Real-time 3D Hand Pose Estimation with 3D Convolutional Neural Networks

Liuhao Ge, Hui Liang, *Member, IEEE,* Junsong Yuan, *Senior Member, IEEE,* and Daniel Thalmann

Abstract—In this paper, we present a novel method for real-time 3D hand pose estimation from single depth images using 3D Convolutional Neural Networks (CNNs). Image-based features extracted by 2D CNNs are not directly suitable for 3D hand pose estimation due to the lack of 3D spatial information. Our proposed 3D CNN-based method, taking a 3D volumetric representation of the hand depth image as input and extracting 3D features from the volumetric input, can capture the 3D spatial structure of the hand and accurately regress full 3D hand pose in a single pass. In order to make the 3D CNN robust to variations in hand sizes and global orientations, we perform 3D data augmentation on the training data. To further improve the estimation accuracy, we propose applying the 3D deep network architectures and leveraging the complete hand surface as intermediate supervision for learning 3D hand pose from depth images. Extensive experiments on three challenging datasets demonstrate that our proposed approach outperforms baselines and state-of-the-art methods. A cross-dataset experiment also shows that our method has good generalization ability. Furthermore, our method is fast as our implementation runs at over 91 frames per second on a standard computer with a single GPU.

Index Terms—3D hand pose estimation, 3D convolutional neural networks, deep learning

1 Introduction

With the success of real-time human body pose estimation [1], [2], [3] and the availability of mid-range and short-range depth cameras, such as Intel RealSense, SoftKinetic and Primesense Carmine, accurate real-time 3D hand pose estimation has aroused a lot of research attention in recent years [4], [5], [6], [7], [8], [9], [10], [11], [12]. Articulated 3D hand pose estimation is one of the core technologies for human computer interaction in virtual reality and augmented reality applications, since this technology provides a natural way for users to interact with virtual environments and virtual objects. The estimated 3D hand pose can also be used for gesture recognition, such as sign language recognition [13], [14] and driver hand gesture analysis [15], [16]. However, it is still challenging to achieve efficient and robust hand pose estimation performance because of large variations in hand pose, high dimensionality of hand motion, severe self-occlusion and self-similarity of fingers in the depth image.

Many recent works on hand pose estimation have achieved good performance due to the success of Convolutional Neural Networks (CNNs) [7], [17], [18], [19], [20], [21], [22], [23], [24], [25] and the availability of large hand pose datasets [7], [8], [26]. Most of these methods directly take the depth image as input to 2D CNNs which output heat-maps [7] (Figure 1a), 3D joint locations [17], [18], [20], [21], [24] (Figure 1c) or hand model parameters [22]. Nevertheless, we argue that image-based features extracted by 2D CNNs are not directly suitable for 3D hand pose estimation due to the lack of 3D spatial information. For example, in [18], the initial result of 2D CNN is poor, and it is iteratively refined by a feedback loop to incorporate 3D information from a generative model. Ge et al. [19] better utilize the depth cues by projecting the depth image onto three views and applying multi-view CNNs to regress three views' heat-maps (Figure 1b). However, the multi-view CNNs still cannot fully exploit 3D spatial information in the depth image, since the projection from 3D to 2D will lose certain information. Although increasing the number of views may improve the performance, the computational complexity will increase when using more views.

In this paper, we propose a 3D CNN-based hand pose estimation approach that can capture the 3D spatial structure of the input and accurately regress full 3D hand pose in a single pass, as illustrated in Figure 1d. Specifically, human hand is first segmented from the depth image; the 3D point cloud of the hand is encoded as 3D volumes storing the projective Directional Truncated Signed Distance Function (D-TSDF) [27] values, which are then fed into a 3D convolutional neural network. We design a 3D deep dense network to boost the learning ability of the network. The output of this network is a lower dimensional representation of 3D hand joints' relative locations in the 3D volume. By performing PCA reconstruction and coordinate transformations, we can finally obtain the 3D hand joint locations in the camera's coordinate system. Benefiting from 3D features extracted by 3D convolutions, our method is able to understand the hand pose structure in 3D space and infer 3D hand joint locations efficiently and robustly.

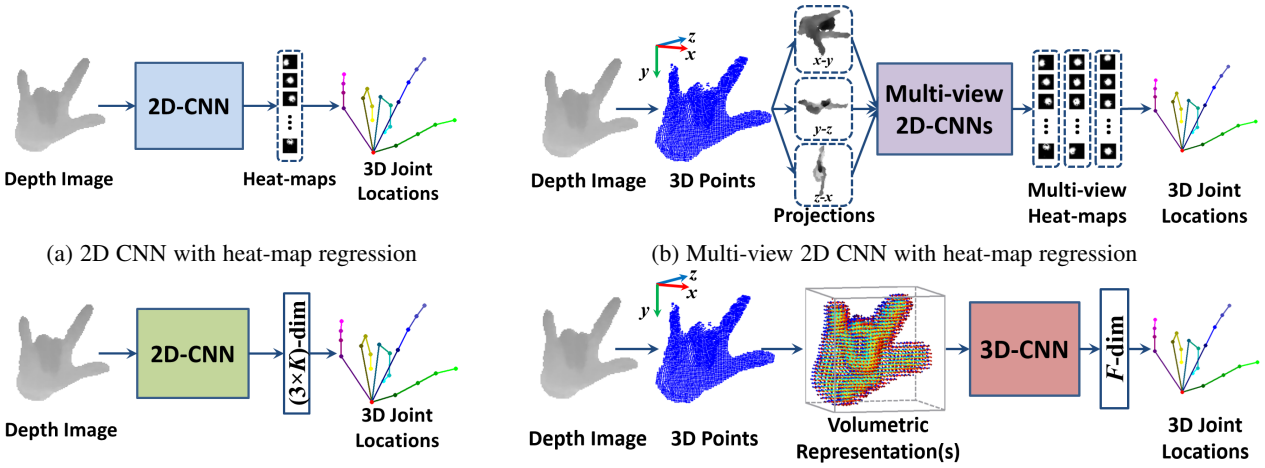
Compared to previous CNN-based methods for hand pose estimation, our proposed 3D CNN-based method has the following advantages:

- Our proposed 3D CNN is capable of learning 3D features from the 3D volumetric representation for accurate 3D hand pose estimation. Compared to 2D CNN-based methods that regress 3D joint locations from 2D features [17], [18], [21], [28], 3D CNN can directly regress 3D joint locations from 3D features in a single pass. This not only achieves superior estimation accuracy, but also avoids the time-consuming

- *L. Ge is with the Institute for Mdeia Innovation, Nanyang Technological University, Singapore 639798. E-mail: ge0001ao@e.ntu.edu.sg*
- *H. Liang and D. Thalmann were with the Institute for Mdeia Innovation, Nanyang Technological University, Singapore 639798. E-mail: hulia@amazon.com, daniel.thalmann@epfl.ch*
- *J. Yuan is with the Department of Computer Science and Engineering, State University of New York at Buffalo, Buffalo NY 14260. E-mail: jsyuan@buffalo.edu*

Manuscript received March 28, 2018.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence



(a) 2D CNN with heat-map regression

(b) Multi-view 2D CNN with heat-map regression

(c) 2D CNN with direct regression

(d) 3D CNN with PCA-based regression

Fig. 1: Different schemes for 3D hand pose estimation. (a) 2D CNN taking depth image as input and outputting heat-maps. (b) Multi-view 2D CNNs taking multi-view projections as inputs and outputting multi-view heat-maps. (c) 2D CNN taking depth image as input and regressing 3D joint locations directly. (d) We use 3D CNN which takes volumetric representations as input and regresses a lower dimensional representation of 3D joint locations.

iterative refinement process.

- Our proposed 3D CNN can be easily trained in an end-to-end manner. Both the 3D shallow plain network and the 3D deep dense network we designed in this paper can run at real-time speed on a single GPU.
- Our proposed method is robust to variations in hand sizes and global orientations, since we perform 3D data augmentation on the training set. Different from traditional data augmentation that performs 2D transformations on 2D images, our proposed 3D data augmentation applies 3D transformations on 3D point clouds, thus can better enrich the training data in 3D space.

This paper is an extension of our conference paper [29]. The new contributions of this paper are summarized as follows:

- We have proposed leveraging the complete hand surface as intermediate supervision for learning 3D hand pose from depth images. Experimental results have shown that, with the intermediate hand surface completion step, the estimation accuracy of 3D hand pose can be further improved.
- We have investigated the performance of deep neural network architectures [30], [31] when applied in our proposed 3D CNN-based framework for 3D hand pose estimation to improve the learning ability. Experimental results have shown that the 3D deep dense network can achieve better performance than the 3D shallow plain network.
- To better understand 3D CNNs, we have visualized the input patterns that produce given activations in the 3D feature volumes. Local and global 3D structures of hand have been observed in these patterns.
- We have conducted more extensive self-comparison experiments and have compared with more state-of-the-art methods on one additional hand pose dataset ICVL [8]. We have also conducted a cross-dataset experiment and qualitatively compared with the Intel RealSense SDK [32]. Experimental results have shown that our method can achieve good performance in real-time and has good generalization ability.

The reminder of this paper is organized as follows: Some related work for 3D hand pose estimation is discussed in Section 2. Since our method takes 3D volumes as input, we first introduces different volumetric representations in Section 3, then describe the proposed 3D CNN-based method in Section 4. Section 5 provides experimental results, and Section 6 concludes this paper.

2 Related Work

Hand pose estimation Methods for hand pose estimation from depth images can be categorized into model-driven approaches, data-driven approaches and hybrid approaches. Model-driven approaches fit an explicit deformable hand model to depth images by minimizing a hand-crafted cost function. The commonly used optimization methods are Particle Swarm Optimization (PSO) [5], Iterative Closest Point (ICP) [33] and their combination [34]. The 3D shape model is represented by Linear Blend Skinning (LBS) model [35], [36], [37], Gaussian mixture model [2], [38], [39], etc. Some models require to define user-specific parameters and motion constraints. These approaches are sensitive to initialization, since they usually take advantage of temporal information. The estimation errors will be accumulated when previous frames' estimations are inaccurate.

Data-driven approaches learn a mapping from depth image to hand pose from training data. Some early works [40], [41], [42] focus on example-based method that searches the most similar images in a dataset to the input hand image, but cannot work well in high dimensional space. Inspired by the pioneering work [1] of human pose estimation, [6], [8], [13], [26], [43], [44], [45] apply random forests and their variants as a discriminative model. Limited by the hand-crafted features, methods based on random forests are difficult to outperform current CNN-based methods in hand pose estimation. Our work is related to the CNN-based data-driven approach. Tompson et al. [7] first propose to employ CNNs to predict heat-maps representing the probability distribution of 2D joint positions in the depth image. Ge et al. [19] improve this method by predicting heat-maps on multiple views in order to better utilize the depth information. Oberweger et al. [18] train a feedback loop containing a discriminative network for initial

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

pose estimation, a generative network for pose synthesizing and a pose update network for improving the pose estimation. Zhou et al. [22] propose to predict hand model parameters instead of the joint locations by adopting CNNs. Sinha et al. [20] extract activation features from CNNs to synchronize hand poses in nearest neighbors by using the matrix completion algorithm. Ye et al. [21] propose a spatial attention network with a hierarchical hybrid method for hand pose estimation. Wan et al. [23] use deep generative models which can exploit unlabeled depth images for training. Guo et al. [24] propose a region ensemble network by dividing feature maps into multiple regions and ensemble regional features to get final prediction. All these methods use 2D filters in 2D CNNs to extract 2D features which are lack of 3D spatial information. Thus, mapping from 2D features to 3D joint locations is difficult. In this work, we lift the 2D CNN to 3D CNN which can understand 3D spatial information and extract 3D features for 3D hand pose estimation.

