Weakly-supervised Domain Adaptation via GAN and Mesh Model for Estimating 3D Hand Poses Interacting Objects– Supplemental Material

Seungryul Baek
Imperial College London
s.baek15@imperial.ac.uk

Kwang In Kim
UNIST
kimki@unist.ac.kr

Tae-Kyun Kim
Imperial College London
tk.kim@imperial.ac.uk

In this supplemental material, we provide

1. details of the network architectures, mesh renderer g^{MR}, evaluating the estimated skeletons, and localizing hands in images (Sec. 1);
2. a summary of the overall training process (Sec. 2);
3. a description of the algorithms that are compared on the *HO3D* dataset (these participated in HANDS 2019 Challenge; Sec. 3);
4. additional hand-only image restoration results (Sec. 4).

1. Implementation details

Network architectures. Tables 1–4 present architectures of the sub-networks of our domain adaptation network. The 2D feature and pose estimator g^{FPE} receives an input RGB image of size 256×256 and generates 21 32×32-sized 2D heatmaps $\mathbf{h}$ and 128 32×32-sized 2D feature maps $\mathbf{f}$: $\mathbf{h}$ and $\mathbf{f}$ are sampled at layers 31 and 30 of g^{FPE}, respectively (Table 1). Supervision on 2D heatmaps are provided at three different layers (31, 24, and 17) enabling gradual refinement of the estimated 2D heatmaps.

Our mesh renderer g^{MR} combines four sub-networks: hand mesh estimator g^{HME}, texture estimator g^{Tex}, neural renderer g^{NR}, and hand joint regressor g^{Reg}. g^{HME} and g^{Tex} share a common feature extractor which converts input 2D maps $\mathbf{f}$ and $\mathbf{h}$ to a 1,152-dimensional feature vector $\mathbf{k}$ by applying convolution layers to each 32×32-sized 2D map (see Table 3). Given feature vector $\mathbf{k}$, g^{HME} estimates the 63-dimensional MANO parameter vector $\mathbf{p}$ using iterative regression [9, 1]. The output hand mesh $\mathbf{m}$ is then constructed by applying the MANO layer g^{MANO} [1, 3, 7] to $\mathbf{p}$. The texture estimator g^{Tex} converts $\mathbf{k}$ to a 4,614-dimensional texture vector encoding RGB colors of 1,538 3D mesh faces.

Details of the component functions of mesh renderer g^{MR}. Our hand mesh estimator $g^{\text{HME}}: F\times H\rightarrow M$ uses the MANO parameterization $\mathbf{p}\in\mathbb{R}^{63}$ of Baek et al. [1]: $\mathbf{p}$ consists of 55-dimensional principal component analysis (PCA) shape parameters (10 for shape and 45 for articulation) and 8-dimensional camera parameters (global rotation represented by a 4-dimensional quarternion vector, one parameter for global scaling, and 3 for 3D translation).

g^{HME} estimates $\mathbf{p}$ by first converting 2D maps $\{\mathbf{f},\mathbf{h}\}\subset\mathbb{R}^{32\times32}$ to a 1,152-dimensional feature vector $\mathbf{k}$: We apply the standard convolution layers to each 32×32-sized 2D map, eventually reducing the spatial resolution to 1×1 while increasing the number of feature channels to 1,152, leading to a $1152\times1\times1$ feature array.

Based on $\mathbf{k}$, adopting the robust parameter optimization approach of [9, 1], we iteratively estimate $\mathbf{p}$ by first constructing an initial estimate $\mathbf{p}(0)$ and then refining it by recursively performing regression on the parameter offset $\Delta\mathbf{p}$:

$$\mathbf{p}(t+1)=\mathbf{p}(t)+\Delta\mathbf{p}(t), \tag{1}$$

where the number of total iterations is fixed at 3 similarly to [9, 1]. The differentiable MANO layer g^{MANO} (as a component of g^{HME}) then converts the resulting $\mathbf{p}$ to a MANO mesh consisting of 778 vertices and 1,538 faces [1]: g^{MANO} first restores the mesh shape by combining the MANO PCA basis vectors using the first 55 components of $\mathbf{p}$, then it performs linear blend skinning using pose vectors, and globally rotates, scales, and translates the resulting meshes with camera parameters of $\mathbf{p}$. All operations of g^{MANO} are differentiable and as a fixed function, g^{MANO} does not have any trainable parameters.

As in g^{HME}, the texture estimator $g^{\text{Tex}}: F\times H\rightarrow T$ receives $\mathbf{k}$ and estimates 3-dimensional color vectors for each face out of 1,538 in the MANO model generating a 4,614-dimensional feature vector $\mathbf{t}$ (see [10] for details).

