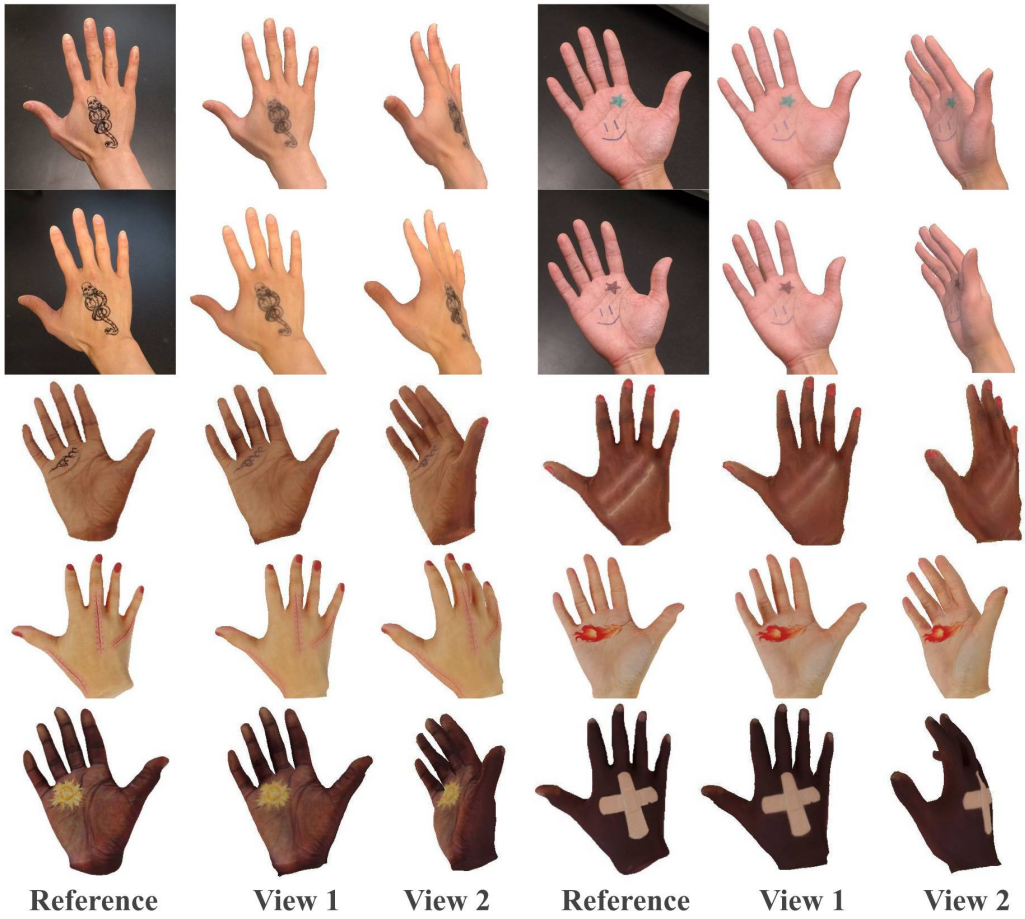

Figure E.1. **Qualitative results of out-of-distribution hands with different lighting conditions.** Our method can accurately capture diverse appearances. The first two rows are captured in different lighting conditions with a different number of ceiling lights. The last three rows are synthetic data rendered with three light sources and ambient light. Please zoom in for details.

E. Discussion

E.1. Additional Results

We show additional results in Fig. E.1 that demonstrate our method's ability to capture diverse patterns such as tattoos, nail colors, and scars faithfully on diverse skin colors.

E.2. Pose-dependent surface deformation

As our model is built on top of MANO, it has the pose-dependent surface deformation modeled by MANO pose blend shapes. Our displacement map is designed to capture detailed, personalized hand shapes that cannot be represented by MANO. We found that conditioning the displacement map on poses results in two different sets of parameters governing the same surface deformation which could be difficult to optimize.

