Supplemental to – Pushing the Envelope for RGB-based Dense 3D Hand Pose Estimation via Neural Rendering

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

Kwang In Kim
UNIST
kimki@unist.ac.uk

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

This supplemental material provides a brief summary of our training and testing processes (Sec. 1.1), details of our skeleton regressor (Sec. 1.2), and data pre-processing and normalization steps (Sec. 1.3); and presents additional results and examples (Sec. 2). Some contents from the main paper are reproduced so that this document is self-contained.

1. Details of our algorithm

1.1. Summary of the training and testing processes

Testing. Our dense hand pose estimator (DHPE) receives an input RGB image $\mathbf{x} \in \mathcal{X}$ and generates the corresponding 2D segmentation mask $\mathbf{m}' \in \mathcal{M}$ and 3D skeleton $\mathbf{j}' \in \mathcal{J}$ estimates. During the testing process, it also synthesizes the 3D mesh $\mathbf{v}' \in \mathcal{V}$ corresponding $\mathbf{x}$: The DHPE decomposes into multiple component functions:

$$\mathcal{X} \underbrace{\overbrace{\xrightarrow{f^{E2D}} \mathcal{Z} \xrightarrow{f^{E3D}}}^{f^{HME}=f^{E3D}\circ f^{E2D}} \mathcal{V} \xrightarrow{f^{Proj}=(f^{Reg},f^{Ren})}}_{f^{DHPE}=f^{Proj}\circ f^{HME}} \mathcal{Y} = (\mathcal{J}, \mathcal{M}), \qquad (1)$$

where the hand mesh estimator (HME) f^{HME} estimates a 3D hand model (MANO)-based parameterization $\mathbf{h}'$ of $\mathbf{v}'$:

$$\mathbf{h} = [\mathbf{p}, \mathbf{s}, \mathbf{c}_q, \mathbf{c}_s, \mathbf{c}_t]^\top. \qquad (2)$$

Here $\mathbf{p}$ and $\mathbf{s}$, respectively represent the shape and pose (articulation) while the other three parameters represent a camera (3D rotation $\mathbf{c}_q \in \mathbb{R}^4$, scale $\mathbf{c}_s \in \mathbb{R}$, and translation $\mathbf{c}_t \in \mathbb{R}^3$).

Once the initial mesh (parameter) $\mathbf{h}'$ is estimated, our algorithm refines it by enforcing its consistency over the intermediate variables generated during testing: It minimizes

$$L_{\mathbf{h}} = \left\| \left[[f^{DHPE}(\mathbf{x})]_{\mathcal{J}} \right]_{XY} - \mathbf{j}'_{J2D} \right\|_2^2 + \lambda \left\| F(\mathbf{x}) - F(f^{Ren}(\mathbf{v}') \odot \mathbf{x}) \right\|_2^2 + L_{Lap}, \qquad (3)$$

where $\odot$ denotes element-wise multiplication and $[\mathbf{j}]_{XY}$ extracts the $x, y-$coordinate values of skeleton joints from $\mathbf{j}$. The Laplacian regularizer L_{Lap} enforces spatial smoothness in the mesh $\mathbf{v}$. This helps avoid generating implausible hand meshes as suggested by Kanazawa et al. [4]. Our renderer f^{Ren} synthesizes a 2D hand segmentation mask from a mesh by simulating the camera view of $\mathbf{x}$. Algorithm 2 summarizes the testing process.

