Robust 3D Hand Pose Estimation in Single Depth Images: from Single-View CNN to Multi-View CNNs

Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann
Institute for Media Innovation
Nanyang Technological University, Singapore
{ge0001ao, hliang1}@e.ntu.edu.sg, {jsyuan, danielthalmann}@ntu.edu.sg

Abstract

Articulated hand pose estimation plays an important role in human-computer interaction. Despite the recent progress, the accuracy of existing methods is still not satisfactory, partially due to the difficulty of embedded high-dimensional and non-linear regression problem. Different from the existing discriminative methods that regress for the hand pose with a single depth image, we propose to first project the query depth image onto three orthogonal planes and utilize these multi-view projections to regress for 2D heat-maps which estimate the joint positions on each plane. These multi-view heat-maps are then fused to produce final 3D hand pose estimation with learned pose priors. Experiments show that the proposed method largely outperforms state-of-the-art on a challenging dataset. Moreover, a cross-dataset experiment also demonstrates the good generalization ability of the proposed method.

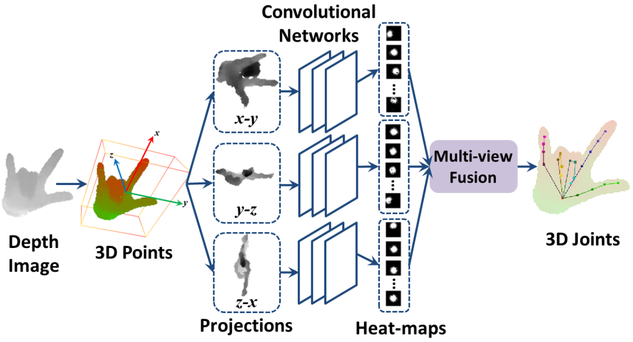


Figure 1: Overview of our proposed multi-view regression framework. We generate heat-maps for three views by projecting 3D points onto three orthogonal planes. Three CNNs are trained in parallel to map each view's projected image to its corresponding heat-maps, which are then fused together to estimate 3D hand joint locations.

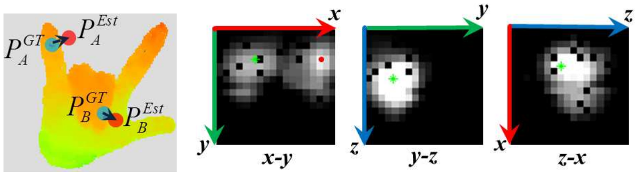
1. Introduction

The problem of 3D hand pose estimation has aroused a lot of attention in computer vision community for long, as it plays a significant role in human-computer interaction such as virtual/augmented reality applications. Despite the recent progress in this field [14, 18, 21, 23, 29], robust and accurate hand pose estimation remains a challenging task. Due to large pose variations and high dimension of hand motion, it is generally difficult to build an efficient mapping from image features to articulated hand pose parameters.

Data-driven methods for hand pose estimation train discriminative models, such as isometric self-organizing map [4], random forests [7, 21, 24, 25] and convolutional neural networks (CNNs) [29], to map image features to hand pose parameters. With the availability of large annotated hand pose datasets [21, 24, 29], data-driven approaches become more advantageous as they do not require complex model calibration and are robust to poor initialization.

We focus on CNN-based data-driven methods in this paper. CNNs have been applied in body and hand pose estimation [27, 29, 30] and have shown to be effective. The main difficulty of CNN-based methods for hand pose estimation lies in accurate 3D hand pose regression. Direct mapping from input image to 3D locations is highly non-linear with high learning complexity and low generalization ability of the networks [27]. One alternative way is to map input image to a set of heat-maps which represent the probability distributions of joint positions in the image and recover the 3D joint locations from the depth image with model fitting [29]. However, in this method, the heat-map only provides 2D information of the hand joint and the depth information is not fully utilized.

In this work, we propose a novel 3D regression method using multi-view CNNs that can better exploit depth cues to recover fully 3D information of hand joints without model fitting, as illustrated in Fig. 1. Specifically, the point cloud of an input depth image is first projected onto three orthogo-



(a) (b)

Figure 2: (a) Illustration of joint estimation in single view. Blue points are true locations, and red points are estimated locations. The little finger tip is misestimated on the background and the middle finger tip is misestimated on the hand palm. (b) Illustration of ambiguous estimation. Despite the heat-map of *x*-*y* view contains two hotspots which are hard to choose, from the heat-map of *z*-*x* view, it is easy to find that the *x* value is small with high confidence. Thus, the left hotspot in *x*-*y* view's heat-map is true.

nal planes, and each projected image is then fed into a separate CNN to generate a set of heat-maps for hand joints following similar pipeline in [29]. As the heat-map in each view encodes the 2D distribution of a joint on the projection plane, their combination in three views thus contains the location distribution of the joint in 3D space. By fusing heat-maps of three views with pre-learned hand pose priors, we can finally obtain the 3D joint locations and alleviate ambiguous estimations at the same time.

Compared to the method of single view CNN in [29], our proposed method of multi-view CNNs has the following advantages:

- In the single view CNN, the depth of a hand joint is taken as the corresponding depth value at the estimated 2D position, which may result in large depth estimation errors even if the estimated 2D position is only slightly deviated from the true joint position, as shown in Fig. 2a. In contrast, our proposed multi-view CNNs generate heat-maps for front, side and top views simultaneously, from which the 3D locations of hand joints can be estimated more robustly.
- In case of ambiguous estimations, the single view CNN cannot well differentiate among multiple hotspots in the heat-map, in which only one could correspond to the true joint, as shown in Fig. 2b (*x*-*y* view). With the proposed multi-view CNNs, the heat-maps from other two views can help to eliminate the ambiguity, such as that in Fig. 2b.

single-view CNN

- Different from [29] that still relies on a pre-defined hand model to obtain the final estimation, our proposed approach embeds hand pose constraints learned from training samples in an implicit way, which allows to enforce hand motion constraints without manually defining hand size parameters.