For the 3D skeleton regressor $g^{\text{Reg}}: F\times H\rightarrow Y$, we adopt the skeleton regressor provided by the authors of the original MANO model [16], and augment it by manually adding 5 finger tips similarly to [1, 7], as these joints are not provided in the MANO model. As discussed shortly, supervision is exercised to g^{Reg} based on hand-only images provided with skeletal annotations. Since g^{Reg} is differentiable, the resulting error can be *propagated back* to g^{HME} and g^{Tex}. For the neural renderer $g^{\text{NR}}: F\times H\rightarrow X$, Kato et al.'s neural render is employed [10]. Both g^{NR} and g^{Reg} are held fixed throughout the training of DAN.

Generation of 2D segmentation masks. The three synthetic datasets that we use for training (*RHD*, *SH* and *Obman*) are provided with 2D foreground segmentation masks.

Table 1: Architecture of 2D feature and pose estimator g^{FPE}.

Layer	Operation	Kernel	Dimensionality
	Input image	-	$256\times256\times3$
1	Conv. + ReLU	3×3	$256\times256\times64$
2	Conv. + ReLU	3×3	$256\times256\times64$
3	Max Pool	4×4	$128\times128\times64$
4	Conv. + ReLU	3×3	$128\times128\times128$
5	Conv. + ReLU	3×3	$128\times128\times128$
6	Max Pool	4×4	$64\times64\times128$
7	Conv. + ReLU	3×3	$64\times64\times256$
8	Conv. + ReLU	3×3	$64\times64\times256$
9	Conv. + ReLU	3×3	$64\times64\times256$
10	Conv. + ReLU	3×3	$64\times64\times256$
11	Max Pool	4×4	$32\times32\times256$
12	Conv. + ReLU	3×3	$32\times32\times512$
13	Conv. + ReLU	3×3	$32\times32\times512$
14	Conv. + ReLU	3×3	$32\times32\times512$
15	Conv. + ReLU	3×3	$32\times32\times512$
16	Conv. + ReLU	3×3	$32\times32\times512$
17	Conv.	1×1	$32\times32\times21$
18	Concat(16, 17)	–	$32\times32\times533$
19	Conv. + ReLU	7×7	$32\times32\times128$
20	Conv. + ReLU	7×7	$32\times32\times128$
21	Conv. + ReLU	7×7	$32\times32\times128$
22	Conv. + ReLU	7×7	$32\times32\times128$
23	Conv. + ReLU	7×7	$32\times32\times128$
24	Conv.	1×1	$32\times32\times21$
25	Concat(16, 17, 24)	–	$32\times32\times554$
26	Conv. + ReLU	7×7	$32\times32\times128$
27	Conv. + ReLU	7×7	$32\times32\times128$
28	Conv. + ReLU	7×7	$32\times32\times128$
29	Conv. + ReLU	7×7	$32\times32\times128$
30	Conv. + ReLU	7×7	$32\times32\times128$
31	Conv.	1×1	$32\times32\times21$

Table 2: Architecture of GAN generator g^{GAN}.

Layer	Operation	Kernel	Dimensionality
	Input feature	-	$32\times32\times298$
1	Conv.+ReLU	3×3	$32\times32\times298$
2	Conv.+ReLU	3×3	$32\times32\times298$
3	Residual layer	–	–
4	Conv.+ReLU	3×3	$32\times32\times298$
5	Conv.+ReLU	3×3	$32\times32\times298$
6	Residual layer	–	–
7	Conv.+ReLU	3×3	$32\times32\times298$
8	Conv.+ReLU	3×3	$32\times32\times298$
9	Residual layer	–	–
10	Conv.+ReLU	3×3	$32\times32\times298$
11	Conv.+ReLU	3×3	$32\times32\times298$
12	Residual layer	–	–
13	ConvTansPose2D + ReLU	4×4	$32\times32\times298$
14	ConvTansPose2D + ReLU	4×4	$64\times64\times128$
15	ConvTansPose2D + ReLU	4×4	$128\times128\times32$
16	ConvTansPose2D + ReLU	7×7	$256\times256\times3$

Table 3: Architecture of the feature extraction network that converts 2D maps $\mathbf{h}$,$\mathbf{f}$ to the corresponding feature vector $\mathbf{k}$.

Layer	Operation	Kernel	Dimensionality
	Input image	-	$32\times32\times149$
1	Conv.+ReLU	3×3	$32\times32\times32$
2	Conv.+ReLU	3×3	$15\times15\times32$
3	Conv.+ReLU	3×3	$15\times15\times64$
4	Conv.+ReLU	3×3	$7\times7\times64$
5	Conv.+ReLU	3×3	$7\times7\times128$
6	Conv.+ReLU	3×3	$3\times3\times128$
7	Conv.+ReLU	3×3	$1\times1\times1152$

For real-world datasets (*STB* and *CORe50*), we generate such segmentation masks based on the accompanying depth maps or 3D skeleton annotations: *STB* provides 3D skeleton annotations from which 3D bounding boxes can be retrieved. Then, 2D segmentation masks are obtained by first removing the point clouds that lie outside these boxes and then, projecting the remaining point clouds onto the image pane.