Training. For training, our system receives 1) a set of training data $D = \{(\mathbf{x}_i, (\mathbf{j}_i, \mathbf{m}_i))\}_{i=1}^{l}$ which consists of input RGB images $\mathbf{x}_i$, and the corresponding ground-truth 3D skeletons $\mathbf{j}_i$ and 2D segmentation masks $\mathbf{m}_i$, 2) the projection operator f^{Proj}, 3) the MANO model consisting of its PCA shape basis and the mean pose vector. Our algorithm optimizes the weights of the 3D mesh estimator f^{E3D} and 2D feature extractor F based on L (Eq. 8). In parallel, the joint estimation network f^{J2D} is optimized based on L_{J2D} (Eq. 4). Algorithm 1 summarizes the training process. The two training hyperparameters T (the number of epochs) and N' (the size of mini-batch) are determined at 100 and 40, respectively.

$$L_{J2D}(f^{J2D}) = \| f^{J2D}(\mathbf{x}) - \mathbf{j}_{2DHeat} \|_2^2 \qquad (4)$$

$$L_{Feat}(F) = \| F(\mathbf{x}) - F(\mathbf{x} \odot \mathbf{m}) \|_2^2 \qquad (5)$$

$$L_{Sh} = \left\| [f^{DHPE}(\mathbf{x}_i)]_{\mathcal{M}} - \mathbf{m}_i \right\|_2^2 \qquad (6)$$

$$L_{Ref} = \left\| f^{Ref}\left([\mathbf{j}'_{2D}(t), F(\mathbf{x}), \mathbf{h}'(t), f^{Reg}(\mathbf{v}')] \right) - \mathbf{j}_{2DGT} \right\|_2^2 \qquad (7)$$

$$L(f^{E3D}, F) = L_{Art}(f^{E3D}, F) + L_{Lap}(f^{E3D}, F) + L_{Feat}(F) + \lambda L_{Sh}(f^{J3D}, F) + L_{Ref}(f^{J3D}, F), \qquad (8)$$

where $[\mathbf{y}]_{\mathcal{M}}$ extracts the $\mathbf{m}$-component of $\mathbf{y} = (\mathbf{j}, \mathbf{m})$.

1.2. Skeleton regressor f^{Reg}

The skeleton regressor f^{Reg} receives a (predicted) mesh consisting of 778 vertices $\mathbf{v} \in \mathcal{V} \subset \mathbb{R}^{778\times 3}$ and generates 21 skeletal joint positions $\mathbf{j} \in \mathcal{J} \subset \mathbb{R}^{21\times 3}$. Our regressor builds upon the original MANO regressor which is implemented as three multi-dimensional linear regressors, each aligned with a coordinate axis [5]: The $x-$axis regressor receives the $x-$coordinate values of $\mathbf{v}$ and synthesizes the

```
Algorithm 1: Training process
  Input:
    –Training data $D = \{(\mathbf{x}_i, (\mathbf{j}_i, \mathbf{m}_i))\}_{i=1}^{l}$:
        $\mathbf{x}$: RGB image;
        $(\mathbf{j}, \mathbf{m})$: ground-truth
            3D skeleton and 2D segmentation mask;
    –Projection operator $f^{Proj} = (f^{Reg}, f^{Ren})$;
    –MANO model: PCA shape basis;
                 mean pose vector;
    –Hyper-parameters: number $T$ of epochs
                       size $N'$ of mini-batch;
  Output: (Weights of)
    –3D mesh estimator $f^{E3D}$;
    –2D evidence estimator $f^{E2D} = (F, f^{J2D})$;
  Initialization:
    –Randomize (parameters) of $f^{E3D}$;
    –Pre-train $F$ based on [2];
    –Pre-train $f^{J2D}$ based on [7];
  for $t = 1, \ldots, T$ do
      for $n = 1, \ldots, N/N'$ do
          For each data point $\mathbf{x}$ in the mini-batch $D_n$,
          evaluate (feed-forward) $f^{DHPE}$ on $\mathbf{x}$:
          Generate mesh parameter $\mathbf{h}'$, 3D skeleton $\mathbf{j}'$
            and 2D segmentation mask $\mathbf{m}'$;
          Generate 2D evidences $(F(\mathbf{x}), \mathbf{j}'_{2D}(t))$, mesh
            parameter $\mathbf{h}'$, 3D skeleton $\mathbf{j}'$ and 2D
            segmentation mask $\mathbf{m}'$;
          if $t > 20$ then
              Augment $D$ with new synthetic data
                instances generated from $\mathbf{h}'$ (Eq. 2), by
                changing its shape $\mathbf{s}'$ and viewpoint $\mathbf{q}'$;
          end
          Calculate gradient $\nabla L$ with respect to (the
            weights of) $f^{E3D}$ (Eq. 8) on $D_n$, and update
            $f^{E3D}$;
          Calculate gradients $\nabla L_{Feat}$ (Eq. 5) and $\nabla L$
            with respect to $F$ on $D_n$, and update $F$;
          Calculate gradient $\nabla L_{J2D}$ (Eq. 4) with respect
            to $\nabla f^{J2D}$ on $D_n$, and update $f^{J2D}$;
      end
  end
```

