Joint Hand-object 3D Reconstruction from a Single Image with Cross-branch Feature Fusion

Yujin Chen, Zhigang Tu, Di Kang, Ruizhi Chen, Linchao Bao, Zhengyou Zhang, Junsong Yuan

***Abstract*—Accurate 3D reconstruction of the hand and object shape from a hand-object image is important for understanding human-object interaction as well as human daily activities. Different from bare hand pose estimation, hand-object interaction poses a strong constraint on both the hand and its manipulated object, which suggests that hand configuration may be crucial contextual information for the object, and vice versa. However, current approaches address this task by training a two-branch network to reconstruct the hand and object separately with little communication between the two branches. In this work, we propose to consider hand and object jointly in feature space and explore the reciprocity of the two branches. We extensively investigate cross-branch feature fusion architectures with MLP or LSTM units. Among the investigated architectures, a variant with LSTM units that enhances object feature with hand feature shows the best performance gain. Moreover, we employ an auxiliary depth estimation module to augment the input RGB image with estimated depth map, which further improves the reconstruction accuracy. Experiments conducted on public datasets demonstrate that our approach significantly outperforms existing approaches in terms of the reconstruction accuracy of objects.**

***Index Terms*—hand pose and shape estimation; 3D object reconstruction; hand-object interaction**

I. Introduction

There has been a growing interest in analyzing human hands from images, due to a wide range of immersive applications, e.g., virtual reality (VR), augmented reality (AR) and human-computer interaction. Significant progresses on hand detection [1]–[5] and hand pose estimation [6]–[10] have been witnessed. However, most of the existing studies assume bare hands, which is often not the case when hands are interacting with some objects. Recent research began to address the problem by capturing 3D relationships between hands and objects, which can facilitate better understanding of hand-object interactions in hand-related motion recognition, human action interpretation, robotic behavior imitation, etc.

To capture and understand hand-object interaction, there are two commonly used methods to generate 3D shapes from 2D image inputs. One is to restore 3D hand pose and 6D object pose [12], and the 3D relationship can be inferred from sparse hand keypoints and object bounding boxes [13], [14]. Another way is to directly reconstruct the pose and shape of hand and object, so that more detailed information can be obtained, such as the contact between the surfaces. In this case, the 3D model of human hands can be restored by estimating hand pose and using a shape prior model [15]–[17]. And if a 3D model of the object is available, we can retrieve the object shape from object database and align it with the predicted 6D pose into camera coordinate system [18], [19].

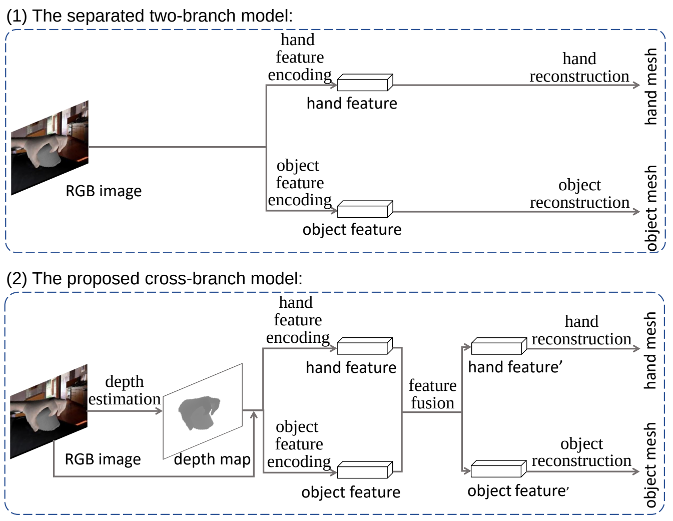


Fig. 1. Overview of the separated two-branch model [11] and the proposed cross-branch model for joint hand-object reconstruction.

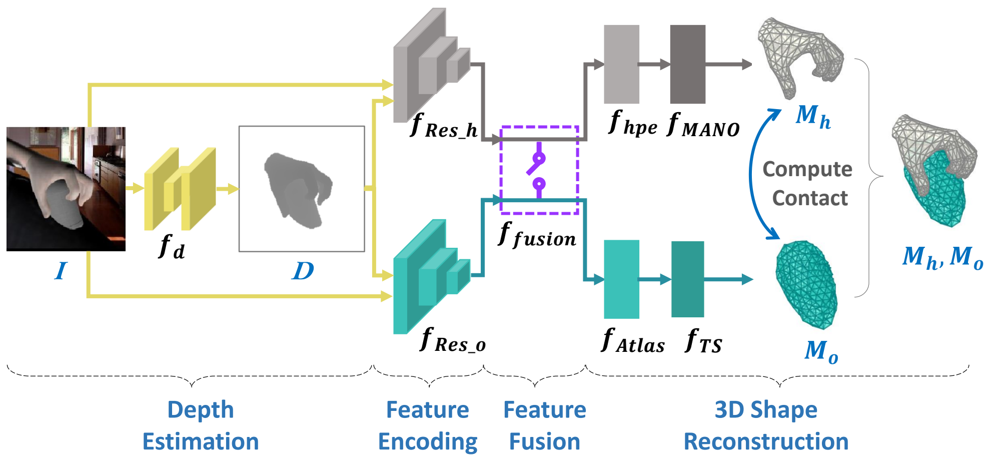
The goal of our work is to recover high-quality 3D hand and object meshes from a single hand-object image without knowing object categories. This task has been studied previously [11], [18], but the state-of-the-art performance is still far from satisfactory due to the following challenges. First, recovering 3D shape from a single-view image is an ill-posed problem, and we need to infer the missing depth information from 2D observations. Second, the severe mutual occlusions between hands and objects coupled with inter-finger occlusions make the reconstruction even more difficult. Third, physical constraints need to be considered to avoid inferring infeasible interactions. For example, surface penetration is usually not allowed and physical contacts are required when a hand holds an object. However, it is challenging for the network to discover these constraints from the training data without providing domain knowledge.

Different from existing methods [11], [18] that reconstruct hand shape and object shape separately using state-of-the-art networks from these two individual tasks, we jointly estimate their 3D shapes and explore the reciprocity of two reconstruction tasks. Specifically, as shown in Fig. 1, our method

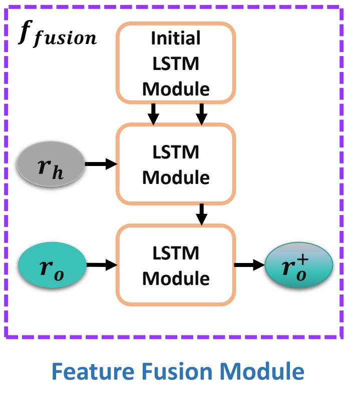
Y. Chen, Z. Tu and R. Chen are with the State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, Wuhan 430079, China. (e-mail:{yujin.chen, tuzhigang, ruizhi.chen}@whu.edu.cn).

D. Kang, L. Bao and Z. Zhang are with Tencent AI Lab, Shenzhen 518057, China (e-mail:{dkang, linchaobao, zhengyou}@tencent.com).

J. Yuan is with the Computer Science and Engineering Department, University at Buffalo, Buffalo, NY 14228, USA. (email: jsyuan@buffalo.edu).



(A)



(B)

Fig. 2. (A): The architecture of our proposed network. First, a depth map D is estimated from the input image I through the depth estimation module f_d. Then the depth map D, concatenated with the input image I, is fed into the rest network. Two separate encoders, f_{Res_h} and f_{Res_o}, are used to extract the hand and object feature $\boldsymbol{r_h}$ and $\boldsymbol{r_o}$, respectively. A feature fusion module f_{fusion} is then employed to perform cross-branch communication. Finally, the fused features are used to reconstruct 3D meshes M_h and M_o. (B): A variant of the feature fusion module f_{fusion}: The hand feature $\boldsymbol{r_h}$ and the object feature $\boldsymbol{r_o}$ are sequentially fed into LSTM modules to obtain the fused hand-aware object feature $\boldsymbol{r_o^+}$. Several other variants are presented in Fig. 3.

first augments the RGB image into RGB-D representation by estimating the corresponding depth map and then feeds the RGB-D image into the feature encoding branches for hand and object individually. Inspired by the "Block LSTM" used for sequential blocks connection in [20], we use the connection information of structural components to benefit the joint reconstruction task. We modify the parallel joint reconstruction task by adding a step-by-step feature fusion process that propagates features in the latent space through a long short-term memory (LSTM) [21] block. Generally, for different scenarios and samples, the shape of hands conforms to some priors while the shape of objects can be more varied. In our setup, we found the hand part is relatively more robust than the object part because the hand part employs a low-dimensional parametric model learned from more than two thousand 3D scans [17], while the object part uses less constrained vertex positions. Therefore, we use the more robust hand feature to enhance the object feature through an LSTM feature fusion module. Specifically, the hand feature and the object feature enter the LSTM module sequentially (see Fig.2-B) so that the later entered object feature is modulated by the LSTM state, which stores information from the hand feature. We further investigate several alternative feature fusion strategies in Section IV-C and demonstrate the effectiveness of the LSTM-based module. Finally, the hand feature and the enhanced object feature are decoded to regress 3D shapes of hand and object.

