3D Convolutional Neural Networks for Efficient and Robust Hand Pose Estimation from Single Depth Images

Liuhao Ge[1], Hui Liang[1,2,*], Junsong Yuan[1,†], Daniel Thalmann[1]
[1]Institute for Media Innovation, Interdisciplinary Graduate School, Nanyang Technological University
[2]Institute of High Performance Computing, A*STAR, Singapore
ge0001ao@ntu.edu.sg, lianghui@ihpc.a-star.edu.sg, {jsyuan, danielthalmann}@ntu.edu.sg

Abstract

We propose a simple, yet effective approach for real-time hand pose estimation from single depth images using three-dimensional Convolutional Neural Networks (3D 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 taking a 3D volumetric representation of the hand depth image as input can capture the 3D spatial structure of the input 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. Experiments show that our proposed 3D CNN based approach outperforms state-of-the-art methods on two challenging hand pose datasets, and is very efficient as our implementation runs at over 215 fps on a standard computer with a single GPU.

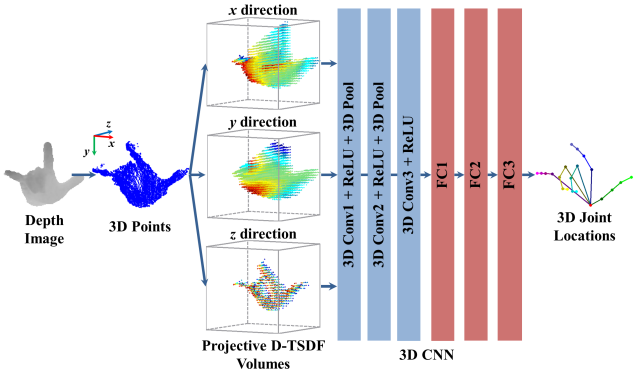


Figure 1: Overview of our proposed 3D CNN based hand pose estimation method. We generate the 3D volumetric representation of hand with projective D-TSDF from the 3D point cloud. 3D CNN is trained in an end-to-end manner to map the 3D volumetric representation to 3D hand joint relative locations in the 3D volume.

1. Introduction

Articulated 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 objects. Accurate real-time 3D hand pose estimation has aroused a lot of research attention in the past few years [7, 10, 15, 18, 21, 30, 32, 34, 39] with the emergence of consumer depth cameras. However, it is still challenging to achieve efficient and robust 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) [4, 17, 23, 34, 40, 42] and the availability of large hand pose datasets [28, 30, 34]. These methods directly take the depth image as input to 2D CNNs which output 3D joint locations [16, 17, 23, 40], hand model parameters [42] or heat-maps [34]. 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 [17], the initial result of 2D CNN is very poor, and it is iteratively refined by a feedback loop to incorporate 3D information from a generative model. Ge et al. [4] better utilize the depth cues by projecting the depth image onto three views and applying multi-view CNNs to regress three views' heat-maps. 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 be increased when using more views.

*This work was done when Hui Liang was a research staff at NTU.
†Corresponding author

In this work, 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 1. 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) [24] values, which are then fed into a 3D CNN containing three 3D convolutional layers and three fully-connected layers. The output of this network is a set of 3D hand joint relative locations in the 3D volume. By applying simple coordinate transformations, we can finally obtain the 3D hand joint locations in the camera's coordinate system. To our knowledge, this is the first work that applies such a 3D CNN in hand pose estimation in order to understand 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 has the ability to effectively learn 3D features from the 3D volumetric representation for hand pose estimation. Compared to the 2D CNN regressing 3D joint locations from 2D features [5, 16, 17, 40], the 3D CNN can directly regress 3D joint locations from 3D features in a single pass without adopting any iterative refinement process, and can achieve superior estimation performance.

- Our proposed method can run fast at over 215 fps on a single GPU. We design a relatively shallow architecture for the 3D CNN which contains only three 3D convolutional layers and three fully-connected layers. In addition, the number of parameters in fully-connected layers is moderate. Thus, our proposed method can meet the real-time requirement for hand pose estimation.

- 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.

We evaluate our proposed method on two challenging hand pose datasets: MSRA dataset [28] and NYU dataset [34]. Comprehensive experiments show that our proposed 3D CNN based method for 3D hand pose estimation outperforms state-of-the-art methods on both datasets, with runtime speed of over 215 fps on a standard computer with a single GPU.