For *CORe50*, we generate segmentation masks for 11,053 video frames, out of total $\approx$ 150,000 frames capturing 11 subjects interacting with 50 objects: First, we sample every 5 frames, as consecutive frames have very similar appearances. Then, we sample only right-handed subjects (6 out of 11) and remove video sequences that contain 'scissors', 'remote controls' and 'glasses' objects as large fractions of frames in these sequences show only objects (without hands). Finally, 2D segmentation masks are generated by thresholding the values of each depth map, and manually removing noisy segmentation masks. Figure 1 shows the segmentation masks generated for hand images sampled from the testing sets. Here, we also show the corresponding hand-only images $\mathbf{x}''$ synthesized by our GAN generator g^{GAN}. These example shows that even when the original segmentation masks are noisy, our GAN generator can accurately restore hand-only parts of the input images.

Hand localization in images. Our domain adaptation network takes a *cropped* hand image as input, i.e. it assumes that all hands in the input images are scale-normalized and centered. As this might not be the case in real-world applications, we explicitly crop hands by estimating their bounding boxes.

At training, such bounding boxes are determined based on 2D segmentation masks or ground-truth skeletons. When the dataset at hand provides ground-truth skeletons (e.g. for hand-only images), we decide the center and size of each bounding box, respectively as the middle finger's metacarpophalangeal (MCP) joint and 1.5 times the size of the tight bounding box of skeleton joints within the image pane. When the dataset provides 2D segmentation masks

Table 4: Architectures of GAN discriminators d_1^{GAN} and d_2^{GAN}.

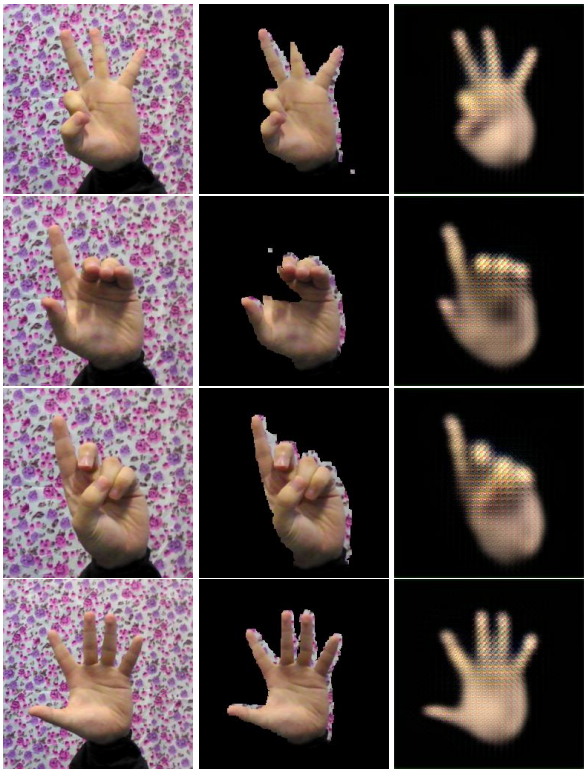
Layer	Operation	Kernel	Dimensionality
	Input image	-	$256\times256\times3$
1	Conv.+ReLU	4×4	$128\times128\times32$
2	Conv.+ReLU	4×4	$64\times64\times64$
3	Conv.+ReLU	4×4	$32\times32\times128$
4	Conv.+ReLU	4×4	$16\times16\times256$
5	Conv.+ReLU	4×4	$8\times8\times512$
6	Conv.+ReLU	4×4	$4\times4\times1024$
7	Conv.+ReLU	4×4	$1\times1\times1024$
8	Conv.+ReLU	4×4	$1\times1\times1$
9	Sigmoid	–	–

instead of skeletons, the location and size of each box is decided as the center and 1.5 times the size of the tight bounding box of the corresponding mask.

At testing, for all images except for these in *HO3D*, we explicitly localize hands using Zimmermann and Brox's hand detection algorithm [20]. These initial bounding box estimates are refined via temporal smoothing over 5 consecutive frames. For *HO3D*, we use the accompanying bounding box annotations.

Evaluation of hand pose estimate. The output $\mathbf{y}$ of our domain adaptation network represents the locations of 21 skeleton joints in normalized coordinates: All coordinate values are bounded in $[0,1]$ and a specific joint (referred to as *root joint*) is located at the origin $[0,0,0]^\top$. To assess the accuracy, these estimated coordinate values need to be mapped to the absolute coordinate system in which the ground-truth skeleton joints are annotated. For *HO3D*, we use 'palm' joints provided by the organizers of HANDS 2019 Challenge [4] as the root joint: For each estimated skeleton, the palm joint is originally set as $[0,0,0]^\top$, which is then translated to the corresponding ground-truth palm location in absolute coordinates. The scale is restored by inverse normalizing the bounding boxes that are estimated during hand localization. Inverse normalizing the estimated hand skeletons in *STB* is similar, but here, we use MCP joint of the middle finger as the root joint.

