Ph.D. DISSERTATION

Exploitation of Synthetic Data for 3D Hand Pose and Shape Estimation

3D 손 포즈 인식을 위한 인조 데이터의 이용

BY

YANG JOHN

AUGUST 2021

Intelligent Systems
Department of Transdisciplinary Studies
Graduate School of Convergence Science and Technology
SEOUL NATIONAL UNIVERSITY

Exploitation of Synthetic Data for 3D Hand Pose and Shape Estimation

3D 손 포즈 인식을 위한 인조 데이터의 이용

지도교수 곽 노 준

이 논문을 공학박사 학위논문으로 제출함

2021년 8월

서울대학교 대학원

융합과학부 지능형융합시스템전공

YANG JOHN

YANG JOHN의 공학박사 학위 논문을 인준함

2021년 8월

Abstract

3D hand pose estimation (HPE) based on RGB images has been studied for a long time. Relevant methods have focused mainly on optimization of neural framework for graphically connected finger joints. Training RGB-based HPE models has not been easy to train because of the scarcity on RGB hand pose datasets; unlike human body pose datasets, the finger joints that span hand postures are structured delicately and exquisitely. Such structure makes accurately annotating each joint with unique 3D world coordinates difficult, which is why many conventional methods rely on synthetic data samples to cover large variations of hand postures.

Synthetic dataset consists of very precise annotations of ground truths, and further allows control over the variety of data samples, yielding a learning model to be trained with a large pose space. Most of the studies, however, have performed frame-by-frame estimation based on independent static images. Synthetic visual data can provide practically infinite diversity and rich labels, while avoiding ethical issues with privacy and bias. However, for many tasks, current models trained on synthetic data generalize poorly to real data. The task of 3D human hand pose estimation is a particularly interesting example of this synthetic-to-real problem, because learning-based approaches perform reasonably well given real training data, yet labeled 3D poses are extremely difficult to obtain in the wild, limiting scalability.

In this dissertation, we attempt to not only consider the appearance of a hand but incorporate the temporal movement information of a hand in motion into the learning framework for better 3D hand pose estimation performance, which leads to the necessity of a large scale dataset with sequential RGB hand images. We propose a novel method that generates a synthetic dataset that mimics natural human hand movements by re-engineering annotations of an extant static hand pose dataset into pose-flows. With the generated dataset, we train a newly proposed recurrent framework, exploiting visuo-temporal features from sequential images of synthetic hands in motion and emphasizing temporal smoothness of estimations with a temporal consistency constraint. Our novel training strategy of detaching the recurrent layer of the framework during domain finetuning from synthetic to real allows preservation of the visuo-temporal features learned from sequential synthetic hand images. Hand poses that are sequentially estimated consequently produce natural and smooth hand movements which lead to more robust estimations. We show that utilizing temporal information for 3D hand pose estimation significantly enhances general pose estimations by outperforming state-of-the-art methods in experiments on hand pose estimation benchmarks.

Since a fixed set of dataset provides a finite distribution of data samples, the generalization of a learning pose estimation network is limited in terms of pose, RGB and viewpoint spaces. We further propose to augment the data automatically such that the augmented pose sampling is performed in favor of training pose estimator's generalization performance. Such auto-augmentation of poses is performed within a learning feature space in order to avoid computational burden of generating synthetic sample for every iteration of updates. The proposed effort can be considered as generating and utilizing synthetic samples for network training in the feature space. This allows training efficiency by requiring less number of real data samples, enhanced generalization power over multiple dataset domains and estimation performance caused by efficient augmentation.

keywords: 3D hand pose estimation, RGB-based hand pose estimation, Synthetic Dataset, Feature-level Auto-Augment, temporal feature, hand motion
student number: 2016-30732

Contents

List of Tables

List of Figures

Chapter 1

Introduction

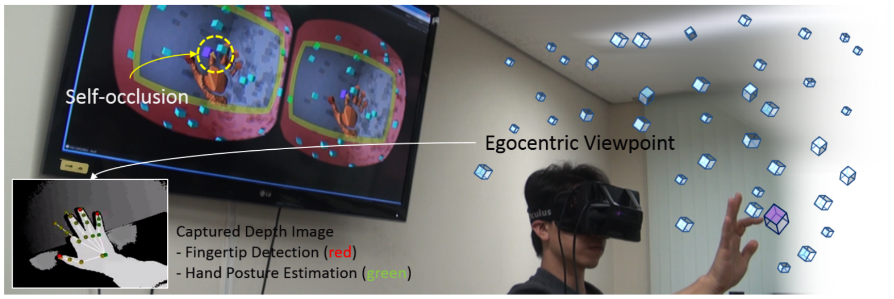

Self-occlusion
Egocentric Viewpoint
Captured Depth Image
- Fingertip Detection (red)
- Hand Posture Estimation (green)

Figure 1.1: An application of 3D hand pose estimation computer vision problems that allows virtual maneuver in AR/VR spaces. Originally used in [37]

Hand pose estimation (HPE) has been one of the critical computer vision problems. With the advance of deep learning technology, various approaches of neural inference have been studied the literature and have caused breakthroughs in general computer vision problems including hand pose estimation tasks [7, 8, 50, 84, 102]. Hand pose estimation is an essential task for augmented reality and virtual reality (collectively called as "extended reality (XR)") systems. For instance, to enable hand-based human-computer interactions (HCI) with objects in XR environments, accurate real-time estimates of the positions of hand joints in 3D world coordinates are needed [35]. Since hand gestures reflect elementary human behavioral patterns, hand pose tracking enables several downstream AI applications such as gesture recognition [37, 66, 74] and human-computer interactions [10]. In [47], Lee et al. detected hands to render an object in AR environment on the hand which was proportional to hand size. Piumsomboon et al. [66] focused on guessability in 40 different tasks in AR environment with studying hand gestures. Jang et al. [37] build an AR/VR system in egocentric viewpoint which was completely controllable via user's hands. Figure 1.1 shows one of the applications of hand pose estimation used in egocentric viewpoint in AR/VR headset [37].

Nevertheless the applications of hand pose estimation are not limited to AR/VR technologies. Sridhar et al. [76] built a system working with a number of finger actions. Markussen et al. [55] also proposed a mid-air keyboard to type on air, as depicted in Figures 1.2 and 1.3.

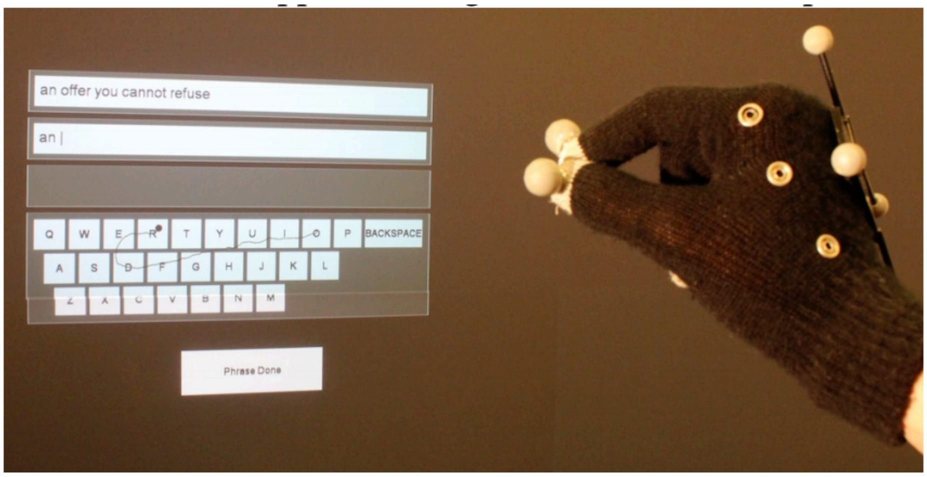


Figure 1.2: Users' hand gestures with a glove on are collected while writing sentences with a mid-air keyboard.

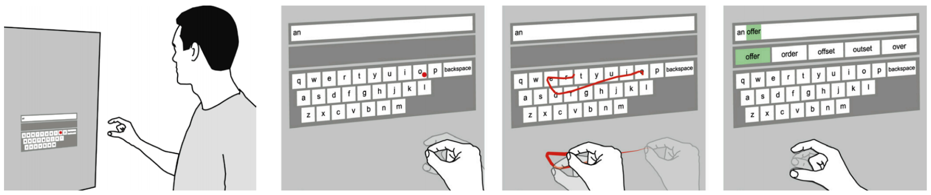


Figure 1.3: An example of Text entry using word-gestures in mid-air.

Yin *et al.* [91] used hand pose estimation to design a system that understands sign language. In a similar study, Chang *et al.* [10] used finger tip detection and tracking to read alphabet written by finger in the air. Outside HCI field, the works of [71] and [68] applied body and hand pose estimation systems for predicting body movements of a piano player and to detect activities.

In recent years, the interest in systems controlled by fingers, made researchers more ambitious to the extent that they discarded 2.5D depth map images and tried to estimate hand pose by a single RGB image. This method is a harder task and needs a considerable larger data to train. Below, we will first explain the hand pose estimation problem and discuss its variations and next we will discuss different methods in solving this problem. At the end of the paper we will briefly investigate new datasets in this field and will see how the size of datasets have changed dramatically through time.

Hand pose estimation is the process of modeling human hand as a set of some parts (e.g. palm and fingers) and finding their positions in a hand image (2D estimation) or the simulation of hand parts positions in a 3D space Although it is also used to estimate hand with the phalanges, in almost all the recent papers hands are modeled as a number of joints and the task is equivalent to finding the position of these joints. We can then estimate the real hand pose using those joints. While lots of the efforts had been focused on 2D pose estimation studies, recently, not only 3D pose estimations are studied but 3D shapes of hands are also expected as supplementary results [69, 29, 6].

1.1 RGB-based 3D Hand Pose Estimation

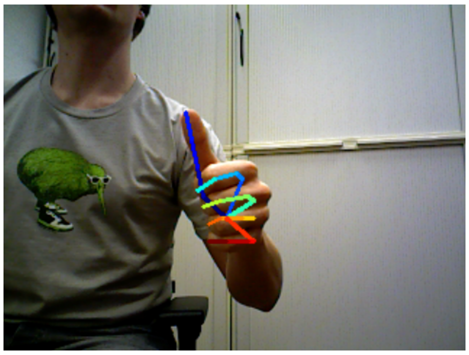


Figure 1.4: An example of an RGB image of a hand with which 3D posture is estimated. Image is originally used in [95]

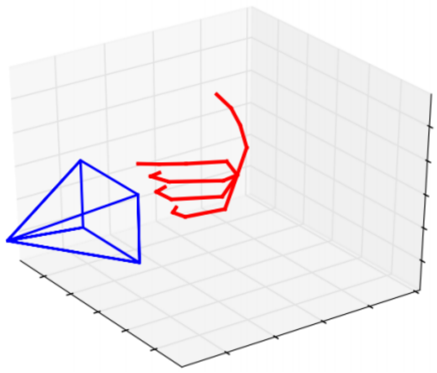


Figure 1.5: An example of 3D hand pose estimation results. Image is originally used in [95]

Hand pose estimation tasks are performed largely based on either depth maps or RGB image inputs, examples of which are depicted in Figures 1.4, 1.5 and

1.6. In spite of the fact that using a simple RGB image as an input gives the model a very good generalization power to be used everywhere, reducing the dimension of the input from 2.5D to 2D will make the task drastically harder. The data needed to train a network using RGB images is much bigger than the data needed to train a similar network using depth maps. Collection of RGB data is also difficult. Since annotating 3D location of finger joints precisely is a challenging work, conventional efforts include putting markers on each joint so that real-time 3D positions of the joints can be collected. As it can be seen in Figure 1.7, accurate 3D annotation of finger joints can be acquired by markers put on each joint. While this effectively logs finger positions in a 3D space, the markers (or gloves) that are put on hands severely deteriorates the RGB values to be considered by computer vision 3D hand pose estimators. Therefore, the literature lacks large scale datasets for RGB hand pose estimation tasks, many methods are reported with depth image inputs despite of generalization power of RGB-based methods [53, 58, 99, 60, 77].

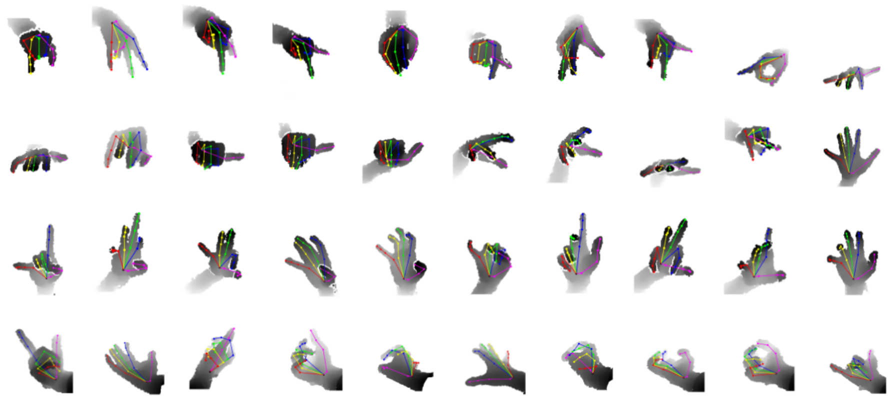


Figure 1.6: Other than RGB image inputs, depth images can be inputted for 3D HPE. This image is retrieved from the work of [93].

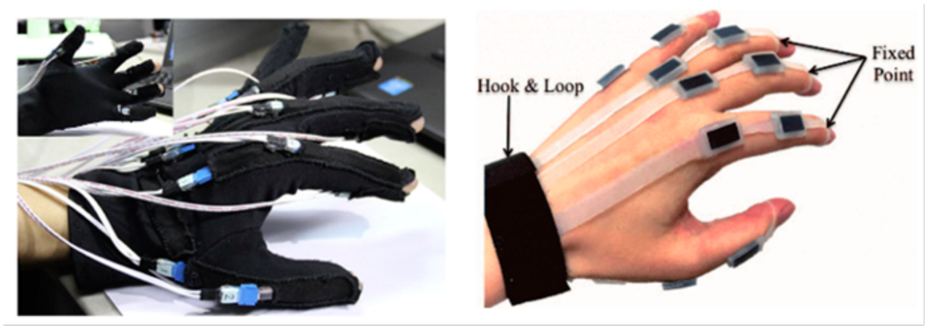


Figure 1.7: Accurate 3D annotation of finger joints can be acquired by markers put on each joint. While this effectively logs finger positions in a 3D space, the markers (or gloves) that are put on hands severely deteriorates the RGB values to be considered by computer vision 3D hand pose estimators. This is image is originally used in [13].

Hand pose estimation is considered as a separate field from hand detection tasks. Many hand pose estimators assume that their inputs are images that cropped around a single hand. To this end, note that image-based methods need to first isolate the hand (cropped and resized) and then are fed as the cropped images to the network in which hands are estimated.

As mentioned earlier, since the RGB image contains less information than depth disparity maps, the RGB-based networks are harder to train and requires a larger dataset. Synthetic visual data can provide practically infinite diversity and rich labels, while avoiding ethical issues with privacy and bias. However, for many tasks, current models trained on synthetic data generalize poorly to real data. The task of 3D hand pose estimation is a particularly interesting example of this synthetic-to-real domain gap problems, because learning-based approaches perform reasonably well given real training data, yet labeled 3D poses are extremely difficult to obtain in the wild, limiting scalability.

