Imperial College London

3D Hand Pose Estimation Using Convolutional Neural Networks

Qi Ye

Supervised by Dr. Tae-Kyun Kim

Submitted for the degree of Doctor of Philosophy
in the
Department of Electrical and Electronic Engineering
October 2018

Abstract

3D hand pose estimation plays a fundamental role in natural human computer interactions. The problem is challenging due to complicated variations caused by complex articulations, multiple viewpoints, self-similar parts, severe self-occlusions, different shapes and sizes.

To handle these challenges, the thesis makes the following contributions. First, the problem of the multiple viewpoints and complex articulations of hand pose estimation is tackled by decomposing and transforming the input and output space by spatial transformations following the hand structure. By the transformation, both the variation of the input space and output is reduced, which makes the learning easier.

The second contribution is a probabilistic framework integrating all the hierarchical regressions. Variants with/without sampling, using different regressors and optimization methods are constructed and compared to provide an insight of the components under this framework.

The third contribution is based on the observation that for images with occlusions, there exist multiple plausible configurations for the occluded parts. A hierarchical mixture density network is proposed to handle the multi-modality of the locations for occluded hand joints. It leverages the state-of-the-art hand pose estimators based on Convolutional Neural Networks to facilitate feature learning while models the

multiple modes in a two-level hierarchy to reconcile single-valued (for visible joints) and multi-valued (for occluded joints) mapping in its output.

In addition, a complete labeled real hand datasets is collected by a tracking system with six 6D magnetic sensors and inverse kinematics to automatically obtain 21-joints hand pose annotations of depth maps.

Declaration of Originality

This thesis is submitted to the Electrical and Electronic Engineering Department of Imperial College London, in fulfilment of the requirements for the degree of Doctor of Philosophy. The contributions contained in this thesis are my own work whose contents have been published in the following listed papers. For all the contents included in the thesis, I made the major contribution in design, implementation and experiments.

- Danhang Tang*, **Qi Ye***, Shanxin Yuan, Jonathan Taylor, Pushmeet Kohli, Cem Keskin, Tae-Kyun Kim, Jamie Shotton. Opening the Black Box: Hierarchical Sampling Optimization for Hand Pose Estimation, in IEEE Transactions on Pattern Analysis and Machine Intelligence. doi: 10.1109/TPAMI.2018.2847688 (* indicates equal contribution)

- **Qi Ye**, Tae-Kyun Kim. Spatial Attention Deep Net with Partial PSO for Hierarchical Hybrid Hand Pose Estimation, Proc. of European Conf. on Computer Vision (ECCV), Munich, Germany, 2018.

- Shanxin Yuan, **Qi Ye**, Bjorn Stenger, Siddhant Jain, Tae-Kyun Kim Big-Hand2.2M Benchmark: Hand Pose Data Set and State of the Art Analysis, Proc. of IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), Honolulu, Hawaii, USA, 2017.

- **Qi Ye**, Shanxin Yuan, Tae-Kyun Kim Spatial Attention Deep Net with Partial PSO for Hierarchical Hybrid Hand Pose Estimation, Proc. of European Conf. on Computer Vision (ECCV), Amsterdam, Netherlands, 2016.

Copyright Declaration

Contents

List of Tables

List of Figures

Glossary

x input depth image. xxi, 33, 39, 40, 55, 78, 81–83, 85, 86

u the normalized cooridinate of a column in the image or feature maps. 36–38, 41

v the normalized cooridinate of a row in the image or feature maps. 36–38, 41

d the normalized depth value of a pixel in the image. 38

y the full hand pose represented by joint locations. The location of each joint is represented by (uvd). xv, xvi, xviii, xxi, 27, 33–41, 44, 48, 54, 55, 57–59, 78, 80–84

φ the full hand pose represented by joint rotations. xv, xvi, 34, 42–44

Y a set of full hand poses. 78, 82, 85, 86

q the index of a pose in pose set Y. 78, 80–84

i the index of a depth image in the dataset. 78–86

I the number of images in the dataset. 78, 82, 83, 85

j the index of a hand joint. xv, 27, 34–39, 44, 78–86

J the number of all the hand joints. 27, 78, 79, 81–83, 85

n the index of a sample. 55

N the number of all the samples. 55

f a CNN regressor. 38–41

l the layer index in the kinematic hierarchy. xvi, 33–37, 39–44, 48, 54, 55

L the number of the hierarchical layers. 54–56

k the stage of the cascaded framework. xvi, 34, 35, 38–41

K the number of the cascaded stages. 34, 39–41, 48

Λ the input feature maps of the spatial transformer. xv, xvi, 29, 35–39, 41

λ the output feature maps of the spatial transformer. 36–39

Φ the spatial transformer. xv, 35–41

θ the global in-plane rotation, the rotation of the spatial transformer. xv, 35–41

b the scaling factor of the spatial transformer. 36, 37

t the translation of the spatial transformer. 36, 37

T the function to get the in-plane rotation from the palm joint locations. 35, 36, 38

E_{Au} golden energy. 55, 56

E_{Ag} silver energy. 55–58

$\mathcal{N}$ Gaussian Density Function . 80, 83, 84

c the index of a component of Gaussian Mixture Model. 80–84

$\mathcal{C}$ the number of components of Gaussian Mixture Density. xxi, 80, 83–85, 87, 90–92

v output vector, the element of which denotes whether a joint is visible or not. 78–83, 85

w the parameters for the Benourlli distribution, the element of which denotes the probablity of a joint is visible. xxi, 79–84

μ the mean for the Gaussian distribution function. xxi, 80–84

σ the standard divation for the Gaussian distribution function. xxi, 80–84

ϵ the means for the function of Gaussian Mixture Model. xxi, 80–84

s the standard divations for the function of Gaussian Mixture Model. xxi, 80, 82–84

π the mixture coefficients for the function of Gaussian Mixture Model. xxi, 81–84, 88

Acronyms

CNN Convolutional Neural Network. x, xxi, 73–75, 77, 78, 85–87, 93–96

SGN Density Network with a Single Gaussian Model. xx, xxi, 73, 75, 85–92, 95

MDN Mixture Density Network. xxi, 77, 78, 84–92

HMDN Hierachical Mixture Density Network. xi, xx–xxii, 26, 73, 75, 81, 82, 84–95

DoF degree of freedom. 34, 35

PSO particle swarm optimisation. viii, xv, xvi, 29, 31, 34, 35, 42, 43, 48, 50, 52, 57

HSO Hierarchial Sampling Optimization. xviii, xix, 56, 57, 60, 61, 65, 66

$\mathbf{HS_FO_N}$ Hierarchial Sampling Optimization based on random forests with brutal-force optimization. 57, 64, 67, 68, 70

$\mathbf{HS_FO_P}$ Hierarchial Sampling Optimization based on random forests with PSO. xix, 57, 62, 64, 67, 68

$\mathbf{HS_CO_N}$ Hierarchial Sampling Optimization based on CNN with brutal-force optimization. 57, 69, 70

$\mathbf{HS_CO_P}$ Hierarchial Sampling Optimization based on CNN with PSO. xix, 57, 62, 64, 66, 67

HolisticCNN Regressing the full pose by a single CNN. 57, 60–63

HierCNN Regressing the full pose following the hierarchy by CNNs without sampling and optmization. xix, 57, 60, 61, 63–67

HolisticForest Regressing the full pose by a single random forest. 57

HierForest Regressing the full pose following the hierarchy by random forests without sampling and optmization. 57, 63–65, 67, 68

Chapter 1

Introduction

1.1 Problem Definition

Enabling human to interact with computers as natural as possible poses a challenge that requires knowledge from domains like natural language processing, gaze estimation, facial analysis, body pose and hand pose estimation. Among these interactions, hand pose estimation plays a fundamental role as in real world, we use our hands to perform various tasks and use intensively.

Hand pose estimation finds its applications in various scenarios, such as man-machine interfaces for robots and autonomous vehicles, for virtual reality (VR) and augmented reality (AR) device, sign language recognition [Chang et al., 2016; Yin and Chai, 2016], activity recognition [Rohrbach et al., 2012; Garcia-Hernando et al., 2018]. See Fig. 1.1 for some applications of hand pose estimation.

For all these applications, can we estimate the hand poses with non-invasive visual sensory inputs and achieve the natural interaction?

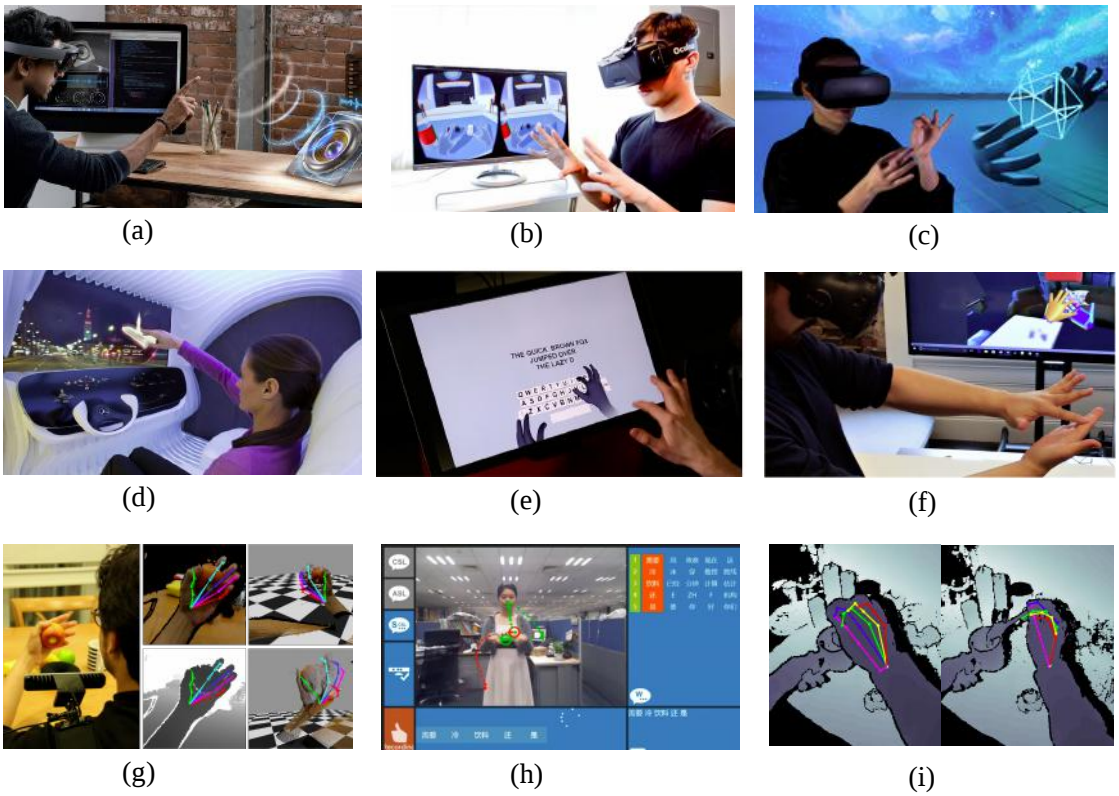


Figure 1.1: Hand pose estimation using RGB or depth images. (a) Microsoft HoloLens (b) Oculus Rift (c) Leap Motion (d) Mercedes Benz (e) Typing in VR [Taylor et al., 2016] (f) Two hands interaction [Taylor et al., 2017] (g) Hand-object interaction [Mueller et al., 2017] (h) Sign language recognition [Yin and Chai, 2016] (i) Action recognition [Garcia-Hernando et al., 2018]

Vision-based hand pose estimation is a promising technology for this. Before 2011, monocular cameras are utilised to estimate the hand pose [Chua et al., 2002; Athitsos and Sclaroff, 2003; Chua et al., 2002; Stenger et al., 2001b, 2006] but due to the ambiguity of monocular 2D images, estimating 3D hand pose is ill-posed.

With the introduction of depth camera in 2011, the Microsoft Kinect system [Shotton et al., 2011] has proved successful in body pose estimation. After the kinect depth sensor, other short-range depth sensors such as Intel RealSense, Primesense Carmine, and Leap Motion cameras are commercially available. Following the suc-

cess in body pose estimation using random forest, many work learns a mapping from an input depth image to the 3D pose configuration by routing the 3D points to branches of the trees [Tang et al., 2013, 2015, 2014; Qian et al., 2014a; Xu and Cheng, 2013; Sridhar et al., 2016, 2015]. These random forest based methods have demonstrated accurate real-time performance in certain scenarios, for example, front faced hand poses with a small number of occlusions in third viewpoints. Due to more Degree of Freedoms (DoF), self-similar parts and severe self-occlusions than the body pose, the hand pose estimation has its unique challenges.

In recent years, Convolutional Neural Networks (CNN) have been demonstrating the power of learning real world knowledge from images with supervision from labels. On top of the success of the depth sensors and CNN, we target at solving the unique challenges of hand pose estimation by learning a mapping from depth images to the 3D hand pose configurations. We use the images captured by the camera, for example images in Fig. 1.1, as input. Given the images, we aim to configure the parameters representing a hand model, which can be used by computer to interact objects in VR and AR, do remote control, scene understanding etc.. The hand model can be the simple skeleton model which is represented by all the joint locations or the angles between joints in Fig. 1.2(a), the cylinder model in Fig. 1.2(b) and the sphere model in Fig. 1.2(c) which encodes the thickness of the hand, or complex models in Fig. 1.2(d)(e) which personalise hand shapes for different users.

In this work, the input for the hand pose estimation is a depth image captured by a camera, which is the projection of 3D points on the hand surface projected to a camera plane. Each pixel value is the distance of the point to the camera plane. Fig. 1.3 shows some example images captured by a Intel RealSense SR300 [Intel] and the hand part cropped while Fig. 1.4 illustrates the 3D point clouds of a

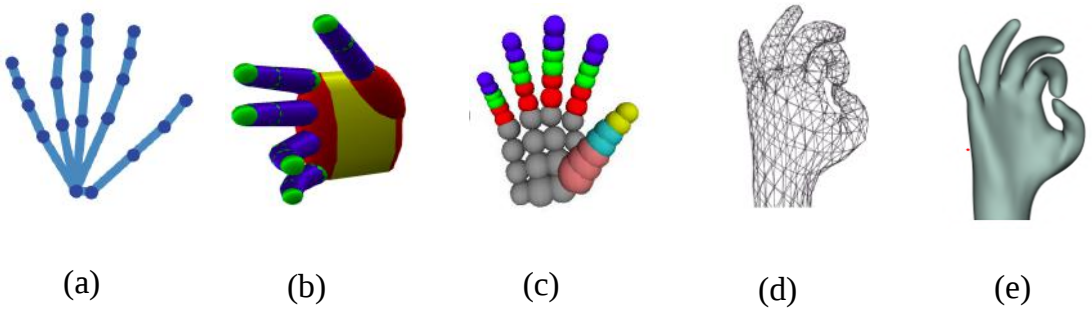


Figure 1.2: Hand Models. (a) Skeleton model [Tang et al., 2013] (b) Cylinder model [Oikonomidis et al., 2011a] (c) Sphere model [Qian et al., 2014a] (d) Mesh model [Tan et al., 2016] (f) Smooth surface model [Taylor et al., 2017]

sample image and its hand area. As the Kinect system [Shotton et al., 2011] is able to estimate the body pose while provides only very limited and noisy information about the hands, we mainly focus on the estimation from a crop hand image, which is denoted as x.

For the output, a simple skeleton model as Fig. 1.2(a) is used, represented by the 21 joint locations or the angles between the joints, which is denoted as y. As such, our problem can be formulated as configuring the parameters y of the hand skeleton model from a hand depth image x.

1.2 Challenges

When the pose parameters are joint locations, the pose estimation task can be treated as detecting the keypoints from input images and therefore shares some similarities with other vision problems such as facial landmarking, 6D Object Detection, human body pose estimation. Among these problems, hand pose estimation has most common challenges with human body estimation which has enjoyed a great

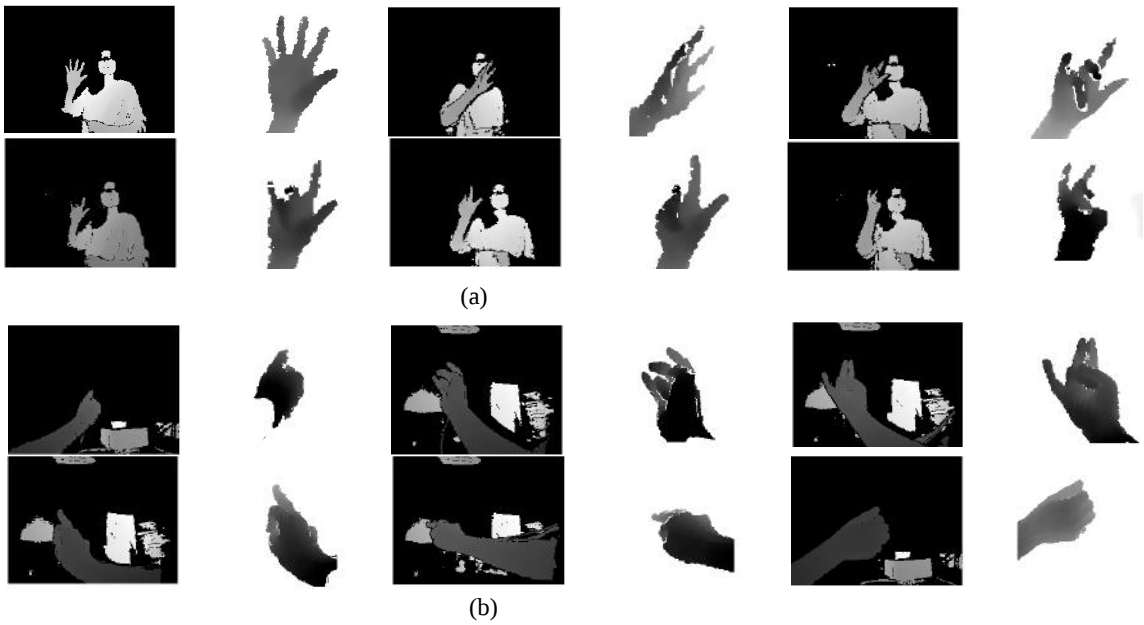


Figure 1.3: Depth images captured by Intel RealSense SR300 [Intel] and cropped hand areas. (a) images captured in third-person viewpoints; (b) images captured in first-person (ego-centric) viewpoints.

progress [Shotton et al., 2011; Jiu et al., 2014; Girshick et al., 2011; Ouyang et al., 2014; Papandreou et al., 2017; Cao et al., 2016] in the last decade. Both of them model the articulated objects with many degrees of freedom and self-occlusions while hand pose estimation has some unique difficulties due to complicated variations caused by high DoF articulations, multiple viewpoints, self-similar parts, severe self-occlusions, different shapes and sizes.

High Degrees of Freedom

Each finger of the human hand follows a kinematic chain with the palm as its base reference and has four DoFs, two for MCP joints (flexion and abduction), one flexion for each of the PIP and the DIP joint [1]. Therefore, if the hand articulation is

[1]MCP, PIP, and DIP stand for the metacarpophalangeal, proximal interphalangeal, and distal interphalangeal joints, respectively.

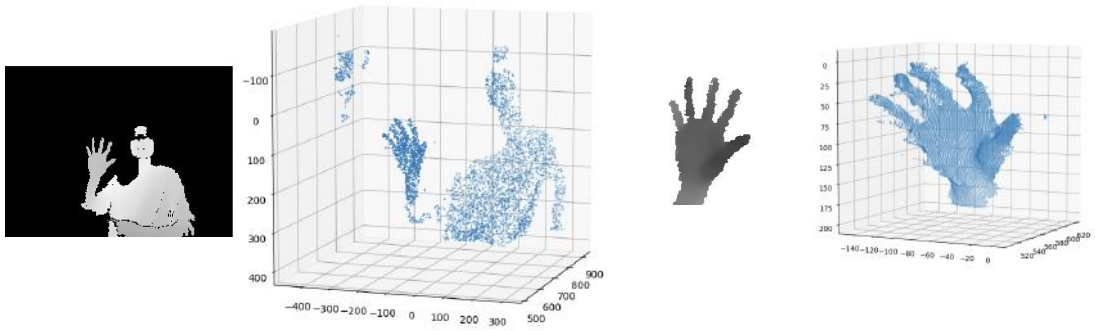


Figure 1.4: (a) a depth image (b) 3D point cloud (c) a crop hand image images (d) 3D point cloud of the hand area.

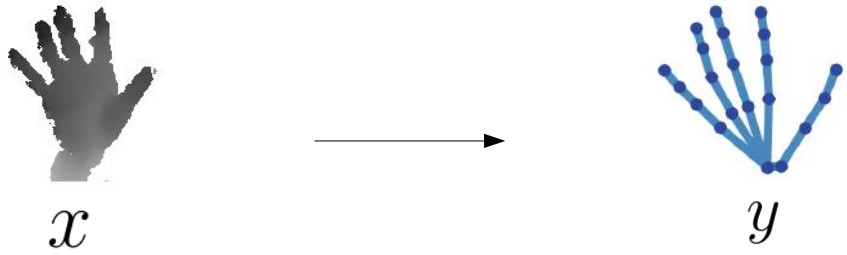


Figure 1.5: The hand pose estimation is configuring the parameters y of the hand skeleton model from a hand depth image x

represented by its joint angles [2], the articulation space is R^{20}, though the articulation space is constrained by the limit of the range of the finger motions and correlation between fingers.

Under the constraint, to achieve the maximum coverage of the articulation space when collecting a hand pose dataset, the hand can have 32 extremal poses (2^5) where the bent finger reaches its maximum limit and there are 496 transitioning motions between these extreme poses ($\binom{32}{2} = 496$ pairs) [Yuan et al., 2017].

Apart from the articulation, the hand pose has other 6 DoFs, the global location and the global viewpoint. As the appearances of the human hand change not only

[2]The thumb is approximated by four DoFs

due to the viewpoint but also the coupled articulation, the viewpoint estimation of the hand pose is more complex than that of the rigid objects.

Self-similar parts

In human body pose estimation, the similarity of the left and the right limb cause ambiguity for the keypoint detector and often results false detections. In hand pose estimation, the ambiguity is more severe as all the fingers look similar; without global information, one can hardly discriminate different fingers.

Self-occlusion

Self-occlusion here refers some parts of the object are occluded by other parts. In the self-occlusion of rigid objects, the locations of occluded keypoints are deterministic when using the evidence of the visible parts and object priors while in the self-occlusion of articulated objects, in most cases, the locations of occluded keypoints are unknown even with the cues from the visible parts. For example, from the images in the first column of Fig. 1.3(b), the occluded joint locations can have many possible configurations though the constraints from the visible joints can help to reduce the configuration space.

Various hand size and shapes

Ballan et al. [Ballan et al., 2012] shows extremely robust hand tracking with a user-specialized hand model. However, as existing techniques for model construction depend on large datasets of high quality scans [Khamis et al., 2015] while the hand data captured by the sensor is noisy and contains fewer foreground pixels due to their size relative to the whole body, the variation of hand size and shapes is less discussed when compared to human body [Anguelov et al., 2005] and face [Blanz and

Vetter, 1999]. The hand model of Ballan et al. [Ballan et al., 2012] requires manual rigging and a multi-camera capture setup. Taylor et al. [Taylor et al., 2014] acquire a user-specialized model from a single depth camera, but require long calibration sequences in which all degrees of freedom of the hand have to be exercised while Khamis et al. [Khamis et al., 2015] improve the method by using a set of short sequences.

Noisy real data

The modern commodity-level depth sensors, such as Intel RealSense, Microsoft Kinect, Google Tango, PrimeSense produce images with missing data when object surfaces are too thin, too close or far from the sensors, too shiny etc.. Even in indoor scenarios, depth images are often missing more than 50 % of the pixels [Zhang and Funkhouser, 2018]. Due to the hand size, the sensors are only capable of sparse sampling and the sensor noisy tends to have larger effect. Also, these sensors typically run at a frequency of 60FPS while human hands can move up to 200FPS in translation [Erol et al., 2007], which results in motion blur within frames.

1.3 Thesis Overview and Contributions

The thesis address the problem of 3D hand pose estimation whose challenges are mentioned in the last section. Before starting the explanation of our proposed methods, we first cover a general literature review on related areas in Chapter 2 and then introduce the public datasets and the proposed dataset in Chapter 3. Chapter 4 to Chapter 6 contains our main contributions, which are summarised as follows. Chapter 7 concludes the thesis with findings and discussion about future

works.

Here is summary of our contributions.

Hierarchical Deep Net with Spatial Attention

Discriminative methods often generate hand poses kinematically implausible, then generative methods are used to correct (or verify) these results in a hybrid method. Estimating 3D hand pose in a hierarchy, where the high-dimensional output space is decomposed into smaller ones, has been shown effective. Existing hierarchical methods mainly focus on the decomposition of the output space while the input space remains almost the same along the hierarchy. In Chapter 4, a hybrid hand pose estimation method is proposed by applying the kinematic hierarchy strategy to the input space (as well as the output space) of the discriminative method by a spatial attention mechanism and to the optimization of the generative method by hierarchical Particle Swarm Optimization (PSO). The spatial attention mechanism integrates cascaded and hierarchical regression into a CNN framework by transforming both the input(and feature space) and the output space, which greatly reduces the viewpoint and articulation variations. Between the levels in the hierarchy, the hierarchical PSO forces the kinematic constraints to the results of the CNNs.

Integration of Hierarchical Deep Net

In 3D hand pose regression, Sun et al. and Tang et al. [Sun et al., 2015; Tang et al., 2015] follow the hand hierarchical kinematic structures and decompose the problem into sub-problems which can be tackled independently. The hierarchical regression in Chapter 4 and these two methods differs regarding to how the hierarchy is constructed, which method is used to prevent error accumulation, whether a full pose energy is used to get the best hypothesis. These different hierarchical regressions are

integrated into a probabilistic framework in Chapter 5. Variants under the framework are constructed and compared to analyse the efficacy and efficiency of each component, which provides guidances for choosing the right components in certain cases.

Probabilistic model with Occlusion awareness

Though having proven to be effective, most existing discriminative methods including the hierarchical regression in Chapter 4 yield a single deterministic estimation of target poses. Due to their single-value mapping intrinsic, they fail to adequately handle self-occlusion problems, where occluded joints present multiple modes. In Chapter 6, we tackle the self-occlusion issue and provide a complete description of observed poses given an input depth image by a novel method called hierarchical mixture density networks (HMDN). The proposed method leverages the state-of-the-art hand pose estimators based on Convolutional Neural Networks to facilitate feature learning, while it models the multiple modes in a two-level hierarchy to reconcile single-valued and multi-valued mapping in its output. The whole framework with a mixture of two differentiable density functions is naturally end-to-end trainable.

Chapter 2

Literature Review

As mentioned in the previous chapter, 3D hand pose estimation can be considered as the keypoint estimation problem when the skeleton model parameterized by the 3D joint locations is utilised.

Keypoint estimation on human and objects has achieved significant progress in the past decades as the keypoint estimation is a prerequisite for comprehensive human understanding, where we want to localize people and objects, understand the activities when they are interacting with the environment, predict their next move based on the understanding etc.. In the facial landmark detection, the detected keypoints in human faces, e.g. the tip of the nose, eyebrows, the eye corner and the mouth help to construct the 3D Morphable model [Zhu et al., 2015a], estimate the head pose and analyse facial deformation [Wu et al., 2017b], preform facial reenactment [Thies et al., 2016]. In the human pose estimation, the joints on the arms, legs and keypoints on the torso form a skeleton-based representation of the human structure which are fed into LSTM to recognise human activities [Du et al., 2015] or are used to reconstruct the 3D meshes [Kanazawa et al., 2018]. 6D object pose estimation often takes the keypoints as an intermediate representation to get the final translation

and rotation [Krull et al., 2017] or the 3D model [Wu et al., 2016]. With the shared problems, the methods and goals of 3D hand pose estimation follow the roughly the same development.

As such, we will first review some related work on 2D/3D keypoint estimation for human body pose, face landmarking/pose, 6D object detection/pose and then dive into the literature of 3D hand pose estimation.

2.1 2D Keypoint Estimation

The related work on 2D keypoint estimation can be categorized into three groups: direct regression, dense pixel-wise prediction and model/template-based methods.

Direct Regression

The regression-based methods estimate 2D keypoint locations explicitly from images or features. Using the random forest, Girshick et al. [Girshick et al., 2011] get the body joint locations by votes from the pixels in the leaf nodes. A number of studies [Valstar et al., 2010; Cootes et al., 2012; Dantone et al., 2012] in facial landmarking use the similar voting scheme to get the locations of the landmarks. Early methods using CNN predict the 2D coordinates of keypoints directly [Toshev and Szegedy, 2013; Carreira et al., 2016].

The direct regression based on the assumption of single instance (object, body, hand, face) in the input. For example, if there are multiple bodies in the input, the regressor should represent the multiple modes of the same joint while the direct regression fails to handle. The following dense pixel-wise prediction is free of this difficulty.