Comprehensive experiments validate the superior performance of the proposed method compared to state-of-the-art methods on public datasets [21], with runtime speed of over 70fps. In addition, our proposed multi-view regression method can achieve relatively high accuracy in cross-dataset experiments [17, 21].

2. Literature Review

Vision-based hand pose estimation has been extensively studied in literature over many years. The most common hand pose estimation techniques can be classified into model-driven approaches and data-driven approaches [22]. Model-driven methods usually find the optimal hand pose parameters via fitting a deformable 3D hand model to input image observations. Such methods have demonstrated to be quite effective, especially with the depth cameras [15, 17, 19, 23]. However, there are some shortcomings for the model-driven methods. For instance, they usually need to explicitly define the anatomical size and hand motion constraints of the hand to match to the input image. Also, due to the high dimensional of hand pose parameters, they can be sensitive to initialization for the iterative model-fitting procedure to converge to the optimal pose.

In contrast, the data-driven methods do not need the explicit specification of the hand size and motion constraints. Rather, such information is automatically encoded in the training data. Therefore, many recent methods are built upon such a scheme [10, 11, 13, 21, 25, 32]. Among them, the random forest and its variants have proved to be reasonably accurate and fast. In [32], the authors propose to use the random forest to directly regress for the hand joint angles from depth images, in which a set of spatial-voting pixels cast their votes for hand pose independently and their votes are clustered into a set of candidates. The optimal one is determined by a verification stage with a hand model. A similar method is presented in [25], which further adopts transfer learning to make up for the inconsistence between synthesis and real-world data. As the estimations from random forest can be ambiguous for complex hand postures, pre-learned hand pose priors are sometimes utilized to better fuse independently predicted hand joint distributions [8, 12]. In [21], the cascaded pose regression algorithm [3] is adapted to the problem of hand pose estimation. Particularly, the authors propose to first predict the root joints of the hand skeleton, based on which the rest joints are updated. In this way the hand pose constraints can be well preserved during pose regression.

Very recently, convolutional neural networks have shown to be effective in articulated pose estimation. In [30], they are tuned to regress for the 2D human poses by directly minimizing the pose estimation error on the training data. The results have shown to outperform the traditional methods largely. However, it takes more than twenty days to train the network and the dataset only contains several thousand images. Considering the relatively small size of the dataset used in [30], it can be difficult to use it on larger datasets such as [21, 24], which consist of more than 70K images. Also, it is reported in [5, 27] that such direct mapping with CNNs from image features to continuous 2D/3D locations is of high nonlinearity and complexity as well as low generalization ability, which renders it difficult to train CNNs in such a manner. To this end, in their work on body pose estimation [27, 28], the CNNs are used to predict the heat-maps of joint positions instead of the original articulated pose parameters, and on each heat-map the intensity of a pixel indicates the likelihood for a joint occurring there. During training, the regression error is instead defined as the L2-norm of the difference between the estimated heat-map and the ground truth heat-map. In this way, the network can be trained efficiently and they achieve state-of-the-art performances. Similarly, such a framework has also been applied in 3D hand pose estimation [29]. However, the heat-map only provides 2D information of the hand joint and the depth information is not fully utilized. To address this issue, a model-based verification stage is adopted to estimate the 3D hand pose based on the estimated heat-maps and the input depth image [29]. Such heat-map based approaches are interesting as heat-maps can reflect the probability distribution of 3D hand joints in the projection plane. Inspired by such methods, we generate heat-maps of multiple views and fuse them together to estimate the probability distribution of hand joints in 3D space.

3. Methodology

The task of the hand pose estimation can be regarded as the extraction of the 3D hand joint locations from the depth image. Specifically, the input of this task is a cropped depth image only containing a human hand with some gesture and the outputs are K 3D hand joint locations which represent the hand pose. Let the K objective hand joint locations be $\mathbf{\Phi} = \{\phi_k\}_{k=1}^{K} \in \mathbf{\Lambda}$, here $\mathbf{\Lambda}$ is the $3 \times K$ dimensional hand joint space, and in this work $K = 21$. The 21 objective hand joint locations are the wrist center, the five metacarpophalangeal joints, the five proximal interphalangeal joints, the five distal interphalangeal joints and the five finger tips.

Following the discussion in Section 1, we propose to infer 3D hand joint locations $\mathbf{\Phi}$ based on the projected images on three orthogonal planes. Let the three projected images be I_{xy}, I_{yz} and I_{zx}, which are obtained by projecting 3D points from the depth image onto *x*-*y*, *y*-*z* and *z*-*x* planes in the projection coordinate system, respectively. Thus, the query depth image I_D is transformed to the three projections I_{xy}, I_{yz} and I_{zx}, which will be used as the inputs to infer 3D hand joint locations in our following derivations.

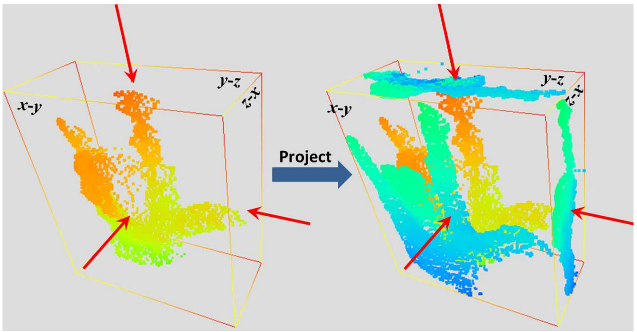


Figure 3: Illustration of 3D points projection. 3D points obtained from the input depth image are projected onto *x*-*y*, *y*-*z* and *z*-*x* planes of the OBB coordinate system, respectively.