1.2 Needs of Synthetic 3D Hand Pose Data

3D hand pose estimation is one of classic computer vision problems, with applications in imitation learning, robotic interaction, and activity understanding. Hand pose estimation is extremely challenging with objects that are articulated, deformable, or have wide intra-class variation, as is the case with humans in many real-world scenarios. Therefore, state-of-the-art approaches rely on neural networks and learning. However, learning-based methods are extremely data-hungry, and acquiring sufficient data in the real world is difficult. First, there is no straightforward way for people to annotate 3D ground truth poses. Also, in settings like industrial warehouses or homes, hundreds of thousands of different object types may appear, and new objects may arrive at random. Here, even simple labeling will generally be impractical, much less 3D poses. And any time data involves real humans, issues with privacy, intellectual property, and bias can become serious obstacles [38].

Synthetic data, however, provides an answer to all these problems, providing a potentially infinite dataset where ground-truth properties are easily accessible. In domains with many objects where labeling is impractical, scanning and simulating objects may not be [32, 19]. Furthermore, synthetic humans do not have any privacy or intellectual property concerns [38], and datasets can be balanced exactly with respect to sensitive attributes like race, gender, and other physical characteristics, minimizing the problems algorithms currently have with bias. Even better, simulations can be made interactive for training robotic policies.

Considering all the advantages, synthetic or artificial simulations should be dominantly for learning in various computer vision problems. One problem is that neural networks trained on synthetic data do not necessarily work on real data as well as methods trained directly on real data. Thus, even though such synthetic-to-real domain transfer has performed [56], the importance of synthetic data usage for training is still rarely

To overcome the occlusion problems and especially the size of the dataset and the high cost of annotating hands frame by frame, a recent work [58] uses a synthesized dataset which is annotated automatically. They use kinematic sensors which has multiple electromagnetic sensors (usually 6 6D sensors) connected to a hand. These sensors are connected to a receiver and a transmitter which generates the 3D hand pose automatically.

Although using synthesized dataset is easy to generate and annotate, they lack generalization power. As the image generated by this devices are computer generated, it will not work very well on real-world hand images. To overcome this issue they used a conditioned GAN [65] called GeoConGAN to transfer the computer generated images to real images. Also, to reach a better one-to-one relation between real and computer generated images, they applied a Cyclic-GAN [100] which has two parts of Real to Synthesized GAN (called real2synth) and Synthesized to Real GAN (called synth2real). Each of this GANs has its own generator and discriminator. Mueller *et al.* controlled the process with two losses; first converting synthesized image to real and calculating synthetic-to-real loss and again converting the result to synthesized image and calculating real-to-synthetic loss. They also randomly put some backgrounds behind the hands to make the images more realistic. Moreover, to create occlusion on the hands, they artificially put some objects in front of the hands to have some occluded frames in the dataset as well. They used ResNet [31] architecture for their feature extraction network to take advantage of residual blocks. Figure 12 shows the different steps of dataset production and hand pose estimation of Mueller *et* *al.*'s paper [56].

Spurr et al. [75] also used this cyclic concept for making a one-to-one relation between RGB image to 3D hand joints pose. They used GAN and Variational Autoencoder (VAE) to transfer the images to a latent space and then transfer it to the other domain. With that, they tried to map every RGB hand image to a 3D pose and use this map in the hand pose estimation task.

1.3 Effective Utilization of Synthetic Samples

Here, we propose two methods that develop convention approaches of utilizing synthetic dataset for effective synthetic-to-real domain generalization, resulting better 3D hand pose estimation performance against real images. First, in Chapter 4, we propose a generation methodof synthetic hand motions, not only to overcome the domain gap with continuous, but to shrink search space of hand pose estimation for a given input image, considering temporal feature of successive poses from streaming frames. With the hand motion dataset artificially created, we are able to extract temporal features and learn continuous estimations with streaming video frames. Secondly, we propose to

1.3.1 Synthetic Sequential RGB Dataset

Most of the RGB-based 3D hand pose estimation studies have performed frame-by-frame estimation based on independent static images. In this paper, we attempt to not only consider the appearance of a hand but incorporate the temporal movement information of a hand in motion into the learning framework, which leads to the necessity of a large-scale dataset with sequential RGB hand images. Conventional attemtps have built synthetic dataset that is generated for frame-

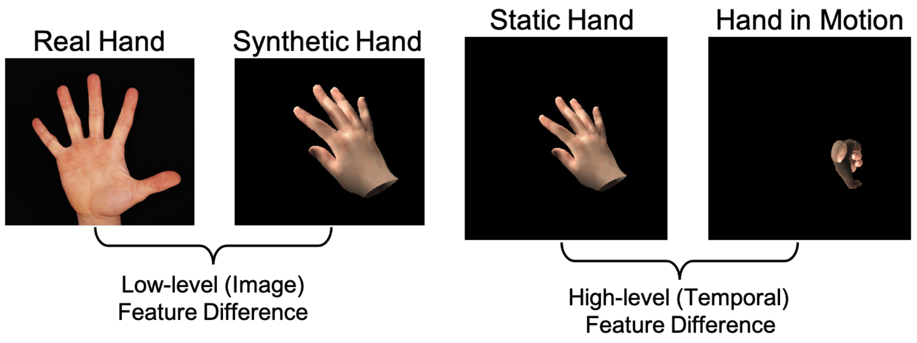


Figure 1.8: Illustration of synthetic-real feature gap and static-successive hand poses temporal feature difference

by-frame training, which have not made the literature to extract temporal information from the continuous estimations from streaming frames. We propose a novel method that generates a synthetic dataset that mimics natural human hand movements by re-engineering annotations of an extant static hand pose dataset into *pose-flows*. With the generated dataset, we train a newly proposed recurrent framework, exploiting visuo-temporal features from sequential synthetic hand images and emphasizing smoothness of estimations with temporal consistency constraints. Our novel training strategy of detaching the recurrent layer of the framework during domain finetuning from synthetic to real allows preservation of the visuo-temporal features learned from sequential synthetic hand images. Hand poses that are sequentially estimated consequently produce natural and smooth hand movements which lead to more robust estimations. Utilizing temporal information for 3D hand pose estimation significantly enhances general pose estimations by outperforming state-of-the-art methods in our experiments on hand pose estimation benchmarks.

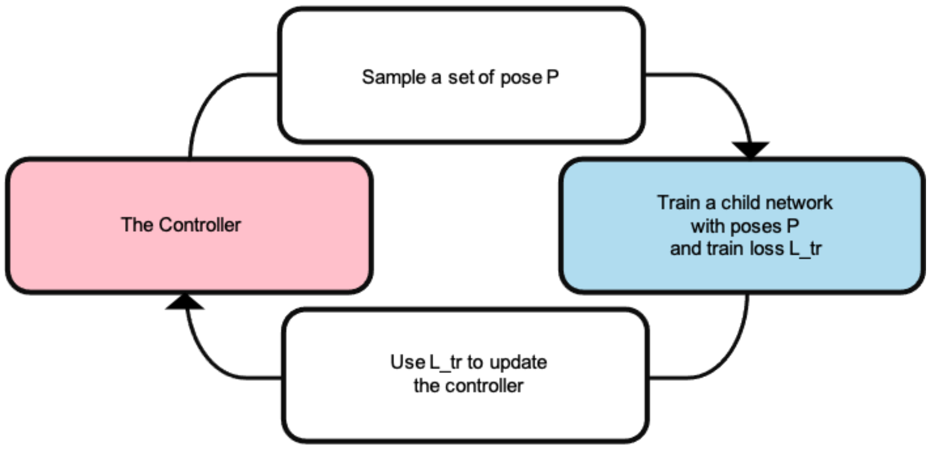

Sample a set of pose P
The Controller
Train a child network with poses P and train loss L_tr
Use L_tr to update the controller

Figure 1.9: A diagram that shows how hand pose automatic augmentation learning system cycles.

1.3.2 Hand Pose Automatic Feature-level Augmentation

For many years, dataset augmentation has been a standard regularization technique used to reduce overfitting while training supervised learning models. Data augmentation is particularly popular for visual recognition tasks as new data can be generated very easily by applying image manipulations such as shifting, scaling, rotation, and other affine transformations. When training LeNet5 [45], one of the most early and well-known convolutional neural network architectures, their work applied a series of transformations to the input images in order to improve the robustness of the model. An early work of deep learning [43] also used image transformations to generate new data when training the renowned AlexNet model for the 2012 Large Scale Visual Recognition Challenge (ILSVRC). They claimed that dataset augmentation reduced the error rate of the model by over 1%. Creating new data has since been a crucial component of all recent large-scale image recognition models. Unfortunately, dataset augmentation is not as straightforward to apply in all domains as it is for images. For example, a recent work [71] investigated a variety of data augmentation techniques for application to singing voice detection. These include adding Gaussian noise to the input, shifting the pitch of the audio signal, time stretching, varying the loudness of the audio signal, applying random frequency filters, and interpolating between samples in input space. They found that only pitch shifting and random frequency filtering appeared to improve model performance. While performing well on audio data, these augmentation techniques cannot be applied to other domains. As such, the process of designing, implementing, and evaluating new data augmentation techniques would need to be repeated for each new problem.

In this work, we consider augmentation not by a domain-specific transformation, but by perturbing, interpolating, or extrapolating between existing examples. However, we choose to operate not in input space, but in a learned feature space. Higher level representations are claimed to expand the relative volume of plausible data points within the feature space, conversely shrinking the space allocated for unlikely data points [5, 62]. As such, when traversing along the manifold it is more likely to encounter realistic samples in feature space than compared to input space. Unsupervised representation learning models offer a convenient way of learning useful feature spaces for exploring such transformations. Recently, there has been a return to interest in such techniques, leading to, e.g., variational autoencoders [41], generative adversarial networks [25], and generative stochastic networks [2], each of which could be used to generate useful feature spaces for augmentation.

By manipulating the vector representation of data within a learned feature space a dataset can be augmented in a number of ways. One of the most basic transformations that can be applied to the data is to simply add random noise to the context vector. In the context of class-imbalanced data, we believe that extrapolation between samples could also be applied. We investigate some of these methods to see which is most effective for improving the performance of supervised learning models when augmented data is added to the dataset.

We propose a novel feature-level auto-augmentation method that augments features that span a continuous space. Previous attempts of auto-augmentation are mainly consisted of discrete action space where each action corresponds to a augmentation policy (*e.g.* Shear-X translation, Brightness, Inversion ane etc.). Searches for augmentation policies in continuous space require for a policy network to have a far more delicate variations [48]. In terms of augmentation, we first emphasize on enlarging the training distribution of humand hand postures. This, however, allows robustness only against 3D pose estimations, and not necessarily in 2D estimations. Hands may appear variously within images depending on the illuminating condition, skin color, shades, hairs, wrinkles and more. A fixed set of training data limits its distribution of image space features. We thus utilize random images (not necessarily with hands in them) and synthesize them in a feature space in order to 'hallucinate' the network, so that general image encoding can be performed. As mentioned earlier in this section, augmentation in feature space is economical and efficient for training data-hungry neural networks. Enlarge the distribution of input images allow our network to generalize over various image inputs. For automation of the augmentation, we sample a hand pose vector from a pre-defined distribution that span a physically plausible human hand pose space. The distribution is trained to evolve to sample better pose hallucination features in an adversarial manner against the training loss.

Chapter 2

Related Works

2.1 RGB Synthetic Hand Pose Datasets

As mentioned earlier, RGB-based 3D hand pose estimation literature has difficulty of collecting scalable dataset due to challenges in annotations and privacy issues. Many hand pose estimation methods therefore have proposed, utilized and publicized their own synthetic datasets.

As an early approach, *SynthHands* [58] dataset is proposed. The dataset consists of total 63,530 images of synthetic hands, with which corresponding depth images are provided. The postures are collected based on two subjects, and the viewpoints toward the poses are in egocentric perspective. While the dataset was one of the first synthetic dataset proposed, the posture distribution within the dataset is limited based on two subjects in egocentric viewpoints. Figures 2.1, 2.2 and 2.3 depict examples of SynthHands data.

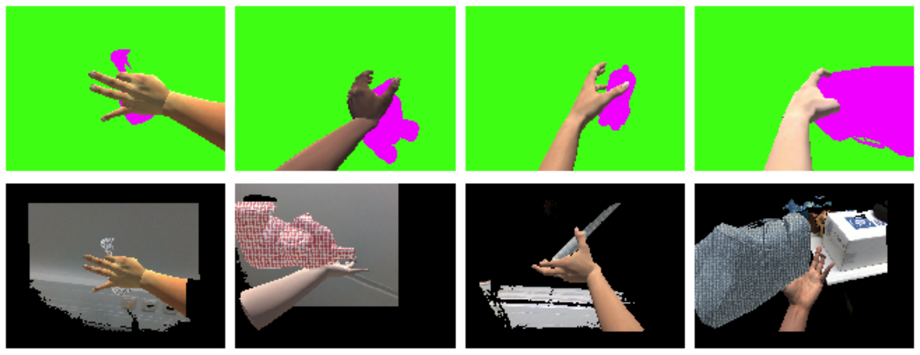


Figure 2.1: Examples of *SynthHands* data samples from [58].

Figure 2.2: Examples of *SynthHands* data samples from [58].

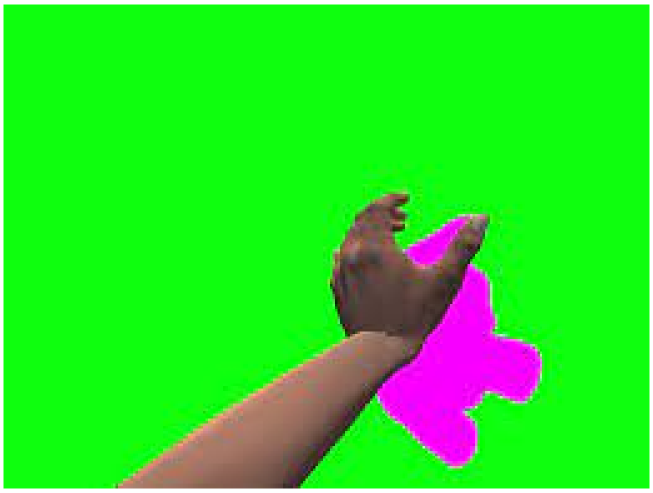


Figure 2.3: Examples of *SynthHands* data samples from [58].

Unlike SynthHands, *Rendered Handpose Dataset* (*RHD*) [102] consists of data samples in 3rd-person view. The dataset has 41K/2.7K training/testing image samples which is a smaller dataset compared to SynthHands, their postures are collected based on twenty different subjects. This allows the dataset to cover a larger distribution of hand pose space. However, as depicted in Figures 2.4 and 2.5, realistic image-level features are not present in the dataset.

