Hand Pose Learning: Combining Deep Learning and Hierarchical Refinement for 3D Hand Pose Estimation

Min-Yu Wu
r04922105@csie.ntu.edu.tw
Ya Hui Tang
d05922027colin@csie.ntu.edu.tw
Pai-Wen Ting
Ck980046@gmail.com
Li-Chen Fu
lichen@ntu.edu.tw

Department of Computer Science
and Information Engineering
National Taiwan University
Taiwan, ROC

Abstract

Hand Pose Estimation aims to predict the position of joints on a hand from an image. This problem is pretty challenging since a hand can perform a variety of poses and tends to cause self-occlusion easily. This paper proposes a hybrid method of training a deep learning model and hierarchical refinement for hand pose estimation in a 3D space using depth images. First, we design a so-called skeleton-difference layer that can allow a convolutional neural network (CNN) training process to effectively learn the shape as well as physical constraints of a hand. Secondly, we employ a refinement method that is capable of hierarchically regressing a hand pose with an energy function. In the experiments we have conducted, the results validate the robustness and the performance of our system, and show that our method is able to predict the joints more accurately. Such appealing results may be attributed to consideration of physical joint relationship, which in turn makes the estimated hand poses quite natural and complete.

Introduction

Hand pose estimation is a specific topic that looks for a good method to extract proper hand poses from a certain input source. Its increasing importance lies in the possibility to naturally convey expressions from a human to a machine, and hence a variety of applications for the future life can be realized through this consideration. As plenty of innovative techniques such as Virtual Reality (VR), Augmented Reality (AR) and Human-Computer Interaction (HCI) have gained popularity in recent years, the trend gives rise to the emergence of several powerful devices in the VR/AR market, via which the interaction between human hands and the virtual objects becomes possible. In the light of the potential demand as well as the rapid development on the topic, there have been lots of researches done on this topic. Meanwhile, as the depth camera was invented recently, 3D geometry in a space is also accessible from these sensors [1, 2]. Therefore, hand pose estimation comes along to be flourishing with these technologies.

Even many researchers have paid much efforts on the intriguing topic, hand pose estimation is still a challenging problem currently for computer vision since a human hand

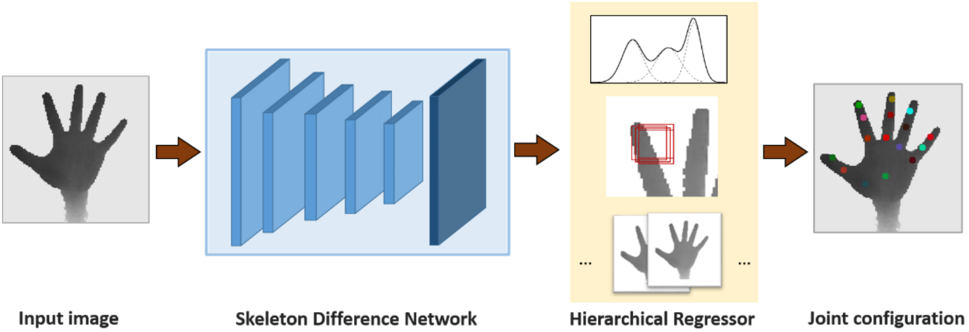


Figure 1: The flowchart of our system that shows how to predict joint positions from an input depth image. In the hierarchical regressor, there are three energy functions employed to optimize the joint configurations.

can perform countless kinds of poses, and hands tend to move with different simultaneous rotations and bends [3]. Some problems such as self-occlusion and large motions on an image take place consequently. In pursuit of a good quality of hand pose estimation, it is necessary to overcome the obstacles. On the other hand, deep learning approach such as convolution neural network (CNN) has demonstrated the excellent performance in many tough cases in the computer vision field [4]. The popular data-driven approach is known for the ability to extract some implicit features from input images. This advantage implies that we don’t need to design a particular hand-craft feature or rely on a specific algorithm which may fail when they are faced with poor image conditions like occlusions and noises. It means that we can be exempt from handling those troublesome cases simply because the properties of the defect information can probably be learned from sufficient amounts of data.

