BiHand: Recovering Hand Mesh with Multi-stage Bisected Hourglass Networks

Lixin Yang
siriusyang@sjtu.edu.cn
Jiasen Li
lijiasen0921@sjtu.edu.cn
Wenqiang Xu
vinjohn@sjtu.edu.cn
Yiqun Diao
diaoyiqun@sjtu.edu.cn
Cewu Lu
lucewu@sjtu.edu.cn

Machine Vision and Intelligence Group
Shanghai Jiao Tong University
Shanghai, China

Abstract

3D hand estimation has been a long-standing research topic in computer vision. A recent trend aims not only to estimate the 3D hand joint locations but also to recover the mesh model. However, achieving those goals from a single RGB image remains challenging. In this paper, we introduce an end-to-end learnable model, BiHand, which consists of three cascaded stages, namely 2D seeding stage, 3D lifting stage, and mesh generation stage. At the output of BiHand, the full hand mesh will be recovered using the joint rotations and shape parameters predicted from the network. Inside each stage, BiHand adopts a novel bisecting design which allows the networks to encapsulate two closely related information (*e.g.* 2D keypoints and silhouette in 2D seeding stage, 3D joints, and depth map in 3D lifting stage, joint rotations and shape parameters in the mesh generation stage) in a single forward pass. As the information represents different geometry or structure details, bisecting the data flow can facilitate optimization and increase robustness. For quantitative evaluation, we conduct experiments on two public benchmarks, namely the Rendered Hand Dataset (RHD) and the Stereo Hand Pose Tracking Benchmark (STB). Extensive experiments show that our model can achieve superior accuracy in comparison with state-of-the-art methods, and can produce appealing 3D hand meshes in several severe conditions. The training codes, model and dataset are publicly available at https://github.com/lixiny/bihand.

1 Introduction

Hand is one of the most crucial elements when human interact with surroundings. Extracting 3D structure of a hand from a single RGB image can benefit plenty of applications including VR/AR, action recognition, and human-computer interaction. Generally, 3D hand estimation is formulated as 3D joint location estimation, or more detailed 3D mesh reconstruction. Since 3D hand mesh contains richer geometric information and is considered indispensable

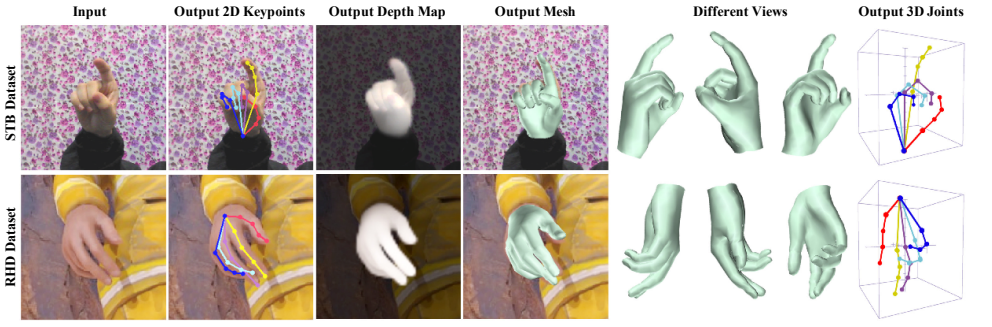

Figure 1: We introduce a multi-stage bisected network, BiHand, for single RGB image hand mesh recovery. BiHand progressively converts RGB image to 2D geometry (2D keypoints and silhouette), then to 3D structure (3D joints and depth map), and finally to a full hand mesh in a single forward pass.

for applications involving grasping or hand-object contact, in this work, we focus on recovering the 3D hand model from a single RGB image.

The past decade has witnessed rapid growth in the study of 3D hand mesh recovery. Early works applied multi-view stereo approaches to reconstruct hand [11, 32]. Later, to reduce the requirement of multi-view images, model templates (*e.g.* Primitives approximation [21], sphere-meshes [29], MANO [26]) were adopted to fit the prior hand shape to a single image. With the advent of deep learning techniques, it became possible to directly estimate hand model parameters from color or depth pixels [2, 17, 22]. Recently, RGB-based learning methods became the mainstream approach. The majority of those methods [1, 2, 12, 36] learned to regress the parameters of hand model, typically MANO. However, since they regressed the lossy compressed PCA component of MANO, their accuracy cannot be guaranteed. To address this, an inverse kinematics based method [37] was proposed to calculate the rotation angle from the location by leveraging the kinematic chains of hand.

Estimating a 3D model from a 2D image is intrinsically an ill-posed problem. Specifically, it suffers from the perspective ambiguity. To alleviate this issue, besides adopting CNN as the main feature extractor to estimate the hand model parameters, former researches also incorporated more geometric information as auxiliary supervision, such as 2D keypoints, silhouette, 3D joints, depth map, etc. Such additional geometric information can be used as intermediate regularization or post regularization. However, how to arrange these information is non-trivial. 2D keypoints and silhouette both represent planar projection with different level of details. They are strong cues to infer the 3D structure. Similarly, 3D joints and depth map encode 3D structure information with different levels of details. When it comes to estimate hand model, rotation parameters and shape parameters also belong to different information flow. Based on this observation, We propose a multi-stage framework with bisecting design, named **BiHand**. As is shown in Fig. 1, it can progressively convert the RGB image to 2D geometry within the 2D seeding module (SeedNet), then to 3D structure within the 3D lifting module (LiftNet), and finally to the mesh model within mesh generation module with a novel shape-aware inverse kinematics network (SIKNet).