Hybrid approaches combine a data-driven approach based per-frame reinitialization with a model driven approach based optimization [10], [11], [46], [47], [48]. These methods are usually applied for hand tracking since they utilize temporal information to achieve smooth results. However, in this paper, we focus on 3D hand pose estimation from single depth images without using any temporal information, which can be used for robust reinitialization in hybrid hand tracking approaches.

3D deep learning 3D CNNs have been successfully applied in video and dynamic hand gesture analysis for recognition tasks [16], [49], [50], [51], which regard time as the third dimension. 3D CNNs are also applied to extract 3D features from 3D data, such as CAD models and depth images. 3D ShapeNets [52] learn powerful 3D features by using the Convolutional Deep Belief Network for modeling 3D shapes. Qi et al. [53] show that the 3D CNN with low input volume resolution can still achieve good object classification accuracy by applying subvolume supervision and anisotropic probing. Song and Xiao [27] propose to use 3D CNN for 3D object detection in RGB-D images. Maturana and Scherer [54] propose VoxNet, a 3D CNN that can process LiDAR, RGB-D and CAD data for object recognition. They also apply the 3D CNN for landing zone detection [55]. Yumer and Mitra [56] propose to use the 3D CNN to learn deformation flows from CAD models for 3D shape deformation. Song et al. [57] propose a 3D CNN for semantic scene completion. Although these works achieve state-of-the-art results in their problems, none of them focuses on articulated pose estimation that requires to localize a set of articulated 3D joints from single depth images in real-time.

3 Volumetric Representations

Our proposed 3D CNN-based hand pose estimation method takes a volumetric representation as the input of the neural network. The objective for encoding the observed hand depth image as a volumetric representation is to generate 3D volumes providing sufficient and meaningful information of the hand in 3D space from the depth image in real-time. The 3D volumes will be fed into the 3D CNNs to learn 3D features for subsequent 3D hand joint location regression.

The input depth image of hand is first converted to a set of 3D points, which is denoted as $\mathcal{P} \subset \mathbb{R}^3$, as shown in the first two rows of Figure 2. To create a 3D volume containing $M \times M \times M$ voxels, we first build an axis-aligned bounding box (AABB) for the 3D hand points. AABB is the minimum bounding box of which x, y, z axes are respectively aligned with x, y, z axes of the camera's coordinate system. The 3D volume's center is set at the center of AABB, and its faces are set to be parallel to those of AABB. The edge length of a voxel is set as:

$$l_{voxel} = \max\{l_x, l_y, l_z\}/M, \tag{1}$$

where l_x, l_y, l_z are AABB's edge lengths along three directions; M is the volume resolution value. We denote the value of a voxel $\boldsymbol{v}$ as $F(\boldsymbol{v})$ that can be determined by using the occupancy model (Section 3.1) or the truncated signed distance function (Section 3.2).

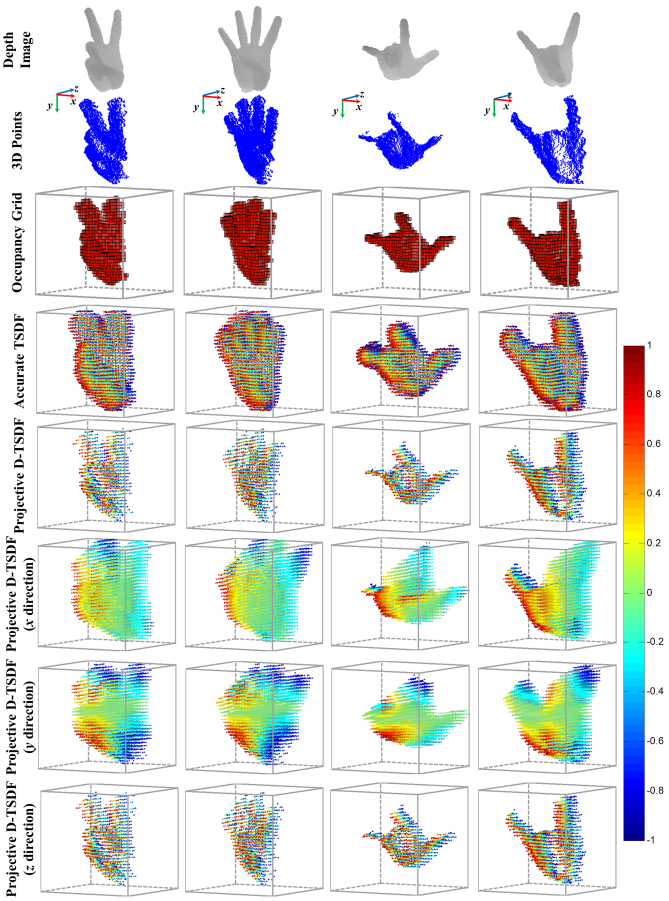


Fig. 2: Visualization of different volumetric representations. For the last five rows, we only visualize voxels of which values are less than 1 and larger than -1 by using the color map shown in this figure. The volume resolution is $32 \times 32 \times 32$.

3.1 Occupancy Grid

The occupancy grid is a binary grid representing occupied and unoccupied voxels in the 3D volume, as shown in the 3rd row of Figure 2. Thus, the voxel value is determined as:

$$F(v) = \begin{cases} 1 & \exists \boldsymbol{p} \in \mathcal{P},\ s.t.\ D_{Che}(\boldsymbol{v}_c, \boldsymbol{p}) \leq l_{voxel}/2 \\ 0 & otherwise \end{cases}, \tag{2}$$

where $D_{Che}(\boldsymbol{p}, \boldsymbol{q}) = \max_i(|\boldsymbol{p}_i - \boldsymbol{q}_i|)$ is the Chebyshev distance; $\boldsymbol{v}_c$ is the center of the voxel $\boldsymbol{v}$.

If the input is a 3D CAD model where the 3D information is fully known, the occupancy grid is sufficient to represent the complete 3D model. However, in our problem, the input is a 2.5D

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

depth image which only captures the observed surface points from the view of camera, which is an incomplete 3D shape of the hand. The occupancy grid cannot differentiate voxels before and behind the observed surface.

3.2 TSDF Volumes

In the accurate Truncated Signed Distance Function (TSDF) based volume, each voxel stores the truncated signed distance from the voxel center to the closest surface point. The value of a voxel $\boldsymbol{v}$ is computed as:

$$F(\boldsymbol{v}) = \min\left\{\max\left\{d(\boldsymbol{v}_c)/\mu, -1\right\}, 1\right\}, \quad (3)$$

where $d(\boldsymbol{v}_c)$ is the signed distance from the voxel center $\boldsymbol{v}_c$ to the closest surface point; when the voxel center's depth value is smaller than the depth value of the closest surface point, its sign is positive; otherwise, its sign is negative. μ is the truncated distance, which is set as $3 \times l_{voxel}$ here.

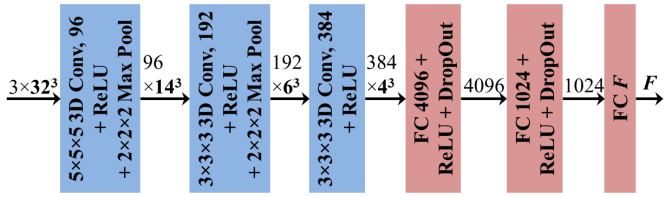
However, it is time-consuming to compute accurate TSDF, as all surface points must be checked to find the closest point for each voxel in the 3D volume. For real-time considerations, the projective TSDF [58], where the closest point is found only on the line of sight in the camera frame, should be used. It can be computed efficiently in parallel on a GPU. Since the projective TSDF is an approximation of the accurate TSDF, some information is inaccurate or lost in the projective TSDF. In this work, we apply the projective Directional TSDF (D-TSDF) [27] to encode more information in the volumetric representation, in which each voxel stores a 3D offset vector $[dx, dy, dz]$ to the closest point instead of a scalar distance.

Figure 2 (the last five rows) shows some examples of accurate TSDF volumes, projective TSDF volumes and projective D-TSDF volumes with different hand poses. As can be seen, in accurate TSDF volumes, the value of TSDF increases when moving from the observed surface to free space and decreases when moving to the occluded space. In projective TSDF volumes, the value of TSDF varies continuously along the line of sight, which keeps positive in front of the observed surface and negative behind the observed surface. The projective D-TSDF volume of z direction is almost the same as the projective TSDF volume, since the z direction is close to the line of sight. Projective D-TSDF volumes of x, y directions can provide more information to compensate the inaccuracy of the projective TSDF volume. Experiments in Section 5.3.2 will show that the projective D-TSDF is computationally efficient and can improve the estimation accuracy.

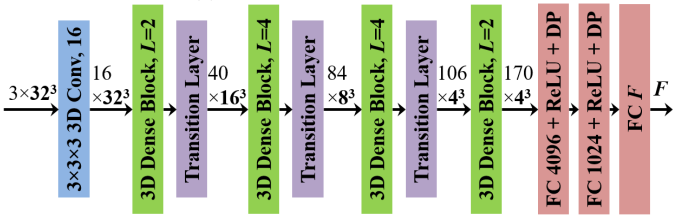
We need to strike a balance between prediction accuracy and computational cost to determine the volume resolution M. If the volume resolution is too large, it will be time-consuming and memory intensive to compute the volumetric representation. If the volume resolution is too small, the volumetric representation cannot give sufficient information for 3D hand pose estimation. In this work, M is chosen as 32, which proves to have good estimation accuracy and computational efficiency for 3D hand pose estimation according to the experiments in Section 5.3.1.

4 3D Convolutional Neural Networks

In this section, we first describe the network architectures and the loss function applied for 3D hand pose regression. To tackle the problems of self-occlusion and large variations in global orientations, we further introduce the methods of 3D hand surface completion and 3D data augmentation. In the end of this section, we visualize and analyze 3D patterns learned by 3D CNNs.



(a) 3D Shallow Plain Network



(b) 3D Deep Dense Network

Fig. 3: Plain and dense network architectures. Both networks take three 32×32×32 projective D-TSDF volumes as input and output F elements. In (a), all the 3D convolutional layers have stride 1 and no padding. In (b), the growth rate of the 3D deep dense network is 32; L in each 3D dense block denotes the number of dense unit; each dense unit consists of a sequence of layers: BN-ReLU-3D Conv (1×1×1)-BN-ReLU-3D Conv (3×3×3); each transition layer reduces the number of feature maps by half using 1×1×1 3D convolution and downsamples the feature map using average pooling. 'BN' denotes the batch normalization layer, 'FC' denotes the fully-connected layer, and 'DP' denotes the dropout layer.