In this paper, we will focus on how to improve the performance of hand pose estimation. We propose a novel architecture based on convolutional neural network using a depth image sequence of consecutive hand poses as input, and then prediction the 3D position of the defined joints on a hand. Figure 1 is an overview of our hand pose estimation system. Our system can be decomposed into two parts. The first part describes the way how we train our skeleton-difference network (SDNet), which is composed of conventional joint-position-loss layer and our proposed skeleton-difference layer. The latter is an additional loss layer that considers physical constraints of a hand pose, keeping the joint relations while training. The second part is the refinement step after we get a primary estimation from the SDNet. We employ an energy function to refine the poses, computing the energy from depth appearance, physical formation and temporal motion constraints. We take a hierarchical optimization strategy to refine a joint from the palm to fingertips one by one with the energy function.

The rest of the paper is organized as follows. In Section 2, we give a simple overview of the previous related methods in this field. In Section 3, we introduce the proposed architecture we have employed to train the model. In Section 4, we introduce the method how to estimate the pose with our model combined with our refinement technique. In Section 5, the experimental results of this method will be shown and some discussions about the results will be made as well. In Section 6, the conclusions of this paper will be drawn.

Related Work

The development of hand pose estimation has been pretty prosperous recently. In this section, we will briefly introduce the relevant works that dedicate to this field. Since a great deal of works in hand pose estimation use hybrid way of multiple approaches, it is not easy for us to decisively categorize these papers into different types, so we talk about the tendency of the ways in which they solve the hand pose estimation problem in the following subsections.

2.1 Hand Pose Estimation using CNN

Advance of theories about convolutional neural network (CNN) has been quite remarkable since the Krizhevsky et *al.* [5] presented a novel network that dramatically raises the accuracy of object classification a few years ago. Afterwards, with diverse designs of network architecture, this approach has also been proved to dramatically raise the performance in a variety of tasks such as object detection [4, 6], segmentation [7], activity recognition [8]. Likewise, some researchers start to employ CNNs in pursuit of better hand pose estimation result. In [9], they create heat maps to make inverse kinematics pose recovery. In the process of recovering depth image, they exploit a CNN for dense feature extraction. Oberweger et *al.*[10] propose feedback loop architecture of a CNN to estimate the 3D pose and show an outstanding result with an excellent efficiency. It is an entirely data-driven approach, whose main core is to iteratively correct the mistakes in each iteration. In [11], the authors also introduce an architecture that is composed of multiple CNNs that refine each joint by cropped region of multi-scales and iteratively approach the best prediction. [12] present a multi-stream CNN coping with the multi-view problem of depth image and fuse the heat map from different views to estimate the positions of the joints. Also, Sinha et *al.* [13] propose a two-stage CNN. In the first stage, they exploit a CNN to globally estimate the primary hand pose. [14] share the convolutional layers for hand detection and hand pose estimation. They next divide the pose into small parts and each part has one local hand pose regressor. They also experiment their method on a public dataset and another synthetic data. Ye et *al.* [15] design a multi-cascaded-CNNs structure and predict joints from the palm to the fingertips layer by layer. Therefore, many works have confirmed the power of the CNN for this regression problem

2.2 Hierarchical Hand Pose Estimation Approach

Since a hand is obviously partitioned and the relationship of joints in a part is strong, it is much easier to predict joints one by one than finding a global solution at one time. This concept is widely taken in human pose estimation due to its effectiveness. Therefore, hierarchical regression is a common scheme for hand pose estimation, and some papers