We estimate the hand joint locations $\mathbf{\Phi}$ by applying the MAP (maximum a posterior) estimator on the basis of projections I_{xy}, I_{yz} and I_{zx}, which can be viewed as the observations of the 3D hand pose. Given $(I_D, \mathbf{\Phi})$, we assume that the three projections I_{xy}, I_{yz} and I_{zx} are independent, conditioned on the joint locations $\mathbf{\Phi}$ [1, 33]. Under this assumption and the assumption of equal a priori probability $P(\mathbf{\Phi})$, the posterior probability of joint locations can be formulated as the product of the individual estimations from all the three views. The problem to find the optimal hand joint locations $\mathbf{\Phi}^*$ is thus formulated as follows:

$$\begin{aligned}
\mathbf{\Phi}^* &= \arg\max_{\mathbf{\Phi}} P\left(\mathbf{\Phi} | I_{xy}, I_{yz}, I_{zx}\right) \\
&= \arg\max_{\mathbf{\Phi}} P\left(I_{xy}, I_{yz}, I_{zx} | \mathbf{\Phi}\right) \\
&= \arg\max_{\mathbf{\Phi}} P\left(I_{xy} | \mathbf{\Phi}\right) P\left(I_{yz} | \mathbf{\Phi}\right) P\left(I_{zx} | \mathbf{\Phi}\right) \qquad (1) \\
&= \arg\max_{\mathbf{\Phi}} P\left(\mathbf{\Phi} | I_{xy}\right) P\left(\mathbf{\Phi} | I_{yz}\right) P\left(\mathbf{\Phi} | I_{zx}\right) \\
s.t.\ & \mathbf{\Phi} \in \mathbf{\Omega}
\end{aligned}$$

where $\mathbf{\Phi}$ is constrained to a low dimensional subspace $\mathbf{\Omega} \subseteq \mathbf{\Lambda}$ in order to resolve ambiguous joint estimations.

The posterior probabilities $P\left(\phi_k | I_{xy}\right)$, $P\left(\phi_k | I_{yz}\right)$ and $P\left(\phi_k | I_{zx}\right)$ can be estimated from heat-maps generated by CNNs. Now we present the details of multi-view 3D joint location regression. We first describe the methods of multi-view projection and learning in Section 3.1 and then describe the method of multi-view fusion in Section 3.2.

3.1. Multi-view Projection and Learning

The objective for multi-view projection and learning is to generate projected images on each view and learn the relations between the projected images and the heat-maps of each view. First, we describe the details of 3D projections. Then, we introduce the architecture of the CNNs.

3D Points Projection: As illustrated in Fig. 1, the input depth image is first converted to a set of 3D points in the world coordinate system by using the depth camera's

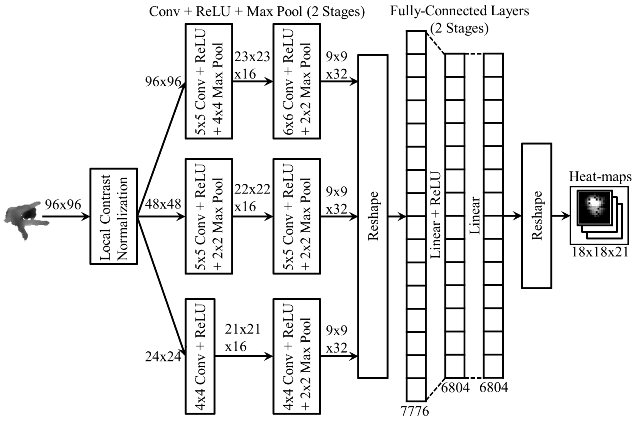

Conv + ReLU + Max Pool (2 Stages)
Fully-Connected Layers
(2 Stages)
96x96
Local Contrast Normalization
96x96
48x48
24x24
5x5 Conv + ReLU + 4x4 Max Pool
23x23 x16
6x6 Conv + ReLU + 2x2 Max Pool
5x5 Conv + ReLU + 2x2 Max Pool
22x22 x16
5x5 Conv + ReLU + 2x2 Max Pool
4x4 Conv + ReLU
21x21 x16
4x4 Conv + ReLU + 2x2 Max Pool
9x9 x32
9x9 x32
9x9 x32
Reshape
Linear + ReLU
Linear
7776
6804
6804
Reshape
Heat-maps
18x18x21

Figure 4: Convolutional Network architecture for each view. The network contains convolutional layers and fully-connected layers. In convolutional layers, there are three banks for multi-resolution inputs. The network generates 21 heat-maps with the size of 18x18 pixels. All of the three views have the same network architecture and the same architectural parameters.

intrinsic parameters, e.g. the position of principal point and the focal length. To generate multi-view's projections, we project these 3D points onto three orthogonal planes. As shown in Fig. 3, an oriented bounding box (OBB) is generated by performing principal component analysis (PCA) on the set of 3D points, which is a tight fit around these 3D points in local space [31]. The origin of OBB coordinate system is set on the center of the bounding box, and its x, y, z axes are respectively aligned with the 1st, 2nd and 3rd principal components. This coordinate system is set as the projection coordinate system.

