UNIVERSITY OF CALIFORNIA,
IRVINE

Deep Learning in 3D Hand Pose and Mesh Estimation

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Liangjian Chen

Dissertation Committee:
Professor Xiaohui Xie, Chair
Professor Charless C. Fowlkes
Professor Shuang Zhao

2020

DEDICATION

To the five years I spent in Irvine

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First of all, I would like to express the deepest appreciation to my awesome advisor, Professor Xiaohui Xie. He is such a wonderful teacher, mentor, advisor and collaborator throughout the whole journey of my Ph.D. time. His knowledge, expertise and vision guides me to be an independent researcher. He built an excellent research environment for all the students in his lab including me to enjoy the life of pursuing pure knowledge. It is my great honor to work with him.

I am extremely grateful to Professor Charless Fowlkes and Professor Shuang Zhao, for their valuable suggestions on this dissertation.It is my great honor to have them in my both advancement and final defense committee. Meanwhile, I would like to extend my sincere thanks to Professor Roy Fox and Professor Jack Xin for joining my advancement committee.

I am fortunate to spend time in Tencent Medical AI Lab, where I meet my mentor Shih-Yao Lin and Yusheng Xie as well as Professor Yen-Yu Lin. I would also like to extend my deepest gratitude to them, for the tremendous help and advice I got from them.

I want to express my gratitude to my lab mates from my research group for having such a great research environment and atmosphere. The time I spent with your guys to study together and learn from each other would be a memorable journey in my life.I would like to extend my sincere thanks to Deying Kong and Hao Tang for their insightful input in tremendous research discussions among us.

I would like to thank all of my friends, especially Shi Dong, Yike Xue, Zhengli Zhao and Xinhan Tong for many emotional support during my difficult time. Also, another special thanks goes to James, Collin, Hickey and Xiaoyun Hu for their help of revising and proofreading this dissertation. I can hardly exhaust this list and hope my other friends can forgive me for not being able to mention their names here. It has been a luxury to have you along in this journey. Without you, I would not be able to make this achievement.

Lastly, but the most importantly, I would love to thank my parents Cong Chen and Ya Liang. The principle they taught me brings who I am today, and their unconditional love and support is the bedrock of my life.

VITA

Liangjian Chen

EDUCATION

Doctor of Philosophy in Computer Science **2016**
University of California, Irvine *Irvine, CA*

Bachelor of Engineer in Computer Science **2012**
Dalian University of Technology *Dalian, Liaoning*

INTERNSHIP EXPERIENCE

Machine Learning Intern **2019**
Facebook, Inc *Seattle, WA*

Research Intern **2018**
Tencent Medical AI Lab *Palo Alto, CA*

Ph.D. Intern **2017**
Google Assistant *Mountain View, CA*

TEACHING EXPERIENCE

Teaching Assistant **2016–2020**
University of California, Irvine *Irvine, California*

REFEREED CONFERENCE PUBLICATIONS

Temporal-Aware Self-Supervised Learning for 3D Hand Pose and Mesh Estimation in Videos **2021**
WACV

MVHM: A Large-Scale Multi-View Hand Mesh Benchmark for Accurate 3D Hand Pose Estimation **2021**
WACV

MM-Hand: 3D-Aware Multi-Modal Guided Hand Generation for 3D Hand Pose Synthesis **2020**
ACM Multi-Media

DGGAN: Depth-image Guided Generative Adversarial Networks for Disentangling RGB and Depth Images in 3D Hand Pose Estimation **2020**
WACV

TAGAN: Tonality-Aligned Generative Adversarial Networks for Realistic Hand Pose Synthesis **2019**
BMVC

Structured Triplet Learning with POS-tag Guided Attention for Visual Question Answering **2018**
WACV

ABSTRACT OF THE DISSERTATION

Deep Learning in 3D Hand Pose and Mesh Estimation

By

Liangjian Chen

Doctor of Philosophy in Computer Science

University of California, Irvine, 2020

Professor Xiaohui Xie, Chair

3D Hand pose estimation is an important problem because of its wide range of potential applications, such as sign language translation, robotics, movement disorder detection and monitoring, and human-computer interaction (HCI). However, despite the previous progress, it remains a challenge problem in the field of computer vision due to the difficulty to acquire high quality hand pose annotation. In this dissertation, we proposed various approaches to address this problem aiming for achieving a better estimation accuracy or providing an easier training environment. First, to bridge the image quality gap between the synthetic dataset and real world dataset, we proposed TAGAN(Tonality-Aligned Generative Adversarial Networks) to produce more realistic hand pose image. Second, to loose the requirement of paired RGB and Depth image requirement for most state-of-the-art 3D hand pose estimators, we proposed DGGAN(Depth-image Guided Generative Adversarial Networks) to enable those hand pose estimators to train on RGB-image-only dataset. Third, since the accurate 3D hand pose estimation is very difficult to obtain, we proposed TASSN(Temporal-Aware Self-Supervised Network) with temporal consistency constraints which learns 3D hand poses and meshes from videos with only 2D keypoint position annotations. Last but not the least, since 3D hand pose from a single image is intrinsically ill-posed. We proposed a multi-view hand pose and mesh benchmark to tackle this problem. A spin match algorithm was designed to enable our rigid mesh model matching with any target mesh ground truth. Based on the match algorithm, we proposed an efficient pipeline to generate a large-scale multi-view hand mesh (MVHM) dataset with accurate 3D hand mesh and joint labels. In addition to this, we also proposed a simple yet efficient multi-view based hand pose estimator which achieves the state-of-the-art result.

Chapter 1

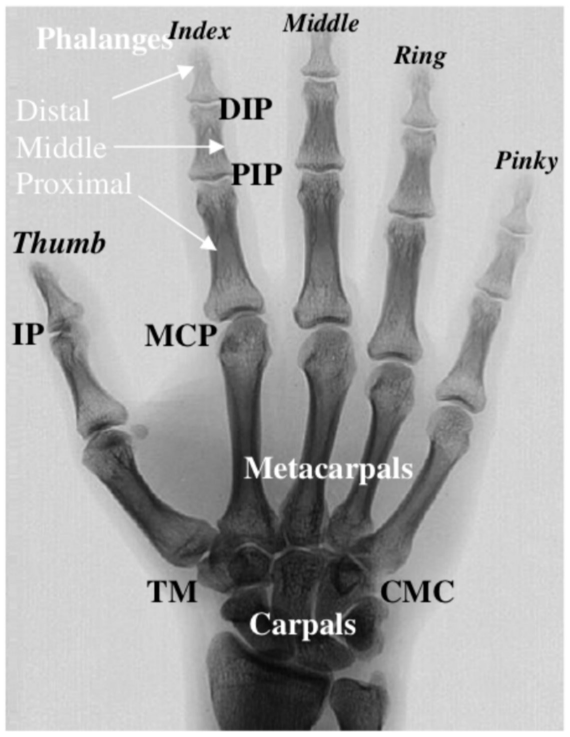
Introduction

1.1 Background

Estimation 3D hand pose from image data is a crucial task in computer vision because it relates to a lot of important downstream tasks such as sign language translation [98], robotics [2], movement disorder detection and monitoring, Augment Reality(AR)/Virtual Reality(VR) application, and human-computer interaction (HCI) [54, 37]. Despite the extensive effort made by prior work [34, 69, 30, 88, 9, 86, 96, 19, 89, 25, 29, 53], however, it remains a big challenge due to the hand intrinsic highly flexibility and a lacking of accurate 3D hand keypoint annotation. In this dissertation, we will first give a comprehensive review on the this problem, and then propose some new approaches to address this challenge.

1.1.1 Problem Formulation

The main goal of 3D hand pose estimation is estimating the hand keypoint location in 3D coordination system based on the input information. The input information can be various



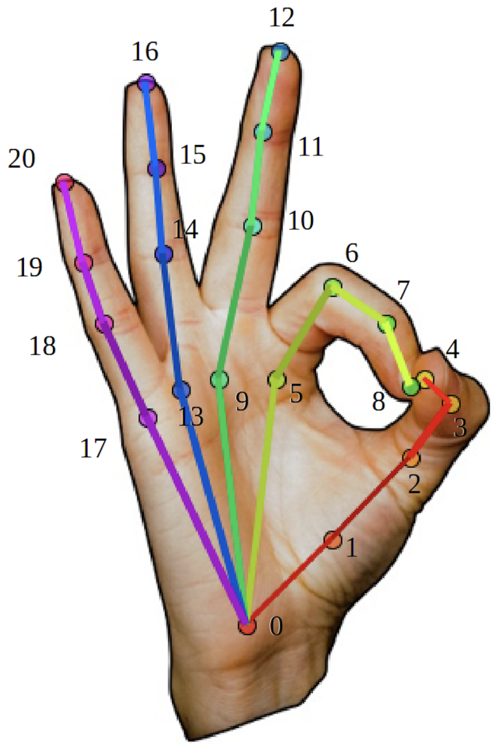


Figure 1.1: Left is a example of Hand Anatomy Graph from [22], right is example of the hand keypoint annotation. [74]

from case-to-case, including but is not limits to hand RGB images, hand video clips, the depth map from depth sensor, and optical flow. The hand keypoints are usually defined following the joints between the hand bones in hand anatomy. [22]. A example is shown in Fig 1.1. There is no convention that predicts how many joints are needed. But, typically, we estimate 21 hand joints including the Metacarpophalangeal(MCP), Proximal Interphalangeal(PIP), Distal Interphalangeal(DIP), and Phalanges tip for 5 fingers, as well as the the location of carpometacarpal joint(CMC).

1.2 Deep Learning Methods in 3D Hand Pose and Mesh Estimation

In recent years, the great success of Deep Learning method in computer vision field has been observed. [34, 69, 30, 88, 9] The 3D hand pose estimation, as an important topic in Computer Vision, also benefits with the progress of the deep learning methods. [105, 29, 26, 25] Pioneering work started on depth based information. Latter, estimating hand pose from RGB image got more attention. The recent research trend, in addition to estimation hand pose, is estimating hand pose and mesh together because the hand mesh provides much richer information.

1.2.1 Depth Based 3D Hand Pose Estimation

Earlier works focused more on depth image because depth image provides the surface depth information of the hands. [86, 96, 19, 89, 25, 29, 53]. Many previous works leveraged a deformed hand model and used regression to fit the model's parameters [55, 46, 49, 86]. Recently, PointNet was formulated to extract depth image features [25, 29, 67]. Each joint's final location was regressed by these extracted features. An end-to-end training framework was proposed by Wu *et al* [89]. It leveraged depth image as the intermediate guidance to regress the joint location. Despite the significant improvements on estimation accuracy, these methods all require accurate depth images as part of training input. This requirement limits their application due to a lacking of accessibility of depth sense in daily life.

1.2.2 RGB based 3D Hand Pose Estimation

RGB cameras are much more widely used than depth sensors. Estimating 3D hand poses merely from monocular RGB images are more practical and active in the literature [8, 15, 40, 59, 82, 93, 105, 15]. The pioneering work by Zimmermann and Brox [105] utilizes convolutional neural networks (CNN) to extract image features, and feed camera parameters with these features to a 3D lift network where depth information is then estimated. Based on [105], Iqbal *et al*[40] leverage depth maps as intermediate supervision. Meanwhile, Cai *et al* [8] propose a weakly supervised approach that reconstructs the depth map and uses it as a regularizer during model training.

1.2.3 3D Hand Mesh Estimation

3D hand pose estimation provides sparse joint locations. However, many computer vision applications would benefit more from hand shape information than sparse joints. Therefore, 3D hand mesh estimation, an effective shape representation, has emerged as an increasingly popular topic [28, 6, 5, 45, 101]. Most methods [6, 5, 101, 52, 92] are developed around a pre-defined deformable hand mesh model called MANO [70]. Because of the high degree of freedom and complexity of the hand gesture, searching for the right hand mesh in such a high dimensional space is quite challenging. Using this MANO model often relies on strong prior to constrain the model to only regress low-dimensional model parameters, and may ignore the high-dimensional information. Ge *et al* [28] argue that mesh is a graph-structure data, and propose to directly regress 3D mesh vertices through graph convolutional neural network (GCN) with a pre-defined mesh graph.

1.3 Difficulties in 3D Hand Pose and Mesh Estimation

Despite the steady progress made by aforementioned works, 3D hand pose and mesh estimation remains a challenge problem due to the following reasons.

1. The highly flexibility of the hand makes it hard to be capture by the model. Recent work usually a model MANO [70] to approximate the hand mesh. However, Mano, as a linear model, may not be able to capture higher dimension information. Ge *et al*[28] try to address this problem by taking hand mesh as graph and made substantial progress. However, their method requires a large additional dataset with hand mesh ground, which is rare among all the public hand benchmark,

2. The annotation of the hand is hard to obtain. Since manually labelling the 3D hand pose groundturth are extensively labor-intensive, in most existing real word datasets, 3D hand keypoint annotations is often obtained by fitting with some models [83, 106, 6], which maybe introduce errors or bias. To tackle this question, Zimmermann & Brox [105] introduce the synthetic hand dataset. However, though synthetic datasets provide high quality 3D hand annotation, they suffer from the low quality of synthetic images.

3. Most existing methods leveraging mesh information for 3D hand pose estimation are derived based on a single view. However, estimating hand pose and shape from a single RGB image is intrinsically ill-posed due the depth ambiguity.

In this thesis, we proposed several methods to address these problems.

1.4 Dissertation Outline and Contribution

The general outline of the rest of the dissertation is as follows:

Chapter 2: In this Chapter, we present an effective method for generating realistic hand poses and show that existing algorithms for hand pose estimation can be greatly improved by augmenting training data with the generated hand poses, whose ground-truth annotations could be easily obtained. Specifically, we adopt an augmented reality simulator to synthesize hand poses with accurate 3D hand-keypoint annotations. These synthesized hand poses look unnatural and are not adequate for training. To produce more realistic hand poses, we propose blending each synthetic hand pose with a real background and developing *Tonality-Alignment Generative Adversarial Networks* (TAGAN), which align the tonality and color distributions between synthetic hand poses and real backgrounds, and can generate high-quality hand poses.

This Chapter is based on the work originally published in: **Liangjian Chen**, Shih-Yao Lin, YuSheng Xie, Hui Tang, Yifan Xue, Yen-Yu Lin, Xiaohui Xie, Wei Fan, "Generating Realistic Training Images Based on Tonality-Alignment Generative Adversarial Networks for Hand Pose Estimation", BMVC 2019. [15]

Chapter 3: In this Chapter, we propose a conditional generative adversarial network (GAN) model, called Depth-Image Guided GAN (DGGAN), to generate realistic depth maps conditioned on the input RGB image and using the synthesized depth maps to regularize the 3D hand pose estimation model, therefore eliminating the need for ground-truth depth maps.

This Chapter is based on the work originally published in: **Liangjian Chen**, Shih-Yao Lin, YuSheng Xie, Yen-Yu Lin, Wei Fan, Xiaohui Xie, "DGGAN: Depth-image Guided Generative Adversarial Networks for Disentangling RGB and Depth Images for 3D Hand Pose Estimation", WACV 2020. [12]

Chapter 4: In this Chapter, we propose a new framework of training 3D pose estimation models from RGB images without using explicit 3D annotations, i.e., trained with only 2D information. This framework is motivated by two observations: 1) Videos provide richer information for estimating 3D poses as opposed to static images; 2) Estimated 3D poses ought to be consistent whether the videos are viewed in the forward order or reverse order. We leverage these two observations to develop a self-supervised learning model called temporal-aware self-supervised network (TASSN). By enforcing temporal consistency constraints, TASSN learns 3D hand poses and meshes from videos with only 2D keypoint position annotations.

This Chapter is based on the work originally published in: **Liangjian Chen**, Shih-Yao Lin, YuSheng Xie, Yen-Yu Lin, Xiaohui Xie, "Temporal-Aware Self-Supervised Learning for 3D Hand Pose and Mesh Estimation in Videos", WACV 2021. [14]

Chapter 5: Training hand pose estimators with 3D hand mesh annotations and multi-view images often results in significant performance gains. However, existing multi-view datasets are relatively small with hand joints annotated by off-the-shelf trackers or automated through model predictions, both of which may be inaccurate and can introduce biases.

In this chapter, we design a spin match algorithm that enables a rigid mesh model matching without any target mesh ground truth. Based on the match algorithm, we propose an efficient pipeline to generate a large-scale multi-view hand mesh (MVHM) dataset with accurate 3D hand mesh and joint labels. We further present a multi-view hand pose estimation approach to verify that training a hand pose estimator with our generated dataset greatly enhances the performance.

This Chapter is based on the work originally published in: **Liangjian Chen**, Shih-Yao Lin, YuSheng Xie, Yen-Yu Lin, Xiaohui Xie, "MVHM: A Large-Scale Multi-View Hand Mesh Benchmark for Accurate 3D Hand Pose Estimation", WACV 2021. [13]

Chapter 6: Concludes this dissertation and presents several open directions for future research.

Chapter 2

TAGAN: Tonality-Aligned Generative Adversarial Networks for Realistic Hand Pose Synthesis

2.1 Introduction

Estimating hand poses from monocular RGB images has drawn increasing attention because it is essential to many applications such as virtual and augmented reality [59], and human computer interaction [77]. It has gained significant progress [24, 59] owing to the fast development of *deep neural networks* (DNN). These DNN-based methods learn hand representations and estimate poses jointly. Despite effectiveness, DNN-based methods highly rely on a vast amount of training data. However, it is expensive to collect all hand poses of interest with manual hand-keypoint annotations for training.

Synthesizing training data has been a feasible way to tackle the lack of training data. Recent studies, *e.g*[59, 105], have adopted *augmented reality* (AR) simulators to generate large-scale

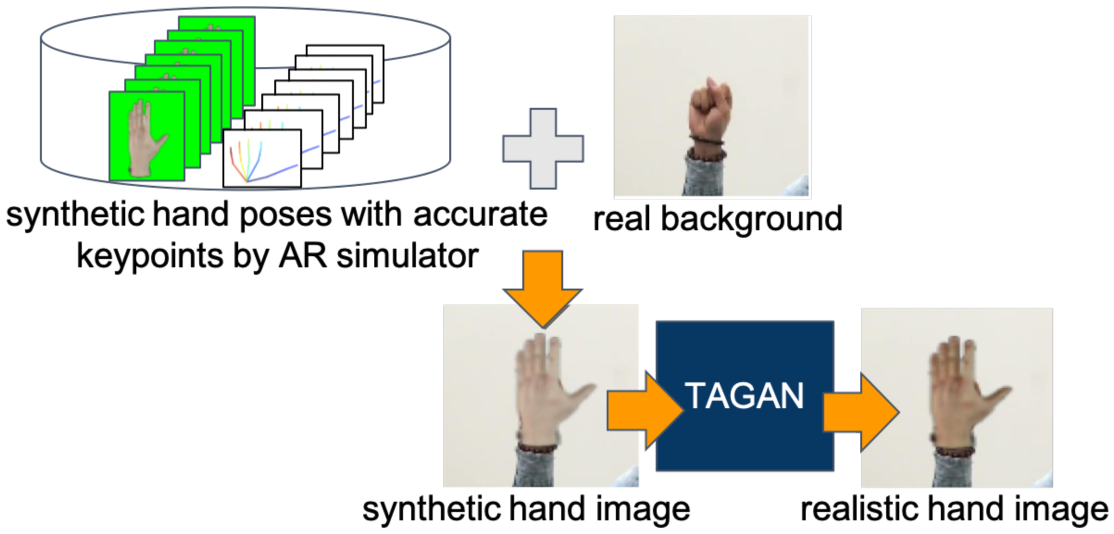

synthetic hand poses with accurate keypoints by AR simulator
real background
TAGAN
synthetic hand image
realistic hand image

Figure 2.1: Overview of our method for realistic hand image synthesis. We blend a synthetic pose by an AR simulator with real background to yield a synthetic hand image, which is then fed to the proposed TAGAN to produce a more realistic hand image.

training examples. In this way, plenty hand images with various poses, skin textures, and lighting conditions can be systematically synthesized. Moreover, accurate hand-keypoint annotations of these synthesized hand images are also available. Training with such synthetic images may not result in a much improved hand pose estimator because of the dissimilarity between the real and synthetic data. In this work, we suggest blending a synthetic hand pose (foreground) image with a real background image so that the blended images are realistic enough to serve as high-quality training data.

We are aware of the dissimilarity between synthetic hand pose images and real background images in styles and appearances. Thus, we present a GAN-based method, *tonality-alignment generative adversarial networks* (TAGAN), to eliminate the dissimilarity. TAGAN employs the image-to-image translation technique based on *conditional GAN* (CGAN) [42], where extra shape features serve as the input to GAN and constrain the object shape in the synthesized photo. In addition to the shape constraint, a tonality-alignment loss in TAGAN is designed to align the color distributions tonality of the input and generated images. It turns out that the hand pose images can be better blended into the background images, resulting in more realistic hand pose images. The hand pose estimator is then considerably improved by using the generated hand pose images as the augmented training data.

Figure 2.1 gives the overview of the proposed method. The main contribution of this work is three-fold: First, we propose to fuse synthetic hand poses and real background images so that the resulting synthesized hand images can be more realistic. Second, we present TAGAN which performs conditional adversarial learning and seamlessly blends synthetic hand poses into real backgrounds. Third, we demonstrate that existing pose estimators trained with the generated hand pose data gain significant improvements over the current state-of-the-arts on both 2D and 3D datasets.

2.2 Related Work

2.2.1 Data Augmentation via Simulator.

Recent work, *e.g*[50], for hand pose estimation has trained the models on synthetic training data. In [105], a synthetic hand pose dataset is generated by an open source simulator, and serves as augmented training data to improve pose estimator learning. However, the synthetic hand images produced by the AR simulator look artificial, leading to limited performance gains. To address this issue, recent work, *e.g*[73, 59], leverages adversarial learning to enhance the quality of synthetic hand images.

2.2.2 Data Augmentation via Adversarial Learning.

Generating realistic images by using *generative adversarial networks* (GAN) [32, 73] has been a research trend. Isola *et al*propose *Pix2Pix Net* [43] to learn a mapping from a sketch to a realistic image, *e.g*transferring a car sketch to a car image. Unlike GAN requiring paired training data, *CycleGAN* [104] employs cycle-consistent adversarial networks for translating images from a source domain to a target domain with unpaired examples. To increase the amount of training data, Shrivastava *et al*present *SimGAN* [73], which employs simulated and unsupervised learning to improve the realism of the output of a simulator with unlabeled real data. However, the simulator's data include only objects, ignoring background scenes. Thus, the resulting synthetic images are filled objects, but the background information is usually crucial in practice. In this work, we explore techniques that directly regularize the foreground (hand) and the background (natural scenes where the hand appears [99, 39]).

2.2.3 Vision-based Hand Pose Estimation.

Hand pose estimation has drawn increasing attention for decades [1, 4, 36, 24, 95, 87, 26, 57, 64, 94, 102, 28, 6, 82, 93]. Research efforts can be categorized by their input data forms, which primarily include 2D RGB images and 3D RGBD images with depth information. Recent progress has tried to estimate the 3D hand pose from a monocular RGB image. For example, Oikonomidis *et al*[62, 63] propose a hand tracking approach based on *particle swarm optimization*. Simon *et al*[75] adopt multiview bootstrapping to calculate hand keypoints from RGB images. Zimmermann and Brox [105] propose a 3D pose regression net, enabling 3D hand pose estimation from an RGB image.

2.2.4 Domain Adaption via Adversarial Learning.

Domain adaption has been introduced by Saenko *et al* [72] for pairwise metric transforming and can be developed by the study of visual dataset bias [84]. For domain adaption, models are often designed to capture the invariant patterns between two different data distributions so that they can perform well cross domains. Hoffman *et al* [35] adapt the CycleGAN loss with the task loss to further improve the results of image translation.

2.3 Methodology

This section describes our approach to hand pose image generation. We first explain how GAN and conditional GAN are applied to this problem, then depict synthetic hand image generation, and finally specify how our approach works to improve the synthetic images.

2.3.1 GAN and Conditional GAN

GAN learns a mapping from a random noise vector z to its generated image y, *i.e.,* $G : z \rightarrow y$, where G is the generator. The *conditional GAN* (CGAN) [56] is an extension of GAN. The inputs to CGAN can be augmented with additional conditions so that CGAN can leverage the additional conditions to constrain the output image y. The conditions can be specified in the form of extra inputs to both the generator network and the discriminator network.

