Recurrent 3D Hand Pose Estimation Using Cascaded Pose-guided 3D Alignments

Xiaoming Deng, Dexin Zuo, Yinda Zhang, Zhaopeng Cui, Jian Cheng, Ping Tan, Liang Chang, Marc Pollefeys, Sean Fanello, Hongan Wang

Abstract—3D hand pose estimation is a challenging problem in computer vision due to the high degrees-of-freedom of hand articulated motion space and large viewpoint variation. As a consequence, similar poses observed from multiple views can be dramatically different. In order to deal with this issue, view-independent features are required to achieve state-of-the-art performance. In this paper, we investigate the impact of view-independent features on 3D hand pose estimation from a single depth image, and propose a novel recurrent neural network for 3D hand pose estimation, in which a cascaded 3D pose-guided alignment strategy is designed for view-independent feature extraction and a recurrent hand pose module is designed for modeling the dependencies among sequential aligned features for 3D hand pose estimation. In particular, our cascaded pose-guided 3D alignments are performed in 3D space in a coarse-to-fine fashion. First, hand joints are predicted and globally transformed into a canonical reference frame; Second, the palm of the hand is detected and aligned; Third, local transformations are applied to the fingers to refine the final predictions. The proposed recurrent hand pose module for aligned 3D representation can extract recurrent pose-aware features and iteratively refines the estimated hand pose. Our recurrent module could be utilized for both single-view estimation and sequence-based estimation with 3D hand pose tracking. Experiments show that our method improves the state-of-the-art by a large margin on popular benchmarks with the simple yet efficient alignment and network architectures.

Index Terms—Hand Pose Estimation, Alignment, Cascaded Neural Networks, Recurrent Model.

1 Introduction

Hand pose estimation is one of the fundamental problems in computer vision and computer graphics, and has many applications in human-computer interactions and augmented reality [1]. Compared to color images, depth images are more suitable to solve the 3D hand pose estimation task due to two main advantages: 1) They encode 2.5D information of the hand, thus facilitate the segmentation between foreground and background. 2) Depth images are generally less sensitive to ambient light, especially in indoor scenes, which leads to reliable imaging of human hand and thus more accurate hand pose estimation performance.

Existing hand pose estimation work either uses model-based method [2], [3] by fitting hand parametric models [4], [5], or learning-based methods. Early learning-based methods mainly use random forest algorithms to regress hand joint positions [6], [7], [8]. Recently, deep learning based hand pose estimation methods have achieved superior performance and dominated the top of the benchmarks. However, accurate 3D hand pose estimation is still challenging due to the high degrees-of-freedom of hand articulated motion space. Moreover, similar poses observed from different viewpoints can be dramatically different, which causes learning 3D hand pose to be hard, or requires a lot of training data.

Aligning data in 3D space, is one of the most effective way to reduce the variations of input data due to different viewpoints or different articulated hand motions and thus reduce the burden of the machine learning model. A global rigid transformation is generally used to bring the observed 3D hands into a canonical coordinate system, e.g. fingers up, and therefore the joint locations can be estimated independently with the arbitrary orientations [9]. More specifically, this idea has been extensively studied in the field of 3D hand pose estimation from depth images. Sun *et al.* [8] proposed to calculate features from the 2D input depth image under different coordinate systems built on palm and fingers, and train a random forest to predict the residual of the pose in cascade. Ye *et al.* [10] extended the similar ideas using deep neural network, in which in-plane rotations are applied iteratively to the input depth, and local features at each joint are cropped by an attention model to predict pose residual. These efforts are tried to make the input data and the extracted features invariant to the camera pose, which will make the training process relatively easier. Despite their

- *X. Deng, D. Zuo, J. Cheng, H. Wang are with are with the Beijing Key Laboratory of Human Computer Interactions, Institute of Software, Chinese Academy of Sciences, Beijing 100190, China, and University of Chinese Academy of Sciences, Beijing, China.*
  E-mail: xiaoming@iscas.ac.cn, hongan@iscas.ac.cn
- *Y. Zhang and S. Fanello are with Google, Mountain View, CA, USA, 94043.*
  E-mail: yindaz@google.com, seanfa@google.com
- *Z. Cui is with the State Key Lab of CAD&CG, Zhejiang University, Hangzhou 310058, China.*
  E-mail: zhpcui@zju.edu.cn
- *P. Tan is with the School of Computing Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada.*
  E-mail: pingtan@sfu.ca
- *M. Pollefeys is with the Department of Computer Science, ETH Zurich, CH-8092 Zurich, Switzerland.*
  E-mail: marc.pollefeys@inf.ethz.ch
- *L. Chang is with the School of Artificial Intelligence, Beijing Normal University, Beijing 100875, China.*
  E-mail: changliang@bnu.edu.cn

Manuscript received September, 2021; revised December, 2021; accepted February, 2022; date of current version March, 2022.
(Corresponding authors: Xiaoming Deng, Yinda Zhang, Zhaopeng Cui)

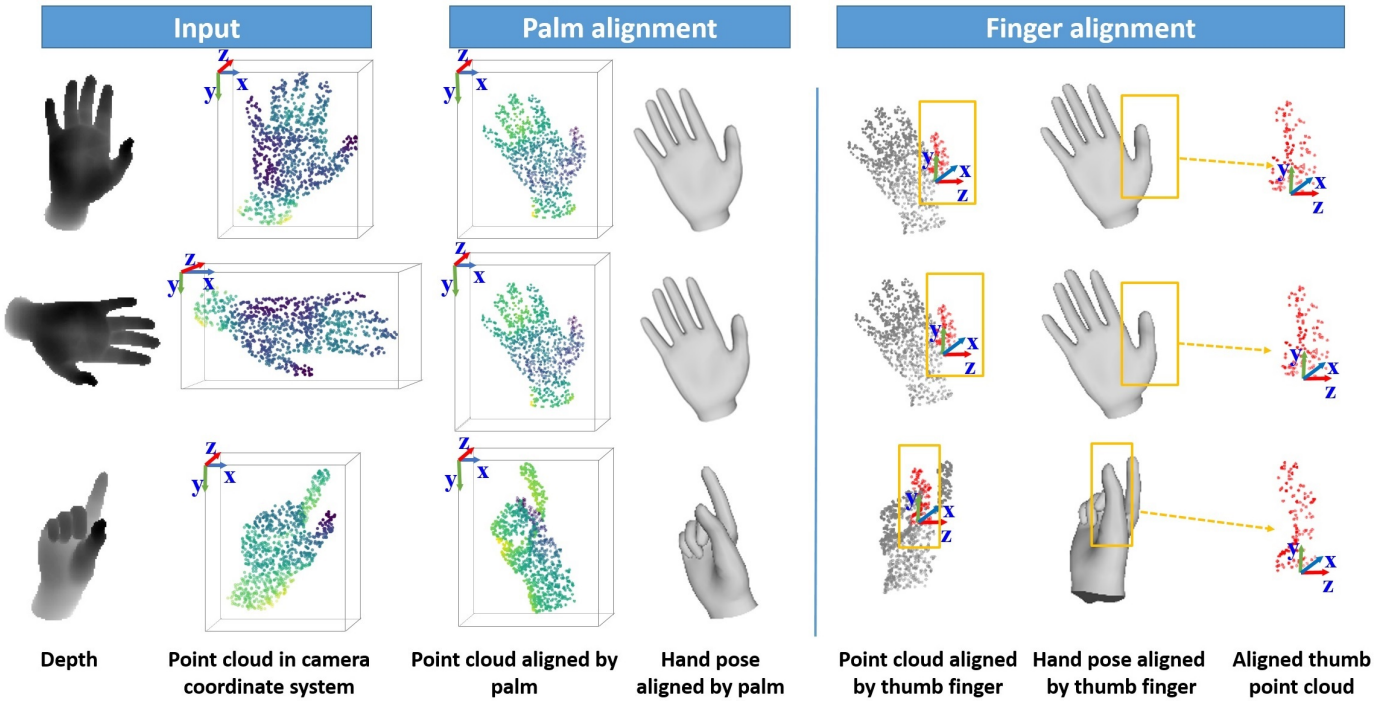


Fig. 1. Illustration of our cascaded pose-guided 3D alignments and feature extraction in 3D space. Although the point clouds of the same hand pose from different viewpoints (row 1 vs. row 2) and the point clouds of different hand poses (row 1 and row 2 vs. row 3) are quite different, they become similar for palm and finger parts using palm alignment and finger alignment (see col 3 and col 7), respectively. The hand shapes in the fourth and sixth column are only used for a clear illustration of alignment transformations.

success, these works do not achieve the full 3D hand pose estimation pipeline in the 3D space – either the alignment transformation is restricted in 2D image plane [8], [10] or the feature is calculated from the perspective 2D depth image [11], [12], and lead to unsatisfactory pose estimation results.

Among the deep neural network estimation models, the cascaded models that adopt multiple sequential stages for coarse-to-fine estimation have been successfully applied in hand pose estimation (e.g. [8], [10], [13]). However, these models only use the features of the current iteration and may overlook the features of the previous cascaded iterations. Since the cascaded hand pose estimation iteratively refines hand skeleton joints, there could be strong correlation between the features from different iterations. Therefore, if the features of the neighboring cascaded hand pose modules are fused during the training of the current cascaded module, the hand pose estimation performance could be possibly further improved.

To address these issues, we firstly investigate the impact of view-independent features for 3D hand pose estimation from a single depth image (Fig. 1). We demonstrate that it is crucial to perform both *data alignment* and *feature extraction* in *3D space*. This enables view-independent features, which greatly boost the accuracy for hand pose estimation. The input depth image is first converted to a 3D point cloud representation, and several 3D transformations guided by the initial hand pose (i.e. *pose-guided 3D alignment*) are obtained to align the input point clouds to the coordinate systems of the palm and each finger. As shown in Fig. 1, although the point clouds of different hand poses are quite different, they become similar for palm and finger parts using palm alignment and finger alignment, respectively. Features are then extracted from the transformed 3D point clouds to make it fully invariant to the camera pose. Adopting pose-guided 3D alignment directly on 3D point clouds produces more intuitive and geometrically reasonable input that fixes the issue due to the different hand viewpoints and facilitates the hand pose estimation. Inspired by Sun *et al.* [8], we conduct both global and local alignments in a cascaded way. In particular, we design *cascaded pose-guided 3D alignments* that are performed in 3D space in a coarse-to-fine fashion. By using the alignments of palm stage and finger stage, we obtain different transformations for palm and each finger (i.e. in total 6) to transform the relevant point clouds for each part to a canonical coordinate system and then extract the view invariant features for each part. Our global and local alignments can address the issue of high degree of freedom in hand pose space in a divide and conquer strategy. We design a new recurrent hand pose module for aligned 3D point clouds to generate recurrent pose-aware features and further enhance the performance of our method. Our recurrent hand pose module could be utilized for both single-view estimation and sequence-based estimation with 3D hand pose tracking. Moreover, our method can be generalized to different 3D point cloud network backbones or other 3D representations such as 3D volume (see the supplementary document).