For *DO* and *ED* datasets, the depth values of the root joints are not provided and therefore, we first restore these values adopting Iqbal et al.'s approach [8]: We calculate the root depth assuming that for each skeleton, the bone-length joining MCP of the index finger and palm is 1. Thereafter, we restore the absolute scales based on the average bone-lengths calculated from *STB*'s skeleton data: We match the lengths of 20 sampled bones in the estimated hand skeleton to the corresponding average bone lengths in the *STB* dataset. See [8, 1] for details.



(a) (b) (c)

Figure 1: Example foreground segmentation masks. (a) original images $\mathbf{x}$, (b) segmented images $\mathbf{x}\odot\mathbf{s}$ generated based on mask annotations $\mathbf{s}$, (c) hand images $\mathbf{x}''$ restored by the GAN generator.

2. Training process

Our 2D feature and pose estimator g^{FPE} is initialized using the weights of Zimmermann and Brox's [20] PoseNet. The other networks are randomly initialized. These initialized networks are then trained by running 100 epochs of gradient descent on the loss L with learning rate fixed at 10^{-5}:

$$L = L_{\text{Heat}} + L_{\text{Pos}} + L_{\text{Img}} + L_d, \tag{2}$$

where

$$L_{\text{Heat}}([g^{\text{FPE}}]_H|D_{\text{Hand}}) = \|g^{\text{FPE}}(\mathbf{x}) - \mathbf{h}_{\text{GT}}\|_2^2, \tag{3}$$

$$\begin{aligned} &L_{\text{Img}}(g^{\text{FPE}}, g^{\text{HME}}, g^{\text{Tex}}|D) \\ &= \sum_{i=1}^{2} \mathbb{E}[\log(1-d_i^{\text{GAN}}(\mathbf{x}''))] + \mathbb{E}[\log(1-d_i^{\text{GAN}}(\mathbf{x}'))] \\ &\quad + \|\mathbf{x}'' - \mathbf{x}\odot\mathbf{s}_{\text{Hand}}\|_1 + \|\mathbf{x}' - \mathbf{x}\odot\mathbf{s}_{\text{Hand}}\|_1, \end{aligned} \tag{4}$$

$$L_{\text{Pos}}(g^{\text{FPE}}, g^{\text{HME}}|D_{\text{Hand}}) = \|[g^{\text{MR}}(g^{\text{FPE}}(\mathbf{x}))]_Y - \mathbf{y}_{\text{GT}}\|_2^2, \tag{5}$$

$$\begin{aligned} L_d(d^{\text{GAN}}|D) = &-\mathbb{E}[\log([d_2^{\text{GAN}}(\mathbf{x}\odot\mathbf{s}_{\text{Hand}})])] \\ &-\mathbb{E}[\log(1-[d_2^{\text{GAN}}(\mathbf{x}\odot\mathbf{s}_{\text{HOI}})])] \\ &-\mathbb{E}[\log([d_1^{\text{GAN}}(\mathbf{x}\odot\mathbf{s}_{\text{Hand}})])] - \mathbb{E}[\log(1-[d_1^{\text{GAN}}(\mathbf{x}'')])]. \end{aligned} \tag{6}$$

Algorithm 1 summarizes the overall training process.

3. Baseline approaches and existing datasets

The *HO3D* dataset was originally used in the Task 3 of the HANDS 2019 Challenge [4]. As we use the experimental settings configured for this challenge, our results can be directly compared with the results of the algorithms that participated in this challenge (Table 3 in the main paper). The three best results were achieved by the participants with IDs *potato*, *Nplwe*, and *lin84*.

The algorithm of *potato* is based on Iqbal et al.'s framework [8] which uses a latent depth map generation module helping lift 2D skeleton heatmap responses to 3D maps. Note that we evaluate and compare with [8] on *ED* and *DO*. *Nplwe*'s algorithm bases on LCRNet++ [15]. This jointly performs bounding box localization, (discrete) pose classification, and regression of skeletal joints, offering state-of-the-art performance in 3D human pose estimation. The algorithm of *lin84* is based on the encoder-decoder architecture of Yang et al. [18]. By exploiting the data generation capability of the MANO 3D hand model [16], this algorithm generates intermediate RGB images and depth maps, and uses them as a prior helping refine their encoder latent space.

Existing datasets for RGB-based 3D hand pose estimation. Collecting quality 3D pose annotations of real RGB images under the HOI scenarios is challenging due to e.g. occlusions. A complete and automatic pipeline for annotating 3D joint locations for severely occluded hands does not exist. It either requires much manual effort to continuously check and refine the labels [21] or the use of magnetic sensors [5]/data gloves [2] that corrupts RGB images. Alternatively, they resort to synthetic data.