CGAN-based methods can be applied to image-to-image translation. In the representative work *pix2pix net* [42], the condition is used to make the object shape in the output image similar to the additional input shape map x_s, which is pre-computed by applying the edge detector HED [91] to the image x. As the additional input, x_s is fed to both the generator and the discriminator. In this way, the condition x_s and the latent space representation z are transformed into a joint hidden representation. CGAN learns a mapping from x_s (inferred from x) and z to the generated output y, G: $\{x, z\} \rightarrow y$. The CGAN objective can be formulated as

$$\mathcal{L}_{CGAN}(G, D) = \mathbb{E}_{x_s,y}[\log D(x_s, y)] + \mathbb{E}_{x_s,z}[\log(1 - D(x_s, G(x_s, z)))], \tag{2.1}$$

where the discriminator D aims to distinguish the data generated by G from real data. Yet, the generator G generates the data to not only fool D but also fulfill the input condition. In Eq. (2.1), the generator G minimizes the differences between the real images and the generated images while G's adversary, D, tries to learn a discriminating function to maximize such differences. In addition, the shape of output y is constrained by x_s. Thus, G is optimized via

$$G^* = \arg\min_G \max_D \mathcal{L}_{CGAN}(G, D) + \lambda \cdot \mathcal{L}_S(G), \tag{2.2}$$

where $\mathcal{L}_s$ and λ are the shape loss and its weight, respectively. The shape loss can be calculated by using L_1 distance to reduce the unfavorable effect of blurring, and is defined by

$$\mathcal{L}_S(G) = \mathbb{E}_{x_s,y,z}[\|y - G(x_s, z)\|_1]. \tag{2.3}$$

Although the existing work, Pix2pix Net is able to contain the shape of the generated object by using the input shape map, it does not take color consistency between the object and background into account. Hence, its generated images might look unnatural.

2.3.2 Synthetic Hand Image Generation

Existing hand pose datasets are not large enough to stably learn a deep network for hand pose estimation. Moreover, manual hand-keypoint annotation is expensive and labor-intensive. Also, the annotated hand-keypoints are still error-prone and often not accurate enough. To address the quantitative and qualitative issues of hand pose training data, we firstly adopt an open-source AR simulator to produce large-scale *synthetic hand pose images* with *accurate 2D/3D hand-keypoint labels*, which cover common and feasible hand poses by observing a

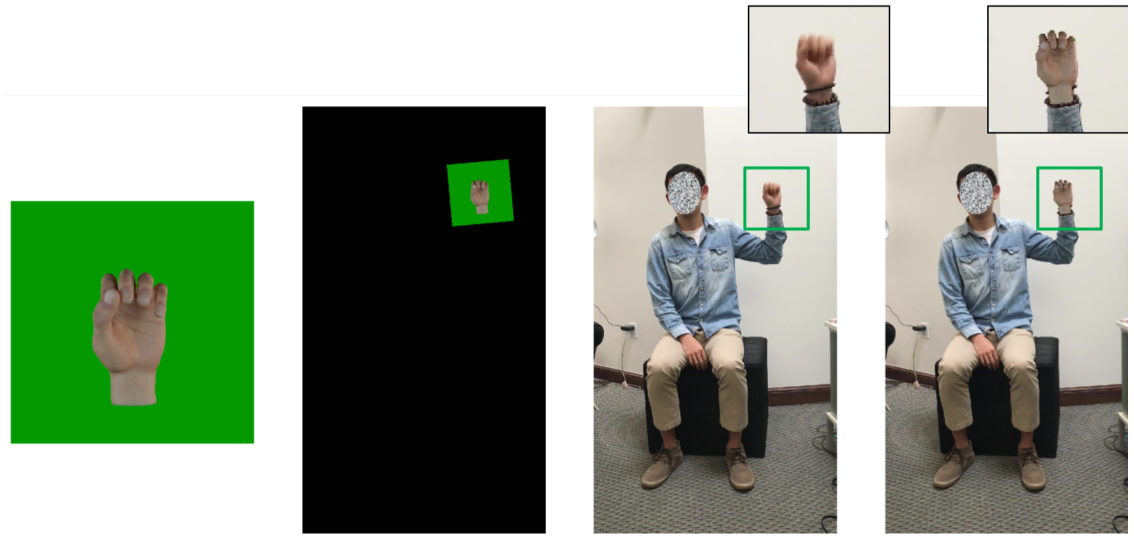


Figure 2.2: Hand image synthesis. From left to right: 1) a hand image **u** in our synthetic dataset, 2) its representation with an affine transformation $g(\mathbf{u})$, 3) target image (enclosed by the green box) **v** via hand detection, and 4) the synthesized hand image obtained by applying TAGAN to blend the synthetic hand pose into the real background image.

group of subjects for a period of time. However, these synthetic hands look unnatural and cannot serve as training data.

We also collect a large-scale, daily-life, and unlabeled hand gesture videos performed by some subjects. Each image in these videos consists of *real hand(s) and background*. See the third image in Figure 2.2 as an example. We detect the hand region in the image using the pose estimation toolkit, OpenPose Library [73]. However, the estimated poses are not good enough to serve as the training data. Thus, we propose to *match* the hand pose images in the AR and real datasets. In this way, the accurate keypoints in the AR dataset and real backgrounds in the real dataset can complement each other, and the proposed TAGAN can produce more realistic hand poses with accurate keypoints.

Given a hand image with the estimated pose **v** in the real dataset, our goal is to synthesize a hand image with its pose consistent with **v**. To find the best match, we use **v** as a query to the AR dataset generated by the AR simulator, which covers millions of hand poses with accurate keypoint annotations. For each candidate hand pose **u** in the AR dataset, its

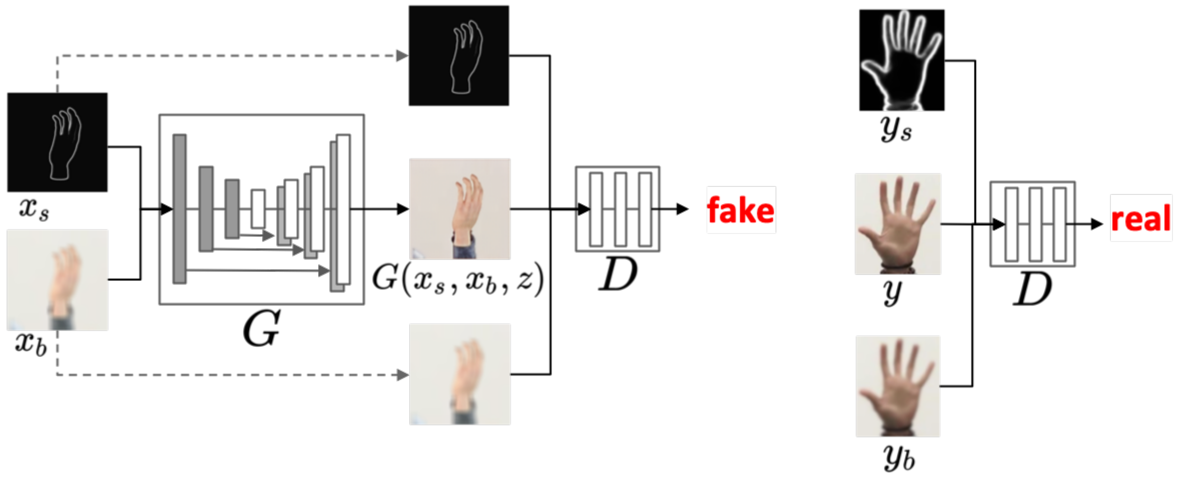


Figure 2.3: TAGAN derives a mapping from the shape map x_s and the color map x_b to the generated image $G(x_s, x_b, z)$. The generator G learns to produce realistic images to fool the discriminator D by blending a synthetic hand pose with a real background image, while the discriminator D aims to separate the fake (synthetic) images from the real images.

similarity to the target pose $\mathbf{v}$ is defined as

$$K(\mathbf{u}, \mathbf{v}) = \langle f \circ g(\mathbf{u}), f(\mathbf{v}) \rangle / (\|f \circ g(\mathbf{u})\| \|f(\mathbf{v})\|), \tag{2.4}$$

where function g is the affine transformation with which the transformed candidate pose $g(\mathbf{u})$ can best match $\mathbf{v}$, and function f is the feature representation of a pose. In this work, each hand pose is expressed as the concatenation vector of its 21 2D keypoints, $e.g \mathbf{u} = [u_{x1}, u_{y1}, u_{x2}, u_{y2}, \ldots, u_{x21}, u_{y21}] \in \mathbb{R}^{42}$. The feature representation f of a hand pose is the ordered collection of pair-wise keypoint differences along the x and y axes, *i.e.*,

$$f(\mathbf{u}) = [\ldots, u_{xi} - u_{xj}, u_{yi} - u_{yj}, \ldots], \text{ for } 1 \leq i < j \leq 21. \tag{2.5}$$

The candidate pose $\mathbf{u}^* = \arg\max_{\mathbf{u}} K(\mathbf{u}, \mathbf{v})$ is selected from the dataset yielded by the AR simulator. We superpose pose $\mathbf{u}^*$ over the scene covering $\mathbf{v}$, and apply the proposed TAGAN to better blend the selected pose into the background scene to produce a more realistic hand image. Figure 2.2 summarizes the process. The proposed TAGAN is elaborated below.

2.3.3 Tonality-Alignment GAN

Although data augmentation using AR simulators can relieve the lack of training data, the background of the synthesized images is artificial. The background tonality and color distributions between the synthetic and real hand poses are inconsistent. These issues make the synthetic hand poses less qualified as training data. Inspired by the pix2pix net [43] that leverages the shape map to constrain the output image, we propose *tonality-alignment* GAN (TAGAN) to take the color distribution and shape features into account.

Given a superposed image x, we utilize its blurred counterpart x_b and shape map x_s as the color and shape reference, respectively. The blurred counterpart in our system is derived by applying an average filter to x, while the shape map x_s is obtained by using the HED detector. For the real image y, we adopt the same scheme to extract the shape map y_s and color maps y_b. In TAGAN, the shape map x_s and color map x_b are fed to both the generator and the discriminator as additional input layers such that the x_s, x_b and the output $G(x_s, x_b, z)$ are transformed into a joint hidden representation. Figure 2.3 illustrates the proposed TAGAN.

During training, the TAGAN learns a mapping from x_s, x_b and a random vector z to the generated output y, *i.e.,* G:$\{x_s, x_b, z\} \rightarrow y$. The objective of TAGAN is designed as

$$\mathcal{L}_{TAGAN}(G, D) = \mathbb{E}_{y_s,y_b,y}[\log D(y_s, y_b, y)] + \mathbb{E}_{x_s,x_b,z}[\log(1 - D(x_s, x_b, G(x_s, x_b, z))]. \quad (2.6)$$

The generator G in TAGAN is optimized via

$$G^* = \arg\min_G \max_D \mathcal{L}_{TAGAN}(G, D) + \mathcal{L}_{TA}(G, x_s, x_b), \quad (2.7)$$

where $\mathcal{L}_{TA}$ is the loss function for enforcing the shape similarity between x_s and y as well as the color consistency between x_b and y. The loss is defined by

$$\mathcal{L}_{TA}(G) = \mathbb{E}_{x_s,x_b,y,z}[\lambda_1 \cdot D_c(x_b, x_s, z, y) + \lambda_2 \cdot D_s(x_b, x_s, z, y)], \tag{2.8}$$

where $D_c(\cdot)$ and $D_s(\cdot)$ denote the color and shape distance functions, respectively. Constants λ_1 and λ_2 are the weights. The shape distance function D_s is expressed as

$$D_s(x_b, x_s, z, y) = \|y - G(x_s, x_b, z)\|_1. \tag{2.9}$$

In addition to the shape condition, we design a tonality-alignment loss to align the color distributions of the input and the generated images via defining D_c as

$$D_c(x_b, x_s, z, y) = -\sum_i h_g(i) \log\left(\frac{h_y(i)}{h_g(i)}\right), \tag{2.10}$$

where h_y and h_g are the color histograms of y and $G(x_b, x_s, z)$, respectively. Thereby, $D_c(\cdot)$ in Eq. (2.10) is the Kullback-Leibler divergence between the two histograms. To train our TAGAN, we collect a real unlabeled hand dataset, called *RHTD*. The hand images in the RHTD are utilized to be real examples y. Some examples in RHTD are shown in Figure 2.4.

2.4 Experimental Setting

Hand Pose Estimators. Two existing hand pose estimators are used to assess the quality of the synthesized hand pose images in our experiments: *Hand3D* [105] and *convolutional pose machine* (CPM) [88]. The two estimators infer the 3D and 2D hand poses from a monocular RGB image, respectively. Both estimators are popular and often serve as the baselines for advanced estimators, such as [8, 59, 65, 74, 96]. Thus, if the synthetic hand pose images we generate can improve Hand3D and CPM, those images can also facilitate

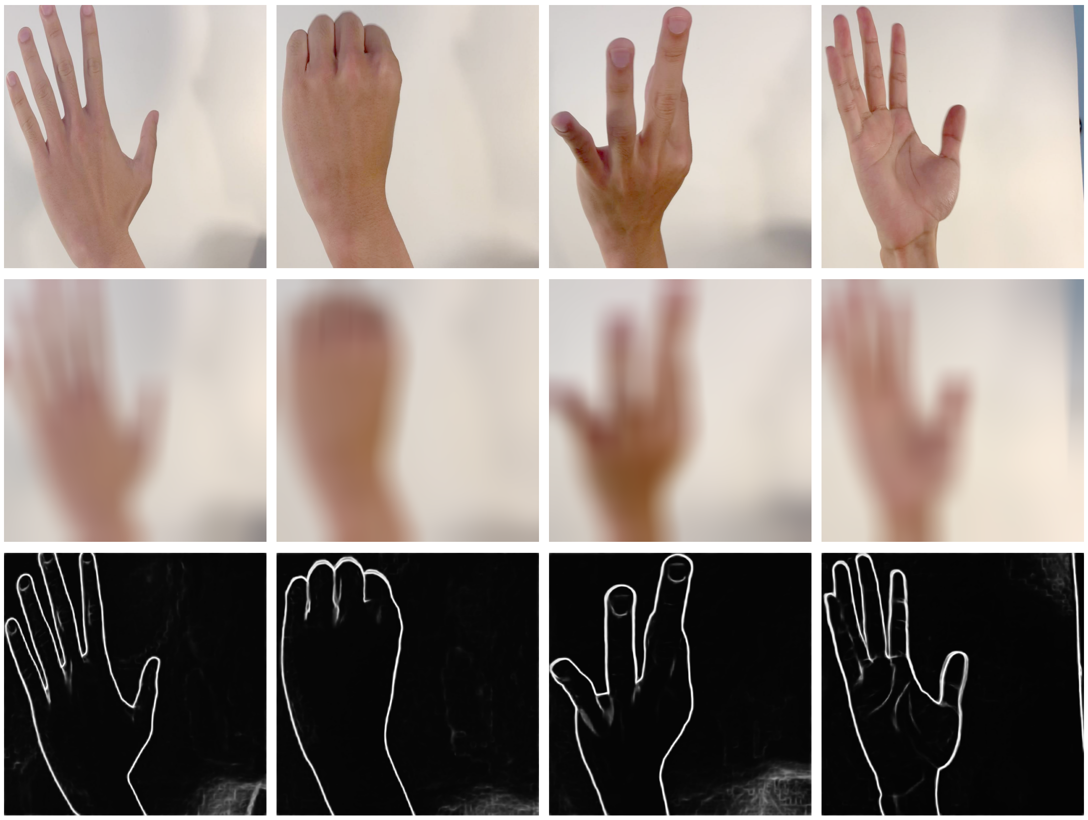


Figure 2.4: Four examples of the RHTD dataset upon which TAGAN is learned. Fist row contains images x, seconds row contains corresponding color maps x_b and third row contains corresponding shape maps x_s.

the follow-up research of Hand3D and CPM.

Dataset for Training. To train the generators, including CycleGAN, pix2pix net, and the proposed TAGAN, for hand image synthesis, we collect $17,040$ real unlabeled hand images captured from people performing various hand gestures. This dataset is called the *real hand training dataset* (RHTD), which contains hand images with various poses, perspective views, and lighting conditions. Some examples of RHTD are shown in Figure 2.4. To train the proposed TAGAN, images in RHTD come with the pre-computed edge [91] and color maps. In addition to RHTD, we adopt the AR simulator to generate $60,000$ synthetic hand images with various poses, perspectives, and lighting conditions. Some synthetic hand image examples are displayed in the first column of Figure 2.6. By using our synthetic hand image

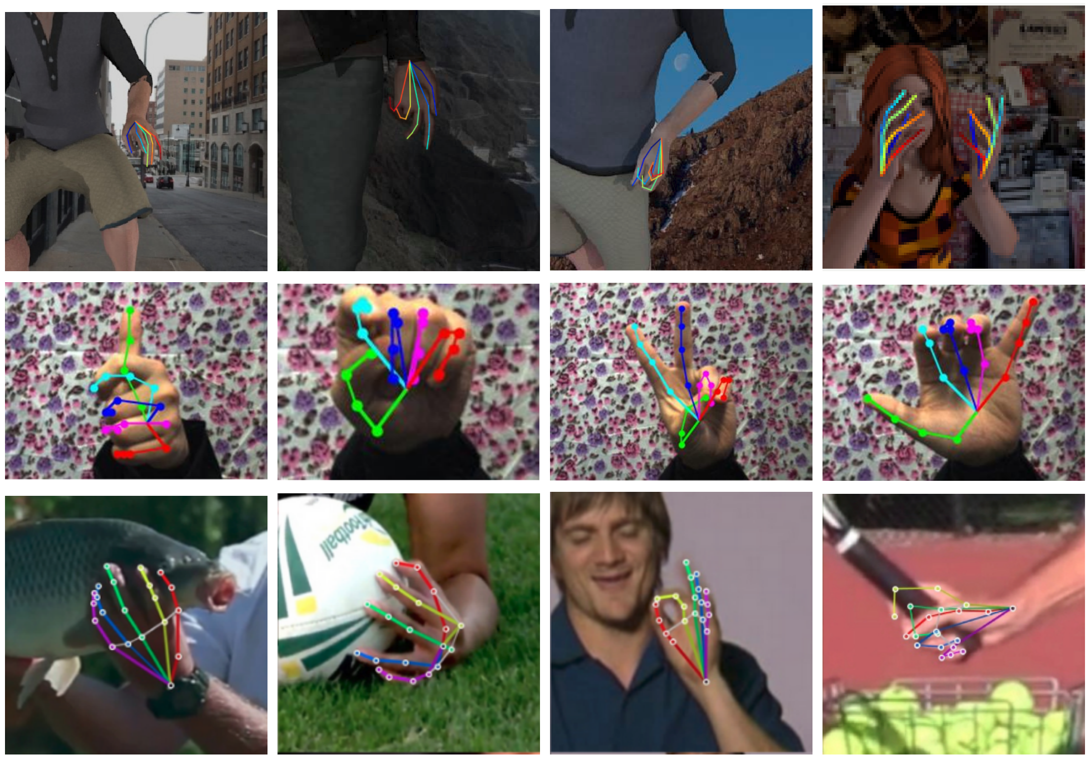


Figure 2.5: Some examples in the three benchmark datasets. First row is the RHP dataset provides synthetic hand images with 3D hand keypoints. Second row is The STB dataset contains real hand images with 3D keypoints. Third row is CMU-PS dataset offers real hand images with 2D keypoints.

generation process shown in Figure 2.2, the synthetic hand images are then present at the appropriate locations of the real images (background). The TAGAN is then applied to blend the synthetic hands with the real background images.

Datasets for Evaluation. To evaluate the quality and the efficacy of the synthesized data, we select three benchmark datasets for evaluation including the *Rendered Hand Pose* (RHP) [105], *Stereo Tracking Benchmark* (STB) [100], and *CMU Panoptic Studio* (CMU-PS) [74] datasets. Figure 2.5 displays some examples of the three datasets. The RHP dataset contains $41,258$ training and $2,728$ testing hand samples captured from 20 subjects performing 39 actions. Each sample consists of an RGB image, a depth map, and the segmentation masks for the background, person, and each finger. Each hand is annotated with its 21 keypoints in both 2D coordinates and 3D world coordinate positions. The RHP dataset is split into a validation set (R-val) and a training set (R-train). The STB dataset provides $18,000$ hand images. It is split into two subsets: the stereo subset (STB-BB) and the color-depth subset (STB-SK). The CMU-PS dataset provides $1,912$ examples for training and 846 examples for testing. The 2D hand keypoints of these examples are available.

Evaluation Metrics. Following [8, 74, 105], we adopt two metrics for evaluating the estimated hand poses, including the average *End-Point-Error* (EPE) and the *Area Under the Curve* (AUC) on the *Percentage of Correct Keypoints* (PCK). We report the performance on both 2D and 3D hand pose estimation where the performance metrics are computed in pixels (px) and millimeters (mm), respectively. The performance of 3D hand joint prediction is measured by using the PCK curves averaged over all 21 keypoints. We use 2D PCK and 3D PCK to evaluate our approach on the RHP, STB, and CMU-PS datasets, respectively.

2.5 Experimental Results

This section evaluates the proposed TAGAN. First, we visually inspect the hand images generated by TAGAN and compare them with those generated via CycleGAN [102] and Pix2pix Net [42]. Second, we perform an ablation study to independently verify the efficacy of additional hand pose data and the importance of using TAGAN as a background-aware generator. Third, we show that the hand pose estimators can be greatly improved by fine-tuning with the data generated by TAGAN.

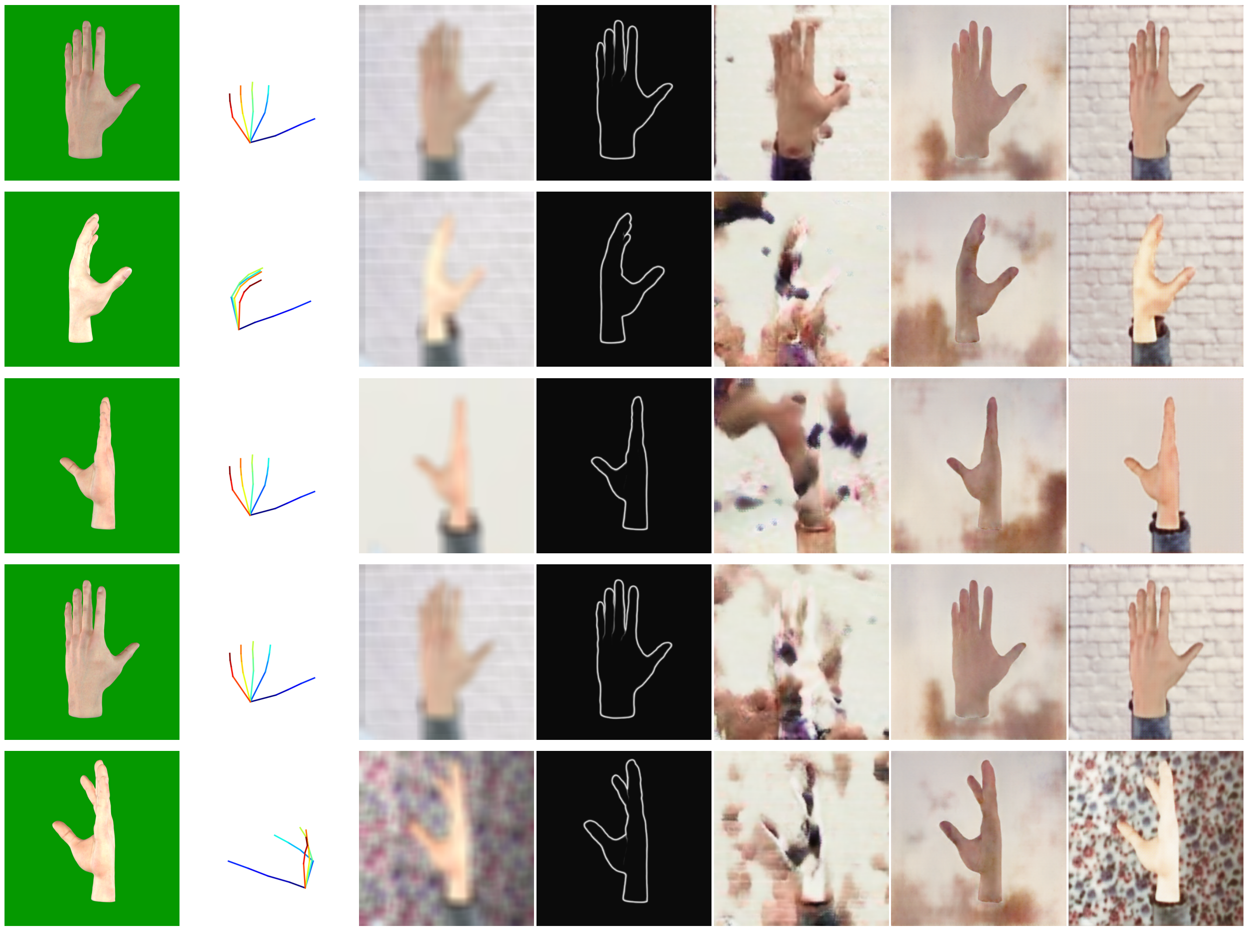


Figure 2.6: Comparison of the hand pose images synthesized by different methods. Images from left to right are 1) hand poses generated by the AR simulator, 2) their keypoint annotations, 3) color maps, 4) shape maps, the synthesized results by 5) CycleGAN [102], 6) Pix2pix Net [42], and 7) the proposed TAGAN, respectively.