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) [18], Iterative Closest Point (ICP) [29] and their combination [20]. The 3D hand model is represented by Linear Blend Skinning (LBS) model [1, 8, 36], Gaussian mixture model [25, 26], *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. Inspired by the pioneering work in human pose estimation [22], [7, 9, 28, 30, 31, 37, 39] apply random forests and their variants as the discriminative model. Limited by the hand-crafted features, random forests based methods 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. [34] first propose to employ CNNs to predict heat-maps representing the probability distribution of 2D joint positions in the depth image. Ge et al. [4] improve this method by predicting heat-maps on multiple views in order to better utilize the depth information. Oberweger et al. [17] train a feedback loop containing a discriminative network for initial pose estimation, a generative network for pose synthesizing and a pose update network for improving the pose estimation. Zhou et al. [42] propose to predict hand model parameters instead of the joint locations by adopting CNNs. Sinha et al. [23] extract activation features from CNNs to synchronize hand poses in nearest neighbors by using the matrix completion algorithm. Ye et al. [40] propose a spatial attention network with a hierarchical hybrid method for hand pose estimation. 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 [21, 32]. These methods are usually applied for hand tracking since they utilize temporal information to achieve smooth results. However, in this work, we focus on hand pose estimation from single depth images without using any temporal information, which can be used for robust reinitialization in hybrid hand tracking approaches.

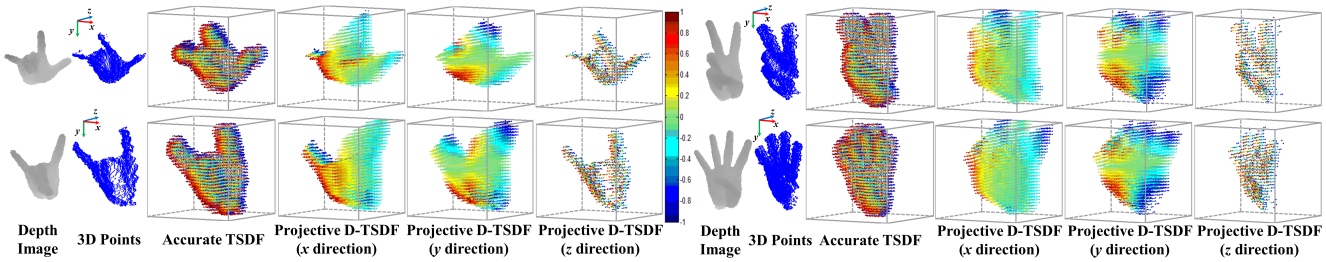


Figure 2: Visualization of TSDF volumes. For comparison, we visualize accurate TSDF and projective D-TSDF. We only visualize voxels of which values are less than 1 and larger than -1 by using color maps shown in the color bar. Positive value (red or yellow) indicates that the voxel is in front of the visible surface; and negative value (blue or cyan) indicates that the voxel is behind the visible surface. The volume resolution is $32\times32\times32$. This figure is best viewed in color.

3D CNN 3D CNNs have been successfully applied in video and dynamic hand gesture analysis for recognition tasks [6, 35, 13], which regard time as the third dimension. 3D CNNs are also applied to extract 3D features from 3D data, such as depth images and CAD models. 3D ShapeNets [38] learn powerful 3D features by using the Convolutional Deep Belief Network for modeling 3D shapes. Qi et al. [19] 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 [24] propose to use 3D CNN for 3D object detection in RGB-D images. Maturana and Scherer [12] 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 [11]. Yumer and Mitra [41] propose to use the 3D CNN to learn deformation flows from CAD models for 3D shape deformation. Although these works achieve state-of-the-art results in their problems, none of them focuses on 3D hand pose estimation that requires to localize a set of articulated 3D points from single depth images in real-time.

3. Methodology

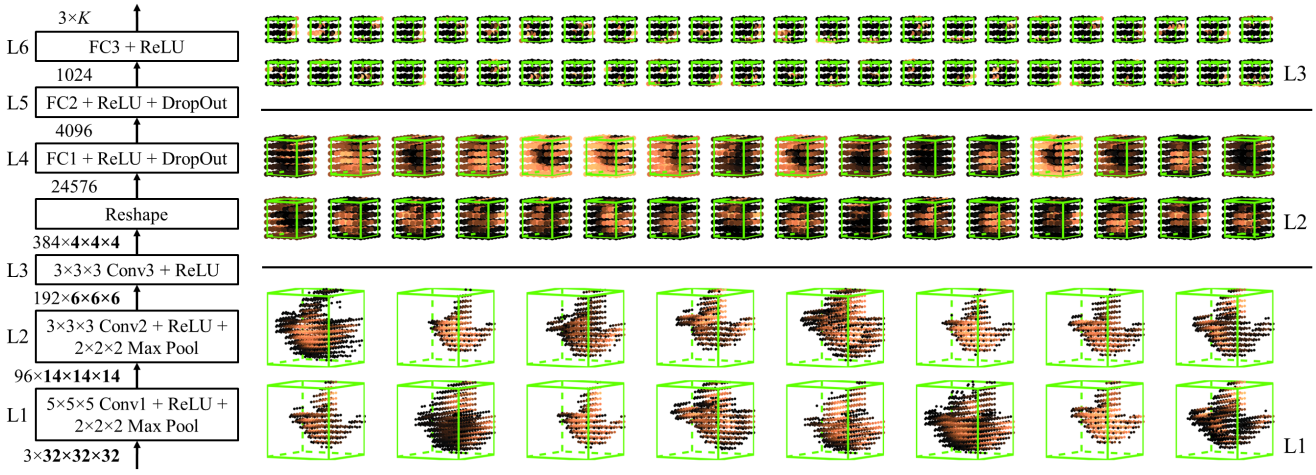
Our method estimates 3D hand pose from single depth images. Specifically, the input of this task is a depth image containing a hand and the outputs are K hand joint locations in 3D space, which represent the 3D hand pose. Let the K objective hand joint locations be $\mathbf{\Phi} = \{\boldsymbol{\phi}_k\}_{k=1}^{K} \in \mathbf{\Lambda}$, here $\mathbf{\Lambda}$ is the $3 \times K$ dimensional hand joint space.