To train SIKNet requires joint location-rotation training pair. However, only a few MoCap samples are available in the MANO dataset. Thus, we create a SIK-1M dataset containing one million synthetic location-rotation pairs. Though the synthesis can potentially

generate unlimited pairs of training data, we find one million is enough.

For quantitative evaluation, we experiment our method on two standard benchmarks, namely the Rendered Hand Dataset (RHD)[38] and the Stereo Hand Pose Tracking Benchmark (STB) [35]. BiHand achieves state-of-the-art performance with AUC 0.951 on the RHD and 0.997 on the STB.

In summary, our contributions are as follows:

- We propose a new end-to-end learnable framework, BiHand, to address the single RGB image hand mesh recovery problem. BiHand leverages both planar and 3D structural information as intermediate representations to stabilize the training. A novel bisecting design is proposed to organize the geometric information flow.
- To recover the mesh from estimated 3D joints, we also propose a shape-aware inverse kinematics network, SIKNet, to map the joint locations to MANO parameters. To train SIKNet with full supervision, we build up a large-scale dataset, SIK-1M.

2 Related Work

Our method closely relates to 3D hand pose estimation and 3D hand mesh reconstruction problems.

2.1 3D Hand Pose Estimation

Early works on 3D hand pose estimation mainly focused on regressing hand joints from a single depth image. They either exploited a model fitting [14] or learned a depth-joint mapping [28]. Recently, several deep learning based methods improved the estimation by employing CNNs [31], multi-view CNN [7], 3D CNN [8] or PointNet [9, 24]. Another stream on this topic is to estimate the 3D hand pose from an RGB image. Former researches adopted CNN feature extractor with several learning strategies including variational auto-encoder [27], iterative skeleton fitting [19], depth regularizer [3, 4], etc.

2.2 3D Hand Mesh Reconstruction

Multi-view Stereo Based Methods Since early single-view algorithms were sensitive to occlusion and noise, several multi-view methods were put forward. Delamarre and Faugeras [6] proposed to fit finger models on the stereo-based scene reconstruction. Ueda *et al*. [32] exploited multi-view silhouette images to reconstruct hand as a voxel model. Guan *et al*. [11] formulated this problem by fusing the multi-view images into a maximum a posterior framework. However, these methods usually require multi-camera setups, which is considered less general than a single depth or RGB camera nowadays.

Depth Based Methods Recovering meshes from a single depth image was mostly regarded as a model fitting problem. Khamis *et al*. [14] proposed to fit a morphable hand model into depth image by linear blender skinning (LBS). LBS function requires two sets of parameters, i.e. poses and shapes, to articulate a hand model. Remelli *et al*. [25] proposed to decouple pose and shape parameters and calibrate the shapes in a low dimensional space. A recent method [17] proposed to directly regress LBS parameters using CNN on the depth image.

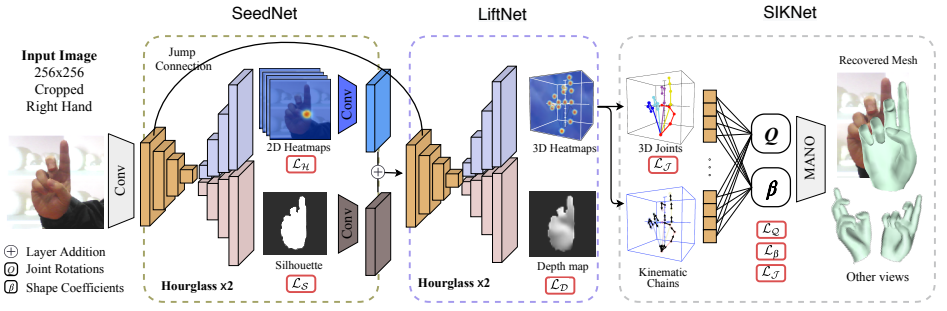

Figure 2: Overview of our proposed multi-stage bisected network (BiHand) on 3D hand mesh reconstruction. It comprises three sequential modules. First, the 2D seeding module, SeedNet, predicts 2D heatmaps and silhouette. Second, the 3D lift module, LiftNet, lifts the 2D predictions to 3D heatmaps and depth map. Third, the mesh generation module with SIKNet recovers the full hand meshes based on the 3D predictions.