2.5.1 Comparison of the Synthesized Hand Pose Images

Figure 2.6 compares the quality of the generated hand images by using CycleGAN, Pix2pix Net, and the proposed TAGAN, respectively. The CycleGAN learns one translation mapping from synthetic hand images to real images, and another mapping from real to synthetic images. The Pix2pix Net derives the translation from a shape map to a realistic image. TAGAN takes both foreground shapes and background tonality into account. In Figure 2.6, the first two columns show the synthetic hand images generated by the AR simulator and their hand-keypoint labels, respectively. The third and the forth columns show the pre-computed color maps and shape maps from those synthetic hand images. The last three columns display the results generated by CycleGAN, Pix2pix Net, and TAGAN, respectively.

As shown in Figure 2.6, the CycleGAN does not have the shape and color constraints, so it tends to generate unnatural hand images. Although the Pix2pix Net can successfully generate hand's shapes by using shape features, it does not consider the consistency of colors. The generated hand images still look unnatural. The proposed TAGAN leverages real background information and gains color and shape constrains. Hence, the generated results jointly maintain the color and shape features, making the synthesized images more realistic.

Table 2.1: Ablation study for 3D pose estimation results.

	AUC↑	EPE mean↓
RHP	0.42	35.6
RHP+AR Hand w/ CBG	0.56	26.4
RHP+AR Hand w/ RBG	0.57	26.4
RHP+AR Hand w/ TAGAN	**0.60**	**24.2**
STB	0.66	15.7
STB+AR Hand w/ CBG	0.67	8.2
STB+AR Hand w/ RBG	0.71	7.1
STB+AR Hand w/ TAGAN	**0.75**	**7.0**

Table 2.2: Ablation study for 2D pose estimation results.

	PCK@20 ↑	EPE mean ↓
RHP	88.45	10.76
RHP+AR Hand w/ RBG	90.00	9.80
RHP+AR Hand w/ TAGAN	**90.01**	**9.78**
STB	96.6	7.61
STB + AR Hand w/ RBG	96.7	7.59
STB + AR Hand w/ TAGAN	**97.0**	**7.55**

2.5.2 Ablation Study

For analyzing the pose estimator learning by using our generated data, we conduct an ablation study on both 2D and 3D hand pose estimation. We adopt three comparative settings of

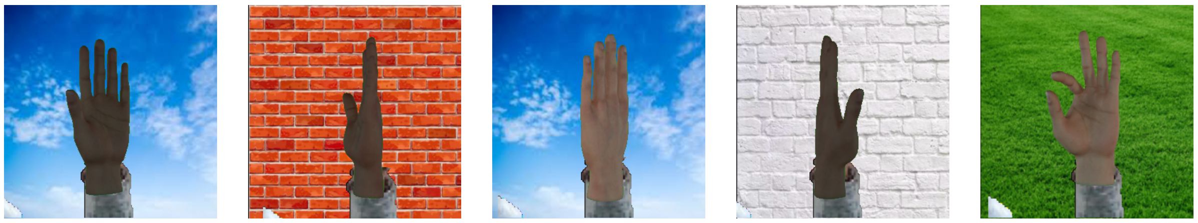


Figure 2.7: Tonality inconsistency in the "AR Hand w/RBG" data.

training data on the STB datasets, including 1) STB data with hand images produced by the AR simulator with clear background "STB+AR hand w/ CBG", 2) STB data with AR hand images with real background "STB+AR hand w/ RBG", and 3) STB with hand images generated by TAGAN "STB+AR hand w/ TAGAN". Some examples of the "STB+AR hand w/ CBG" and "STB+AR hand w/ TAGAN" are shown in the first and the last columns of Figure 2.6, respectively. We train the hand pose estimators, Hand3D and CPM, on the training sets using the settings above, and test them on the validation sets, respectively. All the aforementioned experimental settings are also conducted on the RPH dataset.

Table 2.1 and Table 2.2 show the experimental results on 3D and 2D hand pose estimation, respectively. In EPE and PCK, we find that both 3D and 2D hand pose estimators can be greatly improved by using large-scale synthetic data. Besides, Table 2.1 and Table 2.2 show that the pose estimators trained with AR hands and real background images can be further improved. The reason is that the augmented AR hand images with real backgrounds are closer to the real images. Moreover, it can be observed that training hand pose estimators with data generated by TAGAN has higher PCK values and lower EPE errors than with the "AR Hand w/ RBG" data.

For 3D pose estimation task, the pose estimator trained with "AR Hand w/ CBG" is successfully improved in EPE mean 11.4 $(= 35.6 - 24.2)$mm and 8.7 $(= 15.7 - 7.0)$mm on the RHP and STB datasets, respectively, as shown in Table 2.1. For 2D pose estimation, the pose estimators are improved in EPE mean $0.98(= 10.76 - 9.78)$ pixels and $0.06(= 7.61 - 7.55)$

Table 2.3: Results by training on STB and testing on CMU-PS.

	PCK@20 ↑	EPE mean (mm) ↓
STB	24.09	55.99
STB+AR Hand w/ RBG	24.82	56.67
STB+AR Hand w/ TAGAN	**28.81**	**52.93**

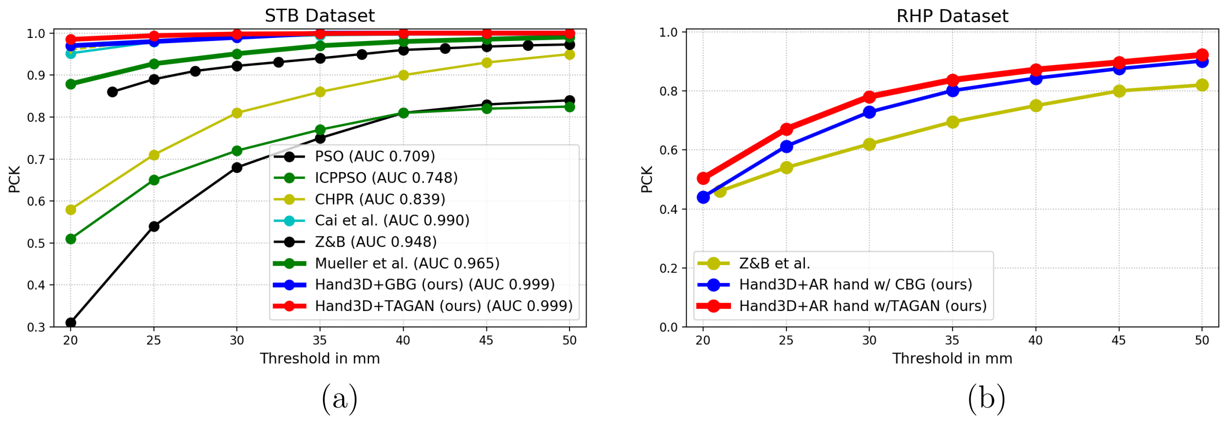


Figure 2.8: Comparison with the state-of-the-art approaches for 3D pose estimation on the (a) STB and (b) RHP datasets.

pixels on the RHP ans STB dataset, respectively, as shown in Table 2.2. The quality of the data generated by TAGAN is better than the synthetic AR hand superimposed with real backgrounds (AR hands w/ RBG). The reason is that the "AR Hand w/ RBG" data do not take the tonality consistency between synthetic hands and background into account. Some examples of tonality inconsistency are shown in Figure 2.7. To test the generation ability, we follow [74] where hand pose estimators are trained on the STB training set and evaluated on the CMU-PS validation set. We adopt CPM in this experiment. As shown in Table 2.3, training the pose estimator with augmented hand images (AR hand w/ RBG) can enhance the performance. Moreover, training it with the data generated by TAGAN can gain more significant improvement in both PCK accuracy and EPE error. The CPM is improved from 24.09 to 28.81 in PCK@20 accuracy and from 55.99 to 52.39 in EPE mean error.

2.5.3 Comparison with State-of-the-Art Results

To compare with the existing 3D hand pose estimators, we select the methods, PSO, ICPPSO, and CHPR as the baselines, and choose the state-of-the art methods including Cai *et al*in [8], Z&B in [105], and Mueller *et al*in [59] for comparison on the STB dataset. We select Z&B [105] for comparison on the RHP dataset. We also provide the results by training our pose estimator with "AR hand w/ RBG" data for comparison. Figure 2.8 shows that training Hand3D with training data generated by TAGAN achieves the best performance. To explore the improvement by training a 2D/3D pose estimator with our generated data, we conduct two experiments on STB and RHP datasets. We adopt Hand3D and CPM as the 2D and 3D pose estimators. Table 2.1 and 2.2 show the results. We find that training either 2D or 3D pose estimators with the additional images generated by TAGAN reaches the best performance.

2.6 Conclusions and Future Work

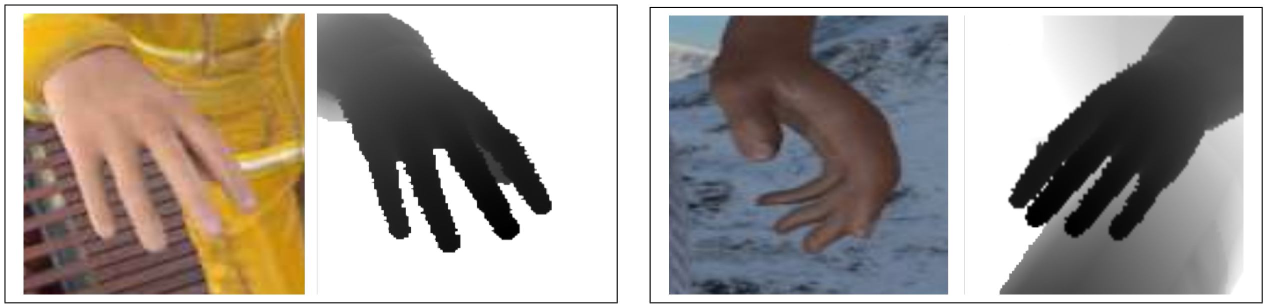
This study presents a novel data augmentation approach for improving hand pose estimation task. To produce more realistic hand images for training pose estimators, we propose TAGAN, a conditional adversarial networks model, to blend the synthetic hand poses with real background images. Our generated results align the hand shape and color tonality distribution between synthetic hands and real background images. The experimental results show that the state-of-the-arts hand pose estimators can be greatly improved with the aid of the training data generated by our method.

Chapter 3

DGGAN: Depth-image Guided Generative Adversarial Networks for Disentangling RGB and Depth Images in 3D Hand Pose Estimation

3.1 Introduction

Vision-based 3D hand pose estimation (3D hand pose estimation) aims to estimate the 3D keypoint coordinates of a given hand image. 3D hand pose estimation has drawn increasing attention owing to its wide applications to human-computer interaction (HCI) [2, 54], sign language understanding [98], augmented/virtual reality (AR/VR) [59, 37], and robotics [2]. RGB images and depth maps are two the most commonly used input data for the 3D hand pose estimation task. An example of a hand image and its corresponding depth map is shown in Figure 3.1(a). Depth map can provide 3D information related to the distance of



(a) (b)

Figure 3.1: Training examples in a generic 3D hand pose estimation dataset: (a) paired RGB and depth images; (b) unpaired RGB and depth images. Our work does not rely on paired training data and therefore is applicable to both RGB-only and depth-only 3D hand pose estimation tasks.

the surface of human hands. Training networks with depth maps has been proven to achieve significant progress on the 3D hand pose estimation task [8, 40]. In addition, with the depth information provided by the depth maps, the hand segmentation task can be effectively solved. Unfortunately, capturing depth maps often requires specific sensors (*e.g* Microsoft Kinect, RealSense), which limits the usability of those state-of-the-art methods based on depth maps. Commercial depth sensors are usually much more expensive than RGB cameras. On the other hand, RGB images are the most commonly used input data in the hand pose estimation task because it can be easily captured by abundant low-cost optical sensors such as webcams and smartphones. However, 3D hand pose estimation from RGB images is a challenging task.

In the absence of depth information, estimating 3D hand pose from a monocular RGB image is intrinsically an ill-posed problem. To address this issue, the state-of-the-art methods such as [8, 28] leverage both RGB hand images and their paired depth maps for the 3D hand pose estimation task. Their 3D hand pose inference process takes an RGB image and the paired depth information into account. They first regress 3D hand poses on RGB images, and then utilize a separate branch to regularize the predicted 3D hand pose by using the paired depth maps. The objective of the depth regularizer is to make the predicted 3D keypoint positions consistent with the provided depth map. It results in two major advantages: 1) training networks with depth maps can efficiently improve the hand pose estimator by using the depth information to reduce the ambiguity and 2) enabling 3D hand pose estimation based on merely RGB images during the inference stage. These approaches require paired RGB and depth training images. Unfortunately, most existing hand pose datasets only contain either depth maps or RGB images, instead of both. It makes the aforementioned approaches not applicable to such datasets. Besides, the unpaired RGB and depth training images cannot be explorited for them. Figure 3.1(b) shows an example of unpaired RGB and depth map images.

To tackle this problem, we propose a novel generative adversarial networks, called *Depth-image Guided GAN* (DGGAN). Our network contains two modules: *depth-map reconstruction* and *hand pose estimation*. The main idea of our approach is to directly reconstruct the depth map from an input RGB hand image in the absence of paired RGB and depth training images. Given an RGB image, our depth-map reconstruction module aims to infer its depth map. Our hand pose estimation module takes RGB and depth information into account to infer the 3D hand pose. In the hand pose estimation module, we infer the 2D hand keypoints on the input RGB image, and regress the 3D hand pose by using the inferred 2D keypoints. The depth map is then used to regularize the inferred 3D hand pose. Unlike most existing 3D hand pose estimation models, the real depth maps used to train our DGGAN model do not require any paired RGB images. Once DGGAN is learned, the proposed hand pose estimation module directly infers the hand pose by using an RGB image and guided (regularized) by a DGGAN-inferred depth map. Since the depth-map can be inferred by our depth-map reconstruction module, the proposed DGGAN no longer requires paired RGB and depth images. Our DGGAN jointly trains the two modules in an end-to-end trainable network architecture. Experimental results on multiple benchmark datasets demonstrate that our DGGAN not only reconstructs the depth map of an input RGB image, but also significantly improves the 3D hand pose estimator via an additional depth regularizer.

The main contributions of this study are summarized as follows:

1. We propose a depth-map guided adversarial neural networks (DGGAN) for 3D hand pose estimation from RGB images. Our network can jointly infer the depth information from input RGB images and estimate the 3D hand poses.

2. We introduce a depth-map reconstruction module to infer the depth maps from input RGB images while learning to predict 3D hand poses. Our DGGAN is trained on readily accessible hand depth maps that are not paired with RGB images.

3. Experimental results demonstrate that our approach achieves new state-of-the-art in 3D hand pose prediction accuracy on three benchmark datasets, including the RHD, STB, and MHP datasets.

3.2 Related Work

3.2.1 3D hand pose estimation from Depth Images

3D hand pose estimation from depth mapshas been extensively studied. Existing approaches in this field make noticeable advances [86, 96, 19, 89, 25, 29, 53]. Wan *et al* [86] propose a dense regression approach to fit the parameters of a deformed hand model. Ge *et al* [25, 29] present PointNet[67] to extract hand features and regress hand joint locations by referring to the extracted features. Wu *et al* [89] adopt the intermediate dense guidance map supervision to generate hand heatmaps. Although the existing methods achieve very accurate estimation results, they typically rely on the hand data captured by high-precision depth sensors, which are still expensive to have in practice and usually require data collection in a lab environment. Different from the models in the aforementioned methods, our model performs inference on RGB data without the need of depth maps.

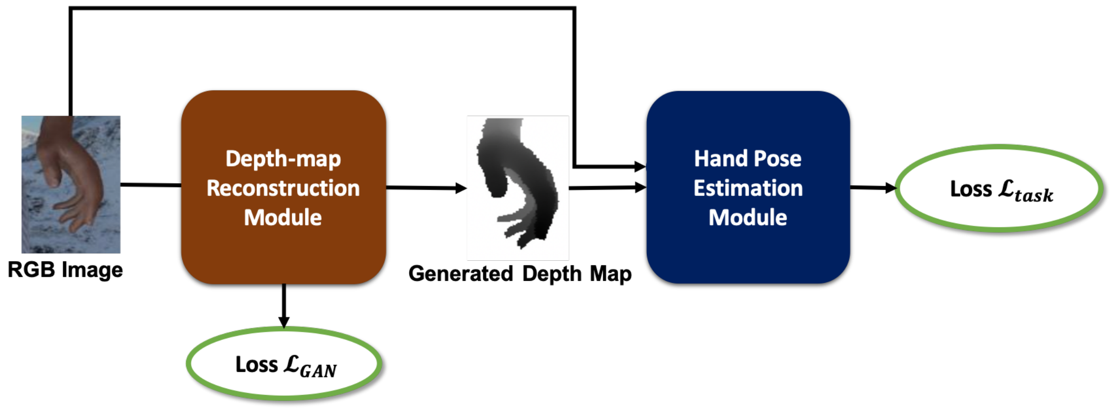

RGB Image
Depth-map Reconstruction Module
Generated Depth Map
Hand Pose Estimation Module
Loss $\mathcal{L}_{task}$
Loss $\mathcal{L}_{GAN}$

Figure 3.2: Overview of the proposed DGGAN. DGGAN consists of two modules, a *depth-map reconstruction module* shown in Figure 3.4 and a *hand pose estimation module* shown in Figure 3.4. The former module trained using the GAN loss aims at inferring the depth map of a hand based on the input RGB image and making the generated depth map looks realistic. The latter module trained using the task loss estimates hand poses from the input RGB and the GAN-reconstructed depth images.

3.3 Related Work

3.3.1 3D hand pose estimation from Monocular RGB Images

Due to the wide availability of RGB cameras, 3D hand pose estimation from monocular RGB images is becoming increasingly popular in computer vision applications. Many recent methods aim at estimating hand joint locations directly from a single RGB image [8, 40, 28, 105, 59, 15, 93, 6, 82]. Zimmermann *et al*[105] use 2D convolutional neural networks (CNN) to extract features from an RGB image, and regress the 3D hand joint locations. However, their method suffers from depth ambiguity due to the absence of depth information. Developing the methods upon the work by Zimmermann *et al*, Iqbal *et al* [40] and Cai *et al* [8] inherit and adopt a similar 2D CNN architecture for extracting image features. Iqbal *et al*use depth maps as intermediate guidance while Cai *et al*treat depth maps as a regularizer in a weakly supervised manner. Though these two methods make substantial progress in terms of estimation accuracy, there currently exist few datasets that fulfill their requirement of paired depth maps and RGB images. Ge *et al* [28] take one step further by predicting the hand mesh from an RGB image and then the 3D hand joint locations based on the mesh. However, their method requires paired mesh information which is even rarer among all existing datasets.

Compared with these methods, our method also uses depth information during training, but it does not require any paired RGB images and depth maps. Thus, it is much more flexible since it can consume RGB images and depth maps from different datasets or sources.

3.3.2 3D Mesh Estimation from RGB Images

To further enhance 3D hand pose estimation [5, 6, 28, 45], hand mesh estimation can be included. Namely, the model estimates not only the hand joints but also the hand surface mesh. However these methods such as [28] have a common drawback: They require additional mesh annotations which are even more expensive to obtain than joint locations. Thus, they are typically trained on synthetic datasets due to this limitation. Seungryul *et al* [6] introduce an iterative learning method to refine mesh shapes and achieve very good performance. However, like 3D hand joint locations, hand meshes highly rely on additional supervision from hand segment maps which are typically not available in nowadays hand pose datasets. The method by boukhayma *et al* [6] is the only extra-data-free method, but its performance is limited.

3.3.3 GAN-based Image Translation

Generating images using generative adversarial networks (GAN) [33] has gained remarkable progress. Many approaches explore how to better manipulate images by applying GAN models [35, 44, 103, 16]. Isola *et al* [44] propose the Pix2Pix network which translates label or edges maps to synthesized photos, reconstructs objects from edge maps, or colorizes images. Zhu *et al* [103] introduce the cycle-consistent generated adversarial network (CycleGAN). CycleGAN uses the cycle consistency loss to disentangle the input and output pair and therefore does not need paired input. Hoffman *et al* [35] propose cycle-consistent adversarial domain adaptation (CyCADA). Compared to CycleGAN, CyCADA contains a segmentation loss. As a result, CyCADA not only translates images from one modality to another but also deals with a specific visual task.

Applying the generative adversarial model to RGB hand images for hand pose estimation is also gaining popularity. Muller *et al* [59] introduce the geometry consistent GAN (GeoConGAN) to generate synthetic image data for training. Chen *et al* [15] propose the tonality-alignment generative adversarial networks (TAGAN) for producing more realistic images from synthetic images for hand pose estimator training. However, these methods only focus on generating RGB images. None of them generates depth maps for assisting hand pose estimator training.

3.4 Our Approach

Our goal is to estimate the 3D hand pose from a monocular RGB hand image. Although the existing state-of-the-art methods [6, 68, 96] have shown that training networks with RGB and depth images can improve the 3D hand pose estimators, few 3D hand pose datasets consist of paired RGB and depth images. To deal with the lack of paired data issue, we propose a novel adversarial neural network, called depth-map guided generated adversarial networks (DGGAN) illustrated in Figure 3.2, which can jointly learn to infer the depth map from an RGB image of hand and to estimate 3D hand pose. In the following, we give an overview of the proposed DGGAN and describe the two major modules of DGGAN in detail.

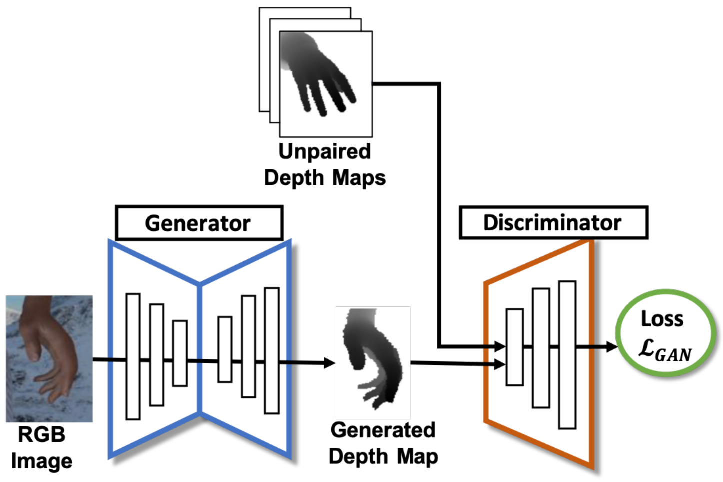


Figure 3.3: Network architecture of the depth-map reconstruction module.

3.4.1 Overview of DGGAN

The proposed DGGAN consists of two major modules, a *depth-map reconstruction* module and a *hand pose estimation* module. Its network architecture is shown in Figure 3.2.