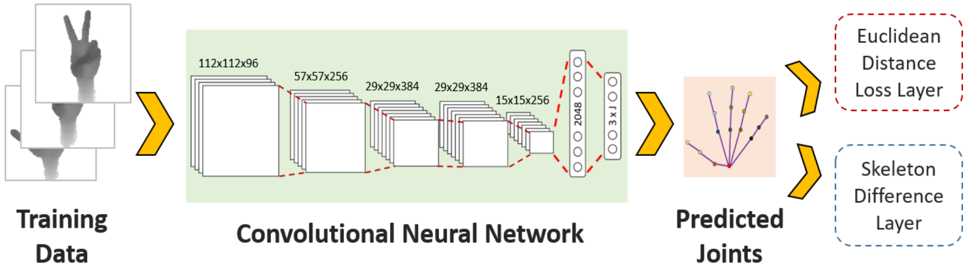
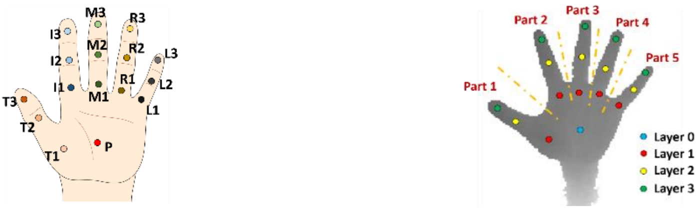


Figure 2: The proposed training approach. Our input image size is 224 by 224, and the images will be processed by a CNN which has 5 convolutional layers and 2 fully connected layers. After we obtain predicted joints, they will be put into the two loss layers for backpropagation.



(a) (b)

Figure 3: (a) The defined 16 joints in ICVL dataset [19], and there are three joints in each fingers. (b) The part categories and layers we use in our skeleton-difference layer and hierarchical regressor. Each finger represents one part and has three joints. Layer 1 to layer 3 contains the joints from the palm to the fingertips, and the P joint (Palm) is layer 0 and it belongs to each part. (Best viewed in color)

combine the hierarchy property with the model-based approach (e.g. Predict the palm first, then predict fingers.) to reach better performance. Sun et *al.* [16] propose a cascaded scheme to train multiple regressors for palm and finger, and the performance is quite excellent in their experimental results. Tang et *al.* [17] present the Latent Regression Forest that predict starting from the root (i.e. the palm joint) to the leafs (i.e. the fingertips) and optimize each result in every step. The same group introduce a hierarchical sampling optimization approach in [18], which iteratively optimize the hand configurations in each layer to obtain the final pose. Some works also combine CNN with hierarchical strategy such as the aforementioned articles [13, 15], and the authors show that this idea can work very well.

Training a CNN with Skeleton-Difference Layer

In this section, we will exhibit the approach to train our skeleton-difference network (SDNet) for hand pose estimation. Our training architecture is mainly based on the CNN. Figure 2 illustrates every component in our training process, which can be divided into two parts, where the first one is the normal joint prediction CNN, while the second one is the skeleton-difference layer. We will elaborate the two parts by turn in the following text.

3.1 Joint prediction CNN

For CNN, we fundamentally employ ZF-net [20] and make some minor modification to adapt to problem better. We adopt the convolutional layers of [20] while utilizing one fully connected layer of 2048 neurons since the depth images have one channels only and it might not take too many parameters. To prevent the network from overfitting, a dropout layer of ratio 0.5 is applied to this training process. The CNNs help to extract the features from an input depth image, and after 5 convolutional layers, the features will be passed down to one layer of the fully connected layer to retrieve the relationship between the features. At the end, the last layer will output the position of each joint, which is supposed to include 3 × J neurons indicating the 3D position ($i.e.$ (x, y, z)) of J joints on a hand. For example, in our experiments, J is 16 as defined in [19]. Then, we can use the predicted joint positions to compute the difference against the ground truth. We use the Euclidean distance as the loss function for predicting joints in our network, and it is defined as follows:

$$\psi_D(\boldsymbol{X}_j) = \left\|\boldsymbol{X}_j - \boldsymbol{X}_j^{GT}\right\|_2 \tag{1}$$

here, $\boldsymbol{X}_j$ is the predicted position of joint j and $\boldsymbol{X}_j^{GT}$ is the ground truth. This Euclidean loss term directly reflects the spatial prediction result according to the appearance in depth images, and with it, we expect to train the whole network until convergence.