4.1 Network Architectures

Our proposed 3D CNN takes three volumes of the projective D-TSDF as inputs and output a vector containing F elements. As shown in Figure 3, we design two network architectures: the 3D shallow plain network and the 3D deep dense network.

4.1.1 3D Shallow Plain Network

The 3D shallow plain network contains three 3D convolutional layers and three fully connected layers, as shown in Figure 3a. For the three 3D convolutional layers, the kernel sizes are 5^3, 3^3 and 3^3, all with stride 1 and no padding. The first two 3D convolutional layers are followed by 3D max pooling layers with kernel size 2^3, stride 2 and no padding. After 3D feature extraction by 3D convolutional layers, three fully-connected layers are used to map 3D features to a lower dimensional space of 3D hand joint locations. In the first two fully-connected layers, we apply dropout layers with dropout rate 0.5 in order to prevent the neural network from overfitting [59].

4.1.2 3D Deep Dense Network

Deep convolutional networks with shortcut connections have shown powerful learning ability for image recognition [30], [31], since the shortcut connection can alleviate the gradient vanishing problem. We apply the deep dense network architecture [31] in our proposed 3D CNN-based hand pose estimation method. As shown in Figure 3b, we design a 3D deep dense network containing 28 convolutional layers and 3 fully-connected layers. Following the architecture in [31], we apply bottleneck layers in each dense block to reduce the number of model parameters. Experiments in Section 5.3.5 will show that the 3D deep dense network can

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

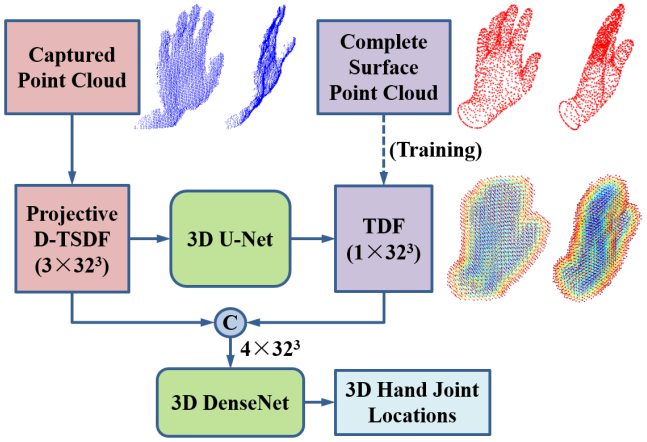


Fig. 4: The framework for hand surface completion and 3D hand pose estimation. 3D U-Net is applied to estimate the TDF volume of complete hand surface points from the projective D-TSDF volumes corresponding to the captured partial hand surface points. The projective D-TSDF volumes concatenated with the estimated TDF volume are fed into the 3D CNN for 3D hand pose estimation. 'C' denotes the concatenation operation.

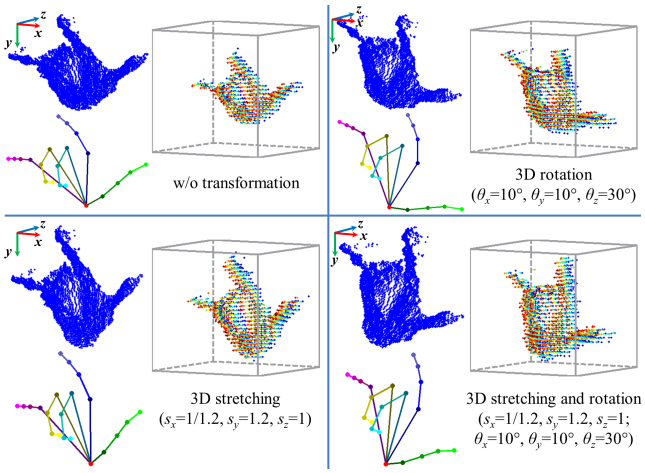


Fig. 5: An example of 3D data augmentation. **Top-left**: original point cloud, ground truth and TSDF volume. **Bottom-left**: point cloud, ground truth and TSDF volume after 3D stretching. **Top-right**: point cloud, ground truth and TSDF volume after 3D rotation. **Bottom-right**: point cloud, ground truth and TSDF volume after 3D stretching and rotation. For illustration purpose, we only draw the projective D-TSDF volume on z direction.

achieve more accurate estimation result compared with that of the 3D shallow plain network.

4.2 Loss Function for 3D Hand Pose Regression

Our method estimates K hand joint locations in 3D space, which represent the 3D hand pose. Let the K objective hand joint locations be $\mathbf{\Phi} = \{\boldsymbol{\phi}_k\}_{k=1}^{K} \in \mathbf{\Lambda}$, here $\mathbf{\Lambda}$ is the $3 \times K$ dimensional hand joint space. We denote a training sample as $(X_n, \mathbf{\Phi}_n)$, where X_n is the depth image, $\mathbf{\Phi}_n$ is corresponding joint locations in the camera's coordinate system, $n = 1, \ldots, N$. The depth image X_n is converted to the volumetric representation V_n as described in Section 3. The center of V_n is located at the center of the AABB generated from the 3D hand points. Thus, the ground truth $\mathbf{\Phi}_n$ should be transformed into coordinates in the volume's coordinate system and normalized between 0 and 1. We denote the transformed and normalized joint locations as $\boldsymbol{Y}_n = \{y_{nk}\}_{k=1}^{K} \in \mathbf{\Lambda}$. The 3D location $y_{nk} \in \mathbb{R}^3$ of the k-th hand joint is transformed and normalized as:

$$y_{nk} = (\phi_{nk} - c_n)/(M \cdot l_{voxel}) + 0.5, \tag{4}$$

where c_n is the center of the 3D volume. The original ground truth location ϕ_{nk} is transformed into the volume's coordinate system by subtracting the center c_n, and is normalized between -0.5 and 0.5 by dividing the volume's edge length l_{voxel} (we assume that all joints are within the 3D volume). To make the coordinates of y_{nk} be between 0 and 1, we add 0.5 in this formula.

Since the degree of freedom (DOF) of 3D hand joints is usually lower than the dimension of hand joint locations that is $3 \times K$, our designed 3D CNN explicitly enforces the hand configuration constraints on the estimation of hand joint locations, thus can alleviate infeasible hand pose estimations. To learn the hand pose priors, we perform PCA on the transformed and normalized joint locations $\{\boldsymbol{Y}_n\}_{n=1}^{N}$ in the training dataset, which is similar to the method in [17]. The coefficients of the principal components are $\boldsymbol{\alpha}_n = \boldsymbol{E}^T \cdot (\boldsymbol{Y}_n - \boldsymbol{u})$, where $\boldsymbol{\alpha}_n$ contains F coefficients, $F < 3 \times K$; $\boldsymbol{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_F]$ are the principal components; $\boldsymbol{u}$ is the mean vector of $\boldsymbol{Y}$.

During the training stage, we minimize the following objective function using the SGD algorithm:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \sum_{n=1}^{N} \|\boldsymbol{\alpha}_n - \mathcal{F}(V_n, \boldsymbol{w})\|^2. \tag{5}$$

During testing, given the input volumetric representation V, the transformed and normalized hand joint locations estimated by the 3D CNN is $\hat{\boldsymbol{Y}} = \boldsymbol{E} \cdot \mathcal{F}(V, \boldsymbol{w}^*) + \boldsymbol{u}$.

4.3 Hand Surface Completion

One of challenges in 3D hand pose estimation is that the input depth image only captures partial hand surface and suffers from the self-occlusion problem. To tackle this problem, we leverage the complete hand surface as intermediate supervision for learning 3D hand pose from depth images. More specifically, we estimate the complete hand surface using a data-driven approach and take advantage of the estimated complete hand surface for 3D hand pose estimation. As shown in Figure 4, we apply the 3D U-Net [61] architecture for estimating the complete hand surface from the captured partial hand surface. Similar to [62], we use the unsigned truncated distance function (TDF) as the network output, since it is not necessary to differentiate known and unknown space for the distance function of the complete hand surface. The ground truth of the complete hand surface is generated by using the model fitting method [7] with a 3D hand model. When training the 3D U-Net, we minimize the smooth L_1 loss defined in [63] using the ADAM optimizer [64]. After generating the TDF volume of the complete hand surface, we concatenate it with the original projective D-TSDF volumes and feed them into a 3D DenseNet for 3D hand pose estimation. The 3D U-Net and the 3D DenseNet are pre-trained separately, and are fine tuned in an end-to-end manner. Experiments in Section 5.3.6 will show that, with the intermediate supervision of hand surface completion, the accuracy of 3D hand pose estimation can be further improved.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

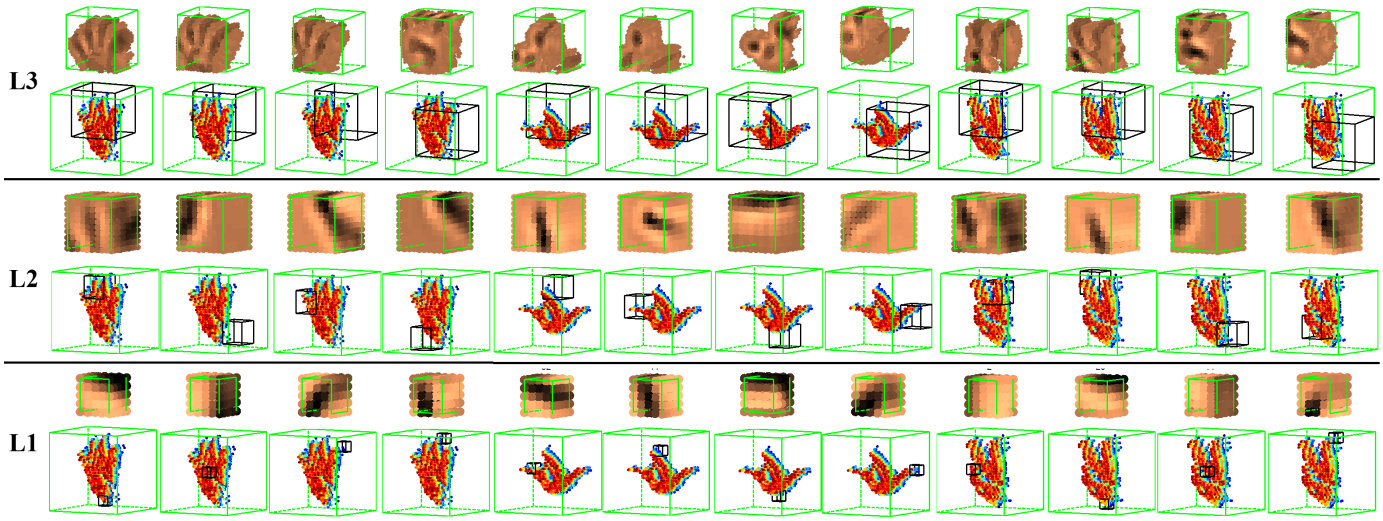


Fig. 6: Visualization of patterns learned in a fully trained 3D CNN model. For each layer, we show reconstructed patterns in the 1st row, and corresponding receptive fields indicated by black boxes in the 2nd row. These patterns are reconstructed by using the guided backpropagation method proposed in [60]. For patterns of L3, we only draw voxels of which absolute values are larger than a threshold. Voxels with large values are shown in bright color, and voxels with small values are shown in dark color. Neurons in the 1st convolutional layer (L1) can capture local structures, such as corners and edges; neurons in the 2nd convolutional layer (L2) can capture structures of hand part, such as fingers; neurons in the 3rd convolutional layer (L3) can capture global structures of hand.