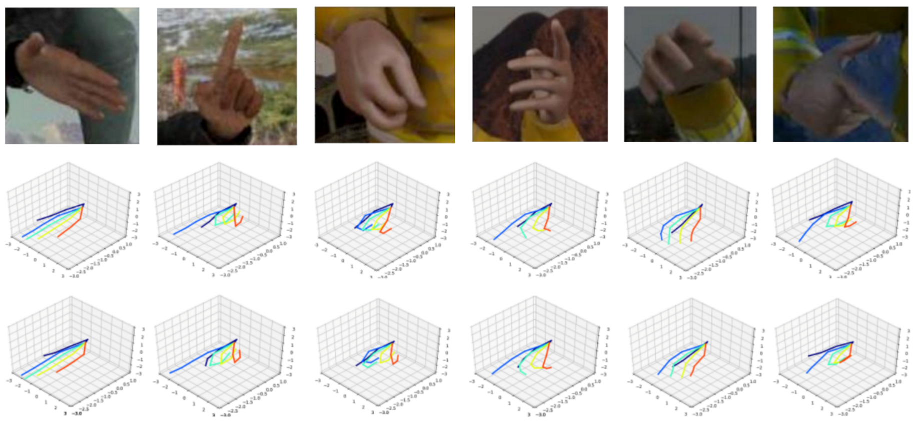


Figure 2.4: Examples of *Rendered Handpose Dataset* data samples from [102].

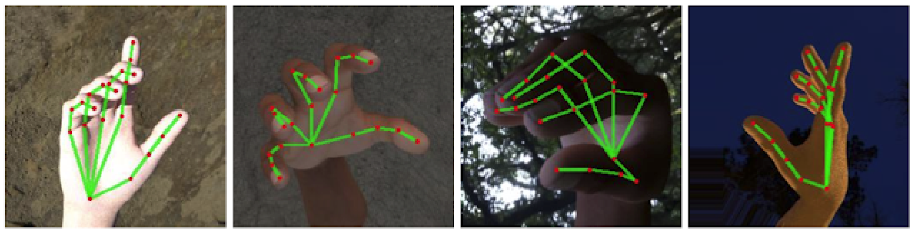


Figure 2.5: Examples of *Rendered Handpose Dataset* data samples from [102].

However, the generalization power of RGB synthetic hand dataset is notorious for cross-domain usage for real images samples. The encoded features based on learning from the synthetic datasets are known to be not robust against variations in skin texture and color along with various RGB background of real images. Since RGB real image samples with accurate annotations are difficult to obtain, as mentioned earlier, there have been attempts to augment/generate real images based on synthetic samples. *FreiHand* (FH) dataset is mainly composed of approximately 40,000 real hand image samples of which background is composited with chroma key setting [103]. Figures 2.6 and 2.7 illustrate their data samples. The background can then be replaceable with any random images, allowing various RGB versions to be generated with a limited set of hand samples. Although their final data adds up to 120,000 single hand images, the distribution of hand postures that are spanned by the data is limited. Their image features can, theoretically, be augmented in infinite versions.

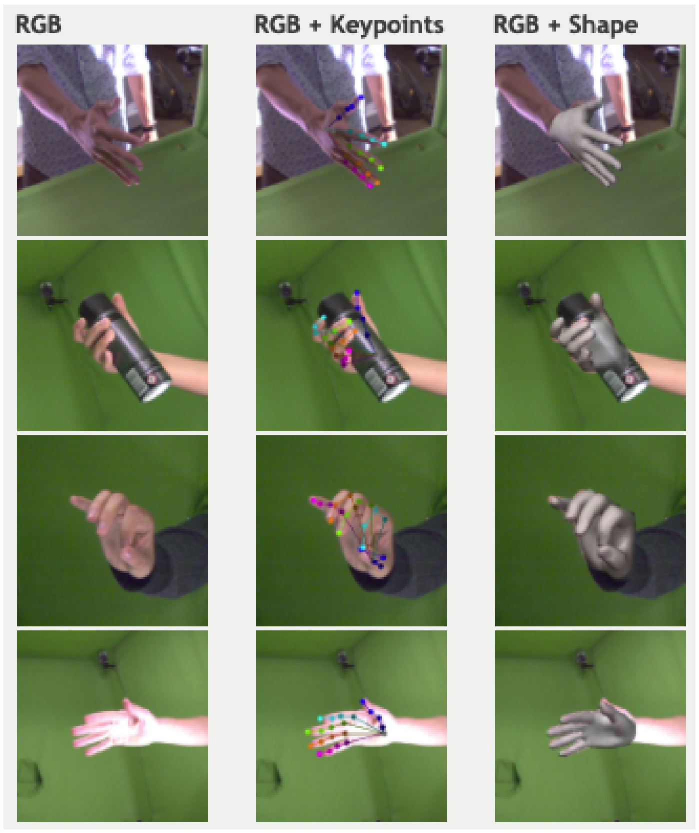


Figure 2.6: Examples of *FreiHand* data samples from [103].

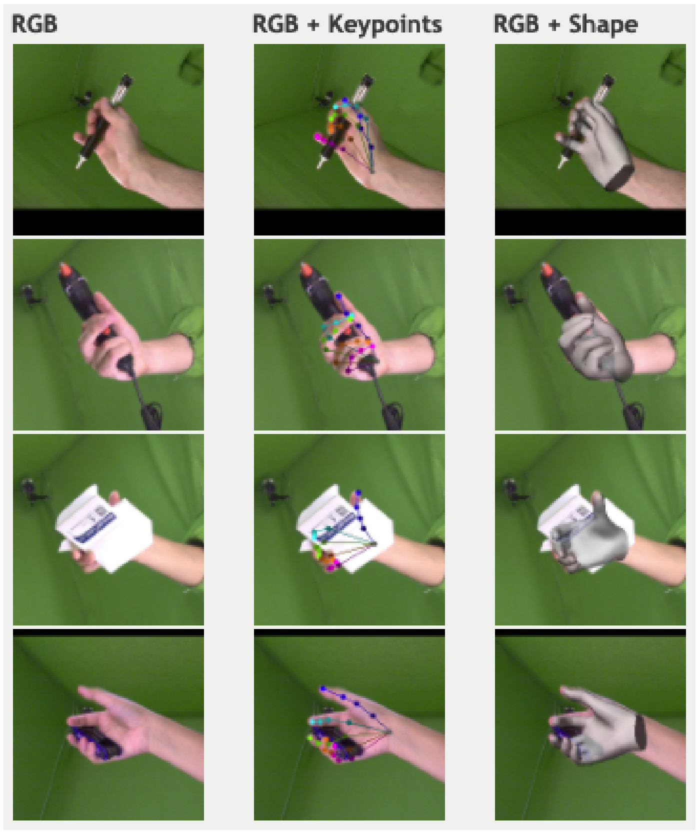


Figure 2.7: Examples of *FreiHand* data samples from [103].

Such effort is extended with style transfer from synthetic to real domain. The work [56] proposes an approach for the synthetic generation of training data that is based on a geometrically consistent image-to-image translation network. They use a neural network that translates synthetic images to real images, such that the so-generated images follow the same statistical distribution as real-world hand images. For training this translation network they combine an adversarial loss and a cycle-consistency loss with a geometric consistency loss in order to preserve geometric properties (such as hand pose) during translation.

Although using synthesized dataset is easy to generate and annotate, they lack generalization power. As the image generated by this devices are computer generated, it will not work very well on real-world hand images. To overcome this issue they used a conditioned GAN [65] called GeoConGAN to transfer the computer generated images to real images. Also, to reach a better one-to-one relation between real and computer generated images, they applied a Cyclic-GAN [100] which has two parts of Real to Synthesized GAN (called real2synth) and Synthesized to Real GAN (called synth2real). Each of this GANs has its own generator and discriminator. Mueller *et al.* controlled the process with two losses; first converting synthesized image to real and calculating synthetic-to-real loss and again converting the result to synthesized image and calculating real-to-synthetic loss. They also randomly put some backgrounds behind the hands to make the images more realistic. Moreover, to create occlusion on the hands, they artificially put some objects in front of the hands to have some occluded frames in the dataset as well. They used ResNet [31] architecture for their feature extraction network to take advantage of residual blocks. Figure 12 shows the different steps of dataset production and hand pose estimation of Mueller *et al.*'s paper [56]. The resultant images of their work are illustrated in Figures 2.8, 2.9 and 2.10

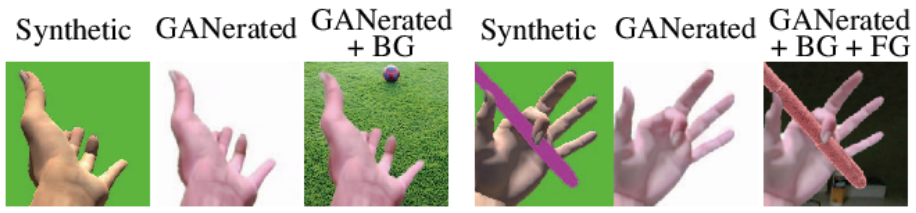


Figure 2.8: Comparisons data samples that are differently generated. The image is retrieved from [56].

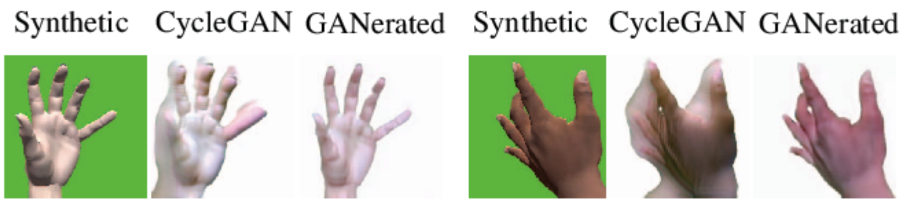


Figure 2.9: Comparisons data samples that are differently generated. The image is retrieved from [56].

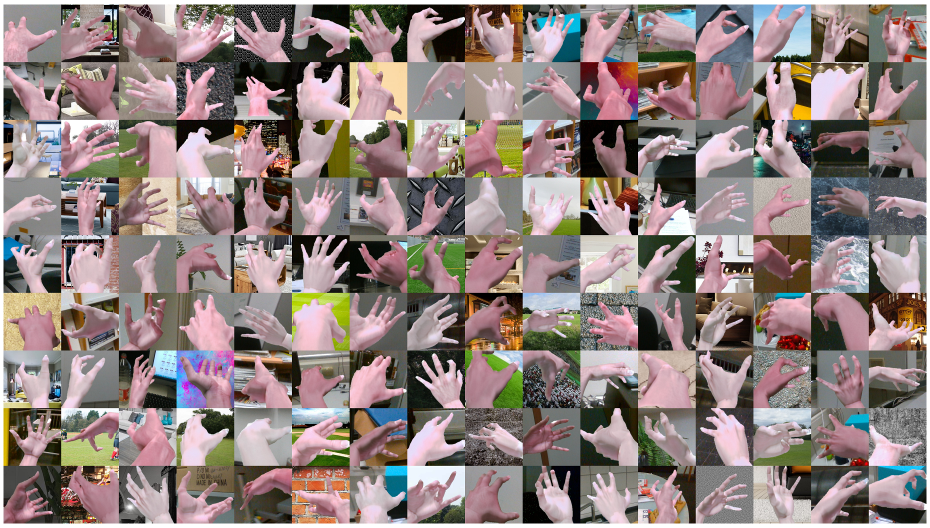


Figure 2.10: Examples of data samples generated by [56].

Spurr et al. [75] also used this cyclic concept for making a one-to-one relation between RGB image to 3D hand joints pose. They used GAN and Variational Autoencoder (VAE) to transfer the images to a latent space and then transfer it to the other domain. With that, they tried to map every RGB hand image to a 3D pose and use this map in the hand pose estimation task. The results of traverses of latent variables learned by the method is illustrated in Figure 2.11.

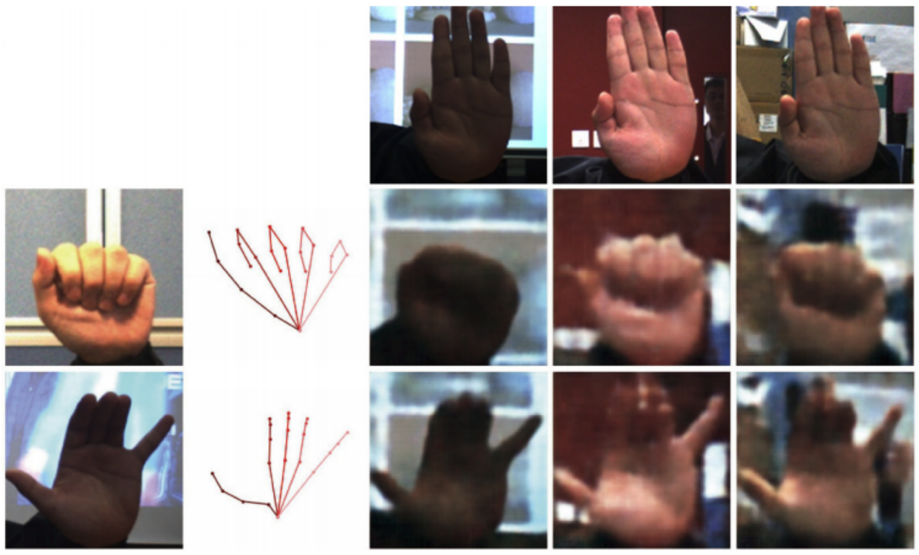


Figure 2.11: Examples of data samples generated by [75].

Hands are very likely to be occluded when captured in a 2D plane. This is mainly because of interactions with various objects that hands maneuver in mundane basis. This naturally brings up some theories that if object shapes can be approximated from the image, the hands that co-pose with the object can be more easily estimated in terms of their postures and shapes. So, recently, *ObMan* [28], a RGB synthetic dataset of hands that interact with daily objects, is proposed in order to solve such occlusion issues. While the dataset provides 150K images of hands in 3rd/egocentric view points along with annotations for vertices of 3D hand mesh estimations, it also provides annotations pose/shapes for objects. Many attempts recently are tried to tackle hand pose estimations when inputs are based on hands that are interacting with objects. Some examples of ObMan dataset are illustrated in Figures 2.12 and 2.13.

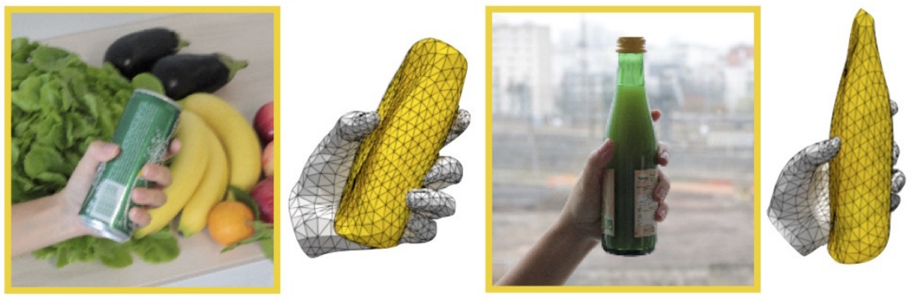


Figure 2.12: Examples of synthetic data samples generated by [28].