3.2 Skeleton-Difference Layer

In Section 3.1, only the direct depth appearance from the view of the camera is taken into consideration during the course of prediction, but there are supposed to be some physical constraints that are likely to remedy the difficulty of predicting the high-dimensional joints. Therefore, to form these physical constraints in the training stage, we design a novel loss layer, namely skeleton-difference layer, to model the feasibility of a hand pose.

Given a hand pose, we can view the connection between a joint and another as a bone, and all the bones will form the skeleton of the hand. The structure of the defined skeleton is illustrated in Figure 3a. The term ‘joints’ here refer to the defined 16 joints in our conducted experiment, so the palm is regarded as a joint as well, and the other joints are situated on respective five fingers. Therefore, the whole structure will be a tree structure with the palm as the root, and each edge stands for a bone. In the skeleton-difference layer, we compute the loss, namely skeleton-difference loss, throughout every bone in the structure. To represent the spatial relationship better, we transform the bones into vectors, and the loss is defined as follows:

$$\psi_{SDL} = \alpha Loss_a + \beta Loss_b \tag{2}$$

The equation includes two terms which indicate the angle loss term and the bone loss term, respectively, while α and β are the weights of two terms, respectively. The first term represents the angle loss, explaining the angle between two bones, and consequently it is derived as:

$$Loss_a = \sum_{p=1}^{P}\sum_{l=0}^{L}\left(\omega_1\left|\tan(\frac{\theta_{p,l}}{2} - \frac{\theta_{p,l}^{GT}}{2})\right| + \omega_2 F(\theta_{p,l})\right) \tag{3}$$

where

$$\theta_{m,n} = \cos^{-1}(\frac{\boldsymbol{B}_{m,n}\cdot\boldsymbol{B}_{m,n+1}}{\|\boldsymbol{B}_{m,n}\|\|\boldsymbol{B}_{m,n+1}\|}) \tag{4}$$

$$F(\theta) = \begin{cases} 1, & \theta \geq threshold \\ 0, & else \end{cases} \tag{5}$$

Here, P is the part number out of 5 fingers on a hand, and L is the angle number depending on how many angles we have on a finger. $\boldsymbol{B}_i$ is the vector between two joints (*i.e.* $joint_i$ and $joint_{i+1}$) in each part, and θ_i is a angle between the two vectors $\boldsymbol{B}_i$ and $\boldsymbol{B}_{i+1}$ while ω_k is a weight. $F(\theta)$ is a function aiming to measure the feasibility of the estimated angle, and if an angle is greater than a threshold, it will suffer an additional penalty to hold the natural bend of a hand finger. The other loss term of this layer is the bone length loss, which can be defined as below:

$$Loss_b = \sum_{p=1}^{P}\sum_{l=0}^{L}\|G_{p,l} - G_{p,l}^{GT}\| \tag{6}$$

where

$$G_{m,n} = (\frac{\left\|\boldsymbol{B}_{m,n+1}\right\|}{\left\|\boldsymbol{B}_{m,n}\right\| + \left\|\boldsymbol{B}_{m,n+1}\right\|}) \tag{7}$$

The skeleton-difference layer enforces the relations between joints to be maintained during training. Furthermore, the loss is computed from five finger parts, which implies that skeleton-difference layer not only can impose physical constraints on a hand pose but also implicitly models five fingers. When training a hand on the basis of hand parts, even if some hand information may be missing in images of poor conditions like broken pixels or occlusions, the model will still try to keep the shape of the part in the space and then compensate for insufficient depth data, authentically learning the formation of a hand better. The advantage of the training architecture we design here is that we can simultaneously utilize both joint relation properties and depth appearance with the combination of the two layers mentioned above in the CNN, and the whole network becomes our skeleton-difference network (SDNet). Interestingly, we found that in the early stage, the Joint prediction loss term has more impact, but afterwards the skeleton-difference layer will take over the loss term in the last stage of fine-tune process. Thus, we can eventually obtain a model for accurate hand pose estimation after the training process.