RGB Based Methods Previous methods focused on fitting a rigid articulated hand model on an RGB image. Paschalis *et al*. [22] proposed to estimate hand joint position and adopted iterative model fitting to obtain the joint rotations. Similar ideas were also proposed by Kokic *et al*. [15]. However, those methods are incapable of capturing hand shape variants. Recently, with a parametric hand model MANO, several researchers proposed to fit its pose and shape parameters into pixels. Boukhayma *et al*. [2] first proposed to directly regress the MANO parameters from the input of image and heatmaps. Zhang *et al*. [36] forwarded it by adopting a neural render to employ silhouette supervision on meshes. Adopting neural render to supervise mesh was also proposed in [1, 10]. Since both Boukhayma and Zhang's methods only regressed the MANO's PCA components, their accuracy suffered from its lossy property. Recently, instead of regressing on PCA, Zhou *et al*. [37] proposed to directly estimate the rotations of all hand joints. Zhou's methods can achieve higher robustness on both joint and mesh estimation. Instead of using MANO, Ge *et al*. [10] proposed to directly regress the vertex positions on a template hand mesh using GraphCNN [5]. However, their method requires training on a dataset with dense vertices annotations, which is not practical in the real world. Compared with most of the aforementioned methods, our BiHand is more informative and feasible. The bisected hourglass allows the networks to encapsulate homogeneous information during training, and the decoupled stages make the training process more effective.

3 Method

3.1 Overview

The proposed BiHand seeks to predict a 3D hand mesh model from a single RGB image. To achieve this, given an image $\mathcal{I}$ as input, BiHand first predicts 2D heatmaps $\mathcal{H}_{2D} \in \mathbb{R}^{K\times H\times W}$ and silhouette $\mathcal{S} \in \mathbb{R}^{H\times W}$ through 2D seeding module, SeedNet (Sec. 3.2). Then 2D geometries are lifted to 3D heatmaps $\mathcal{H}_{3D} \in \mathbb{R}^{K\times Z\times H\times W}$ and depth map $\mathcal{D} \in \mathbb{R}^{H\times W}$ through 3D lifting module, LiftNet (Sec. 3.3). And thus 3D joint locations $\mathcal{X} \in \mathbb{R}^{K\times 3}$ can be obtained from $\mathcal{H}_{3D}$, and be put through the mesh generation module with SIKNet (Sec. 3.4) to recover

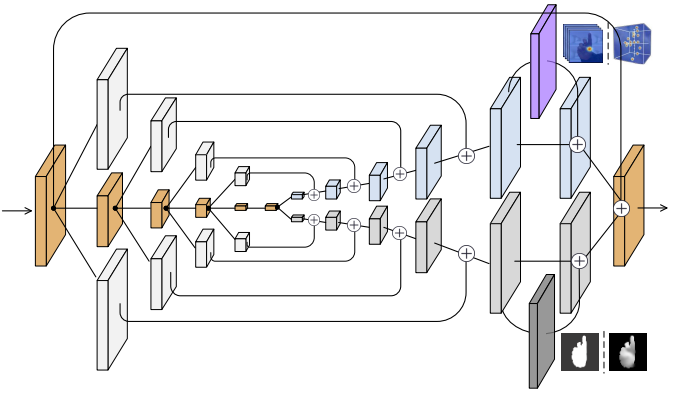

Figure 3: Illustration of a single bisected hourglass network. Each block represents a residual module. The resolutions of input and output layer are both 64×64 and the number of channels across the whole network is 256.

the full hand mesh $\mathcal{M}(\theta,\beta) \in \mathbb{R}^{N\times 3}$, where $\theta \in \mathbb{R}^{16\times 3}$, $\beta \in \mathbb{R}^{10}$ are the rotation and shape parameters from MANO. The overall pipeline is illustrated in Fig. 2.

If not specified, $K = 21$ indicates the number of hand joints and $N = 778$ indicates the number of mesh vertices. $(H,W) = (64,64)$ means the resolution of 2D heatmaps, silhouette, and depth map. $Z = 64$ is the depth resolution of 3D heatmaps.

3.2 2D Seeding Module, SeedNet

The first stage of BiHand is 2D seeding module, which transforms the RGB image $\mathcal{I}$ to 2D heatmaps $\mathcal{H}_{2D}$ and 2D silhouette $\mathcal{S}$. Adopting the 2D information as the bridge between RGB and 3D structure has three advantages: interpretable, robust, and easy to train. Some studies on 3D hand pose estimation confirmed the observation [3, 4, 18, 19, 38].

$\mathcal{H}_{2D}$ and $\mathcal{S}$ both represent the 2D structure with different levels of details, which is double-edged. For the up-side, they can be encoded simultaneously since the source is the same, and when $\mathcal{H}_{2D}$ is sensitive to motion blur or self-occlusion, $\mathcal{S}$ can provide stable cues. For the down-side, they should be separately treated when decoding, since the amount of information to be restored are different. Of course, we can leverage the high capacity of CNN to force a single decoder to encapsulate two different information flows, but it is easier to just bisect the flow with two decoders to learn.