For 3D points projection onto a plane, the distances from 3D points to the projection plane are normalized between 0 and 1 (with nearest points set to 0, farthest points set to 1). Then, 3D points are orthographically projected onto the OBB coordinate system's x-y, y-z and z-x planes respectively, as shown in Fig. 3. The corresponding normalized distances are stored as pixel values of the projected images. If multiple 3D points are projected onto the same pixel, the smallest normalized distance will be stored as the pixel value. Notice that the projections on the three orthogonal planes maybe coarse because of the resolution of the depth map [9], which can be solved by performing median filter and opening operation on the projected images.

Architecture of CNNs: Since we project 3D points onto three views, for each view, we construct a convolutional network having the same network architecture and the same architectural parameters. Inspired by the work of Tompson et al. in [29], we employ the multi-resolution convolutional networks architecture for each view as shown in Fig. 4. The input projected images are resized to 96x96 pixels and then filtered by local contrast normalization (LCN) [6] to normalize the contrast in the image. After LCN, the 96x96 image is down-sampled to 48x48 and 24x24 pixels. All of these three images with different resolutions are propagated through three banks which consist of two convolutional stages. The output feature maps of these three banks are concatenated and fed into a fully-connected network containing two linear stages. The final outputs of this network are 21 heat-maps with 18x18 pixels, of which the intensity indicates the confidence of a joint locating in the 2D position on a specific view.

3.2. Multi-view Fusion

The objective for multi-view fusion is to estimate the 3D hand joint locations from three views' heat-maps. Let ϕ_{kx}, ϕ_{ky} and ϕ_{kz} denote the x, y and z coordinates of the 3D hand joint location ϕ_k in the OBB coordinate system.

The CNNs generate a set of heat-maps for each joint, each view. Since the intensity on a heat-map indicates the confidence of a joint locating in the 2D position of the x-y, y-z or z-x view, we can get the corresponding probabilities $P(\phi_{kx}, \phi_{ky}| I_{xy})$, $P(\phi_{ky}, \phi_{kz}| I_{yz})$, and $P(\phi_{kz}, \phi_{kx}| I_{zx})$ from three views' heat-maps.

Assuming that, conditioned on the x-y view, the distribution of z variable is uniform, we have:

$$\begin{aligned} P(\phi_k| I_{xy}) &= P(\phi_{kx}, \phi_{ky}, \phi_{kz}| I_{xy}) \\ &= P(\phi_{kx}, \phi_{ky}| I_{xy}) P(\phi_{kz}| I_{xy}) \\ &\propto P(\phi_{kx}, \phi_{ky}| I_{xy}) \end{aligned} \tag{2}$$

With similar assumptions, for the other two views, it can be derived that $P(\phi_k| I_{yz}) \propto P(\phi_{ky}, \phi_{kz}| I_{yz})$ and $P(\phi_k| I_{zx}) \propto P(\phi_{kz}, \phi_{kx}| I_{zx})$.

We assume that the hand joint locations are independent conditioned on each view's projected image. Thus, the optimization problem in Eq. 1 can be transformed into:

$$\begin{aligned} \mathbf{\Phi}^* &= \arg\max_{\mathbf{\Phi}} P(\mathbf{\Phi}| I_{xy}) P(\mathbf{\Phi}| I_{yz}) P(\mathbf{\Phi}| I_{zx}) \\ &= \arg\max_{\mathbf{\Phi}} \prod_k P(\phi_k| I_{xy}) P(\phi_k| I_{yz}) P(\phi_k| I_{zx}) \\ &= \arg\max_{\mathbf{\Phi}} \prod_k Q(\phi_{kx}, \phi_{ky}, \phi_{kz}) \end{aligned} \tag{3}$$

where $Q(\phi_{kx}, \phi_{ky}, \phi_{kz})$ denotes the product of probabilities $P(\phi_{kx}, \phi_{ky}| I_{xy})$, $P(\phi_{ky}, \phi_{kz}| I_{yz})$, and $P(\phi_{kz}, \phi_{kx}| I_{zx})$ for each joint.

Eq. 3 indicates that we can get the optimal hand joint locations by maximizing the product of $Q(\phi_{kx}, \phi_{ky}, \phi_{kz})$ for all the joints which can be calculated from the intensities of three views' heat-maps. In this work, a set of 3D points in the bounding box is uniformly sampled and projected onto three views to get its corresponding heat-map intensities.

Then the value of $Q\left(\phi_{kx}, \phi_{ky}, \phi_{kz}\right)$ for a 3D point can be computed.

For simplicity of this problem, the product of probabilities $Q\left(\phi_{kx}, \phi_{ky}, \phi_{kz}\right)$ is approximated as a 3D Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, where $\boldsymbol{\mu}_k$ is the mean vector, $\boldsymbol{\Sigma}_k$ is the covariance matrix. These parameters of the Gaussian distribution can be estimated from the sampled data.

Based on above assumptions and derivations, the optimization problem in Eq. 3 can be approximated as follow:

$$\begin{aligned} \boldsymbol{\Phi}^* &= \arg\max_{\boldsymbol{\Phi}} \sum_k \log Q\left(\phi_{kx}, \phi_{ky}, \phi_{kz}\right) \\ &= \arg\max_{\boldsymbol{\Phi}} \sum_k \log \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \\ &= \arg\min_{\boldsymbol{\Phi}} \sum_k \left(\boldsymbol{\phi}_k - \boldsymbol{\mu}_k\right)^T \boldsymbol{\Sigma}_k^{-1} \left(\boldsymbol{\phi}_k - \boldsymbol{\mu}_k\right) \\ s.t.\ \boldsymbol{\Phi} &= \sum\nolimits_{m=1}^{M} \alpha_m \boldsymbol{e}_m + \boldsymbol{u} \end{aligned} \tag{4}$$

where $\boldsymbol{\Phi}$ is constrained to take the linear from. In order to learn the low dimensional subspace $\boldsymbol{\Omega}$ of hand configuration constrains, PCA is performed on joint locations in the training dataset [12]. $\boldsymbol{E} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \cdots, \boldsymbol{e}_M]$ are the principal components, $\boldsymbol{\alpha} = [\alpha_1, \alpha_2, \cdots, \alpha_M]^T$ are the coefficients of the principal components, $\boldsymbol{u}$ is the empirical mean vector, and $M \ll 3 \times K$.