Estimate Joints with Hierarchical Regression

In this section, we will elaborate the way how to predict the joint configuration. Using an input depth image, we put it into our SDNet as introduced in Section 3, and we will get the primarily estimated position of each joint. Then, to have the points minor regressed to the better position, we propose an energy function to achieve the further refinement that is too minor for CNN to precisely predict. Here, we adopt a hierarchical scheme to regress the joints from palm to fingertips one by one, whose structure has been shown in Figure 3b. Inspired by [21], we design the energy function to refine the slight dislocation, and it also provides a good extension like in [22], where we can make characterized optimization for a certain layer of joints, and it can be conducted in future application like hand-object interaction. The prediction process of our system consists of two part, which are a convolutional neural network and a joint regressor, respectively. Here, we explain in details about the regressor, and the objective function of optimizing a certain joint $\boldsymbol{X}_j$ can be defined as following:

$$\arg\min_{\boldsymbol{X}_j}(\varphi_G(\boldsymbol{X}_j) + \gamma_D\varphi_D(\boldsymbol{X}_j) + \gamma_M\varphi_M(\boldsymbol{X}_j)) \tag{8}$$

where $\varphi_G(\cdot)$, $\varphi_D(\cdot)$ and $\varphi_M(\cdot)$ are three energy terms. The first term is physical constrains of bone length, which is similar to the concept we introduce in Section 3.2. Here, our regression considers the rationality of a bone length, so we take advantage of Gaussian Mixture Model (GMM) because of its ability to evaluate a samples with a probability distribution model [23]. We train GMM with the ground truth to find the distribution of every bone length in a hand, then we can evaluate if the predicted bone length is out of the possible range, and it can be derived as below:

$$\varphi_G(\boldsymbol{X}_k) = -PDF(\|\boldsymbol{B}_k\|, \boldsymbol{GMM}_k) \tag{9}$$

where $\boldsymbol{X}_k$ is the 3D position of joint k, and $\boldsymbol{B}_k$ is the bone vector from joint k to joint $k+1$ in each part. PDF is the score function that is proportional to probability density function that gives a score while $\boldsymbol{GMM}_k$ is the Gaussian Mixture Model of the corresponding bone length. We train these GMM model via EM algorithm, with number of Gaussian is 5.

The second term here measures the 2D projection feasibility. If a predicted joint falls into the background area, it is likely to be dislocated. In implementation, we use a kernel that convolute with the area around the predicted joint. If the area contains more background pixels, it will receive more penalties because we hope the joint can be as centred in a finger as possible. 2D projection feasibility is defined as following:

$$\varphi_D(\boldsymbol{X}_k) = penalty_D \cdot \sum_{r=-w}^{W} \sum_{c=-w}^{W} (\mathbf{1} | I(\boldsymbol{X}_k, r, c) \in background)) \tag{10}$$

where W defines the kernel size, $I(\boldsymbol{X}_k, r, c)$ is the intensity value at the testing point.

The third term is a temporal constraint to model the finger movement. It indicates that the displacement of the leafs should be larger than the parent joints in a hierarchical tree structure of a hand. It makes sense because the movement is addictive, and the child joint moves basically according to its parent. The observation is more apparent for fingertip joints, where we impose this constraint. To model this relationship, we compute the relative length of the joints from the second and the third layer respect to the first layer. If a joint of third layer move less dramatically than its parent joint. This relation is derived as:

$$\varphi_M(\boldsymbol{X}_{p,l,t}) = penalty_M \cdot (\mathbf{1} | \boldsymbol{H}(\boldsymbol{X}_{p,l,t}) < \boldsymbol{H}(\boldsymbol{X}_{p,l-1,t}) \ and \ l = 3) \tag{11}$$

where

$$\boldsymbol{H}(\boldsymbol{X}_{p,l,t}) = \boldsymbol{r}(\boldsymbol{X}_{p,l,t}) - \boldsymbol{r}(\boldsymbol{X}_{p,l,t-1}) \tag{12}$$

$\boldsymbol{X}_{r,s,t}$ is the position of the joint at the layer r of part s in frame t, and $\boldsymbol{r}(\cdot)$ means the relative length for a joint in layer 2 or 3 respect to layer 1. Moreover, this energy term is valid for the fingertip point, for the relation is more obvious.