Bisected Hourglass Network To separate the information flow, SeedNet is designed as a stacked bisected hourglass network, as shown in Fig. 3. Bisected hourglass means the network has one encoder, two decoders where the encoder and decoder are the same with hourglass [20]. Specifically, it first encodes the input to latent features $\mathcal{F}_{2D}$ through 4 down-sampling blocks. The latent features $\mathcal{F}_{2D}$ are then passed through two separated decoders, one for heatmaps $\mathcal{H}_{2D}$ and the other for silhouette $\mathcal{S}$. The decoder also consists of 4 up-sampling blocks. Each pixel in the k^{th} heatmap $\mathcal{H}_{2D}^{(k)}$ indicates the confidence of that pixel being covered by the k^{th} keypoint. And each pixel in silhouette $\mathcal{S}$ indicates the confidence of that pixel belongs to hand's segmentation. The two decoders are the same in architecture and symmetrical to the encoder, but they do not share the same weight. Similar to the original hourglass, the bisected hourglass can be stacked as well. We stack two bisected hourglasses in SeedNet. For each hourglass, we add intermediate supervision on the estimated

2D heatmaps and silhouette, which are then concatenated as the input of the subsequent networks.

Loss Terms The loss function of SeedNet

$$\mathcal{L}_{2D} = \lambda_{\mathcal{H}}\, \mathcal{L}_{\mathcal{H}} + \lambda_{\mathcal{S}}\, \mathcal{L}_{\mathcal{S}} \tag{1}$$

comprises two terms. First, the heatmaps loss $\mathcal{L}_{\mathcal{H}}$ is defined as the pixel-wise mean squared error (MSE) between the estimated and ground-truth 2D heatmaps. The ground-truth 2D heatmap $\mathcal{H}_{2D}^{(k)}$ of keypoint k is generated by 2D Gaussian distributions with mean at k's annotations and standard deviation $\sigma = 1.5$. The second term silhouette loss $\mathcal{L}_{\mathcal{S}}$ is the cross-entropy loss between the predicted and the ground-truth silhouette.

3.3 3D Lifting Module, LiftNet

The objective of LiftNet is to infer the 3D joint location given the 2D heatmaps and silhouette from SeedNet. Several solutions [30, 38] were proposed to directly lift the estimated 2D heatmaps to 3D coordinates. However, as was argued in [3], one can hardly resolve the inherent ambiguity in projection. Similar to [3], we alleviate this issue by maintaining the contextual information alongside with heatmaps in the network. Besides, We choose to estimate 3D heatmaps instead of joint vectors (x, y, z coordinates of K joints) since the regression on discrete coordinates imposes high non-linearity to this problem.

Similar to 2D information, 3D heatmaps and depth map are homogeneous in terms of 3D spatial structures. Training the depth map predictor could provide sufficient information on the relative depth to prevent joint predictors from abnormalities. However, 3D heatmaps and depth map are also different in the grained level. Therefore, we also adopt the bisected design in LiftNet.

LiftNet Design The architecture of our LiftNet follows SeedNet, while changing the output from 2D heatmaps, silhouette to 3D heatmaps, depth map. Our joint decoder tackles 3D joint estimation as classification by outputting K volumetric blocks (3D heatmaps) $\mathcal{H}_{3D}$. Each voxel in k^{th} block $\mathcal{H}_{3D}^{(k)}$ represents the likelihood of this voxel being covered by the k^{th} joints. We design the joint decoder to output 3D heatmaps in the normalized-*uvd* form, where *uv* corresponds to UV coordinates on the input image and d to the joint's root-relative scale-invariant depth value. The volumetric 3D heatmaps can be transferred to *uvd* coordinates through a differentiable *argmax* function. Technically, for j^{th} joints, its *uvd* coordinates can be obtained by:

$$[u_j,\ v_j,\ d_j]^\top = \sum_{d=1}^{Z} \sum_{v=1}^{H} \sum_{u=1}^{W} \mathcal{H}_{3D}^{(j)}(u,v,d) \cdot [u,\ v,\ d]^\top \tag{2}$$

where $\sum_{u,v,d} \mathcal{H}_{3D}^{(j)}(u,v,d) \equiv 1$. Finally, the *uvd* coordinates are converted to joint locations $\mathcal{X}^{(j)} = (x_j, y_j, z_j)^\top$ using camera intrinsic matrix. Alongside with the joint decoder, the depth decoder predicts the normalized depth map $\mathcal{D}$, where each pixel in $\mathcal{D}$ represents the normalized depth value of that pixel. Similar to SeedNet, we stack two bisected hourglass in LiftNet. Inspired from [23], we adopt a coarse-to-fine 3D heatmaps predictor with the resolution along z-axis $Z = 32$ in the first hourglass and $Z = 64$ in the second. Exploiting depth during 3D joint estimation was previously proposed in [3, 4, 10]. But in their methods the depth predictor was used as a regularizer after joint estimation, while in ours as a parallel counterpart.