4.4 3D Data Augmentation

Another challenge of 3D hand pose estimation is that hand pose has large variations in global orientations and hand sizes. In order to make the 3D CNN model robust to different orientations and sizes and improve its generalization ability, we perform 3D data augmentation on the training data for both pose regression and surface completion networks. Different from existing 2D image data augmentation, our method directly rotate and stretch the hand points in 3D space.

We first stretch the point cloud along x, y, z axes of the camera's coordinate system with stretch factors s_x, s_y and s_z, respectively. Then, the point cloud is rotated around x, y, z axes of the camera's coordinate system with rotation angles θ_x, θ_y and θ_z, respectively. For a 3D point $\boldsymbol{p}$, after stretching and rotating, the point $\boldsymbol{p}$ is transformed into $\boldsymbol{p}'$:

$$\begin{aligned} \boldsymbol{p}' &= \mathcal{R} \cdot \mathcal{S} \cdot \boldsymbol{p} \\ \mathcal{R} &= \mathcal{R}_x(\theta_x) \cdot \mathcal{R}_y(\theta_y) \cdot \mathcal{R}_z(\theta_z) \\ \mathcal{S} &= \mathrm{Diag}(s_x, s_y, s_z), \end{aligned} \tag{6}$$

where $\mathcal{R}_x$, $\mathcal{R}_y$ and $\mathcal{R}_z$ are 3×3 rotation matrices around x, y, z axes, respectively; $\mathrm{Diag}(s_x, s_y, s_z)$ is a 3×3 diagonal matrix whose diagonal entries starting in the upper left corner are s_x, s_y and s_z. Figure 5 shows an example of 3D data augmentation. 3D stretching and rotation are performed on the hand point cloud and corresponding ground truth joint locations. TSDF volumes are then generated from the transformed point cloud.

In this work, an augmented training set is generated by randomly stretching and rotating original training samples. The stretch factors s_x and s_y are chosen log-uniformly at random from the interval $[1/1.5, 1.5]$. Since it is the relative size rather than the absolute size that affects the TSDF volume, we can set the stretch factor s_z as 1. In the camera's coordinate system, the point cloud generated from the depth image is not only in-plane rotated around z axis, but also out-of-plane rotated around x, y axes, as shown in Figure 5. The out-of-plane rotated point cloud is different from the real point cloud due to the incompleteness of the point cloud. We assume that with small out-of-plane rotation angles θ_x and θ_y, the TSDF volume generated from the out-of-plane rotated point cloud can be approximately the same as the TSDF volume generated from the real point cloud when the camera's viewpoint is at the rotated angle, and we choose θ_x and θ_y uniformly at random from the interval $[-30°, 30°]$. The in-plane rotation angle θ_z is chosen uniformly at random from the interval $[-180°, 180°]$. During the training stage, both the original training set and the augmented training set are used for training. Experiments in Section 5.3.3 will show the effectiveness of the 3D data augmentation.

4.5 Visualizing 3D CNN

In order to analyze 3D features extracted by the 3D CNN, we visualize the input patterns that produce given activations in the 3D feature volumes by adopting the guided backpropagation method proposed in [60] which is a modification of the deconvolution based visualizing approach proposed in [65].

In Figure 6, we visualize some 3D patterns learned in a 3D shallow plain network which is fully trained on the MSRA hand pose dataset [26]. We take three input volumes with different hand poses as examples in Figure 6. In order to reconstruct 3D patterns, 3D feature volumes generated from 3D convolutional layers are projected down to the input voxel space by using the guided backpropagation method. For each convolutional layer of each hand pose, we choose four feature volumes as examples in Figure 6. To show which parts of the input volume cause high activations in the 3D feature volume, we crop the reconstructed 3D pattern volume inside the receptive field corresponding to the highest activation in the 3D feature volume. We also show the relative location of the receptive field in the input 3D volume. The receptive field sizes of the 1st convolutional layer (L1), the 2nd

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

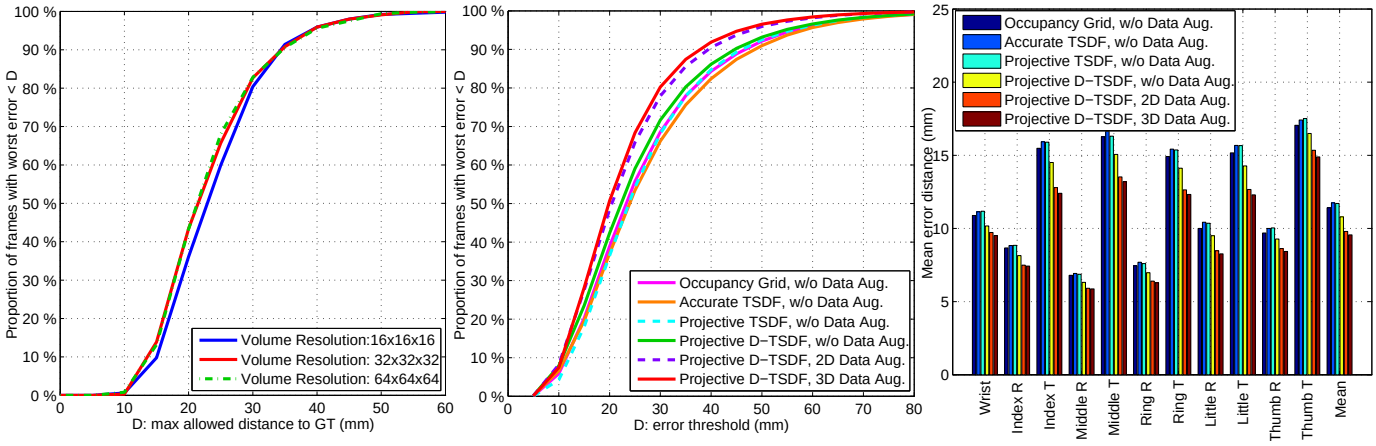


Fig. 7: Self-comparison of different volume resolutions and different volume types with/without data augmentation on MSRA dataset [26]. **Left**: the impact of different volume resolutions on the proportion of good frames. **Middle**: the impact of different volume types and data augmentation on the proportion of good frames. **Right**: the impact of different volume types and data augmentation on the per-joint mean error distance (R:root, T:tip).

convolutional layer (L2) and the 3rd convolutional layer (L3) in the 3D shallow plain network are 5, 10 and 20, receptively.

As can be seen in Figure 6, from low layer (L1) to high layer (L3), neurons can capture patterns from local to global. Neurons in L1 capture low-level local geometry structures, such as the corners (e.g., L1 column 3, 8, 12) and edges (e.g., L1 column 1, 5, 7). Neurons in L2 capture mid-level shape structures of the hand, such as the hand finger tips (e.g., L2 column 5, 6, 8, 10) and palm edges (e.g., L2 column 2, 4, 7, 11). Neurons in L3 capture high-level global structures of the hand. It is interesting to note that, in patterns of L3, the contrast in regions of hand joints is obvious, which indicates that the learned high layer feature volumes focus on regions of hand joints. The hierarchical nature of 3D features is consistent with that in the 2D CNNs as observed in [65].

5 Experiments

5.1 Datasets and Evaluation Metrics

We evaluate our proposed method on three public hand pose datasets: MSRA hand pose dataset [26], NYU hand pose dataset [7] and ICVL hand pose dataset [8].

The MSRA dataset [26] contains nine subjects' hand depth images captured by the Intels Creative Interactive Gesture Camera. Each subject performs 17 hand gestures, and each hand gesture contains about 500 frames. There are more than 76K frames in this dataset. In the following experiments on this dataset, we train on eight subjects and test on the remaining subject. This is repeated nine times for all subjects. The ground truth of each frame contains $K = 21$ hand joints' 3D locations including four joints for each finger and one joint for the wrist.

The NYU dataset [7] contains more than 72K frames for training and 8K frames for testing, which are captured by the $\mathrm{PrimeSense}^{\mathrm{TM}}$ 3D sensor. The ground truth of each frame contains 36 hand joints' 3D locations. Following previous work in [7], [18], we estimate a subset of $K = 14$ hand joints' 3D locations including three joints for thumb, two joints for index, middle, ring and little fingers, one joint for palm center and two joints for the wrist. Since the NYU dataset provides the original depth image containing human body and background, we apply a simplified hourglass network [66] to detect 2D hand joint locations and use the corresponding depth information to segment the hand.

The ICVL dataset [8] contains 12 training sequences having 22K frames and two testing sequences having 1.6K frames captured by the Intel's Creative Interactive Gesture Camera. This dataset additionally provides a training set in which the original training samples are in-plane rotated 14 times. However, in our experiments, instead of using the additional training set, we apply the 3D data augmentation by randomly rotating and stretching the original training samples eight times. Thus, the augmented training set in our experiments contains 176K frames. The ground truth of each frame contains $K = 16$ hand joints' 3D locations including three joints for each finger and one joint for the palm center.

Three metrics are employed to evaluate the hand pose estimation performance in our experiments. The first metric is the per-joint mean error distance over all test frames. The second metric is the proportion of good frames in which the worst joint error is below a threshold [67], which is a strict measure. The third metric is the proportion of joints within an error threshold [10].

5.2 Implementation Details

We train and evaluate our proposed 3D CNN models for 3D hand pose estimation on a computer with two Intel Core i7 5930K 3.50GHz CPUs, 64GB of RAM and an Nvidia GeForce GTX 1080 GPU having 8GB of GPU memory. The 3D CNN models are implemented within the PyTorch framework. For network training parameters, we choose the batch size as 16, the momentum as 0.9 and the weight decay as 0.0005. The learning rate is set as 0.01 for the 3D regression network, and is divided by 10 after 50 epochs. For the 3D U-Net, the learning rate is set as 0.001. The training is stopped after 60 epochs to prevent overfitting. We apply the same hyper-parameters for all experiments on all datasets. All the weights of 3D convolutional layers in 3D CNNs are randomly initialized using the method proposed in [68].