Experiments

5.1 Dataset

We use ICVL Hand Posture Dataset [19] in our experiments. The dataset has more than 24,000 depth images of hands captured by Intel Creative depth sensor, and there are about 22,000 images are for training (augmented ones excluded) and 1,596 testing images in two consecutive sequences. Each depth images contains one hand of different poses. Each hand has is annotated with the positions (*i.e.* (*x*, *y*, *z*)) of 16 joints, as we illustrated in Figure 3a. As for the training data we take to train our model, we use the augmented data provided by this dataset, whose frames are rotated every 22.5 degrees.

5.2 Evaluation

In this paper, we follow the two kinds of evaluations that are commonly used for testing joint positions. The first evaluation is the average Euclidean error between the estimated position and the annotated one, which intuitively reflects the prediction difference in Euclidean distance. The second evaluation is the fraction of success, which means the fraction of one hand whose Euclidean distance errors of all its joints are under the set threshold. Throughout

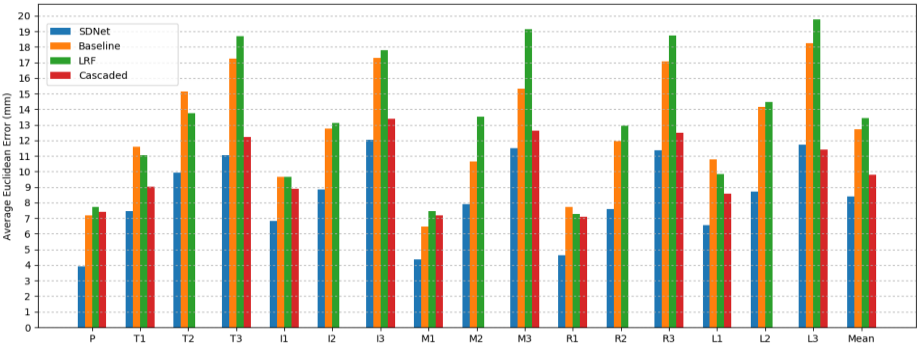


Figure 4: The average Euclidean Error of each joint. We compare our results with LRF [17], Cascaded [16] and our baseline that uses simple CNN architecture.

the difference threshold, we can observe how many poses are severely distorted, or on the contrary, to say how many poses are complete. If an error of one dislocated joint on a hand is over the threshold, the hand pose will be regarded as a failure, so these criterion is a rigorous one that emphasize the completeness of the whole estimation result. We get used to analysing the quality of a model with the two evaluations at the same time, which allows us to see the general performance and the extreme cases. Besides, we set a baseline for comparison, which is that we use only joint distance error as loss for CNN training and that skeleton-difference layer and hierarchical regression is neglected.

5.3 Results

For hand pose estimation, our experiment is conducted on a machine equipped with a GPU of NVIDIA GeForce GTX 980. For the implementation of CNN, we employ Caffe [24] tool. The runtime of our testing process is 7 ms in SDNet stage and 12 ms in refinement stage. We compare our experimental results with the previous state-of-the-art papers [16, 17] and the baseline as we describe in Section 3.2, and for evaluation of the average Euclidean error, the results are in Figure 4. Only the tip and the root joints of a finger part are provided in [16], so we show merely 11 joints of mean average error from their result. Our result performs better than the other two papers in all joints except the finger *L3* (*i.e.* Pinky finger

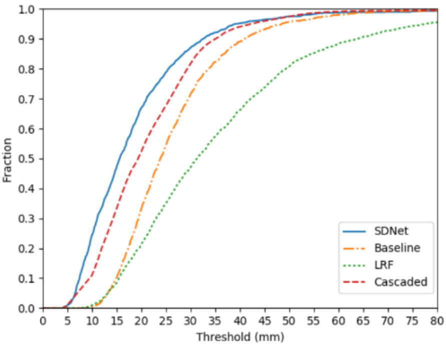


Figure 5: The successful pose estimation fraction of a full hand under the threshold compared with other papers [16, 17].