Our main contributions are summarized as follows:

- We design an end-to-end cross-branch network that significantly improves accuracy over existing methods for reconstructing 3D shapes of hand and the interacting object from a single RGB image. The code and model will be publicly available.
- We propose the first approach to extensively investigate the feature fusion scheme for jointly reconstructing hands and objects.
- We introduce a depth estimation module to augment the RGB input into an RGB-D image, which effectively brings additional performance gains.

II. Related Work

This work is primarily related to the 3D reconstruction of hands and objects from hand-object interaction images. Secondly, this work is relevant to hand pose and shape estimation. Thirdly, it is also related to single view 3D object reconstruction. Moreover, the proposed LSTM feature fusion scheme is related to approaches that employ recurrent neural network (RNN) in 3D vision tasks. We briefly review the related work in this section.

3D parsing of hand-object image. Analyzing hand-object interaction in 3D is a challenging topic in computer vision, including estimating the 3D shape of hands in interaction with object [22], [23], tracking hands while it is interaction with an object [24]–[27], and reconstructing objects for in-hand scanning systems [28], etc. Recently, the development of deep learning has further advanced the progress of this topic. Recent learning-based works conduct joint hand-object pose estimation either from depth image [13] or from RGB image [29]–[31], however, sparse joint estimation is probably not sufficient for reasoning about the hand model or hand-object contact. Very recently, some works [11], [18], [19], [32] take into account 3D reconstruction of hand-object interaction. Instead of tracking hand that interacts with a deformable object as in [32], we focus on estimating 3D shapes of a hand and a rigid object. Although [18], [19] also model 3D hand-object shapes, the object categories are known in their settings. In order to adapt to more types of objects, we directly estimate object shapes and do not assume that the model of the object is pre-defined. Our proposed method is most similar to [11] which uses a two-branch network to estimate

3D representation of hand and object from the RGB image. There are two main differences between our method and [11]. First, we introduce a depth estimation module to estimate the depth map containing abundant structure information and use the RGB-D information for the 3D reconstruction. Second, the hand-object relationship has not been well explored in [11], while we introduce the connection between the two branches through the proposed LSTM feature fusion module.

Hand pose and shape estimation. Hand pose estimation has developed rapidly since the advance of commodity RGB-D sensors [33], [34]. Significant progress in 3D hand pose estimation from either RGB image [7], [8], [35]–[39] or depth map [9], [40]–[45] has been witnessed. To better display the surface information of the hand, many works focus on producing a dense hand mesh which is achieved through depth input [46] or RGB input [11], [47]–[49]. In our work, to better infer hand interactions with objects, we also focus on predicting accurate hand meshes from RGB image. The MANO hand model [17], whose parameters are regressed from a neural network, is employed to produce the hand joints and meshes.

Single-image 3D object reconstruction. Recovering 3D object shape from a single image is a challenging problem. Recent approaches have attempted to represent objects in voxels [50], [51], point clouds [52], [53], octrees [54], [55], or polygon meshes [56]–[61]. Our work closely relates to methods that represent objects by meshes. They usually generate 3D structures by deforming a set of primitive square faces [58] or a generic pre-defined mesh [59], [61]. Similar to AtlasNet [58], we also form 3D meshes by deforming a set of primitive square faces. A view-centered variant of AtlasNet is adopted to generate 3D objects from the hand-object images [11].

Recurrent neural network in 3D vision tasks. RNN is well-accepted for its effectiveness in processing temporal sequence [62], [63]. Recently, RNN becomes a powerful regulator in 3D vision tasks, such as human pose estimation [64]–[66] and 3D object reconstruction [67]. Very recently, [65] presents a ConvLSTM to allow the propagation of contextual information among different body parts for 3D pose estimation. [20] proposes 3D shape programs, which take a 3D shape as input and outputs a sequence of primitive programs, to describe the corresponding 3D shape of an object. Unlike them, our goal is not to parse an individual structure by using the contextual information among structural parts. Instead, we aim to link the information of two different individuals with an LSTM feature fusion module in our joint hand-object reconstruction task.

III. Proposed Approach

The goal of our method is to reconstruct both hand and object in mesh representation from an image where a hand is interacting with an object. We propose a two-branch branch-cooperating network to solve the problem. A depth map is predicted from the RGB image first, and then the RGB-D image, which is a concatenation of the original RGB image and the estimated depth map, is fed into the following reconstruction branches. A two-branch network is adopted to estimate hand and object shapes, and a feature fusion module bridging the two branches is proposed to enhance the individually learned hand and object features (see Fig. 2). In the following, we describe our proposed method in detail.

A. Depth Estimation

Previous works [68], [69] show that utilizing depth map can bring benefits to the monocular RGB-based shape estimation task. However, ground truth depth maps are not always available with RGB images. In this work, we employ a depth estimation module to augment the RGB input to an RGB-D input.

A depth estimation module f_d is used to estimate depth map D from the color image I: $D = f_d(I)$. Similar to [68], ResNet-18 [70] is used to encode the input image into latent representation, and a decoder is used to predict the depth map. The decoder consists of four sets of 5×5 convolutional layers and four sets of 1×1 convolutional layers, and a ReLU layer is added after each convolutional layer. The output is a one-channel depth map of the same size as the input image. We concatenate the estimated depth map D with the source RGB image I, and fed the RGB-D image into the following network.

The loss for this depth estimation task consists of two parts: the least absolute deviations loss (also referred as L1 distance) $\mathcal{L}_1$ and the structural similarity (SSIM) loss $\mathcal{L}_{ssim}$ (Eq. 1). λ_{ssim} is a weighting factor which is set to 1000 empirically. We use $\frac{1}{SSIM}$ as a loss term to encourage the structural similarity between the predicted depth map and its ground truth [71]. The depth loss $\mathcal{L}_D$ is the summation of the least absolute deviation loss $\mathcal{L}_1$ and the structural similarity loss $\mathcal{L}_{ssim}$ (Eq. 2).

$$\mathcal{L}_{ssim} = \frac{1}{\lambda_{ssim} \cdot SSIM} \tag{1}$$

$$\mathcal{L}_D = \mathcal{L}_1 + \mathcal{L}_{ssim} \tag{2}$$

Note that the depth values of the background are missing in the synthetic images, and therefore we only calculate the loss for the foreground regions during network training.

B. Feature Encoding

The raw RGB image is first concatenated with the estimated depth map, and the four-channel RGB-D image is fed into the encoders of the two-branch network. Each branch takes the RGB-D image as input and extracts a latent representation by the ResNet-18 encoder [70] pre-trained on the ImageNet [72]. The hand and object representation is denoted as $\boldsymbol{r_h} = f_{Res\_h}(I \oplus D)$ and $\boldsymbol{r_o} = f_{Res\_o}(I \oplus D)$ respectively, where $\oplus$ denotes concatenation.

C. 3D Shape Reconstruction

1) Hand pose and shape estimation: We regress model parameters from the latent representation and then use the MANO model as a network layer to predict hand pose and shape as [11], [48]. Specifically, we feed the hand feature $\boldsymbol{r_h}$ to the hand parameter estimator f_{hpe} to regress hand parameters, including pose θ and shape β: $\theta, \beta = f_{hpe}(\boldsymbol{r_h})$. Then, a differentiable MANO hand model f_{MANO} is used to compute

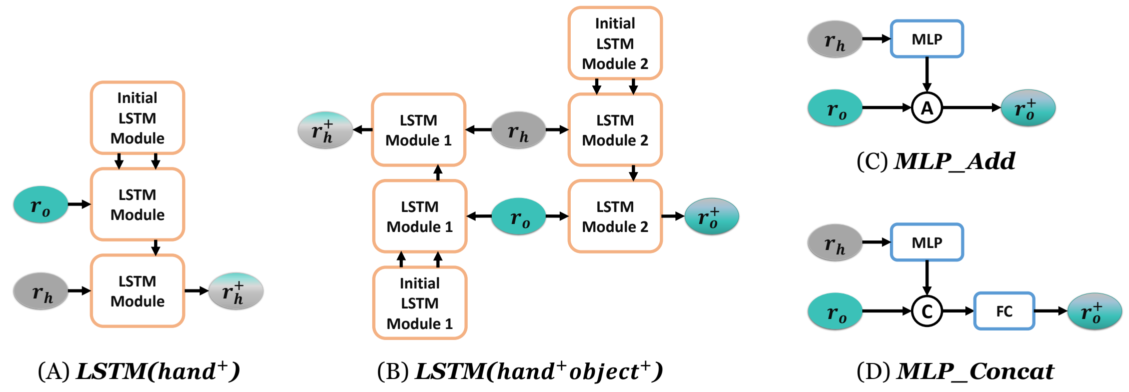


Fig. 3. Alternative feature fusion architectures. (A) ***LSTM(hand$^+$)***. (B) ***LSTM(hand$^+$object$^+$)***. (C) ***MLP_Add***. (D) ***MLP_Concat***.

hand joints J_h and hand mesh M_h according to θ and β: $J_h, M_h = f_{MANO}(\theta, \beta)$. θ determines the joint angles and β adjusts the shape of the hand (see [17] for more details).