As proved in the supplementary material, given the linear constrains of $\boldsymbol{\Phi}$, the optimal coefficient vector $\boldsymbol{\alpha}^* = [\alpha_1^*, \alpha_2^*, \cdots, \alpha_M^*]^T$ is:

$$\boldsymbol{\alpha}^* = \mathbf{A}^{-1}\boldsymbol{b} \tag{5}$$

where $\mathbf{A}$ is a $M \times M$ symmetric matrix, $\boldsymbol{b}$ is an M-dimensional column vector:

$$\mathbf{A}_{ij} = \sum_k \boldsymbol{e}_{j,k}^T \boldsymbol{\Sigma}_k^{-1} \boldsymbol{e}_{i,k},\ \boldsymbol{b}_i = \sum_k \left(\boldsymbol{\mu}_k - \boldsymbol{u}_k\right)^T \boldsymbol{\Sigma}_k^{-1} \boldsymbol{e}_{i,k}$$

$\boldsymbol{e}_i = \left[\boldsymbol{e}_{i,1}^T, \boldsymbol{e}_{i,2}^T, \cdots, \boldsymbol{e}_{i,K}^T\right]^T$; $\boldsymbol{u} = \left[\boldsymbol{u}_1^T, \boldsymbol{u}_2^T, \cdots, \boldsymbol{u}_K^T\right]^T$; $i,\ j = 1,\ 2, \cdots,\ M$.

The optimal joint locations $\boldsymbol{\Phi}^*$ are reconstructed by back-projecting the optimal coefficients $\boldsymbol{\alpha}^*$ in the subspace $\boldsymbol{\Omega}$ to the original joint space $\boldsymbol{\Lambda}$:

$$\boldsymbol{\Phi}^* = \sum\nolimits_{m=1}^{M} \alpha_m^* \boldsymbol{e}_m + \boldsymbol{u} \tag{6}$$

To sum up, the proposed multi-view fusing method consists of two main steps. The first step is to estimate the parameters of Gaussian distribution for each joint using the three views' heat-maps. The second step is to calculate the optimal coefficients $\boldsymbol{\alpha}^*$ and reconstruct the optimal joint locations $\boldsymbol{\Phi}^*$. The principal components and the empirical mean vector of hand joint configuration are obtained by applying PCA on training data during the training stage.

4. Experiments

4.1. CNNs Training

The CNNs of multiple views described in Section 3.1 were implemented within the Torch7 [2] framework. The optimization algorithm applied in CNNs training process is stochastic gradient descent (SGD) with a mean squared error (MSE) loss function, since the task of hand pose estimation is a typical regression problem. For training parameters, we choose the batch size as 64, the learning rate as 0.2, the momentum as 0.9 and the weight decay as 0.0005. Training is stopped after 50 epochs to prevent overfitting. We use a workstation with two Intel Xeon processors, 64GB of RAM and two Nvidia Tesla K20 GPUs for CNNs training. The CNNs of three views can be trained at the same time since they are in parallel. Training the CNNs takes approximately 12 hours.

4.2. Dataset and Evaluation Metric

We conduct a self-comparison and a comparison with state-of-the-art methods on the dataset released in [21], which is the most challenging hand pose dataset in the literature. This dataset contains 9 subjects and each subject contains 17 gestures. In the experiment, we use 8 subjects as the training set for CNNs training and the remaining subject as the testing set. This is repeated 9 times for all subjects.

In addition, we conduct a cross-dataset evaluation by using the training data from the dataset in [21] and the testing data from another dataset in [17].

We employ two metrics to evaluate the regression performance. The first metric is the mean error distance for each joint across all the test samples, which is a standard evaluation metric. The second metric is the proportion of good test samples in the entire test samples. A test sample is regarded as good only when all the estimated joint locations are within a maximum allowed distance from the ground truth, namely the error tolerance. This worst case accuracy proposed in [26] is very strict.

4.3. Self-comparisons

For self-comparison, we implement two baselines: the single view regression approach and the multi-view regression approach using a coarse fusion method. In the single view regression approach, only the projected images on OBB coordinate system's *x*-*y* plane are fed into the CNNs. From the output heat-maps, we can only estimate the *x* and *y* coordinates of joint locations by using the Gaussian fitting method proposed in [29]. The *z* coordinate can be estimated from the intensity of the projected image. If the 2D point with the estimated *x*, *y* coordinates is on the background of the projected image, the *z* coordinate will be specified as zero in OBB coordinate system instead of the maximum depth value, which can reduce the estimation errors on *z* di-

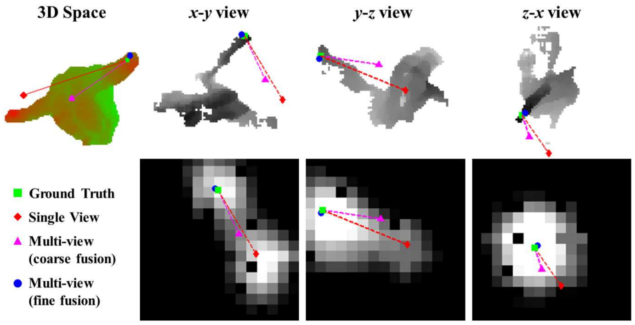


Figure 6: An experimental example for self-comparison. **Top-left**: 3D point cloud with ground truth and estimated 3D locations. **Top-right**: Projection images in three views. **Bottom-right**: Heat-maps of three views. The ground truth and estimated 3D locations are back-projected onto three views and their heat-maps for comparison. Lines indicate the offsets between ground truth and estimations.

rection. The multi-view regression approach using a coarse fusion method can be considered as a degenerated variant of our fine fusion method. This method estimates the 3D hand joint locations by simply averaging the estimated *x*, *y* and *z* coordinates from three views' heat-maps.