Loss Terms The loss function of LiftNet

$$\mathcal{L}_{3D} = \lambda_{\mathcal{J}}\ \mathcal{L}_{\mathcal{J}} + \lambda_{\mathcal{D}}\ \mathcal{L}_{\mathcal{D}} \tag{3}$$

consists of two terms. First, the joint loss $\mathcal{L}_{\mathcal{J}}$ is defined as MSE loss on K joint positions: $\mathcal{L}_{\mathcal{J}} = \sum_{j=1}^{K} \|\mathcal{X}^{(j)} - \mathcal{X}^{*(j)}\|_2^2$, where $\mathcal{X}^*$ is the ground-truth joint location. Second, the depth map loss $\mathcal{L}_{\mathcal{D}}$ is defined as the smooth L1 loss between the estimated and ground-truth depth map: $\mathcal{L}_{\mathcal{D}} = |\mathcal{D} - \mathcal{D}^*|_{smoothL1}$, where $\mathcal{D}^*$ is the normalized ground-truth depth map.

3.4 Mesh Generation Module with SIKNet

In this section, we recover the full hand mesh by employing a parametric hand model, MANO [26]. MANO is a statistical model based on the SMPL for the human body [16]. Similar to SMPL, MANO formulates an articulated hand $\mathcal{M}$ with pose parameters θ and shape parameters β. Technically, θ represents the 16 joint rotations, excluding 5 fingertips, in the form of axis-angle vectors, and β represents the shape PCA coefficients learned from various hand scans. Similar to the majority of CG characters, MANO drives the articulated hand through the joint rotations.

Previous MANO-based methods in [2, 12, 36] directly regressed the PCA component. Since PCA is a lossy-compression method, in theory, joint information cannot be perfectly recovered. On the other hand, the human hand is by nature a kinematic chain. Therefore we can convert joint locations to its rotations through inverse kinematics (IK). Zhou *et al.* [37] designed an IK network to estimate θ and applied iterative optimization to obtain β. By observing that θ and β is closely related, we modify the original IK network to a bisected design. Since the network can simultaneously produce joint rotations and shape coefficients, we name it as shape-aware IK network (SIKNet).

Parametric Hand Model With the joint rotations and shape coefficients obtained, We can utilize MANO to reconstruct the full hand meshes. MANO first deforms a mean template mesh $\bar{\mathbf{T}}$ through two blend functions $\mathcal{B}(\theta)$ and $\mathcal{B}(\beta)$, and then the deformed template mesh $\mathcal{T}(\theta,\beta)$ is transferred to final mesh $\mathcal{M}(\theta,\beta)$ through a linear blend skinning (LBS) function $W(\cdot)$:

$$\mathcal{M}(\theta,\beta) = W(\mathcal{T}(\theta,\beta),\theta,\mathcal{W},\mathcal{J}(\beta)) \tag{4}$$

where $\mathcal{W}$ is blend weights, and $\mathcal{J}(\beta)$ is joint locations. Following the method in [12], we integrate MANO as a differentiable layer in our SIKNet.

SIKNet Design SIKNet mainly consists of a bisected regression network: one for rotation regression (θ-Reg) and the other for shape regression (β-Reg). Both θ-Reg and β-Reg are defined as network with seven fully-connected layers. We construct the input of θ-Reg as $\mathcal{I} = [\bar{\mathcal{X}}, \bar{\mathcal{K}}] \in \mathbb{R}^{2\times 21\times 3}$, where $\bar{\mathcal{X}}$ stands for the root-relative, scale-invariant joint locations and $\bar{\mathcal{K}}$ is the normalized direction vectors along the hand's kinematic chain. Based on the discussion in [37], we design θ-Reg to output quaternion $\mathcal{Q} \in \mathbb{R}^{16\times 4}$ instead of axis-angle. Another bisection in SIKNet is the β-Reg, which also takes as input $\mathcal{I} = [\bar{\mathcal{X}}, \bar{\mathcal{K}}]$ and outputs the shape parameters β.

Inverse Kinematics Dataset Noticing that the training of θ-Reg and β-Reg is decoupled from LiftNet, we can, therefore, train the whole SIKNet on a large hand-crafted dataset with direct quaternion supervision and indirect joint supervision. We construct our synthetic

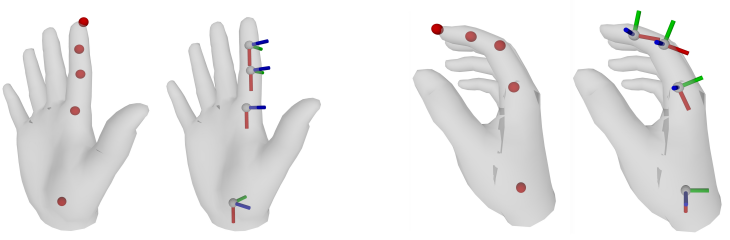

Figure 4: The Inverse Kinematics Dataset. We show two samples of the location-rotation data pair along the index finger. For each sample, the left column shows the 3D joint locations and the right column illustrate the relative joint rotations.

dataset through randomly sampling 20k pose and shape parameters with Gaussian distributions $\mathcal{N}(\mu_F, (2\sigma_F)^2)$, where μ_F and σ_F denote the mean and standard deviation of hands in FreiHand dataset [39] respectively. For each hand (θ_i, β_i), we then randomly sample 50 camera views. Thus the whole synthetic dataset contains one million hands with ground-truth joint location-rotation data pair (See Fig. 4). We split 80% of the synthetic SIK dataset for training and the remaining 20% for testing.