E.3. Failure Cases

In this section, we discuss the noticeable failure cases of our system. The examples are shown in Fig. E.2. First, HARP mainly uses the silhouette from a monocular view to guide the personalized geometry. As a consequence, it is crucial that the images and the masks provide sufficient information about the shape. When the data is not sufficient, the hand mesh can deform in an unexpected way to satisfy the mask. We show this failure case in Fig. E.2(a) where some vertices extend perpendicular to the palm as there are not enough side view images. Note that the optimization can always compensate for the bumpy geometry with a change in the texture in order to replicate the input images. A potential solution to this problem

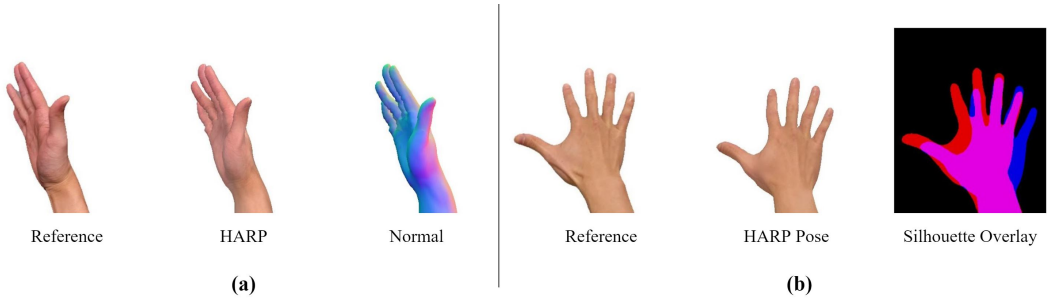

Figure E.2. **Failure cases and limitations.** (a) With limited pixel information, the geometry could deform in an undesirable way as it uses mainly the silhouette for supervision. (b) HARP pose optimization is sensitive to initialization and could stuck in local minima when the initialization and foreground mask do not align. In this case, a large immediate increase in silhouette loss will prevent the optimizer from leaving the minima.

is to vary the geometry regularization terms based on the characteristic of the hand in the video. Second, as our method relies on the initialization from a hand pose estimator and the foreground mask, the final pose and the appearance quality are influenced by the performance of the pose estimator and the segmentation tool. In some cases, the pose might be stuck in a local minimum due to the initialization (Fig. E.2(b)).

E.4. Pose Refinement via Appearance Optimization

In the main paper, we show that by optimizing the hand pose parameters with HARP, we can refine the estimated hand pose to better fit the image, which can lead to a slight improvement in the Procrustes-aligned hand pose error. Our intuition is that if the hand's appearance is known in advance, it should be possible to leverage pixel color optimization to obtain more accurate poses. We compare the results between (1) the initial estimation, (2) HARP with only geometry term E_{geo} (HARP-sil), (3) normal HARP (HARP-full), and (4) HARP with known appearance (HARP-known).

Case (2) is a known task that is often associated with a differentiable renderer [10, 15] where the silhouette is used to optimize for an object pose. However, we observe that for a highly articulated object such as a hand, using silhouette alone might not be enough to obtain the correct pose. We visualize such scenarios in Fig. E.3.

Case (3) leverages only the appearance consistency within the optimized video. As both the poses and the appearance are optimized together, it is possible to obtain the colors that are associated with wrong poses.

Case (4) leverages the appearance that is obtained from possibly easier hand motion. All of the hand parameters, except for hand poses, are given as initialization. Those parameters, including the appearance, are obtained from running HARP on another sequence. The given parameters are frozen during the optimization and only the hand poses are updated. All loss terms are the same as normal HARP.