Pixel-wise Prediction

Methods in this category provide dense pixel-wise predictions and then infer object pose or joint coordinates from these predictions. At each pixel, a network predicts whether (or where) the pixel is located on a part of human body, or the surface of the object. Brachmann et al. [Brachmann et al., 2014] get the pixel-wise prediction using random forests and generate hypotheses of the objects by sampling a small set of pixels and comparing the forest prediction with input depth images. In human pose estimation [Shotton et al., 2011], the pixels are classified into different body parts by random forests and those belonging to the same part are aggregated to infer the joint location using mean shift.

In recent years, heatmap regression in the domain of deep learning is most commonly used for dense pixel-wise prediction and has been successfully applied to human pose estimation [Papandreou et al., 2017; Newell et al., 2016; Wei et al., 2016; Cao et al., 2016], 6D object reconstruction [Wu et al., 2016], face landmarking [Sun et al., 2013], where the coordinate of the keypoint is the location of the highest response on the heatmap. For example, the two heatmap based methods [Papandreou et al., 2017; Cao et al., 2016] in human pose estimation are the winners of the COCO keypoints challenge of 2016 and 2017, respectively.

Most of the work in both the regression and classification-based estimation exploits cascaded framework to refine an initial guess of the locations iteratively for higher accuracy [Sun et al., 2013; Cao et al., 2016; Wei et al., 2016; Carreira et al., 2016]. Carreira et al. [Carreira et al., 2016] also combines the direct regression and the heatmap regression within the cascade stages, where the heatmaps are input into the iterated modules and the 2D coordinates are the output.

Template-based Estimation

For most of the history, the template-based methods have heavily based on the idea of part-based models, where face/body templates are built to fit the input images. The seminal work can trace back to the Pictorial Structure of Fischler and Elschlager [Fischler and Elschlager, 1973], which is extended and applied on object recognition [Felzenszwalb and Huttenlocher, 2005], image parsing [Ramanan, 2007], body pose space reduction [Ferrari et al., 2008]. Following this idea, Felzenszwalb et al. propose Deformable Part Model for 2D keypoint estimation, which is applied on human body pose [Felzenszwalb et al., 2008] and on facial landmarking [Zhu and Ramanan, 2012]. These work spurs intensive work on template-based 2D keypoint detection in body [Andriluka et al., 2009; Yang and Ramanan, 2011; Gkioxari et al., 2013], face [Asthana et al., 2013; Yu et al., 2013] and 6D object [Hinterstoisser et al., 2012; Rios-Cabrera and Tuytelaars, 2013].

2.2 3D Pose Regression

In this section, we provide a brief review on estimating the full 3D pose from a single image. As discussed in the overview, 2D keypoints often serves as an input for further full pose estimation, reconstruction and image understanding. Estimating the 3D pose from input images generally learns to regress the 3D pose directly or via intermediate 2D pose representation.

3D pose/shape from images

Estimating the 3D pose directly shares the same problem with that of 2D pose estimation: the number of the pose should be fixed.

PoseNet [Kendall et al., 2015] uses CNN to directly regress the translation and rotation for camera pose and PoseCNN [Xiang et al., 2018] introduces ShapeMatch-Loss for pose estimation for symmetric objects to train a CNN with the 6D object pose as output.

When estimating the 3D body pose directly [Zhou et al., 2016a; Li et al., 2015; Li and Chan, 2014], the discriminatively learned regressor may output kinematic implausible poses. To enforcing the structural constraints, Zhou et al. [Zhou et al., 2016a] embed a kinematic pose model into a CNN that regresses the 3D pose directly and Li et al. [Li et al., 2015] propose maximum-margin structured learning to compare image and 3D pose pairs.

3D pose via intermediate regressions

When estimating the pose of rigid objects, most work in the literature first predicts dense pixel-wise coordinates of the objects [Brachmann et al., 2014; Michel et al., 2015; Krull et al., 2015, 2017; Kehl et al., 2017; Wu et al., 2016, 2017a]. From the estimated coordinates, object classes or rotations, multiple pose hypotheses are generated and are scored by comparing the rendered and observed image patches. They all refine the best hypotheses, but differs in their scoring functions and network for the discriminative learning part. For the scoring functions, [Brachmann et al., 2014; Michel et al., 2015] use simple pixel-wise distance function while Krull et al. [Krull et al., 2015, 2017] learn a CNN for comparing the similarity. Kehl et al. [Kehl et al., 2017] replace the random forests used in [Brachmann et al., 2014; Michel et al., 2015; Krull et al., 2015, 2017] by CNN to produce dense predictions of object classes, viewpoints, rotations, projected bounding boxes corners to get the 6D pose while Wu et al. reconstruct 3D shape by estimating the 2D keypoint locations in [Wu et al., 2016] and normals, depth and silhouette in [Wu et al., 2017a] in a end-to-end

CNN framework.

In contrast to the rigid object, human body is highly articulated; with known 2D joint positions, the possible 3D poses consistent with the 2D positions are infinite. 3D geometric pose proirs [Akhter and Black, 2015; Ramakrishna et al., 2012] and temporal constraints [Akhter et al., 2011; Lee et al., 2017] are modelled and enforced to find the correct possible 3D poses. Though human face is not articulated, the 3D shape of the face is highly deformable. [Simon et al., 2017] construct a Kronecker Markov Random Field as a prior distribution over the time-varying 3D shapes like human face and body while Zhao et al. [Zhao et al., 2017] reconstruct 3D shape of rigid objects like cars and non-rigid objects like flag and human body/face from 2D landmarks by a deep neural network.

Estimating the 3D body pose directly requires a large number of annotated 3D locations for the joints which is hard to get while the 2D annotation is comparably abundant. Some methods [Tome et al., 2017; Zhou et al., 2016c] thus decouple the 2D and 3D training data but estimate 2D landmark and 3D pose jointly, instead of previous pipeline methods.

2.3 3D Hand Pose Regression

3D hand pose regression shares the similar development in 3D pose regression. In the previous sections, the related work are most about learning a mapping function between the input images to the pose parameters as our work mainly forces on learning the mapping function. In this section, to give an overview of the progress in hand pose regression, we also introduce approaches based on model fitting, and

the hybrid of learning based methods and model-based methods.

Discriminative Approaches

Discriminative (or learning-based) approaches learn a mapping function from input image to the configuration of pose parameters.

In Section 2.1, we mentioned that in 3D body pose estimation the Kinect [Shotton et al., 2011] treats the estimation as a dense per-pixel classification problem by classifying each foreground pixel into one of the body parts, followed by a mean-seeking method to locate body part centres and hence skeletons. On top of that, Girshick et al. [Girshick et al., 2011] propose to use regression forest to estimate body joints directly, including occluded ones.

Follow the same routes in body pose, per-pixel classification based methods [Xu and Cheng, 2013; Sridhar et al., 2016, 2015; Tang et al., 2013; Neverova et al., 2014; Ionescu et al., 2014] assign each pixel with a hand part label and then infer hand joint locations. Direct regression based methods [Wan et al., 2016; Oberweger et al., 2015a; Ye et al., 2016; Tang et al., 2014] estimate a subset of the parameters directly without intermediate representation.

Both per-pixel classification and the direct regression for subset of pose parameters make a decision from local appearance, i.e. part-based methods, and thus have a better generalization ability than holistic approaches. Sharing the same disadvantages of part-based body pose algorithms, they lack the kinematic information, since the output parameters are treated independently, and may produce many false alarms due to the self-similar hand parts.

On the other hand, part-based methods estimate the parameters of the underlying

latent variables separately and thus each regressor has a much lower dimensional problem to tackle. However, post-processing is usually required to recover kinematically consistent parameters.

Holistic regressors [Tompson et al., 2014; Oberweger et al., 2015b] that estimate all parameters together with the context from the whole image, keep the hand kinematic information implicitly, but requires more data to cover the variations in the whole images.

Generative Approaches

Generative (or model-based) approaches treat hand pose estimation as an inverse problem, searching for an optimal pose configuration to minimize an energy function that captures the difference between a 3D hand model and observation. Since the hand model is driven by pose parameters in the target space (joint angles), kinematic consistency is preserved. However, previous generative methods in body pose [Zhu et al., 2008; Deutscher and Reid, 2005] and hand pose [Rehg and Kanade, 1994; Stenger et al., 2001a, 2006; Oikonomidis et al., 2011b, 2012; Qian et al., 2014b; Panteleris et al., 2015; Tan et al., 2016; Taylor et al., 2016] involve expensive evaluations of the energy function in a high dimensional observation space. On the optimization side, a popular strategy in hand pose estimation is to use particle swarm optimization (PSO) for solving the inverse problem [Oikonomidis et al., 2011b, 2012]. PSO hypothesizes a 'swarm' of particles (pose hypotheses) that are iteratively perturbed based on their own and the other particles' energies. In order to have good performance, hundreds of particles are required easily saturating a high-end GPU to achieve interactive frame rates. Chen et al. [Qian et al., 2014b] propose a simplified sphere hand model to circumvent the computational expensive rendering part. The method evaluates the sphere-based model over the input point cloud, and combines

ICP and PSO to reduce the number of the evaluations and accelerate parameter search. The ICP-PSO optimization, however, like PSO, also relies on good initializations from the previous frame, and the optimization fails when motions are abrupt.

Hybrid Approaches

Considering the limitations in discriminative and generative approaches, a possible improvement is to combine the two for better efficiency and kinematic constraints. In practice, discriminative approaches generate a set of candidates, joint positions for example, to narrow the parameter search space for following generative approaches. This strategy has been applied to both body pose estimation [Salzmann and Urtasun, 2010; Baak et al., 2011; Ye et al., 2011; Taylor et al., 2012; Yang and Ramanan, 2011; Wang and Li, 2013; Yub Jung et al., 2015] and hand pose estimation [Sridhar et al., 2016, 2013; Tzionas et al., 2016b; Ballan et al., 2012; Sharp et al., 2015; Krejov et al., 2015; Poier et al., 2015; Choi et al., 2015; Tompson et al., 2014]. Similar to [Salzmann and Urtasun, 2010] in body pose, a PSO is initialized with predictions from CNN [Tompson et al., 2014] or decision forest [Sharp et al., 2015]. This kind of approaches treats the mapping from the pose parameters to the rendered images as a black box: prior knowledge about the generation process and relationships between different parameters are ignored.

Discussion

Among these three different approaches, heatmap regression using 3D CNN [Moon et al., 2018] produces the best performance for the single frame pose estimation task in the Hands In the Million Challenge (HIM2017) [HIM] and the other heatmap regression using 2D CNN comes at the second place for per-frame hand pose esti-

mation. Overall, the heatmap based regression in the keypoints detection benefits from keeping the spatial of the images and the 3D heatmap regression using 3D CNN further makes full use of the depth data. However, the boost in the accuracy is at the cost of the memory consumption and runtime efficiency when compared with direct regression of the joint location.

Both direct and heatmap regression requires proper normalization of foreground hand pixels (voxels), which depends on the performance of hand detectors while the hand detection also poses a problem. The challenges [HIM] have two tasks: single frame pose estimation task, where the frames are shuffled so no information from previous frames can be used but the bounding boxes are provided and the tracking task, where the video clips are provided and the ground truth of the first frame of each clip is given. Even with the ground truth pose for the first frame of each clip and the availability of temporal information, the average mean joint error (12.3mm) of the top 3 submissions for tracking task (5 submission) are still higher than that (10.3mm) for single frame task (17 submission). Based on the figures, we point out that apart from the difficulty in the pose estimation in the extreme viewpoints and occlusions in itself, the hand detection in various scenarios hinders the overall performance of the whole system.

In all the work mentioned, the euclidean distance between the estimated joint locations and the ground truth are measured to evaluate the methods. While simple and effective, the metric lacks the statistics information in different aspects and therefore, is insufficient in evaluating current pose estimator. With the availability of large data, the top 3 methods in the challenge can achieve the mean joint error as low as 10mm. The evaluation metrics for the corner cases of the hand estimation system, such as occlusions, extreme poses, unseen poses, are to be proposed.

For other finding regarding the achievements in 3D hand pose estimation and the existing challenges, we refer reader to the work [Yuan et al., 2018]. Also, a detailed comparison of methods before 2015 can be found in [Supancic et al., 2015].

Chapter 3

Datasets

3.1 Public Datasets and Proposed Dataset

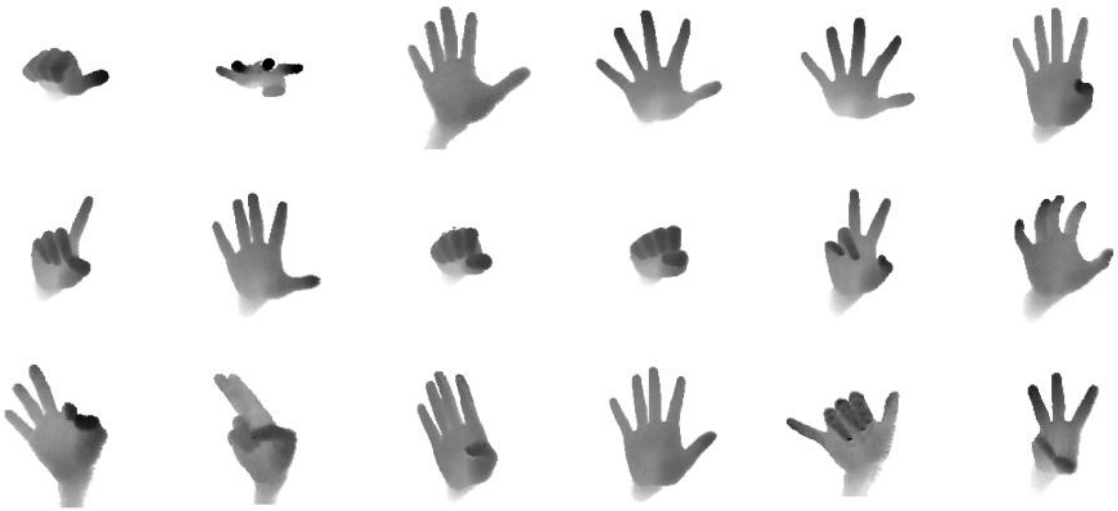
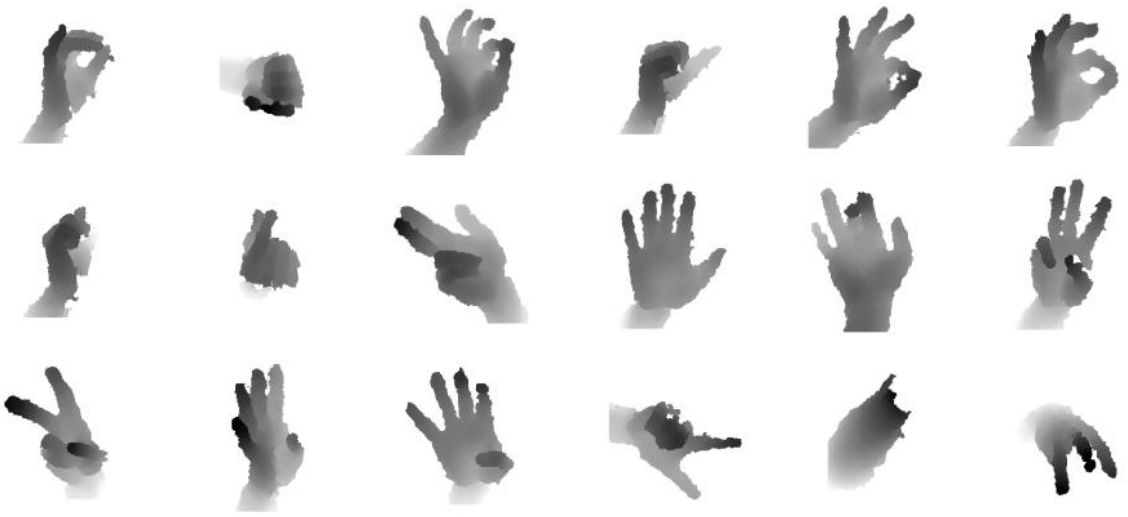
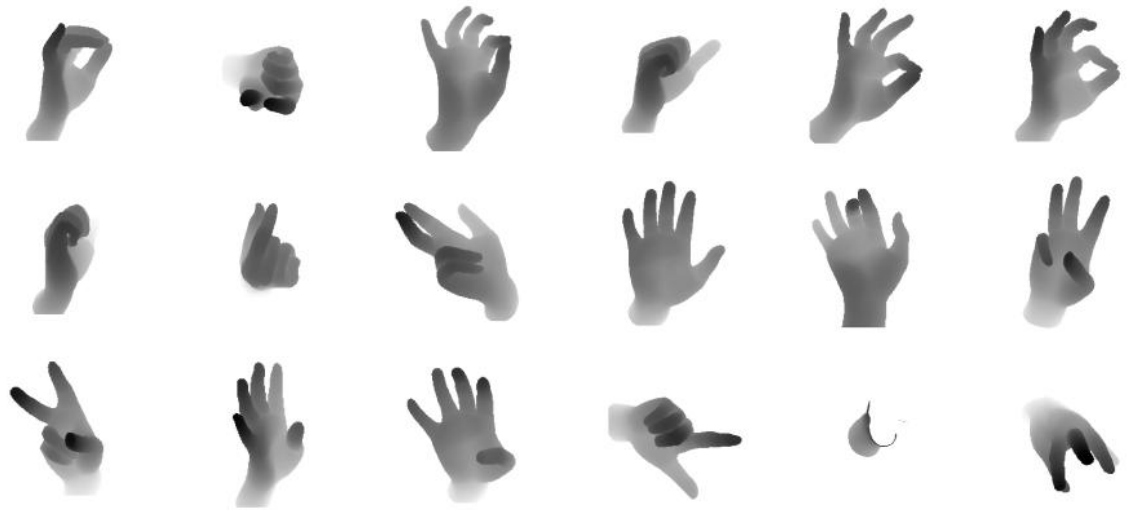


Figure 3.1: ICVL dataset [Tang et al., 2014].

Existing datasets are either generated synthetically or captured using depth sensors. Examples images in Fig. 3.1, Fig. 3.2 are real images captured by Intel Creative Sensor [Tang et al., 2014] and PrimeSense [Tompson et al., 2014] while images in Fig. 3.3 are synthesized [Sharp et al., 2015] (all the images are randomly selected).



(a) real images



(b) synthetic images

Figure 3.2: NYU dataset [Tompson et al., 2014]. A real data with

ICVL [Tang et al., 2015] dataset is a real sequence captured by Intel RealSense with the range of view about 120 degrees consisting 1596 test frames and 16008 training frames. 16 bone centre locations are provided for each hand pose.

NYU [Tompson et al., 2014] dataset is a real sequence acquired by PrimeSense containing 8252 test-set and 72757 training-set frames with a full range of views. 36 joint locations are provided for each hand pose. The images of the training set is

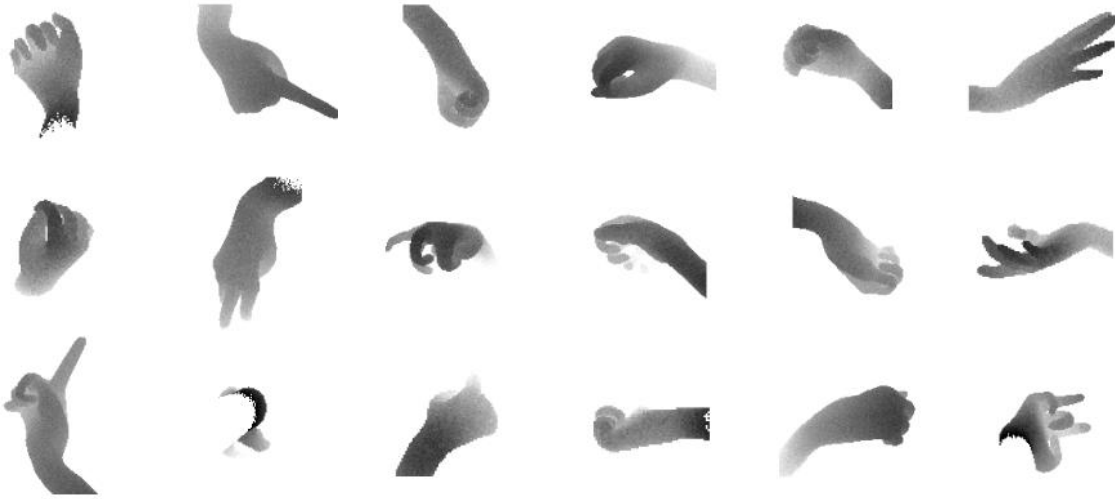


Figure 3.3: MSHD dataset [Sharp et al., 2015]

Figure 3.4: Bighand dataset [Yuan et al., 2017]

collected from one subject and those of testing set are from two subject (one subject is seen in the training set).

MSHD [Sharp et al., 2015] dataset is a challenging dataset that covers a full range of views and complex articulations with 100000 synthetic images in the training-set and 2000 synthetic images in the test set. 22 joint locations are provided for each hand pose. For the hand shape, a scale factor is uniformly drawn in a interval.

We summarize the characteristics in Table 3.3. As the annotations of these datasets do not conform to each other, we use the annotation version in [Tang et al., 2015] that labels locations of the joints.

Proposed Bighand Dataset

Synthetic images exhibit a certain level of appearance difference from real depth images. Fig. 3.2 (a)(b) shows the pairs of real images and their synthetic ones. Real datasets are limited in quantity and coverage, mainly due to the difficulty to annotate them. For examples, images from ICVL Fig. 3.1 and NYU Fig. 3.2 dataset most appear with palm facing front. We therefore propose a tracking system with six 6D magnetic sensors and inverse kinematics to automatically obtain 21-joints hand pose annotations of depth maps captured with minimal restriction on the range of motion.

The proposed Bighand dataset is collected in both third-person viewpoint and first-person viewpoint and have a full and even coverage of both the viewpoint and articulation space. Considering the variation of hand shape, the hand images of 10 subjects are captured. Fig. 3.4 shows some examples images from the dataset.

To measure the quality of these datasets, both qualitative and quantitative analyses are conducted. For the qualitative analyses, 2D t-SNE embeddings of the pose space are shown in Fig. 3.5. The Bighand dataset has broader and evener coverage than other datasets both in the viewpoint space and articulation space.

For the quantitative comparison, CNN models with the same structure are trained on the training sets of ICVL, NYU, MSHD and Bighand and then are cross-tested on the testing sets. The mean joint errors of the cross testing are reported in Table 3.2. All CNN models training on ICVL, NYU, or MSHD does not generalize

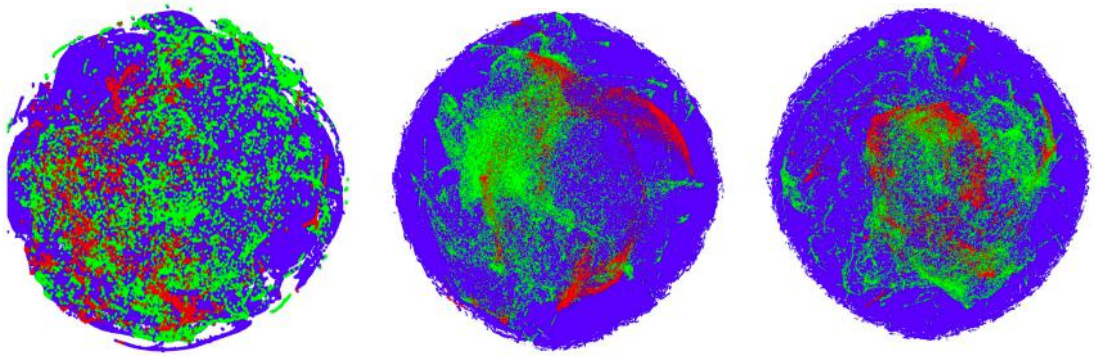


Figure 3.5: 2D t-SNE embedding of the hand pose space of different datasets. Blue: Bighand; Red: ICVL; Green: NYU. The global viewpoint and the articulation coverage are shown in the first and the middle figure. The combination of the viewpoint and articulation is demonstrated in the last figure. Bighand dataset spreads broader and evener than the other existing datasets.

well to other datasets, whose errors are much higher than errors of models trained on corresponding training sets.

On the other hand, the model trained on BigHand performs well on ICVL and NYU, achieving comparable or even lower errors than those of models on the corresponding training sets. Though, the performance of the same model when tested on MSHD dataset is worse than that of the model training on MSHD dataset, which is mainly due to the appearance difference between the real images and synthetic images, it still achieves lower error than other models training on ICVL and NYU.

The cross-tested errors therefore verify sufficient variations of the Bighand dataset in shape, articulation, viewpoint and also the high annotation accuracy of the system.

Table 3.1: Description of datasets

Dataset	ICVL	NYU	MSHD	BigHand
sensor	Intel Creative Sense	PrimeSense	Kinect2	Intel Real Sense
data type	real	real	synthetic	real
subjects	10	2	*	10
train	16,008	72,757	100,000	2,200,000
test	1,596	8,252	2,000	-
viewpoint	3rd	3rd	full	full

Table 3.2: Cross-benchmark comparison.

train \ test	ICVL	NYU	MSHD	BigHand
ICVL	**12.3**	35.1	65.8	46.3
NYU	20.1	**21.4**	64.1	49.6
MSHD	25.3	30.8	**21.3**	49.7
BigHand	**14.9**	**20.6**	**43.7**	**17.1**

3.2 Labelling Hand Parts and Visibility

As one of our targets is to tackle the occlusion problem, which will be discussed in Chapter 6, the ground truth about the occlusion are required. We define that when the bone corresponding to a joint (the end of the bone) have no 3D points (pixels if representing in 2D images instead of 3D point clouds) captured from by the depth sensors, the joint is occluded; otherwise, the joint is visible.

According to the definition,to get the visibility of a joint, the belonging of each pixel, hand part labels, are to be acquired.

The labels available for most hand datasets are joint locations. The joint locations can be converted to the hand part labels with a hand model. The sphere model of

[Qian et al., 2014a] is chosen from the hand models shown in Chapter 1 due to its simplicity, less computation from the joint location, less requirement for the shape parameters. The hand model for the labelling is color-coded for different hand parts, shown in Fig. 3.6. The shape parameters required from the model are the thickness of the joints, which are recorded in the datasets or can be acquired by manually aligning the sphere model to the point clouds.

Figure 3.6: the hand sphere model

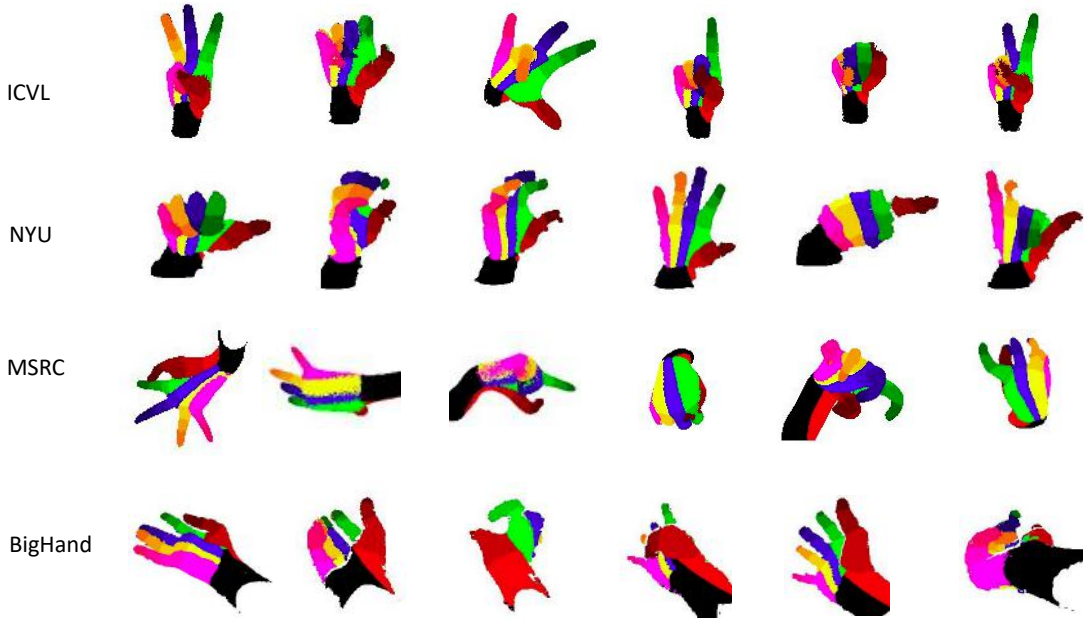


Figure 3.7: Images with the labelled hand parts

First, we follow the steps below to get the hand part labels.

1. For each image, a sphere model is constructed from the joint locations. In the sphere model, there is a sphere centred on each joint with its radius being the half of the thickness of the joint. For the spheres between two neighbouring joints, the centres and the radii are linearly interpolated.

2. With this sphere mode, we define 21 hand parts (corresponding the 21 joints). Each hand part is represented by several spheres and these spheres share the same part label, see Fig. 3.6.

3. Each point in the point clouds is assigned to the nearest sphere and is labelled with the hand part the sphere is assigned.

After all pixels are assigned to the hand parts, we know whether a hand part has points or not. As the depth sensor has noises, in experiment a joint is set to be occluded when the number of points assigned to the corresponding hand part is below a threshold. When setting the threshold, the distance of the hand to the camera should be considered. To this end, we normalize the number of pixels belonging to different parts as follows.