5.3 Self-comparison

5.3.1 *Choice of Volume Resolution*

To evaluate the impact of different volume resolutions, we experiment with projective D-TSDF volumes with different resolution

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

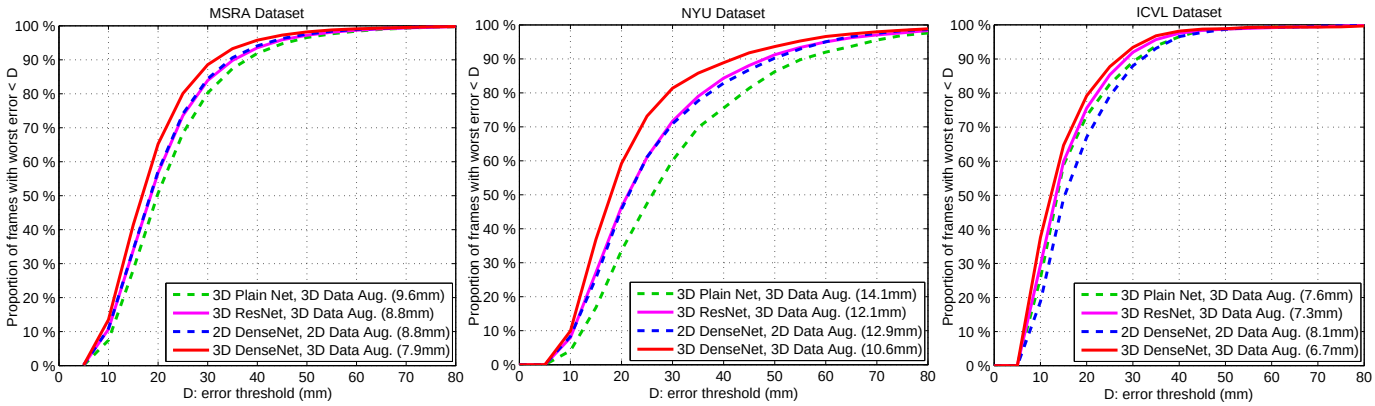


Fig. 8: Self-comparison of 2D/3D CNNs with different network architectures for 3D hand pose estimation on MSRA [26], NYU [7] and ICVL [8] hand pose datasets. The mean error distances over all joints of different methods are shown in the legends.

values: 16, 32 and 64 using the 3D plain networks with direct regression. Since training the network with 64×64×64 volume resolution is very time consuming, we only train and test these three networks with different volume resolutions on a small subset of the MSRA dataset without data augmentation in this experiment. As shown in Figure 7 (left), the estimation accuracy of 16×16×16 resolution is slightly inferior to those with 32×32×32 and 64×64×64 resolutions. The estimation accuracy of the latter two resolutions is almost the same. However, computing TSDF volume with 64×64×64 resolution is more time consuming and memory intensive. Thus, the volume resolution 32×32×32 is most suitable for hand pose estimation, and we use this volume resolution in the following experiments. This experiment also shows that our method is robust to relatively low volume resolution, since the estimation accuracy does not decrease a lot when the resolution value is 16.

5.3.2 Choice of Volume Types

We evaluate the impact of different volume types on the estimation accuracy on MSRA dataset using the 3D plain network without data augmentation. As can be seen in Figure 7 (middle and right), among occupancy grid, accurate TSDF, projective TSDF and projective D-TSDF, the projective D-TSDF performs best. It is worth noting that the performance of occupancy grid is comparable with those of accurate TSDF and projective TSDF, which indicates that the 3D CNNs can learn effective 3D features from the occupancy grid, although the occupancy grid cannot differentiate voxels before and behind the observed surface. But the projective D-TSDF, which encodes more information on three directions, outperforms the occupancy grid. For the real-time performance, the average computation time to generate occupancy grid, accurate TSDF, projective TSDF and projective D-TSDF on the same GPU are 1.4ms, 30.2ms, 1.9ms and 2.9ms, respectively. Thus, considering both the estimation accuracy and the real-time performance, the projective D-TSDF is overall best. In the following experiments, we apply the projective D-TSDF with 32×32×32 volume resolution as the network input and apply 3D data augmentation for training.

5.3.3 Evaluation of Data Augmentation

We compare the method without using data augmentation and the methods using 2D/3D data augmentation. For 2D data augmentation, we randomly rotate and stretch the 3D hand point cloud in the 2D image plane. As shown in Figure 7 (middle and right), when using the same input volume, the method using 2D data augmentation outperforms the method without using data augmentation. When using 3D data augmentation in the training stage, the estimation accuracy is further improved. It is worth noting that, although the 3D rotated point cloud is not exactly the same as the real point cloud, the network trained with 3D data augmentation can still achieve better performance than the network trained with 2D data augmentation, which indicates that the network can benefit from the 3D augmented data.

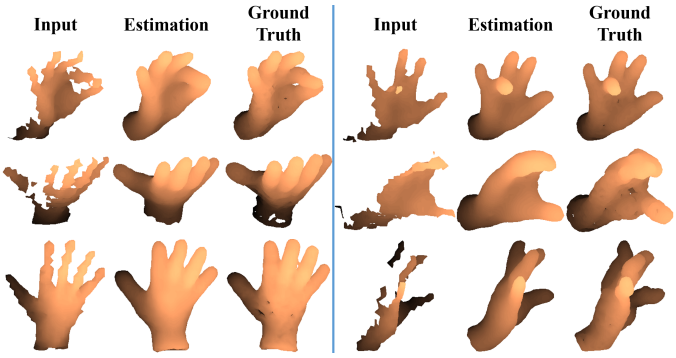


Fig. 9: Examples of hand surface completion with our method on NYU hand pose dataset [7]. The hand surfaces are extracted from the 32^3 volumes of distance function.

5.3.4 2D CNNs Versus 3D CNNs

We compare the performance of the 3D CNN-based methods and the 2D CNN-based methods in this experiment. For 2D CNN-based methods, we segment the hand from the depth image and resize it to a 96×96 image while keeping the aspect ratio. The outputs of 2D CNNs are 2D image locations and depth values of hand joints, which are converted to 3D locations using camera parameters. For fair comparison, we experiment with the 2D deep dense network and 3D deep dense network having similar network architecture and comparable number of parameters. The 2D deep dense network has 5 dense blocks, 29 convolutional layers and 3 full-connected layers with 65M parameters; and the 3D deep dense network has 4 dense blocks, 28 convolutional layers and 3 full-connected layers with 50M parameters. In addition, we perform

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

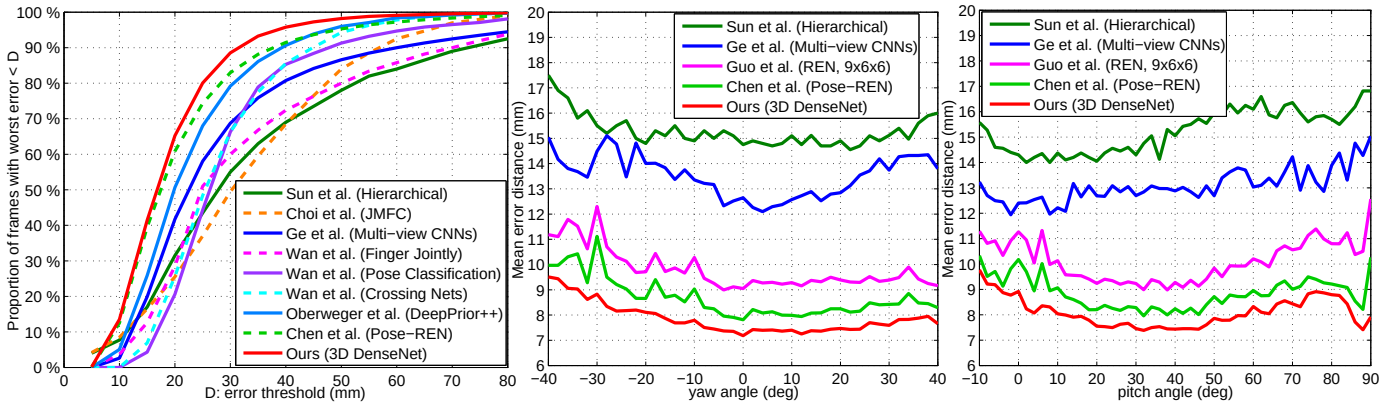


Fig. 10: Comparison with state-of-the-art methods [19], [23], [26], [45], [69], [70], [71], [72] on MSRA dataset [26]. **Left**: the proportion of good frames over different error thresholds. **Middle & right**: the mean error distance over different yaw and pitch viewpoint angles with respect to the camera frame.

TABLE 1: Impact of Hand Surface Completion on 3D Hand Pose Estimation using 3D ResNet and 3D DenseNet. The Average Estimation Errors on NYU Dataset [7] are Listed in This Table.

	3D ResNet	3D DenseNet
w/o Surface Completion	12.1mm	10.6mm
/w Surface Completion	**11.7mm**	**10.2mm**
/w Ground Truth of Complete Surface	11.0mm	8.3mm

2D data augmentation when training the 2D deep dense network, and the number of training samples is the same as that used in training the 3D deep dense network. As shown in Figure 8, the 3D deep dense network consistently performs better than the 2D deep dense network on all the three hand pose datasets, which indicates that the 3D CNNs can better utilize the depth information and provide more accurate estimation.

5.3.5 Shallow Network Versus Deep Network

We compare the estimation accuracy of the 3D shallow plain network and the 3D deep networks. For 3D deep networks, apart from the deep dense network, we also experiment with the deep residual network [30] which has 4 residual blocks, 9 convolutional layers and 3 full-connected layers. As can be seen in Figure 8, the performance of 3D deep dense network is better than that of 3D deep residual network, and the performance of 3D deep residual network is better than that of 3D shallow plain network on all the three hand pose datasets. Although the 3D deep networks have more convolutional layers, the convolutional layers in the 3D shallow plain network have more output channels. Thus, the forward propagation time of these three networks is comparable, which is 3.5ms for 3D shallow plain network, 3.4ms for 3D deep residual network, and 4.5ms for 3D deep dense network. We use the 3D deep dense network in the following experiments.

5.3.6 Evaluation of Hand Surface Completion

We evaluate the impact of hand surface completion on the accuracy of 3D hand pose estimation. Some examples of the estimated complete hand surface as the intermediate results of our method are shown in Figure 9. We extract the hand surface from the 32^3 volume of distance function using Matlab's *isosurface* function [62]. As can be seen from Figure 9, our method is able to generate complete hand surface from the input partial hand surface. Compared with the ground truth, our estimation is more blurry and loses some details of the complete hand surface. The average L_1 loss of estimated TDF volume against the ground truth TDF volume of the complete surface is 0.112 in voxel space of which the truncation distance is 2.5. The average errors of 3D hand pose estimation using hand surface completion and without using hand surface completion are presented in Table 1. As can be seen, no matter using the 3D deep residual network or the 3D deep dense network, the surface completion step can further improve the estimation accuracy. In addition, in order to evaluate the importance of the complete hand surface on hand pose estimation, we use the ground truth of the complete hand surface combined with the original partial hand surface as network input to estimate the 3D hand pose. As presented in the last row of Table 1, the networks can achieve much smaller estimation errors when using the ground truth of the complete hand surface, which shows the potential of our method to achieve smaller pose estimation error if the complete hand surface could be estimated more accurately.

5.4 Comparison with State-of-the-art

5.4.1 Comparison on MSRA Dataset

On MSRA dataset, we compare our 3D CNN-based hand pose estimation method with seven state-of-the-art methods: the hierarchical regression method [26], the joint matrix factorization and completion (JMFC) method [69], the multi-view CNN-based method [19], the local surface normals (LSN) based method [45], the crossing nets using deep generative models [23], the improved 2D CNN with hand pose prior and refinement (DeepPrior++) [70], the region ensemble network (REN) [72] and the pose guided structured REN (Pose-REN) [71]. Note that since the hierarchical regression method [26] has been shown superior to the methods in [1], [6], we indirectly compare our method with [1], [6].