Our contributions can be summarized as follows:

1) To be best of our knowledge, we propose the first pose-guided data alignment of 3D point clouds for 3D hand pose estimation. With the proposed alignment, we design a 3D cascaded deep learning framework, which learns fully view-independent

features by performing cascaded pose-guided data alignment and feature extraction in 3D progressively via both global and local alignments;

2) We present a new recurrent hand pose module for aligned 3D representation that can extract recurrent pose-aware feature and iteratively refine the estimated hand pose. Besides the aligned 3D point clouds, the recurrent module extracts recurrent pose-aware features by taking as input the features from previous iterations via the long short term memory (LSTM) cells, which model the dependency of features in different iterations of refinement. The proposed recurrent module could be utilized for both single-view estimation and sequence-based estimation with 3D hand pose tracking;

3) Extensive experiments demonstrate that our method improves the state-of-the-art (SoTA) by a large margin on the main hand pose benchmark datasets as well as excellent efficiency.

2 Related Works

Our method is related to model-based hand pose estimation methods, learning-based hand pose estimation methods, data alignment and cascaded models.

2.1 Model-based Hand Pose Estimation

Model-based hand pose methods assume predefined 3D hand deformable models, and they try to fit observed depth maps by minimizing a nonlinear function over the hand poses [1]. They are mostly based on slow, but accurate local optimization such as particle swarm optimization (PSO) and require good initialization [14]. A few methods [2], [3], [4] overcome issues with slow optimization, yet these methods still suffer in presence of strong self occlusions and poor re-initialization. The widely used 3D hand models consist of a rigged polygonal mesh model [15], a statistical hand shape model named MANO [5] and a mixture of spheres [16]. In these methods, an additional subject calibration that personalizes the hand model can improve the overall hand pose prediction accuracy [17]. However, high-fidelity hand shape modeling itself is challenging [15], [18], [19].

2.2 Learning-based Hand Pose Estimation

With the recent availability of large scale training data, hand pose can be formalized as machine learning problem using random forest [6], [7], [8], 2D CNN [11], [13], [20], [21], [22], [23], [24] and 3D CNN [25], [26], [27], [28].

In 2D CNN based methods, the hand skeleton joints are usually modeled as a heatmap in the depth image obtained with a trained classifier [20], [21], or directly via regression [11], [13], [22], [23], [24]. These methods use 2D CNN to extract 2D features. Due to the domain difference between 2D features and 3D joints, they struggle to learn an accurate mapping from 2D features to 3D joint locations [26].

3D CNN based methods directly operate in 3D space such as voxels or point cloud and extract 3D features, which have been proved to be effective for hand pose estimation [25], [26], [27]. Deng *et al.* [25] and Ge *et al.* [26] regress hand joints from a 3D volume via the truncated signed distance function (TSDF). Moon *et al.* [27] estimate hand joints by 3D heatmaps via classification. This method has ranked first in the Hands 2017 Challenge [29], proving once again the importance of 3D representations. Recently, several works use PointNet [30], [31] or other point cloud network such as PEL [32] to estimate hand pose. Ge *et al.* [28] estimate hand pose on raw point clouds by a hierarchical network. Ge *et al.* [33] propose a method which estimates hand pose with 3D heatmaps.

Differently from previous work, we propose a new pose-guided alignment of 3D representation from coarse to fine. Compared to HandPointNet [28] that uses PCA to align the input point cloud and leads to fixed alignment even with multiple PCA alignments, our pose-guided 3D alignment iteratively aligns the input point cloud with the estimated 3D pose in the previous stage and converges to a proper canonical coordinate, and can leverage an effective recurrent architecture to refine the hand pose.

2.3 Data Alignment for Hand Pose Estimation

Data alignment is an effective pre-process step for hand pose estimation, which can reduce the viewpoint variation and enhance hand pose estimation performance. Previous methods adopt global 2D rigid transformations [9] or a PCA model [33] for alignment.

Compared to previous alignments such as principal component analysis (PCA) in [28] or learning-based alignment in 3D space [30], [31], [34], we align the input data with estimated hand pose learned from supervision, while the others rely a canonical coordinate that is vaguely defined from distribution of the raw input data. As a result, our alignment benefits from the initial learning stage, and is more interpretable and robust against hand viewpoint diversity in the input. Moreover, we apply this pose-guided alignment in multiple scales (from global structure to local details), and use the alignment for recurrent hand pose refinement, which is trained in an end-to-end manner and is found to be effective all the time.

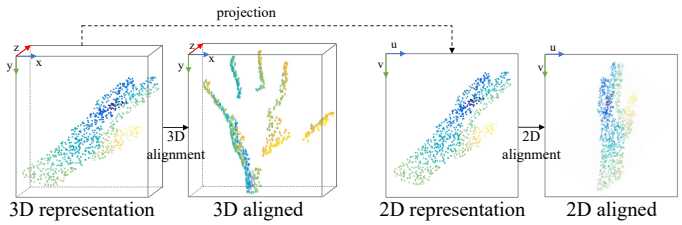


Fig. 2. Illustration of different data representations and alignments (2D/3D).

2.4 Cascaded Models for Hand Pose Estimation

Cascaded models have been successfully applied in hand pose estimation [8], [10], [13], [35]. Sun *et al.* [8] propose to use hierarchical data alignment that estimates partial hand poses by extracting rotation invariant features from 2D depth images. Ye *et al.* [10] apply 2D transformation on the input depth map and employ spatial attention mechanism to estimate residual of pose error in cascade. Du *et al.* [36] propose a multi-task hand pose regression network named

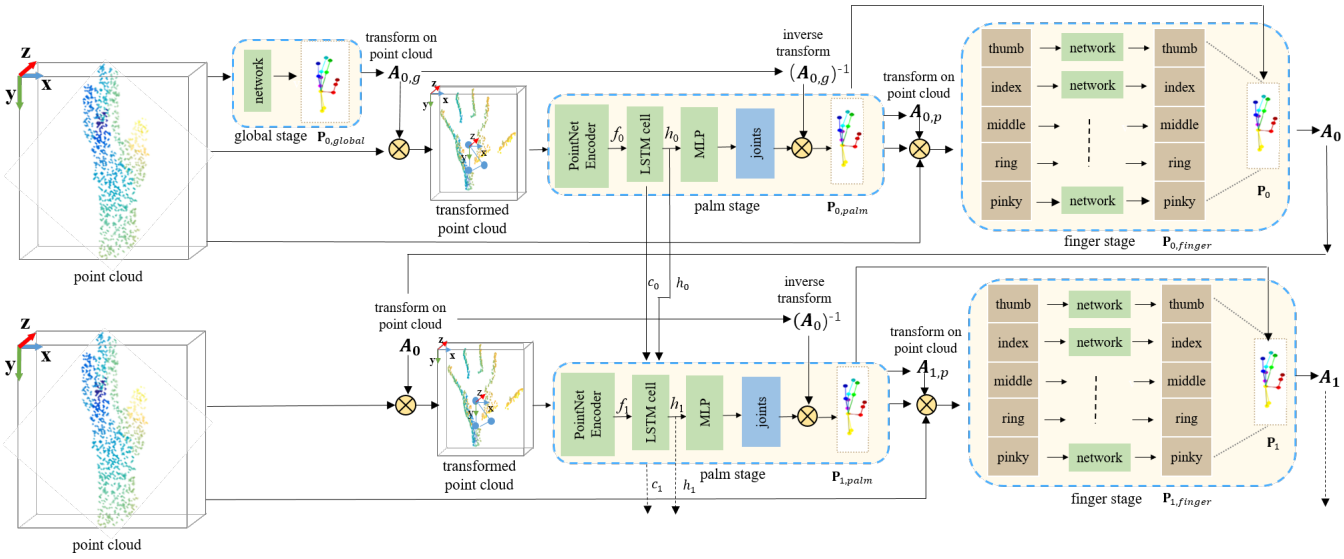


Fig. 3. Illustration of our recurrent hand pose network using cascaded pose-guided alignments. We first convert the input hand foreground depth to point cloud. Then we adopt multiple recurrent iterations to estimate the 3D hand pose. Specifically, we introduce several LSTM modules among multiple palm stages to refine the hand pose. In each recurrent iteration, we adopt a multi-stage network (i.e. global, palm and finger stages) to predict hand joints by iterative pose regression and cascaded pose-guided 3D alignment, and we adopt the hand pose of the previous iteration to align the input point cloud of the current iteration. "PointNet Encoder" denotes the network before the last multi-layer perception (MLP) of PointNet++. "$\mathbf{A}_{0,g}$" is the transformation via the estimated hand pose $\mathbf{P}_{0,global}$ of the global stage in the initial recurrent iteration, "$\mathbf{A}_{t,p}$" are the transformations to align each finger via the estimated hand pose of the palm stage $\mathbf{P}_{t,palm}$ in the t-th recurrent iteration, and "$\mathbf{A}_t$" is the transformation via the composited hand pose "$\mathbf{P}_t$" of the palm stage and the finger stage in the t-th recurrent iteration. "$\otimes$" denotes matrix multiplication.

CrossInfoNet that uses hierarchical model to decompose hand pose estimation into palm joint regression and finger joint regression. Although CrossInfoNet adopts hierarchical model to share the complementary information between different tasks, it does not use the effective alignment strategy. Some methods also adopt an iterative way for pose estimation including iterative error feedback for 2D human pose estimation [37] and point cloud registration [38].

Compared to these learning based cascaded solutions that are mostly built on 2D depth maps and 2D alignments, our full 3D model performs both data alignment and feature extraction in 3D space. This allows us to extract 3D view-independent features, and also to reduces the domain gap issue between the input space (i.e. 3D point clouds) to the output space (i.e. 3D joints). Fig. 2 illustrates the different data representations and alignments in 2D and 3D.

2.5 Recurrent Models for Hand Pose Estimation

The existing recurrent models for hand pose estimation are mainly designed for the input of image sequences. Wu *et al.* [39] propose to jointly model the spatial-temporal context for 3D hand pose estimation from depth image sequences, in which the temporal network extracts features considering the temporal coherence of input images via LSTM. Recently, a recurrent hand pose model named SeqHAND [40] is proposed to estimate hand pose and hand shape using sequential color images as input. Different from the above recurrent hand pose models [39], [40] that are mainly designed for the input of image sequences, our method designs an effective recurrent hand pose model for the aligned 3D representation of a single frame based on our novel pose-guided alignment module. In other words, without the proposed pose-guided 3D alignment, it is non-trivial or not straightforward to apply the recurrent network from sequential color images to 3D point clouds/depth map.