$x-$axis coordinate values of 16 skeletal joints. The $y-$axis and $z-$axis regressors are constructed similarly. In this way, the original MANO regressor estimates only 16 joint positions. The remaining five, finger tip positions are estimated by simply *selecting* a point of $\mathbf{v}$ that corresponds to a finger tip, per axis.[1] These additional regressors are implemented for each finger tip and for each axis, as a 778-dimensional

[1]Unlike other skeletal joint locations which lie inside the mesh $\mathbf{v}$, finger tips lie on (the surface) $\mathbf{v}$. Therefore, selecting a point on $\mathbf{v}$ can give a good joint location estimate.

```
Algorithm 2: Testing process
  Input: Test image $\mathbf{x}$;
  Output:
    –3D mesh $\mathbf{v}'$;
    –2D segmentation mask $\mathbf{m}'$;
    –3D skeleton $\mathbf{j}'$;
  Feed-forward $f^{DHPE}$ on $\mathbf{x}$: Generate 2D evidence
    $\mathbf{j}'_{2D}(t)$, and 3D mesh $\mathbf{v}'$ and it's parameter $\mathbf{h}'$;
  for $t = 1, \ldots, 50$ do
      Update $\mathbf{h}'(t)$ using Eq. 3.
  end
  Generate $\mathbf{v}'$ from $\mathbf{h}'(t)$ and $\mathbf{m}'$;
  Generate $\mathbf{j}'$ from $\mathbf{v}'$;
```

vector where all elements are zero except for the entry corresponding to the vertex location of the corresponding finger tip, where value 1 is assigned. As a whole, our regressor f^{Reg} is represented as three matrices of size 778×21. As a linear regressor, f^{Reg} is differentiable with respect to its input and output arguments.

1.3. Data pre-processing and normalization

To facilitate the training of the skeleton regressor f^{Reg}, similarly to [3, 7], we explicitly normalize its output space $\mathcal{J}$: Our articulation loss L_{Art} measures the deviation between the skeleton estimated from the training input $\mathbf{x}_i$ and the corresponding growth-truth $\mathbf{j}_i$:

$$L_{Art} = \|[f^{DHPE}(\mathbf{x}_i)]_{\mathcal{J}} - \widehat{\mathbf{j}}_i\|_2^2, \tag{9}$$

where $[\mathbf{y}]_{\mathcal{J}}$ extracts the $\mathbf{j}$-component of $\mathbf{y} = (\mathbf{j}, \mathbf{m})$.

Here, $\widehat{\mathbf{j}}$ spatially normalizes $\mathbf{j}$: First, a tight 2D hand bounding box is extracted from the corresponding ground-truth 2D skeleton of $\mathbf{x}_i$, and the center of the skeleton is moved to its middle finger's MCP position. Then, each axis is normalized to a unit interval $[0, 1]$: The $x, y-$coordinate values are divided by g (=1.5 times the maximum of height and width of the bounding box). The $z-$axis value is divided by $(z_{Root} \times g)/\mathbf{c}_f$ where z_{Root} is the depth value of the middle finger's MCP joint and $\mathbf{c}_f$ is the focal length of the camera.