The hand depth image is encoded by a volumetric representation which is the input of our proposed 3D CNN. Through 3D convolution and 3D pooling operations in the 3D CNN, 3D features can be extracted from the volumetric representation and are used for regressing 3D hand joint relative locations in the 3D volume. To make the 3D CNN robust to various hand sizes and global orientations, we also perform 3D data augmentation on the training data.

3.1. Volumetric Representation

The objective for encoding volumetric representation is to generate 3D volumes representing the hand in 3D space as raw as possible from the depth image in real-time. These 3D volumes will be fed into the 3D CNN for learning 3D features and regressing 3D hand joint locations.

If the input is a 3D CAD model where the 3D information is fully known, we can use a binary grid to represent occupied and unoccupied voxels in the 3D volume. However, in our problem, the input is a 2.5D depth image which only captures the visible surface points from the view of camera. 3D ShapeNets [38] classify voxels as free space, surface and occluded space. The probability distribution of occupancy in the occluded space is estimated for shape classification. But this method requires to traverse multiple camera views and different possible shapes, thus is hard to achieve real-time performance. KinectFusion [14] applies the Truncated Signed Distance Function (TSDF) based volumetric representation for environment mapping and localization with depth camera. In accurate TSDF, each voxel stores the signed distance from the voxel center to the closest surface point which is positive when the voxel is in front of the visible surface and negative when the voxel is occluded by the visible surface. The distance is cut off at a truncation distance and is normalized between -1 and 1. However, computing accurate TSDF is time consuming, because it requires to search the closest point among all surface points for all voxels in the 3D volume. For real-time considerations, the projective TSDF, 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 may be inaccurate or lost in the projective TSDF. In order to encode more information in the volumetric representation, in this work, we apply the projective Directional TSDF (D-TSDF) proposed in [24] that replaces the Euclidean distance with a 3D vector



(a) (b)

Figure 3: (a) Architecture of our proposed 3D convolutional neural network. The network contains three 3D convolutional layers and three fully-connected layers. (b) Visualization of extracted 3D features output from layers L1, L2 and L3 during the forward pass in a fully trained CNN model. For illustration purpose, we only draw 16, 32, 48 feature volumes output from L1, L2, L3, respectively. For feature volumes output from L1, we only draw voxels of which 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. This figure is best viewed in color.

$[dx, dy, dz]$ representing three directions' distances in the camera's coordinate system.

Figure 2 shows some examples of accurate TSDF volumes and projective D-TSDF volumes with different hand poses. As can be seen, in accurate TSDF, the value of TSDF increases when moving from the visible surface, formed by the point cloud, to the free space and decreases when moving to the occluded space. But in projective D-TSDF, the values of three directions vary continuously along their corresponding directions and keep positive in front of the surface, negative behind the surface. Experiments in Section 4.1 will show that the projective D-TSDF is computationally efficient and can improve the estimation accuracy.

To create a 3D volume containing $M \times M \times M$ voxels, we first build an axis-aligned bounding box (AABB) for 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 three edge lengths; M is the volume resolution value. The truncation distance is set as $3 \times l_{voxel}$. We balance the volume resolution value M with computational cost. If the volume resolution is too large, it will be time-consuming and memory intensive. If the volume resolution is too small, the volumetric representation cannot give sufficient information for 3D hand pose estimation. In this work, we choose the volume resolution value M as 32. Some experiments will be conducted in Section 4.1 to show that this resolution is suitable for 3D hand pose estimation when considering both estimation accuracy and computational efficiency.

3.2. Network Architecture

Our proposed 3D CNN takes three volumes of the projective D-TSDF as inputs and outputs a column vector containing $3 \times K$ elements corresponding to the K 3D hand joint relative locations in the volume. Figure 3a shows our proposed network architecture. 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 feature extraction by 3D convolutional layers, three fully-connected layers are used to map 3D features to 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 [27].