As shown in Figure 10, our 3D CNN-based method outperforms state-of-the-art methods on the MSRA dataset. The proportion of good frames over different error thresholds is shown in Figure 10 (left). Our method achieves the best performance when the error threshold is larger than 10mm. For example, when the error threshold is 30mm, the proportion of good frames of our method is about 6%, 10%, 20%, 22%, 28%, 33% and 39% higher than those of the methods in [71], [70], [19], [23], [45]

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

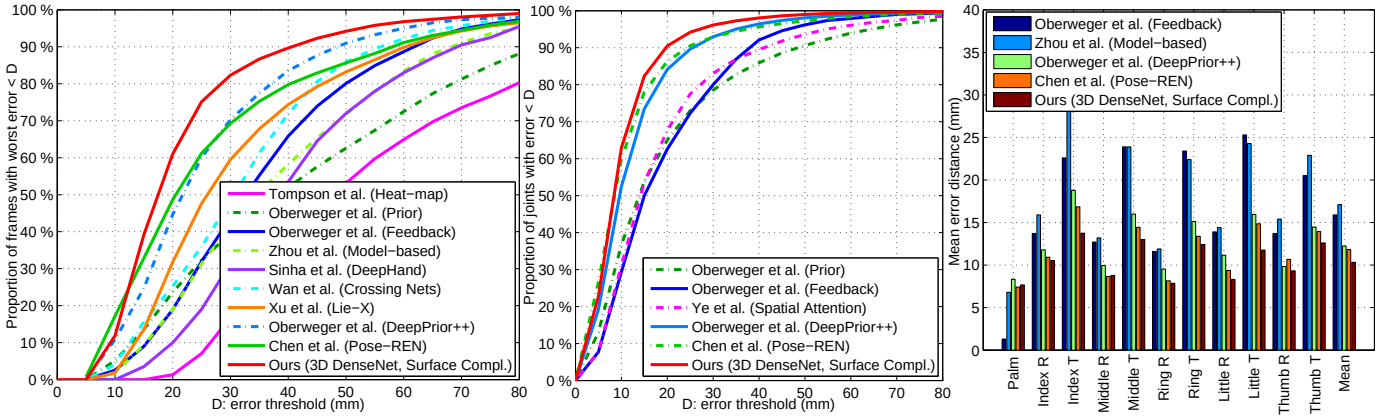


Fig. 11: Comparison with state-of-the-art methods [7], [17], [18], [20], [21], [22], [23], [70], [71], [73] on NYU dataset [7]. **Left**: the proportion of good frames over different error thresholds. **Middle**: the proportion of joints within different error thresholds. **Right**: the per-joint mean error distance (R:root, T:tip).

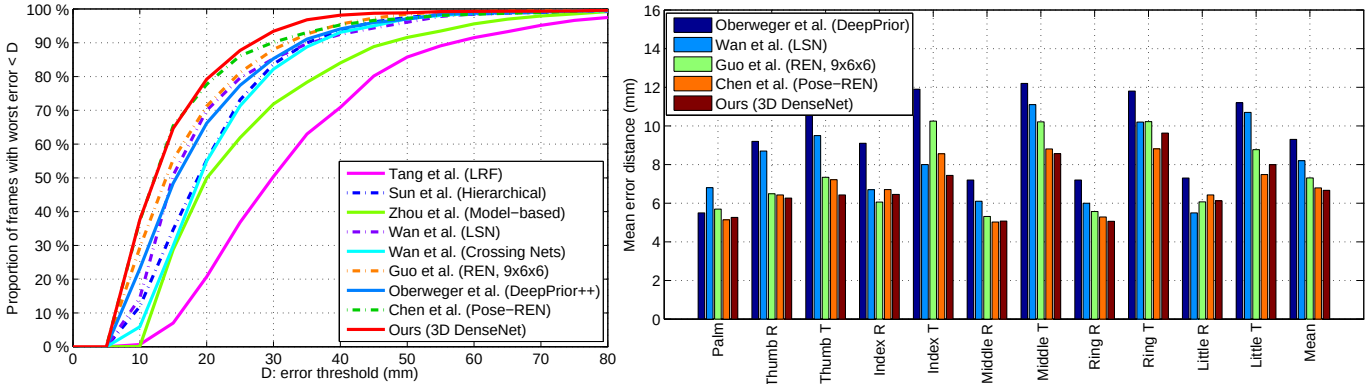


Fig. 12: Comparison with state-of-the-art methods [8], [22], [23], [26], [45], [70], [71], [72] on ICVL dataset [8]. **Left**: the proportion of good frames over different error thresholds. **Right**: the per-joint mean error distance (R:root, T:tip).

(finger jointly regression), [26] and [69], respectively. When the error threshold is 5mm, the proportion of good frames of our method is slightly worse than those of the methods in [26] and [69]. This may be caused by the relatively large edge length of the voxel, which is 5.5mm on average when the volume resolution is $32\times32\times32$. In Figure 10 (middle and right), we compare the mean error distance over different yaw and pitch viewpoint angles with the methods in [19], [26], [71], [72]. As can be seen, the mean errors over different viewpoint angles of our method are about 7mm, 4.5mm, 1.8mm and 0.8mm smaller than those of the methods in [26], [19], [72] and [71], respectively. Our method exhibits less variance to the pitch viewpoint angle changes with a smaller standard deviation (0.58mm) than those of the methods in [26] (0.79mm), [19] (0.64mm), [72] (0.82mm) and [71] (0.63mm).

5.4.2 Comparison on NYU Dataset

On NYU hand pose dataset, we first compare our 3D CNN-based hand pose estimation method with nine state-of-the-art methods: the 2D CNN-based heat-map regression method [7], the 2D CNN-based direct regression method with a pose prior (DeepPrior) [17], the 2D CNN-based method using a feedback loop [18], the 2D CNN-based hand model parameter regression method [22], the deep feature based matrix completion method (DeepHand) [20], the crossing nets using deep generative models [23], the Lie-X method applying the Lie group theory [73], the DeepPrior++ method [70], and the Pose-REN method [71]. For the 2D CNN-based heat-map regression method [7], we estimate the 2D joint locations from heat-maps and convert them to 3D locations using corresponding depth values. As shown in Figure 11 (left), our method outperforms these nine methods over all the error thresholds. For example, the proportion of good frames of our method is about 10% more than that of the Pose-REN method [71] when the error threshold is between 20mm and 50mm.

In order to make a fair comparison with the spatial attention network based hierarchical hybrid method in [21], we evaluate the proportion of joints within different error thresholds on the subset of 11 hand joints following the experiment in [21] (removing palm joints except the root joint of thumb). As shown in Figure 11 (middle), our method is superior to the methods in [17], [18], [21], [70], [71] over all the error thresholds. For example, the proportion of joints within error threshold 20mm of our method is about 20% more than that of the method in [21].

We also compare the mean error distance of our method with those of the methods in [18], [22], [70], [71]. As shown in Figure 11 (right), our method achieves the smallest mean error distance on most joints, and the mean error distance over all joints of our method is about 5.5mm, 6.5mm, 2mm and 1.5mm smaller than those of methods in [18], [22], [70] and [71], receptively.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

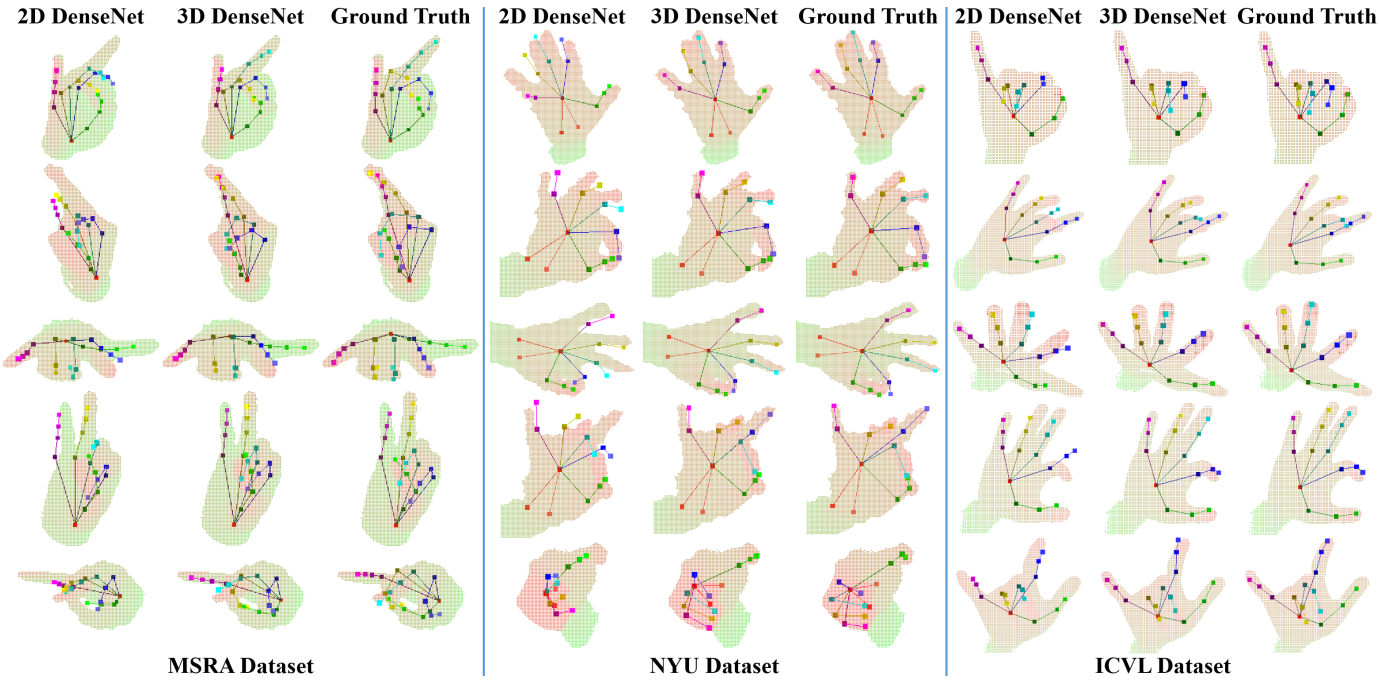


Fig. 13: Qualitative results for MSRA, NYU and ICVL datasets. We compare our method based on the 3D deep dense network (in the 2nd column for each dataset) with the method based on the 2D deep dense network (in the 1st column for each dataset). The ground truth hand joint locations are presented in the last column for each dataset. We show hand joint locations and bones with the point cloud. Different hand joints and bones are visualized using different colors. This figure is best viewed in color.

5.4.3 Comparison on ICVL Dataset

On ICVL hand pose dataset, we compare our 3D CNN-based hand pose estimation method with eight state-of-the-art methods: the latent regression forest (LRF) [8], the hierarchical regression method [26], the hand model parameter based method [22], the LSN method [45], the crossing nets using deep generative models [23], the REN method [24], the DeepPrior++ method [70], and the Pose-REN method [71]. As shown in Figure 12 (left), our method outperforms these eight methods over most of the error thresholds on this dataset.

We also compare the mean error distance of our method with those of the methods in [17], [45], [71], [72]. As shown in Figure 12 (right), our method achieves the smallest mean error distance on most joints. The mean error distance over all joints of our method is 6.7mm, while those of methods in [17], [45], [72], and [71] are 9.3mm, 8.2mm, 7.3mm and 6.8mm, receptively.