1. The numbers of pixels assigned to spheres belonging to a hand part are sum up and normalized by the mean depth of all the pixels;

2. The numbers are further normalized by that of a depth image with a open pose facing the sensor;

3. If the normalized number of pixels is below a threshold (0.01 in our experiment), the bone (joint) is treated occluded.

3.3 EgoBigHand

To train and test the method proposed in Chapter 6 aiming at the occlusion challenge, a dataset with a considerable number of images with occlusions are required.

We first investigate the occlusion rate of the dataset covered in section Section 3.1. The rate of occluded finger joints and the total number of training and testing images are listed in Table 3.3 using the visibility labels generated in Section 3.2. Either training set or testing set of ICVL, NYU and MSHD has a low rate of occlusion. Bighand dataset is collected with aiming at a full coverage of viewpoints and articulations. The egocentric subset of the Bighand dataset includes many frames with occlusions. To evaluate the proposed methods and compare with the state-of-the-art in handling with occlusion, we utilise the egocentric subset and we name it EgoBigHand. EgoBigHand includes 8 subjects: frames of 7 subjects are used for training and frames of 1 subject for testing.

Table 3.3: The rate of occluded finger joints and the total number of frames

Dataset		ICVL	NYU	MSHD	EgoBigHand
Train	occ. rate	0.06	0.09	0.33	0.48
	total no.	16,008	72,757	100,000	969,600
Test	rate	0.01	0.36	0.16	0.24
	total no.	1,596	8,252	2,000	33,468

In Chapter 4 and Chapter 5, we use ICVL, NYU and MSHD datasets while in Chapter 6 we exploit those containing a higher portion of occluded joints in the experiments i.e. EgoBighand and MSHD.

Augmentation of EgoBigHand Dataset

Though the egocentric subset includes lots of self-occlusions, it lacks diverse artic-

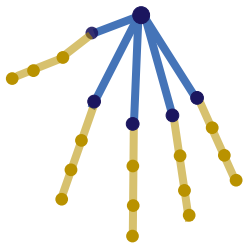


Figure 3.8: The finger joints are in orange and the location labels for the occluded fingers are augmented by using the labels from the third-person viewpoint subset of BigHand.

ulations. The other part of Bighand dataset, the third-person viewpoint subset, spans the full articulation space while the proportion of occluded joints, especially severe occlusions, is low. As the possible configurations of the joint locations under occlusion are to be modelled, adequate configurations for these occluded joints are required. We therefore augment the egocentric subset using the articulations of the third-person view dataset. The augmentation is implemented by adding joint location labels from the third-person viewpoint subset for the occluded finger joints.

The steps for the augmentation are detailed. The hand joints shown in Fig. 3.8 can be seen as a tree, where the wrist is the root. The tree has five branches from the root to the finger tips and for a joint, all the joints in the tip side along the same branch are its children.

The augmentation for the pose label $y = \{y^j | j = 1, ..., J\}$ of an image with occlusion in the egocentric subset follows two steps.

1. For each branch, starting from the finger joint on the top level, we search the first joint satisfying two conditions: 1) the joint is occluded; 2) the children of the joint are occluded. The partial pose label for the joint and all its children joints is $\{y^j | j \in J_1\}$ and the partial label for the other joints is $\{y^j | j \in J_2\}$.

2. For the joint and its children joints, we randomly choose partial pose labels (rotated to the same viewpoint of y) from the third-person viewpoint subset of

the same subject. These partial pose labels are then combined with $\{y^j | j \in J_2\}$ to form the whole pose labels. Those pose labels whose hand part labels are the same as that of its original label y are selected.

Chapter 4

Hierarchical Deep Net with Spatial Attention

4.1 Motivation

Estimating 3D hand pose in a hierarchy, i.e. estimating each partial pose from the wrist joint to the finger tips, where the high-dimensional output space is decomposed into smaller ones, has been shown effective. However, existing hierarchical methods mainly focus on the decomposition of the output space while the input space remains almost the same along the hierarchy. Therefore, a spatial attention mechanism is proposed to transform both the input(and feature space) and the output space of the discriminative learning, which greatly reduces the viewpoint and articulation variations under the hierarchical structure.

Due to the sequential estimation, the errors accumulate. To mitigate the error accumulation, two measurements are proposed 1) cascaded regression is incorporated into the hierarchy to refine the estimation of each layer in the hierarchy; 2) hierarchical PSO forces the kinematic constraints to the results of the discriminative

methods.

4.2 Overview

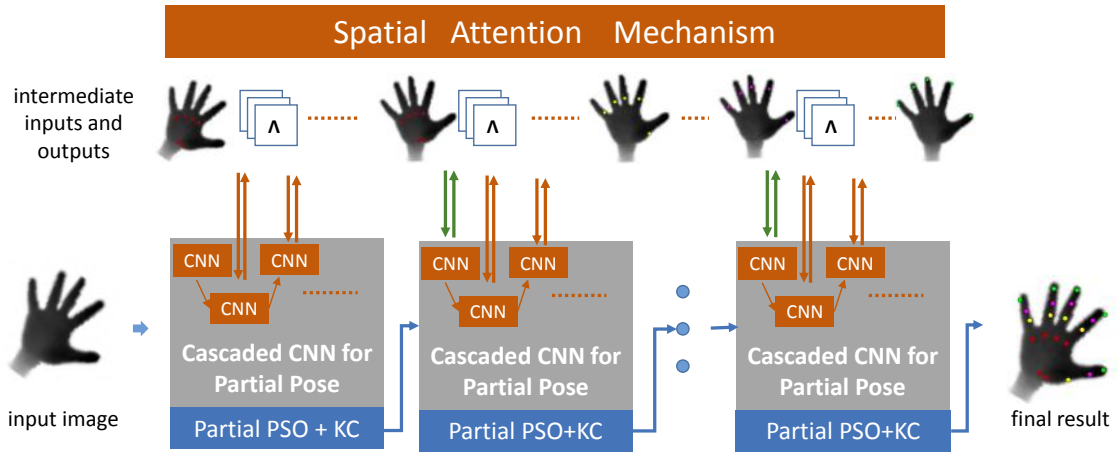


Figure 4.1: Structure of the proposed method. The Spatial Attention Mechanism integrates the cascaded and hierarchical hand pose estimation into one framework. The hand pose is estimated layer by layer in the order of the articulation complexity, with the spatial attention module to transform the input/feature and output space. Within each layer, the partial pose is iteratively refined both in viewpoint and location with the spatial attention module, which leads both the feature and output space to a canonical one. After the refinement, the partial PSO is applied to select estimations within the hand kinematic constraints (short as KC in the figure) among the results of the cascaded estimation. Λ denotes the CNN feature maps.

Under the hierarchical regression, Sun *et al.* [Sun et al., 2015] refine the hand pose by two levels of a hierarchy (palm, and fingers) in a cascaded manner by viewpoint-invariant pixel difference features in random forest while the discriminative and generative methods are combined in a hierarchy [Tang et al., 2015]: in each layer, random forests are first used to regress partial poses and a partial joint energy function is introduced to evaluate the results and select the best one to the next layer. The hierarchical optimization with refinement that estimates the hand pose in the order of articulation complexity of the hand is a promising framework as the

searching space is decomposed into smaller parts and the refinement leads to more accurate results.

However, the method in [Sun et al., 2015] and the discriminative part of HSO [Tang et al., 2015] only focus on breaking down the complexity in the output space hierarchically, i.e., decomposing the hand variables; in other words, the hierarchical strategy is carried out in the output space while the input space or the feature space stays the same along the hierarchy. For the cascaded refinement [Sun et al., 2015; Oberweger et al., 2015b], the input or feature space is only partially updated with results from previous stages, either by cropping or rotating, and the features [Sun et al., 2015] are computed on the original whole images in each iteration. In addition, the optimization of the energy function is performed in a brute force way in [Tang et al., 2015].

In this Chapter, we propose a hybrid method with iterative (cascade) refinement for hand pose estimation, illustrated in Fig 4.1, which not only applies the hierarchical strategy to the output space but also the feature space of the discriminative part and the optimization of the energy function of the generative part.

For the discriminative part, a spatial attention mechanism is introduced to integrate cascaded (with multiple stages) and hierarchical (with multiple layers) regression into a CNN framework by transforming both the input (and feature space) and the output space. In the transformed space, the viewpoint and articulation variations of the feature space and the output space is largely reduced, which greatly simplifies the estimation. Along the hierarchy, with the spatial attention mechanism, the features for the initial stage of each layer are spatially transformed from input images based on the estimation results of the last stage of the previous layer. Within each layer, the features are iteratively updated by the spatial attention mechanism. By this

dynamic spatial attention mechanism, not only the most relevant features for the hand variable estimation are selected but also the features are transformed to a canonical, expected viewpoint gradually, which simplifies the estimations in the following stages and layers. As such, discriminative features are extracted for each partial pose estimation in each iteration. In this way, we learn a deep net with spatial transformation tailored towards our hand pose estimation problem.

In the generative part, the optimization organized in the hierarchy prevents error accumulation from previous layers. Between the levels of the hierarchy, partial PSO with a new energy function is incorporated to enforce hand kinematic constraints. It generates samples under the Gaussian distribution centered on the results of the discriminative method, and selects estimations within the hand kinematic constraints. The search space of the generative method is largely reduced by estimating partial poses.

To evaluate our method, extensive experiments have been conducted on three public benchmarks. The experimental results show that our method significantly outperforms state-of-the-art methods on these datasets.

4.3 Related Work

4.3.1 Feature Selection with Attention

Learning or selecting transformation-invariant representations or features by neural networks has been studied in many prior works and among them, attention mechanism has gained much attention in object recognition and localization recently.

Girshick *et al.* [Girshick et al., 2014] produce region proposals as representations for CNN to focus its localization capacity on these regions instead of a whole image. DRAW [Gregor et al., 2015] integrates a spatial attention mechanism mimicking that of human eye into a generative model to generate image samples in different transformations. Sermanet*et al.* [Sermanet et al., 2014] use an attention model to direct a high resolution input to the most discriminative regions to do fine-grained categorization. An end-to-end spatial transformation neural network is proposed in [Jaderberg et al., 2015].

The attention mechanism is tailored to our highly articulated problem by breaking down to the viewpoint and articulation complexity in a hierarchy and refining estimation results in a cascade. The hierarchical structure with cascade refinement enables us to use a spatial transformation to not only select most relevant features as in prior works aforementioned and also transform the feature and the output space into a new one which leads to our expected, canonical space.

4.3.2 Cascaded and Hierarchical Estimation

The cascaded regression strategy has shown good performances in the face analyses [Zhao et al., 2014; Zhu et al., 2015b], human body estimation [Dollár et al., 2010; Toshev and Szegedy, 2013] and hand pose estimation [Sun et al., 2015; Oberweger et al., 2015a] and in most of these works, the features are hand-crafted, such as pixel difference features [Dollár et al., 2010; Sun et al., 2015], landmark distance features [Zhao et al., 2014], SIFT [Xiong and Torre, 2013]. Oberweger *et al.* [Oberweger et al., 2015a] use CNN to learn features automatically but with only partial spatial transformations by cropping input images and in another work [Oberweger

et al., 2015b], they use the images generated by CNN as the feedback to refine the estimation. Sun *et al.* [Sun et al., 2015] refine the hand pose using pixel difference features updated for viewpoints of whole images in each iteration. The features in both works are partially transformed either by cropping patches from the input images or rotating features calculated from the whole image. On the other hand, a hierarchical strategy that estimates hand poses in the order of hand articulation complexity achieves good performance [Tang et al., 2015; Sun et al., 2015]. HSO [Tang et al., 2015] estimates partial poses separately in the kinematic hierarchy while the input space remains unchanged. Sun *et al.* [Sun et al., 2015] estimate partial poses holistically in two layers of a hierarchy by calculating rotation invariant pixel difference features from the whole image.

Our proposed method fully transforms the feature space and the output space together in both cascaded and hierarchical manner. For each iteration of the cascade, no new features are learned as features are obtained by a spatial transformation applied to the feature maps of an initial stage. For the hierarchy, only a small region which has been transformed to a canonical view is fed into CNN. In this way, the hierarchical and cascaded strategy is not only applied to the output space as in prior work but also the transformed input and feature space.

4.4 Method overview

Hand pose estimation is to estimate the 3-D locations of the hand's 21 key joints y given depth image x, which is normalized by the size of the depth image. The ground truth of y is denoted as y^*. In our approach we divide the 21 joints into four layers $\{y_l\}_{l=0}^{3}$ where the value of l is also the order of our hierarchical estimation,

see Fig 4.2. For each layer l, j is used to denote a single joint on one finger, in the order from thumb to pinky with the number starting from 1 to 5 (for the wrist joint in the first layer, j is 0). With all the definitions, the hand variables to be estimated are expressed as $\{\{y_{lj}\}_{j=0}^{5}\}_{l=0} \cup \{\{y_{lj}\}_{j=1}^{5}\}_{l=1}^{3}$.

Our method estimates (and trains) 4 layers sequentially with the spatial attention mechanism(see Sec 4.5.1) linking the layers by transforming the input (and feature) and output space interactively and partial PSO enforcing kinematic constraints to the CNN prediction, which is shown in Fig 4.1. In each layer l, the estimation is refined iteratively by learning the residual of the ground truth y_{lj}^{*} to the results y_{lj}^{k-1} of the previous stage, where k denotes the k^{th}cascaded stage (for details, see Sec 4.5.2). The spatial transformation modules are applied to the feature maps from the initial stage of the cascade and the outputs of stage $k-1$ to get aligned attention features and output space for the learning of residual of stage k.

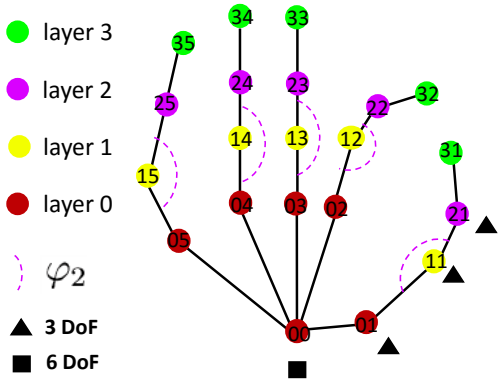


Figure 4.2: Hand model. 21 joints are divided into four layers, each joint overlaid with index number. φ_2 is the bone rotations for five joints in Layer 2.

The result $y_{lj}^{K_l}$ of the final stage K_l is fed into the post-optimization process using PSO for initialization. The partial PSO(see Sec 4.6) is introduced to enforce kinematic constraints to the results from the cascaded estimation and refine the partial pose. Along with PSO, we adopt the hand bone model (Fig 4.2), which has 51 DoFs: layer 0 has 6 DoFs, denoting the global orientation (represented by a 4-D unit quaternion) and global location (3 DoFs); each of layer 1, 2, 3, has 15 DoFs,

denoting the five bone rotations. Our hand model fixes the six joints on the palm and keeps the bone lengths of the fingers.

The optimal of the PSO is passed to layer $l+1$. Before the estimation of the next layer, the spatial attention mechanism is applied on input images and estimation results of current layer (and the ground truth for next layer during training).

4.5 Partial Pose Estimation by Spatial Attention Deep Net

4.5.1 Spatial Attention Mechanism for Hand Space

Before the elaboration of the hand pose estimation, the mechanism of spatial attention is explained. For notational simplicity, we skip the layer index l and the stage index k as the mechanism is applied to all layers and all stages similarly. The inputs of the spatial attention module are the estimation result of y_j , where j denotes j_{th} joint in the layer, and the features maps of CNN (and input images), denoted by $\Lambda \in \mathbb{R}^{W \times H}$.

The spatial module A, illustrated in the left figure of Fig 4.3, can be split into two parts: the calculation of rotation T and the pixel mapping Φ. The global in-plane rotation θ (see the right figure of Fig 4.3) is the angle between the vector of the wrist joint (joint 00 in Fig. 4.2) to the root joint of middle finger (joint 03 in Fig. 4.2) in Layer 0 and the vector $(0, 1)$ representing the upright hand pose and can be expressed as $\theta = T(y_3, y_0)$. For the other layers l $(l > 0)$, the rotation is obtained from Layer 0.

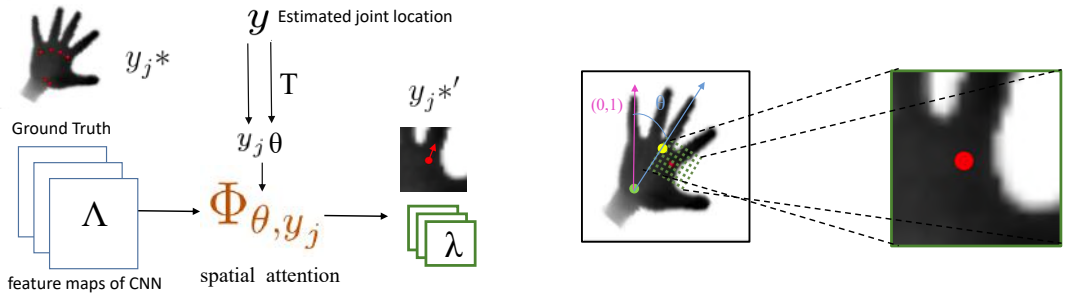


Figure 4.3: Spatial attention mechanism. Left: both the feature maps, and ground truth in training are transformed to a new space by Φ_{θ,y_j}. The locations can be transformed back by the inverse function $\bar{\Phi}_{\theta,y_j}$. The parameters (θ, y_j) of the transformation module are determined by the estimated joint locations. Right: the illustration of θ and y_j(the rot dot).

For the pixel mapping, displayed in the right figure of Fig 4.3, in which *pixel* here means an element of the feature maps (and input images), we use (u^i, v^i) to denote a pixel on the input feature map Λ and (u^o, v^o) on the output feature maps $\lambda \in \mathbb{R}^{W' \times H'}$. For the deep features for joint j, the translation parameter is the uv coordinates of y_j on the feature map(Λ) coordinate system, i.e. (t_u, t_v). The mapping between (u^i, v^i) and (u^o, v^o) is

$$\begin{bmatrix} u^i \\ v^i \end{bmatrix} = \begin{bmatrix} b \cdot \cos(\theta) & b \cdot \sin(\theta) & t_u \\ -b \cdot \sin(\theta) & b \cdot \cos(\theta) & t_v \end{bmatrix} \begin{bmatrix} u^o \\ v^o \\ 1 \end{bmatrix} \tag{4.1}$$

$$(\lambda, y*') = \Phi_{\theta,y_j}(\Lambda, y*) \tag{4.2}$$

where (u^i, v^i) and (u^o, v^o) are normalized by its corresponding width and height of the input and output feature maps. b is the ratio of the width of λ to the width of Λ (or the height as we keep the aspect ratio). If b is 1, the transformation is

rotation and translation. When b is less than 1, the transformation allows cropping and the cropping size is the same as the size of the output feature maps λ. The parameters of the transformation The setting of b is decided by the estimation from the proceeding regressor for each joint in each frame while b stays constant, which is manually set and Section 4.5.3 gives some guidance in the setting.

Once we get the transformation parameters, the mapping between λ and Λ are established by interpolating the pixel values. We also apply the transformation to the ground truth y^* during training by Eq.4.1 (only on their uv coordinates, the value of the d coordinate remains unchanged). All the inputs are in a new coordinate system, or a new space. We use Φ_{θ,y_j} in Eq. 4.2 to wrap the mapping function in Eq. 4.1 for all the pixels on the feature maps Λ and also symbolize the transformation for y. $\bar{\Phi}_{\theta,y_j}$, denoting the inverse function of Φ_{θ,y_j}, acquired by replacing θ by $-\theta$, (t_u, t_v) by $-(t_u, t_v)$ and b by $1/b$ in Eq. 4.1, transforms the output space of CNNs to the original one.

4.5.2 Cascaded Regression

The cascaded regression, which estimates the target in stages, with an initial stage producing a coarse estimation and the following stages refining the initial estimation by learning the residual pattern, has performed well in many computer vision problems, such as facial landmarking, human body pose estimation, and hand pose estimation. To mitigate the error passing to the next hierarchical layer, we adopt the cascaded strategy in the layer but with adaptations in the transforming the input and output space by spatial attention module between the stages.

In this part, we illustrate how the cascaded regression with spatial attention module

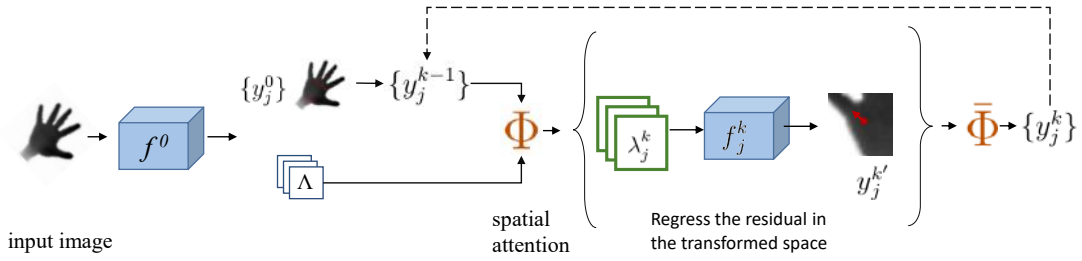


Figure 4.4: Cascaded partial pose estimation with spatial attention modules for Layer 0. f^0 provide the initial estimation while $\{f_j^k\}$ refines it by estimating the transformed residual for each joint in Layer 0. The feature maps Λ from the initial stage is transformed by spatial attention module which is parameterized by the result y^{k-1} form previous stage before feeding into the current stage k.

work in each layer.

Within each layer of the hierarchy, the joint locations $\{y_j\}$ are estimated in a cascaded manner, shown in Fig. 4.4. We leave out the layer subscript l as the cascaded regression is applied to all layers. At first, an initial CNN model ($\{f_j^0\}$) regresses the joint location $\{y_j\}$. It not only provides an initial state $\{y_j^0\}$ for the following iterative refinements but also deep feature maps Λ for other regressors. In the following stages, the joint locations are refined iteratively. Between the refinement stages, the spatial attention modules Φ transform the deep feature maps Λ to a new space based on the estimation result $\{y_j^{k-1}\}$ from the previous stage to achieve viewpoint-invariant and discriminative features for the following regressors.

For a certain joint j in stage k, the features λ_j^k is mapped from Λ by $\Phi_{\theta^{k-1}, y_j^{k-1}}$, where the y_j^{k-1} is the result of the previous stage and θ^{k-1} is calculated by $T(y_0^{k-1}, y_3^{k-1})$ (the updating of θ happens only in Layer 0, and for other layers the value of θ is fixed after Layer 0). At the same time, the ground truth y_j^* is transformed by the module, resulting $y_j^{k*'}$ which is the transformed residual for training the k stage. Note each y_j consists of (u, v, d), the transformation is only for u and v. For d, the

residual is the third dimension of $y_j^* - y_j^{k-1}$.

Therefore, all the inputs for the regressor f_j^k in stage k of joint j are in a transformed space. After training or testing, the output of the regressor is then transformed back by $\bar{\Phi}_{\theta,y_j}$. For the joint j, the process is repeated until a satisfactory result is achieved (seen Sec. 4.8 for the choice of cascaded stages) and we use K_l to denote the final stage for Layer l. For other joints, the refinement is carried out in parallel with the same process.

The above refinements for a single joint can be mathematically expressed as

$$(\lambda_j^k, y_j^{k*'}) = \Phi_{\theta^{k-1}, y_j^{k-1}}(\Lambda, y_j^*) \tag{4.3}$$

$$y_j^k = \bar{\Phi}_{\theta^{k-1}, y_j^{k-1}}(f_j^k(\lambda_j^k)) \tag{4.4}$$

where Eq. 4.3 is the spatial attention mechanism which transforms all the inputs of stage k to a new space and Eq. 4.4 produce estimation $y_j^{k'}$ for the residual by $f_j^k(\lambda_j^k)$ in the transformed space. When the inverse transformation $\bar{\Phi}_{\theta^{k-1}, y_j^{k-1}}$ is applied to the residual, the location for the joint is updated.

4.5.3 Hierarchical Regression

In the cascaded regression part, the refinement of the location estimation for the joint within the layers are elaborated and in this part, we will see how the refinement results are connected between the layers, i.e. how the input and output for the current layers are constructed from the results from the previous layer.

For the regression in layer 0, all the joints in the initial stage are learned together

in order to keep the kinematic constraints among them as the values of these joints are highly correlated. The input of the initializor f_0^0 is multi-resolution images x: the original image and the images downsampled from the original one by the factor of 2 and 4. The output is the joint locations. In the cascaded stages of Layer 0, the input and feature space of the regressors for different joints are updated separately by the spatial function $\Phi_{\theta^{k-1},y_{0j}^{k-1}}$. The regressor in the stage k refines the estimation result y_{0j}^{k-1} in the previous stage $k-1$ by estimating the residual in a new space which are transformed back by $\bar{\Phi}_{\theta^{k-1},y_{0j}^{k-1}}$. The cascaded regression stop in stage K_0. The whole refinement stages are the same as in Section 4.5.2.

For the hierarchical estimation in layer $l(l > 0)$, the inputs are multi-resolution input images x, the estimation result $\{y_{l-1,j}^{K_{l-1}}\}$ from the previous layer $l-1$ and the viewpoint estimation θ^{K_0} from layer 0. For notational simplicity, we denote θ^{K_0} as θ and skip the joint index j. θ is fixed for all layers ($l > 0$) and the same process is applied to all the joints separately.

The input space for the initializor f_l^0 of layer l is transformed from multi-resolution images x by the spatial attention module. The mapping is

$$(x', y_{l-1}^{K_{l-1}*'}) = \Phi_{\theta,y_{l-1}^{K_{l-1}}}(x, y_l^*) \tag{4.5}$$

so the input for the initializor f_l^0 is patches x' cropped from multi-resolution input images I centred at $y_{l-1}^{K_{l-1}}$ and its corresponding coordinates in the downsampled images, and rotated by θ. The offset labels for training f_l^0 is $y_{l-1}^{K_{l-1}*'}$, which is the rotated and scaled offset of $\Delta y_l^* = y_l^* - y_{l-1}^{K_{l-1}}$.

The Δy_l^* is equivalent to the sum of the ground truth offset $y_l^* - y_{l-1}^*$ and the remaining residual of the previous layer $y_{l-1}^* - y_{l-1}^{K_{l-1}}$. This implies the initializor f_l^0

not only predicts the joint offsets of the current layer to the previous layer but also corrects the residual errors of the previous layer.

The initializor f_l^0 provides the initial offset state $y_l^{0'}$ and feature maps Λ for the refinement stages. For the refinement stages, the procedure is the same as discussed in Sec 4.5.2. The only difference from Sec 4.5.2 is that the viewpoint is static, whose value is the result of the final cascaded stage in Layer 0. As feature maps Λ has already been transformed by rotation in the initial stage; thus for the stage k, the feature space is transformed and updated by the function $\Phi_{0,y_l^{k-1}}$(no rotation transformation) and the output space is transformed with $\Phi_{\theta,y_l^{k-1}}$ (rotation and translation transformation).

The parameter b in the spatial attention module needs to be set. Here we give a reference to roughly set the scaling factor. For the initial stage (except the initial stage of layer 0), it is set according to the offset range. The u v coordinate of the ground truth y_l^* of layer $l-1$ is first transformed by $\Phi_{\theta,y_{l-1}^{K_{l-1}}}$ with $b=1$. The transformed value is denoted by u_l^{tmp}, v_l^{tmp}. b is set to be two or three times of $max(mean(\{|u_l^{tmp}|\}), mean(\{|v_l^{tmp}|\}))$, where $\{|u_l^{tmp}|\}, \{|v_l^{tmp}|\})$ refers to all the absolute offsets in the training set.

For the refinement stages, the method is similar. The difference is that b is set according to the residual range of the estimation results in the initial stage.

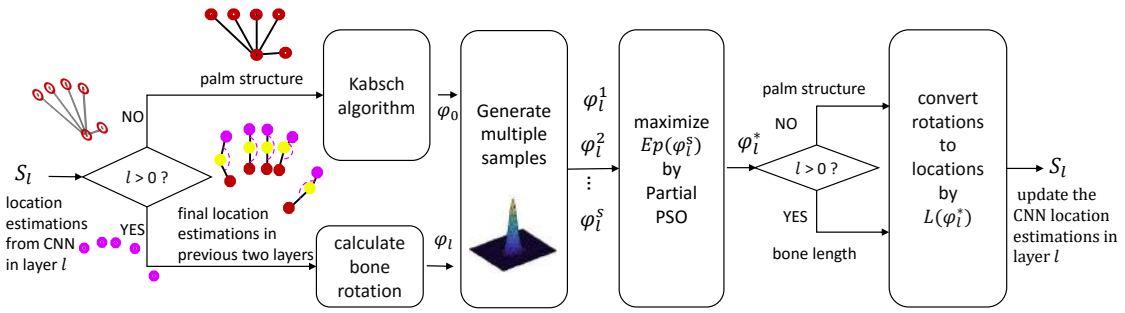


Figure 4.5: Pose refinement with partial PSO enforcing kinematic constraint. Given palm spatial structure and layer 0's location estimation by CNN, we first inference the φ_0 using Kabsch algorithm [Kabsch, 1976], and then find φ_0^* maximizing the energy function by partial PSO. The rotation φ_0^* is converted to locations using the palm structure to update the CNN estimation result. For other partial pose $\varphi_l(l > 0)$, the optimization is the same with layer 0 while the inference for the initialization of the optimization is calculating the bone rotation and the conversion back to locations uses the bone length.