In Table 5, we present several datasets popularly used for training RGB-based 3D hand pose estimator. The Table shows that most large-scale datasets in the hand pose estimation community (e.g. *RHD* [20], SynthHands (*SH*) [13], GANerated [12] and *Obman* [7]) are synthetic. Real datasets have either limited annotations such as discrete grasp types (e.g. GUN-71 [14]), only 5 finger tips (e.g. *DO* [17], EGO [13]) or limited number of frames (e.g. around 10K frames for HOI in *HO3D* [6]). FPHA [5] dataset is real and fair-sized; however their RGB frames are corrupted since the magnetic sensors used are visible. FreiHand [21] is of the latest benchmark having a moderate-scale (35k). However, less than half of it contains HOI images. Considering diverse objects, backgrounds and large hand pose space combined, far more samples are needed. In [7], authors reported the accuracy of hand pose estimator trained and tested using either hand-only or HOI data. When the hand pose estimator is trained by HOI, it does not perform well on hand-only testing images in comparison to the model trained by hand-only, though increasing the accuracy on HOI testing images. Note that our algorithm uses hand-only images in the *RHD*, *SH* and *STB* datasets, unlabeled real-world HOI images from the *CORe50* dataset, and synthetic HOI image pairs from the *Obman* dataset.

4. Additional examples

Figures 2–4 shows example hand-only restoration and skeleton estimation results: By employing the MANO model and GAN generators, and iteratively enforcing consistency of the final mesh reconstruction over 2D maps, our algorithm often faithfully recovers the hand-only counterpart of the input HOI image. Furthermore, even for challenging cases where hand restorations are inaccurate, by fully exploiting them in combination with features extracted via our 2D feature and pose estimator, our method can estimate accurate or reasonably accurate final skeletons (Fig. 5).

References

[1] Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim. Pushing the envelope for RGB-based dense 3D hand pose estimation via neural rendering. In *CVPR*, 2019. 1, 3

[2] Abhishake Kumar Bojja, Franziska Mueller, Sri Raghu Malireddi, Markus Oberweger, Vincent Lepetit, Christian Theobalt, Kwang Moo Yi, and Andrea Tagliasacchi. HandSeg: An automatically labeled dataset for hand segmentation from depth images. In *Conference on Computer and Robot Vision (CRV)*, 2019. 4

[3] Adnane Boukhayma, Rodrigo de Bem, and Philip H. S. Torr. 3D hand shape and pose from images in the wild. In *CVPR*, 2019. 1

[4] ICCV HANDS2019 Challenge. https://sites.google.com/view/hands2019/challenge, (accessed 15 Nov. 2019). 3, 4

[5] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with RGB-D videos and 3D hand pose annotations. In *CVPR*, 2018. 4, 5

[6] Shreyas Hampali, Markus Oberweger, Mahdi Rad, and Vincent Lepetit. HO-3D: A multi-user, multi-object dataset for joint 3D hand-object pose estimation. In *ArXiv*, 2019. 4, 5

[7] Yana Hasson, Gül Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J. Black, Ivan Laptev, and Cordelia Schmid. 3D hand shape and pose estimation from a single RGB image. In *CVPR*, 2019. 1, 4, 5

[8] Umar Iqbal, Pavlo Molchanov, Thomas Breuel, Juergen Gall, and Jan Kautz. Hand pose estimation via latent 2.5D heatmap regression. In *ECCV*, 2018. 3, 4

[9] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *CVPR*, 2018. 1

[10] Hiroharu Kato, Yoshitaka Ushiku, and Tatsuya Harada. Neural 3D mesh renderer. In *CVPR*, 2018. 1

[11] Vincenzo Lomonaco and Davide Maltoni. CORe50: a new dataset and benchmark for continual object recognition. In *Proceedings of the 1st Annual Conference on Robot Learning (PMLR)*, 2017. 5

[12] Franziska Mueller, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt. GANerated hands for real-time 3D hand tracking from monocular RGB. In *CVPR*, 2018. 4, 5