To supervise the learning of hand estimation, we adopt a common joint position loss $\mathcal{L}_J$. Meanwhile, a vertex position loss $\mathcal{L}_M$ is used to supervise the training of hand mesh when the ground truth hand meshes are available. Both losses are calculated with the L2 distance between the predicted value and the ground truth value. The shape regularization term is defined as $\mathcal{L}_\beta = \parallel \beta - \bar{\beta} \parallel^2$ to encourage the estimated hand model shape β to be close to the average shape $\bar{\beta} = \vec{0} \in \mathbb{R}^{10}$. The pose regularization term is defined as $\mathcal{L}_\theta = \parallel \theta - \bar{\theta} \parallel^2$ to penalize the unreasonable pose which is far from the average pose $\bar{\theta} = \vec{0} \in \mathbb{R}^{30}$. In summary, the overall loss for hand part $\mathcal{L}_{Hand}$ is the weighted sum of $\mathcal{L}_M$, $\mathcal{L}_J$, $\mathcal{L}_\beta$ and $\mathcal{L}_\theta$ (Eq. 3), where μ_J, μ_β and μ_θ are the weighting factors.

$$\mathcal{L}_{Hand} = \mathcal{L}_M + \mu_J \mathcal{L}_J + \mu_\beta \mathcal{L}_\beta + \mu_\theta \mathcal{L}_\theta \tag{3}$$

2) Object shape reconstruction: Our object reconstruction branch is similar to AtlasNet [58]. In AtlasNet, the latent shape representation and a set of 2D points sampled uniformly in the unit square are fed into the network, and a collection of parametric surface elements is generated. These parametric surface elements infer a surface representation of the shape. AtlasNet is designed to reconstruct object meshes in a canonical view, and [11] has validated its effectiveness in view-centered coordinate in the hand-object task. Unlike AtlasNet, the latent representation is extracted from the RGB-D input in our method. Then the AtlasNet block f_{Atlas} decodes the latent representation $\boldsymbol{r_o}$ to an object mesh M_{co} in the object canonical coordinate system (Eq. 4). A multi-layer perceptron (MLP) neural network f_{TS} is used to predict the parameters of translation T and scale s (Eq. 5). Then, according to T and s, we translate the regressed mesh M_{co} into the final object mesh $M_o = sM_{co} + T$.

$$M_{co} = f_{Atlas}(\boldsymbol{r_o}) \tag{4}$$

$$T, s = f_{TS}(\boldsymbol{r_o}) \tag{5}$$

We use a symmetric Chamfer distance loss $\mathcal{L}_{CD}$ as [58], which measures the difference between the predicted vertices q of the regressed object mesh M_o and point set p which is uniformly sampled on the surface of ground truth object mesh M_g (Eq. 6). Except for comparing the object meshes M_o with M_g in the hand-relative coordinate, we also compute the Chamfer distance $\mathcal{L}_{CDcan}$ in the object canonical coordinate (Eq. 7), where M_{co} is the regressed object mesh in Eq. 4 and M_{cg} is normalized from the ground truth mesh M_g (Eq. 8) by the ground truth object centroid $\hat{T}$ and the maximum radius $\hat{s}$.

$$\mathcal{L}_{CD} = \frac{1}{2}\Big(\sum_{p \in M_g} \underset{q \in M_o}{min} \parallel p - q \parallel_2^2 + \sum_{q \in M_o} \underset{p \in M_g}{min} \parallel q - p \parallel_2^2\Big) \tag{6}$$

$$\mathcal{L}_{CDcan} = \frac{1}{2}\Big(\sum_{p \in M_{cg}} \underset{q \in M_{co}}{min} \parallel p - q \parallel_2^2 + \sum_{q \in M_{co}} \underset{p \in M_{cg}}{min} \parallel q - p \parallel_2^2\Big) \tag{7}$$

$$M_{cg} = (M_g - \hat{T})/\hat{s} \tag{8}$$

The translation loss is defined as $\mathcal{L}_T = \parallel T - \hat{T} \parallel_2^2$ and the scale loss is defined as $\mathcal{L}_s = \parallel s - \hat{s} \parallel_2^2$, where the ground truth object centroid $\hat{T}$ and the maximum radius $\hat{s}$ are computed in the hand-relative coordinates. An edge-regularization loss $\mathcal{L}_E$ is used to penalize edges' length difference and a curvature-regularizing loss $\mathcal{L}_L$ is used to encourage reasonably smooth surface. The overall loss for object part $\mathcal{L}_{Object}$ is the weighted sum of $\mathcal{L}_{CD}$, $\mathcal{L}_{CDcan}$, $\mathcal{L}_T$, $\mathcal{L}_s$, $\mathcal{L}_L$ and $\mathcal{L}_E$ (Eq. 9), where μ_T, μ_s, μ_L, μ_E are the weighting factors.

$$\mathcal{L}_{Object} = \mathcal{L}_{CD} + \mathcal{L}_{CDcan} + \mu_T \mathcal{L}_T + \mu_s \mathcal{L}_s + \mu_L \mathcal{L}_L + \mu_E \mathcal{L}_E \tag{9}$$

3) Connecting two branches via feature fusion module: For this hand-object reconstruction task, we propose to link the independent features of the hand and the object branches. To facilitate joint reconstruction, we propose to connect the two branches in latent space via a feature fusion module. In our model, the hand representation $\boldsymbol{r_h}$ is more robust than the object representation $\boldsymbol{r_o}$ because the hand part uses the MANO model [17], which is a low-dimensional parametric model trained on the hand data from 2018 scans, thus it contains rich prior knowledge about the valid hand space. In contrast, objects are represented using vertex positions optimized only by training data and therefore contain less shape prior information. Therefore, we choose the hand feature

$\boldsymbol{r_h}$ as the provider to enhance the object feature $\boldsymbol{r_o}$ by enabling the object branch to perceive information from the hand branch. As shown in Fig. 2-B, we employ a two-timestep LSTM f_{fusion} as the feature fusion module. The hand feature $\boldsymbol{r_h}$ is fed to the LSTM module at the first timestep, and then the state of the LSTM layer, storing the information of the current hand's pose and shape, is propagated into the object branch, resulting in enhanced hand-aware object feature $\boldsymbol{r_o^+} = f_{fusion}(\boldsymbol{r_h}, \boldsymbol{r_o})$, where $\boldsymbol{r_o}$ is the original object feature. The two-timestep LSTM uses one recurrent layer and the number of features in the hidden state is 1000. After adding the feature fusion module, the object representation $\boldsymbol{r_o}$ is replaced by the enhanced object feature $\boldsymbol{r_o^+}$ in Eq. 4 and Eq. 5. To better explore the effectiveness of the feature fusion module, we investigate several fusion strategies (Fig. 3) and compare their performance in Section IV-C.

Additionally, we take into account the contact between the hand and the object when recovering their meshes. Different from 3D shape reconstruction from a hand-only or object-only image, jointly reconstructing the hand and the object needs to deal with the contact problem. [11] formulates the contact constraint as a differentiable loss $\mathcal{L}_{Contact}$, which consists of an attraction term and a repulsion term. We adopt the same contact loss, and more details can be found in [11]. The overall loss function $\mathcal{L}$ of our proposed network is the weighted sum of the above mentioned four parts (Eq. 10), where the μ_H, μ_O, μ_C are the weighting factors.

$$\mathcal{L} = \mathcal{L}_D + \mu_H \mathcal{L}_{Hand} + \mu_O \mathcal{L}_{Object} + \mu_C \mathcal{L}_{Contact} \tag{10}$$

IV. Experiments

In this section, we first present datasets, evaluation metrics (Section IV-A), and implementation details (Section IV-B) of our experiments. Then, we analyze the effect of the proposed modules (Section IV-C) and evaluate the overall performance of our method (Section IV-D).

A. Datasets and Evaluation Metrics

We evaluate our method on two publicly available datasets: a synthetic third-person perspective dataset and a real-world egocentric dataset.

ObMan Dataset: The ObMan dataset [11] is a large-scale synthetic dataset containing hand-object images, in which the hand is generated through the MANO model [17] and the object is sampled from the ShapeNet dataset [73]. More than 2000 object meshes are presented in the ObMan dataset whose objects are selected from eight categories in the ShapeNet dataset. The plausible grasps are generated by using the GraspIt software [74]. The ObMan dataset contains 141K training frames and 6K test frames. For each frame, it provides the RGB-D image, 3D hand and object meshes, and 3D hand keypoints.

First-Person Hand Action Benchmark Dataset (FHB): The FHB dataset [75] collects RGB-D video sequences of daily hand action categories with hand-object interaction. 3D hand joints and 6D object poses are automatically annotated. Following [11], a subset of FHB named FHBc containing around 5K frames for training and 5.6K frames for testing is used in the experiments.The same object categories appear in both the training and test set.