4.6 Partial PSO with Kinematic Constraints for Final Refinement

Due to the error accumulation in the hierarchical regression, a post sampling and evaluation process is introduced to remove bad estimation. For each layer, based on our discriminative part's prediction, we explicitly introduce partial kinematic constraints with Particle Swarm Optimization.

Particle Swarm Optimization (PSO) is a stochastic optimization algorithm introduce by Kennedy and Eberhart [Kennedy and Eberhart, 1995] in 1995, originated in the social behaviors' studies of synchronous bird flocking and fish schooling. The original PSO algorithm has been modified by several researchers to improve its convergence properties and search capabilities. We adopt the variant of PSO with an inertia weight parameter [Shi and Eberhart, 1998].

Our whole hand pose for PSO is defined as $\{\varphi_0, \varphi_1, \varphi_2, \varphi_3\}$, where $\varphi_0 \in \mathbb{R}^7$ and

$\varphi_l(l = 1, 2, 3) \in \mathbb{R}^{15}$ are our partial poses. φ_0 consists of a 4-D unit quaternion [Oikonomidis et al., 2011a; Sharp et al., 2015] representing the global rotation and a global 3-D location of the whole hand. φ_l denotes five 3D Euler angles in layer l, each angle representing a bone rotation which is the angle between the bone connecting the joint in layer l and the corresponding joint in layer $l-1$, and the other bone connecting the joint in layer $l-1$ and the corresponding joint in layer $l-2$ (when $l-2 < 0$, the corresponding joint in layer $l-2$ is wrist). Fig. 4.2 demonstrates φ_2, five angles in layer $l = 2$.

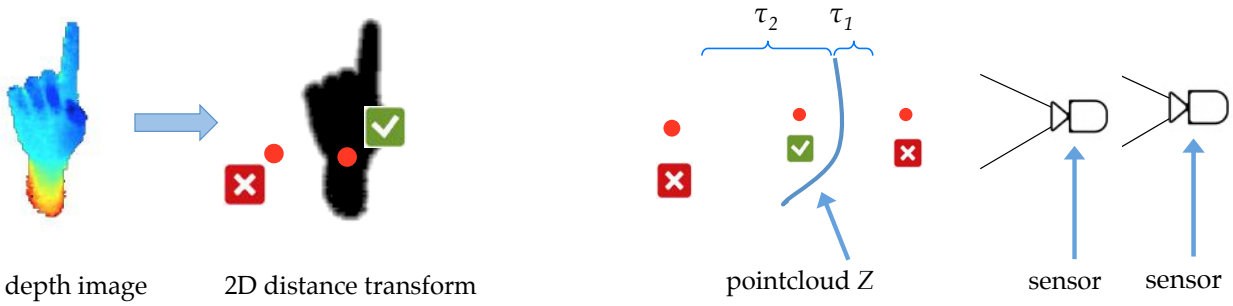
Energy Function

For each layer, PSO is used to estimate the final partial pose base on the inferred partial pose. In prior work, different energy functions are proposed to evaluate the pose hypotheses, many of which are for full poses [Sharp et al., 2015; Qian et al., 2014b; Oikonomidis et al., 2011a]. Among the full pose energy, the golden energy proposed in [Sharp et al., 2015] renders a hand depth image from a hypothesis and compares it with the input image. It is effective in penalizing the incorrect poses. However, the rendering consumes much computation power and due to high DoF, thousands of hypotheses are required to get a good estimation. Different from the full pose energy, Tang et al. [Tang et al., 2014, 2018] propose a silver energy to evaluate partial pose hypotheses and remove bad hypotheses. The silver energy takes the form

$$E_{\text{Ag}} = \sum_{y} [B(y) + D(y)] \tag{4.6}$$

where y is a estimated 3D joint location and the energies f the joint locations forming a subset are summed up. The first term $B(y)$ forces these locations to lie into the

foreground area and the second term $D(y)$ makes the predicted depth of the joint consistent with the observed depth. The ideas of these two terms are demonstrated in Fig. 4.6. For the details of the definitions, we refer readers to the original paper [Tang et al., 2018].



(a) $B(x)$ (b) $D(x)$

Figure 4.6: The bckground term $B(y)$ (left) and the depth term $D(y)$ (right) of the silver energy. Red dots denote the estimated wrist joint locations. Tick means the estimation is accepted by the term and cross means otherwise. The figure is from [Tang et al., 2015]

From the silver energy, we designed a new energy function that applied to partial pose and explicitly taking into account the kinematic constraints. Our energy function Ep for each layer is as follows:

$$Ep(\varphi_l') = P(\varphi_l')Q(\varphi_l'), \tag{4.7}$$

where the first item, $P(\varphi_l') \propto N(\varphi_l'; \varphi_l, \Sigma)$, is the prior probability of the Gaussian sample from mean φ_l, Σ is a diagonal covariance matrix that is manually set to ensure that each parameter varies in valid ranges. We draw 100 samples for each layer in our experiments. $P(\varphi_0')$ encodes the spatial structure of the six joints on the palm and $P(\varphi_l')(l = 1, 2, 3)$ keeps the bone length information.

To acquire the prior probability $P(\varphi_0')$, we first choose Kabsch algorithm [Kabsch,

1976] to find the optimal affine transformation matrix (global translation and rotation, i.e. φ_0)from our hand model for the six joints on the palm to the CNN results, as shown in the top pipeline of Fig. 4.5. The hand model for the palm joints can be seen as the palm joint locations of an upfront reference hand pose with wrist located on the original coordinate. By generating samples from Gaussian distribution centred on φ_0 instead of y_0 from CNN which usually violates the kinematic constraint, we get the $P(\varphi'_0)$ that keeps the spatial structure of palm joints.

For $P(\varphi'_l)$ of other layers $l > 0$, we first get five bone rotations φ_l by calculating the angles which are demonstrated in the bottom pipeline of Fig. 4.5 with joint estimation locations of CNN in current layer l and the joint estimation locations from layer $l-1$ and $l-2$, and then sample from the Gaussian distribution centred on φ_l. When converting rotations to locations for evaluations of the second term, we enforce the constraint of the bone length.

The second item, $Q(\varphi'_l) \propto \sum_{y'_{lj} \in L(\varphi_l)} [B(y'_{lj}) + D(y'_{lj})]$, denotes the likelihood of all joint $\{y'_{lj}\}$ belongs to the hand, where $L(\varphi'_l)$ converts rotations φ'_l into locations $\{y'_{lj}\}$. Similar to Tang *et al.* [Tang et al., 2015] silver function, the term $B(y'_{lj})$ forces each joint joint y'_{lj} to lie inside the hand silhouette. The term $D(y'_{lj})$ makes sure joint y'_{lj} lies inside the depth range of a major point cloud.

4.7 Structures of the CNN Models

To evaluate our proposed method (Hybr_Hier_SA) and the discriminative part Hier_-SA, we implement three baselines. The first baseline (Holi) estimates the whole hand pose with a single CNN. The second baseline(Holi_Derot) consists of two steps: one

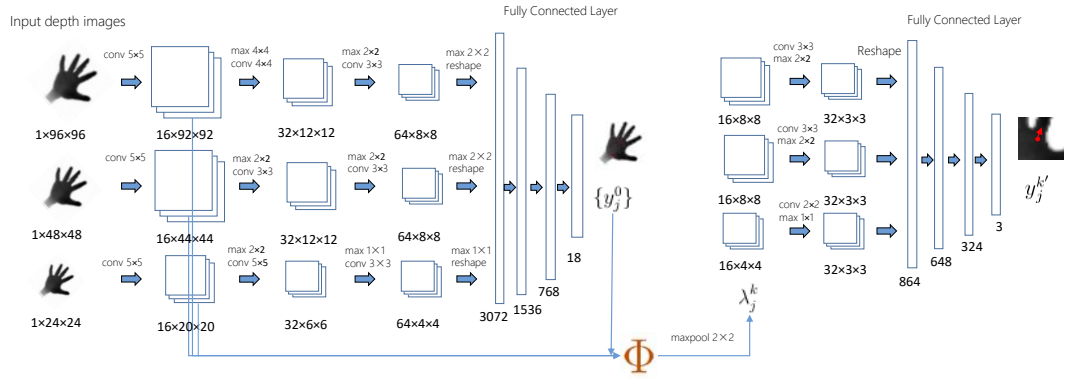


Figure 4.7: Structure of the CNN models in Layer 0 of our method. Left: the

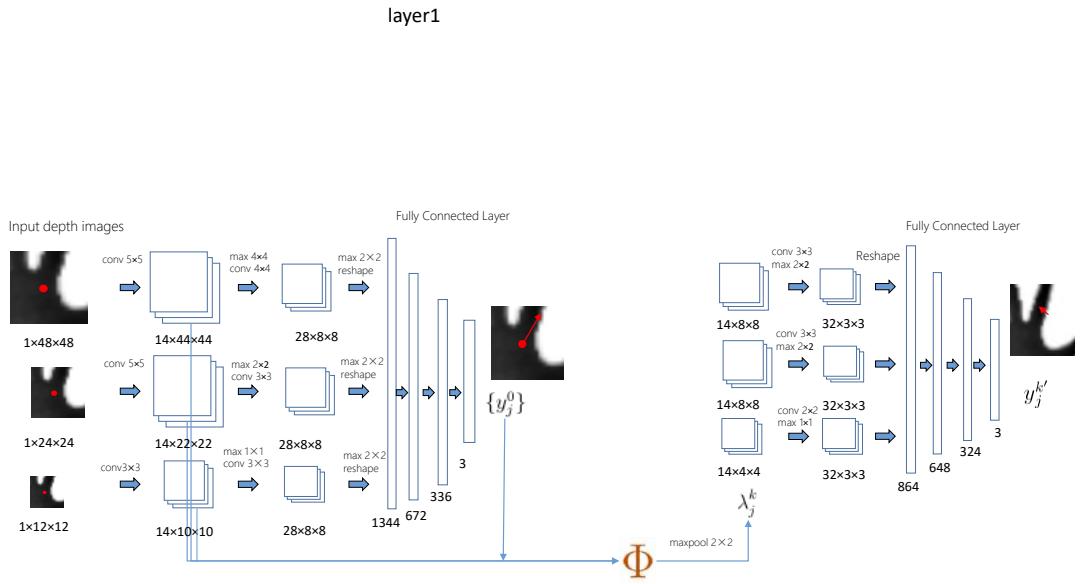
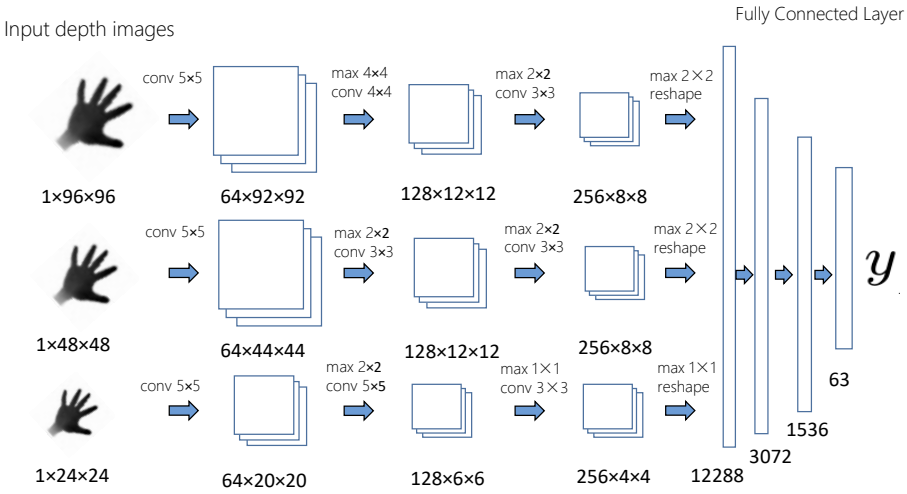


Figure 4.8: Structure of the CNN models in Layer $l(l = 1, 2, 3)$ of our method. Left: the CNN regressing a single joint in the initial stage. Right: a CNN for a single joint in the refinement stage k.

step predicting the in-plane rotation of the hand pose by a CNN and rotating the hand pose to upright view; the other step estimating the whole hand pose by another CNN. The third one (Holi_SA) is a holistic cascaded regression network without hierarchy, which initializes the whole hand pose with a CNN and refines the hand pose joint by joint via spatial attention mechanism by a set of CNNs. For fair



holi

Figure 4.9: Structure of the CNN model for the baseline Holi. The CNN model estimates all the 21 joints with multi-resolution input images.

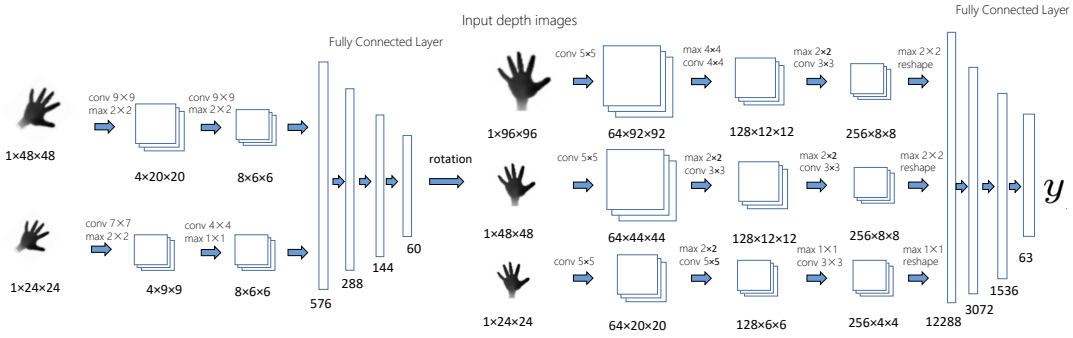


Figure 4.10: Structure of the CNN models for the baseline Holi_Derot. Left: a CNN model classifies the depth image into 60 in-plane rotation bins. Right: a CNN model estimates all the 21 joints with input images rotated by the classification result.

Holi_rot

comparison, we set the size of the parameters of the methods to be roughly the same: the parameters are stored in 32 bit float and the size of parameters is 130MB.

We present all the CNN models in our experiment section. Fig 4.7 and Fig 4.8 illustrate the CNN models used in our method. Fig 4.9 and Fig 4.10 are the CNN models for the baseline Holi and Holi_Derot. Fig 4.11 shows the overall structure

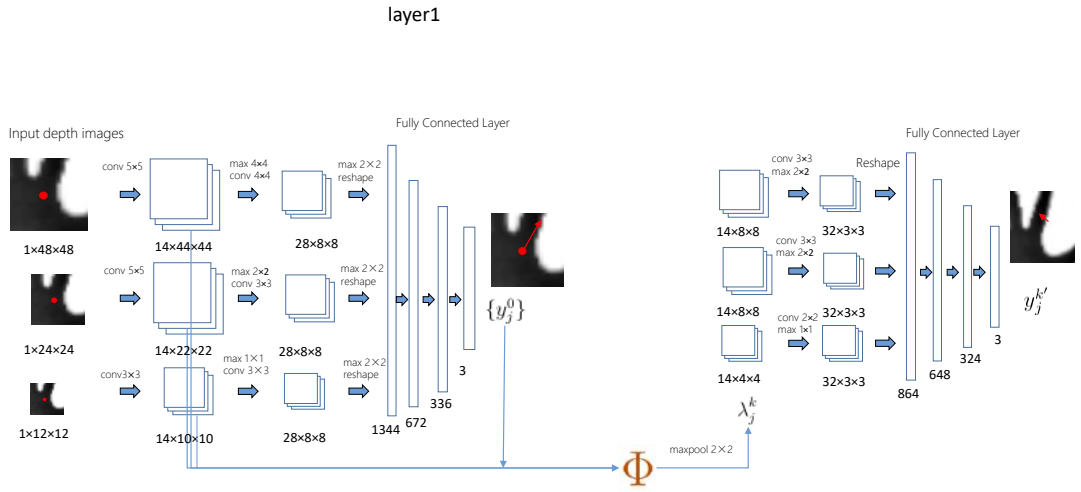


Figure 4.11: Structure of the CNN models for the baseline Holi_SA. Left: the CNN model in the initial stage. Right: a CNN model for a single joint j refinement in the stage k.

of the baseline Holi_SA and Fig 4.11 demonstrates the CNN models used in all the parts of Holi_SA.

The structures for our CNN models are based on the one proposed in [Tompson et al., 2014], where the number of convolutional layers/channels, the number of fully connected layers are changed to fit the GPU memory requirement. Instead of regressing the 2D heatmaps, we learn to regress the 3D locations directly. The loss functions for training the CNN models are the mean square error. The parameters are updated with the stochastic gradient descent, where the learning rate for different networks is chosen by searching from 0.1 to 0.00001 with a decrease factor of 10. The networks are implemented by Theano [Thies et al., 2016]. All the models are trained and tested on a PC with Intel i7, 24GB RAM and NVIDIA GeForce GTX 750 Ti.

In Fig 4.7 and Fig 4.8, the features λ_{lj}^k for the CNN $f_{lj}^k (k > 0)$ is transformed from the feature maps in the initial CNN and the transformation results are max-pooled with pool size 4×4, 2×2, 2×2 for different resolution channels. Thus, the size of

output feature maps of the spatial attention module before maxpooling is 32×32, 16×16, 8×8.

4.8 Experiment

The evaluation of our proposed method is conducted on three publicly datasets. **ICVL** [Tang et al., 2015],**NYU** [Tompson et al., 2014], **MSHD** [Sharp et al., 2015].

We compare the results of different methods by the proportion of joints within a certain maximum error of the distance of the predicted results to the ground truth [Sharp et al., 2015].

The number of iterations K is set on the observation of the error saturates after a certain stage in the cascaded stages, shown in the first figure of Fig. 4.12. The palm joint error almost saturates after one refinement while the finger joint achieves a fine estimation in the initial stage. Though more refinement stages can help to get lower error, in order to have a good balance between the accuracy and the memory consumption, we set K_0 for Layer 0 to 1 and $K_l, l > 0$ to 0, which gives us . For our partial PSO, we generates 100 samples for each layer, and iterate 5 generations.

4.8.1 Self-comparison

On all the datasets, Hier_SA outperforms the baselines significantly, see Fig. 4.13. The improvement margin is related to the range of viewpoints and the complexity of articulations. The in-plane rotation distributions of original data, the ones after the initial stage and the first stage are shown in Fig. 4.12. As MSHD dataset covers

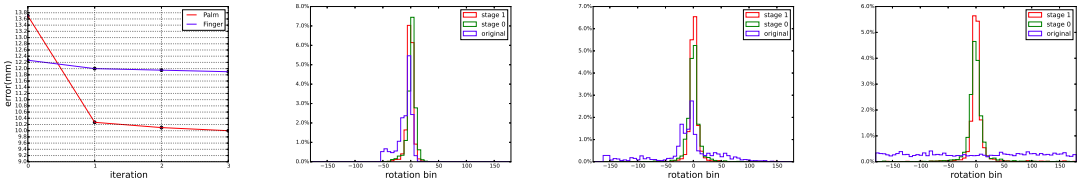


Figure 4.12: First: Errors for a joint on the palm y_{00} and a joint on the middle finger y_{13} for 4 stages. Second, third and forth: In-plane viewpoint distribution of testing set for different stages on ICVL, NYU, MSHD respectively. The blue, green and red line corresponds to the in-plane rotation distribution of original ground truth, ground truth rotated after initial stage and the first stage. The rotation estimation error after the initial stage and the first stage is 5.9 and 4.4 for ICVL, 8.0 and 6.1 for NYU, 10.9 and 9.2 for MSHD in the unit of degree.

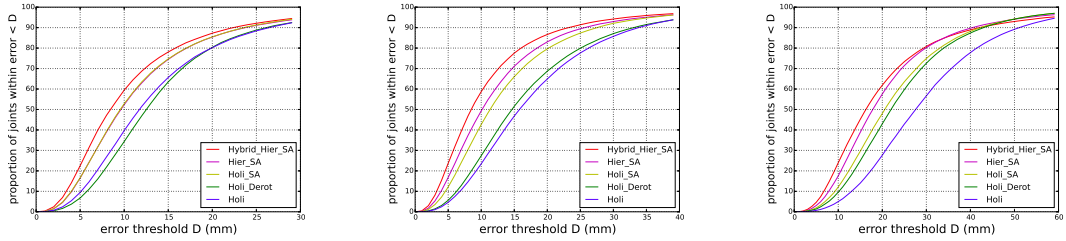


Figure 4.13: Comparison of different methods on three datasets (the higher the curve is, the better the performance is). Left:ICVL; Middle: NYU; Right: MSHD. Our proposed method, Hybr_Hier_SA, achieves best performance when compared with the baselines. Holi represents regressing all joints in a single CNN network. Holi_Derot rotates input images to a viewpoint by estimating in-plane rotation before being fed into Holi. Holi_SA refines the results of Holi using spatial transformation. Hier_SA estimates the joint locations in a kinematic hierarchy, which is our proposed method without partial PSO.

a full range of viewpoints and articulations, the improvement on this dataset from the baseline Holi is the largest. For example, the percentages of frames under 20mm on ICVL, NYU and MSHD are improved by Hier_SA with margins of 5%, 18% and 30% respectively, compared to that of Holi.

The curves of Hier_SA and Holi_SA on three datasets illustrate the efficacy of hierarchical strategy in conquering the articulations, while the curves of Holi_SA, Holi and Holi_Derot show that spatial attention mechanism is effect in reducing the view-

point and articulation complexity. By refining viewpoints with stages and spatially transforming the feature space to focus on the most relevant area for a certain joint estimation, Holi_SA achieves better results than Holi and Holi_Derot. Note that the curve of Holi_Derot is under that of Holi on ICVL dataset, which implies that the error of estimating the rotation by a separate network may deteriorate the later estimation when the variations of the viewpoint in training set is small.

Hybr_Hier_SA further improves the result of Hier_SA by a large margin consistently on all the datasets, which verifies that the kinematic constraints by the partial PSO is effective.

4.8.2 Comparison with Prior Works

We compare our work with 4 state-of-the-arts methods: Hierarchical Sampling Optimization(HSO) [Tang et al., 2015], Sharp *et al.* [Sharp et al., 2015], HandsDeep [Oberweger et al., 2015a], FeedLoop [Oberweger et al., 2015b] on three datasets, see Fig. 4.14. The former two are hybrid methods and the the latter two are refinement method based on CNN. The results are obtained either from the authors for HSO [Tang et al., 2015] or from the reported accuracies [Sharp et al., 2015; Oberweger et al., 2015a,b]. On ICVL and NYU dataset, many methods report the number of frames whose mean errors are below certain threshold while on MSHD dataset, the compared methods use the number of joints whose errors are below certain threshold. The former metric is slightly stricter than the latter one but they are highly correlated. The examples of the estimation results of HandsDeep, FeedLoop, HSO and our method are shown in Fig. 4.15.

On ICVL dataset, we compare HSO with parameters set as $N = 100, M = 150$. Our

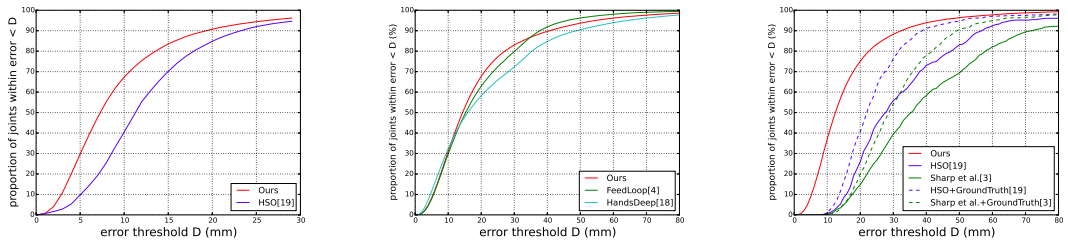


Figure 4.14: Comparison of prior work on three datasets. Left:ICVL; Middle: NYU; Right: MSHD

method is better by 26% of joints within $D = 10mm$. On NYU dataset, we compare our method with HandsDeep and FeedLoop which are all based on CNN. As the hand model of these methods are different, we evaluate the result by comparing the error of the subset of 11 joint locations(removing the palm joints except the root joint of thumb). Our estimation result is better than HandsDeep by a large margin, for example, an improvement of 10 % within $D = 30mm$, and achieves roughly the same accuracy with FeedLoop.

We finally test our method with HSO and Sharp *et al.* on MSHD dataset. The dataset is more challenging than the above two as it covers a wider range of viewpoints and articulations. The curves demonstrate the superiority of our method under large variations. For example, the proportion of joints (when $D = 30mm$) of our method is 35% and 50% more than those of HSO and Sharp *et al.* respectively. Note that our estimation is even better than the results of HSO and Sharp *et al.* using ground truth rotation [Tang et al., 2015].

4.9 Summary

To apply the hierarchy strategy to the input and feature space and enforce the hand kinematic constraints to the hand pose estimation, we present a hybrid method

by applying the kinematic hierarchy to both the input and feature space of the discriminative method and the optimization of the generative method. For the integration of hierarchical input and feature space of the discriminative, a spatial attention mechanism is introduced to spatially transform the input(and feature) and output space interactively, leading to new spaces with lesser viewpoint and articulation complexity and gradually refining th estimation results. In addition, the partial PSO is incorporated between the layers of the hierarchy to enforce the kinematic constraints to the estimation results of the discriminative part. This helps reduce the error from previous layer to accumulate. Our method demonstrates good performance on three datasets, especially on the dataset under large variations.

In this Chapter, we explore the estimation of partial poses in the kinematic hierarchy using CNN and how to achieve the best performance using the spatial attention. Beyond this, we raise a question about different existing methods under the hierarchical strategy: can we integrate and compare them in one framework?

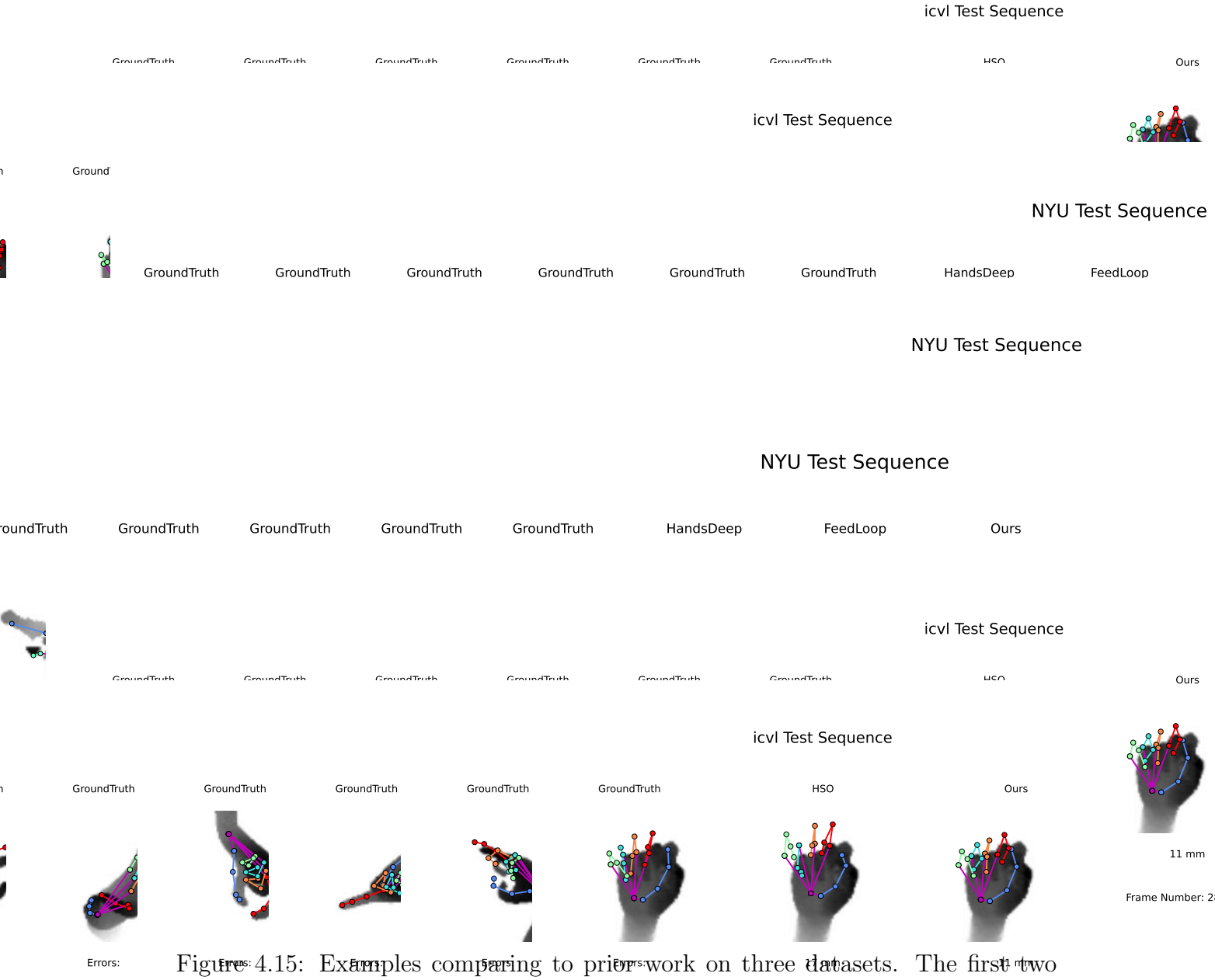


Figure 4.15: Examples comparing to prior work on three datasets. The first two rows, the middle three rows, and the last two rows are examples from ICVL dataset, NYU dataset and MSHD dataset, respectively. Compared to other methods, our method has a good performance in discriminating the fingers and a better precision. For many challenging viewpoints, our method still has a good estimation.