At testing, once normalized skeletons are generated, they are inversely normalized to the original scale based on the parameters g and z_{Root}.

Estimation of the bounding box size g. First, we use Zimmermann and Brox's hand detector [7] to infer bounding boxes in each video frame. The corner coordinates of the detected boxes are then temporally smoothed by taking an average over the past five frames.

For *RHD*, following Cai et al.'s experimental settings [1], the bounding boxes are extracted from the ground-truth 2D skeletons provided in the dataset, to facilitate a fair comparison with their algorithm.

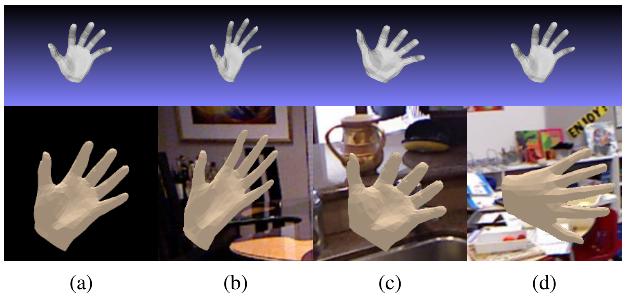
Figure 1: Data augmentation examples: (a) the original mesh, (b) and (c) shape variations from (a), and (d) viewpoint variation from (a). 3D meshes on top are textured and rendered on random backgrounds (bottom).

Estimation of the hand depth z_{Root}. We use the 3D root depth estimation algorithm proposed by Iqbal et al. [3] for *DO*. For *RHD* and *SHD*, for pair comparison with [1], the ground-truth depth values are used.

2. Additional results and examples

Figure 1 shows example outputs of our data augmentation method. Figure 2 shows additional dense hand pose estimation results extending Fig. 7 of the main paper. When compared with the results obtained by a variation of our algorithm that does not use the shape loss (b–d: L_{Sh}; Eq. 6), our final algorithm (e–g) acheived much higher shape estimation accuracy (c and f, especially in 1st, 2nd, 5th, and 7th examples), which led to better alignment of hand contours (c and f) and eventually to significantly lower pose estimation error (b and e). These examples confirm the quantitative results shown in Fig. 3 and demonstrate the benefits of shape estimation even when the final goal is to estimate skeletal poses.

References

[1] Y. Cai, L. Ge, J. Cai, and J. Yuan. Weakly-supervised 3d hand pose estimation from monocular rgb images. In *ECCV*, 2018. 2, 3

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[3] U. Iqbal, P. Molchanov, T. Breuel, J. Gall, and J. Kautz. Hand pose estimation via latent 2.5D heatmap regression. In *ECCV*, 2018. 2, 3

[4] A. Kanazawa, S. Tulsiani, A. A. Efros, and J. Malik. Learning category-specific mesh reconstruction from image collections. In *ECCV*, 2018. 1

[5] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. In *SIGGRAPH Asia*, 2017. 1

[6] D. J. Tan, T. Cashman, J. Taylor, A. Fitzgibbon, D. Tarlow, S. Khamis, S. Izadi, and J. Shotton. Fits like a glove: Rapid and reliable hand shape personalization. In *CVPR*, 2016. 4

[7] C. Zimmermann and T. Brox. Learning to estimate 3D hand pose from single RGB images. In *ICCV*, 2017. 2