Loss Terms The loss function of SIKNet

$$\mathcal{L}_{IK} = \lambda_{\mathcal{Q}}\,\mathcal{L}_{\mathcal{Q}} + \lambda_{\beta}\,\mathcal{L}_{\beta} + \lambda_{\mathcal{J}}\,\mathcal{L}_{\mathcal{J}} \tag{5}$$

consists of three terms. First, same as in [37], the quaternion loss $\mathcal{L}_{\mathcal{Q}}$ is a combination of another three terms: cosine loss, L2 loss, and norm loss. The cosine and L2 loss measures the angular cosine error and Euclidean error between the predicted and ground-truth quaternions, respectively, and the norm loss performs regularization.

The second term in Eq. 5 is the shape loss, which is defined as:

$$\mathcal{L}_{\beta} = \sum_{b} \|\bar{L}_b(\mathcal{Q}, \beta) - \bar{L}_b^*\|_2^2 + \|\beta\|_2^2 \tag{6}$$

where $\bar{L}_b(\mathcal{Q}, \beta)$ and $\bar{L}_b^*$ in first term are the estimated and ground-truth scale-invariant bone length for bone b respectively, and the second term is L2 regularization.

The third term in Eq. 5 is the joint MSE loss between the estimated joint positions $X(\mathcal{M}(\mathcal{Q}, \beta))$ and ground-truth $\mathcal{X}^*$, where $\mathcal{M}(\cdot)$ is the mesh and $X(\cdot)$ stands for interpolation from meshes to joints. With $\mathcal{L}_{\mathcal{J}}$, we can indirectly supervise the θ-Reg and β-Reg when training on datasets without ground-truth quaternions annotation.

4 Experiment

4.1 Datasets and Metrics

Datasets We mainly train and evaluate our network on two public datasets: the Rendered Hand Dataset (RHD) [38] and the Stereo Hand Pose Tracking Benchmark (STB) [35]. The RHD is a synthetic dataset with 41258 training and 2728 testing samples. The STB is a real-world dataset containing 12 sequences with 1500 frames per-sequence. We choose 10 sequences in STB for training and the remaining 2 for testing the same as in [38]. Similar to [3], we shift the root joints in STB from palm to wrist to make it consistent with RHD. In both datasets, we mirror all left hands to the right.

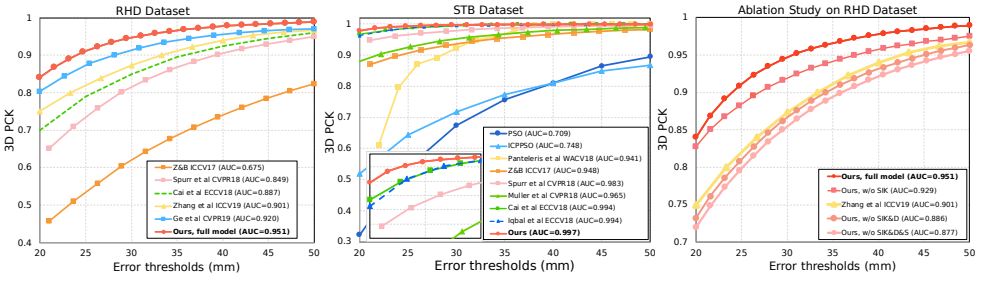

Figure 5: Quantitative results of BiHand in terms of PCK. The left and middle columns show the comparisons with state-of-the-art methods on RHD and STB dataset. The right column shows the ablation studies on RHD dataset.

Method		**Ours**	Zhou [37]	Ge [10]	Zhang [36]	Yang [34]	Baek [1]	Boukhayma [2]	Xiang [33]
AUC	STB	.997	.991	**.998**	.995	.996	.995	.994	.994
	RHD	**.951**	.893	.920	.901	.943	.926	-	-

Table 1: Additional comparisons with state-of-the-art methods in terms of AUC. " - " denotes methods that did not report the results.

Metrics We report the percentage of correct 3D keypoints (PCK) and the area under the PCK curve (AUC) as two main evaluation metrics. The distance thresholds of PCK ranges from 20 *mm* to 50 *mm*.

4.2 Implementation Details

We implement our framework using PyTorch and train it on two NVIDIA GTX 1080Ti graphic cards. The network's weights are initialized by *kaiming_normal* in PyTorch. For both datasets, we crop the image to be centered at hand and resize it to the input resolution 256×256. During training, input images from RHD and STB are mixed into the same batch.