3 Method

In this section, we introduce our recurrent model to estimate 3D hand joint locations from 3D point clouds representation of a single depth image, which has the distinct characteristics of cascaded pose-guided 3D alignments for view-independent feature extraction. As shown in Fig. 3, we adopt multiple recurrent iterations to estimate the 3D hand pose. In each recurrent iteration, we adopt a cascaded multi-stage architecture to predict hand joints by iteratively pose regression and cascaded 3D pose-guided alignment, and each of the cascaded stage focuses on the pose estimation for different parts of the hand. The key idea of the cascaded stages is to transform each part of the hand to an *aligned* local coordinate system, so that we can learn to extract view-invariant features from the input data. Each recurrent iteration starts from a global transformation ($\mathbf{A}_{0,g}$ via a global stage of the initial iteration or $\mathbf{A}_{t-1}$ via the previous recurrent iteration) applied to the palm and then more local ones applied to each finger. The alignment transformation of subsequent stages is guided by the hand pose estimated from the previous stages, thus we name the alignment transformations as *cascaded pose-guided 3D alignment*. Our recurrent module takes as input the features from the palm stage of the previous iterations via the long short term memory (LSTM) cells, which model the dependency of features in different iterations of refinement. The final predicted pose is composed of the joints of the last recurrent iteration. We first estimate the joints from their own local coordinate systems, and then transformed to the camera coordinates. Our full recurrent model can be trained in an end-to-end

manner. Practically, our method can be applied to inputs in various 3D representations, and we use the point cloud representation for illustration.

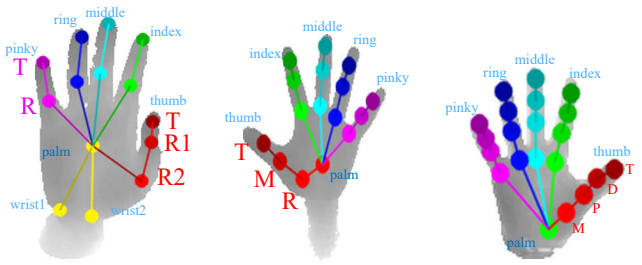
3.1 Cascaded Pose-guided 3D Alignment

We now detail how to predict proper coordinate systems for different hand parts, and apply the pose-guided 3D alignment to 3D point clouds representation.

As shown in Fig. 3, the whole system contains multiple recurrent iterations. In each recurrent iteration, we adopt a multi-stage network (including global stage, palm stage and finger stage) to predict hand pose by iterative pose regression and pose-guided 3D alignments. In the global stage, we train a network to predict the 3D hand pose directly in the camera coordinate system, which is then converted into homogeneous coordinates of 3D joints denoted as $\mathbf{P}_{global} \in \mathcal{R}^{4\times J}$ (J is the number of hand joints). In the palm stage, we first transform the 3D point cloud from the camera coordinate to the palm coordinate via a data alignment $\mathbf{A}_g$ inferred from $\mathbf{P}_{global}$. A new hand pose is estimated with the transformed point cloud, which is then transformed back to the camera coordinate via $\mathbf{A}_g^{-1}$ and then converted into homogeneous coordinates of 3D joints denoted as $\mathbf{P}_{palm} \in \mathcal{R}^{4\times J}$. Then similarly but in a finer scale, we build data alignment $\mathbf{A}_p = \{\mathbf{A}_p^i\}_{i=1}^5$ ($\mathbf{A}_p^i$ is the alignment for the i-th finger) from $\mathbf{P}_{palm}$, transform the input to each finger coordinate, predict a hand pose $\mathbf{P}_{finger} = \{\mathbf{P}_{finger}^i\}_{i=1}^5$, and transform back to the camera coordinate via $\mathbf{A}_p^{-1}$. Each network in the finger stage only predicts a subset of the joints in that hand part (shown in Table 1). The way to composite the final output from these intermediate estimation results of palm stage and finger stage can be found in Sec. 3.1.5. Though presented with point cloud as the input, similar pipeline can be naturally adapted to 3D volume.

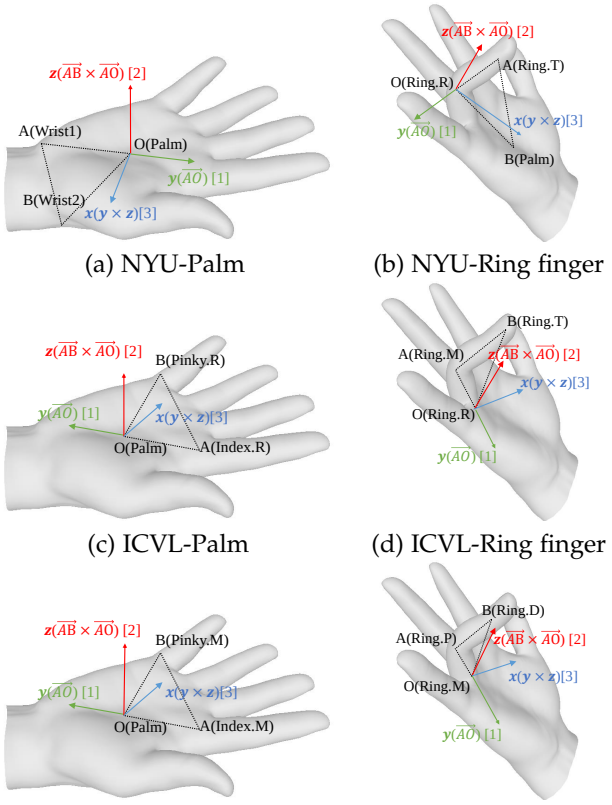
3.1.1 Hierarchical Coordinate Systems

We define the palm and finger coordinate systems to align the data to boost the accuracy of hand pose estimation. We use three fixed joints from the estimated hand joints of the previous stage to define a coordinate system. Before we elaborate on the hand coordinate systems, we illustrate the hand skeleton definitions for the main benchmark datasets such as NYU [20], ICVL [7], and MSRA [8] in Fig. 4.



(a) NYU (b) ICVL (c) MSRA

Fig. 4. Illustration of the hand skeleton models for the main benchmark datasets. (a) Hand joints in NYU hand dataset [20], which contains 14 joints. (b) Hand joints in ICVL hand dataset [7], which contains 16 joints. (c) Hand joints in MSRA hand dataset, which contains 21 joints.



(a) NYU-Palm (b) NYU-Ring finger

(c) ICVL-Palm (d) ICVL-Ring finger

(e) MSRA-Palm (f) MSRA-Ring finger

Fig. 5. Illustration of the coordinate systems for the pose-guided alignments. (a)(b), (c)(d) and (e)(f) show palm and ring finger coordinate systems of NYU, ICVL and MSRA datasets, respectively. The number in a bracket indicates the order index to obtain a direction.

3.1.2 Palm Coordinate System

We define the palm coordinate system with three joints on the palm from the estimated hand pose $\mathbf{P}_{global}$ of the global stage. For NYU hand skeleton, we use the joints 'Wrist1', 'Wrist2' and 'Palm', to build the palm coordinate system (Fig. 5(a)). The origin is set at 'Palm'. The y-axis is the direction from 'Wrist1' to 'Palm', the z-axis is perpendicular to the plane defined by the three joints and pointing from hand back to the front, and the x-axis is the cross product of y-axis and z-axis.

3.1.3 Finger Coordinate System

We define each finger coordinate system using three joints from the estimated hand pose $\mathbf{P}_{palm}$ of the palm stage. Take the ring finger as an example, on NYU dataset, we use the joints 'Ring.T', 'Palm', 'Ring.R' to build the ring finger coordinate system $\mathbf{\Omega}_{ring} = \{\mathbf{\Omega}_{ring,O}, \mathbf{\Omega}_{ring,x}, \mathbf{\Omega}_{ring,y}, \mathbf{\Omega}_{ring,z}\}$ (Fig. 5(b)). The origin $\mathbf{\Omega}_{ring,O}$ is at 'Ring.R'. The y-axis $\mathbf{\Omega}_{ring,y}$ is the direction from 'Ring.T' to 'Ring.R'. The z-axis $\mathbf{\Omega}_{ring,z}$ is the direction of cross product of 'Ring.T' to 'Palm' and y-axis, and the x-axis $\mathbf{\Omega}_{ring,x}$ is the cross product of y-axis and z-axis.

Fig. 5(c)(d), (e)(f) illustrate the palm and ring finger coordinate systems for ICVL dataset and MSRA dataset, respectively. A complete list of hand joints to define the coordinate systems is shown in Table 1. For each hand part, the third joint is the origin of a coordinate system, the y-axis of the coordinate system is the direction from the first joint

TABLE 1
Definitions of the hand parts in the NYU and ICVL datasets for the pose-guided alignment. The palm and fingers are indexed as palm, thumb, index, middle, ring, pinky, and wrist. The definitions of symbols 'R', 'M', 'P' , 'D' and 'T' can be found in Fig. 4.

Part	NYU	ICVL	MSRA
Palm	Wrist1, Wrist2, Palm	Index.R, Pinky.R, Palm	Index.M, Pinky.M, Palm
Thumb	Thumb.R1,Thumb.T,Thumb.R2	Thumb.M, Thumb.T, Thumb.R	Thumb.P, Thumb.D, Thumb.M
Index	Index.T, Palm, Index.R	Index.M, Index.T, Index.R	Index.P, Index.D, Index.M
Middle	Middle.T, Palm, Middle.R	Middle.M, Middle.T, Middle.R	Middle.P, Middle.D, Middle.M
Ring	Ring.T, Palm, Ring.R	Ring.M, Ring.T, Ring.R	Ring.P, Ring.D, Ring.M
Pinky	Pinky.T, Palm, Pinky.R	Pinky.M, Pinky.T, Pinky.R	Pinky.P, Pinky.D, Pinky.M

to the origin, the normal of the plane passing three joints of the hand part is z-axis, and the x-axis is the cross product of y-axis and z-axis.

3.1.4 Alignment Transformation

For each hand part (i.e., palm and five fingers), we calculate a similarity transformation $\mathbf{A} \in \mathcal{R}^{4\times4}$ to align the coordinate system of the predicted joints in the previous stage to a canonical coordinate system $\mathbf{\Omega}^{can} = \{\mathbf{\Omega}_O^{can}, \mathbf{\Omega}_x^{can}, \mathbf{\Omega}_y^{can}, \mathbf{\Omega}_z^{can}\}$, where the origin is set to $\mathbf{\Omega}_O^{can} = \mathbf{0}$, and the axes are set to $\mathbf{\Omega}_x^{can} = [1,0,0]^T$, $\mathbf{\Omega}_y^{can} = [0,1,0]^T$, $\mathbf{\Omega}_z^{can} = [0,0,1]^T$. The transformation $\mathbf{A}$ consists of scale s, rotation $\mathbf{R} \in \mathcal{R}^{3\times3}$, and translation $\mathbf{t} \in \mathcal{R}^{3\times1}$. The key idea to include scale for transformation $\mathbf{A}$ is to reduce the effect of different hand scales on the hand pose estimation network. For each local coordinate system $\mathbf{\Omega} = \{\mathbf{\Omega}_O, \mathbf{\Omega}_x, \mathbf{\Omega}_y, \mathbf{\Omega}_z\}$ defined in Sec. 3.1.3, we estimate a transformation $\mathbf{A}$ to align the origin and axis orientations with the canonical coordinate system $\mathbf{\Omega}^{can}$. The translation and rotation are set to $\mathbf{t} = -\mathbf{\Omega}_O$ and $\mathbf{R} = [\hat{\mathbf{\Omega}}_x, \hat{\mathbf{\Omega}}_y, \hat{\mathbf{\Omega}}_z]^{-1}$, where $\hat{\mathbf{\Omega}}_{x,y,z}$ means the ℓ_2-normalized vector of $\mathbf{\Omega}_{x,y,z}$. The scale of $\mathbf{A}$ is set to $s = \bar{l}/|\mathbf{\Omega}_y|$ ($\bar{l}$ is a pre-defined hand length). Therefore, the scale s enforces the length of transformed vector of $\mathbf{\Omega}_y$ for different inputs to be equal.