Given an RGB hand image $\mathbf{I}$, we want to estimate the K 3D hand joint locations $\mathbf{J}^{xyz} \in \mathbb{R}^{3\times K}$. Each column in the $3 \times K$ matrix is a vector of size 3 and represents the (x, y, z) coordinates of a joint, i.e., $\mathbf{J}^{xyz} = [J_1^{xyz}, J_2^{xyz}, \ldots, J_K^{xyz}]$.

The two modules in the proposed DGGAN G are trained by using the GAN loss $\mathcal{L}_{GAN}$ and the task loss $\mathcal{L}_{task}$, respectively. The objective of learning G is formulated as a min-max game:

$$G^* = \arg\min_G \max_D (\lambda_t \mathcal{L}_{task} + \lambda_g \mathcal{L}_{GAN}), \tag{3.1}$$

where λ_t and λ_g control the relative importance of these two loss terms.

Given an RGB hand image, our depth-map reconstruction module tries to generate its corresponding depth map. A set of unpaired training depth images is adopted to train the depth-map reconstruction module so that its inferred depth maps are similar to real ones.

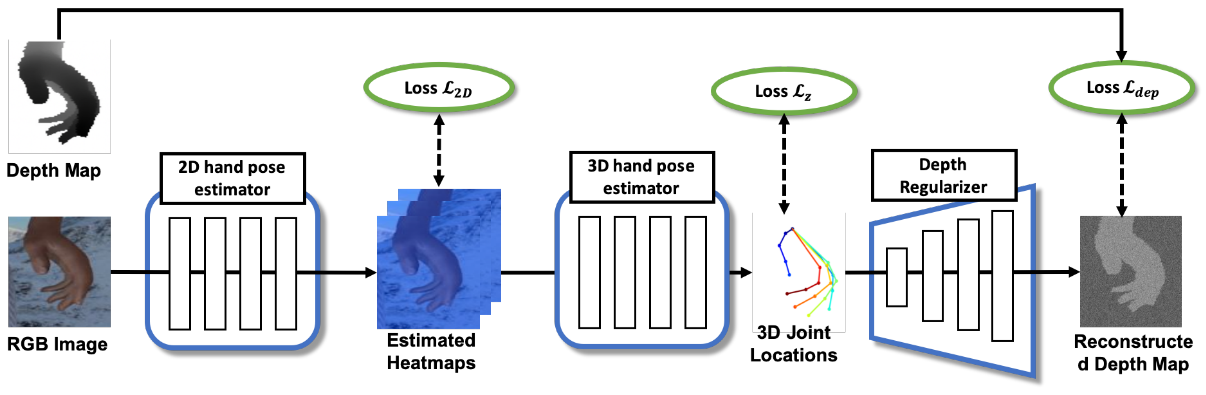


Figure 3.4: Architecture of the hand pose estimation module. This module takes paired RGB images and inferred depth maps as inputs. 2D CPM consumes an RGB image as input and produces the hand joint heatmap. The joint heatmap is fed to the regression network to estimate the 3D joint locations with the aid of a depth regularizer. The depth regularizer reconstructs the depth map from 3D joint locations and is trained using L1 loss and the GAN-synthesized depth map as guidance.

To achieve that, the discriminator in this module works on distinguishing real depth maps from fake (generated) ones. Section 3.4.2 describes the details of depth-map reconstruction. The depth map inferred from the depth-map reconstruction module together with the input RGB image are fed to the hand pose estimation module for estimating the 3D hand pose. In the hand pose estimation module, the input RGB image is used to regress the 3D hand pose. The inferred depth-map is adopted to regularize the predicted 3D hand pose. The loss for hand pose estimation $\mathcal{L}_{task}$ is adopted for optimization. Section 3.4.3 describes the details.

3.4.2 Depth-map Reconstruction Module

The depth-map reconstruction module aims at relaxing the requirement of paired RGB and depth images during training. This module is constructed via an adversarial network that infers the depth map according to an input RGB image. Figure 3.3 shows the network architecture of this module. In the training phase, our network requires both depth and

RGB training images. Nevertheless, the RGB and depth images do not need to be paired. We consider the process of inferring depth map from its corresponding RGB image as an unsupervised adaptation problem, where the RGB modality S and depth modality T are both provided. We are given a set of RGB images X_S and a set of real depth maps X_T. To translate from S to T, we adopt an encoder-decoder architecture $G_{S \to T}$. The generator $G_{S \to T}$ is trained to generate a realistic depth map to fool the discriminator D while D id derived to distinguish the real data x_t and generated fake data $G_{S \to T}(x_s)$. The loss for the depth-reconstruction modules is as follows:

$$\mathcal{L}_{GAN}(G_{S \to T}, D, X_S, X_T) = \mathbb{E}_{x_t \sim X_T}[\log D(x_t)] + \mathbb{E}_{x_s \sim X_S}[\log(1 - D(G_{S \to T}(x_s)))]. \quad (3.2)$$

This loss also provides semantic constraints to force the generator to produce more realistic depth maps. By taking as input unpaired RGB and depth images, our depth-map reconstruction module becomes applicable to vastly more hand pose datasets. Furthermore, we can train the network with a large amount of unpaired RGB and depth images.

3.4.3 Hand Pose Estimation Module

Given an inferred depth map computed by the depth-map reconstruction module, we combine it with the input RGB image and feed both to the hand pose estimation module. The network architecture of the hand pose estimation module is shown in Figure 3.4. The hand pose estimation module calculates the task loss $\mathcal{L}_{task}$, which is composed of two terms $\mathcal{L}_{task} = \mathcal{L}_{2D} + \mathcal{L}_z$. The 3D hand regression loss $\mathcal{L}_{2D}$ and depth regularization loss $\mathcal{L}_z$ are described in section 3.4.3 and 3.4.3, respectively.

3D Hand Pose Regression

Previous studies [8] show that depth information can be used to build a powerful regularizer. We leverage the depth regularizer for improving the result of 3D hand pose estimation. Unlike most previous works where the ground-truth depth maps are needed, our model uses a synthetic depth map generated by the depth-map reconstruction module. Our experimental results show that training with such synthetic depth maps substantially helps improve the result of direct regression.

3D hand pose regression takes an RGB image and an inferred depth map as input and outputs joint locations in two steps. In the first step, we adopt a popular variant of the CPM architecture [10, 88] as the 2D joint location predictor. This predictor consists of six stages. Each stage contains seven convolutional layers followed by a Rectified Linear Unit (ReLu). It predicts K heatmaps $\{H_s^k\}_{k=1}^K$ for K different hand joints. The pixel value in k^{th} heatmap at stage s, H_s^k, indicates the confidence that the k^{th} joint is located at this position. Following the convention [88], the ground-truth heatmap is denoted as $\{H_*^k\}_{k=1}^K$. Each H_*^k is the Gaussian blur of the Dirac-δ distribution centered at the ground-truth location of k^{th} joint. We train this part of Hand Pose module by standard backpropogation and the mean square error (MSE) loss. In addition to the MSE loss, we add the intermediate supervision for each stage. The final loss for 2D location prediction is

$$\mathcal{L}_{2D} = \frac{1}{6K} \sum_{s=1}^{6} \sum_{k=1}^{K} ||H_s^k - H_*^k||_F^2. \tag{3.3}$$

In the second step, the regression network takes the heatmap from CPM as input, and outputs the relative depth. Its architecture is a mini-CPM (one stage instead of six) followed by three fully connected layers. $Z \in \mathbb{R}^{K\times 1}$ denotes the relative depth of each hand joint. We employ smooth L1 loss between Z and the ground-truth Z^*. The loss of depth regression $\mathcal{L}_z$

is summarized as follows:

$$\mathcal{L}_z = \frac{1}{K}\sum_{k=1}^{K}\begin{cases}\frac{1}{2}(Z_k - Z_k^*)^2, \text{ if } |Z_k - Z_k^*| \le 0.5 \\ |Z_k - Z_k^*|, \text{ otherwise.}\end{cases} \tag{3.4}$$

Depth Regularizer

To provide supervision on every pixel on a depth map, we employ the depth regularizer (DR) proposed in [8]. The depth regularizer takes the relative depth as input and predicts a relative depth map D. It reshapes $Z \in \mathbb{R}^{K\times 1}$ to a $K \times 1 \times 1$ tensor, which is considered as a K-channel image input. We then up-sample this image from K-channel with resolution 1×1 to 1-channel with the original depth map resolution ($n \times m$) through the 6 layers of transposed CNN.

We take L1 norm between D and the ground-truth relative depth map D^* as depth regularizer loss $\mathcal{L}_{dep}$, i.e.,

$$\mathcal{L}_{dep} = ||D - D^*||, \tag{3.5}$$

where D^* is obtained by input depth map $\hat{D}^*$ as follows

$$D^* = \frac{\hat{D}^* - \min \hat{D}^*}{\max \hat{D}^* - \min \hat{D}^*}. \tag{3.6}$$

Note that, we only use the ground-truth depth map $\hat{D}^*$ during the initialization stage. It would be replaced by DGGAN-generated depth maps once the initialization stage ends.

Combining the loss terms described in Section 3.4.3 and Section 3.4.3, we summarize the

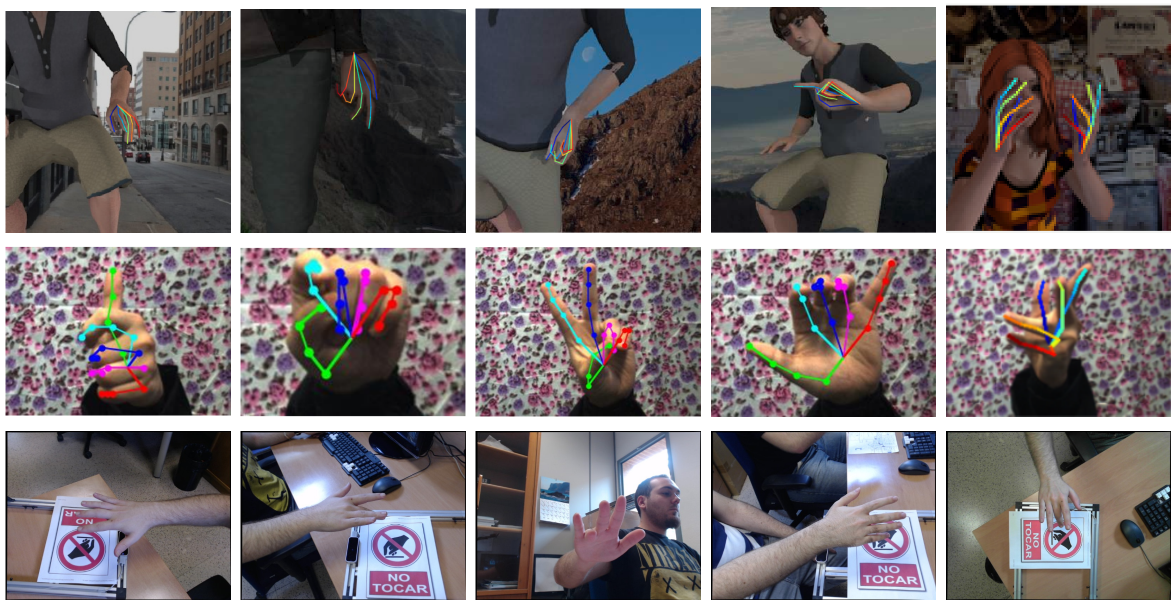


Figure 3.5: Some examples of the three benchmark datasets used for evaluation. **Top Row:** The RHD dataset [105] provides synthetic hand images with 3D hand keypoint annotations. **Middle Row:** The STB dataset [100] contains real hand images with 3D keypoints. **Bottom Row:** The MHP [31] offers real hand images with 3D keypoints.

loss function for the hand pose estimation module as

$$\mathcal{L}_{task} = \lambda_z * \mathcal{L}_z + \lambda_{2D} * \mathcal{L}_{2D} + \lambda_{dep} * \mathcal{L}_{dep}, \tag{3.7}$$

where λ_z, λ_{2D}, λ_{dep} control the importance of three different loss terms, respectively.

3.5 Experimental Settings

This section introduces our experimental settings. The selected benchmark datasets for performance evaluation are first given. The evaluation metric and training details are then presented.

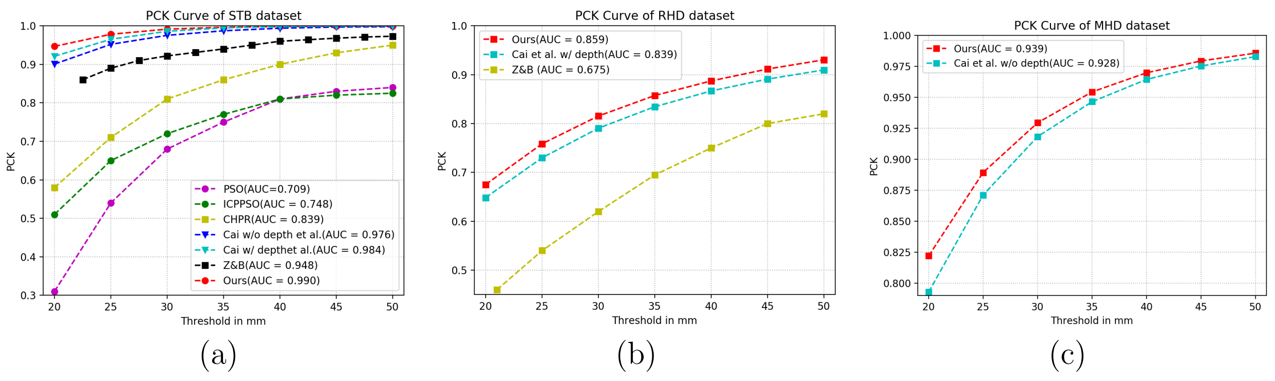


Figure 3.6: Comparisons with the state-of-the-art approaches on the (a) STB, (b) RHD, and (c) MHP datasets for 3D hand pose estimation.

3.5.1 Datasets for Evaluation

We conduct the experiments on three benchmark datasets, including the stereo hand tracking benchmark (STB) [100], the render hand pose dataset (RHD) [105], and the multi-view hand pose (MHP) dataset[31].

The STB dataset is a dataset of real hands. It contains two different subsets called SK and BB. The images in SK are captured by Point Grey Bumblebee2 stereo camera while images in BB are from a depth sensor. In our experiments, we use the BB subset for DGGAN training, and leverage the SK subset for unpaired testing.

RHD is a synthetic dataset. Zhang *et al* [100] use a 3D simulator, Maya, to render the images from 20 different characters doing 39 actions. Each data entry consists of an RGB image and the corresponding depth image, and both 2D/3D annotations. This dataset is challenging since its images are captured with various view points and of many different hand shapes.

The MHP dataset provides color hand images as well as the bounding boxes of hands and the 2D and 3D location of each joint. It consists of hand imaegs of 21 people with different hand movements. For each frame, it provides the images from four different angles of view. The 2D and 3D annotations are obtained by Leap Motion Controller.

Before training, we first crop the hand regions from the original canvas to make sure that hand parts have dominating proportion in the frame. Notice that the STB and MHP datasets use the center of a palm rather than a wrist as one of its hand keypoints. Hence, we revise the annotation to move the center of the palm to the wrist in the same way performed in [8].

3.5.2 Evaluation Metric

Following the previous works [8, 15, 105], we evaluate the results of hand pose estimation by using 1) the *area under the curve* (AUC) on *percentage of correct keypoints* (PCK) between threshold 20mm and 50mm (AUC 20_50) and 2) the *end-point-error* (EPE): the distance between predicted 3D joint locations and the ground truth. In Table 3.1, we report the AUC 20_50 as well as the mean and the median of EPE over all hand keypoints.

3.5.3 Training

During training, we first initialize the weights of the depth-map reconstruction and hand pose estimation modules in the proposed DGGAN. Both modules are initialized by fitting the STB dataset (see 3.5.1) but trained separately. Then, we connect the two modules and fine-tune the whole network in an end-to-end manner. For training with the RHD and STB dataset, the discriminator is derived to distinguish the $G_{S \to T}(x_s)$ and x_t, a randomly chosen depth-map from the respective dataset. For the MHP dataset, we simply randomly assign a depth-map from RHD dataset as x_t because the MHP dataset does not contain any dense depth maps.

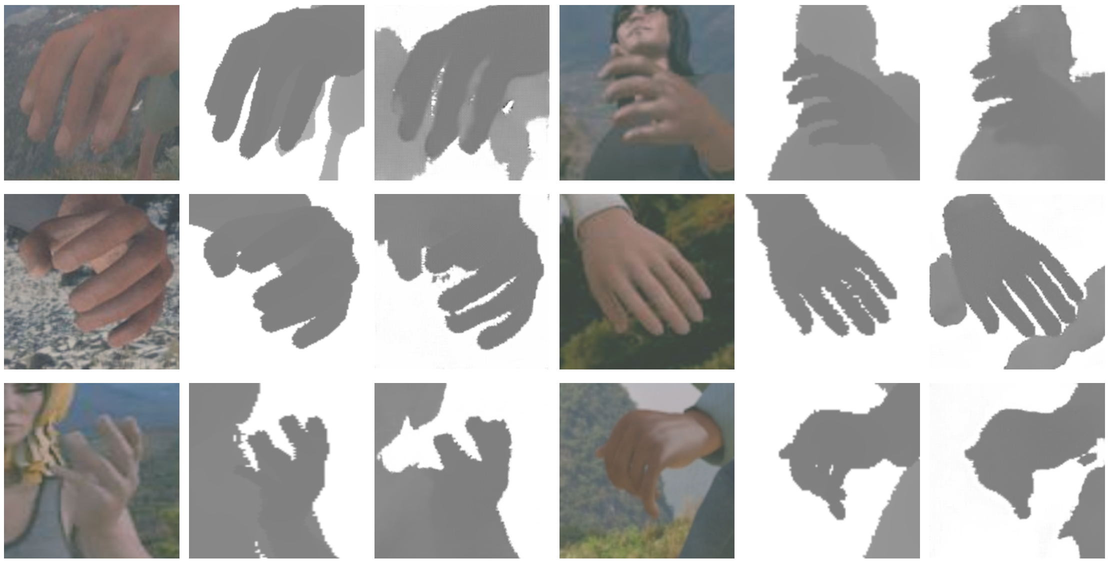


Figure 3.7: Comparison between the generated and ground-truth depth maps on the RHD dataset. The first and fourth columns show the RGB images. The second and fifth columns display the real depth maps. The third and sixth columns give the generated depth maps.

3.6 Experimental Results

For evaluation on the STB dataset, we choose PSO [48], ICPPSO [68], and CHPR [79] as the baselines. In addition, we select the state-of-the-art approaches, Z&B [105] and that by Cai *et al* [8] for comparison.

On the RHD dataset, we compare our method with Z&B [105] and that in [8]. Also, on the MHP dataset, we compare our method to that in [8]. Note that Cai *et al* [8] have not released their code yet. We re-implement their method and report the results according to our implementation.

3.6.1 Ablation Study

For analyzing the effectiveness of the proposed DGGAN, we conduct ablation studies for DGGAN on three different datasets. The detailed results are summarized in Table 3.1.

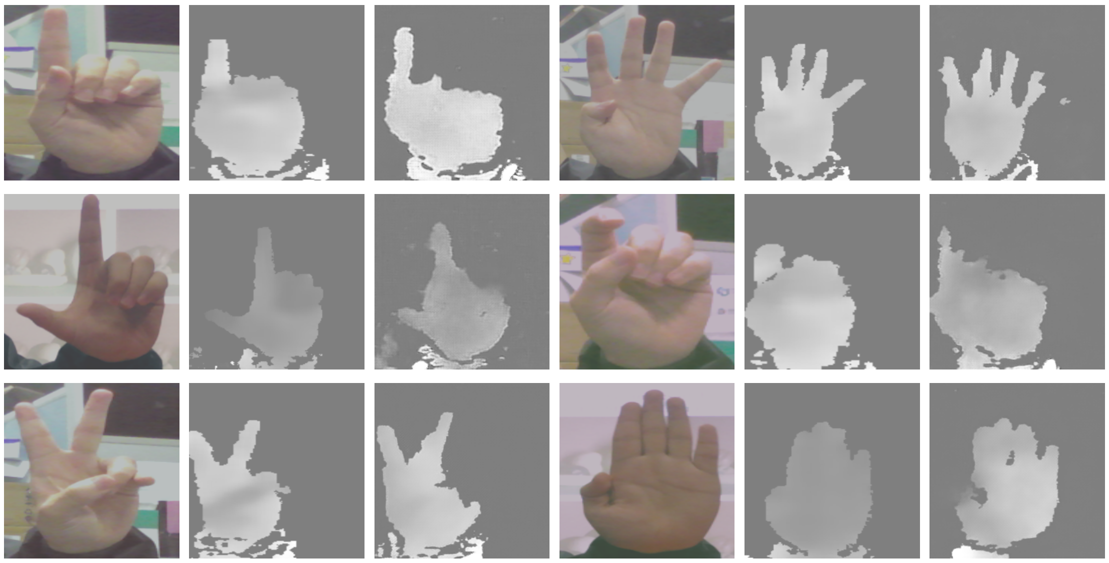


Figure 3.8: Comparison between the generated and ground-truth depth maps on the STB dataset. The first and fourth columns show the RGB images. The second and fifth columns display the real depth maps. The third and sixth columns give the generated depth maps.

Specifically, we conduct the experiments for the following three different settings:

1. Regression: It represents training the regression network only on RGB images and without any depth regularizer.
2. Regression + DR + DGGAN: We learne the depth-regularized regression network using RGB images with the depth maps generated by DGGAN.
3. Regression + DR + true depth map: We derive the depth-regularized regression network using RGB images with their paired true depth maps.

To measure the effectiveness of the generated depth maps, we compare settings Regression and Regression + DR + DGGAN. As illustrated in Table 3.1, using the generated depth map significantly boosts the performance of the model in Regression. The AUC 20_50 is improved by **0.043**, **0.024**, **0.011** on the RHD, STB, and MHP datasets, respectively. The EPE mean is also considerable reduced by **13.2%** and **19.7%** and **7.3%** on the RHD, STB and MHP datasets respectively.

Table 3.1: 3D pose estimation results on the RHD, STB, MHP datasets. ↑: higher is better. ↓: lower is better. *Regression* is the previous State-of-the-art without using paired depth maps.

	AUC 20-50 ↑	EPE mean (mm) ↓	EPE median (mm) ↓
RHD Dataset			
Regression	0.816	21.5	13.96
Regression + DR + DGGAN	**0.839**	**19.0**	**13.17**
Regression + DR + true depth map	0.859	18.0	13.16
STB Dataset			
Regression	0.976	10.91	9.11
Regression + DR + DGGAN	**0.990**	**9.11**	**7.70**
Regression + DR + true depth map	0.984	10.05	8.44
MHP Dataset			
Regression	0.928	14.08	10.75
Regression + DGGAN	**0.939**	**13.12**	**9.91**

Table 3.2: EPE mean comparison on the STB dataset between our approach and the method by Boukhayma *et al* [6]

Method	EPE mean (mm) ↓
Regression + DR + DGGAN (Ours)	**9.11**
Boukhayma *et al*[6]	9.76

To compare the generated depth map with the real depth maps, we conduct two more experiments. Comparing results of Regression + DR + true depth map and Regression + DR + DGGAN shows that the generated depth maps are a key factor of performance boosting. On the RHD dataset, training with the generated depth maps is only slightly worse than the true RHD depth maps by 0.02 in AUC 20_50 and 1 mm in EPE mean. However, on the STB dataset, the results of training with generated depth maps are even better than training with the real depth maps (by 0.006 in AUC 20_50 and 0.94 mms in EPE mean). This result is probable due to the fact that the depth maps collected from depth sensors are less stable and noisier than the depth maps collected from a 3D simulator. By training the DGGAN with unpaired high-quality depth maps from RHD, our generator can potentially reduce the noise, and further benefit the training in the hand pose estimation module. It is worth noting that Regression + DR + true depth map requires the paired depth and RGB image.

In addition to the quantitative analysis, Figure 3.7 and Figure 3.8 provide some examples for visual comparison between the generated and true depth maps on the RHD and STB datasets, respectively. We can see that the generated depth maps are visually very similar to the ground-truth ones.