Network Settings (weight)	Mean Euclidean Distance Error (mm)
Baseline	12.6990
SDNet (0.125)	8.8209
SDNet (0.25)	8.7291
SDNet (0.5)	8.4002

Table 1. The comparison of setting different weights to skeleton-difference layer in our CNN training process.

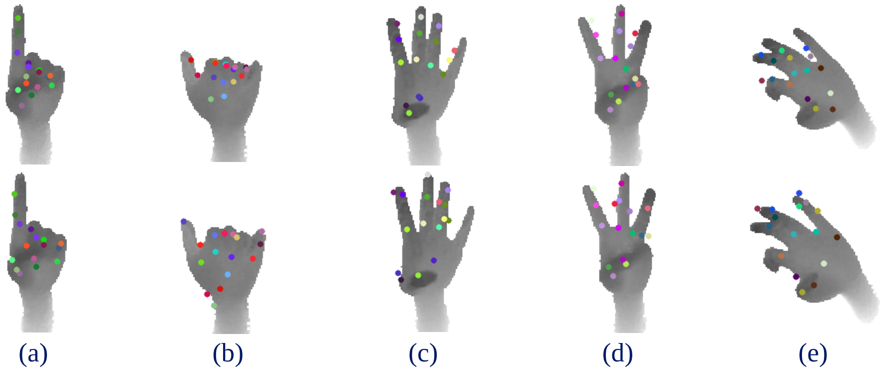


Figure 6: Some examples of our predicted results (upper row) and of [17] (lower row).

(a)-(c) Our estimated hand poses have better performance than other works even in some difficult cases such as severe occlusions, and the poses keep in good shape.

(d)-(e) Even in some challenging poses our prediction result may not be absolutely accurate, our method still makes a hand pose complete and natural.

tip), which is only a bit worse than [16], and it verifies that our overall prediction is reasonably more accurate. Besides, we dramatically improve the results of the tip fingers, which we put much effort especially in the hierarchical regression stage. Our hierarchical regressor improves 0.05~0.150 mm in average Euclidean errors. Though the increase is relative minor, it indeed refines the results of tip fingers in most cases, which shift very easily in every predicted result.

We also illustrate the prominent quality of estimation in Figure 5, which shows the comparison in the fraction under threshold evaluation. This evaluation illustrates that the higher the curve, the more complete a hand pose is estimated, and also, we have a remarkable enhancement in this evaluation criterion compared with other works. In fact, this can show the power of our skeleton-difference layer, which takes many physical constraints into consideration in our training step. From the figure provided in [16], their mean Euclidean error is about 9.8 mm while ours is 8.4 mm. Nevertheless, the results in Figure 5 confirms that our predicted hand poses can maintain the good shapes because we impose the relations between joints on our model.

Figure 6 can demonstrate the completeness of our estimated hand pose. We compare with the predicted results of [17], and we can see that their estimated pose can be highly distorted from time to time while ours, on the other hand, is quite natural and intact in shape, including the angles and the bone length of a joint. We also provide the quantitative experimental results, whose comparison of different weights in skeleton different layer while training can also be read in Table 1. This also demonstrate the importance of skeleton-difference layer with those physical constraints in the training process. As the weight of skeleton-difference layer gets greater, the performance is better as well. However, we cannot remove Euclidean loss layer since skeleton-difference layer is the additional knowledge for improving training, but we indeed need the former to maintain the fundamental performance.