We demonstrate that such pose refinement is possible if the appearance consistency is leveraged in the optimization (both case 3 and case 4). We acknowledge that our synthetic dataset is small relative to the recent hand pose dataset such as InterHand2.6M [12] However, due to the lack of ground truth with accurate 3D annotations, we could only perform the experiment on our synthetic dataset where we have **perfectly accurate ground truth**. The InterHand2.6M dataset [12], which offers the hand motions that are the closest to our target use case, reported the MANO ground truth fitting error at around 5 mm [11]. On the other hand, the Procrustes-aligned MANO vertex error of the METRO [8] prediction on our selected sequence is at 6 mm. Any quantitative improvement below 1 mm would be statistically meaningless as it is an order of magnitude smaller than the supposed ground truth error. Therefore, we do not report the pose refinement on this dataset and other existing datasets due to similar reasons.

Future work. We foresee that the ideal scenario for this use case would be when a user starts using AR/VR equipment, they do a hand-flipping motion to provide a hand appearance. And with that appearance, the pose estimation can be improved. Practically, however, the optimization speed would still prevent real-time pose refinement. As such improving the speed and pose estimation error which would be interesting to explore in future work.

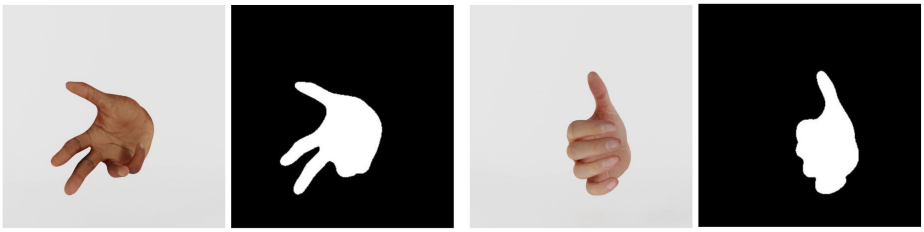

Figure E.3. Example cases where the hand mask is not informative enough for determining the hand pose.

References

[1] Yujin Chen, Zhigang Tu, Di Kang, Linchao Bao, Ying Zhang, Xuefei Zhe, Ruizhi Chen, and Junsong Yuan. Model-based 3d hand reconstruction via self-supervised learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10451–10460, 2021. 2

[2] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. 1

[3] Philip-William Grassal, Malte Prinzler, Titus Leistner, Carsten Rother, Matthias Nießner, and Justus Thies. Neural head avatars from monocular rgb videos. *arXiv preprint arXiv:2112.01554*, 2021. 2

[4] Kaleido AI GmbH. Unscreen. February 2021. 1

[5] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[6] 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), 2017. 2

[7] Yuwei Li, Longwen Zhang, Zesong Qiu, Yingwenqi Jiang, Yuyao Zhang, Nianyi Li, Yuexin Ma, Lan Xu, and Jingyi Yu. Nimble: A non-rigid hand model with bones and muscles. *arXiv preprint arXiv:2202.04533*, 2022. 1, 2

[8] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. In *CVPR*, 2021. 3, 7

[9] Shanchuan Lin, Linjie Yang, Imran Saleemi, and Soumyadip Sengupta. Robust high-resolution video matting with temporal guidance, 2021. 1

[10] Shichen Liu, Tianye Li, Weikai Chen, and Hao Li. Soft rasterizer: A differentiable renderer for image-based 3d reasoning. *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. 7

[11] Gyeongsik Moon and Kyoung Mu Lee. Neuralannot: Neural annotator for in-the-wild expressive 3d human pose and mesh training sets. *arXiv preprint arXiv:2011.11232*, 2020. 7

[12] 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 *European Conference on Computer Vision (ECCV)*, 2020. 1, 7

[13] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019. 2

[14] Neng Qian, Jiayi Wang, Franziska Mueller, Florian Bernard, Vladislav Golyanik, and Christian Theobalt. HTML: A Parametric Hand Texture Model for 3D Hand Reconstruction and Personalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, 2020. 2

[15] 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. 7

[16] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), 2017. 2

[17] Christian Zimmermann, Max Argus, and Thomas Brox. Contrastive representation learning for hand shape estimation. In *arxive*, 2021. 1, 2