3.1.5 Pose Composition

The final hand pose is a direct composition of the estimated hand pose from the palm stage and the finger stage. We use the location for each hand finger joints from the pose estimated in the coordinate system defined by them (Table 1), and get the location of the other joints from the pose estimated in the palm stage. The pose from the global stage is only for initialization, and it is not used in the final output.

3.2 3D Representation and Networks

Here, we detail the used 3D representation and how to feed them into the network architecture. We use PointNet++ [31] as the network backbone, but practically other networks can also be used, e.g. DGCNN [41] for point clouds and 3D CNN [25], [26] for 3D volume.

The input depth can be converted into point clouds using the known camera intrinsic parameters, and thus the input depth is represented as a list of points with their 3D coordinates. To favor the learning, the point cloud is centered on the center of mass (CoM), which can be calculated reliably via traditional algorithms [13]. For the global stage and the palm stage, we adopt the same network architecture used in

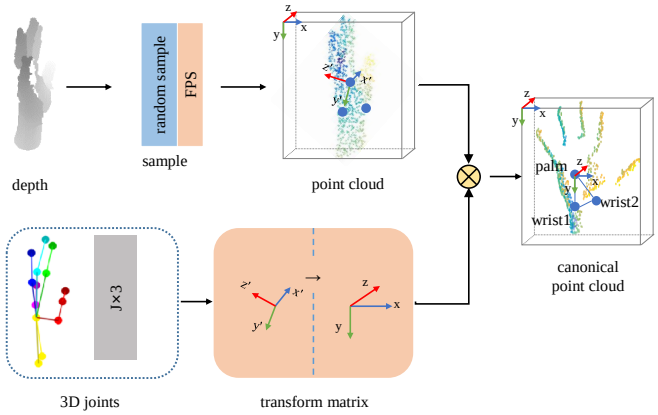


Fig. 6. Illustration of the pose-guided 3D alignment for point clouds. We use the predicted hand joints from the previous stage to get an alignment transformation to a canonical pose. "⊗" denotes matrix multiplication.

[28], which is built upon PointNet++ [31]. In the finger stage, we only collect relevant neighbor point clouds of each finger as input. In our experiments, we keep 64 nearest points of each finger joint within a distance threshold (set to 60mm in our experiments). Notice that the geometric structure of finger's local neighbor region is simpler than that of the whole hand, thus for finger stage, we use PointNet architecture [30] for efficiency. In order to balance the density of input point and efficiency, we downsample input points using random sampling and farthest point sampling (FPS) (we sample 1024 points and feed them to the networks). Finally, we transform the location of sampled 3D points into the new coordinate system (Fig. 6).

Details of the networks using point cloud can be found in the supplementary document.

3.3 Recurrent Hand Pose Model

Although the above network is capable of recovering 3D hand pose under different hand articulated motion and different viewpoints, we leverage a recurrent hand pose model to generate recurrent pose-aware features that is effective to further improve the hand pose estimation performance.

Fig. 3 illustrates our recurrent hand pose refinement network (Hereafter, named *our full model*). For the initial iteration, we use a global stage to get the initial hand pose $\mathbf{P}_{0,g}$ (The subscribe g means "global stage") and get the transformation $\mathbf{A}_{0,g}$ to align the input point cloud to a canonical palm coordinate system, and perform a palm stage and a finger stage to composite the hand pose $\mathbf{P}_0$ and

get the alignment transformation $\mathbf{A}_0$ to a canonical palm coordinate system. For the t-th recurrent iterations, we align the input point cloud to a canonical palm coordinate system with a transformation $\mathbf{A}_{t-1}$ determined with an estimated hand pose of the t-1-th iteration, and extract the pose-aware feature $\mathbf{f}_t$ of the aligned point cloud by PointNet++ (the feature before the last multi-layer perception (MLP)), and add LSTM module between features of the neighboring recurrent iterations to learn higher-order dependencies. Formally,

$$(\mathbf{c}_t, \mathbf{h}_t) = LSTM(\mathbf{c}_{t-1}, \mathbf{h}_{t-1}, \mathbf{f}_t) \quad (1)$$

where the memory cell $\mathbf{c}_t$ and the hidden state $\mathbf{h}_t$ are the functions of previous $\mathbf{c}_{t-1}$, $\mathbf{h}_{t-1}$ and the pose-aware feature $\mathbf{f}_t$ extracted by PointNet++ for the t-th iteration.

After we obtain the hidden state $\mathbf{h}_t$, we use two MLP layers to regress the hand pose $\mathbf{P}_{t,palm}$ at the palm stage. Then, we use $\mathbf{P}_{t,palm}$ to get the transformations $\mathbf{A}_{t,p} = \{\mathbf{A}_{t,p}^i\}_{i=1}^5$ to align each finger coordinate system, perform the hand pose estimation of the finger stage, and composite the hand pose $\mathbf{P}_t$ and get the alignment transformation $\mathbf{A}_t$. Similar to Sec. 3.1.5, the final hand pose is composed by the estimated hand pose from the palm stage and the finger stage of the last recurrent iteration.

In hand gesture related applications, depth sequences are often used as input rather than single depth (e.g. NYU dataset [20]), and thus temporal priors of hand pose sequences can be exploited to improve the robustness of hand pose tracking. Our method can be easily integrated in a tracking framework, which we refer as *hand pose tracking strategy*. In particular, for the first frame, we use the network structure in Fig. 3. For the rest of the sequence, we rely on the estimated hand pose from the previous frame to replace the transformation $\mathbf{A}_{0,g}$ via the global stage in Fig. 3.

3.4 Loss Function

As shown in Fig. 3, our model adopts T recurrent iterations, it has a global stage, and each recurrent iteration contains palm stage and 5 finger stages. So, our total loss function contains loss function of the global stage L_{global}, $T+1$ loss functions of palm stage, i.e. $L_{palm}^t (t = 0, ..., T$, and $T+1$ loss functions of the finger stage for each finger, i.e. $L_{finger,i}^t (i = 1, ..., 5, t = 0, ..., T)$.

The total loss function can be defined as follows:

$$L = L_{global} + \sum_{t=0}^{T} \lambda_p L_{palm}^t + \sum_{t=0}^{T} \sum_{i=1}^{5} \lambda_f L_{finger,i}^t \quad (2)$$

where L_{global} is set to be the ℓ_2 distance between the predicted joints of the global stage and the ground truth ones, and L_{palm}^t is set to be the ℓ_2 distance between the predicted joints of the palm stage at stage t and the ground truth joints. Similarly, $L_{finger,i}^t$ is set to the ℓ_2 distance between the corresponding predicted joints in the i-th finger stage and the ground truth. In our experiments, we set the loss weights λ_p and λ_f to 1.

3.5 Implementation Details

Next, we describe the implementation details of our method.

3.5.1 Data Augmentation

Hand pose datasets are usually collected with a small number of subjects, e.g. two users for NYU hand pose dataset [20], and thus fail to include hands with different configurations, e.g. bone length, hand shape. To overcome these limitations, we adopt a standard data augmentation strategy as [23], which shows improvements for the final performance. Instead of augmenting the dataset once as preprocessing step [26], we conduct an online augmentation during the training phase. We perform data augmentation by applying 3D rotation and scaling on the ground truth pose and the point cloud from the input depth.

We perform in-plane rotations around z-axis of the camera's coordinate system, and also out-plane rotation around x, y axis of the camera's coordinate system. We select the in-plane rotation angle (around z axis) uniformly from the interval [0, 360] degrees, and choose the out-plane rotation (around x, y axis) uniformly from the interval $[-30, 30]$ degrees. We scale the point clouds independently along x, y, z axes of the camera's coordinate system with scaling factors chosen randomly from $[1/1.2, 1.2]$.

3.5.2 Training Schema

We follow Oberweger *et al.* [13] to crop the hand regions and preprocess the raw depth, which is used for training and testing of all models. We train our models on a computer with Intel CPU i7 4790K 4.00GHz, 32GB of RAM and an Nvidia GeForce Titan X GPU. Our model is implemented with Tensorflow framework. During training, the batch size and weight decay are set to be 64 and 0.0005, respectively. The training epochs are set to 20, the initial learning rate is set to be 0.001, and then we reduce learning rate by half when loss stops decreasing for five epochs. Different recurrent iterations share the network parameters, and different fingers also share the network parameters.

4 Experiments

In this section, we first compare our hand pose model with SoTA methods on three main datasets with different skeleton structures and joint numbers, and then conduct ablation study on the impact of pose-guided 3D alignment and the impact of recurrent hand pose refinement.