Algorithm 1: Training process

Input:
–Training data $D = [D^{\mathrm{R}}_{\mathrm{Hand}}, D^{\mathrm{S}}_{\mathrm{Hand}}, D^{\mathrm{R}}_{\mathrm{HOI}}, D^{\mathrm{S}}_{\mathrm{paired}}]$, $D^{\mathrm{R}}_{\mathrm{HOI3D}}$.
–MANO model: PCA shape basis; mean pose vector;
–Hyper-parameters: number T_1, T_2 of epochs, size N' of mini-batch;
Output: (Weights of)
–Feature and pose estimator g^{FPE};
–GAN generator g^{GAN};
–GAN discriminator d^{GAN};
–Neural mesh renderer $g^{\mathrm{MR}} = [g^{\mathrm{HME}}, g^{\mathrm{Tex}}]$ (g^{NR}, g^{MANO} have no trainable parameters and the weights of g^{Reg} are fixed).
Initialization:
–Pre-train g^{FPE} based on [20];
–Randomize (parameters) of g^{GAN}, d^{GAN} and $g^{\mathrm{MR}} = [g^{\mathrm{HME}}, g^{\mathrm{Tex}}]$.
for $t = 1,...,T_1+T_2$ **do**
  for $n = 1,...,N/N'$ **do**
    if $t > T_1$ **then**
      For each data point $\mathbf{x}$ in the mini-batch D_n, generate MR output $\mathbf{x}'$ and GAN generator output $\mathbf{x}''$;
      Calculate gradient ∇L_D with respect to (the weights of) d^{GAN} (Eq. 6) on D_n, and update d^{GAN};
      Calculate gradient ∇L_{Img} with respect to (the weights of) g^{GAN} and g^{MR} (Eq. 4) on D_n, and update g^{GAN} and g^{MR};
    end
    if $D_n \in D^R_{Hand}, D^S_{Hand}, D^R_{HOI3D}$ **then**
      For each $\mathbf{x} \in D_n$, generate 2D heatmaps $\mathbf{h}$, 3D skeleton $\mathbf{y}$;
      Calculate L_{Heat} (See Eq. 3) and gradient ∇L_{Heat} with respect to (the weights of) g^{FPE} and update g^{FPE};
      Calculate L_{Pos} (See Eq. 5) and gradients ∇L_{Pos} with respect to g^{MR} on D_n, and update g^{MR};
      if $t > T_1$ **then**
        Generate 2D heatmaps $\mathbf{h}$, 3D skeleton $\mathbf{y}$ for $\mathbf{x}''$;
        Calculate L_{Heat} (See Eq. 3) and gradient ∇L_{Heat} with respect to (the weights of) g^{FPE} and update g^{FPE};
        Calculate gradient ∇L_{Pos} with respect to (the weights of) g^{MR}, g^{FPE} on D_n, and update g^{MR}, g^{FPE};
      end
    end
  end
end

Table 5: A comparison of existing RGB hand pose estimation benchmarks: Dexter+Object (*DO*) [17], EgoDexter (*ED*) [13], First person hand action (FPHA) [5], SynthHands (*SH*) [13], Ganerated Hands (GANerated) [12], *Obman* [7], and *HO3D* [6].

	Scenario	#Object	RGB	Depth	Hand mask	Real vs. Synthetic	Frames	Pose annotation	Viewpoint
CORe50 [11]	HOI	50	✓	✓	✗	Real	150k	✗	ego+3rd
GUN-71 [14]	HOI	28	✓	✓	✗	Real	12k	discrete grasp type	ego
STB [19]	H	✗	✓	✓	✗	Real	15k	21 joints	3rd
RHD [20]	H	✗	✓	✓	✓	Synthetic	41k+2.7k	21 joints	3rd
DO [17]	HOI	1	✓	✓	✗	Real	3k	5 tips	3rd
ED [13]	HOI	6	✓	✓	✗	Real	1.5k	5 tips	ego
FPHA [5]	HOI	26	✗	✓	✗	Real	100k	21 joints	ego
SH [13]	H+HOI	7	✓	✓	✓	Synthetic	63k	21 joints	ego
GANerated [12]	HOI	7	✓	✗	✗	Synthetic	330k	21 joints	ego
Obman [7]	H+HOI	8	✓	✓	✓	Synthetic	140k	21 joints	ego+3rd
HO3D [6]	HOI	6	✓	✓	✓	Real	10k	21 joints	3rd
FreiHand [21]	H+HOI	>26	✓	✗	✓	Real	35k (less than half for HOI)	21 joints	3rd+ego

Figure 2: Example results of hand-only image restoration and skeleton estimation on *HO3D*. (row 1) input images $\mathbf{x}$, (row 2) images $\mathbf{x}'$ generated by our initial mesh renderer g^{MR}, (row 3) images $\mathbf{x}''$ generated by the GAN generator g^{GAN}, (row 4-5) final images $\mathbf{z}$ synthesized by the mesh renderer g^{MR} and the corresponding skeleton estimates $\mathbf{y}$.