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 3.1. Ground truth $\mathbf{\Phi}_n$ is transformed to coordinates in the volume's coordinate system and nor-

malized between 0 and 1, denoted as $\boldsymbol{Y}_n$:

$$\boldsymbol{Y}_n = \mathbf{T}_{camera}^{volume}\left(\boldsymbol{\Phi}_n\right)/(M \cdot l_{voxel}) + 0.5, \qquad (2)$$

where $\mathbf{T}_{camera}^{volume}\left(\cdot\right)$ is a coordinate transformation operation converting joint locations $\boldsymbol{\Phi}$ in camera's coordinate system to those in volume's coordinate system. Since the origin of the volume's coordinate system is at the center of the 3D volume, coordinate values in the volume divided by the volume's edge length are between -0.5 and 0.5 (assume that all joints are within the 3D volume). Thus, we add 0.5 in Equation 2 to make values in $\boldsymbol{Y}_n$ between 0 and 1. During the training stage, we minimize the following objective function using stochastic gradient descent (SGD) algorithm:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} \sum_{n=1}^{N} \left\| \boldsymbol{Y}_n - \mathcal{F}\left(V_n, \boldsymbol{w}\right) \right\|^2, \qquad (3)$$

where $\boldsymbol{w}$ denotes network weights, $\mathcal{F}$ represents the 3D CNN regressor.

In Figure 3b, we visualize some extracted 3D features output from layers L1, L2 and L3 during the forward pass using a fully trained 3D CNN model. The corresponding input is the example shown in Figure 1. As can be seen, from low layer to high layer, the size of feature volume decreases and features are more and more abstract. For each layer, different parts (e.g., finger tips and hand palm) are exaggerated in different 3D feature volumes. In addition, the receptive field size of the third convolutional layer (L3) is 20, which can cover a large region in the 3D volume. With such a large receptive field size, the network can capture spatial dependencies of 3D hand joints and embed the 3D joint constraints in an implicit way without using any explicit hand model or post-processing.

3.3. 3D Data Augmentation

One challenge of 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, we propose to perform 3D data augmentation on the training data. Different from existing 2D image data augmentation, our method can directly rotate and stretch the hand point cloud in 3D space.

We first stretch the point cloud along x, y, z axes of the camera's coordinate system by 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 rotation, the point $\boldsymbol{p}$ is transformed into $\boldsymbol{p}'$:

$$\begin{aligned} \boldsymbol{p}' &= \mathcal{R} \cdot \mathcal{S} \cdot \boldsymbol{p} \\ \mathcal{R} &= \mathcal{R}_x\left(\theta_x\right) \cdot \mathcal{R}_y\left(\theta_y\right) \cdot \mathcal{R}_z\left(\theta_z\right) \\ \mathcal{S} &= \mathrm{Diag}\left(s_x, s_y, s_z\right), \end{aligned} \qquad (4)$$

where $\mathcal{R}_x$, $\mathcal{R}_y$ and $\mathcal{R}_z$ are 3×3 rotation matrices around x, y, z axes, respectively; $\mathrm{Diag}\left(s_x, s_y, s_z\right)$ is a 3×3 diagonal matrix whose diagonal entries starting in the upper left corner are s_x, s_y and s_z. Figure 4 shows an example of 3D data augmentation. 3D stretching and rotation are performed on hand point cloud and corresponding ground truth. TSDF volumes are generated from the transformed point cloud.

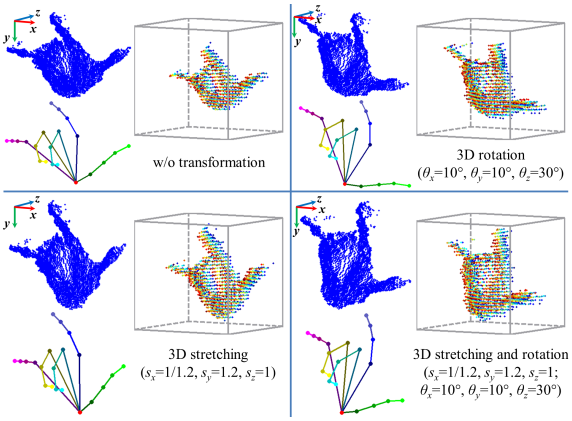


Figure 4: An example of 3D data augmentation. **Top-left**: original point cloud, ground truth and TSDF volume. **Bottom-left**: point cloud, ground truth and TSDF volume after 3D stretching. **Top-right**: point cloud, ground truth and TSDF volume after 3D rotation. **Bottom-right**: point cloud, ground truth and TSDF volume after 3D stretching and rotation. For illustration purpose, we only draw the projective D-TSDF volume on z direction.

In this work, a transformed training set is generated by randomly stretching and rotating original training samples. The rotation angles θ_x and θ_y are chosen uniformly at random from the interval $[-45°, 45°]$. The rotation angle θ_z is chosen uniformly at random from the interval $[-90°, 90°]$. The stretch factors s_x and s_y are chosen log-uniformly at random from the interval $[1/1.3, 1.3]$. 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. During the training stage, both the original training set and the transformed training set are used for training.