The full shape of a hand pose is significant since many real-world applications emphasize on the hand manipulation, and sometimes a mild dislocation of the whole pose is tolerable, but it will be very tough and troublesome if a hand pose loses its shape. The experimental results can show that our method achieves the both, accuracy and completeness.

Conclusions

In this paper, we propose a novel architecture for hand pose estimation that combines the skeleton-difference network (SDNet) and a hierarchical refinement method. In SDNet stage, we propose a skeleton-difference layer that takes physical constrains of a hand into consideration for training a deep learning model. In the refinement stage, we design an energy function to regress the positions of the joints that takes advantage of physical constrains, 2D projection appearance and temporal knowledge. The experimental results show that the average Euclidean distance error of our approach achieves 8.40 mm, and our result outperforms the previous works. Moreover, our method emphasizes the completeness of the prediction hands, and seldom can a highly distorted pose be seen in our estimation results. Therefore, our method is not only accurate but so robust that it can be adapted to hand-object interaction applications.

As mentioned, we would like to further develop the method for future real-world systems. Since the hand pose estimation is likely to be used in cases such as manipulating an object, it is inevitable that we need to integrate the hand-object interaction in these kinds of works. Hence, we will focus on not only improving the hand pose estimation quality but also extending the future applications.

References

[1] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun, "Real time hand pose estimation using depth sensors," in *Consumer Depth Cameras for Computer Vision*, ed: Springer, 2013, pp. 119-137.

[2] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys, "Accurate 3d pose estimation from a single depth image," in *IEEE International Conference on Computer Vision (ICCV), 2011*, 2011, pp. 731-738.

[3] J. S. Supancic, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan, "Depth-based hand pose estimation: data, methods, and challenges," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1868-1876.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.

[5] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097-1105.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PP, pp. 1-1, 2016.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 3431-3440.

[8] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1949-1957.

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

[10] M. Oberweger, P. Wohlhart, and V. Lepetit, "Training a feedback loop for hand pose estimation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3316-3324.

[11] M. Oberweger, P. Wohlhart, and V. Lepetit, "Hands deep in deep learning for hand pose estimation," *arXiv preprint arXiv:1502.06807,* 2015.

[12] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 3593-3601.

[13] A. Sinha, C. Choi, and K. Ramani, "Deephand: Robust hand pose estimation by completing a matrix imputed with deep features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 2016, pp. 4150-4158.

[14] T.-Y. Chen, M.-Y. Wu, Y.-H. Hsieh, and L.-C. Fu, "Deep learning for integrated hand detection and pose estimation," in *23rd International Conference on Pattern Recognition (ICPR), 2016* 2016, pp. 615-620.

[15] Q. Ye, S. Yuan, and T.-K. Kim, "Spatial Attention Deep Net with Partial PSO for Hierarchical Hybrid Hand Pose Estimation," in *European Conference on Computer Vision*, 2016, pp. 346-361.

[16] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, "Cascaded hand pose regression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 824-832.

[17] D. Tang, H. Jin Chang, A. Tejani, and T.-K. Kim, "Latent regression forest: Structured estimation of 3d articulated hand posture," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3786-3793.

[18] 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 *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3325-3333.

[19] D. Tang, T.-H. Yu, and T.-K. Kim, "Real-time articulated hand pose estimation using semi-supervised transductive regression forests," in *Proceedings of the IEEE international conference on computer vision*, 2013, pp. 3224-3231.

[20] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European conference on computer vision*, 2014, pp. 818-833.

[21] M. Oberweger, G. Riegler, P. Wohlhart, and V. Lepetit, "Efficiently creating 3D training data for fine hand pose estimation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4957-4965.

[22] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt, "Real-time joint tracking of a hand manipulating an object from rgb-d input," in *European Conference on Computer Vision*, 2016, pp. 294-310.

[23] B. Jian and B. C. Vemuri, "Robust point set registration using gaussian mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 33, pp. 1633-1645, 2011.

[24] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 675-678.