To evaluate the hand estimation results, we use three metrics: (i) MPJPE: the mean per joint position error (MPJPE) in the Euclidean space for all joints on all test frames; (ii) HME: the hand mesh error (HME) is the average error in the Euclidean space between the corresponding vertices of the hand mesh on all test frames; (iii) 3D PCK: the percentage of correct key points (PCK) whose joint error does not exceed a certain threshold.

To evaluate the object reconstruction results, we report the surface-to-surface distance between the predicted mesh and the ground truth mesh. Specifically, the symmetric Chamfer distance (CD) (Eq. 6) is calculated on 600 points sampled on the ground truth mesh and all 642 vertices of the estimated mesh [11], [58].

To evaluate function of the contact between the hand and the object, we report the penetration depth (PD) which is the maximum penetration between the hand and the object averaged on all the test frames as [11].The maximum penetration is defined as the maximum distance between the surfaces of the two meshes in the intersection space if exist. Otherwise, the error is 0 if the two meshes do not penetrate each other.

B. Implementation Details

We use Adam [76] to optimize the network with batch size 16. The proposed network consists of three parts, i.e., the depth estimation part, the hand branch, and the object branch (together with the feature fusion module). Instead of directly optimizing the whole network, we adopt a stage-wise training strategy to train the network.

To train our model on the ObMan dataset, we divide the training procedure into several stages. We first train the depth estimation part for 90 epochs, we initialize the learning rate to 3×10^{-4} and decrease it to 10^{-4} and 10^{-5} after every 30 epochs. Then the depth estimation part is frozen. The hand branch is trained for 250 epochs with the learning rate initialized to 10^{-4} and decreased to 10^{-5} at epoch 200. After that, the hand branch is frozen and the object branch is trained in the same procedure as the hand branch. Finally, we fine-tune the hand and object branches with the contact loss for 50 epochs by setting the learning rate to 10^{-5}.

For the small FHBc dataset, the same training procedure is used as that on the ObMan dataset except that the hand and object branches are initialized with the weights pre-trained on the ObMan dataset. The two-branch 3D reconstruction part is trained for 150 epochs with learning rate deceased at epoch 100. The weight of the depth map estimation part is randomly initialized since the regular depth map is used in the FHBc dataset while the depth foreground map is used in the ObMan dataset.

For the hand estimation Eq. 3, the weighting factors are set as $\mu_J = 0.167$, $\mu_\beta = 0.167$ and $\mu_\theta = 0.167$. For the object shape reconstruction Eq. 9, $\mu_T = 0.167$, $\mu_s = 0.167$, $\mu_L = 0.1$ and $\mu_E = 2$. For the overall loss function Eq. 10, we set $\mu_H = 0.001$, $\mu_O = 0.001$ and $\mu_C = 0.1$.

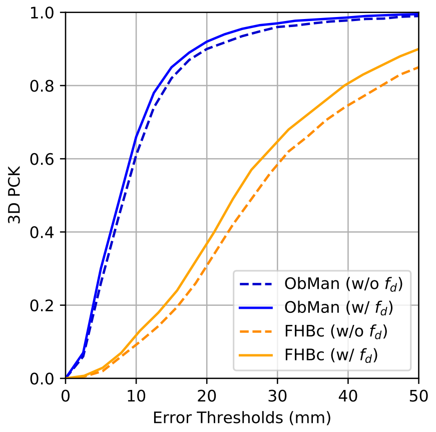


Fig. 4. Comparison of whether to use the depth estimation module on the FHBc dataset and the ObMan dataset. The proportion of correct keypoints over different error thresholds are presented.

C. Ablation Study

We present an ablative analysis to evaluate the effectiveness of the proposed modules. We analyze the effects of depth estimation, feature fusion module, two-branch pipeline, and contact computation. Table III shows the results of different versions of our model.

*1) **Effect of depth estimation module:*** To illustrate the effect of depth estimation, we conduct a comparison on both datasets to evaluate the hand estimation task. For the setting without the depth estimation, we use an empty map to replace the estimated depth map and leave other components unchanged. We evaluate the performance by the percentage of correct keypoints (PCK). As shown in Fig. 4, the performance is improved compared to the method without the depth estimation module. From the curves, we can see that the PCK is improved by about 10% compared to the method without the depth estimation module when the error threshold is between 20mm and 40mm on the FHBc dataset, and the PCK is improved by about 3% when the error threshold is between 10mm and 30mm on the ObMan dataset. Apart from evaluating the improvement on hand estimation, we also report the improvement on object reconstruction of the ObMan dataset. Comparing index 2 and index 5 in Table III, after removing the depth estimation module f_d from index 2, the object CD will increase by 4%, the hand MPJPE will increase by 6.2%, and the hand HME will increase by 6.9%. Intuitively, the depth map is complementary to RGB image as it provides abundant and more direct structural information, which is beneficial for the inference of 3D reconstruction. It is worth mentioning that the introduced depth estimation module does not need additional ground-truth information during training, as its supervision can be derived from ground-truth 3D meshes.

*2) **Comparison of different feature fusion modules:*** In this section, we conduct experiments to validate the effectiveness of the feature fusion module. Several alternative architectures of the module are investigated. When no fusion is used, the hand branch and the object branch are trained separately (see ***Baseline*** in Table I). Three fusion strategies are tested based on the provider and receiver of the cross-branch information. (i) ***LSTM($object^+$)***: LSTM means the proposed LSTM

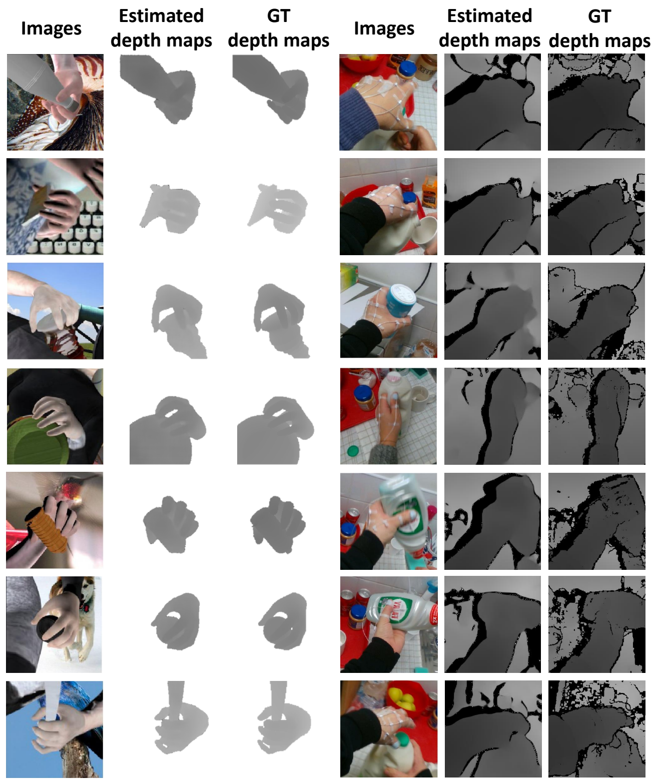


Fig. 5. Qualitative visualization of the depth estimation on the ObMan dataset (left) and the FHBc dataset (right).

feature fusion module is used to fuse the features and "+" after object means the object feature is enhanced by fusing information from the other branch, which is the hand branch in this case (See Fig. 2-B). (ii) ***LSTM($hand^+$)***: The hand feature is enhanced by fusing information from the object branch via the LSTM feature fusion module (See Fig. 3-A); (iii) ***LSTM($hand^+object^+$)***: We use two LSTM modules, one to enhance object by hand and vice versa (See Fig. 3-B). Unlike the branch-wise training scheme in ***LSTM($hand^+$)*** and ***LSTM($object^+$)***, the hand and object branches are trained together with mutual enhancement. As shown in Table I, the object CD error of ***LSTM($object^+$)*** decreases by 5% with the object feature enhanced by perceiving information from the hand branch, and the MPJPE of ***LSTM($hand^+$)*** decreases by 2% with the hand feature enhanced. When two individual LSTM feature fusion modules are used to enhance both features (as ***LSTM($hand^+object^+$)***, the results show that mutual fusion helps to improve the shape of the object but makes the hand worse. We think that the more robust hand feature helps the object feature when they are fused and trained with each other. Meanwhile, the object feature drags the hand feature down to a certain extent, resulting in slightly worse hand estimation and better object estimation. As for the choice of the direction of information transmission, we think it is a trade-off between improvements of the hand and the object. The ***LSTM($object^+$)*** is chosen in this paper because both hand and object get good results.

Apart from the proposed LSTM feature fusion module, we conduct ablative experiments to study the commonly-used MLP-based feature fusion modules, including concatenation

TABLE I
EVALUATION RESULTS ON THE OBMAN DATASET IN TERMS OF THE FUSION STRATEGY AND WHETHER TO PERFORM CONTACT COMPUTATION.