The training process falls under the multi-task learning scheme. The hyper-parameters $\{\lambda_{\mathcal{H}}, \lambda_{\mathcal{S}}, \lambda_{\mathcal{J}}, \lambda_{\mathcal{D}}, \lambda_{\mathcal{Q}}, \lambda_{\beta}\}$ are empirically set as $\{100, 1, 1000, 1, 1, 1\}$ to balance different types of supervision. In our experiment, we first train SeedNet for 100 epochs, and then exploit its outputs to train LiftNet for another 100 epochs. In the meantime, we train SIKNet on SIK-1M dataset for 100 epochs. We use Adam optimizer and start with an initial learning rate of 10^{-4} among all experiments. The learning rate decreases to 10^{-5} at epoch 50. Finally, we fine-tune the SIKNet on the predicted 3D joints from the LiftNet. Since no ground-truth joint rotation is provided in STB and RHD dataset, only the shape loss $\mathcal{L}_{\beta}$ and joint loss $\mathcal{L}_{\mathcal{J}}$ are used during fine-tuning. The fine-tuning process starts with the leaning rate of 10^{-5} and lasts for 50 additional epochs.

4.3 Results

We evaluate our method both quantitatively and qualitatively. For quantitative results, we compare our methods in terms of PCK and AUC with state-of-the-art methods on both RHD and STB dataset. As is shown in Fig. 5 and and Table 1, on RHD dataset, our method

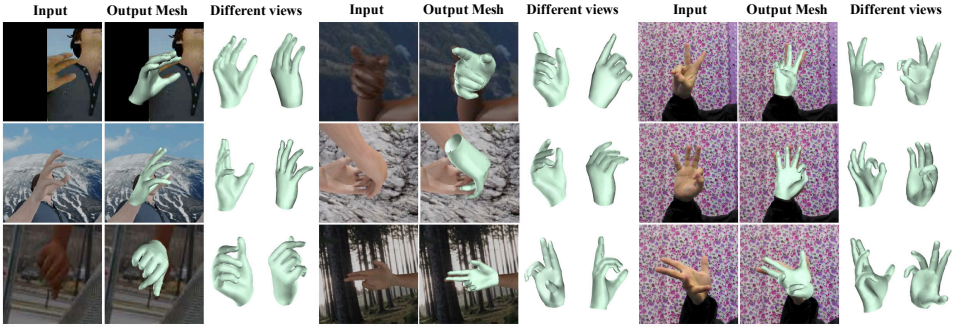

Figure 6: Recovered meshes on RHD (left, middle) and STB (right) dataset.

outperforms those methods in [1, 3, 10, 34, 36, 37, 38] over all error thresholds. On STB dataset, our method is also competitive with those in [1, 2, 3, 10, 13, 19, 22, 27, 33, 34, 36, 37, 38]. We also show the qualitative results through the generated mesh in Fig. 6. We intentionally choose the images that are self-occluded, truncated, or in poor lighting conditions. The results show that our method is competent to generate appealing and robust hand meshes.

4.4 Ablation Study

We conduct this ablation study to better understand the impact of different components in our network. Specifically, we evaluate: 0) our full architecture (**ours: full model**) in comparison to another three settings: 1) without SIKNet (**ours: w/o SIK**); 2) without SIKNet and depth predictor (**ours: w/o SIK&D**); 3) without SIKNet, depth and silhouette predictor (**ours: w/o SIK&D&S**). The following comparisons are controlled by only one variable. 0) *vs* 1) shows the accuracy improvement of adding SIKNet. 1) *vs* 2) and 2) *vs* 3) also show the improvement of employing depth and silhouette decoder as bisected branch, respectively. These experiments are conducted only on RHD dataset since the STB contains little variation and is easily saturated. As a baseline, we choose the result reported by Zhang *et al*. [36]. It is shown in Fig. 5 (right column) that our method outperforms theirs by a large margin. The result is consistent with our expectation that employing silhouette and depth map predictor can greatly help lift the pose estimation from 2D to 3D, and employing hand pose prior through SIKNet can further correct noise estimation.

5 Conclusion

In this paper, we proposed a novel method to address the challenging task of 3D hand mesh reconstruction from a single RGB image, named BiHand. Since the mesh is highly irregular and hard to be directly optimized, the procedures in BiHand are decoupled into 3 stages, i.e. 2D seeding, 3D lifting, and mesh generation stage. We have argued that 2D keypoints and silhouette, 3D joints and depth map, joint rotations and shape parameters, are closely related yet different in details, thus we adopted a stacked bisected design across the whole network. Experiments have showed the efficacy of such design on both 3D hand estimation and mesh reconstruction. In the future, we will adapt BiHand to hand interacting with an object, and further exploit the bisected design in that scenario.

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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1067–1076, 2019.

[2] Adnane Boukhayma, Rodrigo de Bem, and Philip HS Torr. 3d hand shape and pose from images in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10843–10852, 2019.

[3] Yujun Cai, Liuhao Ge, Jianfei Cai, and Junsong Yuan. Weakly-supervised 3d hand pose estimation from monocular rgb images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 666–682, 2018.

[4] Liangjian Chen, Shih-Yao Lin, Yusheng Xie, Yen-Yu Lin, Wei Fan, and Xiaohui Xie. Dggan: Depth-image guided generative adversarial networks fordisentangling rgb and depth images in 3d hand pose estimation. In *The IEEE Winter Conference on Applications of Computer Vision*, pages 411–419, 2020.