3.6.2 Comparison with State-of-the-arts

We select the state-of-the-art approaches [6, 8, 65, 76, 100, 59, 105] for comparison. The comparison results are reported in Figure 3.6 and Table 3.2. As shown in Figure 3.6 and Table 3.2, our approach outperforms all existing state-of-the-art methods. Although the results of the method by Cai *et al* [8] come close to ours, we emphasize that our DGGAN has an crucial advantage of *not* requiring any paired RGB and depth images.

3.7 Conclusion

The lack of large-scale datasets of paired RGB and depth images is one of the major bottlenecks for improving 3D hand pose estimation. To address this limitation, we propose a conditional GAN-based model called DGGAN to bridge the gap between RGB images and depth maps. DGGAN synthesizes depth maps from RGB images to regularize the 3D hand pose prediction model during training, eliminating the need of paired RGB images and depth maps conventionally used to train such models.

The proposed DGGAN is integrated into a 3D hand pose prediction framework, and is trained end-to-end together for 3D pose estimation. DGGAN not only generates more realistic hand depth images, which can be used in many other applications such as 3D shape estimation but also results in significant improvement in 3D hand pose estimation, achieving new state-of-the-art results.

Chapter 4

Temporal-Aware Self-Supervised Learning for 3D Hand Pose and Mesh Estimation in Videos

4.1 Introduction

3D hand estimation is an important research topic in computer vision due to a wide range of potential applications, such as sign language translation [98], robotics [2], movement disorder detection and monitoring, and human-computer interaction (HCI) [54, 37].

Depth sensors and RGB cameras are popular devices for collecting hand data. However, depth sensors are not as widely available as RGB cameras and are much more expensive, which has limited the applicability of hand pose estimation methods developed upon depth images. Recent research interests have shifted toward estimating 3D hand poses directly from RGB images by utilizing color, texture, and shape information contained in RGB images. Some methods carried out 3D hand pose estimation from monocular RGB images [8, 40, 105].

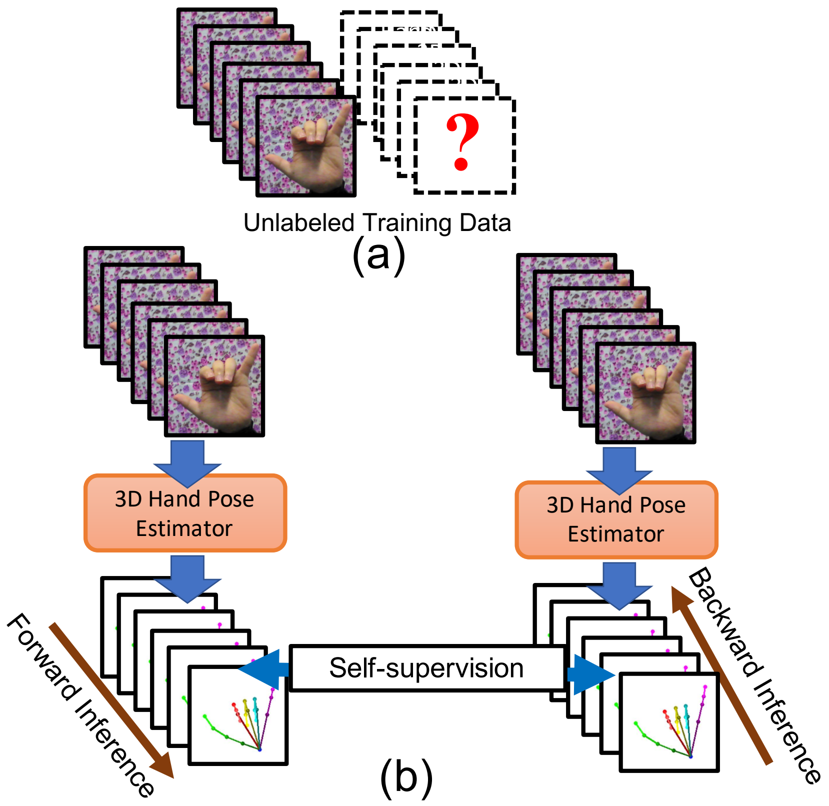


Figure 4.1: Motivation and idea: (a) Training a robust 3D hand pose estimator from RGB images relies on plenty images with 3D hand pose annotations, but obtaining 3D annotations from 2D images is quite difficult; (b) We leverage bi-directional temporal consistency in videos and enable hand pose estimators to make more plausible predictions. It turns out that the hand pose estimator can be derived in a self-supervised fashion without using 3D annotations.

More recently, progresses have been made on estimating 3D hand shape and mesh from RGB images [5, 6, 28, 101, 90, 13]. Compared to poses, hand meshes provide richer information required by many immersive VR and AR applications. Despite the advances, 3D hand pose estimation remains a challenging problem due to the lack of accurate, large-scale 3D pose annotations.

In this work, we develop a new approach to 3D hand pose and mesh estimation by taking the following two observations into account. First, most existing methods rely on training data with 3D information, but capturing 3D information from 2D images is intrinsically difficult. Although there are a few datasets providing annotated 3D hand joints, the amount is too small to train a robust hand pose estimator. Second, most studies focus on hand pose estimation from a single image. Nevertheless, important applications based on 3D hand poses, such as augmented reality (AR), virtual reality (VR), and sign language recognition, are usually carried out in videos.

According to the two observations, our approach exploits video temporal consistency to address the uncertainty caused by the lack of 3D joint annotations on training data. Specifically, our approach, called *temporal-aware self-supervised network (TASSN)*, can learn and infer 3D hand poses without using annotated 3D training data. Figure 4.1 shows the motivation and core idea of the proposed TASSN. TASSN explores video information by embedding a temporal structure to extract spatial-temporal features. We design a novel temporal self-consistency loss, which helps training the hand pose estimator without requiring annotated 3D training data. In addition to poses, we estimate hand meshes since meshes bring salient evidences for pose inference. With meshes, we can infer silhouettes to further regularize our model. The main contributions of this work are given below:

1. We develop a temporal consistency loss and a reversed temporal information technique for extracting spatio-temporal features. To the best of our knowledge, this work makes the first attempt to estimate 3D hand poses and meshes without using 3D annotations.
2. An end-to-end trainable framework, named temporal-aware self-supervised networks (TASSN), is proposed to learn an estimator without using annotated 3D training data. The learned estimator can jointly infer the 3D hand poses and meshes from video.
3. Our model achieves high accuracy with 3D prediction performance on par with state-of-the-art models trained with 3D ground truth.

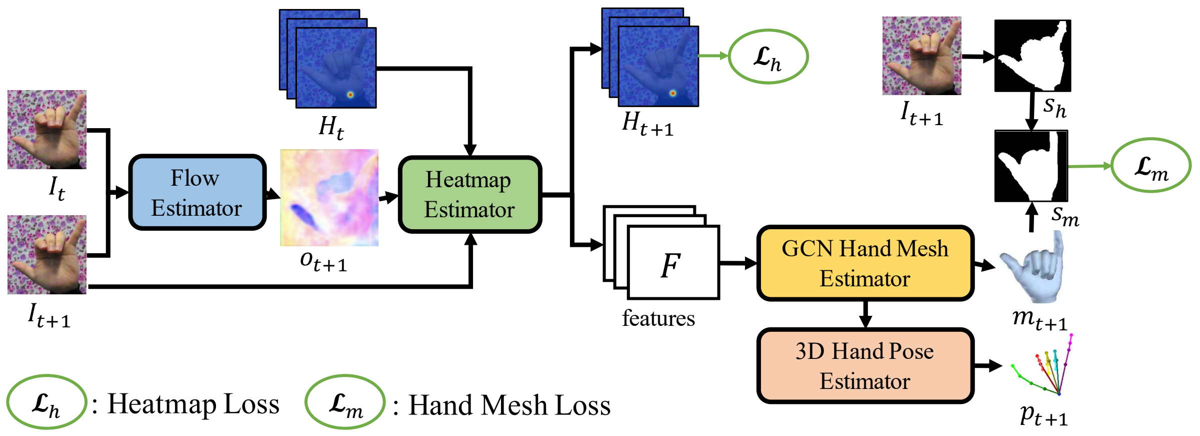


Figure 4.2: Network architecture of the pose and mesh estimation (PME) module. PME module consists of four sub-modules, including the flow, 2D keypoint heatmap, 3D hand mesh, and 3D hand pose estimators. The flow estimator computes the optical flow $\boldsymbol{o}_{t+1}$ from two consecutive frames $\boldsymbol{I}_t$ and $\boldsymbol{I}_{t+1}$. With $\boldsymbol{I}_{t+1}$, $\boldsymbol{o}_{t+1}$, and $\boldsymbol{H}_t$, the 2D heatmap estimator computes the keypoint heatmap $\boldsymbol{H}_{t+1}$ at timestamp $t+1$, as well as extract the image features. Based on the extracted image features, the 3D hand pose and mesh estimators predict the 3D hand pose $\boldsymbol{p}_{t+1}$ and mesh $\boldsymbol{m}_{t+1}$ at timestamp $t+1$. Two loss terms, the heatmap loss $\boldsymbol{\mathcal{L}}_h$ and the hand mesh loss $\boldsymbol{\mathcal{L}}_m$, are used for optimization.

4.2 Related Work

4.2.1 3D Hand Pose Estimation from Depth Images

Since depth images contain surface geometry information of hands, they are widely used for hand pose estimation in the literature [86, 96, 19, 89, 25, 29, 53, 15]. Most existing work adopts regression to fit the parameters of a deformed hand model [55, 46, 49, 86]. Recent work [25, 29] extracts depth image features and regress the joints through PointNet [67]. Wu *et al* [89] leverage the depth image as the intermediate guidance and conduct an end-to-end training framework. Despite the effectiveness, the aforementioned methods highly rely on accurate depth maps, and are less practical in the daily life since depth sensors are not available in many cases due to the high cost.

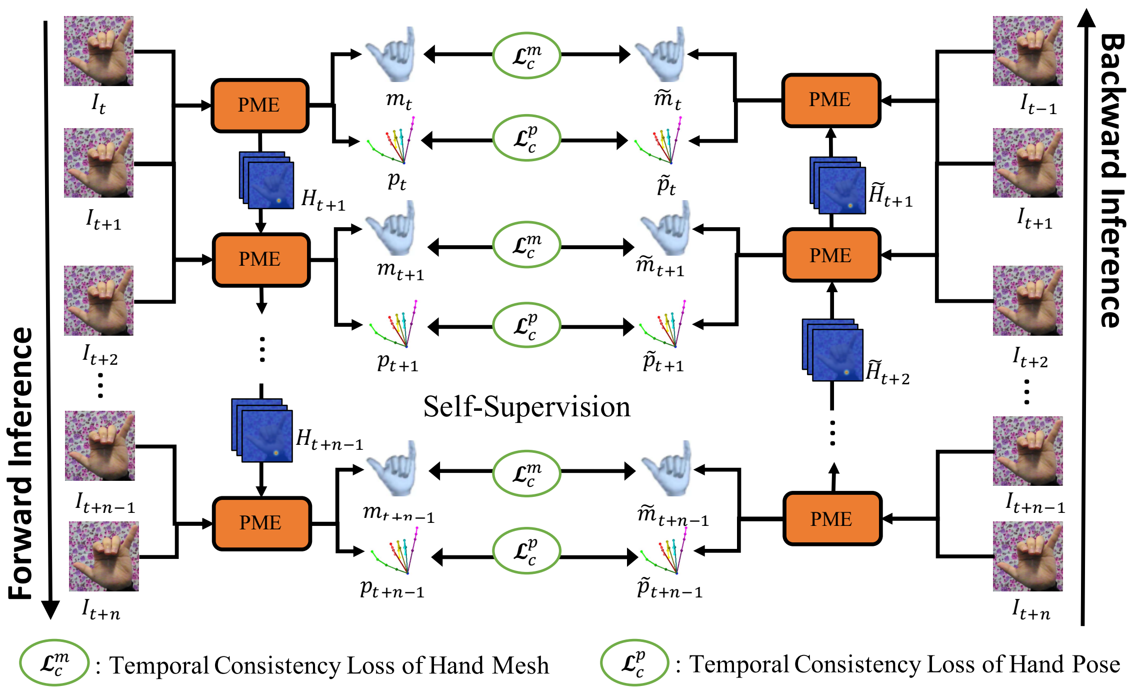


Figure 4.3: Overview of the proposed TASSN. TASSN involves both forward and backward inference to utilize temporal information. Namely, the hand poses estimated by forward and backward inference should be consistent. Our hand pose estimator leverages this observation and can be trained by using self-supervised learning without the need of 3D hand joint labels. Moreover, with the constraints of temporal consistency, either forward or backward inference can gain more accurate hand pose estimation results. In the network, the pose and mesh estimation (PME) module is developed to infer 3D hand poses, meshes, and 2D hand keypoint heatmaps on two consecutive frames.

4.2.2 3D Hand Pose Estimation from RGB Images

Owing to the wide accessibility of RGB cameras, estimating 3D hand poses from monocular images becomes an active research topic [8, 40, 59, 82, 93, 105] and significant improvement has been witnessed. These methods use convolutional neural networks (CNN) to extract features from RGB images. Zimmermann and Brox [105] feed these features to the 3D lift network and camera parameter estimation network for depth regression. Building on Zimmermann and Brox's work, Iqbal *et al*[40] add depth maps as intermediate guidance while Cai *et al* [8] propose a weakly supervised approach to utilize depth maps for regularization. However, these methods suffer from limited training data since 3D hand annotations are hard to acquired. Also, they all dismiss the temporal information.

4.2.3 3D Hand Mesh Estimation

3D hand mesh estimation is an active research topic [28, 6, 5, 45, 101]. Methods in [6, 5, 101] estimate hand meshes by using a pre-defined hand model, named MANO [70]. Due to the high degree of freedom of hand gestures, hand meshes lie in a high dimensional space. The MANO model serves as a kinematic and shape prior of meshes and can help reduce the dimension. However, since MANO is a linear model, it is not able to capture the nonlinear transformation for hand meshes [28]. Thus, mesh estimators based on MANO suffer from this issue. On the other hand, Ge *et al* [28] regress 3D mesh vertices through graphical convolutional neural network (GCN) with down-sampling and up-sampling. Their work achieves the state-of-the-art performance, but it is trained on a dataset with 3D mesh ground truth which is even more difficult to label than 3D joint annotations. This drawback limits its applicability in practice.

4.2.4 Self-supervised Learning

Self-supervised learning [20, 66, 21] is a type of training methodologies, where training data are automatically labeled by exploiting existing information within the data. With this training scheme, manual annotations are not required for a given training set. This scheme is especially beneficial when data labeling is difficult or the data size is exceedingly large. Self-supervised learning has been applied to hand pose estimation. Similar to ours, the method in [21] adopts temporal cycle consistency for self-supervised learning. However, this method uses soft nearest neighbors to solve the video alignment problem, which is not applicable to 3D pose and mesh estimation. Simon *et al* [74] adopt multi-view supervisory signals to regress 3D hand joint locations. While their approach resolves the hand self-occlusion issue using multi-view images, it in the training stage requires 3D joint annotations, which are difficult and expensive to get in this task. Another attempt of using self-supervised learning for hand pose estimation is presented in [85], where an approach leveraging a massive amount of unlabeled depth images is proposed. However, this approach may be limited due to the high variations of depth maps in diverse poses, scales, and sensing devices. Instead of leveraging multi-view consistency or depth consistency, the proposed self-supervised scheme relies on temporal consistency, which is inexpensive to get and does not require 3D keypoint annotations.

4.3 Proposed Method

We aim to train a 3D hand pose estimator from videos without 3D hand joint labels. To tackle the absence of 3D annotations, we adopt the temporal information from hand motion videos, and address the ambiguity caused by the lack of 3D joint ground truth. Specifically, we present a novel deep neural network, named temporal-aware self-supervised networks (TASSN). By developing the temporal consistency loss on the estimated hand gestures in a video, TASSN can learn and infer 3D hand poses through self-supervised learning without using any 3D annotations.

4.3.1 Overview

Given an RGB hand motion video $\boldsymbol{x}$ with N frames, $\boldsymbol{x} = \{\boldsymbol{I}_1, ..., \boldsymbol{I}_N\}$, we aim at estimating 3D hand poses in this video, where $\boldsymbol{I}_t \in \mathbb{R}^{3\times W\times H}$ is the t-th frame, and W and H are the frame width and height, respectively. The 3D hand pose at frame t, $\boldsymbol{p}_t \in \mathbb{R}^{3\times K}$, is represented by a set of K 3D keypoint coordinates of the hand. Figure 4.3 illustrates the network architecture of TASSN.

Leveraging the temporal consistency properties of videos, the hand poses and meshes predicted in the forward and backward inference orders can perform mutual supervision. Our model can be fine-tuned on any target dataset using this self-supervised learning and the temporal consistency is a good substitute for the hard-to-get 3D ground truth. TASSN alleviates the burden of annotating 3D ground-truth of a dataset without significantly sacrificing model performance.

Recent studies [28, 101] show that training pose estimators with hand meshes improves the performance because hand meshes can act as intermediate guidance for hand pose prediction. To this end, we propose a hand pose and mesh estimation (PME) module, which jointly estimates the 2D hand keypoint heatmaps, 3D hand poses and meshes from every two adjacent frames $\boldsymbol{I}_i$ and $\boldsymbol{I}_{i+1}$.

4.3.2 Pose and Mesh Estimation Module

The proposed PME module consists of four estimator sub-modules, including flow estimator, 2D keypoint heatmap estimator, 3D hand mesh estimator, and 3D hand pose estimator. Given two consecutive frames as input, it estimates the 3D hand pose and mesh. Figure 4.2 shows its network architecture.

Flow Estimator: To capture temporal clues from a hand gesture video, we adopt FlowNet [38] to estimate the optical flow $\boldsymbol{o}_{t+1} \in \mathbb{R}^{2\times W\times H}$ between two consecutive frames $\boldsymbol{I}_t$ and $\boldsymbol{I}_{t+1}$. In forward inference, FlowNet computes $\boldsymbol{o}_{t+1}$, the motion from frame $\boldsymbol{I}_t$ to frame $\boldsymbol{I}_{t+1}$. In backward inference, FlowNet computes the reverse motion.

Heatmap Estimator: Our heatmap estimator computes 2D hand keypoints and generates the features for the 3D hand pose and mesh estimators. The estimated 2D keypoint heatmaps are denoted by $\boldsymbol{H} \in \mathbb{R}^{K\times W\times H}$, where K represents the number of keypoints. We adopt a two stacked hourglass network [61] to infer the hand keypoint heatmaps $\boldsymbol{H}$ and compute the features $\boldsymbol{F}$. We concatenate $\boldsymbol{I}_{t+1}$, $\boldsymbol{o}_{t+1}$, and $\boldsymbol{H}_t$ as input to the stacked hourglass network, which produces heatmaps $\boldsymbol{H}_{t+1}$, as shown in Figure 4.2. The estimated $\boldsymbol{H}_{t+1}$ includes K heatmaps $\{\boldsymbol{H}^k_{t+1} \in \mathbb{R}^{W\times H}\}^K_{k=1}$, where $\boldsymbol{H}^k_{t+1}$ expresses the confidence map of the location of the kth keypoint. The ground truth heatmap $\bar{\boldsymbol{H}}^k_{t+1}$ is the Gaussian blur of the Dirac-δ distribution centered at the ground truth location of the kth keypoint. The heatmap loss $\boldsymbol{\mathcal{L}}_h$ at frame t is defined by

$$\boldsymbol{\mathcal{L}}_h = \frac{1}{K}\sum_{k=1}^{K} ||\boldsymbol{H}^k_t - \bar{\boldsymbol{H}}^k_t||^2_F. \tag{4.1}$$

3D Hand Mesh Estimator: Our 3D hand mesh estimator is developed based on Chebyshev spectral graph convolution network (GCN) [28], and it takes hand features $\boldsymbol{F}$ as input and infers the 3D hand mesh. The output hand mesh $\boldsymbol{m}_t \in \mathbb{R}^{3\times C}$ is represented by a set of 3D mesh vertices, where C is the number of vertices in a hand mesh.

To model hand mesh, we use an undirected graph $\boldsymbol{G}(\boldsymbol{V}, \boldsymbol{E})$, where $\boldsymbol{V}$ and $\boldsymbol{E}$ are the vertex and edge sets, respectively. The edge set $\boldsymbol{E}$ can be represented by an adjacent matrix $\boldsymbol{A}$, where $\boldsymbol{A}_{i,j} = 1$ if edge $e(i,j) \in \boldsymbol{E}$, otherwise $\boldsymbol{A}_{i,j} = 0$. The normalized Laplacian normal matrix of $\boldsymbol{G}$ is obtained via $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{A}\boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{D}$ is the degree matrix and $\boldsymbol{I}$ is the identity matrix. Since $\boldsymbol{L}$ is a positive semi-definite matrix [7], it can be decomposed as $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$, where $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, ..., \lambda_C)$, and C is the number of vertices in $\boldsymbol{G}$.

We follow the setting in [18], and set the convolution kernel to $\hat{\boldsymbol{\Lambda}} = diag(\sum_{i=0}^{S}\alpha_i\lambda_1^i, ..., \sum_{i=0}^{S}\alpha_i\lambda_C^i)$, where α is the kernel parameter. The convolutional operations in $\boldsymbol{G}$ can be calculated by $\boldsymbol{F'} = \boldsymbol{U}\hat{\boldsymbol{\Lambda}}\boldsymbol{U}^T\boldsymbol{F}\boldsymbol{\theta}_i = \sum_{i=0}^{S}\alpha_i\boldsymbol{L}^i\boldsymbol{F}\boldsymbol{\theta}_i$, where $\boldsymbol{F} \in \mathbb{R}^{N\times F_{\text{in}}}$ and $\boldsymbol{F'} \in \mathbb{R}^{N\times F_{\text{out}}}$ indicate the input and output features respectively, S is a preset hyperparameter used to control the receptive field, and $\boldsymbol{\theta}_i \in \mathbb{R}^{F_{\text{in}}\times F_{\text{out}}}$ is trainable parameter set used to control the number of output channels.

The Chebyshev polynomial is used to reduce the model complexity by approximating convolution operations, leading to the output features $\boldsymbol{F'} = \sum_{i=0}^{S} \alpha_i T_i(\hat{\boldsymbol{L}})\boldsymbol{\theta}_i$ where $T_k(x)$ is the k-th Chebyshev polynomial and $\hat{\boldsymbol{L}} = 2\boldsymbol{L}/\lambda_{max} - \boldsymbol{I}$ is used to normalize the input features.

We adopt the scheme in [18, 28] to construct the hand mesh in a coarse-to-fine manner. We use the multi-level clustering algorithm for coarsening the graph, and then store the graph at each level and the mapping between graph nodes in every two consecutive levels. In forward inference, the GCN first up-samples the node features according to the stored mappings and graphs and then preforms the graph convolutional operations.

Mesh Silhouette Constraint: In our model, without 3D mesh ground truth, the model tends to collapse to any kind of mesh as long as it is temporally consistent. To avoid this issue, we introduce the mesh loss $\boldsymbol{\mathcal{L}}_m$ to calculate the difference between the silhouette of the predicted hand mesh $\boldsymbol{s}_t$ and the ground-truth silhouette $\bar{\boldsymbol{s}}_{\boldsymbol{t}}$ at frame t. The silhouette loss is defined by

$$\boldsymbol{\mathcal{L}}_m = ||\boldsymbol{s}_t - \bar{\boldsymbol{s}}_t||_F^2. \tag{4.2}$$

To obtain $\bar{\boldsymbol{s}}_t$, we use GrabCut [71] to estimate the hand silhouettes from the training images. Some silhouettes estimated from training images are shown in Figure 4.4. The silhouette of our predicted hand mesh $\boldsymbol{s}_t$ is obtained by using the neural rendering approach in [47].

3D Hand Pose Estimator: The proposed 3D pose estimator directly infers 3D hand keypoints $\boldsymbol{p}_t$ from the predicted hand mesh $\boldsymbol{m}_t$. Taking the mesh as the input, we adopt a network of two stacked GCNs, which has a similar structure to that used in 3D hand mesh estimator. We add a pooling layer to each GCN to extract the pose features from the mesh. Those pose features are then fed to two fully connected layers to regress the 3D hand pose $\boldsymbol{p}_t$.