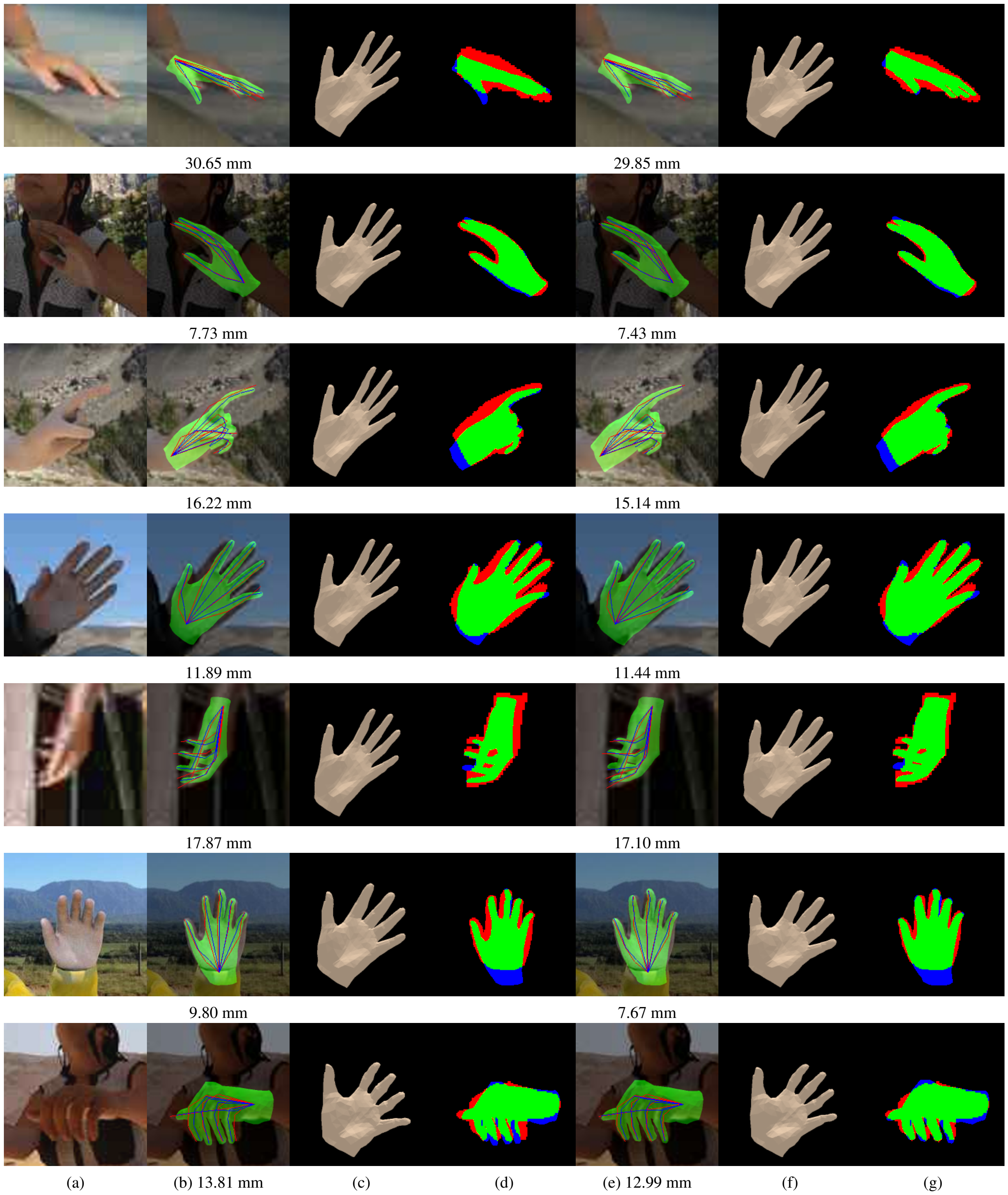

Figure 2: Dense hand pose estimation examples. (a) input images, (b-d) and (e-g) results obtained without and with shape loss, respectively. (b,e) estimated hand meshes overlaid on the input image and the corresponding estimated skeletons (Blue) overlaid with their ground-truths (Red), (c,f) estimated shapes rendered in canonical articulation and viewpoints, and (d,g) Color-coded 2D segmentation masks: (Green and Blue: estimated masks; Green and Red: ground-truth masks; Red and Blue highlight errors). Our visualization method in (d) and (g) is inspired by [6].