Chapter 5

Integration of Hierarchical Regression

5.1 Motivation

In 3D hand pose regression, [Sun et al., 2015; Tang et al., 2015] also have hierarchical kinematic structures and decompose the problem into sub-problems which can be tackled independently. The hierarchical regression in the previous chapter and these two methods differs regarding to how the hierarchy is constructed, which method is used to prevent error accumulation, whether a full pose energy is used to get the best hypothesis.

In this chapter, we integrate all these hierarchical regression into a probabilistic framework to discuss and analyse the variants, give an insight into them and explore the applications under different requirements.

5.2 Overview

Under all the hierarchical regression, the full pose y is arranged in a standard kinematic tree defined by the skeletal structure of the hand, giving the complete set of partial poses as: $y = \{y_l\}_{l=1}^{L}$.

The framework is illustrated as follows. In the testing, each layer l first estimates a conditional probabilistic distribution $P(y_l|y_{l-1}^*, ..., y_1^*, Z)$, conditioned on the best partial poses produced by the previous layers. From the distribution, a set of hypotheses for the partial pose y_l are drawn and the best one y_l^* is chosen by a energy for partial poses, for example the energies explained in Section 4.6. We use the term silver energy referring this local energy. Therefore, for each layer l, the process can be expressed as

$$y_l^* = \underset{y_l \in Y_l}{\operatorname{argmin}} E_{\mathrm{Ag}}(y_l) \tag{5.1}$$

where $Y_l = \{y_l^m\}_{m=0}^{M}$ and $y_l^m \sim P(y_l|y_{l-1}^*, ..., y_l^*, x)$. Temporal information can optionally be incorporated by additionally evaluating hypotheses from the previous frame. After layer L, a full pose hypothesis y is returned by concatenating all the best partial poses, i.e. $y = \{y_l^*\}$.

We then repeat the process from layer 1 to layer L multiple times to get a set of full pose hypotheses $\{y^n\}_{n=1}^{N}$ and the final best full pose y^* is chosen by E_{Au}, which is expressed as

$$y^* = \underset{y \in \{y^n\}_{n=1}^{N}}{\operatorname{argmin}} E_{\mathrm{Au}}(y) \tag{5.2}$$

Note that under the hierarchical structure, the estimation error in each layer will accumulate in the layers, which is an inherent problem of all hierarchical methods. The error accumulation can be alleviated by per-joint optimization (5.1) and full-pose optimization (5.2). In per-joint optimization (5.1), distributions are produced and by sampling from the distributions and evaluating the samples using E_{Ag}, we can prune hypotheses with large errors, reducing the error accumulation. In contrast, deterministic prediction provides only one hypothesis for the partial pose: when the hypothesis fails, the following estimations all fail. The technique of multiple outputs with selection has been proven in the past to be effective in reducing error [Guzman-Rivera et al., 2014]. It is embedded into each level of the hierarchy in order to reduce the accumulated error. Further, the repetition sampling and evaluation from layer 1 to layer L provides a pool of full pose hypotheses and these hypotheses are evaluated by the stricter full pose energy in (5.2). We refer this full pose energy as golden energy E_{Au}. borrowing the terminology from [Sharp et al., 2015].

5.3 Variants

Depending on the sampling and optimization used, the hierarchical regression discussed in this chapter has four variations which are listed in Table 5.1, where we follow the term HSO used in [Tang et al., 2015]. **H** stands for hierarchical, **S** for sampling, and **O** for optimization. The third column indicates whether the methods listed in the first column decompose the output pose space or not. The fourth to the final column represent the regressor used to provide the estimation for the sampling, whether multiple hypotheses are sampled, and which optimization method is used. To demonstrated the effect of each components of HSO, the baselines are

also provided in Table 5.1, holistic regression or hierarchical regression without a post-process to alleviate error accumulation.

Among the variants listed in Table 5.1, there are two kinds of the regressors: random forest and CNN. Please refer to [Tang et al., 2015] for the implementations of the random forest and Chapter 4 for the implementations of CNNs. For the energy, we use the silver energy defined in [Tang et al., 2015] for the partial pose optimisation and the golden energy defined in [Sharp et al., 2015] for the whole pose optimisation, which is briefly introduced in Chapter 4. For the optimization methods, we give a brief explanation in the following section.

Under this framework, HS_CO_P is the method we proposed in the previous chapter, using the CNN to get the distribution, sampling multiple hypotheses and searching the best hypothesis by PSO. HS_CO_N differs from HS_CO_P in the searching method, using the direct brutal-force searching. HierCNN is one baseline from previous chapter, hierarchical regression using CNN without PSO optimisation.

Sun et al. [Sun et al., 2015] use cascaded random forests to get the best hypothesis without providing the conditional distribution explicitly, sampling and evaluation of the hypotheses. The hypothesis is aggregated by all the votes in the leaf nodes, which can be represented by HierForest in Table 5.1.

Tang et al. [Tang et al., 2015] use the random forests to produce the conditional distribution and do a brutal-force optimisation, which is HS_FO_N. HS_FO_P replaces the brutal-force optimisation with PSO.

Table 5.1: Baselines and variants of HSO

	Methods	Decomp.	Regressor	Sam.	Opt.
Baseline	HolisticForest	Holistic	Random forest	No	No
	HierForest	Hierarchical	Random forest	No	No
	HolisticCNN	Holistic	CNN	No	No
	HierCNN	Hierarchical	CNN	No	No
HSO	HS_FO_N	Hierarchical	Random forest	Yes	Naive
	HS_FO_P	Hierarchical	Random forest	Yes	PSO
	HS_CO_N	Hierarchical	CNN	Yes	Naive
	HS_CO_P	Hierarchical	CNN	Yes	PSO

5.3.1 Optimizing Energy

Given the samples drawn from conditional distribution $P(y_l|y^*_{l-1}, ..., y^*_l, Z)$, to find the pose y^*_l with silver energy E_{Ag}, we adopt two strategies: naive optimization and Particle Swarm Optimization.

Naive Optimization The naive approach is a brute-force comparison: all M samples from the distributions produced by the regressors are evaluated by the silver energy and the one y^*_l which has the lowest energy is selected.

Particle Swarm Optimization Instead of selecting the best pose among the M samples all at once, our second strategy uses the samples to initialize the search process of the optimization. The best pose found in this way is not necessary among the samples drawn from the distribution.

Particle Swarm Optimization has been widely used for hand pose tracking [Oikonomidis et al., 2011b, 2012; Qian et al., 2014b] and we use it to do per-joint optimization. Different from the optimization on the whole hand pose, the PSO is performed on vector with no more than three DoF(the full pose has 26 DoF), which is very

Algorithm 1 Per-joint optimization by PSO

```
Initial a swarm of particles {y^i}_{i=1,..,S}
for iter = 1, ..., nIter do
    for each particle i = 1, ..., S do
        update particle by (5.3) (5.4)
        if E_Ag(y^i) < E_Ag(y^i_pbest) then
            y^i_pbest ← y^i
            if E_Ag(y^i_pbest) < E_Ag(y_gbest) then
                y_gbest ← y^i_pbest
```

efficient. A swarm of particles $\{y_l^i\}_{i=1}^S$ of the PSO is initialized with S samples and then the particles are updated in $nIter$ generations. In every iteration, each particle is updated by two "best" values in (5.3) and (5.4) which is evaluated by the silver energy E_{Ag}: y_{pbest} is the best value the particle has achieved so far and y_{gbest} is the best value that have been obtained by all the particles in the swarm; therefore, the new 'sampled' values are generated by the goodness of the samples in the previous iterations, where the value of a particle is treated as a sample. For other symbols, v is the particle velocity, U is a random number between (0,1) and c_1, c_2 are learning factors. The algorithm is shown in Algorithm 1.

$$v^i \leftarrow v^i + c_1 * U * (y_{pbest}^i - y_l^i) + c_2 * U * (y_{gbest} - y_l^i) \tag{5.3}$$

$$y_l^i \leftarrow y_l^i + v^i \tag{5.4}$$

Compared with the result selected from M samples in the naive optimization, the final solution of PSO can be considered as a weighted combination of the M samples.

5.4 Comparisons

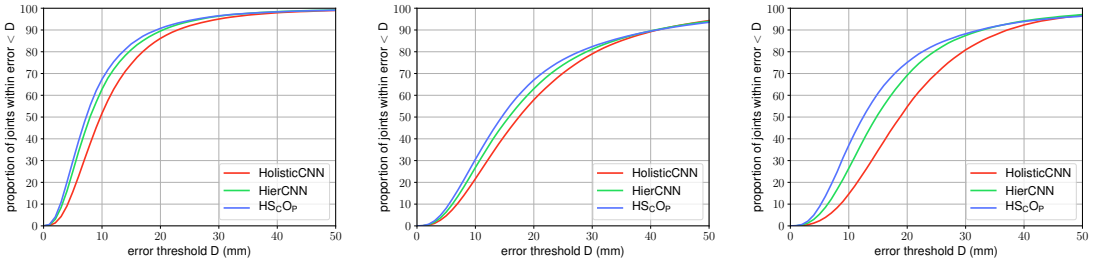
In the following sections, we conduct extensive comparisons between the variants and the baselines both in accuracy and efficiency in three aspects: the hierarchical structure, the silver energy and the optimization methods.The comparisons can provide some guidance on the selection of regressors and optimization methods under certain requirements when using the hierarchical framework.

For CNN based sampling, the architectures and the training is the same with Chapter 4. For decision forest based sampling, we follow the setting in [Tang et al., 2015] where each forest consists of 3 trees and maximum depth of each tree is 15. On MSHD, due to the diversity of the pose space, very deep trees are required to achieve a satisfactory performance, the authors quantize the viewpoints into 128 clusters and for each cluster, train a HSO.

5.4.1 Hierarchy

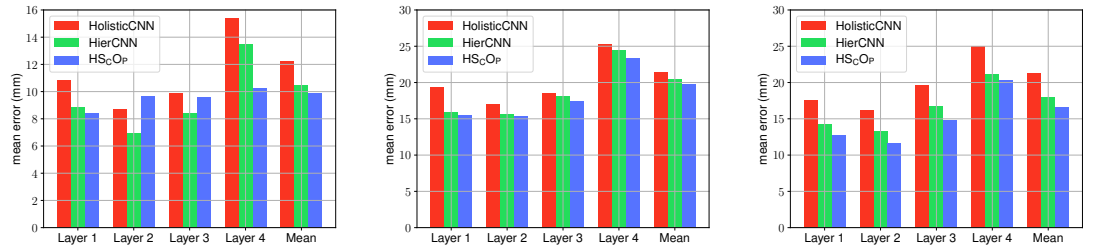
To evaluate the impact of using the kinematic hierarchy against the conventional black box methods, we propose two types of baselines. The first is a holistic regression method estimating the full pose, denoted with the prefix Holistic. The second one, denoted with the prefix Hier, estimates the partial poses in layers deterministically (without sampling)–essentially a degraded HSO with $N = 1$ and $M = 1$. These baselines are listed in Table 5.1.

For CNN-based methods, the parameters of HolisticCNN is the same (if possible) with that of HierCNN to have a fair comparison, as in Section 4.7. We use Theano [Team et al., 2016] in implementation. Models are trained with SGD. The best



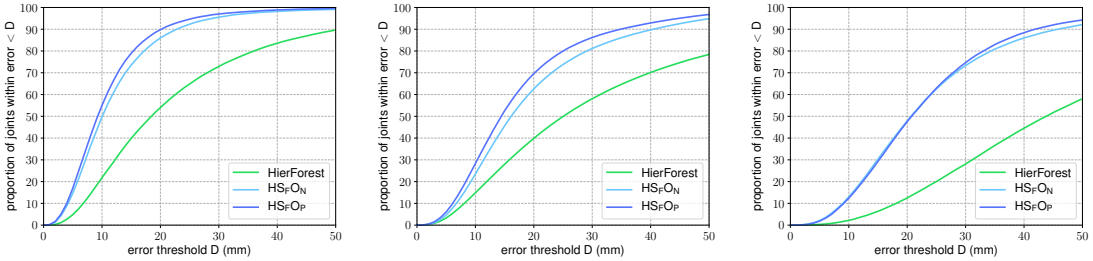
(a) ICVL dataset (b) NYU dataset (c) MSHD dataset

Figure 5.1: Self-comparisons for HSO based on CNN sampling by the proportion of joints under a certain error threshold. HS_CO_P, i.e. the variant using CNN for sampling and PSO for optimisation, outperforms the baseline, HolisticCNN, regressing all the joint locations by a single CNN and the baseline, HierCNN, regressing the joint locations in a hierarchy but without sampling and evaluation of the samples



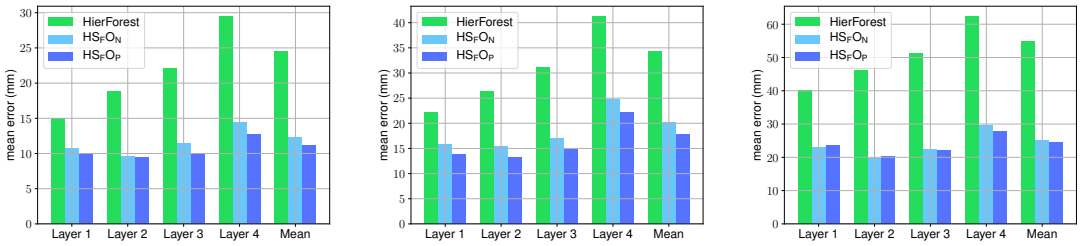
(a) ICVL dataset (b) NYU dataset (c) MSHD dataset

Figure 5.2: Self-comparisons for HSO based on CNN sampling by the mean joint errors of partial poses in layers.



(a) ICVL dataset (b) NYU dataset (c) MSHD dataset

Figure 5.3: Self-comparisons for HSO based on decision forest sampling by the proportion of joints under a certain error threshold.HS_FO_P and HS_FO_N, i.e. the variants using decision forest for sampling and silver energy for evaluation of the samples, outperform the baseline, HierForest, that regresses the joint locations in a hierarchy but without sampling and evaluation of the samples significantly.



(a) ICVL dataset (b) NYU dataset (c) MSHD dataset

Figure 5.4: Self-comparisons for HSO based on random forest sampling by the mean joint errors of partial poses in layers.

learning rate is empirically determined for different models. The experiments are conducted with a Nvidia GTX 750.

For evaluation, we adopt as a metric the proportion of joints under a maximum error threshold [Sharp et al., 2015]. Fig. 5.1 demonstrates that the curves of Hier-CNN are all above those of HolisticCNN on the ICVL, NYU and MSHD dataset by a considerable margin. Further, to observe the performance of different methods within each layer, we show the errors of estimated partial poses in different layers in Fig. 5.2. HierCNN has lower errors than that of HolisticCNN in all layers. Especially in layer 1, the errors are significantly reduced from that of HolisticCNN on all datasets. After decomposition, the output space for CNN in layer 1 is much smaller than that of HolisticCNN, resulting in an easier learning problem and the estimation in layer 1 has no error accumulation which occurs in the following layers.

The NYU training set only contains one subject while the testing set includes this same subject but also includes a new subject with a rather different shape of hand. To evaluate the generalization ability, Fig. 5.6 breaks down the errors of the two subjects. HierCNN and HolisticCNN achieve similar accuracy for the subject appeared in the training set. For the unseen person, however, HierCNN has better performance, which indicates it has better generalization ability. This is because

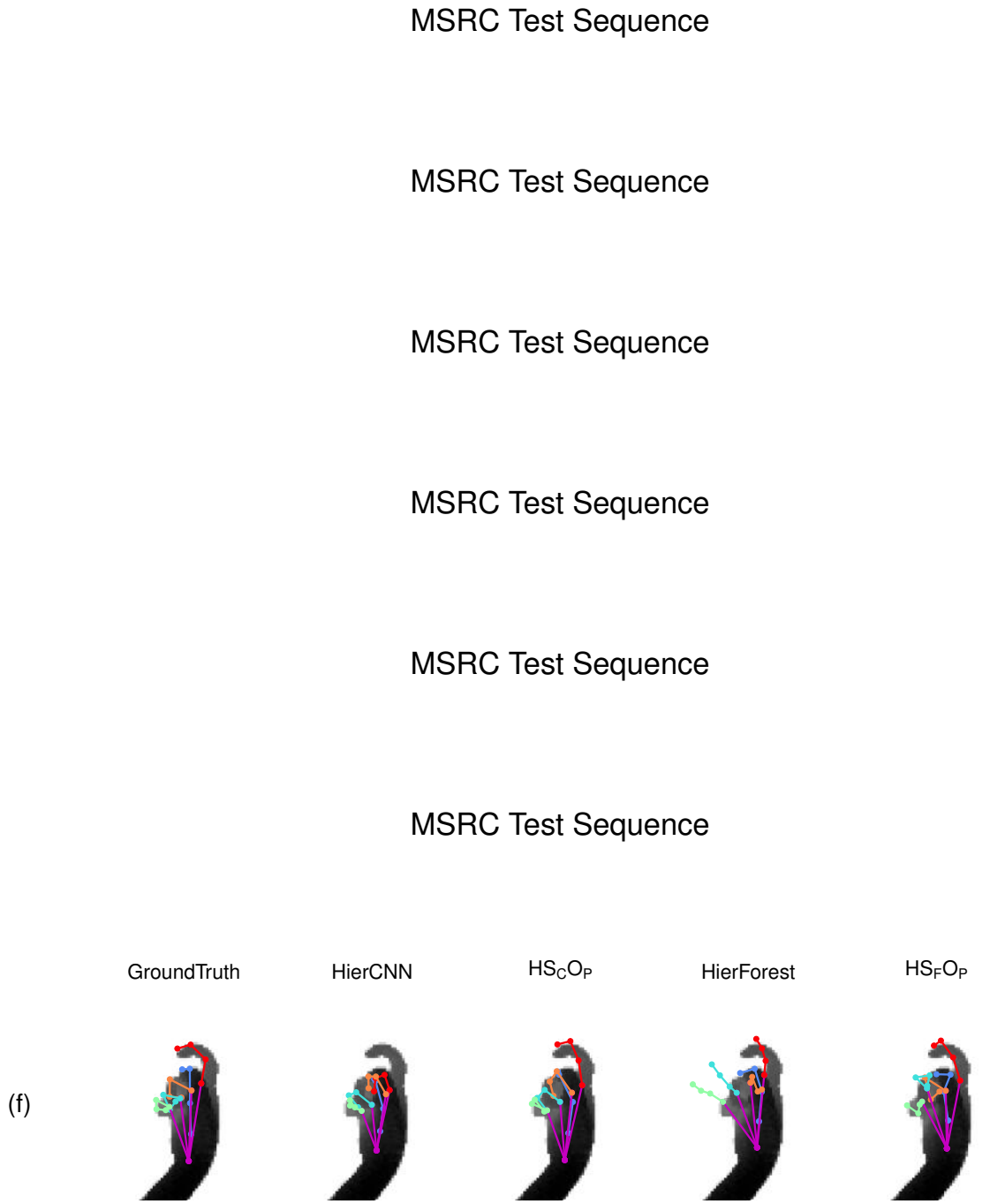
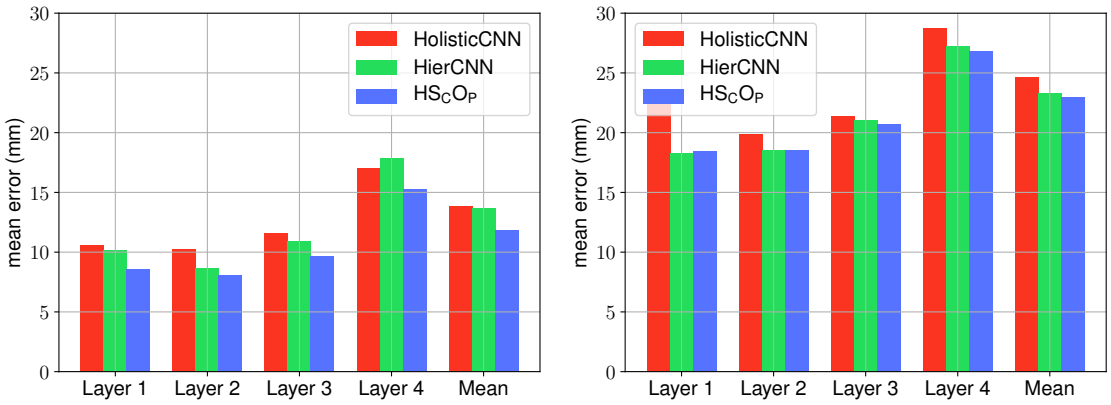


Figure 5.5: HS_COP and HS_FO_P refines the estimation of HierForest and HierCNN by the silver energy respectively on MSHD dataset(zoom out to get better visualization).

HierCNN is essentially a part-based method that trained on local appearance. On the other hand, we observe that in layer 4, the error of the seen subject produced by HierCNN is larger than that of HolisticCNN, reflecting the error accumulation problem of hierarchical structure we mentioned in Section 5.2. The estimation of



(a) seen (b) unseen

Figure 5.6: Compare different methods on images of the seen subject and the unseen subject of NYU test set.

HierCNN for the index fingers of examples (c)(e) in Fig. 5.5 shows that when the partial pose estimation fails, the following layers also fail. The error accumulation is also observed in HierForest, such as the middle finger in examples (a)(e). The errors of HierForest are shown in Fig. 5.3 and Fig. 5.4.

For the random forest based methods, a similar accuracy boost is also observed in the hierarchical baseline as opposed to the holistic one [Tang et al., 2016]. Hence the conclusion: Regardless of using a random forest or a CNN, hierarchical methods improve the estimation performance consistently, which means that the benefit of decomposing the high dimensional output space outweighs the disadvantage of error accumulation in general. In the following two sections, we demonstrate how the silver energy reduces the error accumulation and further improves the performance.

5.4.2 Silver energy

To demonstrate the efficacy of the silver energy proposed, we first compare CNN-based HSO, i.e. $\mathrm{HS_CO_P}$, with HierCNN. The curves of $\mathrm{HS_CO_P}$ in Fig. 5.1 on the three datasets are produced by setting $N = 100$, $S = 30$, and $nIter = 5$. As shown in Fig. 5.1, $\mathrm{HS_CO_P}$ further improves the estimation performance of HierCNN on all datasets. Similar improvement but much more significant is observed on all three datasets with forest-based HSO, $\mathrm{HS_FO_N}$ ($N = 100$, $M = 150$) and $\mathrm{HS_FO_P}$($N = 100$, $S = 30$, $nIter = 5$), as opposed to HierForest.

The difference in the improvement margin when applying the silver energy to HierCNN and HierForest depends on two factors. One is the quality of the estimated conditional distribution. To show the quality of a distribution, we draw an error curve whose x-axis is the number of samples drawn from the distribution and the y-axis is the lowest error of the samples. Fig. 5.7 lists two curves of the distributions produced by HierCNN and HierForest for the partial poses in layer 2, i.e. MCP joints. Results show that with the same amount of samples, the distribution estimated by HierCNN can produce samples closer to ground-truth than that of HierForest on the ICVL dataset. Note that the silver energy, is an approximation and a compromise of efficiency and accuracy. Hence there is a lower bound for the error in Fig. 5.7.

In both Fig. 5.2 and Fig. 5.4, the errors of HSO are lower than that of the HierCNN and HierForest (no sampling). Note that the larger the estimation error of HierForest and HierCNN is in a layer, the larger the error is reduced by HSO. Due to full coverage in viewpoints, articulations and shapes in the MSHD dataset, the errors of HierForest and HierCNN are larger and therefore the improvement by sampling and optimization is also larger.

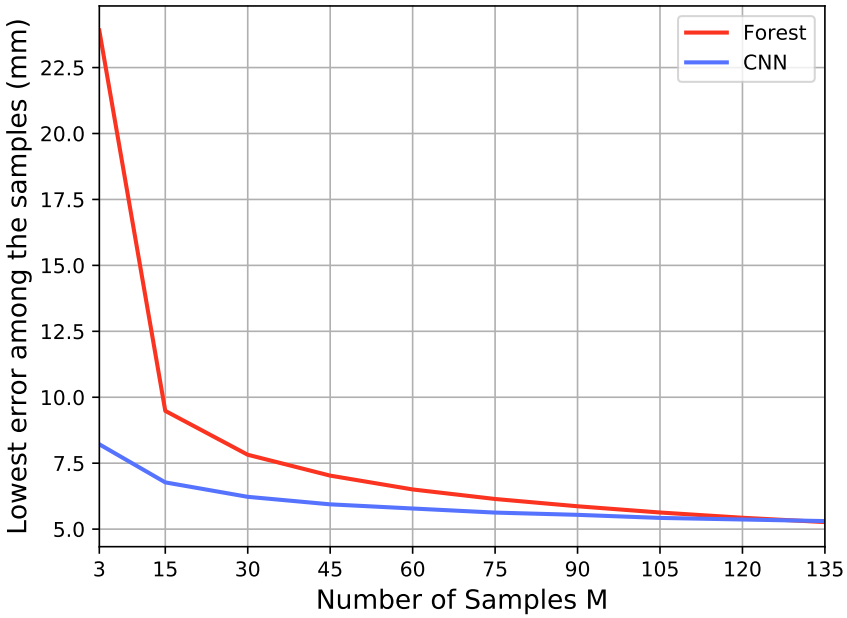


Figure 5.7: Compare samples for MCP joints drawn from the distributions produced by random forest and CNN on ICVL dataset. With the same amount of samples, the distribution estimated by HierCNN can produce samples closer to ground-truth than that of HierForest.

When the estimation of HierCNN is already close to the ground truth, adding sampling may worsen the accuracy. In Fig. 5.2a the estimation errors of HierCNN in the layer 2 and layer 3 are about 7mm and 8.5mm and after the sampling and optimization, the errors become 9.7mm, which appears that HSO worsens the estimation of HierCNN. However, although the solutions with sampling are farther from ground truth, they are still plausible configurations in most cases (note that the thickness of a finger is about 15mm). The index finger tip and the thumb tip in the first row in Fig. 5.8 is an example. The estimation of HierCNN is rather close to the ground truth while the solution given by HS_CO_P is farther but is more plausible. Also, for the joints on the ring finger in the second row, HierCNN gives results close to the ground truth that is actually a wrong label, but HSO rectifies this wrong estimation.

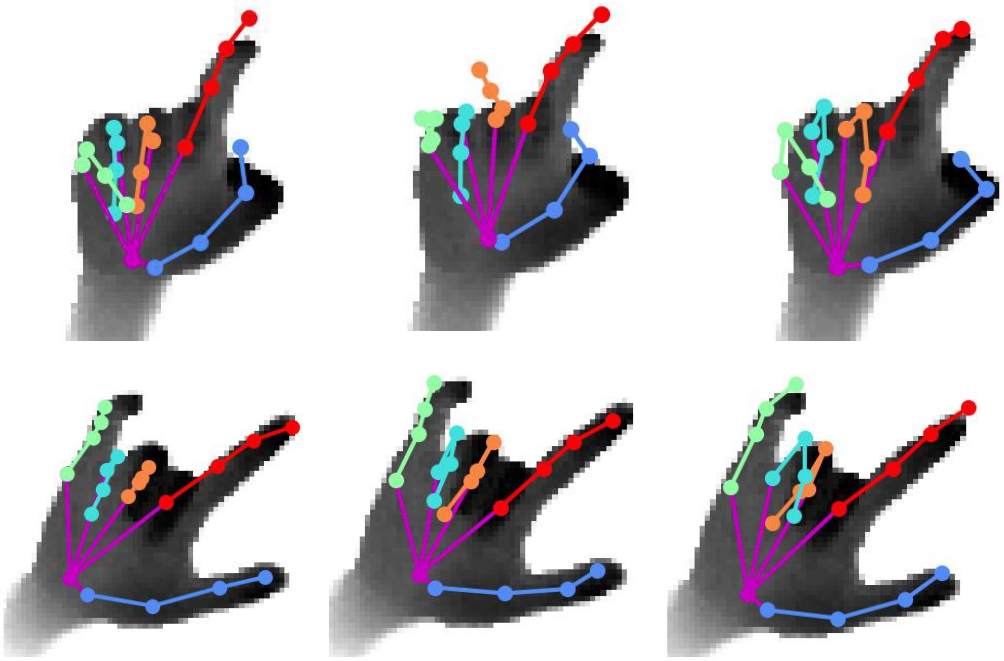


Figure 5.8: Impact of the silver energy. Left: ground truth; Middle: HierCNN; Right: HS_CO_P.