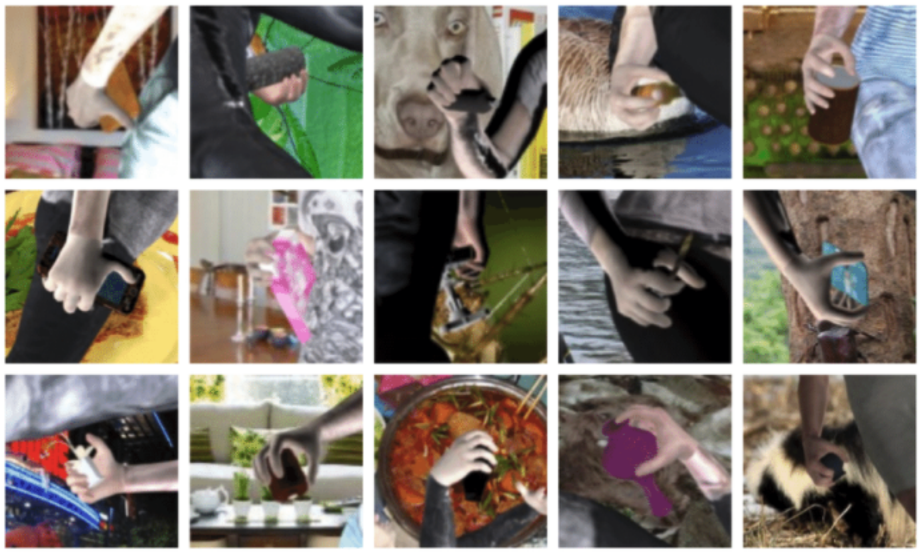


Figure 2.13: Examples of synthetic data samples generated by [28].

Our synthetic data generation is mainly motivated by the work of [6]. They implement MANO to generate synthetic samples based on MANO outputs and reproject them to 2D planes. Their synthetic dataset is mainly for pre-training their network before fine-tuning to real domain, which we later in Chapter 4 show that such finetuning is not working from synthetic to real without additional network structure constraints. Examples of their dataset is shown in Figure 2.14.

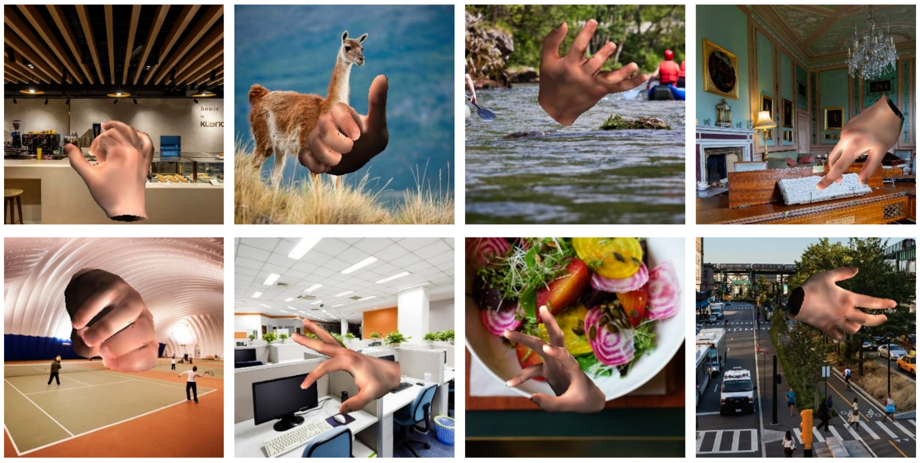


Figure 2.14: Examples of synthetic data samples generated by [6].

2.2 Use-Cases of Synthetic Data

2.2.1 Pretraining with Synthetics

It is common in the literature of 3D hand pose estimation to pretrain a network with a synthetic dataset and then finetune to real domain. This is due to the issue that RGB hand datasets are not large compared to other computer vision problems. In order to make methods to generalize over the given training set, the network needs to be able to learn the pose distribution of physically realistic postures in 3D space. To overcome the issue many methods pretrain their networks with synthetically generated samples. *SynthHands* dataset [58] is first introduced for this purpose. To reduce the learning space of complex networks such as ResNet50 [31], synthetic samples were naively generated with MANO

[69] and used for pretraining before finetuning to real domain [6, 102, 56]. The complexity is even more required when postures of objects that are interacting target hands are also considered to be estimated. This also leads more powerful set of synthetic samples. Recent works like [20, 39] utilize *ObMan* dataset for pretraining in order to reduce the search space of manifolds of human hand poses.

2.2.2 Simultaneous Feeding

Synthetic samples can also be fed along with real samples as concatenated samples in order to induce network training to learn general image semantic encoding that can be universally utilized for both synthetic and real [57]. Such approach is valid in only depth-based HPE tasks where domain gap from real to synthetic is not large compared to RGB images. HandAugment [98] also augments depth-based data because of its image-level domain gap between synthetic and real are small.

2.2.3 Generative Inference: Generate and Match Synthetic Templates

Instead of discriminative approach where methods directly regress 3D coordinates for each joint, there exist generative approaches in which methods try to rather match pre-defined template to current estimation. Conventional methods include fitting a pre-defined model, setting the system as an inverse kinematic problem [64]. Recently, MANO hand model [69, 29] is a synthetic mesh generative model that takes two low-dimensional parameters θ and β as inputs for controlling the pose and the shape, respectively, of the 3D hand mesh outputs. MANO innately constratints itself from estimating physically unrealistic hand postures as outputs unless very extreme values are inputted. After MANO is introduced, recent works thus have developed to estimate MANO parameters instead of 3D coordinates for each joints [6, 27, 26]. More details of MANO will be introduced in Chapter 3.

Chapter 3

Preliminaries: 3D Hand Mesh Model

3.1 MANO Hand Model

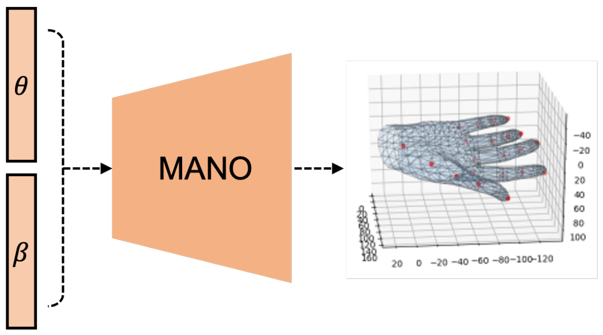


Figure 3.1: Illustration a MANO layer that takes two latent parameters, pose θ and shape β. MANO outputs 3D hand mesh model that is decided by the parameters.

As depicted in Figure 3.1, MANO hand model [29, 69] is a mesh deformation model that takes two low-dimensional parameters θ and β as inputs for controlling the pose and the shape, respectively, of the 3D hand mesh outputs. The general formulation of MANO model M, with a given mean template $\bar{T}$, is as follows:

$$M(\theta, \beta) = W(T_p(\theta, \beta), J(\beta), \theta, \omega), \tag{3.1}$$

$$T_p(\theta, \beta) = \bar{T} + B_p(\theta) + B_s(\beta) \tag{3.2}$$

where $J(\cdot)$ yields 3D joint locations using a kinematic tree The locations depend on shape parameters. These are learned as a sparse linear regression matrix $\mathcal{J}$ from mesh vertices. $W(\cdot)$ represents the linear blend skinning function that is applied with blend weights ω. $T_p(\cdot)$ defines the overall shape for the mesh model based on pre-defined deformation criteria with pose and shape While $B_s(\cdot)$ allows the base shape to vary with identity, $B_p(\cdot)$ captures deformations of the mesh as a function of the bending of the joints. The pose and shape blend shapes are defined as the linear combination of a set of deformations, i.e. vertex offsets:

$$B_p(\theta; \mathcal{P}) = \sum_{n=1}^{9K} (R_n(\theta) - R_n(\theta^*))\mathbf{P}_n, \tag{3.3}$$

$$B_s(\theta; \mathcal{S}) = \sum_{n=1}^{|\beta|} \beta_n \mathbf{S}_n. \tag{3.4}$$

Here $\mathbf{P}_n \in \mathcal{P}$ are the pose blend shapes and K is the number of parts in the hand model. $R_n(\theta)$ indexes into the n-th element of a vector The β_n are linear coefficients and the vectors $\mathbf{S}_n \in \mathcal{S}$ are principal components in a low-dimensional shape basis that are learned.

A PyTorch implementation of MANO modeling function with trained parameters $(\mathcal{S}, \mathcal{P}, \mathcal{J}, \bar{T}, \omega)$ is provided by [29]. MANO model takes up to 45-dimensional pose parameters θ and 10-dimensional shape parameters β while the original MANO framework uses 6-dimensional PCA (principal component analysis) subspace of θ for computational efficiency. We refer the original work of MANO [69] for more insights.

3.2 2D Reprojeciton of MANO Hands

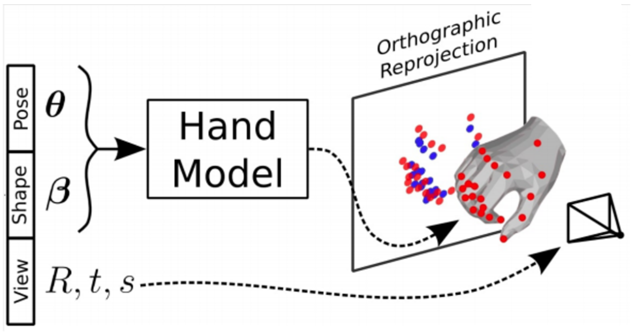


Figure 3.2: Orthographic reprojection 3D Mesh vertices and joints on a 2D plane based on rotation R, translation t and scale s factors.

After 3D estimations for mesh vertices $M(\theta, \beta)$ and joints $J(\theta, \beta)$ are computed by MANO model, in [6], the location of joints $J(\beta)$ can be globally rotated based on the pose θ, denoted as R_θ, to obtain a hand posture P with corresponding 3D coordinates of 21 joints:

$$J(\theta, \beta) = R_\theta(J(\beta)). \tag{3.5}$$

The 3D joint localization estimations can then be re-projected to 2D image plane with a weak-perspective camera model to acquire 2D estimations with a given rotation matrix $R \in SO(3)$, a translation $t \in \mathbb{R}^2$ and a scaling factor $s \in \mathbb{R}^+$:

$$M_{2D} = s\Pi RM(\theta, \beta) + t \tag{3.6}$$

$$J_{2D} = s\Pi RJ(\theta, \beta) + t \tag{3.7}$$

where Π represents orthographic projections. Hand mesh $M(\theta, \beta)$ is composed of 1,538 mesh faces and defined by 3D coordinates of 778 vertices, and joint locations $J(\theta, \beta)$ are represented by 3D coordinates of 21 joints. As illustrated in Figure 3.2, the re-projected 2D coordinates of M_{2D} and J_{2D} are represented in 2D locations in the image coordinates. MANO hand model can be utilized for both synthetic hand data generation and pose and shape estimator [6].

Chapter 4

SeqHAND: RGB-Sequence-Based 3D Hand Pose and Shape Estimation

4.1 Motivation

Since expressions of hands reflect much of human behavioral features in a daily basis, hand pose estimations are essential for many human-computer interactions, such as augmented reality (AR), virtual reality (VR) [36] and computer vision tasks that require gesture tracking [11]. Hand pose estimations conventionally struggle from an extensive space of pose articulations and occlusions including self-occlusions. Most recent 3D hand pose estimators that take sequential depth image frames as inputs have tried to enhance their performance considering temporal information of hand motions [33, 87, 59, 52]. Some works in the literature has tried to enhance estimation performance for RGB-only situations with stereo vision data, stereoscopic vision data is, however, expensive to acquire and easily distracted from high complexity within scenes. Motion context provides temporal features for narrower search space, hand personalizing, robustness to occlusion and refinement of estimations. We focus on the hand pose estimation considering its movements using only RGB image sequences for better inference of 3D spatial information.

Although the problem of estimating a hand pose in a single RGB image is an ill-posed problem, its performance is rapidly improving due to the development of various deep learning networks [6, 56, 23]. However, most studies have focused on accurately estimating 3D joint locations for each image without considering motion tendency. Pose of hands changes very quickly and in many cases contains more information on the movements of the successive poses than on the momentary ones. In addition, the current pose is greatly affected by the pose from the previous frames. Until now, there has been a lack of research on the estimation network considering the continuous changes of poses. The main reason that conventional RGB-based deep 3D hand pose estimators [6, 96, 3, 56] have only proposed frameworks with per-frame pose estimation approaches is that any large scale RGB sequential hand image dataset has not been available unlike the datasets with static images of hand poses. The diversity and the authenticity of hand motions along with generalization over skin colors, backgrounds and occlusions is a challenging factor for a dataset to be assured.

In this section, we present a novel perspective on hand pose and shape estimation tasks and propose to consider temporal movements of hands as well as their appearances for more accurate 3D estimations of hand poses based on RGB image inputs. In order to train a framework that exploits visuo-temporal features to manage successive hand pose images, we are required to have sufficient pose data samples that are sequentially correlated. We thus propose a new generation method of dataset, SeqHAND dataset, with sequential synthetic RGB images of natural hand movements, re-enginerring extant static hand pose annotations of BigHand2.2M dataset [93]. To effectively test our generated dataset, we extend the framework of [6] with a recurrent layer based on empirical validity of its structure. Also since it is widely accepted that models trained with synthetic images perform poorly on real images [56], we present a new training pipeline to preserve pre-trained image-level temporal mapping during synthetic-real domain transition. Our contributions to this end are as follows :

- We design a new generation method for sequential RGB image dataset with realistic hand motions that allows 3D hand pose and shape estimators to learn the dynamics of hand pose variations (See Figure 4.1) by proposing a pose-flow generation procedure.
- We propose a new recurrent framework with convolution-LSTM layer to directly exploit visuo-temporal information from hand pose and shape variations in image space and map to 3D space.
- We present a novel training pipeline of preserving extracted spatio-temporal features from sequential RGB hand images during domain finetuning from synthetic to real.
- Our approach achieves not only state-of-the-art performance in standard 3D hand pose estimation dataset benchmarks, but also smooth human-like 3D pose fittings for the image sequences.

To the best of our knowledge, we propose the first deep-learning based 3D hand pose and shape estimator without any external 2D pose estimator that exploits temporal information directly from sequential RGB images.

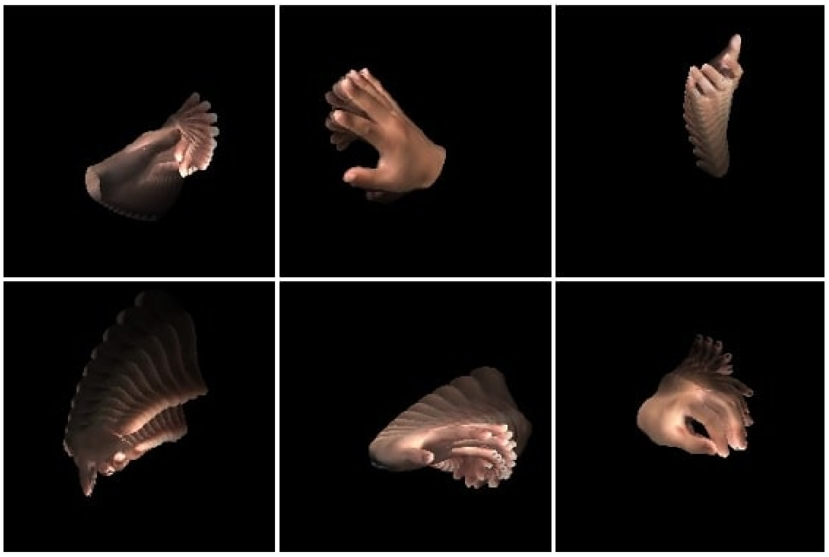


Figure 4.1: Illustrations of sequential 2D images of hand pose-flows that are generated by the proposed method. Hand poses in each frame are ensured to be physically feasible.

4.2 Related Works

Many approaches of hand pose estimation (HPE) have been actively studied. To acquire hand information, the literature of single hand 3D pose estimation has been mainly based on visual inputs of depth sensors and/or RGB cameras.