We compare the accuracy performance of these two approaches with the multi-view fine fusion method described in Section 3. The mean error for each joint and the worst case accuracy of these three methods are shown in Fig. 5 (left and middle) respectively. As can be seen, the multi-view regression is effective since our two multi-view regression approaches significantly outperform the single view regression method. In addition, the fine fusion method is better than the coarse fusion method when considering the mean error performance, which is about 13 mm on the dataset in [21]. When considering the worst case accuracy, the fine fusion method performs worse than the coarse fusion method only when the error tolerance is large. However, the high accuracy corresponding to small values of error tolerance should be more favorable, because the large values of error tolerance indicate that imprecise estimations will be considered as good test samples. Thus, the fine fusion method is overall better than the coarse fusion method and we apply this fusion method in the following experiments.

Fig. 6 shows an example of the ambiguous situation where the index fingertip is very likely to be confused with the little fingertip. As can be seen, the single view regression method only utilizes the *x*-*y* view's heat-map which contains two hotspots and gives an estimation with large error distance to the ground truth. However, the multi-view fine fusion method fuses the heat-maps of three views and estimates the 3D location with high accuracy. The multi-view coarse fusion method gives an estimation in between the results of the above two methods due to its underutilization of heat-maps' information. Fig. 9 shows qualitative results of these three methods on several challenging examples to further illustrate the superiority of the multi-view fine fusion method over the other two methods.

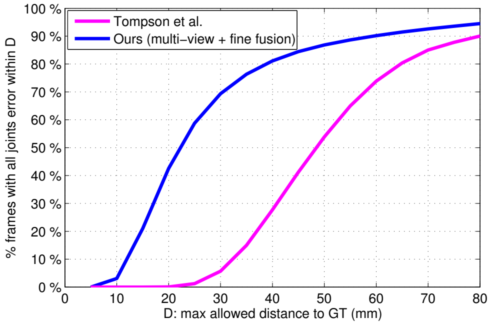


Figure 7: Comparison with the approach proposed in [29]. In this method, 14 hand joints are estimated. For fair comparison, in our method, 14 corresponding joints of 21 estimated joints are used to calculate the worst case accuracy.

In addition, we study the impact of different number of principal components used in joint constraints on the worst case accuracy under different error tolerances, as shown in Fig. 5 (right). It is reasonable to use 35 principal components in joint constraints considering the worst case accuracy. We use this setting in all the other experiments.

4.4. Comparison with State-of-the-art

We compare our multi-view fine fusion method with two state-of-the-art methods on the dataset in [21]. The first method is the CNNs based hand pose estimation proposed in [29]. The second method is the random forest based hierarchical hand pose regression proposed in [21].

The method in [29] requires a model fitting process to correct large estimation errors. Since the dataset in [21] does not release the hand parameters for each subject, we conduct model fitting with an uncalibrated hand model and set the hand size and finger lengths as the variables in optimization. In our implementation, this method estimates 14 hand joint locations which are a subset of the 21 hand joints used in our method. For fair comparison, we calculate the worst case accuracy of the 14 corresponding joints from the 21 joints estimated by our method. As shown in Fig. 7, our multi-view regression with fine fusion method significantly outperforms the method in [29] for the worst case accuracy. Essentially, the method in [29] is a single view regression method which only uses the depth image as the input of the networks. This result further indicates the benefit of using multi-view's information for CNN-based 3D hand pose estimation. Even though an accurately calibrated hand model

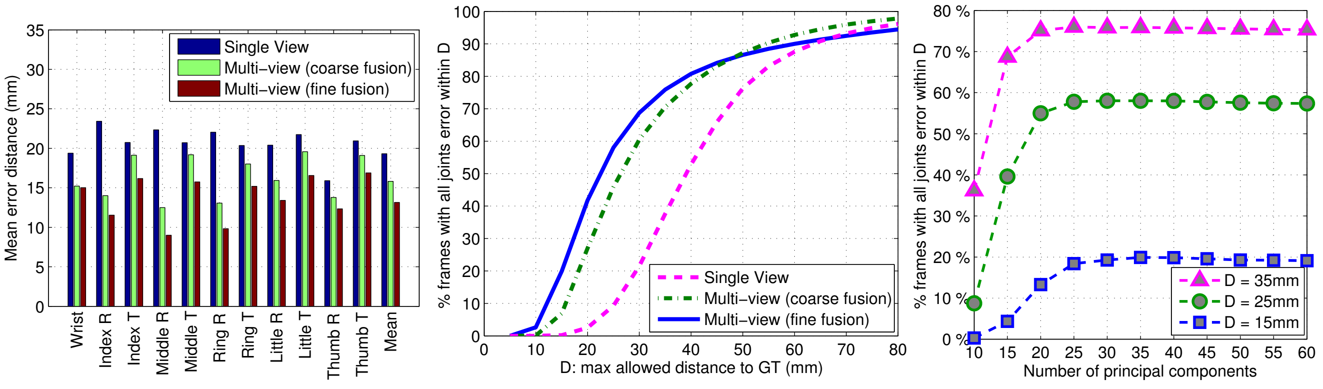


Figure 5: Self-comparison of different methods on the dataset in [21]. **Left**: the mean error distance for each joint across all the test samples (R:root, T:tip). **Middle**: the proportion of good test samples in the entire test samples over different error tolerances. **Right**: The impact of different number of principal components used in joint constraints on accuracy performance.