Method	MPJPE (mm) ↓	CD (mm) ↓
Baseline	9.7	426.1
Baseline+LSTM(hand$^+$)	9.5	426.1
Baseline+LSTM(object$^+$)	9.7	405.1
Baseline+LSTM(hand$^+$ object$^+$)	10.1	392.6
Baseline+contact	9.7	422.5
Baseline+LSTM(object$^+$)+contact **(Ours)**	9.6	403.8

TABLE II
OBJECT RECONSTRUCTION RESULTS ON THE OBMAN DATASET WITH DIFFERENT FEATURE FUSION MODULES.

Method	CD (mm) ↓
Baseline	426.1
Baseline+MLP_Add(object$^+$)	437.2
Baseline+MLP_Concat(object$^+$)	428.1
Baseline+LSTM(object$^+$)	405.1
Baseline+LSTM_2layers(object$^+$)	402.3

TABLE III
ABLATION STUDIES FOR DIFFERENT COMPONENTS USED IN OUR METHOD ON THE OBMAN DATASET. THE DEPTH ESTIMATION MODULE f_d CONTRIBUTED THE MOST TO THE HAND SHAPE AND THE FEATURE FUSION MODULE f_{fusion} CONTRIBUTES THE MOST TO THE OBJECT SHAPE.

Index	f_d	Separate Encoders	f_{fusion}	Contact	MPJPE (mm) ↓	HME (mm) ↓	CD (mm) ↓
1	✓	✓	✓	✓	**9.6**	**9.8**	**403.8**
2	✓	✓	✓	×	9.7	10.1	405.1
3	✓	✓	×	×	9.7	10.0	426.1
4	✓	×	×	×	10.5	10.8	404.9
5	×	✓	✓	×	10.3	10.8	421.3
6	×	✓	×	×	10.3	10.8	860.6

TABLE IV
COMPARISON OF MPJPE (MM) ON THE OBMAN DATASET AND THE FHBC DATASET.

Method	FHBc	ObMan
[11](w/o contact)	28.1	11.6
[11](w/ contact)	28.8	11.6
Ours(w/o contact)	27.9	9.7
Ours(w/ contact)	**27.5**	**9.6**

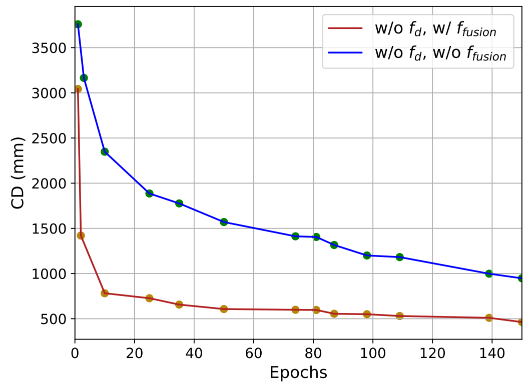


Fig. 6. Ablation of the LSTM fusion module on the ObMan dataset. This plot shows the first 150 epochs of training: the f_{fusion} makes the object CD error in the evaluation set decrease fast.

and element-wise addition. We also use a stacked LSTM in the LSTM module to see if more capacity can further improve the performance. Three alternative feature fusion strategies are tested to enhance the object feature using information from the hand feature. (i) ***MLP_Add(object$^+$)***: The object feature is enhanced via element-wise addition with information from the hand branch through a MLP mapping (See Fig. 3C); (ii) ***MLP_Concat(object$^+$)***: The object feature is enhanced via concatenation with information from the hand branch through an MLP mapping, and then the enhanced object feature is converted to a fixed size vector via a fully connected (FC) layer (See Fig. 3D); (iii) ***LSTM_2layers(object$^+$)***: The same feature fusion method is used as in ***LSTM(object$^+$)***, while two LSTMs are stacked together to form a stacked LSTM in the LSTM module. As shown in Table II, the object reconstruction results become worse after using the MLP-based modules, while the ***LSTM*** and ***LSTM_2layers*** can greatly improve the performance of the object reconstruction. Note that in our experiments, MLP-based modules use similar parameters as in ***LSTM(object$^+$)***, while ***LSTM_2layers(object$^+$)*** uses double parameters. Among these modules, the LSTM-based modules are most helpful and the LSTM module stacks two LSTMs is a little better than another but it has double parameters in the feature fusion module. We use ***LSTM(object$^+$)*** in this paper.

As shown in Fig 6, we compare the decreasing speed of the object CD in the evaluation set. To better evaluate the effect of the feature fusion module ***LSTM(object$^+$)***, the depth estimation module is removed. It can be seen from the curve that the proposed feature fusion module greatly accelerates the speed of the object CD error reduction. Finally, if both the depth estimation module and the feature fusion module are removed from our model (shown as index 6 in Table III), the reconstruction performances of both hands and objects will drop dramatically.

3) ***Effect of two-branch encoding:*** In addition to the above methods of encoding features with two encoders and then fusing cross-branch features, we also implemented an extreme bridging method, that is, the hand branch and the object branch shares the same features encoded by one encoder. In this setup, a ResNet-34 is used as a shared encoder f_{Res_s} in order to balance the total trainable parameters to make a fairer comparison. As present in Table III (index 4), we use a shared backbone and remove the hand encoder, object encoder and the fusion module from index 2. By replacing the separate encoders with the shared backbone network, comparable results can be obtained on object reconstruction, while the hand MPJPE increased by 10%. We think that each branch specializes in its task while the fusion module helps with perceiving cross-branch information in the two-branch branch-cooperation pipeline, thus the two-branch pipeline is more efficient than the shared encoder setup for this joint hand-object reconstruction task.

4) ***Contact loss and its connection to feature fusion module:*** To study the effect of the feature fusion module and the contact term, different modules are

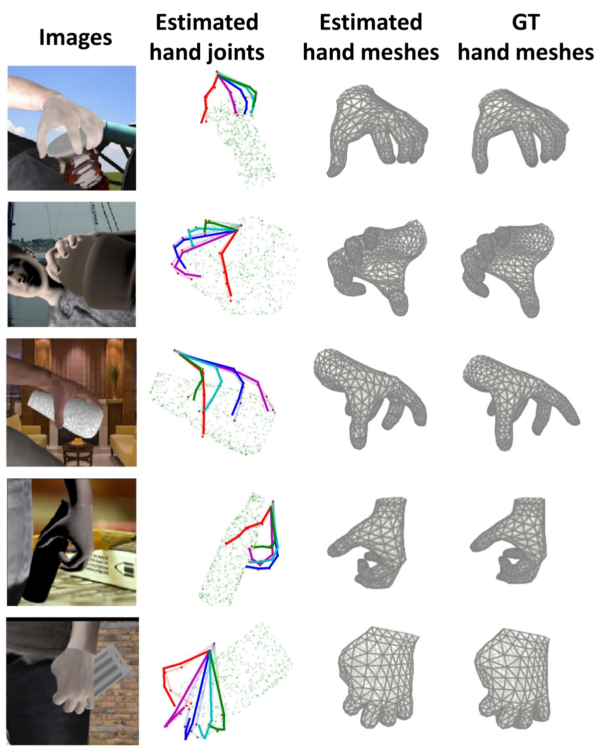


Fig. 7. Qualitative comparison between our estimated hand meshes and ground truth meshes on the ObMan dataset. In the column of hand joints, we use the skeleton with opaque color to represent the estimated hand joints and the skeleton in semi-transparent color to represent the ground truth. Green points are randomly sampled from the ground truth object meshes. Results show that the estimated hand joints and meshes are close to the ground truth. Note that scales are changed in the visualization.

employed in the experiments. ***Baseline+contact***: We train the hand and object branches without feature fusion as the ***Baseline*** and then add a contact loss to fine-tune both branches; ***Baseline+LSTM(object$^+$)+contact (Ours)***: +contact means a contact loss is used to fine-tune both the branches. Note that each stage has been trained to convergence. As shown in Table I, comparing ***Baseline+contact***, ***Baseline+LSTM(object$^+$)*** and ***Baseline+ LSTM(object$^+$)+contact (Ours)***, the object error decreases by 0.8%, 4.9% and 5.2% accordingly. The proposed LSTM module and the contact computing both are "bridges" to connect hand and object, the results reveal that our proposed LSTM block is more helpful than the contact loss in reducing the object reconstruction error, and the combination of the LSTM module and the contact loss effectively improves performance. Both the proposed LSTM feature fusion module and the contact constraint link two branches of hand and object, but they achieve this goal in different spaces. The feature fusion module connects branches in the feature space while the contact constraint is implemented directly on the meshes. We think that the connection in the feature space is easier to learn, so the proposed LSTM module contributes more than the contact loss. In summary, the LSTM block can cooperate well with the contact term and helps to get better shapes from the hand-object image, and ***Baseline+LSTM(object$^+$)+contact (Ours)*** achieves best overall performance on hand-object reconstruction task.

TABLE V
COMPARISON OF 3D OBJECT RECONSTRUCTION WITH [11] ON THE OBMAN SYNTHETIC DATASET. CD IS THE SYMMETRIC CHAMFER DISTANCE. PD IS THE PENETRATION DEPTH.