4.2.1 Per-frame RGB-based 3D HPE

As views of a single 3D scene in multiple perspectives are correlated, efforts of 3D estimation based on multiple RGB images of a hand have also been introduced [79, 73, 24, 61, 17]. Multi-view camera setups allow refinements against occlusions, segmentation enhancements and better sense of depth. In the work of [73], bootstrapping pose estimations among images from multiple perspectives help the estimator to retrain badly annotated data samples and refine against occlusions. A pair of stereo images provides similar effects in a more limited setting. Integration of paired stereo images has yielded better 3D hand pose estimations through manipulations of disparity between paired images [63, 70, 95, 67].

By providing both color and depth information, the RGB-D inputs in a hybrid input type overcome limitations of depth-only images such as occlusions. Additional color information presents pixel-level clusters based on color similarities and light contrasts which enables more robustness to occlusions occurring even in egocentric views [53, 58, 99, 60, 77]. In [92], after a depth-based model is trained with abundant depth image data, an RGB-based 3D pose estimation framework is trained through transfer learning with privileged high-level features learned from the depth-only setting. Cai et al. [7] is trained in a weakly-supervised manner with a depth regularizer which is trained through depth map matching on RGB-only estimations.

Monocular RGB-only setup is even more challenging because it only provides visual 2D vision of hand poses. With deep learning methods that have allowed successful achievements of hand detection [31, 44], deep pose estimators have recently been able to concentrate on per-frame hand 3D pose estimation problems [102]. To overcome the lack of 3D spatial information from the 2D inputs, there are needs of constraints and guidance to infer 3D hand postures [64]. Most recently, works of [6, 96, 3] employ a prior hand model of MANO [69] and have achieved significant performance improvement in the RGB-only setup.

4.2.2 Exploitation of Temporal Information in 3D HPE

Considering temporal features of depth maps, sequential data of hand pose depth images [95, 93, 58, 59] have been trained with hand pose estimators. The temporal features of hand pose variations are used for encoding temporal variations of hand poses with recurrent structure of a model [33, 87], modeling of hand shape space [40], and refinement of current estimations [59, 52]. With sequential monocular RGB-D inputs, Taylor et al. [83] optimize surface hand shape models, updating subdivision surfaces on corresponding 3D hand geometric models. Temporal feature exploitation has not been done for deep-learning based 3D hand pose estimators that take color images as inputs because large scale sequential RGB hand pose datasets have not been available in the literature. We share the essential motivation with the work of [8], but believe that, even without the assistance of 2D pose estimation results, sequential RGB images provide sufficient temporal information and spatial constraints for better 3D hand pose inference with robustness to occlusions.

4.2.3 Generated Synthetic Hand Data

Since RGB images also consist of background noise and color diversity of hands that distract pose estimations, synthetic RGB data samples are generated from the hand model to incite the robustness of models [7, 6, 23, 58]. In [75, 89], cross-modal data is embedded in a latent space, which allows 3D pose labeling of unlabeled samples generated from (disentangled) latent factor traverses. Mueller et al. [56] had applied cycleGAN [101] for realistic appearances of generated synthetic samples to reduce the synthetic-real domain gap. While there have been recent attempts to solve an issue of lacking reliable RGB datasets

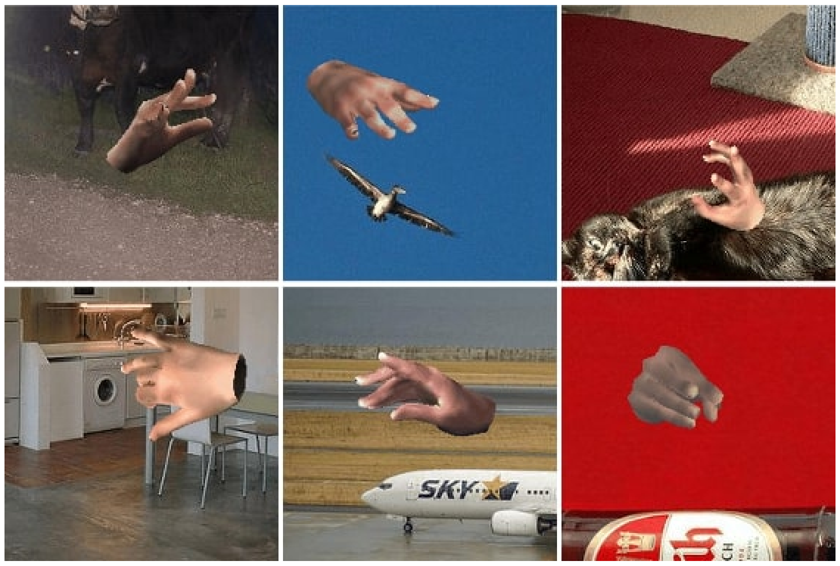


Figure 4.2: Each frame of sequential hand motion videos is composed of varying poses and moving backgrounds.

through generations of hand images [6, 102, 56, 7, 75], most of the works have focused on generation of realistic appearances of hands that are not in motions. To strictly imitate human perception of hand poses, it is critical for RGB-based hand pose estimators to understand the dynamics of pose variations in a spatio-temporal space. We further consider that synthetic hand pose dataset in realistic motions provides efficient information for pose estimations as much as appearances.

4.3 Generation of SeqHAND Dataset

Although the potential of temporal features have been shown promising results for 3D HPE tasks [8, 59, 83], large scale RGB sequential hand image datasets have not been available during recent years in the literature of RGB-based 3D

HPE. In this section, we propose a generation method to create short clip samples of hand motions that consist of sequential RGB frames of synthetic hands. In this section, we describe a new generation method of hand motions that consist of sequential RGB frames of synthetic hands.

To generate sequential RGB image data with human-like hand motions, all poses during the variation from an initial pose to a final pose need to be realistic. We thus utilize BigHand2.2M (BH) [93] for sequential hand motion image dataset generation. BH dataset consists of 2.2 million pose samples with 3D annotations for joint locations acquired from 2 hour-long hand motions collected from 10 real subjects. With BH datasets, the generated samples are expected to inherit the manifold of its real human hand articulation space and kinematics of real hand postures. As 3D mapping of BH samples using t-SNE [51] in Figure 4.3 shows, BH is known that the pose samples are densely and widely collected. Such density of BH dataset with a more complete range of variation is considered sufficient for various pose generations.

We firstly define a *pose-flow*, a set of poses at each time step during the variation. Putting gradually changing poses in a sequential manner, we newly propose a *pose-flow* generation method. For each pose-flow generation, an initial and a final poses, $P_{initial}$ and P_{final}, are independently and randomly selected from BH dataset. While varying from the initial to the final pose during n frames, the coordinates of joints are updated by α/n of the difference between the current coordinates and the ones of the final pose.[1] The update size α is empirically chosen for the desirable speed of pose variations. A pose P_i^{BH} from BH dataset that is the nearest to the updated pose in terms of Euclidean distance

[1]Note that direct random samplings from continuous pose parameter space $\theta \in \mathbb{R}$ does not assure diversity and authenticity of poses [69].

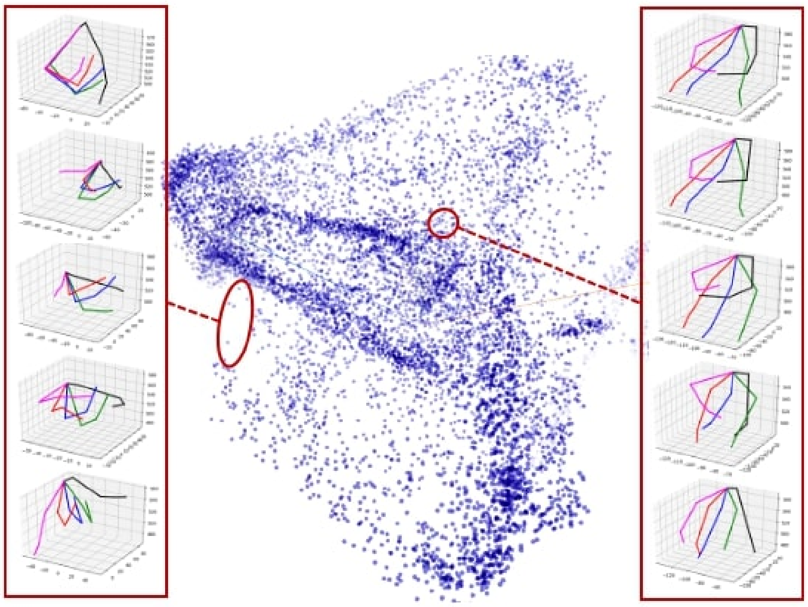


Figure 4.3: 3D t-SNE visualization of 10,000 of BH data samples randomly selected. BH dataset completes a pose space that covers previously reported datasets, having a dense pool of related neighboring poses.

is then newly selected as the current pose for the k-th frame:

$$P_0 = P^{BH}_{initial} \tag{4.1}$$

$$P_{updated} = P_{k-1} - \frac{\alpha}{n}(P_{k-1} - P^{BH}_{final}) \tag{4.2}$$

$$P_k = P^{BH}_i \quad \text{s.t.} \quad \min_i ||P_{updated} - P^{BH}_i||. \tag{4.3}$$

The overall procedure of the Pose-flow generation is summarized in Figure 4.4. The intermediate pose ($P_{updated}$) is calculated as stochastic update. Such stochasticity of our pose updates helps avoiding strict updates of pose gradients and encourages wandering more within the pose space. This is intended to lower the frequency of the mean pose and encourage various trajectories from a pose to

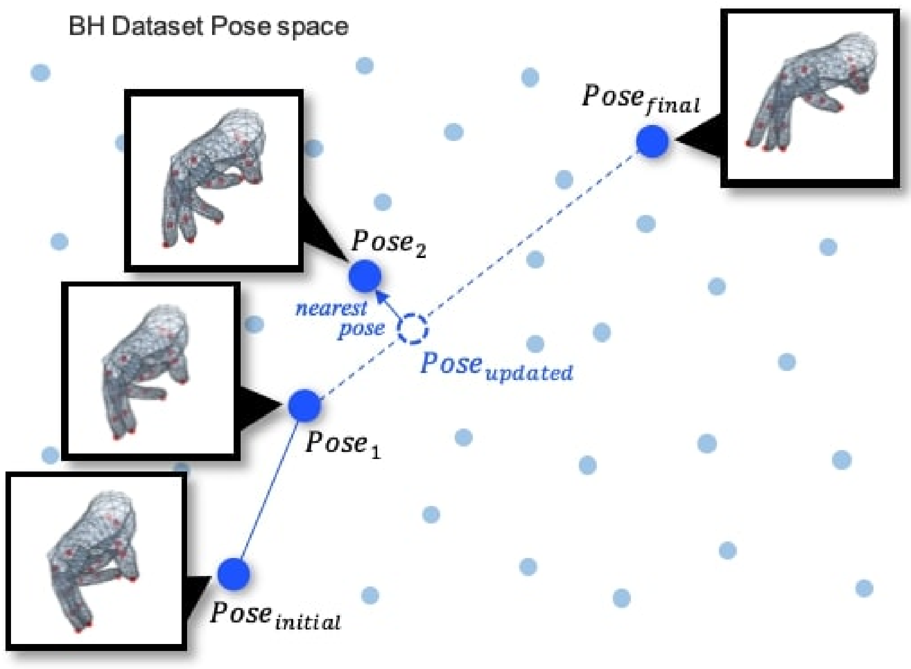


Figure 4.4: An illustration of the pose-flow generation procedure. All poses per pose-flow are selected from the annotations of BH dataset. At each frame, a current pose is updated by the difference between the previous pose and the final pose. The pose nearest to the updated pose is then selected for the frame.

another. Pose selections from the BH annotations, again, allows assurance on the authencity of hand poses during the variation. The procedure of the Pose-flow generation is illustrated in Fig 4.4.

To generate RGB images for a pose, a shallow four-layer network with which takes inputs of 3D coordinates for joints of BH annotations is trained to output corresponding pose parameters θ for MANO hand model. For each pose at a frame, we feed corresponding 21 joint location coordinates to the this network to acquire a hand mesh model in the desired pose, which is then re-projected to an image plane. As done in [6], we assign each vertex in a mesh the RGB value of predefined color templates of hands to create appearances of hands. Sampled hand shape parameter $\beta \in [-2, 2]^{10}$ and selected color template are set unchanged along per flow. Camera parameters of rotation R, scale s and translation t factors are independently sampled for initial and final poses and updated at each frame in the same way as the poses are. All frames are in the size of w and h. Figure 4.1 depicts illustrations of our generated pose-flows.

Further mimicking images of hand motions in the wild, we sample two (initial and ending) random patches from VOC2012 data [21] with the size of w and h and move the location of the patch for backgrounds along the frames. As Table 4.1 denotes, the generated SeqHAND dataset provides not only both 3^{rd}-person and egocentric viewpoints of hand postures but also sequential RGB images of hand poses that firstly allow data-hungry neural networks to exploit visuo-temporal features directly from RGB inputs. [2]

4.4 SeqHand-Net for Visuo-Temporal Feature Exploitation

With SeqHAND dataset, we are able to overcome the scarcity of sequential RGB dataset which limits conventional RGB-based 3D HPE methods from exploiting temporal image features. The overall hand pose estimation scenario in our problem scope is conducted with an exterior hand detector that localizes a hand from streaming frames. Motivated by [6], we design sequential hand pose and shape estimation network (SeqHAND-Net). On top of the encoder network of [6], we incorporate convolution-LSTM (ConvLSTM) layer [88] to capture se-

[2]Although we can generate as many synthetic data as we want, our SeqHand dataset contains 400K/10K samples used for training/validation.