5.5 Cross-dataset Experiment

To evaluate the generalization ability of our 3D CNN-based hand pose estimation method, we perform a cross-dataset experiment, in which the 3D CNN model is trained on the whole MSRA dataset [26] and is evaluated on the whole dataset released in [34]. According to the evaluation metric in [34], we calculate the mean error distances for the wrist and the five fingertips. As shown in Table 2, we compare our 3D CNN-based method with model based tracking methods reported in [34], which are FORTH [5], PSO [34], ICP [74], ICP-PSO [34] and ICP-PSO* (ICP-PSO with finger-based initialization) [34], as well as the multi-view CNN-based method [19]. As can be seen, our method achieves the second best result on three subjects and on the average error over all subjects.

TABLE 2: Average Estimation Errors (in *mm*) of 6 Subjects for 7 Methods Tested on the Dataset Released in [34].

Subject	1	2	3	4	5	6	Avg
FORTH	35.4	19.8	27.3	26.3	16.6	46.2	28.6
PSO	29.3	14.8	40.2	17.3	16.2	24.3	23.6
ICP	29.9	20.7	30.8	23.9	18.5	32.8	26.1
ICP-PSO	**10.1**	24.1	13.0	12.8	11.9	20.0	15.3
MVCNN	30.1	19.7	24.3	19.9	21.8	20.7	22.8
3D DenseNet	14.2	**11.4**	**11.4**	**10.8**	**10.9**	**11.6**	**11.7**
ICP-PSO*	8.6	7.4	9.8	10.4	7.8	11.7	9.2

It is worth noting that the model-based methods in [34] require a carefully calibrated hand model for each subject. However, our method does not use the calibrated hand model and thus is more flexible for different subjects. In such situation, our method still outperforms the FORTH, PSO, ICP and ICP-PSO methods, as shown in Table 2. Our method performs a little bit worse than the ICP-PSO* method which uses the ground truth to initialize the first frame. But our method does not use any ground truth of the testing data and is performed on cross-dataset which is more challenging. Thus, the overall second best result achieved by our method in this cross-dataset experiment indicates that our 3D CNN-based method has good generalization ability.

5.6 Qualitative Results

Some qualitative results for MSRA, NYU and ICVL datasets are shown in Figure 13. We compare our 3D deep dense network with the baseline method of 2D deep dense network as described in Section 5.3.4. As can be seen in Figure 13, when using the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TPAMI.2018.2827052, IEEE Transactions on Pattern Analysis and Machine Intelligence

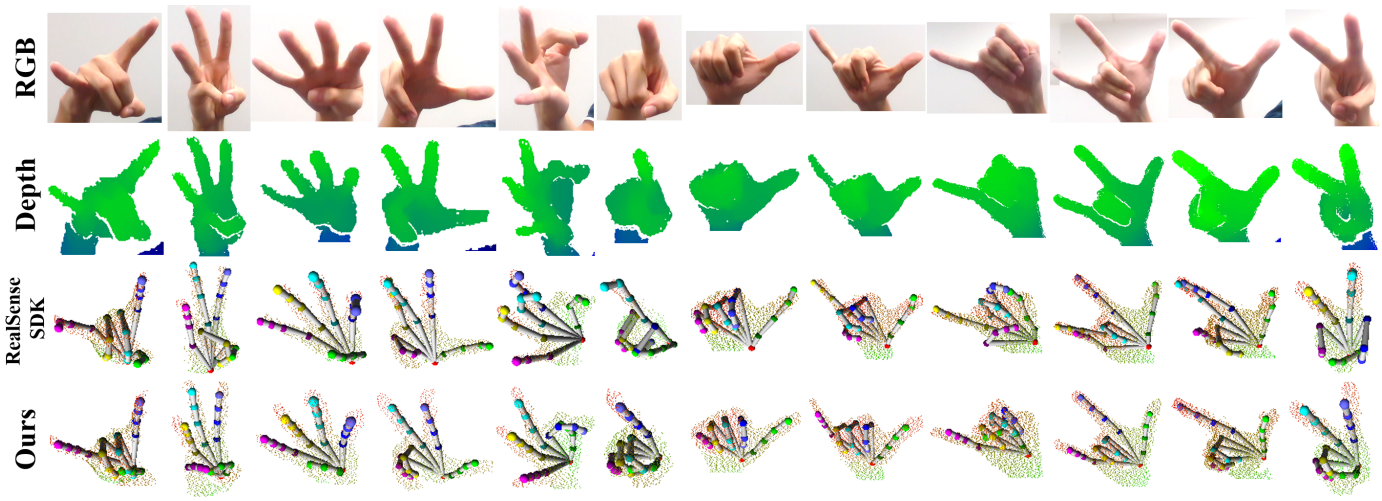


Fig. 14: Qualitative results for testing in real scenarios using Intel RealSense SR300 Depth Camera. For each hand pose, the 1st row shows the RGB image of the hand; the 2nd row shows the depth image of the hand; the 3rd row shows the hand pose estimated by the Intel RealSense SDK [32]; the 4th row shows the hand pose estimated by our 3D CNN-based method. We show hand joint locations and bones with the point cloud. Joints of thumb, index, middle, ring and little fingers are visualized in green, blue, cyan, yellow and pink, receptively. This figure is best viewed in color.

same residual network architecture, our 3D CNN-based method can better utilize the depth information and provide more accurate estimation than the 2D CNN-based method.

We also conduct qualitative comparison with the Intel RealSense SDK [32]. We train the 3D CNN model on the whole MSRA dataset and apply this pre-trained 3D CNN model to perform real-time hand pose estimation with the Intel RealSense SR300 Depth Camera in real scenarios. Qualitative results of our 3D CNN-based method and the Intel RealSense SDK [32] are shown in Figure 14. As can be seen, the Intel RealSense SDK does not accommodate complex hand poses as accurately as our method. For example, in the 4th column of Figure 14, when the little finger is occluded by the ring finger, the Intel RealSense SDK makes wrong estimation of ring finger joints' locations; in the 6th column of Figure 14, the Intel RealSense SDK confuses the index finger with the middle finger; in the 2nd, 8th and 9th columns of Figure 14, the Intel RealSense SDK confuses the ring finger with the little finger. By contrast, benefiting from the 3D CNN which can better exploit the 3D information, our method is able to correctly estimate the hand poses in these cases. More comparisons are presented in our demo video, available online.[1]

5.7 Runtime and Model Size

The runtime of our 3D CNN-based hand pose estimation method is 7.9ms on average on the computer described in Section 5.2. The process of generating the projective D-TSDF volumes with $32\times32\times32$ volume resolution on GPU takes 2.9ms on average. The process of 3D deep dense network forward propagation running on GPU takes 4.5ms on average. The process of reconstructing 3D coordinates of hand joints in the 3D volume from PCA coefficients output by the 3D CNN and transforming them to 3D locations in the camera's coordinate system takes 0.5ms on average on CPU. Thus, our method is capable of running in real-time at over 126 frames per second (fps). When adopting the hand surface completion step in our method, it takes 7.6ms on average for the 3D U-Net and the 3D deep dense network forward propagation. Thus, the runtime of the method using hand surface completion is 11.0ms, and the frame rate is 91 fps.

For model size, our 3D deep dense network model takes about 192MB, the 3D U-Net for hand surface completion takes about 200MB, while the multi-view CNNs in [19] take about 1.2GB. The network parameters are stored in 32 bit float.

1. https://youtu.be/xdMebIYt2g8

6 Conclusion

We present a novel 3D CNN-based hand pose estimation method in this paper. By adopting the projective D-TSDF, we encode the hand depth image as a 3D volumetric representation which is then fed into the 3D CNN. We show that the 3D CNN mapping the 3D volumes to 3D joint locations in a single pass is easy to be trained in an end-to-end manner. The 3D deep dense network can further improve the learning ability for 3D hand pose estimation. To make the 3D CNN robust to various hand sizes and global orientations, we perform 3D data augmentation on the training data. To tackle the self-occlusion problem, we leverage the complete hand surface as intermediate supervision for learning 3D hand pose. Experimental results indicate that our proposed 3D CNN-based approach achieves state-of-the-art performance for 3D hand pose estimation on three challenging datasets and is able to run in real-time with good generalization ability.

Acknowledgments

This research is supported by the BeingTogether Centre, a collaboration between Nanyang Technological University and University of North Carolina at Chapel Hill. The BeingTogether Centre is supported by the National Research Foundation, Prime Ministers Office, Singapore under its International Research Centres in Singapore Funding Initiative. This work is also supported in part by Singapore Ministry of Education Academic Research Fund Tier 2 MOE2015-T2-2-114.

REFERENCES

[1] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from a single depth image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2011, pp. 1297–1304.

[2] M. Ye, Y. Shen, C. Du, Z. Pan, and R. Yang, "Real-time simultaneous pose and shape estimation for articulated objects using a single depth camera." *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 8, pp. 1517–1532, 2016.

[3] H. Y. Jung, Y. Suh, G. Moon, and K. M. Lee, "A sequential approach to 3d human pose estimation: Separation of localization and identification of body joints," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 747–761.

[4] Y. Wu and T. S. Huang, "Hand modeling, analysis and recognition," *IEEE Signal Processing Magazine*, vol. 18, no. 3, pp. 51–60, 2001.

[5] I. Oikonomidis, N. Kyriazis, and A. Argyros, "Efficient model-based 3D tracking of hand articulations using Kinect," in *Proc. British Mach. Vis. Conf.*, 2011, pp. 101.1–101.11.

[6] C. Xu and L. Cheng, "Efficient hand pose estimation from a single depth image," in *Proc. Int. Conf. Comput. Vis.*, 2013, pp. 3456 – 3462.

[7] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Trans. Graph.*, vol. 33, no. 5, pp. 169:1–169:10, 2014.

[8] D. Tang, H. J. Chang, A. Tejani, and T. K. Kim, "Latent regression forest: Structured estimation of 3D articulated hand posture," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 3786–3793.

[9] H. Liang, J. Yuan, and D. Thalmann, "Resolving ambiguous hand pose predictions by exploiting part correlations," *IEEE Trans. on Circuits and Systems for Video Technol.*, vol. 25, no. 7, pp. 1125–1139, Jul. 2015.

[10] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, and S. Izadi, "Accurate, robust, and flexible real-time hand tracking." Proc. 33rd Annual ACM Conf. Human Factors in Computing Systems, April 2015, pp. 3633–3642.

[11] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, T. Sharp, E. Soto, D. Sweeney, J. Valentin, B. Luff, A. Topalian, E. Wood, S. Khamis, P. Kohli, S. Izadi, R. Banks, A. Fitzgibbon, and J. Shotton, "Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences," *ACM Trans. Graph.*, vol. 35, no. 4, pp. 143:1–143:12, 2016.

[12] M. Oberweger, G. Riegler, P. Wohlhart, and V. Lepetit, "Efficiently creating 3D training data for fine hand pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3593–3601.

[13] C. Keskin, F. Kra, Y. Kara, and L. Akarun, "Hand pose estimation and hand shape classification using multi-layered randomized decision forests," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 852–863.

[14] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, "Robust part-based hand gesture recognition using kinect sensor," *IEEE Trans. Multimedia*, vol. 15, pp. 1110–1120, 2013.