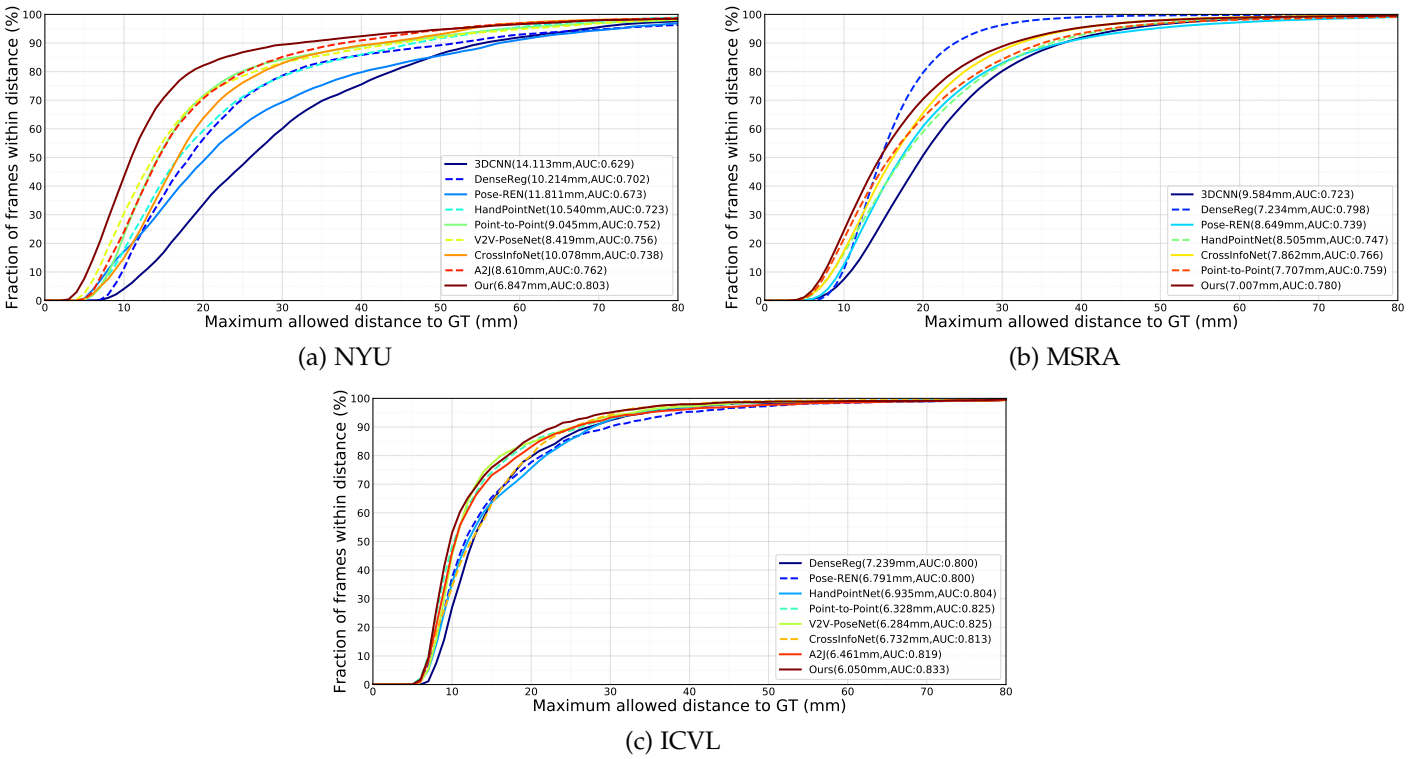
4.1 Datasets and Evaluation Metrics

We evaluate our method on widely-used NYU dataset [20], MSRA dataset [8] and ICVL dataset [7].

NYU Hand Pose dataset (NYU dataset) contains 72,757 frames for training and 8,252 frames for testing. For each frame, depth images of three views (front and two side views) are provided. NYU dataset provides 36 hand joints' 3D locations, but we follow the protocol in [20] where only a subset of 14 hand joints are used for evaluations.

MSRA Hand Pose dataset (MSRA dataset) contains nine subjects, each subject contains 17 hand gestures, and each hand gesture contains about 500 frames. The dataset is annotated with 3D locations of 21 hand joints.

ICVL Hand Pose dataset (ICVL dataset) has over 22K training depth images and two testing sequences, with each about 800 frames.



(a) NYU (b) MSRA

(c) ICVL

Fig. 7. Comparison to SoTA methods on NYU [20], MSRA [8], ICVL [7] datasets. We show percentage of frames in the testing examples under different error thresholds.

TABLE 2
Mean joint error comparison of our method with SoTA methods on NYU, MSRA and ICVL datasets. Ours: our model with 4 LSTM modules, Ours(w/o LSTM): our model without recurrent LSTM modules, Ours(+video): Our full model with hand pose tracking strategy. "*" means that the annotations for MSRA dataset for V2V-PoseNet are different from the other work, which can be found in [43].

Method	NYU	MSRA	ICVL
3DCNN [26]	14.113	9.584	-
DenseReg [11]	10.214	7.234	7.239
Pose-REN [44]	11.81	8.65	6.79
HandPointNet [33]	10.54	8.505	6.94
Point-to-Point [28]	9.045	7.707	6.328
V2V-PoseNet [27]	8.419	7.59*	6.28
CrossInfoNet [36]	10.08	7.86	6.73
A2J [12]	8.61	-	6.46
FeatureMapping [24]	7.441	-	-
Ours(w/o LSTM)	7.693	7.354	6.407
Ours	**6.847**	**7.007**	**6.050**
Ours(+Video)	**6.574**	**6.988**	**6.022**

Evaluation Metrics. We evaluate the hand joint estimation performance using standard metrics as proposed in [7], which are widely used in many hand pose estimation work [8] [13] [42], including the mean joint errors, Area Under Curve (AUC) [9] and the percentage of test examples that have all predicted joint errors within a given distance threshold from the ground truth. Since the MSRA dataset does not contain the splits of the training and testing sets, we follow HandPointNet [33] to evaluate hand pose accuracy on the MSRA dataset using leave-one-subject-out cross-validation protocol and report the average metrics.

4.2 Comparison to State-of-the-art Methods

We compare our method on NYU, MSRA and ICVL datasets with the SoTA methods, including the methods using 3D volume representation such as 3DCNN [26] and V2V-PoseNet [27], the methods using point clouds representation such as HandPointNet [33] and Point-to-Point [28], and the methods using depth image such as DenseReg [11], Pose-REN [44], CrossInfoNet [36], A2J [12] and FeatureMapping [24].

The evaluation results on NYU dataset can be found in Fig. 7 (a), Fig. 8 (a) and Table 2. In the comparison to [27], [28], [11] and [26], we train our network with single view data of NYU, and the mean joint error is 6.847mm (see "Ours" in Table 2). Our method outperforms all the other methods on the mean joint error. Our method outperforms all the other methods over all the error thresholds for the percentage of good frames. Specifically, the AUC of the percentage of good frames within the error threshold of 80mm for our method is about 4.7, 4.1, 5.1, and 8 percentage points higher than those of V2V-PoseNet [27], A2J [12], Point-to-Point [28], and HandPointNet [33], respectively.

The evaluation results on MSRA dataset can be found in Fig. 7(b), Fig. 8(b) and Table 2. From Fig. 8(b) and Table 2, we can see that ours is still significantly better than the SoTA methods. As shown in Fig. 7(b), we observe that the AUC of the percentage of of good frames for our method is higher than most of the methods, and is comparable to DenseReg [11] (Especially, our method performs better than DenseReg at low joint error threshold). As noted in [45] and [33], part of 3D hand joint annotations in MSRA dataset are labeled with significant errors and thus the evaluation on MSRA

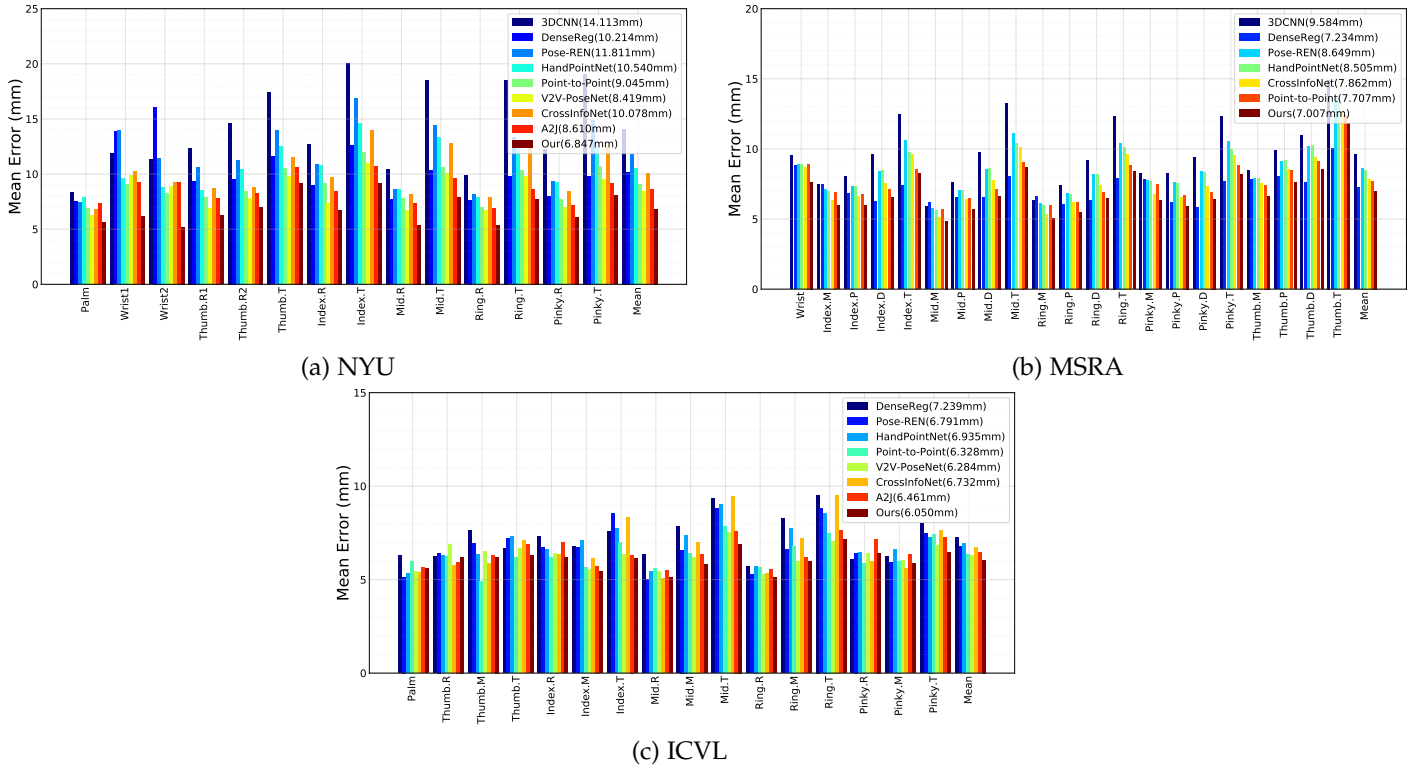


Fig. 8. Comparison to SoTA methods on NYU [20], MSRA [8], ICVL [7] datasets. We show mean joint errors for all the test examples. The palm and fingers are indexed as palm, thumb, index, middle, ring, pinky, wrist. The definitions of joint symbols 'R', 'M', 'P' , 'D' and 'T' can be found in Fig. 4.

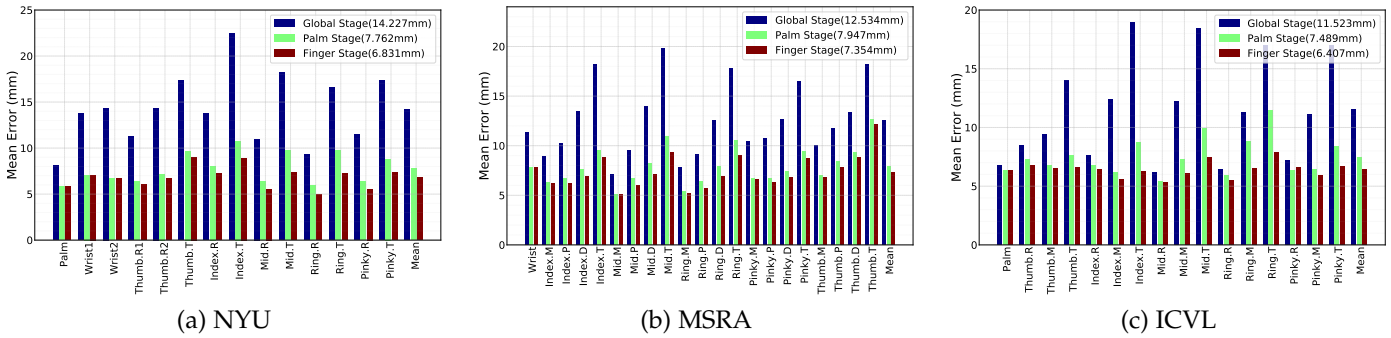


Fig. 9. Influence of stages on NYU, MSRA and ICVL dataset. We show mean joint errors for all the test examples.

dataset may be less meaningful. Since the annotations for MSRA dataset for V2V-PoseNet are slightly different from the other work [43], we do not show the results by V2V-PoseNet in Fig. 7(b) and Fig. 8(b) for fair comparison.