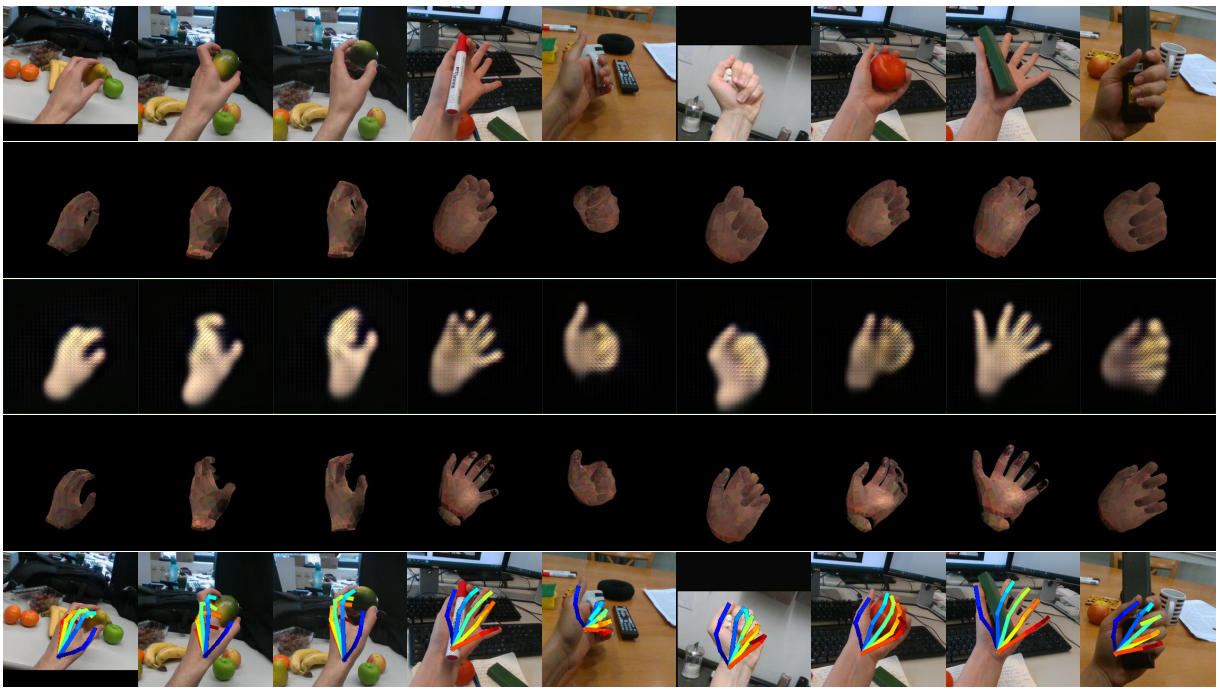


Figure 3: Example results of hand-only image restoration and skeleton estimation on *ED*. (row 1) input images $\mathbf{x}$, (row 2) images $\mathbf{x}'$ generated by our initial mesh renderer g^{MR}, (row 3) images $\mathbf{x}''$ generated by the GAN generator g^{GAN}, (row 4-5) final images $\mathbf{z}$ synthesized by the mesh renderer g^{MR} and the corresponding skeleton estimates $\mathbf{y}$.

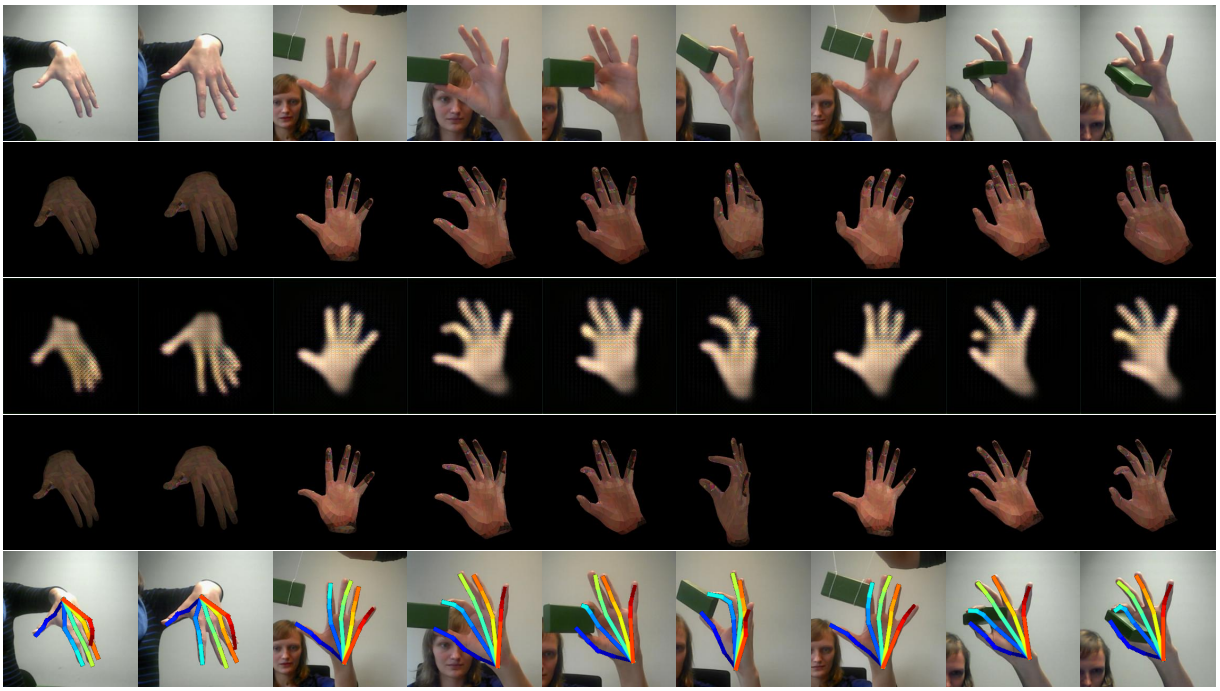
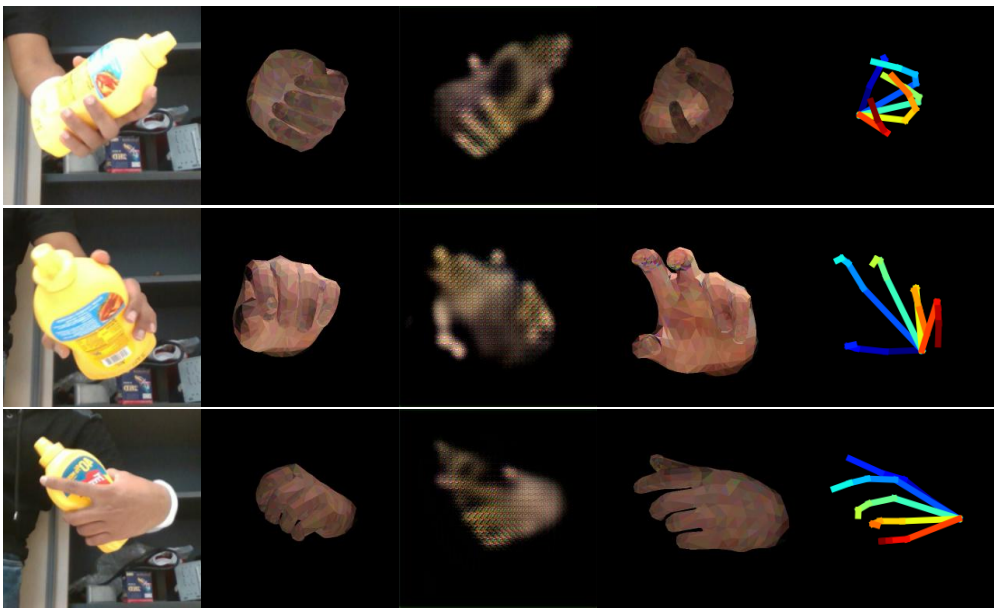