4. Experiments

We evaluate our proposed method on two public hand pose datasets: MSRA dataset [28] and NYU dataset [34]. 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 [33], which is a strict measure. The third metric is the proportion of joints within an error threshold [21].

All experiments are conducted on a computer with two

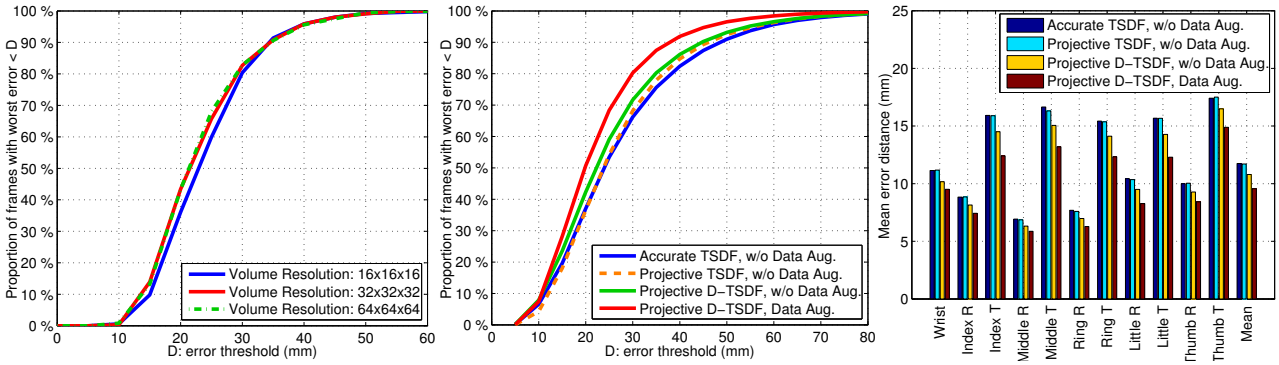


Figure 5: Self-comparison of different methods on MSRA dataset [28]. **Left**: the impact of different volume resolutions on the proportion of good frames. **Middle**: the impact of different TSDF types and data augmentation on the proportion of good frames. **Right**: the impact of different TSDF types and data augmentation on the per-joint mean error distance (R:root, T:tip).

Intel Core i7 5930K 3.50GHz, 64GB of RAM and an Nvidia Quadro K5200 GPU. The 3D CNN model is implemented within the Torch7 [3] framework. For network training parameters, we choose the batch size as 16, the learning rate as 0.01, the momentum as 0.9 and the weight decay as 0.0005. The networks are trained for 50 epochs. Training the 3D CNN proposed in Section 3.2 with 3D data augmentation takes about 12 hours on the MSRA dataset and 13 hours on the NYU dataset.

4.1. MSRA Hand Pose Dataset

The MSRA hand pose dataset [28] contains 9 subjects, each subject contains 17 gestures and each gesture contains about 500 frames. In the experiment, we train on 8 subjects and test on the remaining subject. This experiment is repeated 9 times for all subjects. The ground truth for each frame contains 21 3D hand joint locations including 4 joints for each finger and one joint for the wrist.

Self-comparisons For self-comparison, we experiment with different volume resolutions and different types of TSDF. We also evaluate the effect of 3D data augmentation. Experimental results are shown in Figure 5.

We experiment with projective D-TSDF volumes with different resolution values: 16, 32 and 64. Note that when the volume resolution is $16\times16\times16$ or $64\times64\times64$, the network architecture is different with that in Figure 3a. We modify the architectures according to different volume resolutions and present them in the supplementary material. Since training the network with $64\times64\times64$ volume resolution will consume large amounts of memory, 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 5 (left), the estimation accuracy of $16\times16\times16$ resolution is slightly inferior to those with $32\times32\times32$ and $64\times64\times64$ resolutions. The estimation accuracy of the latter two resolutions is almost the same. But computing TSDF volume with $64\times64\times64$ resolution is more time consuming and memory intensive. Thus, the volume resolution $32\times32\times32$ is most suitable for hand pose estimation. This result 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.

We evaluate the impact of different TSDF types and data augmentation on the estimation accuracy on the whole MSRA dataset with volume resolution $32\times32\times32$. Note that, in this experiment, when the input volume is accurate/projective TSDF which has only one channel, the parameters of the network architecture in Figure 3a should be modified. We present the modified architecture in the supplementary material. As can be seen in Figure 5 (middle and right), the estimation accuracy of accurate TSDF and projective TSDF is almost the same, which indicates that using an approximation of the accurate TSDF to speed up computation will not reduce the estimation accuracy of hand pose estimation. In addition, the estimation accuracy of projective D-TSDF is better than that of projective TSDF. When using 3D data augmentation in the training stage, the estimation accuracy will be further improved. For the real-time performance, the average computation time for generating accurate TSDF, projective TSDF and projective D-TSDF on the same GPU are 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 better than accurate TSDF and projective TSDF. In the following experiments, we apply the projective D-TSDF with $32\times32\times32$ volume resolution as the network input and apply 3D data augmentation for training.