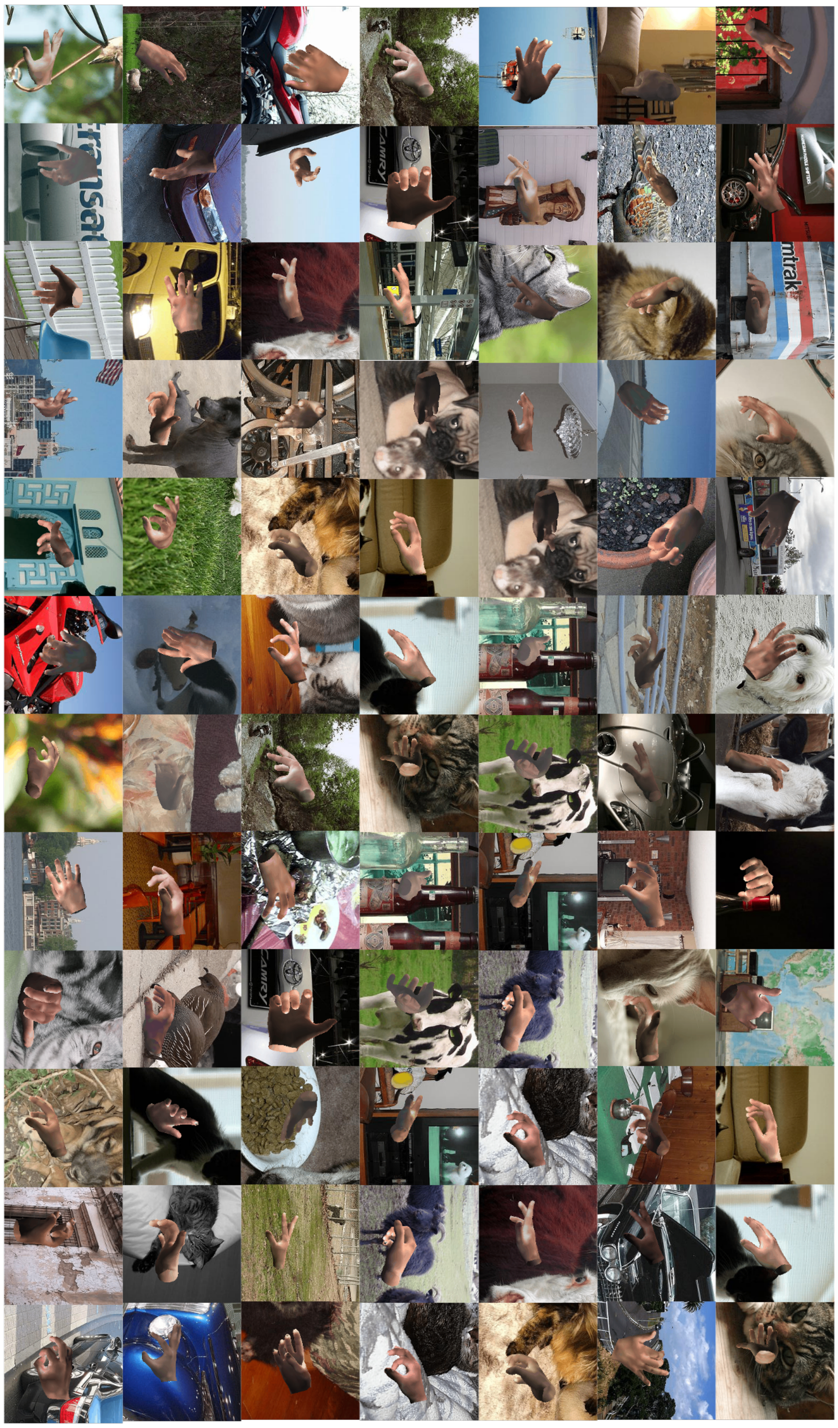


Figure 4.5: Examples of SeqHAND data

Table 4.1: Among the contemporary 3D hand pose datasets, SeqHAND dataset is the first dataset for 3D hand pose estimations that provides sequential RGB hand image frames along with stable annotations in both 3^{rd}-person and egocentric perspectives.

Datasets	RGB/Depth	Real/Synth	Static/Sequential	3^{rd}/Ego view	# of frames
SynthHands[58]	RGB+Depth	Synth	Static	Ego	63k
RHD[102]	RGB+Depth	Synth	Static	3^{rd}	43.7k
FPHA[22]	RGB+Depth	Real	Sequential	Ego	100k
NYU [85]	Depth	Real	Sequential	3^{rd}	80k
ICVL [82]	Depth	Real	Sequential	3^{rd}	332.5k
MSRA15 [80]	Depth	Real	Sequential	3^{rd}	76,375
MSRC [72]	Depth	Synth	Sequential	3^{rd}+Ego	100k
SynHand5M [54]	Depth	Synth	Sequential	3^{rd}	5M
GANerated [56]	RGB	Synth	Static	Ego	330k
SeqHAND (Ours)	RGB	Synth	Sequential	3^{rd}+Ego	410k

quential relationship between consecutive hand poses. With SeqHAND dataset, our model is able to effectively exploit high-level visuo-temporal features in spite of such simple extension with a recurrent layer. Our method does not consider additional hand 2D joint locations as inputs, and purely performs 3D hand pose estimation based on sequentially streaming RGB images in an effort to overcome the dependency on external 2D pose estimators. We also propose, in this section, a training pipeline for domain adaptation from synthetic to real, adapting low-level features with real hand images while preserving high-level visuo-temporal features of hand motions.

From each frame, a cropped hand image is fed into SeqHAND-Net as illustrated in Figure 4.6. Our problem scope is to better perform hand pose estimations on streaming cropped frames that are unseen by the estimator. The encoder of our SeqHAND-Net has the backbone structure of ResNet-50[31] and, for training, expects sequential inputs with k frames. A single ConvLSTM is implemented right before the last layer as a recurrent visual feature extractor so that the dynamics of hand motions are embedded in the highest-level latent space. Learning of hand motion sequential dynamics in the high-level space is important since low-level visual features are changed with the ConvLSTM layer fixed during finetuning for real hand images. After the recurrent layer, a simple linear mapping layer from hidden features to the output vector is set. The encoder's resultant vector consists of parameters for pose $\theta \in \mathbb{R}^{10}$, shape $\beta \in \mathbb{R}^{10}$, scale $s \in \mathbb{R}^{+}$, translation $t \in \mathbb{R}^{2}$ and rotation $r \in \mathbb{R}^{3}$ which turns into a matrix $R \in SO(3)$ through Rodrigues rotation formula for Eqs (3.6) and (3.7).

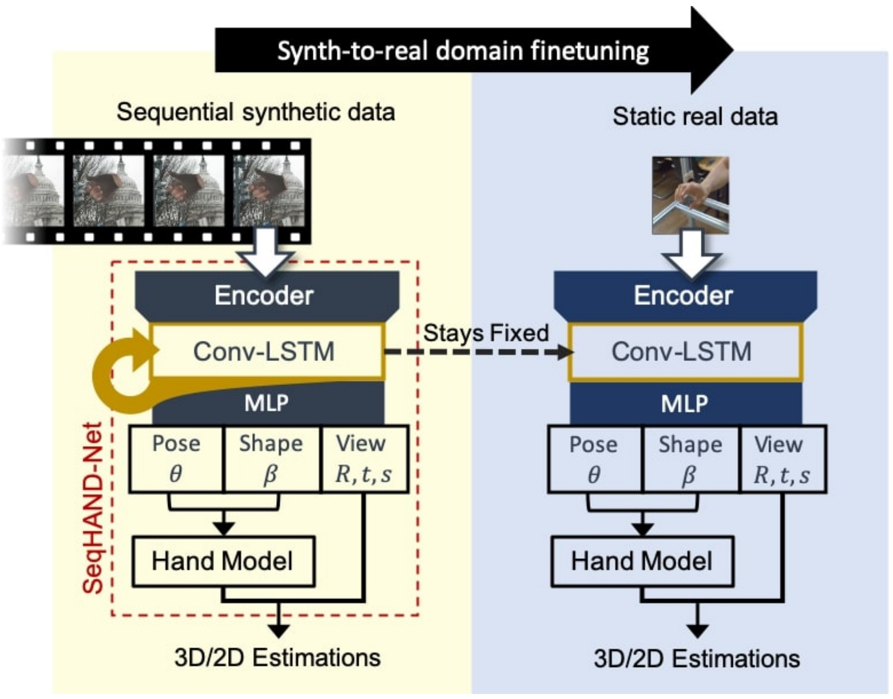


Figure 4.6: Our training strategy for preservation of temporal features during domain adaptation synth-to-real. SeqHAND-Net is created from [6] with an extra visuo-temporal feature exploitation layer for sequential RGB inputs. During finetuning, SeqHAND-Net considers a static real data as 1-frame-long sequence. The temporal high-level feature encoding is preserved while low-level image feature encoding layers are finetuned.

4.4.1 Synth-to-Real Domain Transfer with Preservation of Temporal Features

Since there are no large scale dataset with sequential images of real hands with reliable 3D joint labels, we first train the encoder with SeqHAND dataset for the encoder to understand natural continuity of hand motion dynamics. Then, while the ConvLSTM layers are fixed, the encoder is fine-tuned with real hand image

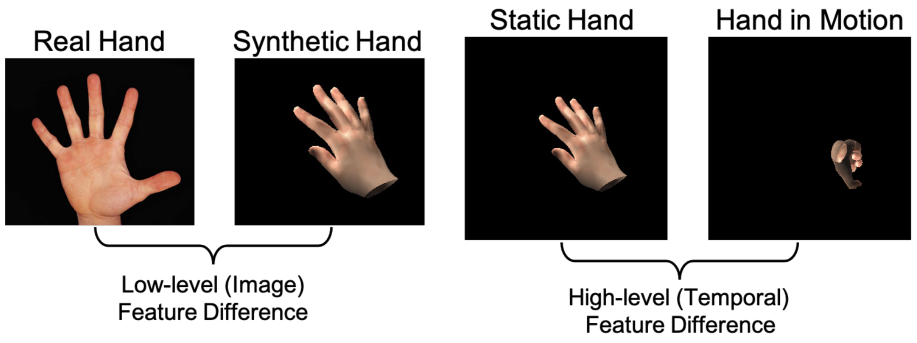

Real Hand
Synthetic Hand
Static Hand
Hand in Motion
Low-level (Image) Feature Difference
High-level (Temporal) Feature Difference

Figure 4.7: Illustration of synthetic-real feature gap and static-successive hand poses temporal feature difference

datasets to reduce the real-synthetic domain gap. As mentioned earlier, many recent researches have used synthetic hand images for pre-training and finetuned into real domain to overcome the scarcity of real hand images. While finetuning into real domain may allow faster training convergence, further training with a smaller dataset not only causes overfitting and may result in catastrophic forgetting [42]. To preserve visuo-temporal features learned from synthetic hand motions of SeqHAND dataset, we exclude the ConvLSTM layer of SeqHAND-Net from domain transfer to real hand images, allowing the network to only finetune low-level image features. Only the 'Encoder' and 'MLP' layers from Figure 4.6 are finetuned with a real static hand image dataset (e.g. FreiHand [103]). SeqHAND-Net is therefore trained, considering each image sample as 1-frame-long sequential image during domain transition to real.

4.4.2 Training Objectives

The followings are the types of criteria used for training our proposed framework to consider visuo-temporal features and emphasize the temporal smoothness of estimations :

2D joint regression loss

The re-projected 2D joint loss is represented as :

$$L_{2D}^{J} = ||J_{2D} - x_{2D}^{J}||_1, \tag{4.4}$$

where x_{2D} represents the ground-truth 2D locations of hand joints within a frame image. We have used the L1 loss because of inaccuracies in annotations in the training datasets.

3D joint regression loss

The ground-truth joint locations and the ones predicted are regressed to be the same using the following loss:

$$L_{3D}^{J} = ||RJ(\theta, \beta) - x_{3D}^{J}||_2^2, \tag{4.5}$$

where x_{3D}^{J} represents ground-truth 3D joint coordinates. If a dataset provides ground-truth coordinates of 3D vertex points (e.g. FreiHand dataset), the 3D coordinates of each vertex predicted and the ones of ground-truth is minimized as done for 3D joint loss, based on the following loss:

$$L_{3D}^{M} = ||RM(\theta, \beta) - x_{3D}^{M}||_2^2 \tag{4.6}$$

where x_{3D}^{M} represents ground-truth 3D mesh vertex coordinates.

Hand mask fitting loss

The hand mask loss is proposed in [**?**] to fit the shape and pose predictions in the binary mask of hands in the image plane. This loss ensures predicted coordinates of mesh vertices to be inside of a hand region when re-projected:

$$L_{mask} = 1 - \frac{1}{N}\sum_i H(M_{2D}^i), \quad H(x) = \begin{cases} 1, & \text{if } x \text{ inside a hand region.} \\ 0, & \text{otherwise.} \end{cases} \tag{4.7}$$

where H is a hand mask indicator function that tells if vertex point x is inside the hand region or not. The loss represents the percentage of vertices that are outside the region. See Figure 4.8 for imagery understanding of the hand mask fitting loss.

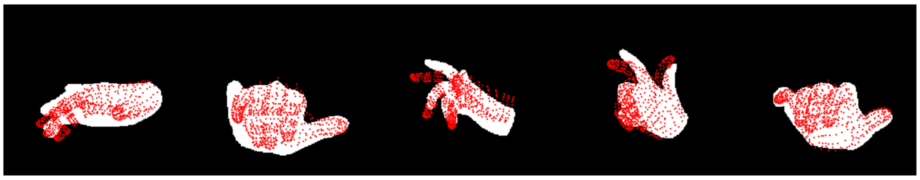


Figure 4.8: Red points represent re-projected 2D vertices of 3D mesh models. Hand mask fitting loss is calculated with the number of 2D vertices outside of the white area of hand mask images.

Temporal consistency loss

For pre-training on SeqHAND dataset, our method needs to be constrained with temporal consistency to ensure smoothness of pose and shape predictions. Similar to [8], we have adopted the temporal consistency loss for smoothness of temporal variation of poses:

$$L_{temp} = ||\beta_{t-1} - \beta_t||_2^2 + \lambda_{temp}^{\theta}||\theta_{t-1} - \theta_t||_2^2. \tag{4.8}$$

Considering the fact that all hands in a sequence is the same hand for all the frames, we have set the constraint hyper-parameter λ_{temp}^{θ} a comparably small number as 0.01 so that temporal variation of hand shapes per image sequence data to be low while pose variation is less constrained but is still assured of temporal smoothness. While finely penalizing current estimations with the previous ones, this loss allows the reduction of search space and natural 3D hand motion estimations.

Camera parameter regression loss

During training with SeqHAND dataset where all ground-truths for pose, shape and viewpoint parameters $\{\theta, \beta, r, t, s\}$ are available, our model is trained with L2-norm loss between predictions and the ground-truth.

$$L_{cam} = \sum_{i \in \{\theta,\beta,r,t,s\}} ||\hat{i} - i||_2^2 \tag{4.9}$$

where $\hat{i}$ and i respectively refer to predicted and ground-truth parameters for pose, shape and viewpoint.

4.5 Experiments

Datasets for Training

For visuo-temporal feature encodings of sequential RGB hand images, we pretrain SeqHAND-Net with our SeqHAND dataset. We have generated 40,000 sequences for training and 1,000 for validate samples each of which is 10-frames-long. All images are generated in the size of 224×224 fitting ResNet input size. SeqHAND data samples are exemplified in Figure 4.1 and 4.2.

To finetune SeqHAND-Net for synthetic-real domain gap reduction, we have used STB (Stereo Hand Pose Tracking Benchmark) [95] and FR (FreiHand) [103] datasets. STB dataset consists of real hand images captured in a sequential manner during 18,000 frames with 6 different lighting conditions and backgrounds. Each frame image is labeled with 2D and 3D annotations of 21 joints. Since STB dataset has annotations for joint locations of palm centers instead of wrist, we have interpolated related mesh vertices of MANO hand model to mach the annotation of STB dataset. The dataset is divided into training and testing sets as done in [6].

FR dataset has 130,240 data samples that are made up of 32,560 non-sequential real hand images with four different backgrounds. Since the dataset has hands that are centered within the image planes, we have modified each sample by repositioning the hand randomly within the image for more robust training results. FR dataset provides MANO-friendly annotations of 21 joint 3D/2D locations along with 778 vertex ground-truth 2D/3D coordinates with hand masks.

We have finetuned the SeqHAND-Net pretrained on SeqHAND dataset with real-hand image datasets mentioned above in a non-sequential manner while conserving hand motion dynamic features detached from further learning.

Datasets for Evaluation

We evaluate various framework structures that consider temporal features on the validation set of SeqHAND dataset for the logical framework choice. During training with real hand images, we have used our synthetic image sequences with various different backgrounds. For the comparison against other state-of-the-art methods, we have selected standard hand pose estimation datasets of the splitted test set of STB, EgoDexter (ED) [58] and Dexter+Obeject (DO) [78] in which there exists temporal relations among data samples since our network requires sequential RGB inputs for fair comparisons. While STB and DO datasets consist of real hand images in 3^{rd}-person viewpoints, ED dataset has samples that are in egocentric perspective. For all datasets, our method is evaluated on every frame of input sequences.