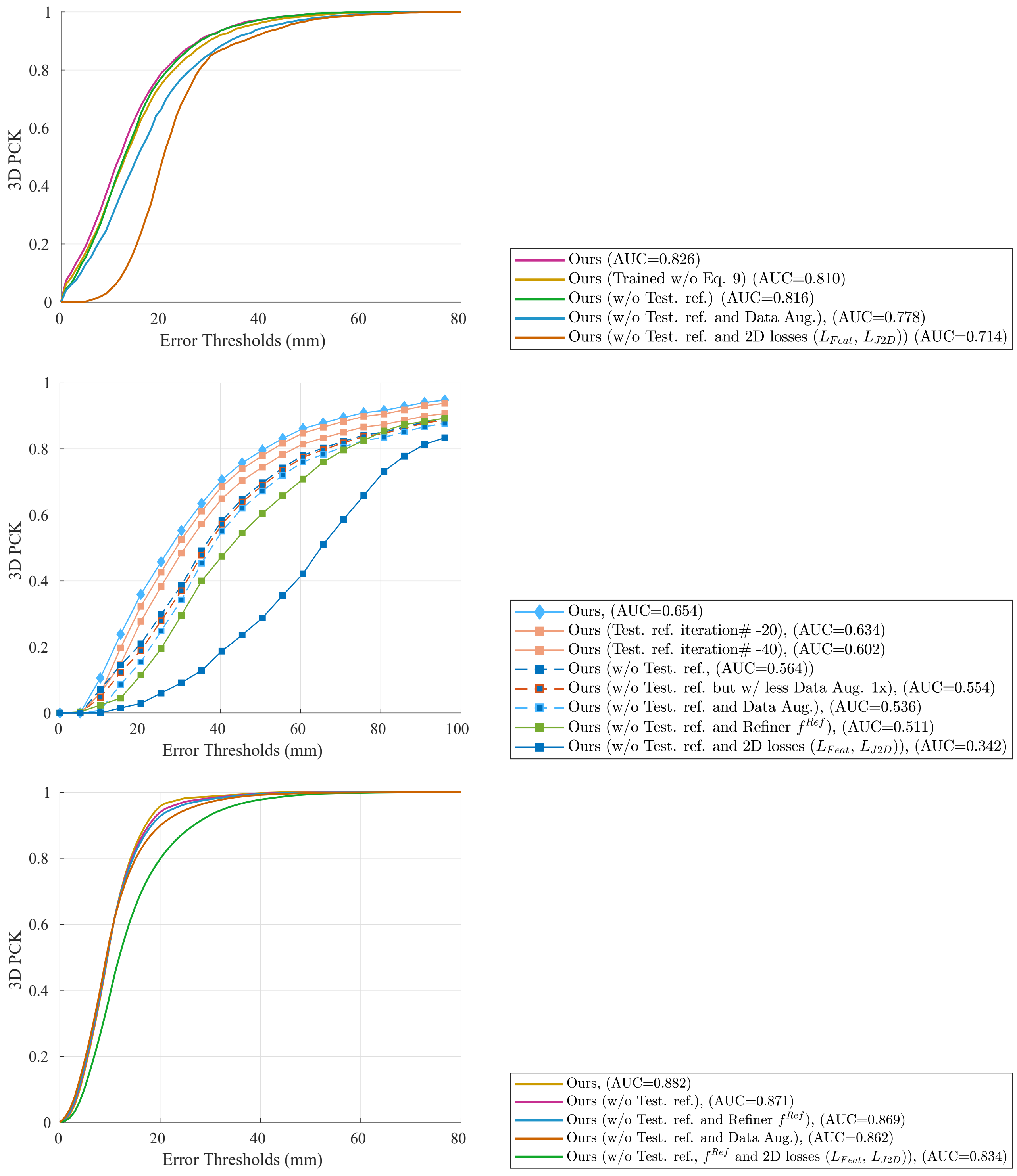

Figure 3: Performance of our algorithm with different design choices. Top to bottom: results on *RHD*, *DO*, and *SHD*, respectively.