Comparison with state-of-the-art We compare our 3D CNN based hand pose estimation method with four state-of-the-art methods: the hierarchical regression method [28], the collaborative filtering method [2], the multi-view CNN

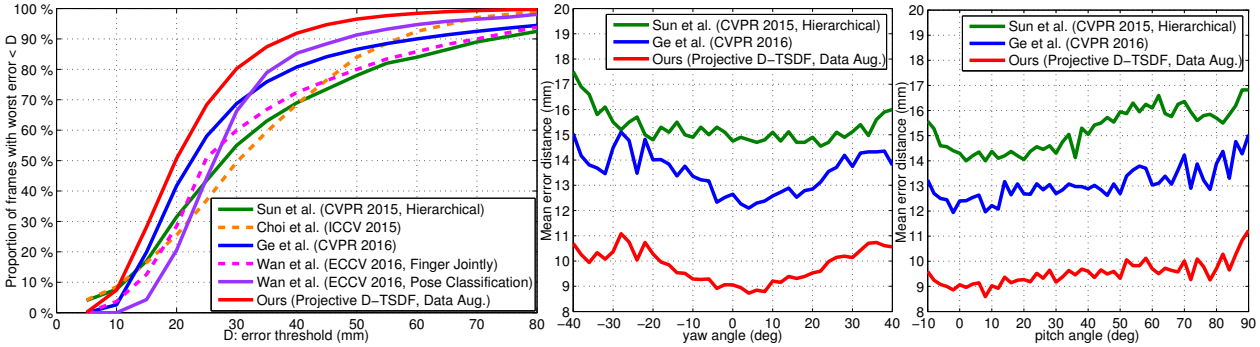


Figure 6: Comparison with state-of-the-art methods [28, 2, 4, 37] on MSRA dataset [28]. **Left**: the proportion of good frames over different error thresholds. **Middle & right**: the mean error distance over different yaw and pitch viewpoint angles with respect to the camera frame. Some curves are cropped from corresponding figures reported in [28, 2, 4, 37].

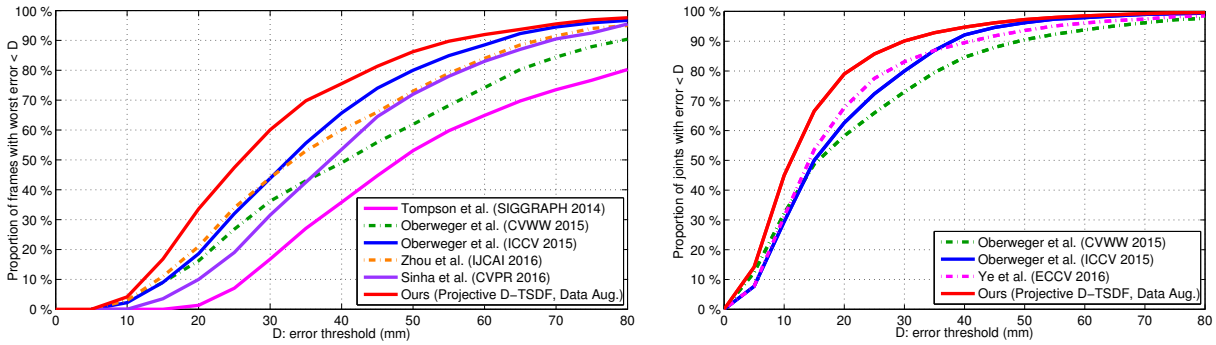


Figure 7: Comparison with state-of-the-art methods [34, 16, 17, 42, 23, 40] on NYU dataset [34]. **Left**: the proportion of good frames over different error thresholds. **Right**: the proportion of joints within different error thresholds. Some curves are cropped from corresponding figures reported in [16, 17, 42, 23, 40].

based method [4] and the local surface normals based method [37] on the whole MSRA dataset. Note that since the hierarchical regression method [28] has been shown superior to the methods in [22, 30, 39], we indirectly compare our method with [22, 30, 39].