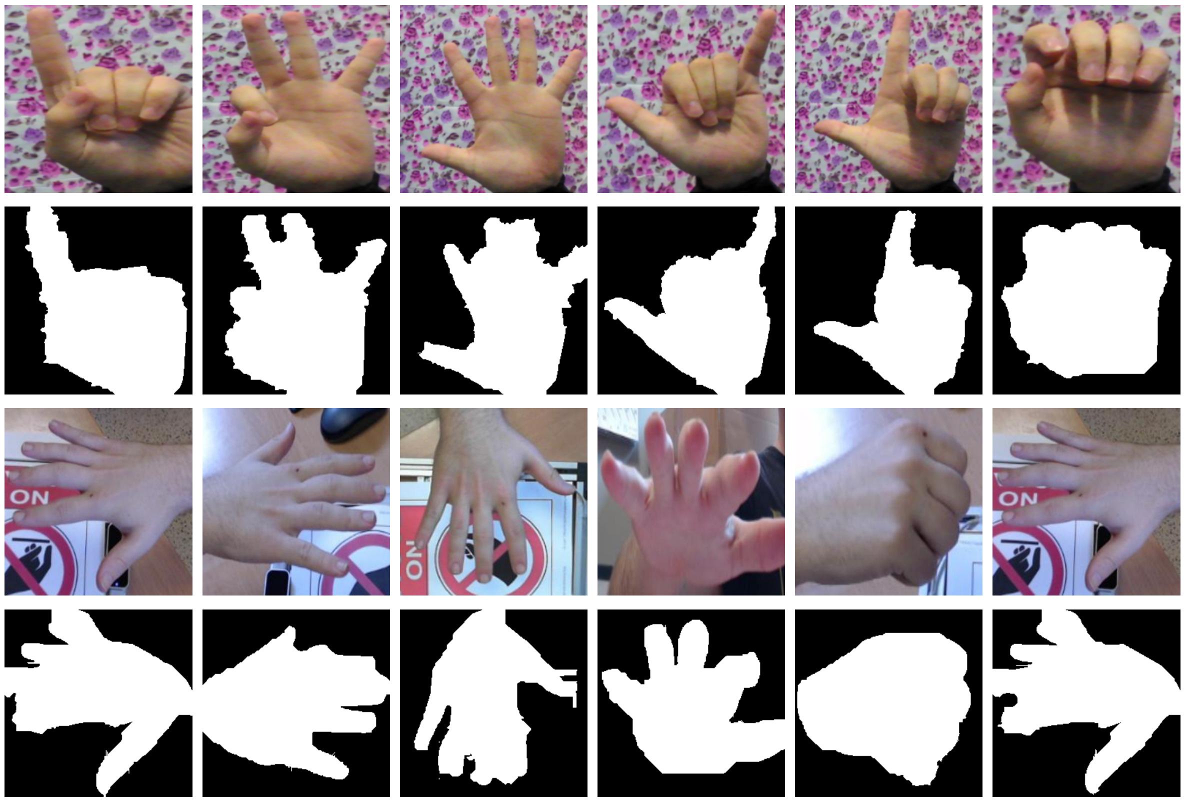


Figure 4.4: Examples of our estimated silhouettes. The first and third rows show the training images in STB and MHP datasets, respectively. The second and the fourth rows show the estimated silhouettes by our method.

4.3.3 Temporal Consistency Loss

Due to the lack of 3D keypoint annotations, conventional supervised learning schemes no longer work in model training. We propose a temporal consistency loss $\boldsymbol{\mathcal{L}}_c$ to solve this problem. Figure 4.3 shows the idea of our approach. Given a video clip with n frames, we feed every two adjacent frames $\{\boldsymbol{I}_i, \boldsymbol{I}_{i+1}\}_{i=t}^{t+n}$ to PME module for hand mesh and pose estimation, *i.e.*, , $\{\boldsymbol{p}_i, \boldsymbol{m}_i\}_{i=t}^{t+n}$. TASSN analyzes the temporal information according to their relative input orders. Thus, we can reverse the input order from $\{\boldsymbol{I}_i, \boldsymbol{I}_{i+1}\}$ to $\{\boldsymbol{I}_{i+1}, \boldsymbol{I}_i\}$ to infer the pose and mesh in $\boldsymbol{I}_i$ from $\boldsymbol{I}_{i+1}$. With this reversed temporal measurement (RTM) technique, we can infer the hand pose and mesh from the reversed temporal order. We denote the estimated pose and mesh in the reversed order as $\{\tilde{\boldsymbol{p}}_i, \tilde{\boldsymbol{m}}_i\}_{i=t}^{t+n}$. As shown in Figure 4.3, the prediction results estimated by the PME module in both forward and backward inference must be consistent with each other since the same mesh and pose are estimated at any frame. The temporal consistency loss on hand pose $\boldsymbol{\mathcal{L}}_c^p$ and mesh $\boldsymbol{\mathcal{L}}_c^m$ can be computed by

$$\boldsymbol{\mathcal{L}}_c^p = \frac{1}{n}\sum_{i=t}^{t+n} ||\boldsymbol{p}_i \ - \tilde{\boldsymbol{p}}_i||_F^2, \tag{4.3}$$

$$\boldsymbol{\mathcal{L}}_c^m = \frac{1}{n}\sum_{i=t}^{t+n} ||\boldsymbol{m}_i \ - \tilde{\boldsymbol{m}}_i||_F^2. \tag{4.4}$$

The temporal consistency loss $\boldsymbol{\mathcal{L}}_c$ is defined as the summation of $\boldsymbol{\mathcal{L}}_c^m$ and $\boldsymbol{\mathcal{L}}_c^p$, *i.e.,* ,

$$\boldsymbol{\mathcal{L}}_c = \lambda^m \boldsymbol{\mathcal{L}}_c^m + \lambda^p \boldsymbol{\mathcal{L}}_c^p, \tag{4.5}$$

where λ^m and λ^p are the weights of the corresponding losses.

4.3.4 TASSN Training

Suppose we are given an unlabeled hand pose dataset $\boldsymbol{X}$ for training, which contains M hand gesture videos, $\boldsymbol{X} = \{\boldsymbol{x}^{(i)}\}_{i=1}^M$, where video $\boldsymbol{x}^{(i)} = \{I_1, ..., I_N\}$ consists of N frames. We divide each training video into several video clips. Each training video clip $\boldsymbol{v}$ is with n frames, *i.e.,* , $\boldsymbol{v} = \{I_t, I_{t+1}, ..., I_{t+n}\}$. With the losses defined in Eq. (4.1), Eq. (4.2), and Eq. (4.5), the objective for training the proposed TASSN is

$$\boldsymbol{\mathcal{L}} = \lambda_s \boldsymbol{\mathcal{L}}_m + \lambda_h \boldsymbol{\mathcal{L}}_h + \boldsymbol{\mathcal{L}}_c, \tag{4.6}$$

where λ^s and λ^h denote the weights of the loss $\boldsymbol{\mathcal{L}}_m$ and the loss $\boldsymbol{\mathcal{L}}_h$, respectively. The details of parameter setting are given in the experiments.

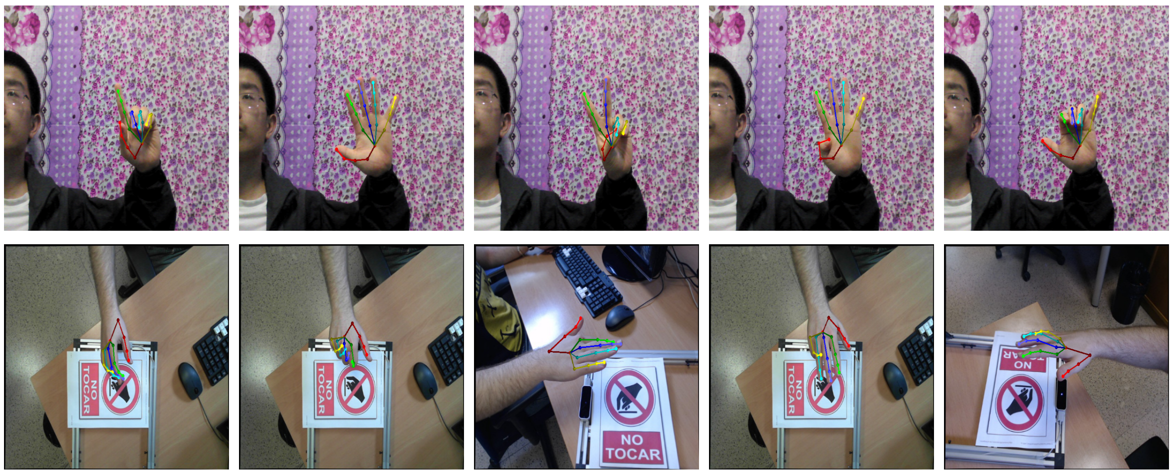


Figure 4.5: Some examples of the two hand pose datasets used for evaluation. The first row shows examples of the STB dataset [100] and the second row gives examples of the MHP dataset [31]. Both datasets include real hand video sequences performed by different subjects and have 3D hand keypoint annotations

4.4 Experiments Setting

4.4.1 Datasets for Evaluation

We evaluate our approach on two hand pose datasets, Stereo Tracking Benchmark Dataset (STB) [100] and Multi-view 3D Hand Pose dataset (MHP) [31]. These two datasets include real hand video sequences performed by different subjects and 3D hand keypoint annotations are provided for the hand video sequences.

For the STB dataset, we adopt its SK subset for training and evaluation. This subset contains 6 hand videos, each of which has $1,500$ frames. Following the train-validation split setting used in [28], we use the first hand video as the validation set and the rest videos for training.

The MHP dataset includes 21 hand motion videos. Each video provides hand color images and different kinds of annotations for each sample, including the bounding box and the 2D

Table 4.1: 3D hand pose estimation results on the STB and MHP datasets. ↑: higher is better; ↓: lower is better; The measuring unit of EPE is millimeter (mm).

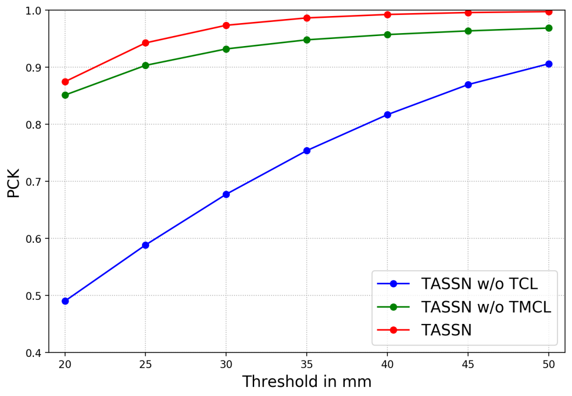
	$\text{AUC}_{0\text{-}50}\uparrow$	$\text{AUC}_{20\text{-}50}\uparrow$	EPE↓
STB Dataset			
TASSN w/o $\boldsymbol{\mathcal{L}}_c$	0.541	0.735	24.2
TASSN w/o $\boldsymbol{\mathcal{L}}_c^m$	0.754	0.936	13.6
TASSN	**0.773**	**0.972**	**11.3**
MHP Dataset			
TASSN w/o $\boldsymbol{\mathcal{L}}_c$	0.492	0.677	28.2
TASSN w/o $\boldsymbol{\mathcal{L}}_c^m$	0.665	0.870	17.5
TASSN	**0.689**	**0.892**	**16.2**

and 3D location on the hand keypoints.

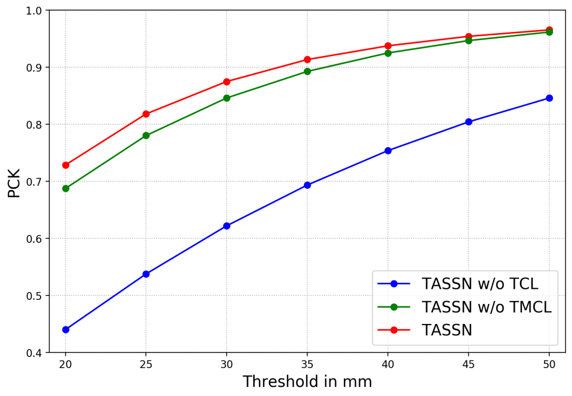
The following scheme of data pre-processing is applied to both STB and MHP datasets. We crop the hand from the original image by using the center of hand and the scale of the hand. Thus, the center of the hand is located at the center of the cropped images, and the cropped image covers the whole hand. We then resize the cropped image to 256×256. As mentioned in [8, 105], the STB and MHP datasets use the palm center as the center of the hand. We use the mechanism introduced by [8] to change the center of hand from the palm center to the joint of wrist.

4.4.2 Metric

We follow the setting adopted in previous work [105, 28] and use *average End-Point-Error* (EPE) and *Area Under the Curve* (AUC) on the *Percentage of Correct Keypoints* (PCK) between threshold 20 millimeter (mm) and 50mm ($\text{AUC}_{20\text{-}50}$) as the two metrics. Beside, we adopt AUC on PCK between threshold 0mm and 50mm ($\text{AUC}_{0\text{-}50}$) as the third metrics for evaluating 3D hand pose estimation performance. The measuring unit of EPE is millimeter (mm).



(a)



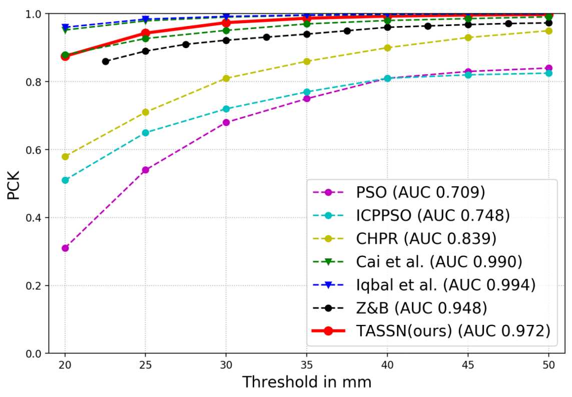
(b)

Figure 4.6: Performance in PCK on the (a) STB and (b) MHP datasets. TCL and TMCL denote the losses $\mathcal{L}_c$ and $\mathcal{L}_c^m$, respectively.

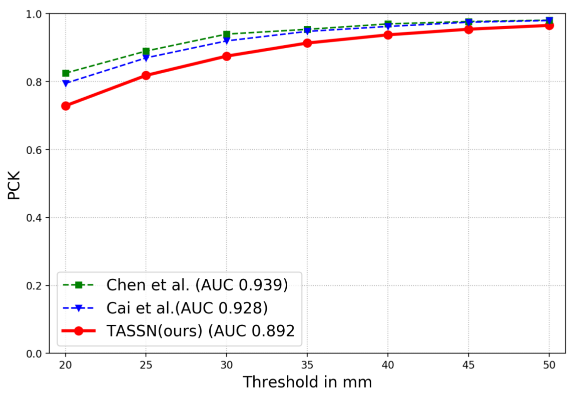
4.4.3 Implementation Details

We implement our TASSN by using PyTorch. In training phase, we set the batch size to 24 and the initial learning rate to 10^{-5}. We train and evaluate our TASSN by using a machine with four GeForce GTX 1080Ti GPUs.

Since end-to-end training a network from scratch with multiple modules is very difficult, we train our TASSN by using a three-stage procedure. In the first stage, we train the heatmap estimator with the loss $\boldsymbol{\mathcal{L}_h}$. In the second stage, the GCN hand mesh estimator is initialized by using the pre-trained model provided by [28]. We jointly fine-tune heatmap and hand mesh estimator with the losses $\boldsymbol{\mathcal{L}_h}$ and $\boldsymbol{\mathcal{L}_m}$ on the target dataset without 3D supervision. In the final stage, we conduct an end-to-end training for our TASSN and fine-tune the weights of each sub-module. The model weights of heatmap, GCN hand mesh estimator, and 3D pose estimators are fine-tuned end-to-end. In this stage, we set $\lambda_s = 0.1$, $\lambda_h = 1$, and $\lambda_c^p = \lambda_c^m = 10$.



(a)



(b)

Figure 4.7: Comparison with the state-of-the-arts. Results in AUC$_{20\text{-}50}$ on (a) the STB dataset and (b) the MHP dataset.

4.5 Experimental Results

4.5.1 Ablation Study of Temporal Consistency Losses

To study the impact of the proposed temporal consistency constraint, we train and evaluate TASSN under the following three settings: 1) TASSN is trained without using temporal consistency loss $\mathcal{L}_c$, i.e., without any temporal consistency constraint; 2) TASSN is trained without using temporal consistency loss of hand mesh $\mathcal{L}_c^m$, i.e., with temporal 3D pose constraint but not 3D mesh constraint; 3) TASSN is trained with all the proposed loss functions.

Table 4.1 shows the evaluation results on two 3D hand pose estimation tasks under the three different settings described above. The PCK curves corresponding to different settings are shown in Figure 4.6.

We note the following two observations from the ablation study. First, the temporal consistency constraint is critical for 3D pose estimation accuracy. This is clearly illustrated by comparing the results between settings 1 and 3. As shown in Figure 4.6, TASSN trained with the temporal consistency loss $\mathcal{L}_c$ (red curve, setting 3) outperforms the TASSN trained

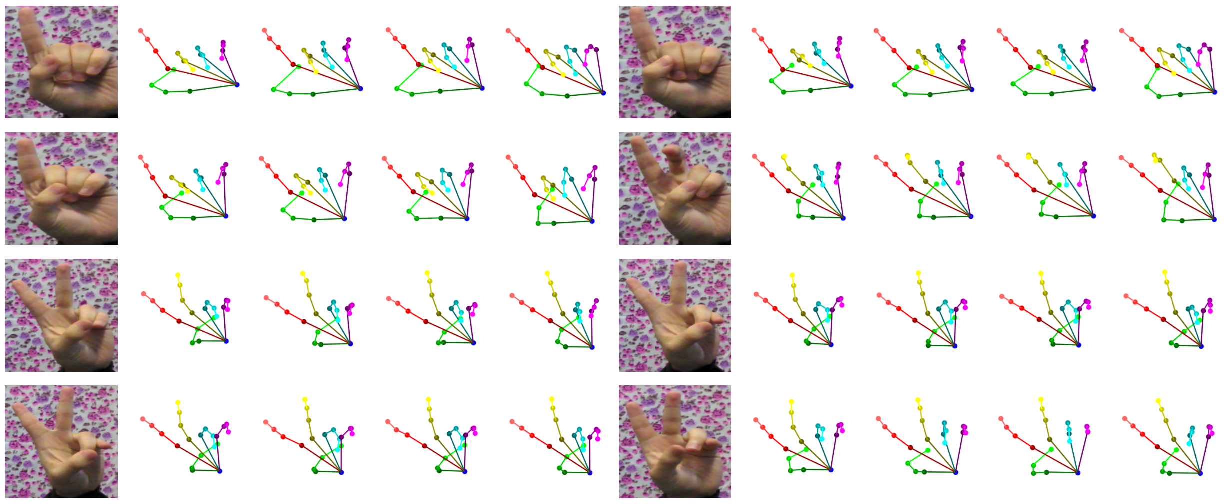


Figure 4.8: Comparison among three different settings on the STB dataset. Columns 1 and 6 are RGB images. Columns 2 and 7 are the result by TASSN trained without temporal consistency loss. Columns 3 and 8 are the result by TASSN trained without temporal mesh consistency loss. Columns 4 and 9 are the result by TASSN. Columns 5 and 10 are the ground truth.

without using temporal consistency loss (blue curve, setting 1) by a large margin on both the STB and MHP datasets. The quantitative results in Table 4.1 show that AUC$_{0\text{-}50}$, AUC$_{20-50}$ and EPE, are improved by 0.232, 0.237, 12.9 on the STB dataset, respectively. A similar trend is also observed on the MHP dataset.

Second, imposing temporal mesh consistency constraints is beneficial for 3D pose estimation. This is illustrated by comparing the results between settings 2 and setting 3. By using the temporal mesh consistency loss $\mathcal{L}_c^m$, AUC$_{0\text{-}50}$, AUC$_{20\text{-}50}$, EPE improves by 0.024, 0.022, 1.3, respectively, on the STB dataset (Table 4.1). Results on MHP dataset share a same trend: Test AUCs are boosted by including the temporal mesh consistency loss $\mathcal{L}_c^m$. It points out that the temporal mesh consistency loss, as an intermediate constraint, facilitates 3D hand pose estimator learning.

In addition to the quantitative analysis, Figure 4.8 and Figure 4.9 display some estimated 3D hand poses for visual comparison among these settings on the STB and MHP datasets,

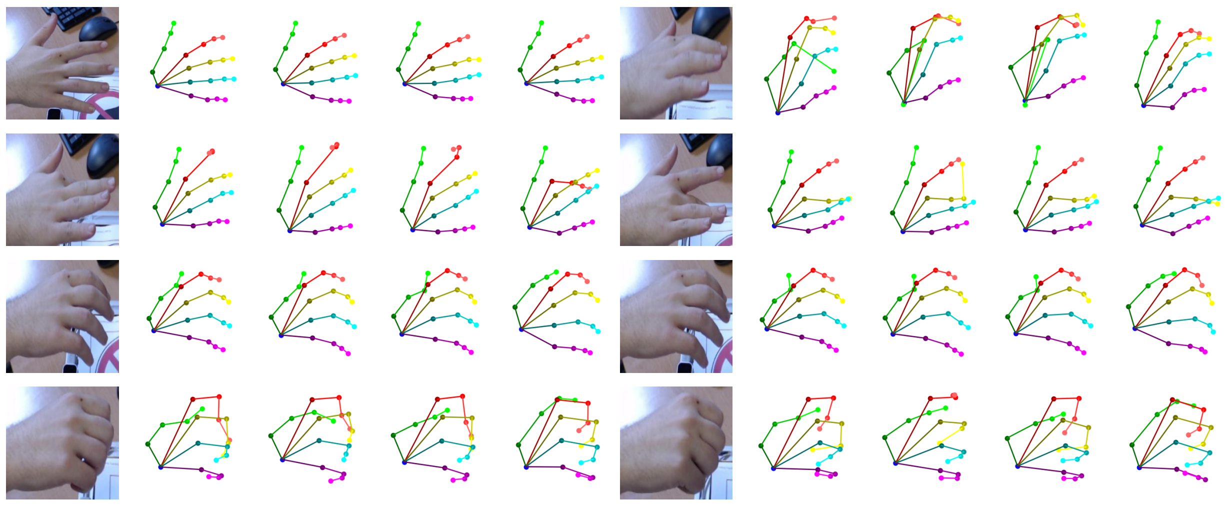


Figure 4.9: Comparison among three different settings on the MHP dataset. Columns 1 and 6 are RGB images. Columns 2 and 7 are the result by TASSN trained without temporal consistency loss. Columns 3 and 8 are the result by TASSN trained without temporal mesh consistency loss. Columns 4 and 9 are the result by TASSN. Columns 5 and 10 are the ground truth.

respectively. We can see that TASSN, when trained with temporal consistency loss, can produce 3D hand pose estimations highly similar to the ground truth in diverse poses. It is worth noting that our GCN model is initialized with model [28] pretrained on the STB dataset. Our results on STB demonstrate that the temporal consistency is critical to enforce the 3D constraints, without which 3D prediction accuracy drops substantially (Table 4.1). Moreover, our method generalizes well on other target datasets, e.g., the MHP dataset, where 3D annotations are not used in either model initialization or training. The pose categories and capturing environments are quite different between the two datasets (Figure 4.5). The effectiveness of our method on the MHP dataset can only be attributed to the temporal consistency constraint (Figure 4.6).

4.5.2 Comparison with the State-of-the-art Methods

The state-of-the-art methods on both STB and MHP datasets are trained with the 3D annotations, while our method is not. Therefore, we take these methods as the upper bound of our method, and evaluate the performance gaps between these methods and ours.

For the STB dataset, we select six the-state-of-the-art methods for comparison. The selected methods include PSO [6], ICPPSO [15], CHPR [100], the method by Iqbal *et al* [40], Cai *et al* [8] and the approach by Zimmermann and Brox [105]. For the MHP dataset, we select two the-state-of-the-art methods for comparison including the approach by Cai *et al* [8] and the method by Chen *et al* [12]. Figure 4.7(a) and Figure 4.7(b) show the comparison results on STB and MHP datasets, respectively. As expected, TASSN has a performance gap with current state-of-the-art methods on both datasets due to the lack of 3D annotation. However, the performance gaps are relative small. In STB dataset, as shown in Figure 4.7(a), our methods could even beat some of the methods trained with full 3D annotations.