Method	CD (mm) ↓	PD (mm) ↓
[11](w/o contact)	641.5	9.5
[11](w/ contact)	637.9	9.2
Ours(w/o contact)	405.1	9.6
Ours(w/ contact)	**403.8**	**9.1**

D. *Results*

1) Performance of depth estimation: For depth estimation, we use a simple encoder-decoder structure to directly predict a depth image from the input monocular RGB input. We estimate depth foregrounds on the ObMan synthetic dataset and produce regular depth maps on the FHBc dataset. The qualitative results are shown in Fig. 5. We find that the estimated depth images preserve intrinsic information about hand-object shape with smooth surfaces. Note that when training the depth estimation part, the weights of the network are randomly initialized on both datasets.

2) Performance of hand estimation: We compare the hand estimation result of our method with the state-of-the-art method [11]. The mean error of all joints over all test frames is presented in Table IV. On the challenging FHBc dataset which only provides a small amount of similar images, our method outperforms [11] in both settings. Our method obtains hand pose error of 27.9mm while the baseline method [11] is 28.1mm under the setting without contact loss. With the contact loss, our method obtains an error decreasing by 4.5% (28.8mm vs 27.5mm), while the MPJPE of [11] increased by 2.5%. On the large-scale ObMan dataset, the performance of our method is 16.4% better than [11], and we obtain a MPJPE of 9.7mm while their MPJPE is 11.6mm. It is worth noting that after adding the contact loss, the accuracy of hand joints remains unchanged or slightly decreases in [11], while it has an obvious improvement in our method. We think this is because the proposed LSTM feature fusion module helps the network reconstructs shapes with more reasonable relative positions, and the shapes with better relative positions can better initialize the fine-tuning stage with the contact loss. We present visualized samples from the testing set in Fig. 7.

3) Performance of object estimation: The ObMan dataset [11] provides precise object annotations, and we report the object reconstruction accuracy on the ObMan dataset in Table V. When no contact loss is used, the object reconstruction result of ours is already much better than [11], obtaining a significant error decrease by 36.8% from 641.5mm to 405.1mm on CD. The object CD error further decreases from 405.1mm to 403.81mm after fine-tuning our network with the contact computation. The object CD error is 36.7% lower than [11] under the setting with contact computation. This great improvement in object reconstruction reveals the remarkable effect of our proposed method.

4) Results on shape penetration: We report the penetration depth between the hand and object meshes. In physical space, to obey the rule that hand and object should not share the same space, good shapes of hand and object show low penetration depth. As shown in Table V, the penetration depth of our

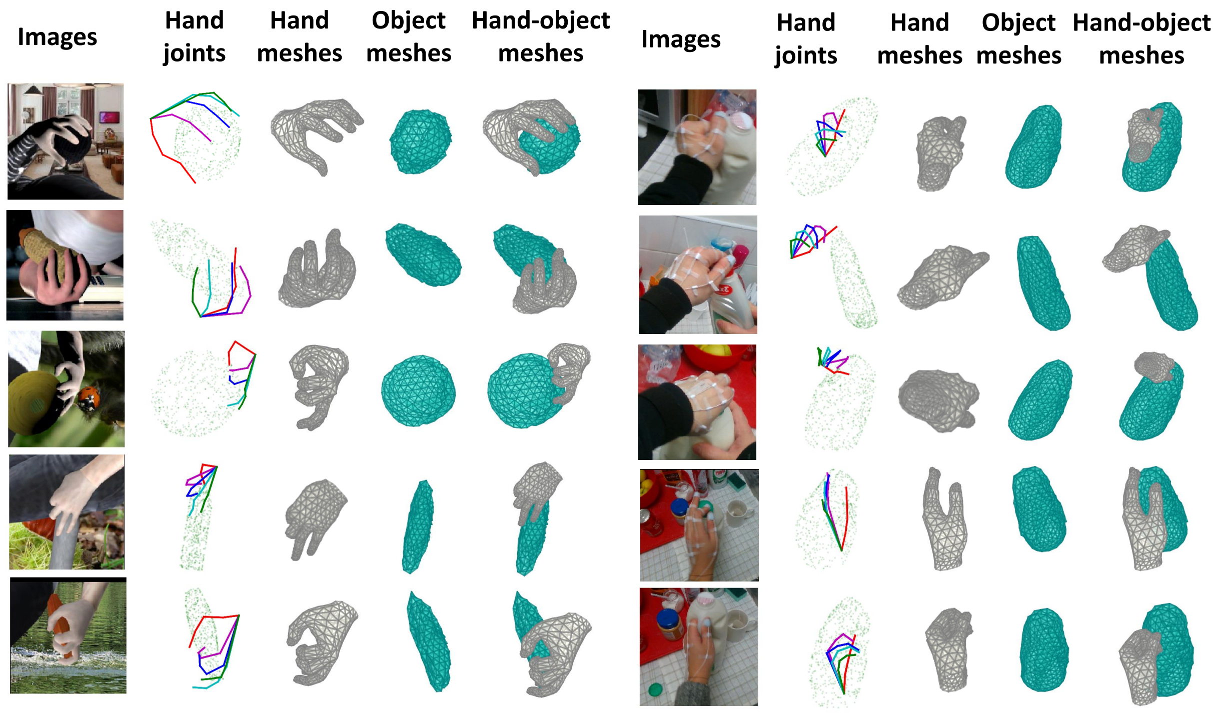


Fig. 8. Qualitative comparison of our method on the ObMan dataset (left) and the FHBc dataset (right). From left to right: input, estimated 3D hand pose, estimated hand mesh, estimated object mesh, estimated hand and object meshes. In the column of hand joints, we sample points from the ground truth object meshes to reveal the relationship between our estimated hand pose and the object. Note that we changed scales in the visualization.

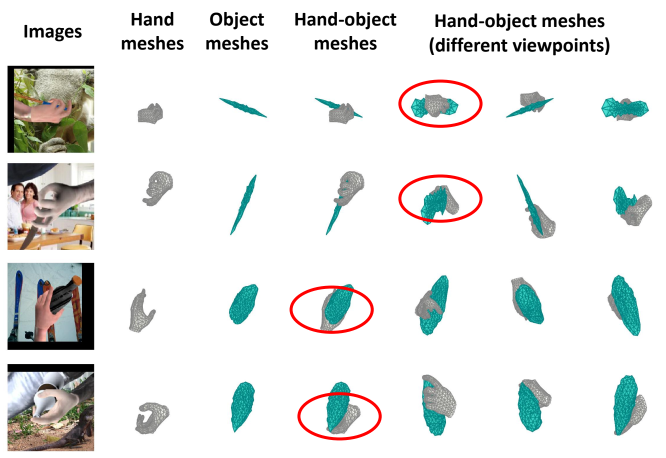


Fig. 9. Failure cases of our method on the ObMan dataset. From first row to second row, the shape of the invisible part is inaccurate (as circled in red). From third row to fourth row, the relative position between hand and object is inaccurate, which usually occurs when fingers covers the object.

method is 9.6mm when without the contact computation, and it decreases by 5.2% and reaches to 9.1mm after adding contact training procedure. When adding the contact computation, our proposed method gets smaller penetration than [11].

Some visual demonstration images from the test sets of the ObMan and the FHBc are presented in Fig. 8. The results demonstrate that our method produces high-quality object meshes that can accurately represent the object shape from monocular hand-object images. The fifth and tenth columns of Fig. 8 show that the produced hand and object meshes can reveal hand-object relationships in 3D space. More generated shapes from the testing set of ObMan are shown in Fig. 10. In Fig. 9, we show failure cases, unreasonable object shapes are produced in some unseen parts, and the relative position of the hand and the object is difficult to recover accurately.

V. Conclusion

To handle the challenging task of joint 3D reconstruction of the hand-object image, we present an end-to-end feature fusion network to recover 3D shapes of hand and object. We introduce a depth estimation module to predict the depth map from the input RGB to enhance the spatial attribute. To jointly consider the hand and the object branches to extract better features for later reconstruction, we propose to connect the two branches and fuse their latent representations with a feature fusion module. The proposed LSTM feature fusion module along with several common alternatives have been studied. We demonstrate that the cross-branch information is helpful for shape reconstruction and that the proposed LSTM feature fusion block is more effective to enhance the object feature after comparing multiple feature fusion strategies. Experimental results on both synthetic and real-world datasets show that our method outperforms the previous work by a large margin. As recommended above, future research could explore other cooperation methods and more efficient intermediate representation to achieve better joint reconstruction. We also believe that in addition to investigating cooperation between the hand and the object, research should also be conducted to find cooperation and relationships among more items such as two hands and many objects.