[5] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, pages 3844–3852, 2016.

[6] Quentin Delamarre and Olivier Faugeras. Finding pose of hand in video images: a stereo-based approach. In *Proceedings Third IEEE International Conference on Automatic Face and Gesture Recognition*, pages 585–590. IEEE, 1998.

[7] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3593–3601, 2016.

[8] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. 3d convolutional neural networks for efficient and robust hand pose estimation from single depth images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1991–2000, 2017.

[9] Liuhao Ge, Yujun Cai, Junwu Weng, and Junsong Yuan. Hand pointnet: 3d hand pose estimation using point sets. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8417–8426, 2018.

[10] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3d hand shape and pose estimation from a single rgb image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 10833–10842, 2019.

[11] Haiying Guan, Jae Sik Chang, Longbin Chen, Rogério Schmidt Feris, and Matthew Turk. Multi-view appearance-based 3d hand pose estimation. In *2006 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06)*, pages 154–154. IEEE, 2006.

[12] Yana Hasson, Gul Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J Black, Ivan Laptev, and Cordelia Schmid. Learning joint reconstruction of hands and manipulated objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11807–11816, 2019.

[13] Umar Iqbal, Pavlo Molchanov, Thomas Breuel Juergen Gall, and Jan Kautz. Hand pose estimation via latent 2.5 d heatmap regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 118–134, 2018.

[14] Sameh Khamis, Jonathan Taylor, Jamie Shotton, Cem Keskin, Shahram Izadi, and Andrew Fitzgibbon. Learning an efficient model of hand shape variation from depth images. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[15] Mia Kokic, Danica Kragic, and Jeannette Bohg. Learning to estimate pose and shape of hand-held objects from rgb images. *arXiv preprint arXiv:1903.03340*, 2019.

[16] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015.

[17] Jameel Malik, Ahmed Elhayek, Fabrizio Nunnari, Kiran Varanasi, Kiarash Tamaddon, Alexis Heloir, and Didier Stricker. Deephps: End-to-end estimation of 3d hand pose and shape by learning from synthetic depth. In *2018 International Conference on 3D Vision (3DV)*, pages 110–119. IEEE, 2018.

[18] 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 *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 1284–1293, 2017.

[19] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–59, 2018.

[20] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European conference on computer vision*, pages 483–499. Springer, 2016.

[21] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC 2011*, 2011.

[22] Paschalis Panteleris, Iason Oikonomidis, and Antonis Argyros. Using a single rgb frame for real time 3d hand pose estimation in the wild. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 436–445. IEEE, 2018.

[23] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-fine volumetric prediction for single-image 3d human pose. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7025–7034, 2017.

[24] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 652–660, 2017.

[25] Edoardo Remelli, Anastasia Tkach, Andrea Tagliasacchi, and Mark Pauly. Low-dimensionality calibration through local anisotropic scaling for robust hand model personalization. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[26] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics (ToG)*, 36(6): 245, 2017.

[27] Adrian Spurr, Jie Song, Seonwook Park, and Otmar Hilliges. Cross-modal deep variational hand pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[28] Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun. Cascaded hand pose regression. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[29] Anastasia Tkach, Mark Pauly, and Andrea Tagliasacchi. Sphere-meshes for real-time hand modeling and tracking. *ACM Transactions on Graphics (ToG)*, 35(6):1–11, 2016.

[30] Denis Tome, Chris Russell, and Lourdes Agapito. Lifting from the deep: Convolutional 3d pose estimation from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2500–2509, 2017.

[31] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics (ToG)*, 33(5):1–10, 2014.

[32] Etsuko Ueda, Yoshio Matsumoto, Masakazu Imai, and Tsukasa Ogasawara. Hand pose estimation using multi-viewpoint silhouette images. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, volume 4, pages 1989–1996. IEEE, 2001.

[33] Donglai Xiang, Hanbyul Joo, and Yaser Sheikh. Monocular total capture: Posing face, body, and hands in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[34] Linlin Yang, Shile Li, Dongheui Lee, and Angela Yao. Aligning latent spaces for 3d hand pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2335–2343, 2019.

[35] Jiawei Zhang, Jianbo Jiao, Mingliang Chen, Liangqiong Qu, Xiaobin Xu, and Qingxiong Yang. A hand pose tracking benchmark from stereo matching. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 982–986. IEEE, 2017.

[36] Xiong Zhang, Qiang Li, Hong Mo, Wenbo Zhang, and Wen Zheng. End-to-end hand mesh recovery from a monocular rgb image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2354–2364, 2019.

[37] Yuxiao Zhou, Marc Habermann, Weipeng Xu, Ikhsanul Habibie, Christian Theobalt, and Feng Xu. Monocular real-time hand shape and motion capture using multi-modal data. *arXiv preprint arXiv:2003.09572*, 2020.

[38] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4903–4911, 2017.

[39] 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 images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 813–822, 2019.