All together, these results illustrate that 3D pose estimator can be trained without using 3D annotations. Estimating hand pose and mesh from single frames is challenging due to the ambiguities caused by the missing depth information and high flexibility of joints. These challenges can be partly mitigated by utilizing information from video, in which pose and the mesh are highly constrained by the adjacent frames. Temporal information offers an alternative way of enforcing constraints on 3D models for pose and mesh estimation.

4.6 Conclusions

We propose a video-based hand pose estimation model, temporal-aware self-supervised network (TASSN), to learn and infer 3D hand pose and mesh from RGB videos. By leveraging temporal consistency between forward and reverse measurements, TASSN can be trained through self-supervised learning without explicit 3D annotations. The experimental results show that TASSN achieves reasonably good results with performance comparable to state-of-the-art models trained with 3D ground truth.

The temporal consistency constraint proposed here offers a convenient and yet effective mechanism for training 3D pose prediction models. Although we illustrate the efficacy of the model without using 3D annotations, it can be used in conjunction with direct supervision with a small number of 3D labeled samples to improve accuracy.

Chapter 5

MVHM: A Large-Scale Multi-View Hand Mesh Benchmark for Accurate 3D Hand Pose Estimation

5.1 Introduction

Estimating 3D hand poses from images has attracted increasing attention because it is essential to a wide range of applications such as human-computer interaction (HCI) [3, 54], virtual reality (VR) [17, 37], augmented reality (AR) [11], medical diagnosis [41], and sign language understanding [98]. Although extensive research efforts have been made on this research topic for decades, there are still several unsolved challenges. One of the most crucial challenges is to handle the issue of depth ambiguity present in single view 3D hand pose estimation.

Conventional studies mainly focus on inferring 3D hand poses from either depth or RGB images directly. To address the problems caused by depth ambiguity, Some of the previous

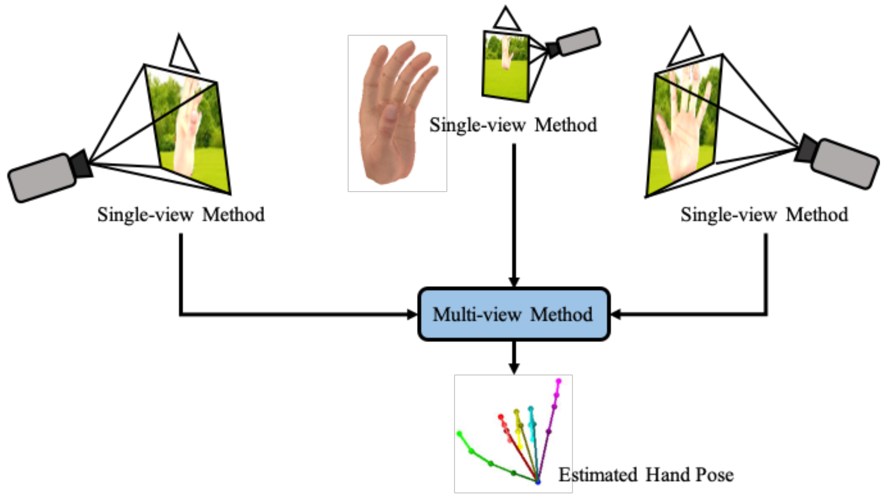


Figure 5.1: Our core idea. We build a synthetic dataset from multi-view perspective, *e.g*, rendering hand images from different a angles. With the aid of this dataset, a single-view method takes the image from each view and generates a possible hand pose candidate. We proposed a multi-view method takes different single-view predictions as input and predicts the final result.

studies [8, 40, 12, 14] want to address this problem by leveraging depthmap information. These works come up with several way to introduce depthmap into the training procedure, such as making depthmap as intermediate supervision [40] or using depth regularizer [8]. On the other hand, recent studies [51, 106] point out that imposing 3D hand shape (mesh) supervision can boost both the performance of 3D hand pose and shape estimators. It is clear that 3D hand shape brings richer hand structure information than hand keypoints. Furthermore, a preset mesh serves as a strong prior to reduce the freedom of the hand, therefore mitigating depth ambiguity. Along this line, several methods such as [6, 5, 101, 52, 92, 90] are proposed. Despite the potential, the aforementioned methods highly rely on a preset hand model learned with a large number of accurate 3D mesh annotations. Hence, a large-scale dataset with accurate annotations of mesh vertices is in great demand.

Accurate mesh ground truth is hard to be manually annotated in general. The hand mesh annotations in most existing datasets are often annotated by hand shape estimator which can be inaccurate, because hand mesh estimation itself is an even more challenging task.

Most existing methods leveraging mesh information for 3D hand pose estimation are derived based on a single view. However, mere mesh information is insufficient to address depth ambiguity. Thus, 3D pose estimation still remains ill-posed in these methods.

The issue of depth ambiguity can be tackled by multi-view vision according to epipolar geometry. Multi-view sensing systems can capture hand images from cameras in different angles and therefore depth information of hands can be accurately inferred as long as camera parameters are known. Inspired by above observations, we aim to build a large-scale multi-view hand mesh dataset that provides hand mesh, multi-view hand images simultaneously for training pose estimators.

In this work, we present an effective mechanism to synthesize 3D hand joint and mesh annotations, and establish a large-scale multi-view hand mesh (MVHM) dataset. We acquired a hand mesh model with rigging system, and a 3D hand groundtruth from existing dataset, and a rigged hand model to match the given groundtruth to perform various gestures. We render the hand model from different angles to collect multi-view image, as well as the 3D keypoints and mesh annotation to built MVHM. Then, we determine if the generated MVHM dataset can be used to improve 3D hand pose estimators. To this end, a multi-view based approach is developed for inferring 3D hand poses. The experimental results show that the resultant pose estimator can be greatly boosted by leveraging the generated MVHM dataset, and performs favorably against existing methods. This work makes three major contributions, which are summarized as follows:

1. We propose an effective mechanism for compiling a large-scale multi-view hand mesh (MVHM) dataset for 3D hand pose estimator training. To the best of our knowledge, this is the first large-scale hand dataset with multi-view hand images, accurate mesh annotations, hand joint keypoints labels.

2. We present an end-to-end graph convolutional neural work based multi-view hand

pose estimation approach which leverage information in the multi-view image system to accurately predict 3D hand pose.

3. Our proposed approach achieves the state-of-the-art performance on the benchmark, the MHP dataset, in both single-view and multi-view settings.

Table 5.1: Comparison between our dataset with publicly available datasets. `Auto` in field `Annotation` represents that the annotation is made by some algorithms and therefore may not be accurate.`Mano` means the emsh annotaion is fitted by Mano Model

Dataset	RGB	Depth	Image	Resolution	Annotation	Size	Multi-View	Mesh
ICVL [81]	✗	✓	real	320 × 320	tracking	18K	✗	✗
NYU [83]	✗	✓	real	648 × 480	tracking	243K	✗	✗
MSRA [80]	✗	✓	real	1920 × 1080	tracking	76K	✗	✗
BigHand2.2M [97]	✗	✓	real	640 × 480	marker	2.2M	✗	✗
STB [100]	✓	✓	real	640 × 480	manual	36K	✗	✗
RHP [105]	✓	✓	synthetic	640 × 480	synthetic	44K	✗	✗
Dexter+Object [78]	✓	✓	real	640 × 480	manual	3K	✗	✗
EgoDexter [60]	✓	✓	real	640 × 480	manual	3K	✗	✗
MHP [31]	✓	✗	real	480 × 480	auto	80K	✓	✗
FreiHand [106]	✓	✗	real	224 × 224	auto	134K	✗	Mano
InterHand [58]	✓	✗	real	512 × 334	auto	2.2M	✓	Mano
Youtube Hand [52]	✓	✗	real	256×256	auto	47K	✗	Mano
Ours	✓	✓	synthetic	256×256	synthetic	320K	✓	✓

5.2 Related Work

5.2.1 Multi-View Hand Pose Estimation

Unlike single-view pose estimation, few research efforts focus on 3D hand pose estimation from multi-view data. Ge *et al* [27] first introduce multi-view CNN to formulate it as an estimation problem. Their method assumes that hand joint locations independently follow 3D Gaussian distributions, and uses CNN to estimate the mean and variance of the location distribution of each joint. The main drawbacks of their method include 1) its inability to train in an end-to-end manner and 2) its impractical assumption about the independence among different joint locations. Simon *et al* [74] propose a multi-view system which is trained to progressively improve hand keypoints detection. Their method would work well on fine-tuning a well pre-trained estimator but could not train a 3D hand pose estimator from scratch.

5.2.2 3D Hand Pose Benchmark

There exist extensive research efforts such as [81, 83, 80, 97, 100, 105, 78, 60, 58, 52, 106] on building hand datasets for 3D hand pose estimation. We summarize the publicly available hand datasets and our dataset in Table 5.1. Most existing datasets do not contain mesh information, since labeling hand meshes manually is almost infeasible for human annotators.

To address the issue of labor-intensive annotations, recent studies [106, 52, 14] propose semi-automatic ways to label RGB images. FreiHand (Zimmermann *et al* [106]) use an iterative process where the trained models first make predictions on the images and then the annotators are asked to make necessary adjustments. YoutubeHand (Kulon *et al* [52]) run OpenPose [9] to get 2D annotations, upon which the parameters of the MANO model are regressed. Thresholding according to confidence scores is applied to remove those with low confidence, and hence ensures annotation quality. Despite the progress on efficiency and efficacy of labeling RGB images, the accuracy of annotation relies heavily on the pre-trained models used in the process. In addition, these methods rely on the MANO model as the ground-truth mesh generator, which could lose high-dimensional information of hands, as mentioned in Section 1.2.3. Compared to existing datasets, our dataset consists of large-scale RGB images and includes a variety of sequences. In addition, synthetic nature provides 100% accurate annotation for both hand joints and mesh. We make the first attempt to collect the dataset that provides large-scale, multi-view training images, thereby enhancing pose estimator training with a multi-view perspective.

5.3 Methodology

5.3.1 Overview

Given an RGB image of a hand $\mathbf{I} \in \mathbb{R}^{W \times H \times 3}$, our goal is to estimate the 3D joint locations of the hand $\mathbf{P}_j \in \mathbb{R}^{k \times 3}$, where W and H denote the image height and width respectively, and K is the number of the hand joints. Recent studies [28, 6] have demonstrated that using the mesh distance loss as an intermediate supervision during training can boost the performance of the learned hand pose estimator. Inspired by the approach [28], we define a hand mesh as a bidirectional graph $\mathbf{G}(\mathbf{V}, \mathbf{\Lambda})$, where $\mathbf{V}$ is the vertex set and $\mathbf{\Lambda}$ is the adjacency matrix. We also assume that $\mathbf{V}$ contains N different elements (*i.e.,* , points on the mesh) and our mesh estimator would predict the 3D locations $\mathbf{P}_m \in \mathbb{R}^{N \times 3}$ for all vertices in $\mathbf{V}$.

In our single-view approach, we use the stacked hourglass [61] as the CNN backbone to extract hand features from an image. The graph convolution network (GCN) is applied to estimate the 3D pose and mesh. Figure 5.2 shows the architecture of our single-view network, which consists of three major components: the 2D evidence network, mesh evidence network, and 3D pose estimator. These components are elaborated in the following subsections.

5.3.2 2D Evidence Network

The 2D evidence network offers two main functionalities. First, it estimates hand keypoint heatmaps to obtain the 2D hand joint locations. Second, it extracts image features that then serve as the input to the mesh evidence network. We denote the estimated heatmaps as $\boldsymbol{H} \in \mathbb{R}^{K \times H \times W}$. As shown in Figure 5.2, the hourglass backbone gives two outputs, including the estimated hand joint heatmaps and the extracted features. The ground-truth heatmaps $\bar{\boldsymbol{H}}_s$ are obtained by smoothing the keypoint location k^{th} with Gaussian blur. To train the

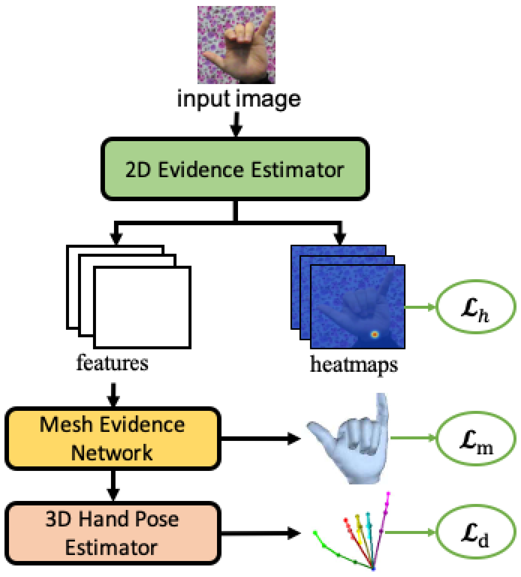


Figure 5.2: Overview of our single-view method. Given a single-view RGB image, the 2D evidence network predicts its heatmap and outputs the encoded image features. The mesh evidence network takes image features as input and outputs the hand mesh. Based on the estimated mesh, the 3D pose estimator gives final hand pose prediction. $\mathcal{L}_h$, $\mathcal{L}_m$, and $\mathcal{L}_d$ represent the heatmap loss, mesh loss, and depth loss for optimization, respectively.

2D evidence network, we apply the heatmap loss $\mathcal{L}_h$ to each hourglass block as supervision. The heatmap loss is defined by

$$\mathcal{L}_h = \frac{1}{S * K} \sum_{s=1}^{S} \sum_{k=1}^{K} ||\boldsymbol{H}_k^s - \bar{\boldsymbol{H}}_k||_F^2, \tag{5.1}$$

where S and K denote the number of the hourglass blocks and keypoints, respectively.

Mesh Evidence Network

Our mesh evidence network is built on the basis of spectral GCN [7]. Given the image features extracted by the 2D evidence network, our mesh evidence network estimates the 3D hand mesh. A 3D hand mesh is represented by a set of vertex coordinates $\mathbf{P}_m \in \mathbb{R}^{N \times 3}$ where N is the number of the vertices in the hand mesh. We represent a hand mesh as a graph $\mathbf{G}(\mathbf{V}, \mathbf{\Lambda})$, where $\mathbf{V}$ is the vertices set, and $\mathbf{\Lambda}$ is the adjacency matrix. $\mathbf{\Lambda}_{i,j}$ is 1 if there is an

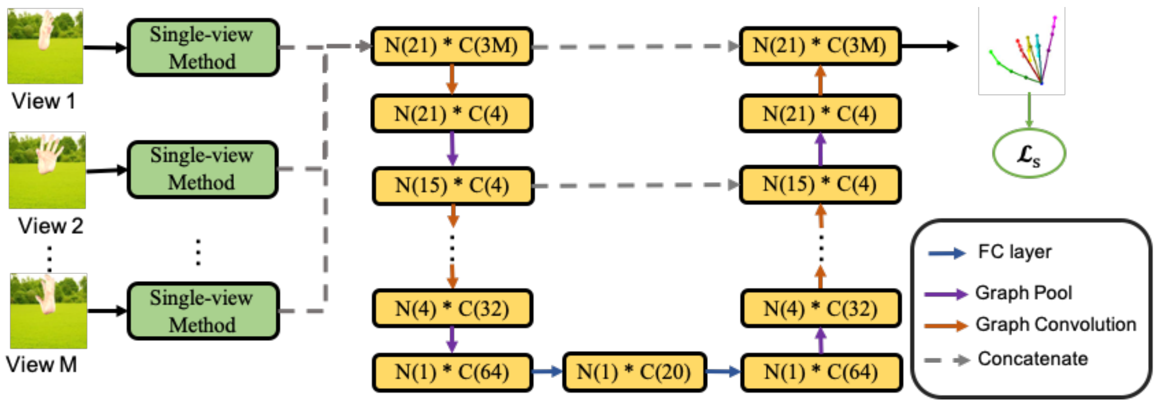


Figure 5.3: Overview of the multi-view method. Given multi-view images $\{I_1, I_2, .., I_M\}$, the single-view method first predicts the hand pose for each view independently. A graph U-Net takes the concatenation of these single-view predictions as input, and estimates the final pose estimation. $N(\cdot)$ and $C(\cdot)$ represent the number of nodes in the graph and feature size of each node, respectively. $\boldsymbol{\mathcal{L}}_s$ is the pose loss used for training the whole network.

edge between vertex i and vertex j, otherwise it is 0.

Specifically, we first normalize the adjacency matrix $\boldsymbol{\Lambda}$ via graph Laplacian operation and obtain a normalized adjacency matrix $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}} \Lambda \boldsymbol{D}^{-\frac{1}{2}}$, where $\boldsymbol{D}$ is the degree matrix of graph $\mathbf{G}$ and $\boldsymbol{I}$ is an identity matrix. Graph spectral decomposition is then used to decompose the normalized adjacency matrix $\boldsymbol{L}$ as $\boldsymbol{U}\boldsymbol{A}\boldsymbol{U}^T$, where $\boldsymbol{A} = diag(\lambda_1, \lambda_2, ..., \lambda_N)$ consists of the eigenvalues of the $\boldsymbol{L}$, where λ_{max} is the largest eigenvalue of $\boldsymbol{L}$.

Following [18], we define the convolution kernel $\hat{\boldsymbol{A}}$ in GCN as

$$\hat{\boldsymbol{A}} = diag(\sum_{i=0}^{S} \alpha_i \lambda_1^i, ..., \sum_{i=0}^{S} \alpha_i \lambda_C^i), \tag{5.2}$$

where α is the kernel parameter and S is a pre-set hyper-parameter used to control the receptive field.

Thus, the GCN convolutional operation is defined by

$$\boldsymbol{F'} = \boldsymbol{U}\hat{\boldsymbol{\Lambda}}\boldsymbol{U}^T \boldsymbol{F}\boldsymbol{\theta}_i = \sum_{i=0}^{S} \alpha_i \boldsymbol{L}^i \boldsymbol{F}\boldsymbol{\theta}_i, \tag{5.3}$$

where $\boldsymbol{F} \in \mathbb{R}^{N \times F_{\text{in}}}$ and $\boldsymbol{F'} \in \mathbb{R}^{N \times F_{\text{out}}}$ indicate the input and output features respectively, and $\boldsymbol{\theta}_i \in \mathbb{R}^{F_{\text{in}} \times F_{\text{out}}}$ is trainable parameter used to refine the input feature and control the output channel size.

Since our hand mesh surface is composed of 2561 vertices, it takes a huge computational cost to apply the above operation to each vertex because the time complexity of matrix multiplication for $\boldsymbol{L}^i$ is $O(N^3)$. Therefore, we utilize the Chebyshev polynomial approximation to reduce the complexity. The convolutional operation is then defined by

$$\boldsymbol{F'} = \sum_{i=0}^{S} \alpha_i T_i(\hat{\boldsymbol{L}})\boldsymbol{\theta}_i, \tag{5.4}$$

where $T_k(x)$ is the k^{th} Chebyshev polynomial and $\hat{\boldsymbol{L}} = 2\boldsymbol{L}/\lambda_{max} - \boldsymbol{I}$ is used to normalize the input features.

To enable our model to learn both local and global features, we adopt a scheme that is used in [18, 28] for generating hand meshes from coarse to fine. We leverage the heavy-edge matching algorithm to coarsen the graph by three different coarsening levels, and record the mapping between graph nodes in every two consecutive levels. In the forward pass, our model first constructs the most coarse hand mesh and then up-samples more nodes from the coarse graph to the fine graph based on the the stored mappings.

At the last layer of the GCN, we set F_{out} to 3 to represent the 3D coordinate vertices. Also we apply the l_2 loss between the ground-truth mesh and prediction map as the mesh loss function:

$$\boldsymbol{\mathcal{L}}_m = \frac{1}{N}||\boldsymbol{P_m} - \bar{\boldsymbol{P}}_{\boldsymbol{m}}||_F^2. \tag{5.5}$$

5.3.3 3D Depth Evidence Network

The proposed 3D evidence network infers the depth of 3D hand keypoints P_d from the predicted hand mesh P_m by the mesh evidence network. Taking P_m as the input, we adopt a two layers GCN with the similar structure of mesh evidence network to predict the pose features. These pose features are then fed to two fully connected layers to regress the depth of 3D hand keypoint locations. The corresponding loss is defined by

$$\boldsymbol{\mathcal{L}}_d = \frac{1}{K}||\boldsymbol{D} - \bar{\boldsymbol{D}}||_F^2, \tag{5.6}$$

where $\boldsymbol{D} \in \mathbb{R}^K$ and $\bar{\boldsymbol{D}} \in \mathbb{R}^K$ represent the predicted and the ground-truth joint depths, respectively.

To infer the 3D hand keypoints, we use non-maximum suppression to get the 2D coordinates from the estimated heatmaps. With the estimated 2D coordinates and the depth map calculated by 3D depth evidence network, we then obtain 3D coordinates in the camera coordinate system. Since the camera parameters are known, we are then able to infer hand keypoints in the world system.

5.3.4 Multi-view Method

Based on our single-view method, we propose a simple yet effective multi-view approach to hand pose estimation . Figure 5.3 illustrates the core idea of our approach.

Our single-view method predicts the 3D hand pose for each view independently. We concatenate these view-specific predictions on their coordinate channel. The concatenated prediction serves as the input features to a graph U-Nets[23] and predicts the final 3D hand keypoints.

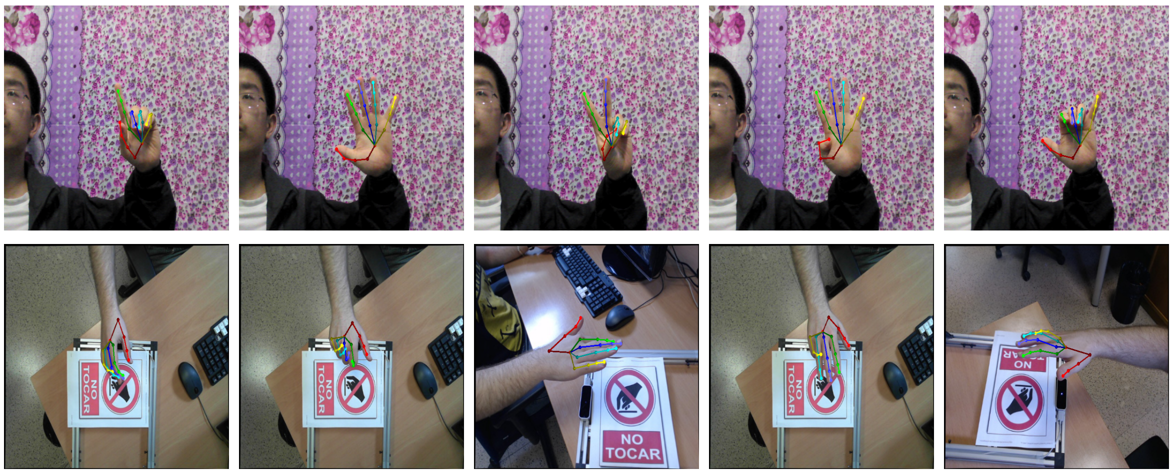


Figure 5.4: Examples of the two hand pose datasets used for evaluation. The first row shows images with the annotated hand poses from the STB dataset [100] while the second row shows those from the MHP dataset [31].

We utilize the l_2 distance as the loss function in our multi-view network, *i.e.,* ,

$$\boldsymbol{\mathcal{L}}_p = \frac{1}{K}||\boldsymbol{P}_j - \bar{\boldsymbol{P}}_j||_F^2, \tag{5.7}$$

where $\boldsymbol{P}_j$ and $\bar{\boldsymbol{P}}_j$ represent the predicted and the ground-truth joint depth, respectively.

5.4 Experiment Setting

5.4.1 Datasets for Evaluation

We evaluate our single-view approach on two benchmark hand pose datasets, including the Stereo Tracking Benchmark (STB) Dataset [100] and the Multi-view 3D Hand Pose (MHP) dataset [31]. The proposed multi-view approach is evaluated on the MHP dataset. Both of the MHP and STB provide real hand video sequences performed by different people in various background. The hand joint annotations of the STB dataset are manually labeled while the annotations of the MHP dataset are obtained by using the Leap Motion sensor.

Table 5.2: Ablation studies of 3D hand pose estimation on the STB and MHP datasets. $\uparrow$: higher is better; $\downarrow$: lower is better; The measuring unit of EPE is millimeter(mm). SV stands for the single-view method and MV represents the multi-view method.