As shown in Figure 6, our 3D CNN based method outperforms state-of-the-art methods by large margin on the MSRA dataset. The proportion of good frames over different error thresholds is shown in Figure 6 (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 10%, 12%, 25% and 30% higher than those of the methods in [4], [37] (pose classification), [28] and [2], respectively. When the error threshold is 5mm, the proportion of good frames of our method is slightly worse than those of the methods in [28] and [2]. This may be caused by the relatively large edge length of the voxel, which is 5.5mm in average when the volume resolution is $32\times32\times32$. In Figure 6 (middle and right), we compare the mean error distance over different yaw and pitch viewpoint angles with the methods in [28, 4]. As can be seen, the mean errors over different viewpoint angles of our method are about 5.5mm and 3mm smaller than those of the methods in [28] and [4], respectively. Our method exhibits less variance to the pitch viewpoint angle changes with a smaller standard deviation (0.48mm) than those of the methods in [28] (0.79mm) and [4] (0.64mm).

4.2. NYU Hand Pose Dataset

The NYU hand pose dataset [34] contains 72,757 training frames and 8,252 testing frames with continuous hand poses. The ground truth for each frame contains 36 3D hand joint locations. In our experiments, the 3D CNN is trained to estimate a subset of 14 hand joints, following previous work in [34, 17]. Since the NYU dataset provides the original depth image containing human body and background, we segment the hand from the depth image by using the Random Decision Forest (RDF) [22] similar to [34].

We first compare our 3D CNN based hand pose estima-

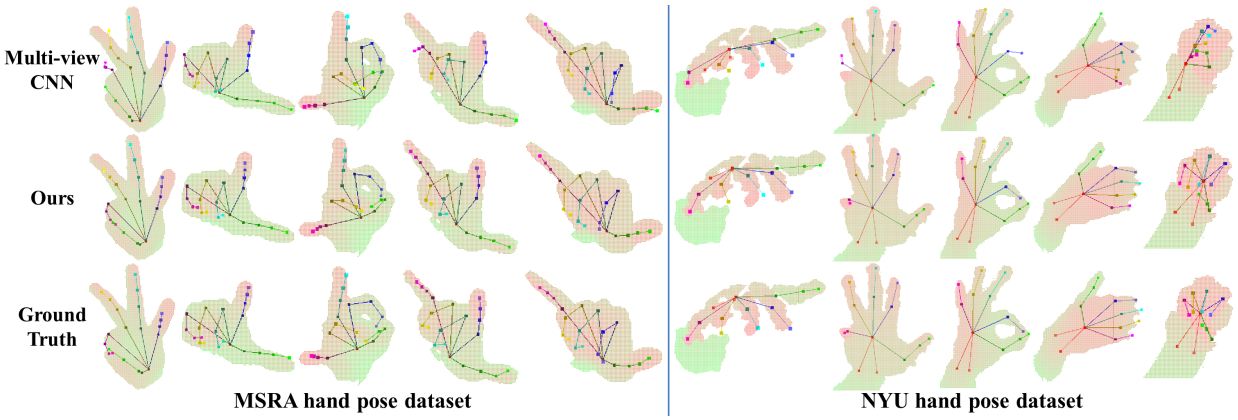


Figure 8: Qualitative results for MSRA dataset [28] and NYU dataset [34]. We compare our 3D CNN based method (in the second line) with the multi-view CNN based method in [4] (in the first line). The ground truth hand joint locations are presented in the last line. We show hand joint locations on the depth image. Different hand joints and bones are visualized using different colors. This figure is best viewed in color.

tion method with five state-of-the-art methods: the 2D CNN based heatmap regression method [34], the 2D CNN based direct regression method with a prior [16], the 2D CNN based regression method using feedback loop [17], the 2D CNN based hand model parameters regression method [42] and the deep feature based matrix completion method [23] on the NYU dataset. This evaluation is performed on the 14 hand joints. As shown in Figure 7 (left), our method significantly outperforms these five state-of-the-art methods over all the error thresholds. For example, the proportion of good frames of our method is about 10% more than that of the method in [17] when the error threshold is between 20mm and 40mm.

In order to make a fair comparison with the spatial attention network based hierarchical hybrid method in [40], we evaluate the proportion of joints within in different error thresholds on the subset of 11 hand joints following the experiment in [40] (removing palm joints except the root joint of thumb). As shown in Figure 7 (right), our method is superior to the methods in [16, 17, 40] over all the error thresholds. For example, the proportion of joints within error threshold 20mm of our method is about 10% more than that of the method in [40].

4.3. Runtime and Qualitative Results

Runtime The runtime of our method using the projective D-TSDF with $32\times32\times32$ volume resolution as network input is 4.6ms in average, including 2.9ms for the projective D-TSDF volume generation, 1.5ms for the 3D CNN forward propagation and 0.18ms for coordinate transformation. Thus, our method runs in real-time at over 215fps. The processes of volume generation and 3D CNN forward propagation are performed on GPU. The coordinate transformation that converts CNN output values to 3D locations in the camera's coordinate system is performed on CPU. In addition, our 3D CNN model takes about 500 MB of GPU memory during testing, while the multi-view CNNs in [4] take about 1.5 GB of GPU memory during testing.