The evaluation results on ICVL dataset can be found in Fig. 8(c), Fig. 7(c) and Table 2. Our method achieves the best performance of the percentage of good frames within the error threshold of 80mm, which outperforms all the SoTA methods. The mean joint error of our method is 0.230mm – 1.189mm lower than those of other methods.

4.3 Impact of Pose-guided 3D Alignment

In order to verify whether the proposed pose-guided 3D alignment can reduce the variance of the input space and improve the hand pose estimation performance, we use our pipeline without the recurrent hand pose refinement to compare the hand pose estimation on NYU dataset with and without pose-guided 3D alignment, influence of scale in the alignment, the other types of alignments, the performance with different kinds of feature extraction (2D/3D) and alignment (2D/3D), and performance of different 3D representations using 3D alignment. Moreover, we conduct visualization analysis of features learned with and without alignment using t-distributed Stochastic Neighbour Embedding (t-SNE) [46]. In this study, we train our model on frames of three views of NYU dataset.

Is 3D alignment important? Fig. 9 shows the performance comparison as the number of stages increases. The palm stage has the same network as the global stage, and the results of the palm stage do not composite the results in the global stage, but with the palm alignment the mean joint errors drop by 6.5mm, 4.6mm and 4.0mm on NYU, ICVL and MSRA datasets, which verifies that the 3D alignment does improve the hand pose performance. The finger stage model has a better performance than the palm stage model, thus more stages are also effective to achieve better hand

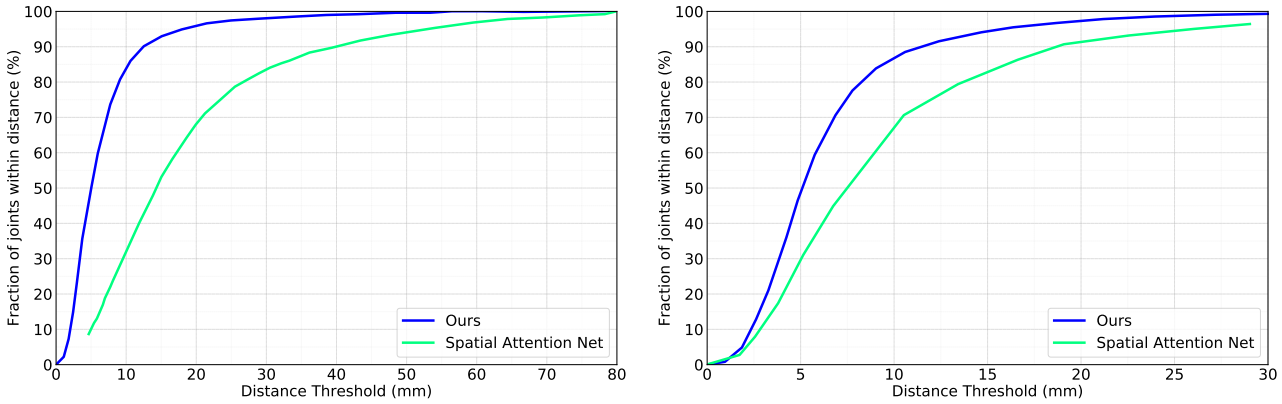


Fig. 10. Comparison to the cascaded method [10] (spatial attention net). (a) NYU. (b) ICVL.

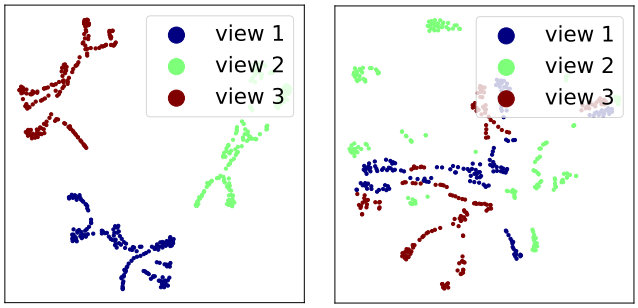


Fig. 11. Visualization of features before the first MLP layer using t-SNE. (a) Feature of the input of global stage. (b) Feature from palm stage. Each point represents a sample, and its color indicates its viewpoint.

pose estimation performance. Even when the error of global stage is large, our method managed to achieve excellent predictions with palm and finger stages, which shows that our method is robust against the potential inaccurate predictions of the previous stage.

We also provide the error lower bound when ground truth alignments are used. These results demonstrate the gain in performance when a perfect alignment module is designed. The mean joint errors with the ground truth alignments are shown in Table 3. The mean joint errors using point cloud (with scale) are 5.99mm and 3.34mm for palm stage and finger stage. We can notice that the mean joint errors for palm stage and finger stage with the estimated alignments are only 1.77mm and 3.49mm worse than the one with perfect alignment.. Therefore, we see that the proposed alignment is effective although the alignment parameters are estimated with the predicted pose.

TABLE 3
Error lower bound (in mm) of our model on NYU dataset.

stage		point cloud
Palm Stage	without scale	7.29
	with scale	5.99
Finger Stage	without scale	5.30
	with scale	**3.34**

Feature visualization with alignment. We further show visualizations of the features learned with and without alignment using t-SNE [46]. NYU dataset consists of images from three camera viewpoints, and we compare the features before the first MLP layer for the NYU testing images from three viewpoints (500 randomly sampled images for each viewpoint). Fig. 11 shows the extracted features of the input of global stage and palm stage using t-SNE. Each point represents a sample projected from a 1024-dimensional vector to 2D. We observe that the features of the three views before alignment are obviously separated, but the gap for the features after alignment is clearly reduced. Therefore, the 3D alignment helps to extract view-invariant features.

Influence of scale in 3D alignment. We evaluate the effect of scale in the 3D alignment for hand pose estimation. Table 3 compares the error lower bound of our model with and without scale. We observe that using scale in the alignment the mean joint errors in palm stage and finger stage drop 1.3mm, 1.96mm. We also compare the mean joint error on NYU dataset with and without estimated scale, and the estimated scale helps to reduce the mean joint error by 0.25mm. Therefore, the scale in 3D alignment is important to improve the hand pose estimation accuracy.

Comparison to 2D alignment in [10]. Our method is related to the spatial attention network based hierarchical hybrid method [10], which conducts alignment in 2D. Fig. 10 shows the performance of our method and 2D alignment [10]. For fair comparison, we follow the same evaluation protocol as [10] using the error of the subset of 11 joint locations for evaluation. We can observe that our method performs consistently better than [10]. For example, the proportion of joints within error threshold 10mm of our method is about 45%, 18% higher than those of the method in [10] on NYU and ICVL, respectively.

Our alignment vs. PCA or T-net alignment. To investigate whether our method learns better alignment and view invariant feature, we compare the 2D t-SNE embedding of 3D hand joints after the 1st alignment (i.e. palm stage) with our alignment, the popular alignment approaches such as learning-based alignment T-net in [30], [31] and PCA in [28] in Fig. 12. Each point represents a 2D embedding of a 3D hand joint, and each color represents the spatial distribution of a specific hand joint. We observe that the intra-joint distance with the network using our alignment is smaller than those without alignment or with T-net and PCA align-

ments, and the different joints are more separated than those without alignment or with T-net and PCA alignments. In Table 4, we compare the mean joint error of our pose-guided alignment to PCA and T-net under our palm stage network. We can see that our alignment is significantly better than the PCA alignment used in [28] and the T-net alignment used in [30], [31], which shows the pose-guided alignment is important for good performance.

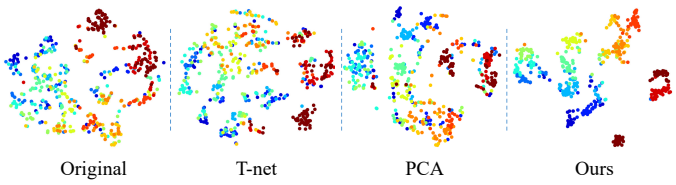


Fig. 12. 2D t-SNE embedding of 3D hand joints after the 1st alignment (i.e. palm stage). Each point represents a 2D embedding of a 3D joint.

TABLE 4
Comparison of mean joint errors (in mm) in the palm stage using our pose-guided alignment, PCA and T-net alignments.

Dataset	PCA	T-net	Ours
NYU	12.17	15.71	**8.75**

Is 3D feature extraction important? We compare the performance with different kinds of feature extraction (2D/3D) and alignment (2D/3D) (see Fig. 2). We adopt the same architecture as that of 3D feature + 3D alignment using point cloud. The transformation only consists of rotation and translation. Details of other kinds of feature extraction and alignment are shown in the supplementary document.

Table 5 shows the performance comparison with different feature extraction and alignment. We observe that the mean joint error with 3D feature extraction and 3D alignment outperforms 2D feature+2D alignment, 2D feature+3D alignment, 3D feature+2D alignment by 2.87mm, 0.98mm and 2.37mm, respectively. Therefore, 3D feature extraction and 3D alignment are both effective for hand pose estimation.

TABLE 5
Comparison of mean joint errors (in mm) with different kinds of feature extraction (feat.) and alignment (align.).

feat. + align.	2D+2D	2D+3D	3D+2D	3D+3D
Global Stage	15.22	15.32	15.41	15.27
Palm Stage	11.00	10.23	10.91	8.75
Finger Stage	10.68	8.78	10.19	**7.80**

4.4 Impact of Recurrent Hand Pose Refinement

In this section, we investigate whether the recurrent hand pose refinement can further enhance hand pose estimation performance, we conduct hand pose comparison study mainly on NYU dataset to verify our model design. More specifically, we analyze two issues, the effect of the stages of LSTM modules, and the performance of our method with LSTM modules for hand pose tracking.

Effect of the iterations of LSTM modules. Table 6 shows the performance under different numbers of LSTM modules. In this experiment, we train our model on frames of single view of NYU dataset. We observe that the mean joint error decreases as the iteration of the LSTM modules increases. Although our method does not use additional dataset for pretraining as [24], our method can achieve the SoTA performance by [24] at the first iteration of LSTM, and gains 0.74mm to [24] at the third iteration of LSTM (see Table 2). Moreover, as shown in Table 2, our full model performs better than our model without recurrent LSTM modules. Therefore, our recurrent module is effective to improve hand pose estimation performance.