	$\text{AUC}_{\text{0-50}} \uparrow$	$\text{AUC}_{\text{20-50}} \uparrow$	$\text{EPE}_m \downarrow$
MHP Dataset			
SV w/o MVHM	0.604	0.802	22.13
SV w/ MVHM	0.660	0.857	18.09
MV w/o MVHM	0.832	0.985	8.43
MV w/ MVHM	**0.895**	**0.990**	**5.20**
STB Dataset			
SV w/o MVHM	0.820	0.987	8.95
SV w/ MVHM	**0.832**	**0.991**	**8.38**

The MVHM dataset we build is used in all of our experiments. We aim at determining if training hand pose estimators with the MVHM dataset can be effectively improved in different experimental settings.

For STB dataset, we use its SK subset, which contains 6 different hand videos, to evaluate our approach. Following the train-validation split setting in [28], we take the first video as the validation set while the rest videos serve as the training set.

The MHP dataset includes 21 different hand motion videos. Each hand motion video provides hand RGB images and multiple types of annotations in each sample, including bounding boxes and 2D/3D hand joint locations. Figure 5.4 displays some examples of the STB and MHP datasets. We follow [8, 105] and apply the standard data pre-processing for both of the STB and MHP datasets. During data pre-processing, we firstly crop the images to remove the irrelevant background and make sure the hands are located at the center of the images. All the cropped images are then resized to resolution 256×256. Secondly, we follow the mechanism used in [8] to change the hand center from the palm center to the joint of wrist for data in both of the STB and MHP datasets.

Table 5.3: Results on the MHP dataset. ↑: higher is better.

	$\text{AUC}_{20\text{-}50}$ ↑
Zimmermann *et al*[106]	0.717
Cai *et al* [8]	0.928
Chen *et al* [12]	0.939
Our multi-view method	**0.991**

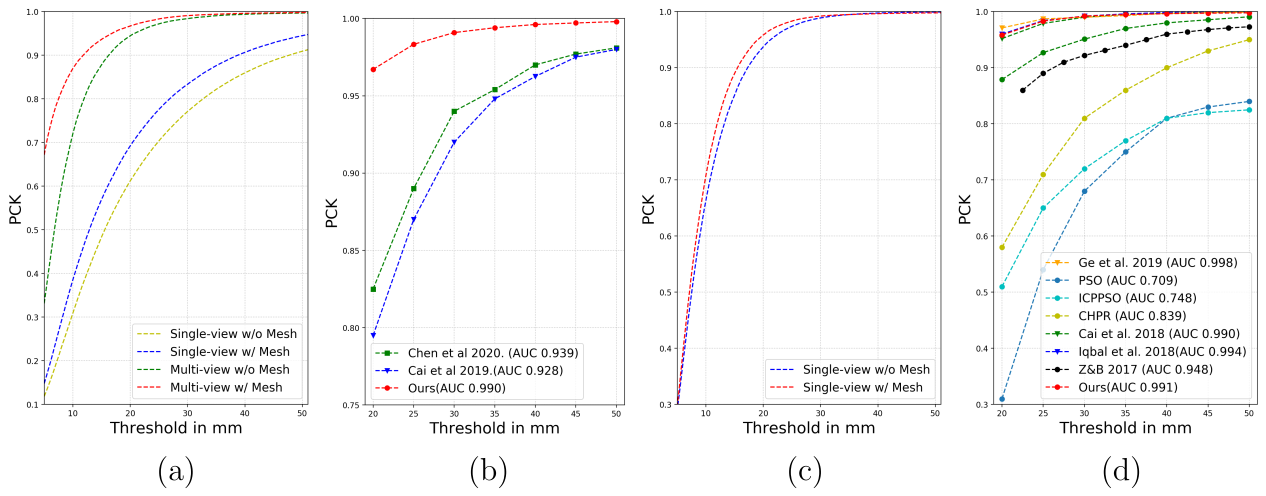


Figure 5.5: Ablation studies and comparison of the state-of-the-art methods for single-view pose estimation. (a) PCK results of different settings on the STB dataset. (b) Comparison results in PCK for the state-of-the-art methods on the STB dataset. (c) PCK results under different settings on the MHP dataset. (d) Comparison results in PCK for the state-of-the-art methods on the MHP dataset.

5.4.2 Metrics

We follow the settings from previous researches [105, 28] and adopt the *average end-point-error* (EPE$_m$), and the *area under the curve* (AUC) on the *percentage of correct keypoints* (PCK) within a threshold range as the metrics to evaluate model effectiveness. We report the performance in both AUC on PCK between 0mm and 50mm as well as between 20mm and 50mm.

5.4.3 Implementation Details

We implement our single-view and multi-view approaches in Python with PyTorch. In the training phase, we set the batch size as 8, and use the Adam solver with an initial learning rate 0.01. Both models are trained on a server with four GeForce GTX 1080-Ti GPUs.

When training the single-view network, we use a multi-stage training strategy. In the first stage, we train our 2D evidence network with the heatmap loss $\boldsymbol{\mathcal{L}}_h$. In the second stage, we fix the weights of the 2D evidence network and train the mesh network with mesh loss $\boldsymbol{\mathcal{L}}_m$. In the third stage, we fix the weights of both of the 2D evidence network and mesh network, and focus on training the joint depth network with loss $\boldsymbol{\mathcal{L}}_d$. In the final stage, the whole network is optimized end-to-end.

For training the multi-view network, we apply the same multi-stage training strategy. In the first stage, we use the pre-trained weights from the single-view network for initializing the 3D hand single-view network, and keep this part fixed for training the 3D hand fusion network. In the second stage, we activate both networks and fine-tune the whole network architecture in an end-to-end manner.

5.5 Experimental Results

5.5.1 Multi-view task

To evaluate the effectiveness of the proposed multi-view method, we compare our single-view method with our multi-view method on the MHP dataset under the setting of with or without using data from the MVHM dataset. Table 5.2 and Figure 5.5(a) show that utilizing the multi-view information from the MHP dataset itself boosts the testing performance in $\text{AUC}_{\text{0-50}}$, $\text{AUC}_{\text{20-50}}$, and EPE_m by large margins, *i.e.,* , 0.218, 0.183, and 13.80mm respectively. When additional data from the MVHM dataset are used, substantial performance gains are achieved, which reveals the effectiveness of using the collected MVHM dataset for training.

Three current state-of-the-art methods are chosen for comparing with our method on the MHP dataset, including Zimmermann *et al* [106] (0.717 in $\text{AUC}_{\text{20-50}}$), Cai *et al* [8] (0.928 in $\text{AUC}_{\text{20-50}}$)[1], and Chen *et al* [12] (0.939 in $\text{AUC}_{\text{20-50}}$). Zimmermann *et al* [106] just report the numerical result so we include their result in Table 5.3 and does not show it in Figure 5.5(b). Our multi-view method achieves the performance of 0.990 in $\text{AUC}_{\text{20-50}}$, outperforming these competing methods by a large margin. This experiment shows that both the proposed multi-view method and the established MVHM dataset are beneficial and can work together to get the new state-of-the-art performance on the MHP dataset.

5.5.2 Single-view task

To further validate the effectiveness of the generated mesh dataset MVHM in addition to multi-view methods, we also conduct the following experiments for comparison on single-

[1]Cai *et al* [8] do not report the results in their paper. Here we report the re-implementaition results by Chen *et al* [12].

view methods. We compare the results when models are trained solely on the MHP/STB datasets and trained on the MHP/STB datasets together with the MVHM dataset. Table 5.2, Figure 5.5(a) and Figure 5.5(c) show, on both MHP and STB datasets, adding the mesh data greatly enhances the performance by granting a model the ability to capture the mesh-level features, therefore leading to better results.

We select seven powerful and recently published methods for comparison with the proposed method, including PSO [6], ICPPSO [15], CHPR [100], Iqbal *et al* [40], Cai *et al* [8], Zimmermann and Brox [105], and Ge *et al* [28]. The AUC curves are plotted in Figure 5.5(d). Ge *et al* [28] also utilize an additional dataset to train their model and got the STOA result, which demonstrates the effectiveness of their mesh dataset. Besides, they introduce more complicated mesh metrics like the surface norm loss. Iqbal *et al* [40] and Cai *et al* [8] leverage additional depth-map information as derive their models, and achieve good results. As a multi-view approach without complicated components, our method is on par with methods by Ge *et al* [28] and Iqbal *et al* [40] while outperforms most of them on single-view tasks.

5.6 Conclusions

Estimating 3D hand poses from monocular images is a ill-posed problem due to its depth ambiguity. Nevertheless, multi-view images could make up its deficiency. To this end, we build a multi-view mesh hand dataset, MVHM, to enable training 3D pose estimators with mesh supervision. We present a multi-view method that effectively fuses single-view predictions. When testing on the real-world multi-view dataset MHP, our multi-view method with the aid of the MVHM dataset achieves the state-of-the-art performance.

Chapter 6

Conclusion and Outlook

6.1 Conclusion and Contributions

The rapid progress in deep learning has reformed the common practice in 3D hand pose estimation. Deep Learning methods, by their nature, requires a large size of accurate data to train. However, providing high quality hand data remains a challenge problem. In this dissertation, we propose several methods to resolve this problem.

In chapter 2, we discussed how to make a synthetic hand image look more real by TAGAN which aims at bridging the gap between real and synthetic images while keeping the accurate hand keypoint annotation.

Paired RGB and Depth images are required for training most state-of-art $3D$ hand pose estimator. In chapter 3, we proposed the DGGAN, which loosens the requirements of paired RGB and Depth images, and make those models be able to train on RGB image-only datasets.

Taken one step further, in chapter 4, we explored, whether it is possible to train a $3D$ hand pose estimator even without explicit 3D annotation. We propose TASSN, which uses self-supervised mechanism and temporal constrain to learns 3D hand poses and meshes from videos with only 2D keypoint annotations.

In chapter 5. we switched our focus from the single-view hand pose estimation to multi-view hand pose estimation. We designed a data pipeline to easily generate a large scale multi-view 3D hand pose and mesh benchmark. In addition to that, we proposed a simple yet efficient method to utilize the multi-view information which achieve the state-of-the-art performance on MHP dataset.

All these methods helped the refine existing data, alleviate the training requirement, and can be potentially applied to large industry application.

6.2 Future Directions

Apart from the progress discussed above, there are still many interesting topics that we want to explore in future work.

In chapter 4, we discussed that self-supervised learning could serve as a potential way to replace groundturth. Till now, not much any research work has focused on this topic Wan *et al* [85] proposed the silhouette constrain for training, we proposed temporal constrain for training 3D hand pose estimation. Besides this two constrain, is there any other possibility for training 3D hand pose estimation along this path remain in interesting problem to us.

In chapter 5, we proposed a simple yet efficient multi-view method for 3 Hand pose estimation. We fused the image features by concatenating them together and sending to the next state, which is a rather naive way to do so. Whether there is another way to better fuse those features remains an interesting question to us.

Compared to the multi-view method, the multi-frame setting would be a more common application. Calibrating the hand size through the multi-frame hand dataset is a promising way for eliminating the depth ambiguity and achieving better accuracy.

Bibliography

[1] M. Abdi, E. Abbasnejad, C. P. Lim, and S. Nahavandi. 3d hand pose estimation using simulation and partial-supervision with a shared latent space. In *BMVC*, 2018.

[2] S. Antoshchuk, M. Kovalenko, and J. Sieck. Gesture recognition-based human–computer interaction interface for multimedia applications. In *Digitisation of Culture: Namibian and International Perspectives*. Springer, 2018.

[3] S. Anwar, S. K. Sinha, S. Vivek, and V. Ashank. Hand gesture recognition: a survey. In *Nanoelectronics, Circuits and Communication Systems*, pages 365–371. Springer, 2019.

[4] S. Baek, K. I. Kim, and T.-K. Kim. Augmented skeleton space transfer for depth-based hand pose estimation. In *CVPR*, 2018.

[5] S. Baek, K. I. Kim, and T.-K. Kim. Pushing the envelope for rgb-based dense 3d hand pose estimation via neural rendering. In *CVPR*, 2019.

[6] A. Boukhayma, R. d. Bem, and P. H. Torr. 3d hand shape and pose from images in the wild. In *CVPR*, 2019.

[7] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun. Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*, 2013.

[8] Y. Cai, L. Ge, J. Cai, and J. Yuan. Weakly-supervised 3d hand pose estimation from monocular rgb images. In *ECCV*, 2018.

[9] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. In *TPAMI*, 2018.

[10] Z. Cao, T. Simon, S.-E. Wei, and Y. Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.

[11] T. Chalasani and A. Smolic. Simultaneous segmentation and recognition: Towards more accurate ego gesture recognition. In *ICCVW*, 2019.

[12] L. Chen, S.-Y. Lin, Y. Xie, Y.-Y. Lin, W. Fan, and X. Xie. Dggan: Depth-image guided generative adversarial networks fordisentangling rgb and depth images in 3d hand pose estimation. In *WACV*, 2020.

[13] L. Chen, S.-Y. Lin, Y. Xie, Y.-Y. Lin, and X. Xie. Mvhm: A large-scale multi-view hand mesh benchmark for accurate 3d hand pose estimation. In *WACV*, 2021.

[14] L. Chen, S.-Y. Lin, Y. Xie, Y.-Y. Lin, and X. Xie. Temporal-aware self-supervised learning for 3d hand pose and mesh estimation in videos. In *WACV*, 2021.

[15] L. Chen, S.-Y. Lin, Y. Xie, H. Tang, Y. Xue, Y.-Y. Lin, X. Xie, and W. Fan. Tagan: Tonality-alignment generative adversarial networks for realistic hand pose synthesis. In *BMVC*, 2019.

[16] Y.-C. Chen, Y.-Y. Lin, M.-H. Yang, and J.-B. Huang. CrDoCo: Pixel-level domain transfer with cross-domain consistency. In *CVPR*, 2019.

[17] U. Côté-Allard, C. L. Fall, A. Drouin, A. Campeau-Lecours, C. Gosselin, K. Glette, F. Laviolette, and B. Gosselin. Deep learning for electromyographic hand gesture signal classification using transfer learning. *IEEE Trans Neural Systems and Rehabilitation Engineering*, 27(4):760–771, 2019.

[18] M. Defferrard, X. Bresson, and P. Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. In *NeurIPS*, 2016.

[19] X. Deng, S. Yang, Y. Zhang, P. Tan, L. Chang, and H. Wang. Hand3d: Hand pose estimation using 3d neural network. *arXiv preprint arXiv:1704.02224*, 2017.

[20] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised visual representation learning by context prediction. In *CVPR*, 2015.

[21] D. Dwibedi, Y. Aytar, J. Tompson, P. Sermanet, and A. Zisserman. Temporal cycle-consistency learning. In *CVPR*, 2019.

[22] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108(1-2):52–73, 2007.

[23] H. Gao and S. Ji. Graph u-nets. *arXiv preprint arXiv:1905.05178*, 2019.

[24] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *CVPR*, 2018.

[25] L. Ge, Y. Cai, J. Weng, and J. Yuan. Hand pointnet: 3d hand pose estimation using point sets. In *CVPR*, 2018.

[26] L. Ge, Y. Cai, J. Weng, and J. Yuan. Hand pointnet: 3d hand pose estimation using point sets. In *CVPR*, 2018.

[27] L. Ge, H. Liang, J. Yuan, and D. Thalmann. Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns. In *CVPR*, pages 3593–3601, 2016.

[28] L. Ge, Z. Ren, Y. Li, Z. Xue, Y. Wang, J. Cai, and J. Yuan. 3d hand shape and pose estimation from a single rgb image. In *CVPR*, 2019.

[29] L. Ge, Z. Ren, and J. Yuan. Point-to-point regression pointnet for 3d hand pose estimation. In *ECCV*, 2018.

[30] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[31] F. Gomez-Donoso, S. Orts-Escolano, and M. Cazorla. Large-scale multiview 3d hand pose dataset. *arXiv preprint arXiv:1707.03742*, 2017.

[32] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014.

[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NeurIPS*, 2014.

[34] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[35] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017.

[36] F. Huang, A. Zeng, M. Liu, J. Qin, and Q. Xu. Structure-aware 3d hourglass network for hand pose estimation from single depth image. In *BMVC*, 2018.

[37] Y.-P. Hung and S.-Y. Lin. Re-anchorable virtual panel in three-dimensional space, Dec. 27 2016. US Patent 9,529,446.

[38] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. In *CVPR*, 2017.

[39] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7), jul 2014.

[40] U. Iqbal, P. Molchanov, T. Breuel Juergen Gall, and J. Kautz. Hand pose estimation via latent 2.5 d heatmap regression. In *ECCV*, 2018.

[41] M. M. Ismail, C. Lam, K. Sundaraj, and M. Rahiman. Hand motion pattern recognition analysis of forearm muscle using mmg signals. *Bulletin of Electrical Engineering and Informatics*, 8(2):533–540, 2019.

[42] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[43] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *CVPR*, 2017.

[44] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017.

[45] H. Joo, T. Simon, and Y. Sheikh. Total capture: A 3d deformation model for tracking faces, hands, and bodies. In *CVPR*, 2018.

[46] D. Joseph 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 *CVPR*, 2016.

[47] H. Kato, Y. Ushiku, and T. Harada. Neural 3d mesh renderer. In *CVPR*, 2018.

[48] J. Kennedy. Particle swarm optimization. *Encyclopedia of machine learning*, 2010.

[49] S. Khamis, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon. Learning an efficient model of hand shape variation from depth images. In *CVPR*, 2015.

[50] A. U. Khan and A. Borji. Analysis of hand segmentation in the wild. In *CVPR*, 2018.

[51] M. Kokic, D. Kragic, and J. Bohg. Learning to estimate pose and shape of hand-held objects from rgb images. *arXiv preprint arXiv:1903.03340*, 2019.

[52] D. Kulon, R. A. Guler, I. Kokkinos, M. M. Bronstein, and S. Zafeiriou. Weakly-supervised mesh-convolutional hand reconstruction in the wild. In *CVPR*, pages 4990–5000, 2020.

[53] S. Li and D. Lee. Point-to-pose voting based hand pose estimation using residual permutation equivariant layer. *arXiv preprint arXiv:1812.02050*, 2018.

[54] S.-Y. Lin, C.-K. Shie, S.-C. Chen, and Y.-P. Hung. Airtouch panel: a re-anchorable virtual touch panel. In *MM*, 2013.

[55] A. Makris and A. Argyros. Model-based 3d hand tracking with on-line hand shape adaptation. In *BMVC*, 2015.

[56] M. Mirza and S. Osindero. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784*, 2014.

[57] 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 *CVPR*, 2018.

[58] G. Moon, S.-I. Yu, H. Wen, T. Shiratori, and K. M. Lee. Interhand2. 6m: A dataset and baseline for 3d interacting hand pose estimation from a single rgb image. In *ECCV*, 2020.

[59] F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt. Ganerated hands for real-time 3d hand tracking from monocular rgb. In *CVPR*, 2018.

[60] 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 *ICCVW*, 2017.

[61] A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499, 2016.

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

[63] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Tracking the articulated motion of two strongly interacting hands. In *CVPR*, 2012.

[64] R. Pandey, P. Pidlypenskyi, S. Yang, and C. Kaeser-Chen. Efficient 6-dof tracking of handheld objects from an egocentric viewpoint. In *ECCV*, 2018.

[65] P. Panteleris, I. Oikonomidis, and A. Argyros. Using a single rgb frame for real time 3d hand pose estimation in the wild. In *WACV*, 2018.

[66] D. Pathak, R. Girshick, P. Dollár, T. Darrell, and B. Hariharan. Learning features by watching objects move. In *CVPR*, 2017.

[67] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

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

[69] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[70] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics*, 36(6):245, 2017.

[71] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: Interactive foreground extraction using iterated graph cuts. In *ACM Transactions on Graphics*, 2004.

[72] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *European conference on computer vision*, pages 213–226. Springer, 2010.

[73] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb. Learning from simulated and unsupervised images through adversarial training. In *CVPR*, 2017.

[74] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017.

[75] T. Simon, H. Joo, I. A. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR*, 2017.

[76] A. Spurr, J. Song, S. Park, and O. Hilliges. Cross-modal deep variational hand pose estimation. In *CVPR*, 2018.

[77] S. Sridhar, A. M. Feit, C. Theobalt, and A. Oulasvirta. Investigating the dexterity of multi-finger input for mid-air text entry. In *CHI*, 2015.

[78] S. Sridhar, F. Mueller, M. Zollhöfer, D. Casas, A. Oulasvirta, and C. Theobalt. Real-time joint tracking of a hand manipulating an object from rgb-d input. In *ECCV*, 2016.

[79] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *CVPR*, pages 824–832, 2015.

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

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

[82] B. Tekin, F. Bogo, and M. Pollefeys. H+ o: Unified egocentric recognition of 3d hand-object poses and interactions. *CVPR*, 2019.

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

[84] A. Torralba, A. A. Efros, et al. Unbiased look at dataset bias. In *CVPR*. Citeseer, 2011.

[85] C. Wan, T. Probst, L. V. Gool, and A. Yao. Self-supervised 3d hand pose estimation through training by fitting. In *CVPR*, 2019.

[86] C. Wan, T. Probst, L. Van Gool, and A. Yao. Dense 3d regression for hand pose estimation. In *CVPR*, 2018.

[87] C. Wan, T. Probst, L. Van Gool, and A. Yao. Dense 3d regression for hand pose estimation. In *CVPR*, 2018.

[88] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, 2016.

[89] X. Wu, D. Finnegan, E. O'Neill, and Y.-L. Yang. Handmap: Robust hand pose estimation via intermediate dense guidance map supervision. In *ECCV*, 2018.

[90] Z. Wu, D. Hoang, S.-Y. Lin, Y. Xie, L. Chen, Y.-Y. Lin, Z. Wang, and W. Fan. Mm-hand: 3d-aware multi-modal guided hand generative network for 3d hand pose synthesis. In *MM*, 2020.

[91] S. Xie and Z. Tu. Holistically-nested edge detection. In *CVPR*, 2015.

[92] J. Yang, H. J. Chang, S. Lee, and N. Kwak. Seqhand: Rgb-sequence-based 3d hand pose and shape estimation. *arXiv preprint arXiv:2007.05168*, 2020.

[93] L. Yang and A. Yao. Disentangling latent hands for image synthesis and pose estimation. *CVPR*, 2019.

[94] Q. Ye and T.-K. Kim. Occlusion-aware hand pose estimation using hierarchical mixture density network. In *ECCV*, 2018.

[95] S. Yuan, G. Garcia-Hernando, B. Stenger, G. Moon, J. Yong Chang, K. Mu Lee, P. Molchanov, J. Kautz, S. Honari, L. Ge, et al. Depth-based 3d hand pose estimation: From current achievements to future goals. In *CVPR*, 2018.

[96] S. Yuan, G. Garcia-Hernando, B. Stenger, G. Moon, J. Yong Chang, K. Mu 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. Argyros, and T.-K. Kim. Depth-based 3d hand pose estimation: From current achievements to future goals. In *CVPR*, 2018.

[97] S. Yuan, Q. Ye, B. Stenger, S. Jain, and T.-K. Kim. Bighand2. 2m benchmark: Hand pose dataset and state of the art analysis. In *CVPR*, 2017.

[98] Z. Zafrulla, H. Brashear, T. Starner, H. Hamilton, and P. Presti. American sign language recognition with the kinect. In *ICMI*, 2011.

[99] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang. 3d hand pose tracking and estimation using stereo matching. In *ICIP*, 2016.

[100] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang. 3d hand pose tracking and estimation using stereo matching. *arXiv preprint arXiv:1610.07214*, 2016.

[101] X. Zhang, Q. Li, W. Zhang, and W. Zheng. End-to-end hand mesh recovery from a monocular rgb image. *arXiv preprint arXiv:1902.09305*, 2019.

[102] Y. Zhou, J. Lu, K. Du, X. Lin, Y. Sun, and X. Ma. Hbe: Hand branch ensemble network for real-time 3d hand pose estimation. In *ECCV*, 2018.

[103] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.

[104] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networkss. In *ICCV*, 2017.

[105] C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single rgb images. In *CVPR*, 2017.

[106] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *ICCV*, 2019.