Figure 4: Example results of hand-only image restoration and skeleton estimation on *DO*. (row 1) input images $\mathbf{x}$, (row 2) images $\mathbf{x}'$ generated by our initial mesh renderer g^{MR}, (row 3) images $\mathbf{x}''$ generated by the GAN generator g^{GAN}, (row 4-5) final images $\mathbf{z}$ synthesized by the mesh renderer g^{MR} and the corresponding skeleton estimates $\mathbf{y}$.



(a) (b) (c) (d) (e)

Figure 5: Examples of failure cases (*HO3D*): (a) input images $\mathbf{x}$, (b) images $\mathbf{x}'$ generated by our initial mesh renderer g^{MR}, (c) images $\mathbf{x}''$ generated by the GAN generator g^{GAN}, (d-e) final images $\mathbf{z}$ synthesized by the mesh renderer g^{MR} and the corresponding skeleton estimates $\mathbf{y}$. Being distracted by the 'yellow bottle', our domain adaptation network generated imprecise initial hand-only reconstructions ($\mathbf{x}''$). Note that even with these initial reconstructions, our model generated accurate or reasonably accurate final mesh and skeleton estimates.

[13] Franziska Mueller, Dushyant Mehta, Oleksandr Sotnychenko, Srinath Sridhar, Dan Casas, and Christian Theobalt. Real-time hand tracking under occlusion from an egocentric RGB-D sensor. In *ICCV*, 2017. 4, 5

[14] Gregory Rogez, James S. Supancic, and Deva Ramanan. First-person pose recognition using egocentric workspaces. In *CVPR*, 2015. 4, 5

[15] Gregory Rogez, Philippe Weinzaepfel, and Cordelia Schmid. LCR-Net++: Multi-person 2D and 3D pose detection in natural images. *PAMI*, 2019. 4

[16] Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: Modeling and capturing hands and bodies together. In *SIGGRAPH Asia*, 2017. 1, 4

[17] Srinath Sridhar, Franziska Mueller, Michael Zollhöfer, Dan Casas, Antti Oulasvirta, and Christian Theobalt. Real-time joint tracking of a hand manipulating an object from RGB-D input. In *ECCV*, 2016. 4, 5

[18] Linlin Yang, Shile Li, Dongheui Lee, and Angela Yao. Aligning latent spaces for 3D hand pose estimation. In *ICCV*, 2019. 4

[19] Jiawei Zhang, Jianbo Jiao, Mingliang Chen, Liangqiong Qu, Xiaobin Xu, and Qingxiong Yang. A hand pose tracking benchmark from stereo matching. In *ICIP*, 2017. 5

[20] Christian Zimmermann and Thomas Brox. Learning to estimate 3D hand pose from single RGB images. In *ICCV*, 2017. 3, 4, 5

[21] Christian Zimmermann, Duygu Ceylan, Jimei Yang, Bryan Russell, Max Argus, and Thomas Brox. FreiHand: A dataset for markerless capture of hand pose and shape from single RGB image. In *ICCV*, 2019. 4, 5