We evaluate our method on STB dataset with pose estimations per frame while, for ED and DO datasets, the dataset is re-designed so that every frame with 3D annotations would be the last frame of 10 frame-long image sequence. Our method is therefore evaluated at the last frame of every input sequence. This is because our network structure is able to perform in both cases, and all image frames of ED and dataset is not annotated and skipped. The first image sample for ED and DO datasets are repeated to make 10-frame long sequential input data.

Metrics

For evaluation results, we measure the percentage of correct key-points for 3D joint locations (3D-PCK) along with the area under the curve (AUC) of various thresholds. The PCK measurement that quantifies the fraction of correct predictions within an error threshold τ [90]. We measure each individual joint respectively and took their average as an overall metric. Using different τ values, we yield a PCK curve. Therefore, the Area Under Curve (AUC) can be obtained as a holistic measurement across different decision thresholds. In addition, we provide average Euclidean distance error for all 2D/3D joint key-points so that more absolute comparisons can be made.

Hand Localizations

or all experiments, we have used MobileNet+SSD version of hand detection implementation [44] trained with a hand segmentation dataset [4] for providing sequential cropped hand images to SeqHAND-Net. For localized hands with tight bounding rectangular boxes, we choose the longer edge with a length size

l and crop the region based on the center point of boxes so that the cropped images have a square ratio with width and height size of $2.2 * l$, as done in [6].

Table 4.2: Ablation study results of various structures of the framework proposed in [6] for sequential inputs.

Frameworks	AUC		Error (px/mm)		# params
	2D	3D	2D	3D	
ResNet50-Encoder (baseline) [6]	0.855	0.979	3.44	7.85	28.8M
ResNet101-Encoder [6]	0.861	0.981	3.31	7.54	47.8M
I3D-Encoder [9]	0.831	0.967	4.19	9.24	31.5M
MFNet-Encoder [46]	0.818	0.912	5.48	10.54	41.7M
ResNet50-Encoder+LSTM	0.826	0.956	4.64	9.63	39.3M
ResNet50-Encoder+ConvLSTM	**0.873**	**0.986**	**3.17**	**7.18**	43.2M

4.5.1 Versatility of Generated Poses

To quantitatively reflect the versatility of generated hand postures of SeqHAND dataset, we measure the average angular difference between the wrist and tips of fingers. In order to measure the angle variations for each finger, we firstly align a metacarpal (bones within a hand palm that are proximal and connected to each finger from a wrist) for each corresponding finger to the origin coordinates and normalize the length to be 1. We then measure a vector from the origin to the tip of the finger in order to measure the angle between the wrist and finger tips. The reason we align the metacarpal line to (0, 0, 0) and (0, 1, 0) is that to identify the measured angles for either inward variation or outward, which conclusively reflect feasible range of finger movements.

Range of joint motion is measured on the interphalangeal joints of the hand which are the hinge joints between the phalanges of the fingers that provide flexion towards the palm of the hand [14]. The degree of flexion of the proximal interphalangeal (PIP) can be generally said to be in the range of 100° to 110°. In the case of the distal interphalangeal (DIP) joints, the greatest degree of flexion is within 80°.

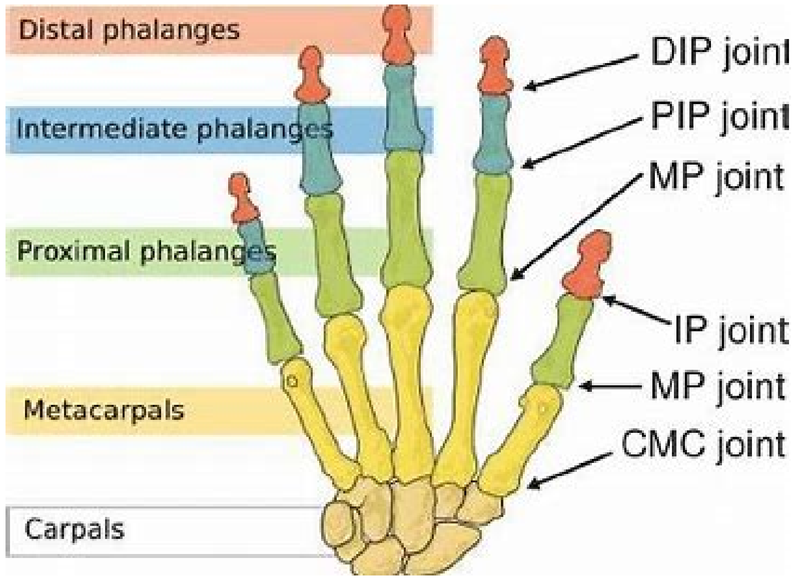


Figure 4.9: Structure of hand bones. The images is obtained from https://quizlet.com.

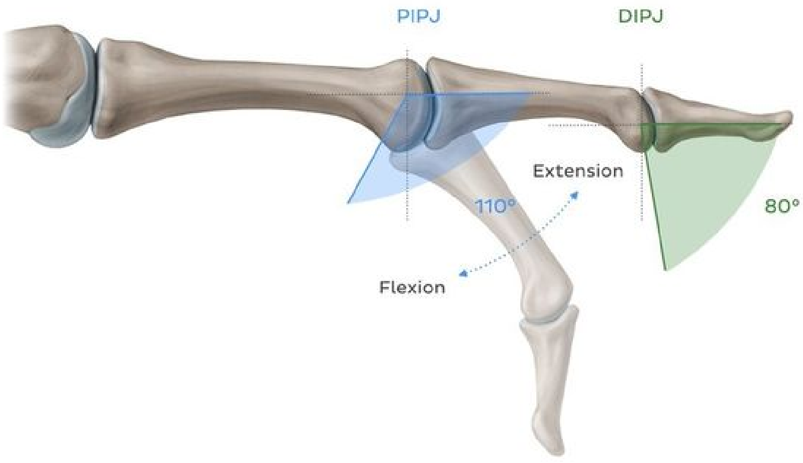
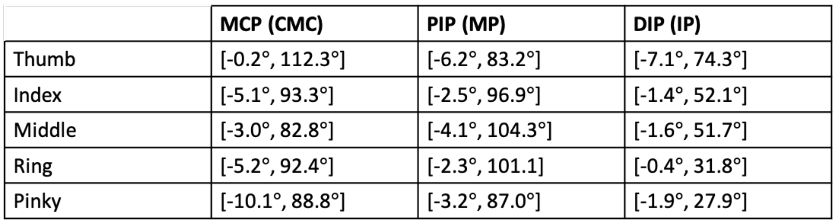


Figure 4.10: Range of motion for PIP and DIP joints. The image is acquired from https://www.physio-pedia.com



	MCP (CMC)	PIP (MP)	DIP (IP)
Thumb	[-0.2°, 112.3°]	[-6.2°, 83.2°]	[-7.1°, 74.3°]
Index	[-5.1°, 93.3°]	[-2.5°, 96.9°]	[-1.4°, 52.1°]
Middle	[-3.0°, 82.8°]	[-4.1°, 104.3°]	[-1.6°, 51.7°]
Ring	[-5.2°, 92.4°]	[-2.3°, 101.1]	[-0.4°, 31.8°]
Pinky	[-10.1°, 88.8°]	[-3.2°, 87.0°]	[-1.9°, 27.9°]

Figure 4.11: Range of motion for MCP, PIP and DIP joints of SeqHAND data samples. The range is within the least and the greatest value noted at each joint for each finger.

Figure 4.11 summarizes overall versatility of generated postures of SeqHAND dataset. As it can be shown in the table, the range of angular difference reflect that the generated postures generally follow the finger movements collected from real hands.

4.5.2 Ablation Study

Framework Selection

To show the logic behind the selection of the proposed framework, we evaluate various forms of extended baseline model [6] shown in Table 4.2 for managing sequential inputs on our newly generated SeqHAND dataset. The extended versions of baseline encoder (ResNet-50) include the baseline model with a LSTM layer [81], the baseline model with a ConvLSTM layer [88], the baseline encoder with the structure of I3D[9] and the baseline encoder with the structure of MF-Net [46]. Both I3D-Encoder and MFNet-Encoder represent methods that incorporate sequential inputs with 3D convolutional neural network. For I3D, we have changed few features from the original form of I3D so that its structure fits into the hand pose estimation task. The original backbone structure of

Table 4.3: Performances of differently (partially) trained models on ED, DO, STB datasets.

Methods	AUC			Avg. 3D Error (mm)		
	ED	DO	STB	ED	DO	STB
Encoder + Train(SynthHAND)	0.350	0.095	0.140	52.11	100.84	68.86
Encoder + Train(SynthHAND) + Train(FH + STB)	0.397	0.516	**0.985**	49.18	33.12	**9.80**
Encoder + C-LSTM + Train(SeqHAND)	0.373	0.151	0.121	52.18	81.51	71.10
Encoder + C-LSTM + Train(SeqHAND) + Train(FH + STB)	0.444	0.581	0.981	40.94	29.41	9.82
Encoder + C-LSTM + Train(SeqHAND) + Train$_C$(FH + STB)	**0.766**	**0.843**	0.978	**17.16**	**18.12**	9.87

I3D with Inception modules have changed into ResNet-50 for a fair comparison. And, it originally performs max-poolings througout the time domain, we however have omitted the part since we desire it to infer hand poses for every input frame from a sequence. MFNet is another examplary 3D convolution network proposed specifically for motion feature extractions. Of the candidates, the encoder with a ConvLSTM layer has performed the best.

The Effectiveness of *SeqHAND-Net* and *SeqHAND* Dataset

To clarify the effectiveness of our proposed framework and our generated dataset, variations of the proposed method and the baseline model are investigated. We report AUCs of 3D PCK curves and average 3D joint location errors for ED, DO and the evaluation set of STB datasets. In Table 4.3, 'Encoder' denotes the baseline model with ResNet50 backbone structure while 'Encoder + C-LSTM' denotes our proposed framework SeqHAND-Net. 'Train(SynthHAND)' and 'Train(SeqHAND)' represent training a model with synthetic hand image dataset respectively in non-sequential and sequential manner. 'Train(FH + STB)' and 'Train$_C$(FH + STB)' refers to training with STB and FreiHand dataset for the synthetic-real domain transfer with the ConvLSTM layer, respectively, at-

Table 4.4: Average 3D joint distance (mm) to ground-truth for RGB Sequence datasets hand pose benchmarks.

	Avg. 3D Error (mm)		
	ED	DO	STB
Our Method	**17.16**	**18.12**	9.87
Bouk. et al. (RGB) [6]	51.87	33.16	**9.76**
Bouk. et al. (Best) [6]	45.33	25.53	**9.76**
Spurr et al. [75]	56.92	40.20	-
Zimmer. et al. [102]	52.77	34.75	-

tached and detached from finetuning.

We show in the Table 4.3 how much performance enhancement can be obtained with SeqHAND dataset and our proposed domain adaptation strategy. Encoder with the ConvLSTM layer finetuned to real domain consequently performs similar to the encoder that does not consider visuo-temporal correlations. If the ConvLSTM layer is detached from finetuning and visuo-temporal features learned are preserved, the performance significantly improves. Also, SeqHAND dataset does not consist with any occluded hands except for self-occlusions. With training for FreiHand dataset, our method is able to learn the visual features of not only real hands but also occluded real hands since FreiHand dataset's augmentations consist of occlusions. Due to the temporal constraint that penalizes large difference among sequential estimations, per-frame estimation performs slightly better for the STB dataset.

Nevertheless, considering successive frames has allowed our method to be more robust to occlusions even before domain finetuning, performing better for DO and ED datasets than the baseline encoder. Hand movements in STB dataset are comparably slow compared to SeqHAND, ED and DO datasets. Thus at the pre-trained stage, temporal dynmaic features learned from SeqHAND dataset is redundant for STB dataset. With training for FH dataset, our method is able to learn the visual features of not only real hands but also occluded real hands since FH dataset's augmentations consist of occlusions. As the result, our method performs the best for the dataset with swift hand motions. Having SeqHAND data sequences with more static hand postures would have helped our method to perform better for STB dataset, but we have decided to clearly focus the problem scope on visuo-temporal information exploitation for robustness against dynamic hand motions.

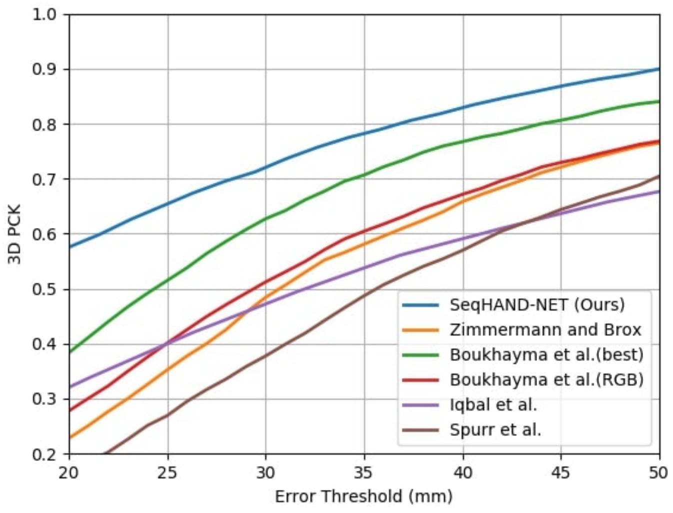


Figure 4.12: 3D PCK for ED

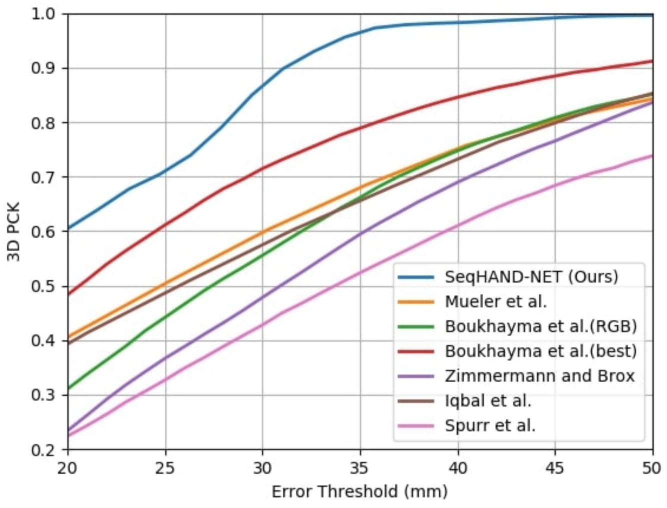


Figure 4.13: 3D PCK for DO

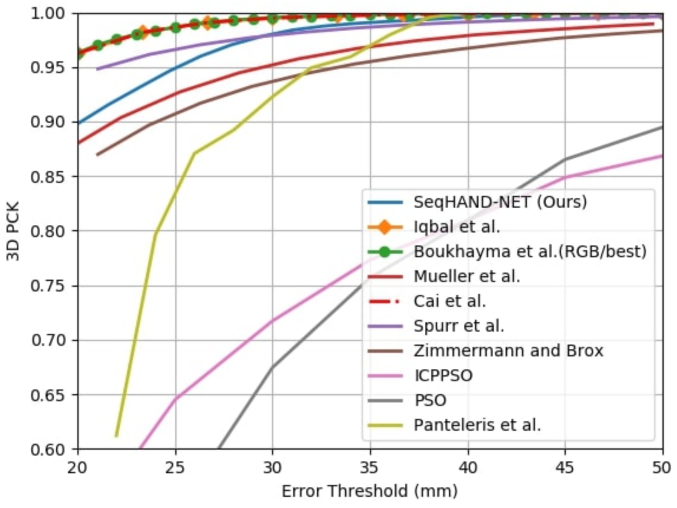


Figure 4.14: 3D PCK for STB

4.5.3 Comparison against State-of-the-art Methods

In Figures 4.12, 4.13 and 4.14, we have plotted 3D-PCK graph with various thresholds for STB, ED and DO datasets. For STB dataset, deep-learning based works of [6, 7, 75, 34, 56, 102] and approaches from [64, 94] are compared. Many previous methods have reached near the maximum performance for STB dataset. With temporal constraints and fixing the ConvLSTM layer during fine-tuning, our method reaches a competitive performance. For both ED and DO datasets, our method outperforms other methods. For ED dataset, contemporary works of [102, 6, 34, 75] are compared to our method. The best performance of our baseline [6] is reached with inputs of RGB and 2D pose estimations provided by an external 2D pose estimator. Our method results in outstanding performance against other compared methods [56, 102, 6, 34, 75] for DO dataset with heavy occlusions, which shows that the learning of pose-flow continuity enhances robustness to occlusions. Temporal information exploitation from sequential RGB images affect our model to be robust against dynamically moving scene. For more absolute comparisons, we provide our average 3D error of joint location in Table 4.4.