Performance of our method with hand pose tracking strategy. Our method can be easily integrated in a tracking framework. Table 6 also shows the performance of our method with the proposed hand pose tracking strategy (see NYU (Video)). We observe that the results with hand pose tracking on NYU dataset are better than those with single-frame hand pose estimation. Moreover, as shown in Table 2, the hand pose tracking strategy (see "Ours(+video)") can further enhance the hand pose estimation performance of our method. Thus, our hand pose tracking strategy is helpful to enhance hand pose estimation performance on temporal sequences.

TABLE 6
Comparison of mean joint error (in mm) with different recurrent iterations T of LSTM module on NYU dataset. "NYU(Video)" means the results using our hand pose tracking strategy.

T	0	1	2	3	4
NYU	7.693	6.926	6.917	6.706	6.847
NYU(Video)	6.797	6.577	6.509	6.493	6.574

4.5 Runtime

The network inference runtime of our method without LSTM is 17ms in average with a Nvidia Titan X GPU. Thus, our method using point cloud runs in real-time at over 58fps. Table 7 shows network inference fps with different recurrent iterations of LSTM module. In practice, the number of recurrent iterations can be used as a hyper-parameter to balance the accuracy and runtime.

TABLE 7
Inference fps with different recurrent iterations T of LSTM module.

T	0	1	2	3	4
fps	58.66	40.99	31.81	25.15	21.44

4.6 Qualitative results

In this section, we provide qualitative results for NYU, ICVL and MSRA datasets.

Fig. 13 shows qualitative results of the different stages in our pipeline on NYU dataset. The estimated hand joints become more accurate from global stage to palm stage, and they are further improved after finger stage. For example, as shown in the first column and sixth column of Fig. 13, our stage-wise pipeline consistently improves the joint locations

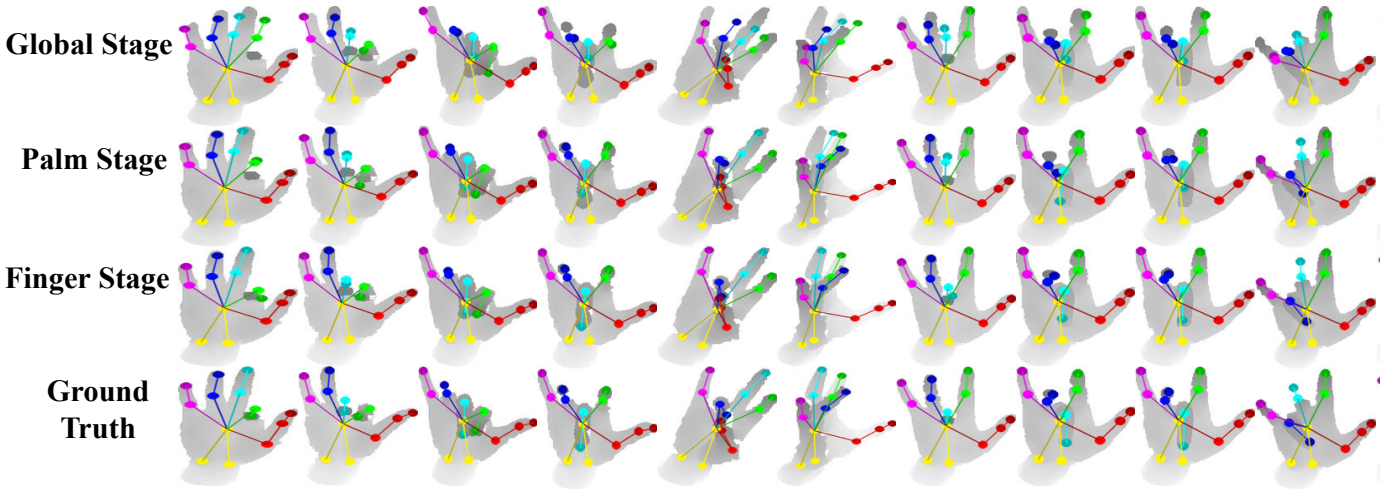


Fig. 13. Qualitative results of different stages on NYU dataset. We show hand pose estimation results of global stage, palm stage and finger stage. As shown in the first column and sixth column, our stage-wise pipeline consistently improves the joint locations of the middle and index fingers.

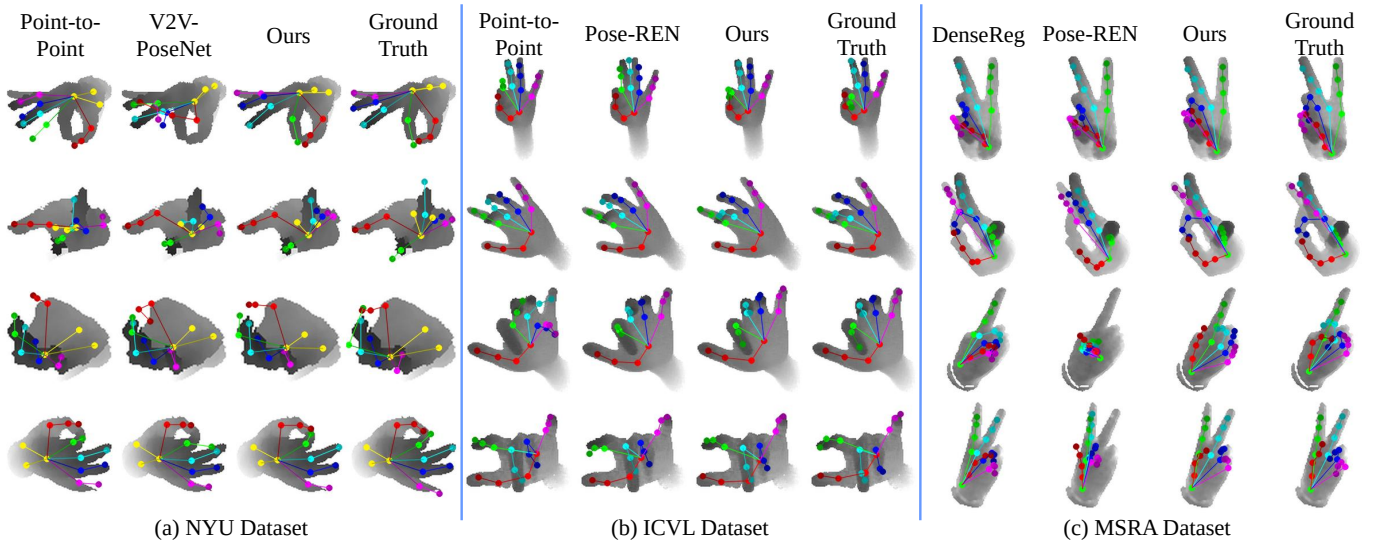


Fig. 14. Qualitative results for NYU, ICVL and MSRA datasets. We compare our method with other methods (in the 1st column and 2nd column for each dataset). The ground truth hand joint locations are presented in the last column for each dataset. We show hand joint locations and bones with the depth image. Different hand joints and bones are visualized using different colors.

of the middle and index fingers. Fig. 14(a) shows the comparison results with the SoTA methods on NYU dataset. It can be clearly seen that our method has better performance than Point-to-Point [28] and V2V-PoseNet [27]. For example, as shown in the first row of Fig. 14(a), all the joints with our method are close to the ground truth, however, the joints in the index, ring, pinky fingers by [28] have large displacement to the ground truth, and the joints of all the fingers by [27] are not correctly located.

Fig. 14(b) shows the comparison results with other methods on ICVL dataset. Our method outperforms Point-to-Point [28] and Pose-REN [44]. For example, as shown in the second row, the joint locations of the middle fingers using our method are better than those generated by [28], [44].

Fig. 14(c) shows the comparison results with other methods on MSRA dataset. We can see that our method is better than DenseReg [11] and Pose-REN [44]. For example, as shown in the third row, all the joints with our method are closer to the ground truth.

We also conduct experiments with other 3D networks such as DGCNN [41] and 3D CNN [25], [26] as the backbones in our method, and the experiments show that our recurrent network architecture based on cascaded pose-guided 3D alignments can still improve the performance over baselines. The evaluation on these backbones and more results can be found in the supplementary document.

5 Conclusion

We presented a recurrent hand pose model using a cascaded pose-guided 3D alignment that operates directly on 3D point clouds. Our approach encodes depth image as 3D representations such as point clouds. We propose a new cascaded 3D pose-guided alignment strategy in a coarse-to-fine fashion to extract view-independent features and facilitate the pose estimation of different hand parts. Specifically, our model consists of three stages, which estimate 3D hand pose

in camera, palm, and finger coordinate systems. Our method produces more intuitive and geometrically reasonable input that reduces the issue due to the different hand viewpoints and facilitates the hand pose estimation. Benefiting from our pose-guided 3D alignment, we also design a new recurrent hand pose model to get more effective recurrent pose-aware features by modeling the dependencies among sequential features for 3D hand pose estimation, and then further enhance the performance of our method. Our method achieves the SoTA performance on main benchmark datasets with different skeleton structures and joint numbers.

Although promising performance has been achieved, there are still some issues to be addressed in future work. For example, in the finger stage, if the relevant neighbor point clouds of a finger is very sparse, it will bring challenges to extract effective spatial features for the finger stage network. In addition, due to heavy occlusions, our method may not achieve high-quality hand pose estimation under hand-object interaction scenarios. We will extend the framework for these challenging scenarios in future.

ACKNOWLEDGMENTS

The authors would like to thank the editors and reviewers for their careful review and inspiring suggestions. This work was supported in part by the National Key R&D Program of China under Grant 2021YFF0307702, National Natural Science Foundation of China (No. 61473276), Beijing Natural Science Foundation (L182052), and the Distinguished Young Researcher Program, Institute of Software, Chinese Academy of Sciences.

REFERENCES

[1] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly, "Vision-based hand pose estimation: A review," *Computer Vision and Image Understanding*, vol. 108, no. 1-2, pp. 52–73, 2007.

[2] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, T. Sharp, E. Soto, D. Sweeney, J. Valentin, B. Luff *et al.*, "Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences," *ACM Transactions on Graphics*, vol. 35, no. 4, p. 143, 2016.

[3] J. Taylor, V. Tankovich, D. Tang, C. Keskin, D. Kim, P. Davidson, A. Kowdle, and S. Izadi, "Articulated distance fields for ultra-fast tracking of hands interacting," *ACM Transactions on Graphics*, vol. 36, no. 6, p. 244, 2017.

[4] A. Tkach, M. Pauly, and A. Tagliasacchi, "Sphere-meshes for real-time hand modeling and tracking," *ACM Transactions on Graphics*, vol. 35, no. 6, p. 222, 2016.

[5] J. Romero, D. Tzionas, and M. J. Black, "Embodied hands: Modeling and capturing hands and bodies together," *ACM Transactions on Graphics*, vol. 36, no. 6, p. 245, 2017.