[15] E. Ohn-Bar and M. M. Trivedi, "Hand gesture recognition in real time for automotive interfaces: A multimodal vision-based approach and evaluations," *IEEE Trans. Intell. Transportation Systems*, vol. 15, no. 6, pp. 2368–2377, Dec 2014.

[16] P. Molchanov, X. Yang, S. Gupta, K. Kim, S. Tyree, and J. Kautz, "Online detection and classification of dynamic hand gestures with recurrent 3D convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 4207–4215.

[17] M. Oberweger, P. Wohlhart, and V. Lepetit, "Hands deep in deep learning for hand pose estimation," in *Proc. Comput. Vis. Winter Workshop*, 2015, pp. 21–30.

[18] ——, "Training a feedback loop for hand pose estimation," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 3316–3324.

[19] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3593–3601.

[20] A. Sinha, C. Choi, and K. Ramani, "Deephand: Robust hand pose estimation by completing a matrix with deep features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 4150–4158.

[21] Q. Ye, S. Yuan, and T.-K. Kim, "Spatial attention deep net with partial PSO for hierarchical hybrid hand pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 346–361.

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

[23] C. Wan, T. Probst, L. Van Gool, and A. Yao, "Crossing nets: Dual generative models with a shared latent space for hand pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 680–689.

[24] H. Guo, G. Wang, X. Chen, C. Zhang, F. Qiao, and H. Yang, "Region ensemble network: Improving convolutional network for hand pose estimation," *Proc. Int. Conf. on Image Process.*, 2017.

[25] O. Koller, H. Ney, and R. Bowden, "Deep hand: How to train a CNN on 1 million hand images when your data is continuous and weakly labelled," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 3793–3802.

[26] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, "Cascaded hand pose regression," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 824–832.

[27] S. Song and J. Xiao, "Deep sliding shapes for amodal 3D object detection in RGB-D images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 808–816.

[28] A. Haque, B. Peng, Z. Luo, A. Alahi, S. Yeung, and F.-F. Li, "Towards viewpoint invariant 3D human pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 160–177.

[29] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 1991–2000.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 770–778.

[31] G. Huang, Z. Liu, L. v. d. Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 2261–2269.

[32] Intel. (2017) Intel realsense sdk. [Online]. Available: https://software.intel.com/en-us/intel-realsense-sdk

[33] A. Tagliasacchi, M. Schroeder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly, "Robust articulated-ICP for real-time hand tracking," *Computer Graphics Forum*, vol. 34, no. 5, pp. 101–114, Jul. 2015.

[34] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, "Realtime and robust hand tracking from depth," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2014, pp. 1106–1113.

[35] L. Ballan, A. Taneja, J. Gall, L. V. Gool, and M. Pollefeys, "Motion capture of hands in action using discriminative salient points," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 640–653.

[36] 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. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 2540–2548.

[37] D. Tzionas, L. Ballan, A. Srikantha, P. Aponte, M. Pollefeys, and J. Gall, "Capturing hands in action using discriminative salient points and physics simulation," *International Journal of Computer Vision*, vol. 118, no. 2, pp. 172–193, 2016.

[38] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt, "Fast and robust hand tracking using detection-guided optimization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 3213–3221.

[39] S. Sridhar, F. Mueller, M. Zollhoefer, D. Casas, A. Oulasvirta, and C. Theobalt, "Real-time joint tracking of a hand manipulating an object from RGB-D input," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 294–310.

[40] G. Shakhnarovich, P. Viola, and T. Darrell, "Fast pose estimation with parameter-sensitive hashing," in *Proc. Int. Conf. Comput. Vis.*, 2003, pp. 750–758.

[41] J. Romero, H. Kjellstrm, and D. Kragic, "Monocular real-time 3D articulated hand pose estimation," in *Proc. IEEE-RAS Conf. Humanoid Robots*, 2009, pp. 87–92.

[42] ——, "Hands in action: real-time 3D reconstruction of hands in interaction with objects," in *Proc. IEEE Conf. Robotics and Automation*, 2010, pp. 458–463.

[43] H. Liang, J. Yuan, and D. Thalmann, "Parsing the hand in depth images," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1241–1253, Aug. 2014.

[44] P. Li, H. Ling, X. Li, and C. Liao, "3D hand pose estimation using randomized decision forest with segmentation index points," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 819–827.

[45] C. Wan, A. Yao, and L. Van Gool, "Direction matters: hand pose estimation from local surface normals," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 554–569.

[46] P. Krejov, A. Gilbert, and R. Bowden, "Combining discriminative and model based approaches for hand pose estimation," in *Proc. IEEE Conf. Automatic Face and Gesture Recog.*, 2015, pp. 1–7.

[47] D. Tang, J. Taylor, P. Kohli, C. Keskin, T.-K. Kim, and J. Shotton, "Opening the black box: Hierarchical sampling optimization for estimating human hand pose," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 3325–3333.

[48] P. Krejov, A. Gilbert, and R. Bowden, "Guided optimisation through classification and regression for hand pose estimation," *Computer Vision and Image Understanding*, vol. 155, pp. 124–138, 2017.

[49] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, 2013.

[50] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 4489–4497.

[51] N. C. Camgoz, S. Hadfield, O. Koller, and R. Bowden, "Using convolutional 3D neural networks for user-independent continuous gesture recognition," in *Proc. Int. Conf. Pattern Recog.*, 2016, pp. 49–54.

[52] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3D shapenets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2015, pp. 1912–1920.

[53] C. R. Qi, H. Su, M. Niessner, A. Dai, M. Yan, and L. J. Guibas, "Volumetric and multi-view CNNs for object classification on 3D data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2016, pp. 5648–5656.

[54] D. Maturana and S. Scherer, "Voxnet: A 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Conf. Intell. Robots and Syst.*, 2015, pp. 922–928.

[55] ——, "3D convolutional neural networks for landing zone detection from lidar," in *Proc. IEEE Conf. Robotics and Automation*, 2015, pp. 3471–3478.

[56] M. E. Yumer and N. J. Mitra, "Learning semantic deformation flows with 3D convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 294–311.

[57] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser, "Semantic scene completion from a single depth image," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 1746–1754.

[58] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon, "Kinectfusion: Real-time dense surface mapping and tracking," in *Proc. IEEE Symp. Mixed and Augmented Reality*, 2011, pp. 127–136.

[59] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[60] J. T. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller, "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806*, 2014.

[61] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Medical Image Computing and Computer-Assisted Intervention*, 2015, pp. 234–241.

[62] A. Dai, C. R. Qi, and M. Nießner, "Shape completion using 3d-encoder-predictor cnns and shape synthesis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2017, pp. 5868–5877.

[63] R. Girshick, "Fast R-CNN," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.

[64] D. Kinga and J. B. Adam, "A method for stochastic optimization," in *Proc. Int. Conf. Learning Representations*, 2015.

[65] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 818–833.

[66] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 483–499.

[67] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon, "The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2012, pp. 103–110.

[68] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 1026–1034.

[69] C. Choi, A. Sinha, J. Hee Choi, S. Jang, and K. Ramani, "A collaborative filtering approach to real-time hand pose estimation," in *Proc. Int. Conf. Comput. Vis.*, 2015, pp. 2336–2344.

[70] M. Oberweger and V. Lepetit, "Deepprior++: Improving fast and accurate 3d hand pose estimation," in *Proc. Int. Conf. Comput. Vis. Workshop*, 2017, pp. 585–594.

[71] X. Chen, G. Wang, H. Guo, and C. Zhang, "Pose guided structured region ensemble network for cascaded hand pose estimation," *arXiv preprint arXiv:1708.03416*, 2017.

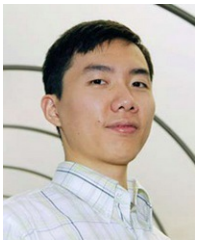
[72] H. Guo, G. Wang, X. Chen, and C. Zhang, "Towards good practices for deep 3d hand pose estimation," *arXiv preprint arXiv:1707.07248*, 2017.

[73] C. Xu, L. N. Govindarajan, Y. Zhang, and L. Cheng, "Lie-X: Depth image based articulated object pose estimation, tracking, and action recognition on lie groups," *International Journal of Computer Vision*, pp. 454–478, 2016.

[74] S. Pellegrini, K. Schindler, and D. Nardi, "A generalization of the ICP algorithm for articulated bodies," in *Proc. British Mach. Vis. Conf.*, 2008.

Liuhao Ge is working towards the PhD degree from the Institute for Media Innovation, Interdisciplinary Graduate School, Nanyang Technological University, Singapore. His research interests mainly include computer vision, multimedia, and machine learning.

Hui Liang received the B.Eng. degree in Electronics and Information Engineering and the M.Eng. degree in Communication and Information System from Huazhong University of Science and Technology in 2008 and 2011, and the Ph.D degree from Nanyang Technological University, Singapore in 2016. He was the Research Associate with the Institute for Media Innovation and Research Fellow with the Rapid-Rich Object Search Lab at Nanyang Technological University, Singapore. He is currently a research scientist in Institute of High Performance Computing, A*STAR, Singapore. His research interests mainly include computer vision, machine learning and human-computer interaction.



Junsong Yuan is an associate professor at School EEE, Nanyang Technological University. He received Ph.D. from Northwestern University in 2009. He is currently Senior Area Editor of Journal of Visual Communication and Image Representation (JVCI), Associate Editor of IEEE Trans. on Image Processing (T-IP), IEEE Trans. on Circuits and Systems for Video Technology (T-CSVT), and served as Guest Editor of International Journal of Computer Vision (IJCV). He is Program Co-chair of ICME18 and VCIP15, and Area Chair of ACM MM18, ACCV1814, ICPR1816, CVPR17, ICIP17 etc. He received 2016 Best Paper Award from IEEE Trans. on Multimedia, Doctoral Spotlight Award from IEEE Conf. on Computer Vision and Pattern Recognition (CVPR'09), Nanyang Assistant Professorship from NTU, and Outstanding EECS Ph.D. Thesis award from Northwestern University.

Daniel Thalmann Prof. Daniel Thalmann is one of the most highly cited scientists in Computer Graphics. He is currently Honorary Professor at EPFL, Switzerland, and Director of Research Development at MIRALab Sarl. Pioneer in research on Virtual Humans, his current research interests also include social robots, crowd simulation and Virtual Reality. Daniel Thalmann has been the Founder of The Virtual Reality Lab (VRlab) at EPFL, Switzerland. From 2009 to 2017, he was Visiting Professor at the Institute for Media Innovation, Nanyang Technological University, Singapore. He is coeditor-in-chief of the Journal of Computer Animation and Virtual Worlds, and member of the editorial board of 12 other journals. Daniel Thalmann was Program Chair and CoChair of several conferences including IEEE-VR, ACM-VRST, and ACM-VRCAI. Daniel Thalmann has published more than 600 papers in Graphics, Animation, and Virtual Reality. He received his PhD in Computer Science in 1977 from the University of Geneva and an Honorary Doctorate from University Paul-Sabatier in Toulouse, France, in 2003. He also received the Eurographics Distinguished Career Award in 2010, the 2012 Canadian Human Computer Communications Society Achievement Award, and the CGI 2015 Career Achievement.