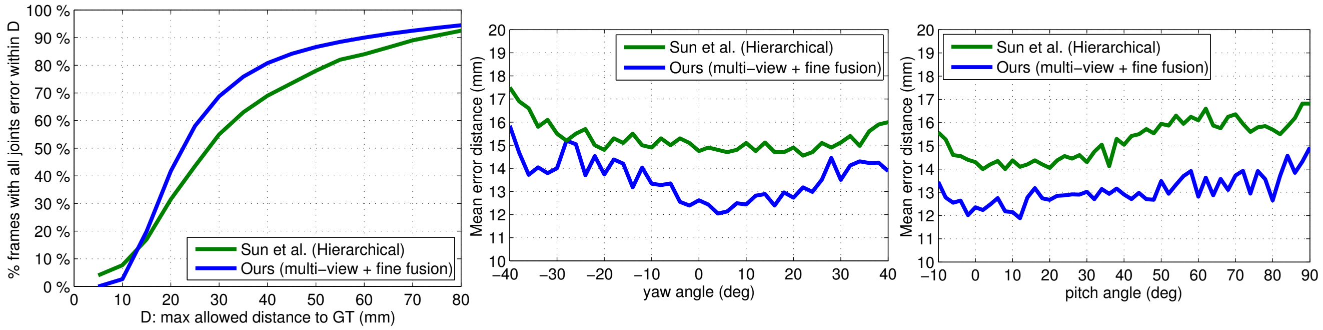


Figure 8: Comparison with the approach proposed in [21]. **Left**: the proportion of good test samples in the entire test samples over different error tolerances. **Middle & right**: the mean error distance over different yaw and pitch angles of the viewpoint. Our method holds smaller average errors in all of the yaw and pitch angles. The curves of the hierarchical regression method are cropped from the results reported in [21].

may improve the accuracy of the method in [29] in a limited degree, it is cumbersome to calibrate the hand model for every subject and the model fitting process will increase the computational complexity.

We compare with the hierarchical regression method proposed in [21]. Note that this method has been presented superior than the methods in [20, 24, 32]. Thus, we indirectly compare our method with the methods in [20, 24, 32]. As can be seen in Fig. 8, our method is superior than the method in [21]. The worst case accuracy of our method is better than the method in [21] over most error tolerances, as shown in Fig. 8 (left). Especially, when the error tolerances are 20mm and 30 mm, the good sample proportions of our method are about 10% and 15% higher than those of the method in [21]. When the error tolerance is smaller than 15mm, the good sample proportion of our method is slightly lower than that of the method in [21]. This may be caused by the relatively low resolution of the heat-maps used in our method. We also compare the average estimation errors over different viewpoint angles of these two methods. As shown in Fig. 8 (middle and right), the average errors of our method are smaller than those of the method in [21] over all yaw and pitch viewpoint angles. In addition, our method is more robust to the pitch angle variation with a smaller standard deviation (0.64mm) than the method in [21] (0.79mm).

The runtime of the entire pipeline is 14.1ms, including 2.6ms for multi-view projection, 6.8ms for CNNs forward propagation and 4.7ms for multi-view fusion. Thus, our method runs in real-time at over 70fps. Note that the process of multi-view projection and multi-view fusion is performed on CPU without parallelism, and the process of CNNs forward propagation is performed on GPU with parallelism for three views.

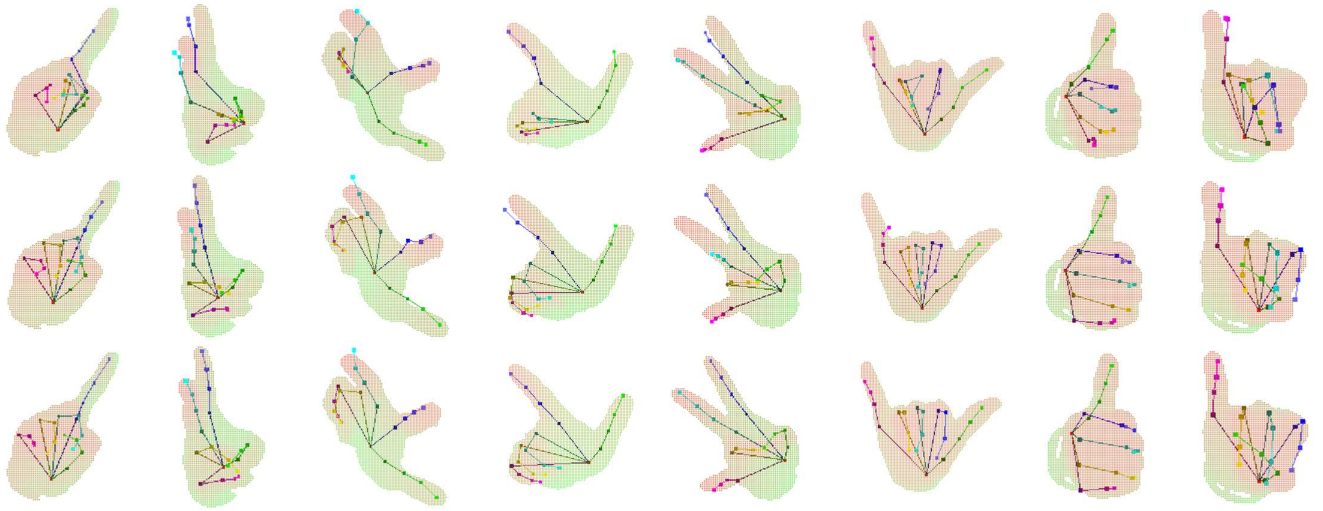


Figure 9: Qualitative results for dataset in [21] of three approaches: single view regression (in the first line), our multi-view regression with coarse fusion (in the second line) and our multi-view regression with fine fusion (in the third line). We show the estimated hand joint locations on the depth image. Different hand joints and bones are visualized using different colors. This image is best viewed in color.