[6] D. Tang, T.-H. Yu, and T.-K. Kim, "Real-time articulated hand pose estimation using semi-supervised transductive regression forests," in *ICCV*, 2013, pp. 3224–3231.

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

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

[9] C. Zimmermann and T. Brox, "Learning to estimate 3d hand pose from single rgb images," in *ICCV*, 2017, pp. 2380–7504.

[10] Q. Ye, S. Yuan, and T.-K. Kim, "Spatial attention deep net with partial pso for hierarchical hybrid hand pose estimation," in *ECCV*, 2016, pp. 346–361.

[11] C. Wan, T. Probst, L. Van Gool, and A. Yao, "Dense 3d regression for hand pose estimation," in *CVPR*, 2018, pp. 5147–5156.

[12] F. Xiong, B. Zhang, Y. Xiao, Z. Cao, T. Yu, J. Zhou Tianyi, and J. Yuan, "A2j: Anchor-to-joint regression network for 3d articulated pose estimation from a single depth image," in *ICCV*, 2019, pp. 793–802.

[13] M. Oberweger, P. Wohlhart, and V. Lepetit, "Hands Deep in Deep Learning for Hand Pose Estimation," in *CVWW*, 2015.

[14] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei *et al.*, "Accurate, robust, and flexible real-time hand tracking," in *CHI*, 2015, pp. 3633–3642.

[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 *CVPR*, 2014, pp. 644–651.

[16] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, "Realtime and robust hand tracking from depth," in *CVPR*, 2014, pp. 1106–1113.

[17] A. Tkach, A. Tagliasacchi, E. Remelli, M. Pauly, and A. Fitzgibbon, "Online generative model personalization for hand tracking," *ACM Transactions on Graphics*, vol. 36, no. 6, p. 243, 2017.

[18] 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 *CVPR*, 2015, pp. 2540–2548.

[19] D. Joseph Tan, T. Cashman, J. Taylor, A. Fitzgibbon, D. Tarlow, S. Khamis, S. Izadi, and J. Shotton, "Fits like a glove: Rapid and reliable hand shape personalization," in *CVPR*, 2016, pp. 5610–5619.

[20] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Transactions on Graphics*, vol. 33, no. 5, pp. 1–10, 2014.

[21] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns," in *CVPR*, 2016, pp. 3593–3601.

[22] X. Zhou, Q. Wan, W. Zhang, X. Xue, and Y. Wei, "Model-based deep hand pose estimation," in *IJCAI*, 2016, pp. 2421–2427.

[23] M. Oberweger and V. Lepetit, "Deepprior++: Improving fast and accurate 3d hand pose estimation," in *ICCVW*, 2018, pp. 585–594.

[24] M. Rad, M. Oberweger, and V. Lepetit, "Feature mapping for learning fast and accurate 3d pose inference from synthetic images," in *CVPR*, 2018, pp. 4663–4672.

[25] X. Deng, S. Yang, Y. Zhang, P. Tan, L. Chang, and H. Wang, "Hand3d: Hand pose estimation using 3d neural network," *arXiv preprint arXiv:1704.02224*, 2017.

[26] L. Ge, H. Liang, J. Yuan, and D. Thalmann, "3d convolutional neural networks for efficient and robust hand pose estimation from single depth images," in *CVPR*, 2017, pp. 5679–5688.

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

[28] L. Ge, Z. Ren, and J. Yuan, "Point-to-point regression pointnet for 3d hand pose estimation," *ECCV*, pp. 489–505, 2018.

[29] S. Yuan, Q. Ye, G. Garcia-Hernando, and T.-K. Kim, "The 2017 hands in the million challenge on 3d hand pose estimation," *arXiv preprint arXiv:1707.02237*, 2017.

[30] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *CVPR*, pp. 77–85, 2017.

[31] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *NIPS*, 2017, pp. 5099–5108.

[32] S. Li and D. Lee, "Point-to-pose voting based hand pose estimation using residual permutation equivariant layer," in *CVPR*, 2019, pp. 11 927–11 936.

[33] L. Ge, Y. Cai, J. Weng, and J. Yuan, "Hand pointnet: 3d hand pose estimation using point sets," in *CVPR*, 2018, pp. 8417–8426.

[34] A. Haque, B. Peng, Z. Luo, A. Alahi, S. Yeung, and L. Fei-Fei, "Towards viewpoint invariant 3d human pose estimation," in *ECCV*, 2016, pp. 160–177.

[35] M. Oberweger, P. Wohlhart, and V. Lepetit, "Training a Feedback Loop for Hand Pose Estimation," in *ICCV*, 2015, pp. 3316–3324.

[36] K. Du, X. Lin, Y. Sun, and X. Ma, "Crossinfonet: Multi-task information sharing based hand pose estimation," in *CVPR*, 2019, pp. 9896–9905.

[37] J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik, "Human pose estimation with iterative error feedback," in *CVPR*, 2016, pp. 4733–4742.

[38] V. Sarode, X. Li, H. Goforth, Y. Aoki, A. Dhagat, R. A. Srivatsan, S. Lucey, and H. Choset, "One framework to register them all: Pointnet encoding for point cloud alignment," *arXiv preprint arXiv:1912.05766*, 2019.
[39] Y. Wu, W. Ji, X. Li, G. Wang, J. Yin, and F. Wu, "Context-aware deep spatiotemporal network for hand pose estimation from depth images," *IEEE Transactions on Cybernetics*, vol. 50, no. 2, pp. 787–797, 2020.
[40] J. Yang, H. J. Chang, S. Lee, and N. Kwak, "Seqhand: Rgb-sequence-based 3d hand pose and shape estimation," in *ECCV*, 2020, pp. 122–139.
[41] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics*, vol. 38, no. 5, pp. 1–12, 2019.
[42] J. S. Supancic III, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan, "Depth-based hand pose estimation: methods, data, and challenges," *ICCV*, pp. 1868–1876, 2015.
[43] X. Chen, "Evaluations on hand pose estimation," 2021. [Online]. Available: https://github.com/xinghaochen/awesome-hand-pose-estimation/tree/master/evaluation
[44] X. Chen, G. Wang, H. Guo, and C. Zhang, "Pose guided structured region ensemble network for cascaded hand pose estimation," *Neurocomputing*, vol. 395, pp. 138–149, 2020.
[45] M. Oberweger, G. Riegler, P. Wohlhart, and V. Lepetit, "Efficiently creating 3d training data for fine hand pose estimation," in *CVPR*, 2016, pp. 4957–4965.
[46] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

Xiaoming Deng is currently a Professor with the Institute of Software, Chinese Academy of Sciences (CAS). He received the Bachelor and Master degrees from Wuhan University, and the Ph.D. degree from the Institute of Automation, CAS. He has been a Research Fellow at the National University of Singapore, and a Postdoctoral Fellow at the Institute of Computing Technology, CAS, respectively. His main research topics are in computer vision, and specifically related to 3D reconstruction, human motion tracking and synthesis, and natural user interfaces.

Dexin Zuo received the Master degree from the Institute of Software, Chinese Academy of Sciences. Before that, he received a Bachelor degree from Nankai University. His main research interest is computer vision and human-computer interaction.

Yinda Zhang is Research Scientist at Google. He received the Ph.D. degree from Princeton University, advised by Professor Thomas Funkhouser. Before that, he received a Bachelor degree from Department of Automation, Tsinghua University and a Master degree from Department of Electrical and Computer Engineering, National University of Singapore under the supervision of Professor Ping Tan and Professor Shuicheng Yan. He is currently working on 3D context model, 3D deep learning, and scene understanding.

Zhaopeng Cui received the Ph.D. degree from Simon Fraser University in 2017. He was a senior researcher at ETH Zurich. He is currently a research professor in the College of Computer Science, Zhejiang University. His research interests include 3D mapping and localization, 3D scene understanding, image and video editing.

Jian Cheng is a Master graduate student at the Institute of Software, Chinese Academy of Sciences. Before that, he received a Bachelor degree from Xiangtan University, in 2019. His main research interest is computer vision.

Ping Tan is currently an Associate Professor with the School of Computing Science of the Simon Fraser University in Canada. Before that, he was an Associate Professor with the National University of Singapore. He received the Ph.D. degree from the Hong Kong University of Science and Technology in 2007, and the bachelors and masters degrees from Shanghai Jiao Tong University in China in 2000 and 2003, respectively. His research interests include computer vision, computer graphics, and robotics. He has served as an Editorial Board Member of the IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), the International Journal of Computer Vision (IJCV), Computer Graphics Forum (CGF), and the Machine Vision and Applications (MVA). He served as an Area Chair for CVPR, SIGGRAPH, SIGGRAPH Asia, and IROS.

Liang Chang is currently currently an Associate Professor with the School of Artificial Intelligence, Beijing Normal University. She received the Bachelor and Master degrees from Wuhan University, and the Ph.D. degree from Institute of Automation, Chinese Academy of Sciences. Her research interests are computer vision and machine learning.

Marc Pollefeys received the MSc degree in electrical engineering and the PhD degree in computer vision from the KU Leuven in Belgium, in 1994 and 1999, respectively. He is a professor of computer science with ETH Zurich and director of science with Microsoft working on HoloLens and Mixed Reality. He is best known for his work in 3D computer vision, having been the first to develop a software pipeline to automatically turn photographs into 3D models, but also works on robotics, graphics, and machine learning problems. Other noteworthy projects he worked on with collaborators with UNC Chapel Hill and ETH Zurich are real-time 3D scanning with mobile devices, a real-time pipeline for 3D reconstruction of cities from vehicle mounted-cameras, camera-based self-driving cars and the first fully autonomous vision-based drone. Most recently his academic research has focused on combining 3D reconstruction with semantic scene understanding. He became an assistant professor with the University of North Carolina in Chapel Hill in 2002 and joined ETH Zurich as a full professor in 2007. He is a fellow of the IEEE.

Sean Fanello is a Research Scientist and Manager at Google where he leads the effort on human performance capture to solve real-world, human perception tasks using machine learning. Previously, he was a Senior Scientist and a Founding Team Member at perceptiveIO, Inc., where he developed computer vision and machine learning algorithms for 3D sensing, visual recognition and human-computer interaction. Prior to that, he was a Post-Doc Researcher in the Interactive 3D Technologies (I3D) group at Microsoft Research Redmond where he substantially contributed to the Hololens 3D sensing capabilities. He was also one of the leading members of the Holoportation project. He obtained his PhD in Robotics, Cognition and Interaction Technologies at the Italian Institute of Technology in collaboration with the University of Genoa in 2013. His research interests include 3D performance capture, photorealistic rendering, neural rendering, relighting, viewpoint synthesis.

Hongan Wang received the PhD degree in computer science from Institute of Software, Chinese Academy of Sciences. He is a Professor and the Director of Beijing Key Laboratory of Human-Computer Interactions Laboratory at Institute of Software, Chinese Academy of Sciences. His research interest is real-time reasoning and intelligent user interaction.