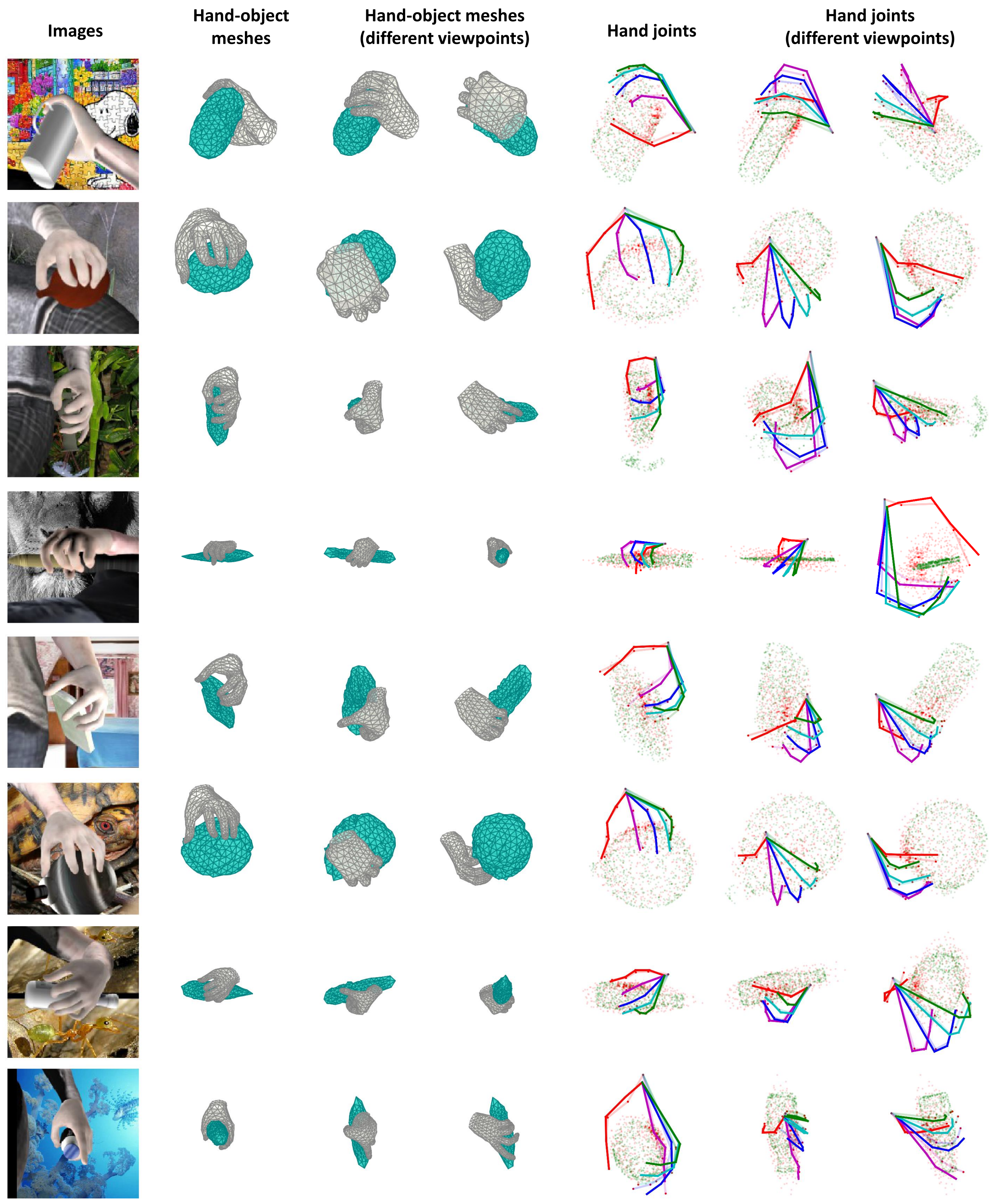

Images
Hand-object meshes
Hand-object meshes (different viewpoints)
Hand joints
Hand joints (different viewpoints)

Fig. 10. Qualitative visualization of our method on the ObMan dataset. From left to right: input, estimated hand and object meshes (3 columns for 3 different views), estimated 3D hand joints (3 columns for 3 different views). In the columns of hand joints, we use the skeleton with opaque color to represent the estimated hand joints and the skeleton in semi-transparent color to represent the ground truth. Besides, the green points are sampled from the ground truth object meshes and the red points are sampled from the estimated object meshes. Note that scales are changed in the visualization.

References

[1] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, “Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions,” in *Proc. ICCV*, 2015, pp. 1949–1957.

[2] X. Deng, Y. Zhang, S. Yang, P. Tan, L. Chang, Y. Yuan, and H. Wang, “Joint hand detection and rotation estimation using cnn,” *IEEE Trans. Image Process.*, vol. 27, no. 4, pp. 1888–1900, 2017.

[3] M. Kölsch and M. Turk, “Robust hand detection.” in *Proc. FGR*, 2004, pp. 614–619.

[4] P. Morerio, L. Marcenaro, and C. S. Regazzoni, “Hand detection in first person vision,” in *Proc. FUSION*. IEEE, 2013, pp. 1502–1507.

[5] S. Narasimhaswamy, Z. Wei, Y. Wang, J. Zhang, and M. Hoai, “Contextual attention for hand detection in the wild,” in *Proc. ICCV*, 2019.

[6] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. D. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, W.-T. Chang, W. Hua, M. Georg, and M. Grundmann, “Mediapipe: A framework for perceiving and augmenting reality,” in *Proc. CVPR Workshops*, 2019.

[7] F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt, “Ganerated hands for real-time 3d hand tracking from monocular rgb,” in *Proc. CVPR*, 2018, pp. 49–59.

[8] A. Spurr, J. Song, S. Park, and O. Hilliges, “Cross-modal deep variational hand pose estimation,” in *Proc. CVPR*, 2018, pp. 89–98.

[9] S. Yuan, G. Garcia-Hernando, B. Stenger, G. Moon, J. Yong Chang, K. Mu Lee, P. Molchanov, J. Kautz, S. Honari, L. Ge *et al.*, “Depth-based 3d hand pose estimation: From current achievements to future goals,” in *Proc. CVPR*, 2018, pp. 2636–2645.

[10] C. Zimmermann and T. Brox, “Learning to estimate 3d hand pose from single rgb images,” in *Proc. ICCV*, 2017, pp. 4903–4911.

[11] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid, “Learning joint reconstruction of hands and manipulated objects,” in *Proc. CVPR*, 2019, pp. 11 807–11 816.

[12] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother, “Learning 6d object pose estimation using 3d object coordinates,” in *Proc. ECCV*. Springer, 2014, pp. 536–551.

[13] M. Oberweger, P. Wohlhart, and V. Lepetit, “Generalized feedback loop for joint hand-object pose estimation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.

[14] B. Doosti, S. Naha, M. Mirbagheri, and D. Crandall, “Hope-net: A graph-based model for hand-object pose estimation,” in *Proc. CVPR*, 2020.

[15] J. Taylor, R. Stebbing, V. Ramakrishna, C. Keskin, J. Shotton, S. Izadi, A. Hertzmann, and A. Fitzgibbon, “User-specific hand modeling from monocular depth sequences,” in *Proc. CVPR*, 2014, pp. 644–651.

[16] S. Khamis, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon, “Learning an efficient model of hand shape variation from depth images,” in *Proc. CVPR*, 2015, pp. 2540–2548.

[17] J. Romero, D. Tzionas, and M. J. Black, “Embodied hands: Modeling and capturing hands and bodies together,” *ACM Trans. Graph.*, vol. 36, no. 6, p. 245, 2017.

[18] M. Kokic, D. Kragic, and J. Bohg, “Learning to estimate pose and shape of hand-held objects from rgb images,” in *Proc. IROS*, 2020, pp. 3980–3987.

[19] Y. Hasson, B. Tekin, F. Bogo, I. Laptev, M. Pollefeys, and C. Schmid, “Leveraging photometric consistency over time for sparsely supervised hand-object reconstruction,” in *Proc. CVPR*, 2020.

[20] Y. Tian, A. Luo, X. Sun, K. Ellis, W. T. Freeman, J. B. Tenenbaum, and J. Wu, “Learning to infer and execute 3d shape programs,” in *Proc. ICLR*, 2019.

[21] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[22] J. Romero, H. Kjellström, and D. Kragic, “Hands in action: real-time 3d reconstruction of hands in interaction with objects,” in *Proc. ICRA*. IEEE, 2010, pp. 458–463.

[23] H. Hamer, J. Gall, T. Weise, and L. Van Gool, “An object-dependent hand pose prior from sparse training data,” in *Proc. CVPR*. IEEE, 2010, pp. 671–678.

[24] H. Hamer, K. Schindler, E. Koller-Meier, and L. Van Gool, “Tracking a hand manipulating an object,” in *Proc. ICCV*. IEEE, 2009, pp. 1475–1482.

[25] L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys, “Motion capture of hands in action using discriminative salient points,” in *Proc. ECCV*. Springer, 2012, pp. 640–653.

[26] D. Tzionas, L. Ballan, A. Srikantha, P. Aponte, M. Pollefeys, and J. Gall, “Capturing hands in action using discriminative salient points and physics simulation,” *Int. J. Comput. Vis.*, vol. 118, no. 2, pp. 172–193, 2016.

[27] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt, “Real-time joint tracking of a hand manipulating an object from rgb-d input,” in *Proc. ECCV*. Springer, 2016, pp. 294–310.

[28] D. Tzionas and J. Gall, “3d object reconstruction from hand-object interactions,” in *Proc. ICCV*, 2015, pp. 729–737.