The major problem of the silver energy is that the sampling may introduce some wrong hypotheses that it is not able to prune them out due to similarity in the local appearance of fingers. These hypotheses are, however, eventually evaluated and filtered out with the golden energy in the final step.

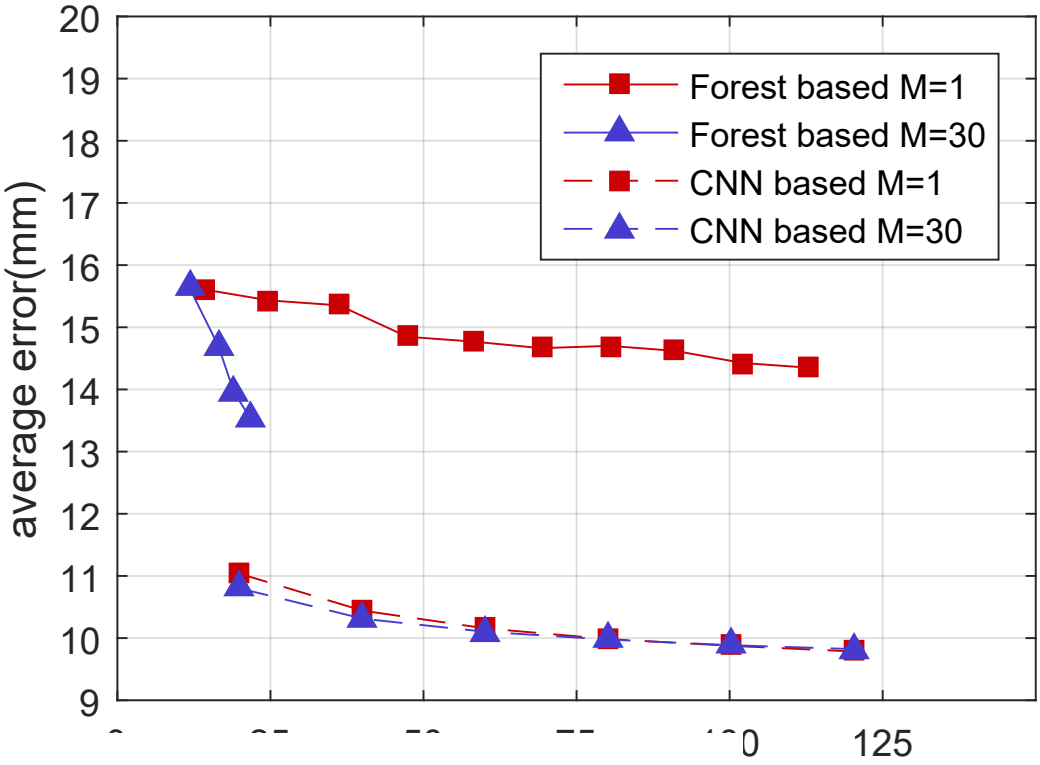
Fig. 5.5 qualitatively shows the comparison between the results of hierarchical regression and HSO. When there is no sampling and rejection before going into the next layers, the hierarchical estimation suffers from error accumulation. For example, the fingers of example (f) HS_CO_P rectifies the estimation of HierCNN for PIP joints on the index finger and HS_FO_P rectifies the estimation of HierForest of PIP joints on the ring and pinky fingers before moving onto the following layers. From the estimation of the thumb of example (b), we observe that the second term in the silver energy penalizes and forces the joint position to fall into the foreground area.

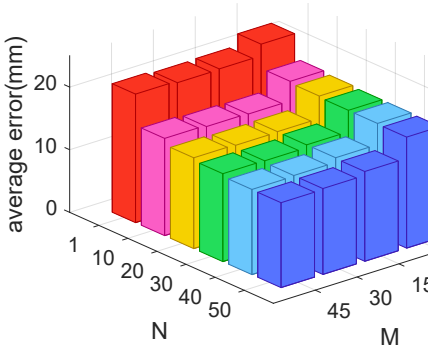
5.4.3 Optimization

In order to compare the performance of the naive selection and PSO, we choose the same number of evaluations for both methods: the maximum iteration $nIter$ and the number of the particals S of PSO to be $nIter \times S = M$, where M is the number of samples drawn for naive optimization. For the optimization in Fig. 5.3 and Fig. 5.4, $S = 30$, $nIter = 5$ and $M = 150$. $\mathrm{HS_FO_P}$ shows considerable improvement from $\mathrm{HS_FO_N}$ in most cases on the ICVL and NYU dataset. Whilst on the MSHD dataset, $\mathrm{HS_FO_P}$ and $\mathrm{HS_FO_N}$ have similar accuracy. The improvement is because the naive selection just chooses one best sample from the samples drawn from the distribution, whilst PSO explores new possible solutions by re-weighting all the samples according to their silver energy, which may result in a better configuration than the drawn samples. However, PSO relies on good initialization to converge. HierForest on the ICVL and NYU provides better initialization. Whereas on the MSHD dataset, the initial samples S are farther from ground truth, resulting in the mean error on MSHD is about 50mm and the error on ICVL and NYU is 22mm and 30mm, as shown in Fig. 5.4. According to (5.3)(5.4), it is possible that the perturbation of θ_{gbest} and θ_{pbest} from S samples in $nIter$ iterations are not enough to explore the large parameter space and thus cannot converge in only 5 iterations. However, if the same runtime efficiency is not considered and let $M = S$, PSO guarantees better, or at least the same accuracy as naive selection.

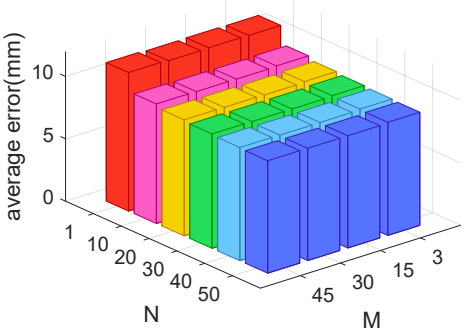
5.4.4 Accuracy, Efficiency and Parameter Analysis

The random forest based HSO, also the method proposed in [Tang et al., 2015], is used in this comparison,i.e. $\mathrm{HS_FO_N}$. In [Tang et al., 2015], to analyse the improve-





(a) Analysis on M and N for for sampling (figure from [Tang et al.



(b) Analysis on M and N for CNN based sampling

Figure 5.10: Analysing forest based and CNN based hierarchical sampling on the ICVL dataset.

ment in both accuracy and efficiency,two baselines are introduced: $M = 1$ (i.e. no per-joint optimization) and $M = 30$ and vary N to obtain a trade-off (Fig. 5.9). As solid lines in Fig. 5.9 shows, the average error drops more drastically when $M = 30$, without increasing too much of time budget.

A parameter analysis on M and N is performed in [Tang et al., 2015]. Sample are uniformly drawn from all trees. As in Fig. 5.10a, the error reduces with increasing M and N.

A similar trend of reduced error when increasing M and N of CNN based HSO, HS_CO_N, is observed in Fig. 5.10b while there is a difference in magnitude of the decrease. For the accuracy and efficiency analysis, for $M = 1$ and $N = 1$, the runtime of HS_CO_N is about 20ms while that of HS_FO_N is less than 0.2ms. Therefore, to obtain the same runtime with HS_FO_N, N varies from 1 to 6. In contrast to HS_FO_N, the errors both $M = 1$ and $M = 30$ of HS_CO_N drop at the same speed as shown by the dashed lines in Fig. 5.9.

The difference in both the magnitude and gradient of the error decrease between HS_FO_N and HS_CO_N is because CNN based HSO produces better distribution, which has been discussed in Section 5.4.2.

5.5 Summary

In this chapter, we have studied two different regressors(random forest and CNN) and optimizations methods (naive optimization and PSO) under the hierarchical probabilistic model to show this modification and compare them regarding memory consumption, runtime efficiency and accuracy. These comparisons provide some

guidance on the selection of regressors and optimization methods under certain requirements.

Generally, CNN based sampling performs better under the same runtime efficiency than random forest based sampling. Especially, when a large dataset covering full viewpoints and articulations is available, the memory requirement of random forest based sampling becomes prohibitive. On the other hand, the CNN regressor is unable to provide a distribution which can produce diverse hypotheses as random forest. One possible solution to overcome the limitation is using the mixture density model [Bishop, 1994] where a Gaussian mixture model is topped upon the CNN output layer

For the per-joint optimization, PSO is a better choice than naive optimization when good initializations are available. For the energy, although the silver energy is extremely efficient, it only looks at a partial pose at one time, without messages from other joint optimization, which can only acts as a filter to remove bad hypotheses but unable to converge to an exact fixed solution. Therefore, when the hypotheses produced by the regressors are already of good quality, like the CNN regressors used, passing all the samples without silver energy evaluation to the final golden energy evaluation is recommended.

Chapter 6

Occlusion Aware Pose Estimation

6.1 Motivation

In the last chapter, the hierarchical regression has proven to be effective in tackling the large hand pose space by decomposing it into smaller one. However, the output of the CNN is a single deterministic estimation of the joint location. To get multiple hypotheses, we jitter around the deterministic estimation with manually set variance. Though it provides a workaround to produce a distribution, the CNN fails to adequately handle self-occlusion problems, where occluded joints present multiple modes.

In this chapter, we tackle the self-occlusion issue and provide a complete description of observed poses given an input depth image by a novel method called hierarchical mixture density networks (HMDN). The proposed method leverages the state-of-the-art hand pose estimators based on Convolutional Neural Networks to facilitate feature learning, while it models the multiple modes in a two-level hierarchy to reconcile single-valued and multi-valued mapping in its output.

6.2 Overview

There are generally two typical camera settings for 3D hand pose estimation: a third-person viewpoint, where the camera is set in front of the user, and an egocentric (or first-person) viewpoint, where the camera is mounted on the user's head (in VR glasses, for example), or shoulder. While both settings share challenges like the full range of 3D global rotations, complex articulations, self-similar parts of hands, self-occlusions are more dominant in the egocentric viewpoints. Most existing hand benchmarks are collected in the third-person viewpoints, e.g. the two widely used ICVL [Tang et al., 2013] and NYU [Tompson et al., 2014] have less than 9% occluded finger joints.

Discriminative methods (cf. generative model fitting) in hand pose estimation learn a mapping from an input image to pose parameters from a large training dataset, and have been very successful in the settings of third-person viewpoints. However, they fail to handle occlusions frequently encountered in egocentric viewpoints. They treat the mapping to be single-valued, not being aware of that an input image may have multiple pose hypotheses when occlusions occur. See Fig. 6.1 where an example image and its multiple pose labels from the BigHand dataset [Yuan et al., 2017] are shown.

Given a set of hand images and their pose labels i.e. 3D joint locations, discriminative methods such as CNN minimize a mean squared error function, and the minimization of such error functions typically yields the averages of joint locations conditioned on input images. When all finger joints in the images are visible, the mapping is single-valued and the conditional average is correct, though the average only provides a limited description of the joint locations. However, for the occlusion

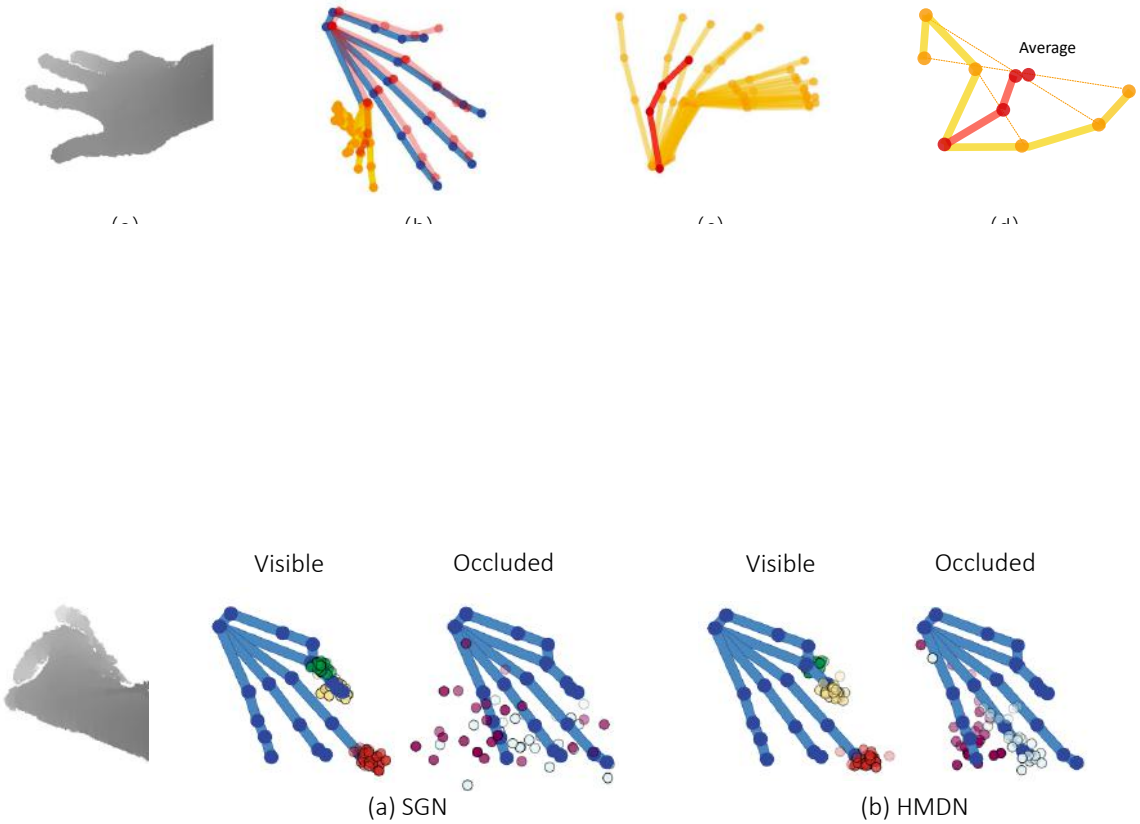


Figure 6.2: Samples drawn from the distributions of SGN and HMDN for finger tips.

cases, which happen frequently in the egocentric and hand-object interaction scenarios [Oikonomidis et al., 2012, 2011b; Poier et al., 2015; Garcia-Hernando et al., 2018], the mapping is multi-valued due to occluded joints that exhibit multiple locations given the same images. The conditional average of the joint locations is not necessarily a correct pose, as shown in Fig. 6.1b and Fig. 6.1c (The skeletons are shown in a 3D rotated view to better illustrate the problem, same for the other 3D skeletons shown in the paper). The prediction of a CNN trained by the mean squared error function is shown in red. It is interpretable and close to the ground truth for the visible joints, whereas it is physically implausible and not close to any of the given poses for the occluded joints. The example is clearer in Fig. 6.1d, where

we are given two available poses for the same image and CNN trained with the mean squared error function produces the pose estimation in red.

Existing discriminative methods, including the above CNN, are mostly deterministic, i.e. their outputs are single poses, thus lacking the description of all available joint locations. A discriminative method often serves as the initialization of a generative model fitting in the hybrid pose estimation approaches [Tang et al., 2015; Sharp et al., 2015]. If the discriminative method yields a probability distribution that well fits the data, than a single deterministic output, it would allow sampling pose hypotheses from its distribution. This, in turn, reduces the search space, helping a faster convergence, and avoids local minima from diverse candidates in the model fitting. Such sampling is crucial also for multi-stage pose estimation [Tang et al., 2015] and hand tracking [Oikonomidis et al., 2011a]. Previous methods ignore the pose space to be explored ahead and their optimization frameworks are not aware of occlusions.

In this Chapter, hierarchical mixture density networks (HMDN) are proposed to give a complete description of hand poses given images under occlusions. The probability distribution of joint locations is modeled in a two-level hierarchy to consider both single- and multi-valued mapping conditioned on the joint visibility. The first level represents the distribution of a latent variable for the joint visibility, while the second level the distribution of joint locations by a single Gaussian model for visible joints or a Gaussian mixture model for occluded joints. The hierarchical mixture density is topped upon the CNN output layer, and the whole network is trained end-to-end with the differentiable density functions. See Fig. 6.2. The distribution of the proposed method HMDN captures diverse joint locations in a compact manner, compared to the network that learns a single Gaussian distribution (SGN). To the

best of our knowledge, HMDN is the first solution that has its estimation in the form of a conditional probability distribution with the awareness of occlusions in 3D hand pose estimation. The experiments show that the proposed method significantly improves several baselines and state-of-the-art methods under occlusions given the same number of pose hypotheses.

6.3 Related Work

6.3.1 Pose estimation under occlusion

For free hand motions, methods explicitly tackling self-occlusions are rare as most existing datasets are collected in third-person viewpoints and the proportion of occluded joints is small. Franziska et al. [Mueller et al., 2017] observed that many existing methods fail to work under occlusions and even some commercial systems claiming for egocentric viewpoints often fail under severe occlusions. Methods developed for hand-object interactions [Oikonomidis et al., 2011b; Tzionas et al., 2016a; Sridhar et al., 2016], where occlusions happen frequently, model hands and objects together to resolve the occlusion issues. Jang et al. [Jang et al., 2015] and Rogez et al. [Rogez et al., 2014] exploit pose piors to refine the estimations. Franziska et al. [Mueller et al., 2017] and Rogez et al. [Rogez et al., 2015] generate synthetic images to train discriminative methods for difficult egocentric views.

In human body pose estimation and object keypoint detection, occlusions are tackled more explicitly [Haque et al., 2016; Charles et al., 2016; Rafi et al., 2015; Ghiasi et al., 2014; Sigal and Black, 2006; Chen and Yuille, 2014; Hsiao and Hebert, 2012; Navaratnam et al., 2007]. Chen et al. [Chen and Yuille, 2014] and Ghiasi et al.

[Ghiasi et al., 2014] learn templates for occluded parts. Hsiao et al. [Hsiao and Hebert, 2012] construct a occlusion model to score the plausibility of occluded regions. Rafi et al. [Rafi et al., 2015] and Wang et al. [Wang et al., 2013] utilize the information in backgrounds to help localize occluded keypoints. Charles et al. [Charles et al., 2016] evaluate automatic labeling according to occlusion reasoning. Haque et al. [Haque et al., 2016] jointly refine the prediction for visible parts and visibility mask in stages. Navaratnam et al. [Navaratnam et al., 2007] tackle the multi-valued mapping for 3d human body pose via marginal distributions which help estimate the joint density.

The existing methods do not address multi-modalities nor do not model the difference in distributions of visible and occluded joints. For CNN-based hand pose regression [Oberweger et al., 2015b; Oberweger and Lepetit, 2017; Tompson et al., 2014; Ye et al., 2016], the loss function used is the mean squared error, bringing in the aforementioned issues under occlusions. For random forest-based pose regression [Tang et al., 2013; Sun et al., 2015; Sharp et al., 2015], the estimation is made from the data in leaf nodes and it is convenient to fit a multi-modal model to the data. However, with no information of which joints are visible or occluded, the data in all leaf nodes is captured either by the mean-shift (a uni-modal distribution) or a Gaussian Mixture Model [Tang et al., 2015].

6.3.2 Mixture Models

Mixture density networks (MDN) were first proposed in [Bishop, 1994] to enable neural networks to overcome the limitation of the mean squared error function by producing a probability distribution. Zen et al. [Zen and Senior, 2014] use MDN

for acoustic modeling and Kinoshita et al. [Kinoshita et al., 2017] for speech feature enhancement. Variani [Variani et al., 2015] proposes to learn the features and the Gaussian Mixture Model model jointly. All these work apply MDN to model acoustic signals without an adaptation of the mixture density model. In addition to applying MDN to model the hand pose space when multiple modes exist due to occlusion, we extend MDN by a two-level hierarchy to fit the specific mixture of single-valued and multi-valued problems, for the application of hand pose estimation under occlusions. To model data under noise, a similar hierarchical mixture model is proposed in [Constantinopoulos et al., 2006] to represent "useful" data and "noise" by different sub-components, and a Bayesian approach is used to learn the parameters of the mixture model. Different from the work, we model a conditional distribution and use CNN to discriminatively learn the model parameters.

6.4 Hierarchical Mixture Density Network

6.4.1 Model Representation

The dataset to learn the model consists of $\{x_i, Y_i^j, v_i^j | i = 1, ..., I, j = 1, ..., J\}$, where x_i, Y_i^j, and v_i^j denote the i-th hand depth image, the multiple pose labels i.e. 3D locations of the j-th joint of the i-th image, and the visibility label of the j-th joint of the i-th image, respectively. The j-th joint is associated with multiple labels $Y_i^j = \{y_{iq}^j\}$, where $y_{iq}^j \in R^3$ is the m-th label i.e. 3D location. See Fig. 6.3 for examples. Each shows different example labels overlaid on the same depth image (in the first three columns), and all available labels in a 3D rotated view (in the last column).Visible joints are in blue and occluded joints in other colors. The visibility label is binary, indicating whether the j-th joint of the i-th image is visible or not. We treat J joints independently.

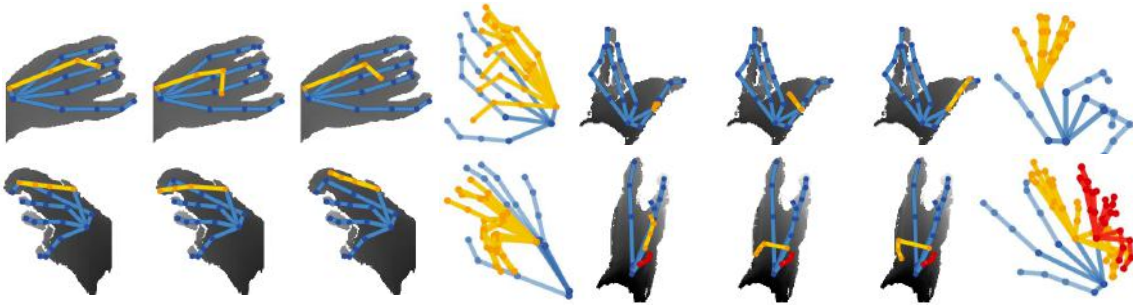


Figure 6.3: Hand images under self-occlusions exhibiting multiple pose labels. Each shows different example labels overlaid on the same depth image (in the first three columns), and all available labels in a 3D rotated view (in the last column).Visible joints are in blue and occluded joints in other colors.

To model hand poses under occlusions, a two-level hierarchy is considered. The top-level takes the visibility label, and the bottom-level switches between a uni-modal distribution and a multi-modal distribution, depending on the joint visibility.

The binary label or variable v_i^j follows the Bernoulli distribution,

$$p(v_i^j|w_i^j) = (w_i^j)^{v_i^j}(1 - w_i^j)^{(1-v_i^j)}, \tag{6.1}$$

where w_i^j is the probability that the joint is visible. As existing hand benchmarks do not provide the joint visibility labels, we use a sphere model similar to [Qian et al., 2014a] to generate the visibility labels from the available pose labels. Please refer Chapter 3 for the details of the labelling method. The visibility labels v_i^j are used for training, and they are inferred at testing.

When $v_i^j = 1$, the joint is visible in the image and the location is deterministic. Considering the label noise, y_{iq}^j is generated from a single Gaussian distribution,

$$p(y_{iq}^j|v_i^j = 1) = \mathcal{N}(y_{iq}^j; \mu_i^j, \sigma_i^j). \tag{6.2}$$

When the joint is occluded i.e. $v_i^j = 0$, it has multiple labels and they are drawn from a Gaussian Mixture Model (GMM) with J components

$$p(y_{iq}^j|v_i^j = 0) = \sum_{c=1}^{\mathcal{C}} \pi_{ic}^j \mathcal{N}(y_{iq}^j; \epsilon_{ic}^j, s_{ic}^j), \tag{6.3}$$

where ϵ_{ic}^j and s_{ic}^j represent the center and standard deviation of the j-th component. The location y_{iq}^j is drawn from the j-th component dependent on a hidden variable z_{ic}^j, where $z_{ic}^j \in \{0, 1\}$ and $\sum_{c=1}^{\mathcal{C}} z_{ic}^j = 1$. The hidden variable is under the distribution $p(z_{ic}^j) = \prod_{c=1}^{\mathcal{C}} (\pi_{ic}^j)^{(z_{ic}^j)}$, where $0 \leq \pi_{ic}^j \leq 1$, $\sum_{c=1}^{\mathcal{C}} \pi_{ic}^j = 1$. With all components defined,

the distribution of the joint location conditioned on the visibility is

$$p(y_{iq}^j|v_i^j) = \left[\mathcal{N}(y_{iq}^j; \mu_i^j, \sigma_i^j)\right]^{v_i^j} \left[\sum_{c=1}^{\mathcal{C}} \pi_{ic}^j \mathcal{N}(y_{iq}^j; \epsilon_{ic}^j, s_{ic}^j)\right]^{(1-v_i^j)} \tag{6.4}$$

and the joint distribution of y_{iq}^j and v_i^j is

$$p(y_{iq}^j, v_i^j) = \left[w_i^j \mathcal{N}(y_{iq}^j; \mu_i^j, \sigma_i^j)\right]^{v_i^j} \left[(1 - w_i^j) \sum_{c=1}^{\mathcal{C}} \pi_{ic}^j \mathcal{N}(y_{iq}^j; \epsilon_{ic}^j, s_{ic}^j)\right]^{(1-v_i^j)}. \tag{6.5}$$

Eqn. (6.4) shows that the generation of joint locations y_{iq}^j given the input image x_i is in a two-level hierarchy: first, a sample v_i^j is drawn from Eqn. (6.1) and then, depending on v_i^j, a joint location is drawn either from a uni-modal Gaussian distribution or GMM. Thus, the proposed model switches between the two cases and provides a full description of hand poses under occlusions. The joint distribution in Eqn. (6.5) is used to define the loss function in Section 6.4.3.

6.4.2 Architecture

The formulations in the previous section are presented for the j-th joint y_{iq}^j. For all J joints of hands, the distribution is obtained by multiplying the distributions of independent joints. The observed hand poses and the joint visibility, given x_i, are drawn from $\prod_{j=1}^{J} \prod_{m} p(y_{iq}^j, v_i^j)$.

Note that the hierarchical mixture density in Eqn. (6.4) and the joint distribution in Eqn. (6.5) are conditioned on x_i. All model parameters are in a functional form of x_i and the joint distribution in Eqn. (6.5) is differentiable. We choose to learn these functions by a CNN and the distribution is parameterized by the output of the CNN. As shown in Fig. 6.4, the input of the CNN is an image x_i and the outputs are the HMDN parameters: $w_i^j, \mu_i^j, \sigma_i^j, \epsilon_{ic}^j, s_{ic}^j, \pi_{ic}^j$, for $j = 1, ..., J$ and $j = 1, ..., J$. The output parameters consist of three parts. w_i^j is the visibility probability in Eqn. (6.1), μ_i^j, σ_i^j for the uni-modal Gaussian in Eqn. (6.2), and $\epsilon_{ic}^j, s_{ic}^j, \pi_{ic}^j$ for the GMM in Eqn. (6.3). Different activation functions are used to meet the defined ranges of parameters. For instance, the standard deviations σ_i^j and s_{ic}^j are activated by an exponential function to remain positive and π_{ic}^j by a softmax function to be in $[0, 1]$.

The prediction of the visibility, the value of w_i^j, is used to compute the visibility loss over the visibility label v_i^j. See Section 6.4.3. Depending on the visibility label v_i^j, the parameters of the uni-modal Gaussian (for visible joints) or GMM (for occluded joints) are chosen to compute the loss, as shown in blue and in orange respectively in Fig. 6.4.

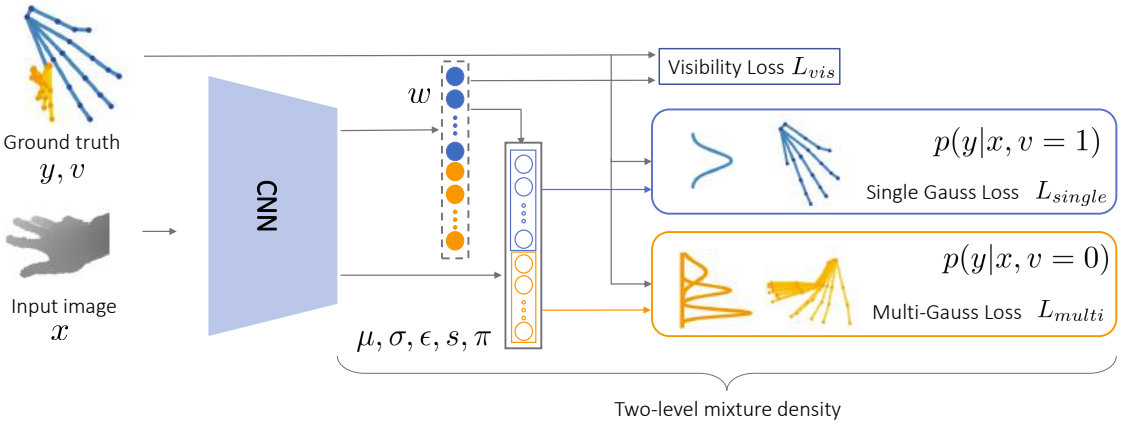


Figure 6.4: Hierarchical Mixture Density Network. Hand joint locations y given the input image x are modeled in a two-level hierarchy: in the first level, the visibility is modeled by Bernoulli distribution whose parameter is w; then depending on the visibility, the joint locations are either modeled by uni-modal Gaussian distributions (visible joints, shown in blue) or GMMs (occluded joint, shown in orange). The CNN outputs the parameters of HMDN, i.e. w, μ, σ, ϵ, s, π.