Qualitative results Some qualitative results for MSRA dataset and NYU dataset are shown in Figure 8. As can be seen, compared with the multi-view CNN based method in [4], our 3D CNN based method can better utilize the depth information and provide more accurate estimation.

5. 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 train. We also perform 3D data augmentation on the training data to make the 3D CNN robust to various hand sizes and global orientations. Experimental results indicate that our proposed 3D CNN based approach achieves state-of-the-art performance for 3D hand pose estimation in real-time on two challenging datasets.

Acknowledgment: 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] L. Ballan, A. Taneja, J. Gall, L. V. Gool, and M. Pollefeys. Motion capture of hands in action using discriminative salient points. In *ECCV*, 2012.

[2] C. Choi, A. Sinha, J. Hee Choi, S. Jang, and K. Ramani. A collaborative filtering approach to real-time hand pose estimation. In *ICCV*, 2015.

[3] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011.

[4] 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.

[5] A. Haque, B. Peng, Z. Luo, A. Alahi, S. Yeung, and F.-F. Li. Towards viewpoint invariant 3D human pose estimation. In *ECCV*, 2016.

[6] S. Ji, W. Xu, M. Yang, and K. Yu. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(1):221–231, 2013.

[7] C. Keskin, F. Kra, Y. Kara, and L. Akarun. Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In *ECCV*, 2012.

[8] 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.

[9] H. Liang, J. Yuan, and D. Thalmann. Parsing the hand in depth images. *IEEE Transactions on Multimedia*, 16(5):1241–1253, 2014.

[10] H. Liang, J. Yuan, and D. Thalmann. Resolving ambiguous hand pose predictions by exploiting part correlations. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(7):1125–1139, 2015.

[11] D. Maturana and S. Scherer. 3D convolutional neural networks for landing zone detection from lidar. In *ICRA*, 2015.

[12] D. Maturana and S. Scherer. Voxnet: A 3D convolutional neural network for real-time object recognition. In *IROS*, 2015.

[13] 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 *CVPR*, 2016.

[14] 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 *ISMAR*, 2011.

[15] M. Oberweger, G. Riegler, P. Wohlhart, and V. Lepetit. Efficiently creating 3d training data for fine hand pose estimation. In *CVPR*, 2016.

[16] M. Oberweger, P. Wohlhart, and V. Lepetit. Hands deep in deep learning for hand pose estimation. In *CVWW*, 2015.

[17] M. Oberweger, P. Wohlhart, and V. Lepetit. Training a feedback loop for hand pose estimation. In *ICCV*, 2015.

[18] I. Oikonomidis, N. Kyriazis, and A. Argyros. Efficient model-based 3D tracking of hand articulations using Kinect. In *BMVC*, 2011.

[19] 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 *CVPR*, 2016.

[20] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *CVPR*, 2014.

[21] 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. In *CHI*, 2015.

[22] 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 *CVPR*, 2011.

[23] A. Sinha, C. Choi, and K. Ramani. Deephand: Robust hand pose estimation by completing a matrix with deep features. In *CVPR*, 2016.

[24] S. Song and J. Xiao. Deep Sliding Shapes for amodal 3D object detection in RGB-D images. In *CVPR*, 2016.

[25] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt. Fast and robust hand tracking using detection-guided optimization. In *CVPR*, 2015.

[26] 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 *ECCV*, 2016.

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

[28] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *CVPR*, 2015.

[29] A. Tagliasacchi, M. Schroeder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly. Robust articulated-icp for real-time hand tracking. *Computer Graphics Forum*, 34(5), 2015.

[30] D. Tang, H. J. Chang, A. Tejani, and T. K. Kim. Latent regression forest: Structured estimation of 3D articulated hand posture. In *CVPR*, 2014.

[31] 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 *ICCV*, 2015.

[32] 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 Transactions on Graphics*, 35(4):143, 2016.

[33] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *CVPR*, 2012.

[34] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics*, 33(5):169, 2014.

[35] D. Tran, L. D. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3D convolutional networks. In *ICCV*, 2015.

[36] 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*, 118(2):172–193, 2016.

[37] C. Wan, A. Yao, and L. Van Gool. Direction matters: hand pose estimation from local surface normals. In *ECCV*, 2016.

[38] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D shapenets: A deep representation for volumetric shapes. In *CVPR*, 2015.

[39] C. Xu and L. Cheng. Efficient hand pose estimation from a single depth image. In *ICCV*, 2013.

[40] Q. Ye, S. Yuan, and T.-K. Kim. Spatial attention deep net with partial pso for hierarchical hybrid hand pose estimation. In *ECCV*, 2016.

[41] M. E. Yumer and N. J. Mitra. Learning semantic deformation flows with 3D convolutional networks. In *ECCV*, 2016.

[42] X. Zhou, Q. Wan, W. Zhang, X. Xue, and Y. Wei. Model-based deep hand pose estimation. In *IJCAI*, 2016.