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
ICP-PSO*	8.6	7.4	9.8	10.4	7.8	11.7	9.2
Ours	30.1	19.7	24.3	19.9	21.8	20.7	22.8

Table 1: Average estimation errors (in *mm*) of 6 subjects for 6 methods tested on the dataset in [17].

4.5. Cross-dataset Experiment

In order to verify the generalization ability of our CNN based multi-view regression method, we perform a cross-dataset experiment. We attempt to adapt the existing CNN based regressors learned from the source dataset in [21] to a new target dataset in [17].

In this experiment, we train the CNNs on all the 9 subjects of the dataset in [21]. The CNNs are directly used for hand pose estimation on all the 6 subjects of the dataset in [17] by using our proposed method. According to the evaluation metric in [17], we calculate the average errors for the wrist and the five fingertips. We compare our method with model based tracking methods reported in [17], which are FORTH [15], PSO [17], ICP [16], ICP-PSO [17] and ICP-PSO* (ICP-PSO with finger based initialization) [17].

According to [17], these model-based tracking methods need an accurate hand model that is calibrated to the size of each subject, and they rely on temporal information. Particularly, to start tracking, these methods use ground truth information to initialize the first frame. However, our method does not use such information and thus is more flexible in real scenarios and robust to tracking failure. Under such situation, our method still outperforms FORTH, PSO and ICP methods, as shown in Table 1, which indicates that our method has good ability of generalization. It is not surprising that our method is worse than ICP-PSO and ICP-PSO*, because we do not use calibrated hand model or any ground truth information or temporal information and we perform this experiment on cross-dataset which is more challenging.

5. Conclusion

In this paper, we presented a novel 3D hand pose regression method using multi-view CNNs. We generated a set of heat-maps of multiple views from the multi-view CNNs and fused them together to estimate 3D hand joint locations. Our multi-view approach can better leverage the 3D information in one depth image to generate accurate estimations of 3D locations. Experimental results showed that our method achieved superior performance for 3D hand pose estimation in real-time.

Acknowledgment: This research, which is carried out at BeingThere Centre, is supported by Singapore National Research Foundation under its International Research Centre @ Singapore Funding Initiative and administered by the IDM Programme Office.

References

[1] R. K. Ando and T. Zhang. Two-view feature generation model for semi-supervised learning. In *ICML*, 2007.

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

[3] P. Dollr, P. Welinder, and P. Perona. Cascaded pose regression. In *CVPR*, 2010.

[4] H. Guan, R. S. Feris, and M. Turk. The isometric self-organizing map for 3d hand pose estimation. In *FGR*, 2006.

[5] A. Jain, J. Tompson, M. Andriluka, G. Taylor, and C. Bregler. Learning human pose estimation features with convolutional networks. In *ICLR*, 2014.

[6] K. Jarrett, K. Kavukcuoglu, M. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *ICCV*, 2009.

[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] F. Kirac, Y. E. Kara, and L. Akarun. Hierarchically constrained 3d hand pose estimation using regression forests from single frame depth data. *Pattern Recognition Letters*, 50:91–100, 2014.

[9] W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *CVPR Workshops*, 2010.

[10] H. Liang, J. Wang, Q. Sun, Y. Liu, J. Yuan, J. Luo, and Y. He. Barehanded music: real-time hand interaction for virtual piano. In *ACM SIGGRAPH I3D*, 2016.

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

[12] 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.

[13] H. Liang, J. Yuan, D. Thalmann, and N. M. Thalmann. Ar in hand: Egocentric palm pose tracking and gesture recognition for augmented reality applications. In *ACM-MM*, 2015.

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

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

[16] S. Pellegrini, K. Schindler, , and D. Nardi. A generalization of the icp algorithm for articulated bodies. In *BMVC*, 2008.

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

[18] Z. Ren, J. Yuan, J. Meng, and Z. Zhang. Robust part-based hand gesture recognition using kinect sensor. *IEEE Transactions on Multimedia*, 15(5):1110–1120, 2013.

[19] M. Schrder, J. Maycock, H. Ritter, and M. Botsch. Real-time hand tracking using synergistic inverse kinematics. In *ICRA*, 2014.

[20] 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.

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

[22] J. S. Supancic III, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan. Depth-based hand pose estimation: methods, data, and challenges. In *ICCV*, 2015.

[23] 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.

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

[25] D. Tang, T. H. Yu, and T. K. Kim. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *ICCV*, 2013.

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

[27] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *CVPR*, 2015.

[28] J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, 2014.

[29] 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.

[30] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, 2014.

[31] J. M. Van Verth and L. M. Bishop. *Essential mathematics for games and interactive applications*. CRC Press, 2015.

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

[33] J. Xu, J. Yuan, and Y. Wu. Multimodal partial estimates fusion. In *ICCV*, 2009.