6.4.3 Training and Testing

The likelihood for the entire dataset $\{x_i, Y_i^j, v_i^j | i = 1, ..., I, j = 1, ..., J\}$ is computed as $P = \prod_{i=1}^{I} \prod_{j=1}^{J} \prod_{q} p(y_{iq}^j, v_i^j)$, where $p(y_{iq}^j, v_i^j)$ in (6.5) has the model parameters dependent on x_i. Thus, our goal is to learn the neural networks that yield the parameters that maximize the likelihood on the dataset. We use the negative logarithmic likelihood as the loss function.

$$L = -logP = \sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{m} \{L_{vis} + L_{single} + L_{multi}\}, \tag{6.6}$$

where

$$L_{vis} = -v_i^j log(w_i^j) - (1 - v_i^j) log(1 - w_i^j), \tag{6.7}$$

$$L_{single} = -v_i^j log(\mathcal{N}(y_{iq}^j; \mu_i^j, \sigma_i^j)), \tag{6.8}$$

$$L_{multi} = -(1 - v_i^j) log(\sum_{c=1}^{\mathcal{C}} \pi_{ic}^j \mathcal{N}(y_{iq}^j; \epsilon_{ic}^j, s_{ic}^j)). \tag{6.9}$$

The three loss functions correspond to the three branches in Fig. 6.4. The visibility loss L_{vis} is computed using the predicated value of w_i^j. When $v_i^j = 1$, $L_{multi} = 0$ and L_{single} is calculated, and when $v_i^j = 0$, vise versa.

During testing, when an image x_i is fed into the network, the prediction for the j-th joint location is diverted to different branches according to the prediction of the visibility probability w_i^j. If w_i^j is larger than 0.5, the prediction (or sampling) for the location is made by the uni-modal Gaussian distribution in Eqn. (6.2); otherwise, the GMM in Eqn. (6.3).

However, when the prediction for the visibility is erroneous, the prediction for the joint location will be wrong. To help the bias problem, instead of using the binary visibility labels v_i^j to compute the likelihood, we use the samples drawn from the estimated distribution in Eqn. (6.1) during training. When the number of samples is large enough, the mean of these samples becomes w_i^j. So, the losses in Eqn. (6.8) and (6.9) change to

$$L_{single} = -w_i^j log(\mathcal{N}(y_{iq}^j; \mu_i^j, \sigma_i^j)), \tag{6.10}$$

$$L_{multi} = -(1 - w_i^j) log(\sum_{c=1}^{\mathcal{C}} \pi_{ic}^j \mathcal{N}(y_{iq}^j; \epsilon_{ic}^j, s_{ic}^j)). \tag{6.11}$$

The modified losses in Eqn. (6.10) and (6.11) can be seen as a soft version of the original ones Eqn. (6.8) and (6.9).

6.4.4 Degradation into Mixture Density Network

HMDN degrades into Mixture Density Network (MDN), without the supervision for learning the visibility variable. The other form of (6.4) is

$$p(y_{iq}^j|w_i^j) = w_i^j \mathcal{N}(y_{iq}^j; \mu_i^j, \sigma_i^j) + (1 - w_i^j) \left[\sum_{c=1}^{\mathcal{C}} \pi_{ic}^j \mathcal{N}(y_{iq}^j; \epsilon_{ic}^j, s_{ic}^j) \right] \tag{6.12}$$

where the visibility probability w_i^j is learned with visibility labels. When the labels are not available, the above equation becomes

$$p(y_{iq}^j) = \sum_{c=1}^{\mathcal{C}+1} \bar{\pi}_{ic}^j \mathcal{N}(y_{iq}^j; \bar{\epsilon}_{ic}^j, \bar{s}_{ic}^j) \tag{6.13}$$

where $\bar{\pi}_{nJ+1}^j = w_i^j, \bar{\epsilon}_{nJ+1}^j = \mu_i^j, \bar{s}_{nJ+1}^j = \sigma_i^j$, and $\bar{\pi}_{ic}^j = (1 - w_i^j)\pi_{ic}^j, \bar{\epsilon}_{ic}^j = \epsilon_{ic}^j, \bar{s}_{ic}^j = s_{ic}^j$ for $j = 1, ..., J$. The visibility probability w_i^j in (6.12) is absorbed into the GMM mixing coefficients $\bar{\pi}_{ic}^j$, and the distribution becomes a GMM with $J+1$ components with no dependency on the visibility.

6.5 CNN architecture

The network used for SGN, MDN and HMDN is adapted from Unet [Ronneberger et al., 2015] by changing the last few convolutional layers, shown in Fig. 6.5. The symbols of the convolutional layers are defined in [Ronneberger et al., 2015]. Each blue box is a multi-channel feature map and the white one is the copied feature map. The number of channels for the models in the paper is shown on top of the box. Different operations are denoted by arrows. Three fully connected layers are used and the dimension of two hidden layers is 1152.

The size of the input image for the CNN is 640 × 480 and the output dimension is the number of the parameters of the density function used to model the target pose. We have 21 joint locations to be estimated and each joint location is 3 dimensional. In order to reduce the output dimension, we make all the dimensions of each joint location share one covariance. The CNN output dimension for SGN is $21 \times (3+1)$, for MDN is $21 \times \mathcal{C} \times (3+2)$ and for HMDN is $21 \times \mathcal{C} \times (3+2) + 21$.

To reduce the number of the parameters, all the joints can further share some parameters, for examples, the mix-coefficients of the Gaussian Mixtures.

For the training, note that the representation of the dataset $\{x_i, Y_i^j, v_i^j | i = 1, ..., I, j = 1, ..., J\}$ is to facilitate the explanation of HMDN. When computing the loss function, we do not need to find the associated multiple labels Y_i^j for a given image x_i. We can feed current frame and its pose label, compute the likelihood for all the joint of this frame, and multiply the likelihood of all the frames.

All the networks are trained using Adam [Kingma and Ba, 2014] and the convergence times of all methods above took about 24 hours using Geforce GTX 1080Ti.

6.6 Experiments

6.6.1 Self-comparisons

The baseline of our comparison is Single Gaussian Network (SGN), which is the CNN trained with a uni-modal Gaussian distribution. In [Bishop, 2006], it is shown that maximization of the likelihood function under a uni-modal Gaussian distribution

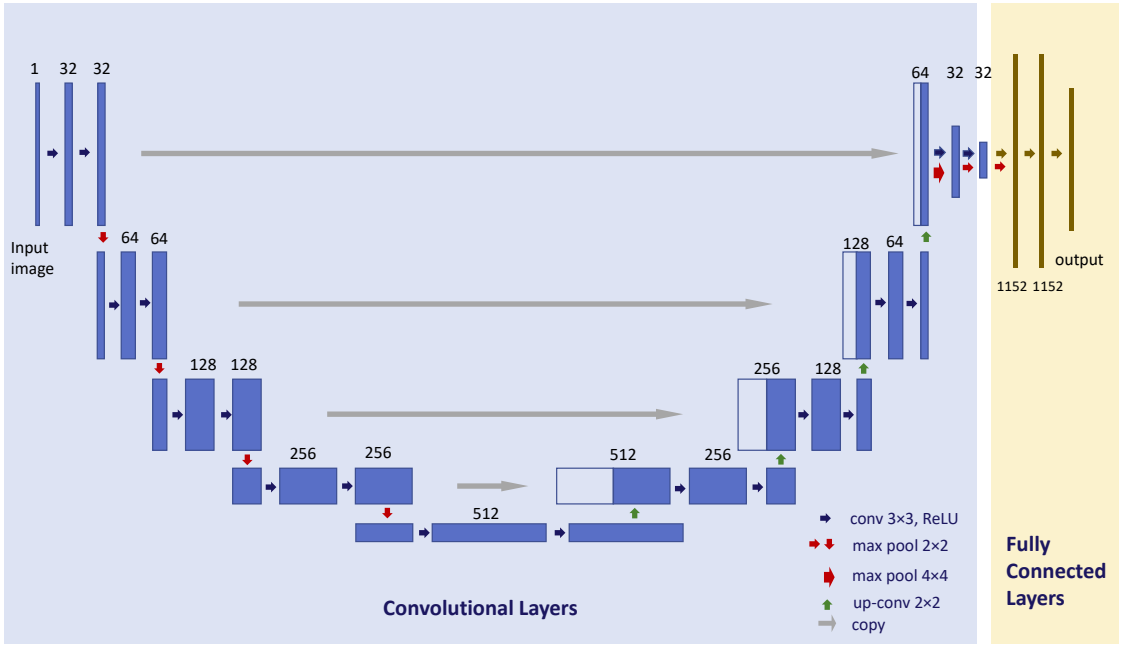


Figure 6.5: The CNN architecture Used for SGN, MDN and HMDN.

for a linear model is equivalent to minimizing the mean squared error errors. In our experiments, we observed that the estimation error of SGN using the Gaussian center is about the same as that of the CNN trained with the mean squared error. For further comparisons under the probabilistic framework, we report the accuracies of SGN.

We also report the experiments of MDN as in the previous section, we showed that HMDN degrades to MDN when there is no visibility label available in training. To compare MDN with HMDN fairly, the number of Gaussian components of MDN is set $\mathcal{C}+1$ and that of GMM branch of HDMN is $\mathcal{C}$.

Qualitative Analyses. See Fig. 6.6. 100 samples for each finger tip are drawn from the distributions of the different methods. HMDN is motivated by the intrinsic mapping difference: single-valued mapping for visible and multi-valued mapping for

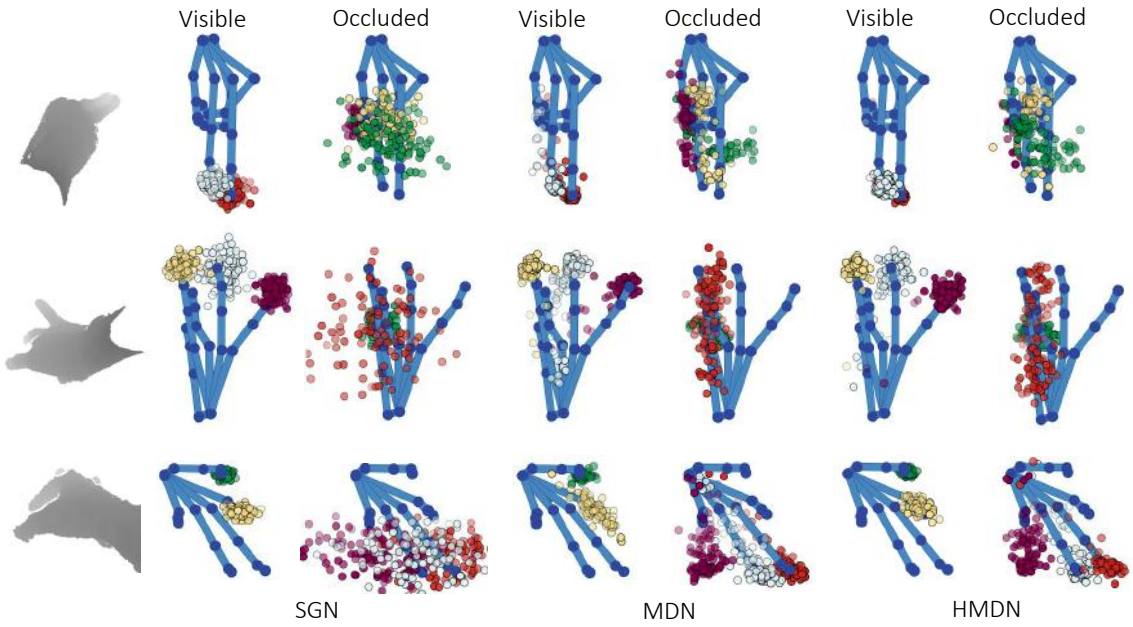


Figure 6.6: Samples drawn from the distributions of SGN, MDN and HMDN for finger tips, shown in comparison to a pose label.

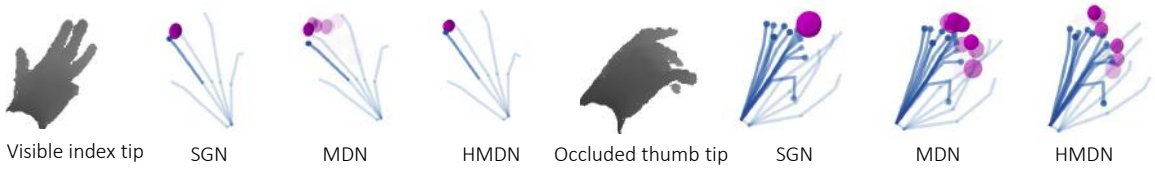


Figure 6.7: Distributions predicted by SGN, MDN and HMDN for visible index tip and occluded thumb tip. Each magenta sphere represents a Gaussian component whose radius is the standard deviation and center is the mean. The degree of transparency is in proportion to the mixture coefficients $\{\pi\}$.

occluded joints. Our results, shown in Fig. 6.6, demonstrate its ability of modeling this difference by producing interpretable and diverse candidate samples accordingly. For visible joints, SGN and HMDN produce the samples distributed in a compact region around the ground truth location, while the samples from MDN scatter in a larger area. For occluded joints, while the samples produced by SGN scatter in a broad sphere range, the samples produced by HMDN and MDN form an arc-shaped region, which indicates the movement range of finger tips within the kinematic

constraints.

With the aid of visibility supervision, HMDN handles well the self-occlusion problem by tailoring different density functions to the respective cases. The examples of the distributions predicted by SGN, MDN and HMDN for visible and occluded joints are shown in Fig. 6.7. The resulting compact distributions that fit both visible joints and occluded joints improve the pose prediction accuracies in the following quantitative analyses.

Such compact and interpretable distributions are also helpful for hybrid methods [Tang et al., 2015; Sharp et al., 2015]. For the discriminative-generative pipelines, the distribution largely reduces the space to be explored and produces diverse candidates to avoid being stuck at local minima in the generative part. For example, the conditional distribution for the occluded joints which confines the occlusion space can be added as a term in energy function defined in [Taylor et al., 2014, 2016] as it is differentiable while the deterministic output can only provide a starting point for optimisation of the energy.

For hand tracking methods [Oikonomidis et al., 2011a], the distributions of occluded joints can be combined with the motion information e.g. speed and direction, to give a sharper i.e. more confident response at a certain location. The model can also find its application in multi-view settings. This is because the distribution is able to give a compact representation for the movement range of the occluded joints. Even with temporal priors or other viewpoint estimation, resolving the uncertainty by a deterministic estimation which lacks of the information of other possible locations or a unimodal gaussian distribution which gives a broad sphere region is not straightforward. For example, if a joint in two images captured in different viewpoints is occluded, the deterministic CNN may produce two different 3D locations and

the unimodel gaussian distribution adds two variances centred on the deterministic results. These outputs are insufficient to locate the exact locations. On other other hand, the distribution of the HMDN models the movement range with physical constraint of the occcuded joint and with this extra movement information, the uncertainty can be resolved.

Modelling the uncertainty is useful in real world problems as it is common that at certain occasions the information is limited and we need to maintain these uncertainty and propagate the current information to future. In the hand shape estimation, Tkach et al. [Tkach et al., 2017] point out the uncertainty in estimating the shape parameter, for example, estimating the finger bone length from the images whose fingers are not bent and then leverage the uncertainty to integrate per-frame estimates over time. HMDN gives a good representation of the uncertainty of the occluded joints and the underlying Gaussian Mixture Model of the representation is to maintain and propagate.

(a) Occluded pinky finger tips (b) Occluded index finger tips

Figure 6.8: The space for occlusion finger tip is dependent on other visible joints.

Dependency Though we do not model the explicit dependency between occluded and visible joints in this work, the dependency is implicitly captured from the data. Our occluded joint distributions are conditioned on input images, and the input images are formed as a function of visible joints. See Fig. 6.8. Albeit the same occluded finger tips, the mixture coefficients and the span of GMM components differ depend-

ing on the input images. As we treat the joint independently, during the sampling, the hypotheses may violate the kinematic constraints. The use of correlation priors [Taylor et al., 2016] will help ensure the constraints during sampling.

Table 6.1: Estimation errors of different models. *see text for the evaluation metric

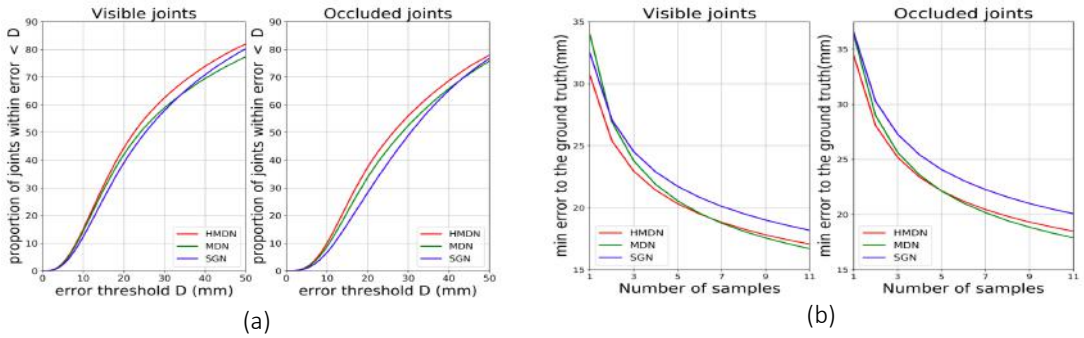


Figure 6.9: Comparison of HMDN, SGN, MDN, when $\mathcal{C} = 20$.

Quantitative Analyses. To conform with the euclidean error metric, i.e. the ground truth joint location to measure the displacement error (in mm), adopted in the prior work, we sample hypotheses from the distributions and compare them with the ground truth. By sampling multiple hypotheses, the error is in fact the average of the error of one hypothesis to the ground truth weighted by its probability. This weighted average measures the quality of the predicted distribution indirectly.

The average errors are reported for visible joints and occluded joints separately in Table 6.1. Fig. 6.9a presents the comparisons under the commonly used metric, the proportion of joints within a error threshold [Tang et al., 2015; Ye et al., 2016;

Sharp et al., 2015], using $\mathcal{C} = 20$ in MDN/HMDN. HMDN outperforms both MDN and SGN for visible and occluded joints using the different numbers of Gaussian components. For occluded joints, HMDN improves SGN by 10% in the percentage of joints within the error 20mm (Fig. 6.9a), and by about 2mm in the mean displacement error (Table 6.1). HMDN also outperforms the baselines for visible joints. One can reason that given the limited network capacity, by specifying density functions by data types, HMDN learns to take a better balance between the visible and occluded, while maximizing the likelihood of the entire training data. As shown in Table 6.1, the estimation errors of HMDN do not change much for $\mathcal{C} = 10, 20, 30$. Note, however, the number of model parameters linearly increases with $\mathcal{C}$.

In Fig. 6.9b, we vary the number of samples drawn from the distributions, and measure the minimum distance error. HMDN consistently achieves lower errors than SGN at all numbers of samples. Compared to MDN, HMDN appears better at the smaller numbers of samples. When the number of samples increases, the error gap between the two methods becomes small.

In both Table 6.1 and Fig. 6.9, we repeated the sampling process 100 times and reported the mean accuracies. The standard deviations were fairly small as: 0.03-0.04 mm for occluded joints, and 0.01-0.02 mm for visible joints.

As our motivation is in modeling the distribution of joint locations, we measure how well the predicted distribution aligns with the target distribution. As shown in Fig. 6.3, multiple pose labels are gathered for the same image with occlusions. We draw multiple samples from the predicted distribution and measure the minimum distance between the set of drawn samples and the set of pose labels. As shown in the last row of Table 6.1, the improvement is significant. Both MDN and HMDN outperform SGN by about 4 mm, which demonstrates that the arc-shaped distribu-

tions produced by MDN/HMDN align better with the target joint locations than the sphere-shaped distribution produced by SGN, as shown in Fig. 6.6. Instead of the minimum distance, we could use other similarity measures between distributions.

Though the improvement of HMDN over MDN is marginal, the number of parameters for the density is largely reduced by the awareness of occlusion in HMDN when joints are visible.

In the first metric, multiple samples are compared to each ground truth given a image and the errors for all the ground truth of the image are averaged. In this setting, the number of ground truth for each frame has little influence on the reported error, especially for video sequences where neighbouring frames are similar. For the second metric, as we compare the set of ground truth and the set of samples, if the number of the ground truth for the testing image is limited and the ground truth does not cover the space for the occluded joints, the superiority of HMDN cannot be displayed. For the extreme case where only one ground truth is available for a image with self-occlusions, the distance between two sets becomes random. Since the EgoBighand dataset is the biggest hand dataset in the ego-centric viewpoint and the configurations for the occluded joints are augmented using the articulation from the third-person viewpoint subset of the Bighand dataset, it provides substantial number of ground truth configurations for the occluded joints. During the augmentation, for a occluded finger, there are at least 10 configurations for the joint locations.

Bias. In Section 6.4.3, we proposed to mitigate the exposed bias during testing, by sampling from the visibility distribution at training. HMDN trained with the loss functions in Eqn. (6.8) and (6.9), is denoted as $\text{HMDN}_{\text{hard}}$, while the one trained with Eqn. (6.10) and (6.11) is $\text{HMDN}_{\text{soft}}$. In Table 6.2 $\text{HMDN}_{\text{soft}}$ consistently achieves lower errors than $\text{HMDN}_{\text{hard}}$ for different numbers of Gaussian components.

Table 6.2: Comparison of $\text{HMDN}_{\text{hard}}$ and $\text{HMDN}_{\text{soft}}$

No. of Gauss.($\mathcal{C}$)	10		20		30	
Model	hard	soft	hard	soft	hard	soft
Vis. Err.(mm)	32.2	30.5	32.9	30.7	33.1	30.5
Occ. Err.(mm)	35.8	34.8	35.9	34.4	36.4	34.2

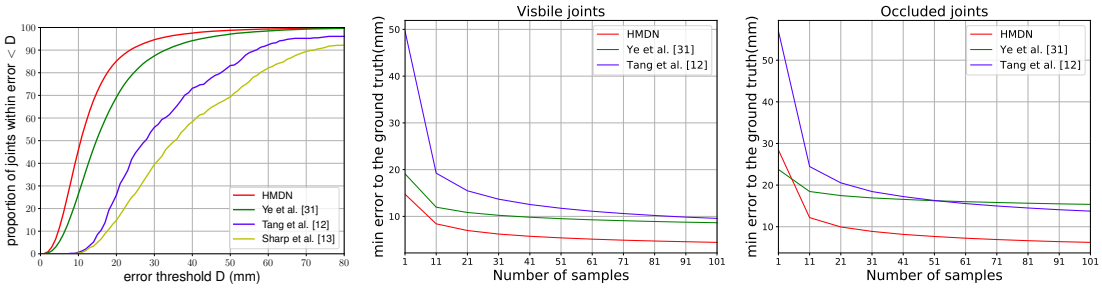
6.6.2 Comparison with the state-of-the-arts

To compete with state-of-the-arts, the following strategies are adopted: first, a CNN network is trained to estimate the global rotation and translation, and conditioned on the estimation, HMDN is then trained; data augmentation, including translation, in-plane rotation, and scaling is used.

MSHD Dataset. MSHD has a considerable number of occluded joints both in training and testing set. We compare HMDN with three methods: Ye et al.[Ye et al., 2016], Tang et al.[Tang et al., 2015], Sharp et al.[Sharp et al., 2015]. For [Sharp et al., 2015], the results of its discriminative part are used. Fig. 6.10a shows the proportion of joints within different error thresholds for the four methods, where a single prediction is used from HMDN.

In Fig. 6.10b and Fig. 6.10c, we further compare Ye et al. [Ye et al., 2016] and Tang et al. [Tang et al., 2015] with HMDN, by varying the number of hypotheses i.e. samples from the output distributions, and measuring the minimum displacement errors. Ye et al. [Ye et al., 2016] use a deterministic CNN. To produce multiple samples, they jitter around the CNN prediction, which can be treated as a uni-modal Gaussian. Tang et al. [Tang et al., 2015] use decision forests (3 trees) and the data points in the leaf nodes are modeled by GMM with 3 components. During

testing, samples are drawn from GMMs of all trees. We used the original codes from the authors in our experiments.



(a) (b) (c)

Figure 6.10: Comparison of HMDN with prior work.

HMDN significantly outperforms both methods for visible joints. For occluded joints, when the number of samples is 1, the errors of HMDN and Ye et al. [Ye et al., 2016] are close. However, Ye et al. [Ye et al., 2016] are not able to produce diverse samples to reach low errors as HMDN when the number of samples increases. Tang et al. [Tang et al., 2015] provide diverse candidates by GMM in its leaf nodes, but the variance of the distribution is much larger than that of Ye et al. [Ye et al., 2016] and HMDN for both visible and occluded joints. From the results, HMDN demonstrates its superiority for both the unimodal Gaussian model and GMM: the compact distribution with lower bias for visible joints and the diverse samples yet having smaller variances for occluded joints. See Fig. 6.11 for example results. The samples from Tang et al. [Tang et al., 2015] for the finger tips spans a large region; those from Ye et al. [Ye et al., 2016] are more compact but many deviate from the ground truth.

NYU Dataset. The proposed method has also been evaluated on NYU dataset. Most joints in the training set are visible while on the testing set, there are up

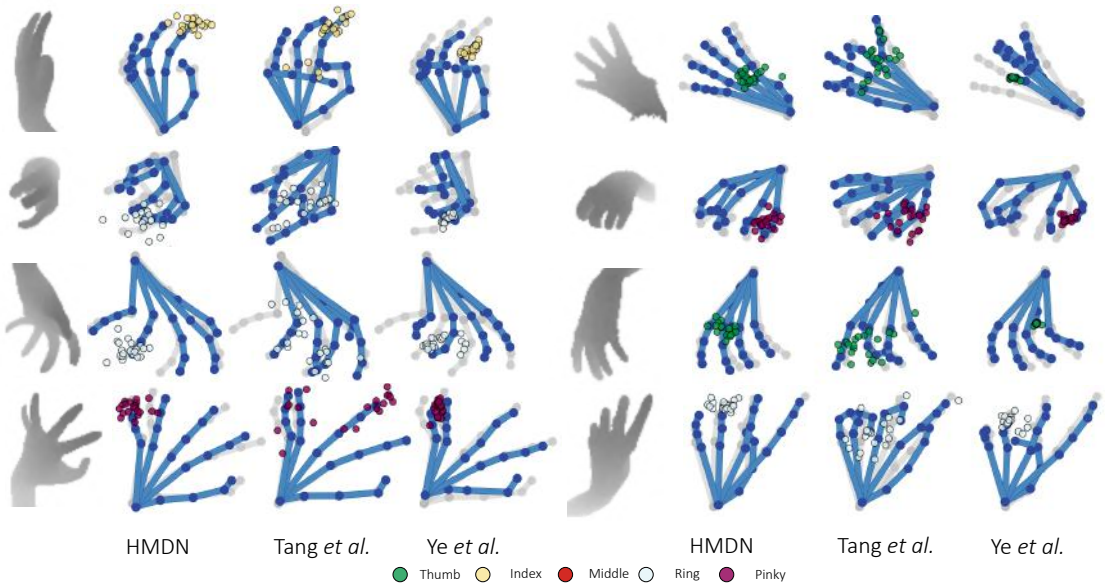


Figure 6.11: Comparison of HMDN with Tang et al. [Tang et al., 2015] and Ye et al. [Ye et al., 2016]. Ground truth: skeletons in gray. Predictions from the models: skeletons in blue. For each image, samples for one tip joint from the three methods are scattered along the skeletons. Visible joints in the left column and occluded joints in the right column.

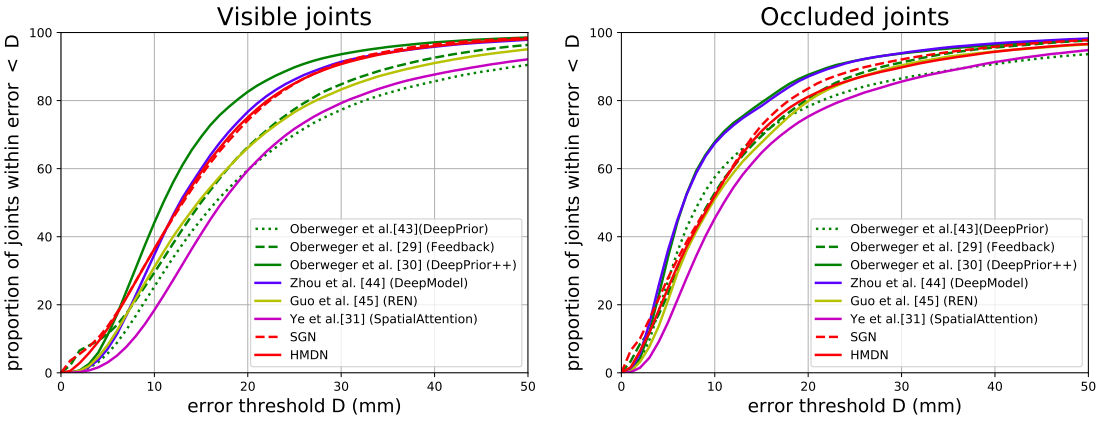


Figure 6.12: Comparison with state-of-the-art approaches on NYU dataset.

to 36% occluded joints. This implies all the joints in the testing dataset will be predicted as visible joints. Despite the ill-setting for HMDN, the method does not fail but degrades into SGN: the performances of SGN and HMDN are similar as shown in Fig. 6.12, and when compared with various state-of-the-arts based on

CNN [Oberweger et al., 2015a,b; Oberweger and Lepetit, 2017; Zhou et al., 2016b; Guo et al., 2017; Ye et al., 2016], HMDN is in the second place for visible joints and third place for occluded joints. Note the best method [Oberweger and Lepetit, 2017] uses a 50-layer ResNet model [He et al., 2016] and 21 more CNN models to refine the estimation.