[29] Y. Gao, Y. Wang, P. Falco, N. Navab, and F. Tombari, “Variational object-aware 3-d hand pose from a single rgb image,” *IEEE Robot. Autom. Lett.*, vol. 4, no. 4, pp. 4239–4246, 2019.

[30] S. Hampali, M. Oberweger, M. Rad, and V. Lepetit, “Honnotate: A method for 3d annotation of hand and object poses,” in *Proc. CVPR*, 2020.

[31] B. Tekin, F. Bogo, and M. Pollefeys, “H+ o: Unified egocentric recognition of 3d hand-object poses and interactions,” in *Proc. CVPR*, 2019, pp. 4511–4520.

[32] A. Tsoli and A. A. Argyros, “Joint 3d tracking of a deformable object in interaction with a hand,” in *Proc. ECCV*, 2018, pp. 484–500.

[33] L. Keselman, J. Iselin Woodfill, A. Grunnet-Jepsen, and A. Bhowmik, “Intel realsense stereoscopic depth cameras,” in *Proc. CVPR Workshops*, 2017, pp. 1–10.

[34] Z. Zhang, “Microsoft kinect sensor and its effect,” *IEEE multimedia*, vol. 19, no. 2, pp. 4–10, 2012.

[35] P. Panteleris, I. Oikonomidis, and A. Argyros, “Using a single rgb frame for real time 3d hand pose estimation in the wild,” in *Proc. WACV*. IEEE, 2018, pp. 436–445.

[36] Y. Cai, L. Ge, J. Cai, and J. Yuan, “Weakly-supervised 3d hand pose estimation from monocular rgb images,” in *Proc. ECCV*, 2018, pp. 666–682.

[37] Y. Cai, L. Ge, J. Liu, J. Cai, T.-J. Cham, J. Yuan, and N. M. Thalmann, “Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks,” in *Proc. ICCV*, 2019, pp. 2272–2281.

[38] U. Iqbal, P. Molchanov, T. Breuel Juergen Gall, and J. Kautz, “Hand pose estimation via latent 2.5 d heatmap regression,” in *Proc. ECCV*, 2018, pp. 118–134.

[39] H. Liang, J. Yuan, D. Thalmann, and Z. Zhang, “Model-based hand pose estimation via spatial-temporal hand parsing and 3d fingertip localization,” *Vis Comput*, vol. 29, no. 6-8, pp. 837–848, 2013.

[40] Y. Chen, Z. Tu, L. Ge, D. Zhang, R. Chen, and J. Yuan, “So-handnet: Self-organizing network for 3d hand pose estimation with semi-supervised learning,” in *Proc. ICCV*, 2019, pp. 6961–6970.

[41] L. Ge, Z. Ren, and J. Yuan, “Point-to-point regression pointnet for 3d hand pose estimation,” in *Proc. ECCV*, 2018, pp. 475–491.

[42] H. Guo, G. Wang, X. Chen, C. Zhang, F. Qiao, and H. Yang, “Region ensemble network: Improving convolutional network for hand pose estimation,” in *Proc. ICIP*. IEEE, 2017, pp. 4512–4516.

[43] G. Moon, J. Yong Chang, and K. Mu Lee, “V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map,” in *Proc. CVPR*, 2018, pp. 5079–5088.

[44] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, “Cascaded hand pose regression,” in *Proc. CVPR*, 2015, pp. 824–832.

[45] D. Tang, H. Jin Chang, A. Tejani, and T.-K. Kim, “Latent regression forest: Structured estimation of 3d articulated hand posture,” in *Proc. CVPR*, 2014, pp. 3786–3793.

[46] J. Malik, A. Elhayek, F. Nunnari, K. Varanasi, K. Tamaddon, A. Heloir, and D. Stricker, “Deephps: End-to-end estimation of 3d hand pose and shape by learning from synthetic depth,” in *Proc. 3DV*. IEEE, 2018, pp. 110–119.

[47] L. Ge, Z. Ren, Y. Li, Z. Xue, Y. Wang, J. Cai, and J. Yuan, “3d hand shape and pose estimation from a single rgb image,” in *Proc. CVPR*, 2019, pp. 10 833–10 842.

[48] X. Zhang, Q. Li, H. Mo, W. Zhang, and W. Zheng, “End-to-end hand mesh recovery from a monocular rgb image,” in *Proc. ICCV*, 2019, pp. 2354–2364.

[49] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox, “Freihand: A dataset for markerless capture of hand pose and shape from single rgb images,” in *Proc. ICCV*, 2019, pp. 813–822.

[50] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese, “3d-r2n2: A unified approach for single and multi-view 3d object reconstruction,” in *Proc. ECCV*. Springer, 2016, pp. 628–644.

[51] J. Wu, C. Zhang, X. Zhang, Z. Zhang, W. T. Freeman, and J. B. Tenenbaum, “Learning shape priors for single-view 3d completion and reconstruction,” in *Proc. ECCV*, 2018, pp. 646–662.

[52] H. Fan, H. Su, and L. J. Guibas, “A point set generation network for 3d object reconstruction from a single image,” in *Proc. CVPR*, 2017, pp. 605–613.

[53] C.-H. Lin, C. Kong, and S. Lucey, “Learning efficient point cloud generation for dense 3d object reconstruction,” in *Proc. AAAI*, 2018.

[54] G. Riegler, A. Osman Ulusoy, and A. Geiger, “Octnet: Learning deep 3d representations at high resolutions,” in *Proc. CVPR*, 2017, pp. 3577–3586.

[55] M. Tatarchenko, A. Dosovitskiy, and T. Brox, “Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs,” in *Proc. ICCV*, 2017, pp. 2088–2096.

[56] T. Deprelle, T. Groueix, M. Fisher, V. Kim, B. Russell, and M. Aubry, “Learning elementary structures for 3d shape generation and matching,” in *Proc. NeurIPS*, 2019, pp. 7433–7443.

[57] G. Gkioxari, J. Malik, and J. Johnson, “Mesh r-cnn,” in *Proc. ICCV*, 2019, pp. 9785–9795.

[58] T. Groueix, M. Fisher, V. Kim, B. Russell, and M. Aubry, “Atlasnet: A papier-mâché approach to learning 3d surface generation,” in *Proc. CVPR*, 2018.

[59] H. Kato, Y. Ushiku, and T. Harada, “Neural 3d mesh renderer,” in *Proc. CVPR*, 2018, pp. 3907–3916.

[60] E. Smith, S. Fujimoto, A. Romero, and D. Meger, “Geometrics: Exploiting geometric structure for graph-encoded objects,” in *Proc. ICML*, 2019, pp. 5866–5876.

[61] N. Wang, Y. Zhang, Z. Li, Y. Fu, W. Liu, and Y.-G. Jiang, “Pixel2mesh: Generating 3d mesh models from single rgb images,” in *Proc. ECCV*, 2018, pp. 52–67.

[62] X. Li and X. Wu, “Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition,” in *Proc. ICASSP*. IEEE, 2015, pp. 4520–4524.

[63] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” in *Proc. NeurIPS*, 2014, pp. 3104–3112.

[64] K. Lee, I. Lee, and S. Lee, “Propagating lstm: 3d pose estimation based on joint interdependency,” in *Proc. ECCV*, 2018, pp. 119–135.

[65] J. Liu, H. Ding, A. Shahroudy, L.-Y. Duan, X. Jiang, G. Wang, and A. K. Chichung, “Feature boosting network for 3d pose estimation,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2019.

[66] A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang, “Ntu rgb+ d: A large scale dataset for 3d human activity analysis,” in *Proc. CVPR*, 2016, pp. 1010–1019.

[67] C. Zou, E. Yumer, J. Yang, D. Ceylan, and D. Hoiem, “3d-prnn: Generating shape primitives with recurrent neural networks,” in *Proc. ICCV*, 2017, pp. 900–909.

[68] J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum, “Marrnet: 3d shape reconstruction via 2.5 d sketches,” in *Proc. NeurIPS*, 2017, pp. 540–550.

[69] X. Zhang, Z. Zhang, C. Zhang, J. Tenenbaum, B. Freeman, and J. Wu, “Learning to reconstruct shapes from unseen classes,” in *Proc. NeurIPS*, 2018, pp. 2257–2268.

[70] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. CVPR*, 2016, pp. 770–778.

[71] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli *et al.*, “Image quality assessment: from error visibility to structural similarity,” *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.

[72] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, “Imagenet large scale visual recognition challenge,” *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[73] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, “Shapenet: An information-rich 3d model repository,” *arXiv:1512.03012*, 2015. [Online]. Available: https://arxiv.org/abs/1512.03012

[74] A. T. Miller and P. K. Allen, “Graspit! a versatile simulator for robotic grasping,” *IEEE Robot. Autom. Mag.*, vol. 11, no. 4, pp. 110–122, 2004.

[75] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim, “First-person hand action benchmark with rgb-d videos and 3d hand pose annotations,” in *Proc. CVPR*, 2018, pp. 409–419.

[76] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. ICLR*, 2014.