6.7 Summary

This paper addresses the occlusion issues in 3D hand pose estimation. Existing discriminative methods are not aware of the multiple modes of occluded joints and thus do not adequately handle the self-occlusions frequently encountered in egocentric views. The proposed HMDN models the hand pose in a two-level hierarchy to explain visible joints and occluded joints by their uni-modal and multi-modal traits respectively. The experimental results show that HMDN successfully captures the distributions of visible and occluded joints, and significantly outperforms prior work in terms of hand pose estimation accuracy. HMDN also produces interpretable and diverse candidate samples, which is useful for hybrid pose estimation, tracking, or multi-stage pose estimation, which require sampling.

In the Chapter, we assume the outputs are independent and do not exploit the temporal continuity. To sample kinematically valid poses, we can consider modeling hand structural information. One approach is to explicitly learn the dependency on top of part regression, e.g. by deep structured models [Yang et al., 2016]. Though the pose estimation benefits from the structural models, they usually result in highly interconnected models and thus difficult learning, and exact inference becomes intractable. The other approach exploits the dependency priors to post-process the

part regression: e.g. a multivariate normal with correlation priors is used to constrain the pose samples [Taylor et al., 2016; Tan et al., 2016; Tome et al., 2017]. We can also further incorporate the temporal dependency using offline priors or learning it with the part regression in the LSTM framework.

Testifying HMDN on hand-object and hand-hand interaction scenarios is interesting. Though it was tested on the datasets with self-occlusions, the generalization to different occlusion types is promising.

Chapter 7

Conclusion and Future Work

7.1 Conclusion

3D hand pose is challenging due to complicated variations caused by high Degree of Freedom(articulations and multiple viewpoints), self-similar parts, severe self-occlusions, different shapes, and sizes. The thesis first tackles the challenge of the multiple viewpoints and complex articulations of hand pose estimation by decomposing and transforming the input and output space by spatial transformations following the hand kinematic hierarchy.

The hierarchical estimation of the hand pose from images is effective in most cases when compared with the method treating the whole hand estimation as a black box as the reduced input and output space makes the learning of CNN easier. However, for cases when the estimation in a certain layer is wrong, the error will propagate to all the following layers. The iterative refinement reduces the error for each joint, resulting in a more precise mapping function, and the post-process component which samples from estimation from the regressors for the joints and evaluates these sam-

ples is able to remove estimation far from ground truth for each frame.

Under the hierarchy, multiple variants are integrated into a probabilistic framework and studies regarding the efficacy and efficiency to give an sight of each component.

When comparing different variants of the hierarchical framework, generally, when a large data is available, CNN based sampling performs better under the same runtime efficiency than decision forest based sampling.

On the other hand, as the prediction of decision forest is made in the leaf nodes, different models, such as one deterministic estimation, a uni-Gauss model, or a Gaussian mixture model, can be fitted to the data in the leaf nodes without retraining the trees while CNN based regressor requires re-training when the model for the output space is changed.

For the per-joint optimization, PSO is a better choice than naive optimization when good initializations are available.

For the energy, although the silver energy is extremely efficient, it only looks at a partial pose at one time, without messages from other joint optimization, which can only acts as a filter to remove bad hypotheses but unable to converge to an exact fixed solution.

The thesis further tackles the challenge of the self-occlusions which happens frequently in ego-centric viewpoint: modelling the multi-modality of the locations for occluded hand joints by a hierarchical mixture density deep network which leverages the state-of-the-art hand pose estimators based on Convolutional Neural Networks to facilitate feature learning while models the multiple modes in a two-level hierarchy to reconcile single-valued (for visible joints) and multi-valued (for occluded

joints) mapping in its output.

CNN trained with the mean squared error gives a deterministic output and thus fails to learn a one-to-many mapping. Instead of predicting the joint locations, HMDN predict the parameters of distributions representing the locations and successfully captures the distributions of visible and occluded joints. HMDN is able produces interpretable and diverse candidate samples, which is useful for hybrid pose estimation, tracking, or multi-stage pose estimation, which require sampling. Upon the submission of the thesis, some work learning the one-to-many image translation or many-to-many image translation with Bidirectional cyleGAN [Zhu et al., 2017; Huang et al., 2018] is published, which is also able to model the conditional distribution in the output space (and in the input space). However, these samples are generate from normal distribution, which is not straightforward to be incorporated as an energy term for model fitting methods.

Also, a real dataset aiming at covering all viewpoints, articulation, sever self-occlusions and different hand shapes is collected by a tracking system with six 6D magnetic sensors. This evenly covered dataset largely improve the performance of the hand estimators.

7.2 Future Work

Though the methods proposed in the previous chapters have achieved state-of-the-art performance, improvements can be made in these following aspects.

7.2.1 Hierarchy

For the hierarchical regression, the error accumulation is due to the local regressors: the error for the full pose is not considered when training for the partial pose regressors. One way to mitigate the error accumulation is what proposed in the thesis. At test time, the hierarchical network samples and evaluates the hypotheses to explore multiple paths, resulting different full poses, and choose the best one by a energy function. While it reduces the error accumulation and improves the performance, it makes the estimation for the full pose $M \times N$ slower, where M is the number of the samples for the partial poses and N is the number of the full pose hypotheses.

An alternative way of improving the performance and efficiency is training networks maximising an error over the full pose. Take CNN for example. If the CNN regressors are deterministic, the spatial transformation can be parameterized by the output of the regressors, i.e. the rotations and translations of the transformation matrices directly acquired from the CNN output. Then all the regressors within the hierarchy can be trained end-to-end with an extra full pose error. If the CNN regressors output distributions as MDN and HMDN, sampling during training is required and the reward for a sample is determined by the final full pose error, which is similar to some recurrent models in [Ranzato et al., 2016; Mnih et al., 2014].

7.2.2 HMDN

For HMDN, applications in scenarios like multiple-viewpoints, tracking and hand-object interactions are be explored and verified. With the possible pose space for occluded joints represented accurately by the distribution, additional information from different sources can be combined to locate the actual position easily. For

example, in multiple-viewpoints, combining the information from all the viewpoints, which can be done by computing the products of the densities for the distributions estimated from images in different viewpoints, can produce a sharp response near the actual location of the occluded joints, even in all the viewpoints, the joints are occluded while deterministic networks find it hard to combine information from different viewpoints.

In scenarios like tracking and hand-object interactions, where model-based methods are usually exploited to search the pose space staring from the results of discriminative methods (and the results from the previous frames), HMDN can provide confidence for the joint prediction and confine the pose space to be searched with the variances. We want to see the performance of model-based methods with/without the variances when using joint predictions from discriminative methods.

7.2.3 Hand Detection

At the beginning, we assume that hand is free in the air and is cropped using ground truth or the results from some general hand detector while in fact the hand detection is non-trivial due to that people wear different kind gloves, touch surfaces, hold various objects etc.. Training a hand detector working on the these cases or able to reject these hard cases before the bad detection results going to the hand estimator is a potential topic.

Bibliography

Hands in the million challenge. `http://icvl.ee.ic.ac.uk/hands17/challenge/`.

I. Akhter and M. J. Black. Pose-conditioned joint angle limits for 3d human pose reconstruction. In *Computer Vision and Pattern Recognition*, 2015.

I. Akhter, Y. Sheikh, S. Khan, and T. Kanade. Trajectory space: A dual representation for nonrigid structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2011.

M. Andriluka, S. Roth, and B. Schiele. Pictorial structures revisited: People detection and articulated pose estimation. In *Computer Vision and Pattern Recognition*, 2009.

D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis. Scape: Shape completion and animation of people. In *ACM Special Interest Group on Computer Graphics*, 2005.

A. Asthana, S. Zafeiriou, S. Cheng, and M. Pantic. Robust discriminative response map fitting with constrained local models. In *Computer Vision and Pattern Recognition*, 2013.

V. Athitsos and S. Sclaroff. Estimating 3d hand pose from a cluttered image. In *Computer Vision and Pattern Recognition*, 2003.

A. Baak, M. Mller, G. Bharaj, H. P. Seidel, and C. Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *International Conference on Computer Vision*, 2011.

L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys. Motion capture of hands in action using discriminative salient points. In *European Conference on Computer Vision*, 2012.

C. M. Bishop. Mixture density networks. 1994.

C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.

V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *ACM Special Interest Group on Computer Graphics*, 1999.

E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6d object pose estimation using 3d object coordinates. In *European Conference on Computer Vision*, 2014.

Z. Cao, T. Simon, S. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Computer Vision and Pattern Recognition*, 2016.

J. Carreira, P. Agrawal, K. Fragkiadaki, and J. Malik. Human pose estimation with iterative error feedback. In *Computer Vision and Pattern Recognition*, 2016.

H. Chang, G. Garcia-Hernando, D. Tang, and T.-K. Kim. Spatio-temporal hough forest for efficient detection-localisation-recognition of fingerwriting in egocentric camera. *Computer Vision and Image Understanding*, 2016.

J. Charles, T. Pfister, D. Magee, D. Hogg, and A. Zisserman. Personalizing human video pose estimation. In *Computer Vision and Pattern Recognition*, 2016.

X. Chen and A. Yuille. Parsing occluded people by flexible compositions. In *Computer Vision and Pattern Recognition*, 2014.

C. Choi, A. Sinha, J. Choi, S. Jang, and K. Ramani. A collaborative filtering approach to real-time hand pose estimation. In *International Conference on Computer Vision*, 2015.

C.-S. Chua, H. Guan, and Y.-K. Ho. Model-based 3d hand posture estimation from a single 2d image. *Image and Vision Computing*, 2002.

C. Constantinopoulos, M. K. Titsias, and A. Likas. Bayesian feature and model selection for gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

T. F. Cootes, M. C. Ionita, C. Lindner, and P. Sauer. Robust and accurate shape model fitting using random forest regression voting. In *European Conference on Computer Vision*, 2012.

M. Dantone, J. Gall, G. Fanelli, and L. Van Gool. Real-time facial feature detection using conditional regression forests. In *Computer Vision and Pattern Recognition*, 2012.

J. Deutscher and I. Reid. Articulated body motion capture by stochastic search. *International Journal of Computer Vision*, 2005.

P. Dollár, P. Welinder, and P. Perona. Cascaded pose regression. In *Computer Vision and Pattern Recognition*, 2010.

Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Computer Vision and Pattern Recognition*, 2015.

A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 2007.

P. Felzenszwalb, D. McAllester, and D. Ramanan. A discriminatively trained, multiscale, deformable part model. In *Computer Vision and Pattern Recognition*, 2008.

P. F. Felzenszwalb and D. P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 2005.

V. Ferrari, M. Marin-Jimenez, and A. Zisserman. Progressive search space reduction for human pose estimation. In *Computer Vision and Pattern Recognition*, 2008.

M. A. Fischler and R. A. Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on computers*, 1973.

G. Garcia-Hernando, S. Yuan, S. Baek, and T. Kim. First-person hand action benchmark with RGB-D videos and 3d hand pose annotations. In *Computer Vision and Pattern Recognition*, 2018.

G. Ghiasi, Y. Yang, D. Ramanan, and C. C. Fowlkes. Parsing occluded people. In *Computer Vision and Pattern Recognition*, 2014.

R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon. Efficient regression of general-activity human poses from depth images. In *International Conference on Computer Vision*, 2011.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014.

G. Gkioxari, P. Arbelaez, L. Bourdev, and J. Malik. Articulated pose estimation using discriminative armlet classifiers. In *Computer Vision and Pattern Recognition*, 2013.

K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. Draw: A recurrent neural network for image generation. In *International Conference on Machine Learning*, 2015.

H. Guo, G. Wang, X. Chen, C. Zhang, F. Qiao, and H. Yang. Region ensemble network: Improving convolutional network for hand pose estimation. In *International Conference on Image Processing*, 2017.

A. Guzman-Rivera, P. Kohli, B. Glocker, J. Shotton, T. Sharp, A. Fitzgibbon, and S. Izadi. Multi-output learning for camera relocalization. In *Computer Vision and Pattern Recognition*, 2014.

A. Haque, B. Peng, Z. Luo, A. Alahi, S. Yeung, and L. Fei-Fei. Towards viewpoint invariant 3d human pose estimation. In *European Conference on Computer Vision*, 2016.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition*, 2016.

S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian Conference on Computer Vision*, 2012.

E. Hsiao and M. Hebert. Occlusion reasoning for object detectionunder arbitrary viewpoint. In *Computer Vision and Pattern Recognition*, 2012.

X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *European Conference on Computer Vision*, 2018.

Intel. Intel realsense. `https://realsense.intel.com/`.

C. Ionescu, J. Carreira, and C. Sminchisescu. Iterated second-order label sensitive pooling for 3d human pose estimation. In *Computer Vision and Pattern Recognition*, 2014.

M. Jaderberg, K. Simonyan, A. Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, 2015.

Y. Jang, S.-T. Noh, H. J. Chang, T.-K. Kim, and W. Woo. 3d finger cape: Clicking action and position estimation under self-occlusions in egocentric viewpoint. *IEEE Transactions on Visualization and Computer Graphics*, 2015.

M. Jiu, C. Wolf, G. Taylor, and A. Baskurt. Human body part estimation from depth images via spatially-constrained deep learning. *Pattern Recognition Letters*, 2014.

W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 1976.

A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. In *Computer Vision and Pattern Recognition*, 2018.

W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *International Conference on Computer Vision*, 2017.

A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *International Conference on Computer Vision*, 2015.

J. Kennedy and R. Eberhart. Particle swarm optimization. In *International Conference on Neural Networks*, 1995.

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 *Computer Vision and Pattern Recognition*, 2015.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2014.

K. Kinoshita, M. Delcroix, A. Ogawa, T. Higuchi, and T. Nakatani. Deep mixture density network for statistical model-based feature enhancement. In *International Conference on Acoustics, Speech, and Signal Processing*, 2017.

P. Krejov, A. Gilbert, and R. Bowden. Combining discriminative and model based approaches for hand pose estimation. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 2015.

A. Krull, E. Brachmann, F. Michel, M. Ying Yang, S. Gumhold, and C. Rother. Learning analysis-by-synthesis for 6d pose estimation in rgb-d images. In *International Conference on Computer Vision*, 2015.

A. Krull, E. Brachmann, S. Nowozin, F. Michel, J. Shotton, and C. Rother. Poseagent: Budget-constrained 6d object pose estimation via reinforcement learning. In *Computer Vision and Pattern Recognition*, 2017.

M. Lee, J. Cho, and S. Oh. Procrustean normal distribution for non-rigid structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

S. Li and A. B. Chan. 3d human pose estimation from monocular images with deep convolutional neural network. In *Asian Conference on Computer Vision*, 2014.

S. Li, W. Zhang, and A. B. Chan. Maximum-margin structured learning with deep networks for 3d human pose estimation. In *International Conference on Computer Vision*, 2015.

F. Michel, A. Krull, E. Brachmann, M. Y. Yang, S. Gumhold, and C. Rother. Pose estimation of kinematic chain instances via object coordinate regression. In *British Machine Vision Conference*, 2015.

V. Mnih, N. Heess, A. Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, 2014.

G. Moon, J. Y. Chang, and K. M. Lee. V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map. In *Computer Vision and Pattern Recognition*, 2018.

F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt. Real-time hand tracking under occlusion from an egocentric rgb-d sensor. In *International Conference on Computer Vision*, 2017.

R. Navaratnam, A. Fitzgibbon, and R. Cipolla. The joint manifold model for semi-supervised multi-valued regression. In *International Conference on Computer Vision*, 2007.

N. Neverova, C. Wolf, G. Taylor, and F. Nebout. Hand segmentation with structured convolutional learning. In *Asian Conference on Computer Vision*, 2014.

A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, 2016.

M. Oberweger and V. Lepetit. Deepprior++: Improving fast and accurate 3d hand pose estimation. In *International Conference on Computer Vision Workshops*, 2017.

M. Oberweger, P. Wohlhart, and V. Lepetit. Hands deep in deep learning for hand pose estimation. In *Computer Vision Winter Workshop*, 2015a.

M. Oberweger, P. Wohlhart, and V. Lepetit. Training a feedback loop for hand pose estimation. In *International Conference on Computer Vision*, 2015b.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *British Machine Vision Conference*, 2011a.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *International Conference on Computer Vision*, 2011b.

I. Oikonomidis, N. Kyriazis, and A. Argyros. Tracking the articulated motion of two strongly interacting hands. In *Computer Vision and Pattern Recognition*, 2012.

W. Ouyang, X. Chu, and X. Wang. Multi-source deep learning for human pose estimation. In *Computer Vision and Pattern Recognition*, 2014.

P. Panteleris, N. Kyriazis, and A. A. Argyros. 3d tracking of human hands in interaction with unknown objects. In *British Machine Vision Conference*, 2015.

G. Papandreou, T. Zhu, N. Kanazawa, A. Toshev, J. Tompson, C. Bregler, and K. P. Murphy. Towards accurate multi-person pose estimation in the wild. In *Computer Vision and Pattern Recognition*, 2017.

G. Poier, K. Roditakis, S. Schulter, D. Michel, H. Bischof, and A. Argyros. Hybrid one-shot 3d hand pose estimation by exploiting uncertainties. In *British Machine Vision Conference*, 2015.

C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *International Conference on Computer Vision*, 2014a.

C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *Computer Vision and Pattern Recognition*, 2014b.

U. Rafi, J. Gall, and B. Leibe. A semantic occlusion model for human pose estimation from a single depth image. In *Computer Vision and Pattern Recognition Workshops*, 2015.

V. Ramakrishna, T. Kanade, and Y. Sheikh. Reconstructing 3d human pose from 2d image landmarks. In *European Conference on Computer Vision*, 2012.

D. Ramanan. Learning to parse images of articulated bodies. In *Advances in Neural Information Processing Systems*, 2007.

M. Ranzato, S. Chopra, M. Auli, and W. Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

J. M. Rehg and T. Kanade. Visual tracking of high dof articulated structures: an application to human hand tracking. In *European Conference on Computer Vision*, 1994.

R. Rios-Cabrera and T. Tuytelaars. Discriminatively trained templates for 3d object detection: A real time scalable approach. In *International Conference on Computer Vision*, 2013.

G. Rogez, J. S. Supancic III, M. Khademi, J. M. M. Montiel, and D. Ramanan. 3d hand pose detection in egocentric rgb-d images. In *European Conference on Computer Vision Workshops*, 2014.

G. Rogez, J. S. Supancic, and D. Ramanan. First-person pose recognition using egocentric workspaces. In *Computer Vision and Pattern Recognition*, 2015.

M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *Computer Vision and Pattern Recognition*, 2012.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer Assisted Intervention*, 2015.

M. Salzmann and R. Urtasun. Combining discriminative and generative methods for 3d deformable surface and articulated pose reconstruction. In *Computer Vision and Pattern Recognition*, 2010.

P. Sermanet, A. Frome, and E. Real. Attention for fine-grained categorization. *arXiv preprint arXiv:1412.7054*, 2014.

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 *ACM Conference on Human Factors in Computing Systems*, 2015.

Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *Proceedings of IEEE International Conference on Evolutionary Computation*, 1998.

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 single depth images. In *Computer Vision and Pattern Recognition*, 2011.

L. Sigal and M. J. Black. Measure locally, reason globally: Occlusion-sensitive articulated pose estimation. In *Computer Vision and Pattern Recognition*, 2006.

T. Simon, J. Valmadre, I. Matthews, and Y. Sheikh. Kronecker-markov prior for dynamic 3d reconstruction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

S. Sridhar, A. Oulasvirta, and C. Theobalt. Interactive markerless articulated hand motion tracking using rgb and depth data. In *International Conference on Computer Vision*, 2013.

S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt. Fast and robust hand tracking using detection-guided optimization. In *Computer Vision and Pattern Recognition*, 2015.

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 *European Conference on Computer Vision*, 2016.

B. Stenger, P. R. S. Mendona, and R. Cipolla. Model-based 3D tracking of an articulated hand. In *Computer Vision and Pattern Recognition*, 2001a.

B. Stenger, P. R. Mendonça, and R. Cipolla. Model-based hand tracking using an unscented kalman filter. In *British Machine Vision Conference*, 2001b.

B. Stenger, A. Thayananthan, P. H. Torr, and R. Cipolla. Model-based hand tracking using a hierarchical bayesian filter. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *Computer Vision and Pattern Recognition*, 2015.

Y. Sun, X. Wang, and X. Tang. Deep convolutional network cascade for facial point detection. In *Computer Vision and Pattern Recognition*, 2013.

J. S. Supancic, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan. Depth-based hand pose estimation: data, methods, and challenges. In *International Conference on Computer Vision*, 2015.

D. J. Tan, T. Cashman, J. Taylor, A. Fitzgibbon, D. Tarlow, S. Khamis, S. Izadi, and J. Shotton. Fits like a glove: Rapid and reliable hand shape personalization. In *Computer Vision and Pattern Recognition*, 2016.

D. Tang, T.-H. Yu, and T.-K. Kim. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *International Conference on Computer Vision*, 2013.

D. Tang, H.-J. Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3D hand posture. In *Computer Vision and Pattern Recognition*, 2014.

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 *International Conference on Computer Vision*, 2015.

D. Tang, H. J. Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

D. Tang, Q. Ye, S. Yuan, J. Taylor, P. Kohli, C. Keskin, T. Kim, and J. Shotton. Opening the black box: Hierarchical sampling optimization for hand pose estimation. *TPMAI*, 2018.

J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The Vitruvian Manifold: Inferring dense correspondences for one-shot human pose estimation. In *Computer Vision and Pattern Recognition*, 2012.

J. Taylor, R. Stebbing, V. Ramakrishna, C. Keskin, J. Shotton, S. Izadi, A. Hertzmann, and A. Fitzgibbon. User-specific hand modeling from monocular depth sequences. In *Computer Vision and Pattern Recognition*, 2014.

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*, 2016.

J. Taylor, V. Tankovich, D. Tang, C. Keskin, D. Kim, P. Davidson, A. Kowdle, and S. Izadi. Articulated distance fields for ultra-fast tracking of hands interacting. *ACM Transactions on Graphics*, 2017.

T. T. D. Team, R. Al-Rfou, G. Alain, A. Almahairi, C. Angermueller, D. Bahdanau, N. Ballas, F. Bastien, J. Bayer, A. Belikov, et al. Theano: A python framework for fast computation of mathematical expressions. *arXiv preprint arXiv:1605.02688*, 2016.

J. Thies, M. Zollhfer, M. Stamminger, C. Theobalt, and M. Niener. Face2face: Real-time face capture and reenactment of rgb videos. In *Computer Vision and Pattern Recognition*, 2016.

A. Tkach, A. Tagliasacchi, E. Remelli, M. Pauly, and A. Fitzgibbon. Online generative model personalization for hand tracking. *ACM Transactions on Graphics*, 2017.

D. Tome, C. Russell, and L. Agapito. Lifting from the deep: Convolutional 3d pose estimation from a single image. In *Computer Vision and Pattern Recognition*, 2017.

J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics*, 2014.

A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Computer Vision and Pattern Recognition*, 2013.

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*, 2016a.

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*, 2016b.

M. Valstar, B. Martinez, X. Binefa, and M. Pantic. Facial point detection using boosted regression and graph models. In *Computer Vision and Pattern Recognition*, 2010.

E. Variani, E. McDermott, and G. Heigold. A gaussian mixture model layer jointly optimized with discriminative features within a deep neural network architecture. In *International Conference on Acoustics, Speech, and Signal Processing*, 2015.

C. Wan, A. Yao, and L. V. Gool. Hand pose estimation from local surface normals. In *European Conference on Computer Vision*, 2016.

F. Wang and Y. Li. Beyond physical connections: Tree models in human pose estimation. In *Computer Vision and Pattern Recognition*, 2013.

T. Wang, X. He, and N. Barnes. Learning structured hough voting for joint object detection and occlusion reasoning. In *Computer Vision and Pattern Recognition*, 2013.

S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Computer Vision and Pattern Recognition*, 2016.

J. Wu, T. Xue, J. J. Lim, Y. Tian, J. B. Tenenbaum, A. Torralba, and W. T. Freeman. Single image 3d interpreter network. In *European Conference on Computer Vision*, 2016.

J. Wu, Y. Wang, T. Xue, X. Sun, B. Freeman, and J. Tenenbaum. Marrnet: 3d shape reconstruction via 2.5d sketches. In *Advances in Neural Information Processing Systems*, 2017a.

Y. Wu, C. Gou, and Q. Ji. Simultaneous facial landmark detection, pose and deformation estimation under facial occlusion. In *Computer Vision and Pattern Recognition*, 2017b.

Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural

network for 6d object pose estimation in cluttered scenes. *Robotics: Science and Systems*, 2018.

X. Xiong and F. Torre. Supervised descent method and its applications to face alignment. In *Computer Vision and Pattern Recognition*, 2013.

C. Xu and L. Cheng. Efficient hand pose estimation from a single depth image. In *International Conference on Computer Vision*, 2013.

W. Yang, W. Ouyang, H. Li, and X. Wang. End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. In *Computer Vision and Pattern Recognition*, 2016.

Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *Computer Vision and Pattern Recognition*, 2011.

M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys. Accurate 3d pose estimation from a single depth image. In *International Conference on Computer Vision*, 2011.

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.

F. Yin and X. Chai, Xiujuanand Chen. Iterative reference driven metric learning for signer independent isolated sign language recognition. In *European Conference on Computer Vision*, 2016.

X. Yu, J. Huang, S. Zhang, W. Yan, and D. N. Metaxas. Pose-free facial landmark fitting via optimized part mixtures and cascaded deformable shape model. In *International Conference on Computer Vision*, 2013.

S. Yuan, Q. Ye, B. Stenger, S. Jain, and T. Kim. Bighand2.2m benchmark: Hand pose dataset and state of the art analysis. In *Computer Vision and Pattern Recognition*, 2017.

S. Yuan, G. Garcia-Hernando, B. Stenger, G. Moon, J. Y. Chang, K. M. Lee, P. Molchanov, J. Kautz, S. Honari, L. Ge, J. Yuan, X. Chen, G. Wang, F. Yang, K. Akiyama, Y. Wu, Q. Wan, M. Madadi, S. Escalera, S. Li, D. Lee, I. Oikonomidis, A. A. Argyros, and T. Kim. 3d hand pose estimation: From current achievements to future goals. In *Computer Vision and Pattern Recognition*, 2018.

H. Yub Jung, S. Lee, Y. Seok Heo, and I. Dong Yun. Random tree walk toward instantaneous 3d human pose estimation. In *Computer Vision and Pattern Recognition*, 2015.

H. Zen and A. Senior. Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis. In *International Conference on Acoustics, Speech, and Signal Processing*, 2014.

Y. Zhang and T. A. Funkhouser. Deep depth completion of a single RGB-D image. In *Computer Vision and Pattern Recognition*, 2018.

R. Zhao, Y. Wang, and A. M. Martínez. A simple, fast and highly-accurate algorithm to recover 3d shape from 2d landmarks on a single image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

X. Zhao, T.-K. Kim, and W. Luo. Unified face analysis by iterative multi-output random forests. In *Computer Vision and Pattern Recognition*, 2014.

X. Zhou, X. Sun, W. Zhang, S. Liang, and Y. Wei. Deep kinematic pose regression. In *European Conference on Computer Vision*, 2016a.

X. Zhou, Q. Wan, W. Zhang, X. Xue, and Y. Wei. Model-based deep hand pose estimation. In *International Joint Conferences on Artificial Intelligence*, 2016b.

X. Zhou, M. Zhu, S. Leonardos, K. G. Derpanis, and K. Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video. In *Computer Vision and Pattern Recognition*, 2016c.

J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In *Advances in Neural Information Processing Systems*, 2017.

S. Zhu, C. Li, C. Change Loy, and X. Tang. Face alignment by coarse-to-fine shape searching. In *Computer Vision and Pattern Recognition*, 2015a.

X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition*, 2012.

X. Zhu, Z. Lei, J. Yan, D. Yi, and S. Z. Li. High-fidelity pose and expression normalization for face recognition in the wild. In *Computer Vision and Pattern Recognition*, 2015b.

Y. Zhu, B. Dariush, and K. Fujimura. Controlled human pose estimation from depth image streams. In *Computer Vision and Pattern Recognition Workshops*, 2008.