We provide qualitative results in Figure 4.15 and 4.16 for visual comparison against a frame-by-frame 3D pose estimator, our reproduced work of [6]. All images in the figure are sequentially inputted to both estimators from left to right. Per-frame estimations that fit postures at each frame result in unnatural 3D hand posture changes over a sequence while our method's leaning trajectories biased by previous frames produces natural hand motions and robust estimations to frames that lack visual information of hand postures. The feature of our method that is trained with the temporal criterion that penalizes large change of pose and shape parameters allows such temporal estimation continuity. When a frame lacks much visual information of hand postures as in the cases in Figure 4.16(b), the frame-by-frame estimator's performance significantly decrements. Our method, on the other hand, considers the motion context and overcomes such issue. As illustrated in Figure 4.16(c), Our method models estimated hand shapes as consistent as possible per sequence. During the qualitative evaluation on a RGB image sequence of a single real hand, our method's average difference among temporal changes of shape parameters $\beta_{t-1} - \beta_t$ is $4.16e^{-11}$ while that of the frame-by-frame estimator is $2.38e^{-5}$. The average difference among temporal changes of the pose parameters $\theta_{t-1} - \theta_t$ are $1.88e^{-6}$ for our method and $6.90e^{-6}$ for the other.

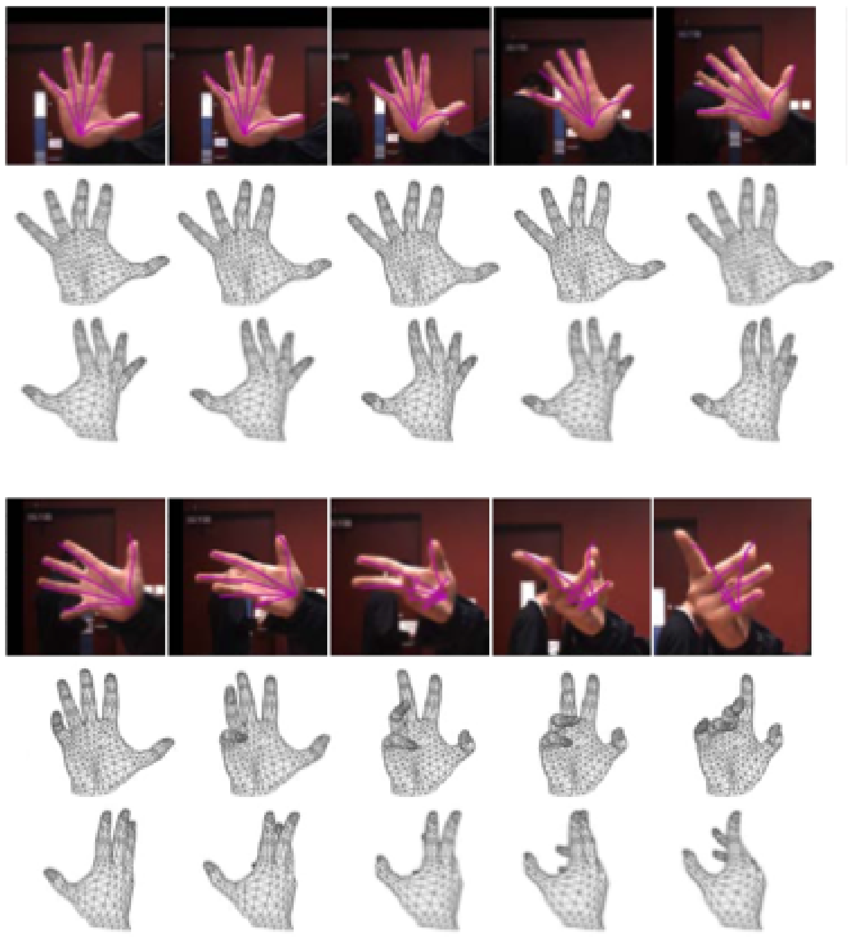


Figure 4.15: Qualitative estimation results of our method on STB dataset.

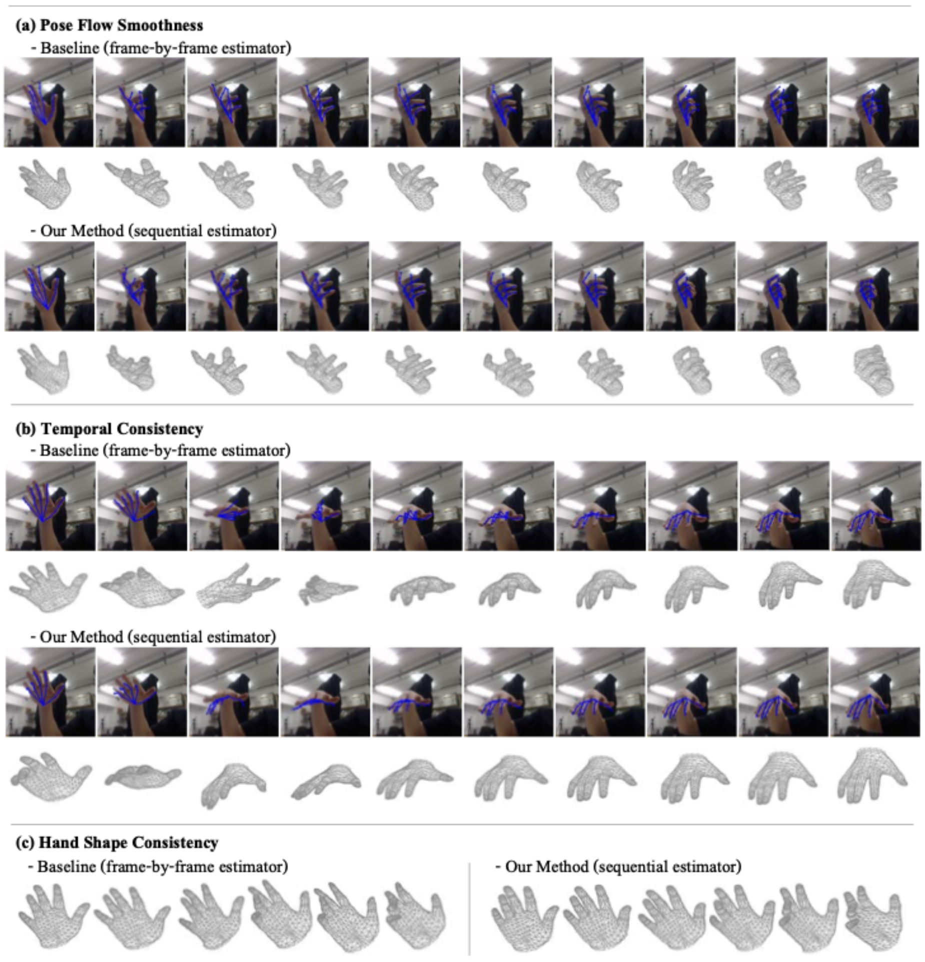

(a) Pose Flow Smoothness
- Baseline (frame-by-frame estimator)
- Our Method (sequential estimator)
(b) Temporal Consistency
- Baseline (frame-by-frame estimator)
- Our Method (sequential estimator)
(c) Hand Shape Consistency
- Baseline (frame-by-frame estimator)
- Our Method (sequential estimator)

Figure 4.16: Qualitative estimation results on Temporal Consistency/Smoothness. Unlike our baseline model which performs estimations frame-by-frame and does not consider temporal features, our method performs smooth and temporally intuitive estimations despited of sudden and gradual successive hand postures.

4.6 Conclusion and Discussion

In this chapter of my dissertation, we have addressed and tackled the scarcity of sequential RGB dataset which limits conventional methods from exploiting temporal features for 3D HPE. We have proposed a novel method to generate SeqHAND dataset, a dataset with sequential RGB image frames of synthetic hand poses in motions that are interpolated from existing static pose annotations. We have then also proposed a framework that exploits visuo-temporal features for 3D hand pose estimations in a recurrent manner. We have implemented cost functions considering the temporal smoothness of sequential hand pose estimations. However, such framework would still be limited due to lack of large scale sequential RGB hand images, so to achieve a stable and efficient solution, we uniquely designed a new large-scaled hand motion dataset generation framework that generates synthetic images with human-like hand movements while preserving realistic appearances of hands as conventionally studied by other works in the literature. Our proposed method outperforms other existing approaches that take RGB-only inputs that are based on solely appearance-based methods, and consequently produces pose-flow estimations that mimic natural movements of human hands. We plan to enable the framework to solve (self-)occlusion problems more robustly.

With sequential inputs, we were able to witness possibility of overcoming conventional struggle against occlusion problems in the literature of 3D hand pose estimations. We plan to further the project towards unifying an external hand detector that localizes hands from raw images into a single 3D hand pose estimator since both tasks correlate closely. In addition, such framework would avoid tracking in-continuity, which is the reason we had to experience errors caused by pester-some hand tracking problems.

Chapter 5

Hand Pose Auto-Augment

5.1 Motivation

Hand pose estimation (HPE) is an essential task for augmented reality and virtual reality (collectively called as "extended reality (XR)") systems. For instance, to enable hand-based interactions with objects in XR environments,accurate real-time estimates of the positions of hand joints in 3D world coordinates are needed. Since hand gestures reflect elementary human behavioral patterns, hand pose tracking enables several downstream AI applications such as gesture recognition [37, 66] and human-computer interactions [35].

Recent methods have used synthetic datasets that are much larger than real dataset to pretrain a network [58, 102, 6, 56, 20, 39]. Such efforts allow neural networks to learn larger spaces of image semantic and hand poses before fine-tuning to real domains, because with synthetic, large spaces of pose and images can be artificially created while such various poses, backgrounds, skin color and lighting conditions are hard to be covered with dataset collected with real subjects. The network's innate semantic encoding mechanism has its own limit of overcoming the differences of features learned from real and synthetic images. In order for synthetic-to-real transfer to more effectively apply for neural network training, the efforts of image feature domain gap reduction should thus be performed inner-feature level, rather than image-level.

For many years, dataset augmentation has been a standard regularization technique used to reduce overfitting while training supervised learning models. Data augmentation is particularly popular for visual recognition tasks as new data can be generated very easily by applying image manipulations such as shifting, scaling, rotation, and other affine transformations. When training LeNet5 [45], one of the most early and well-known convolutional neural network architectures, their work applied a series of transformations to the input images in order to improve the robustness of the model. An early work of deep learning [43] also used image transformations to generate new data when training the renowned AlexNet model for the 2012 Large Scale Visual Recognition Challenge (ILSVRC). They claimed that dataset augmentation reduced the error rate of the model by over 1%. Creating new data has since been a crucial component of all recent large-scale image recognition models. Unfortunately, dataset augmentation is not as straightforward to apply in all domains as it is for images. For example, a recent work [71] investigated a variety of data augmentation techniques for application to singing voice detection. These include adding Gaussian noise to the input, shifting the pitch of the audio signal, time stretching, varying the loudness of the audio signal, applying random frequency filters, and interpolating between samples in input space. They found that only pitch shifting and random frequency filtering appeared to improve model performance. While performing well on audio data, these augmentation techniques cannot be applied to other domains. As such, the process of designing, implementing, and evaluating new data augmentation techniques would need to be repeated for each new problem.

In this work, we consider augmentation not by a domain-specific transformation, but by perturbing, interpolating, or extrapolating between existing examples. However, we choose to operate not in input space, but in a learned feature space. Higher level representations are claimed to expand the relative volume of plausible data points within the feature space, conversely shrinking the space allocated for unlikely data points [5, 62]. As such, when traversing along the manifold it is more likely to encounter realistic samples in feature space than compared to input space. Unsupervised representation learning models offer a convenient way of learning useful feature spaces for exploring such transformations. Recently, there has been a return to interest in such techniques, leading to, e.g., variational autoencoders [41], generative adversarial networks [25], and generative stochastic networks [2], each of which could be used to generate useful feature spaces for augmentation.

By manipulating the vector representation of data within a learned feature space a dataset can be augmented in a number of ways. One of the most basic transformations that can be applied to the data is to simply add random noise to the context vector. In the context of class-imbalanced data, we believe that extrapolation between samples could also be applied. We investigate some of these methods to see which is most effective for improving the performance of supervised learning models when augmented data is added to the dataset.

Variations in hand postures, in this context, are expressed with MANO parameters θ, β, R, t and s which respectively represent pose, shape of hands and rotation, translation and scale factors of camera viewpoints. Distribution of realistic hand postures can be modeled even before training with annotations of existing dataset, such as *BigHand2.2M* dataset [93] from Chapter 4. BH dataset consists of 2.2 million pose samples with 3D annotations for joint locations acquired from 2 hour-long hand motions collected from 10 real subjects. With BH datasets, the generated samples are expected to inherit the manifold of its real human hand articulation space and kinematics of real hand postures. As 3D mapping of BH samples using t-SNE [51] in Figure 4.3 shows, BH is known that the pose samples are densely and widely collected. Such density of BH dataset with a more complete range of variation is considered sufficient for various pose generations. As similarly done in Chapter 4, we first train BH data annotations with a Variational Autoencoder (VAE) to learn a mapping function from BH annotations to MANO pose parameters. This allows to learn the distribution of realistic hand postures that span BH annotation space, which can be utilized for initial distribution to sample hand poses. With each latent pose parameters including camera parameters, our own 3D and 2D annotations can virtually be acquired.

In this section, we propose a novel feature-level auto-augmentation method that augments features that span a continuous space. Previous attempts of auto-augmentation are mainly consisted of discrete action space where each action corresponds to a augmentation policy (*e.g.* Shear-X translation, Brightness, Inversion ane etc.). Searches for augmentation policies in continuous space require for a policy network to have a far more delicate variations [48]. In terms of augmentation, we first emphasize on enlarging the training distribution of humand hand postures. This, however, allows robustness only against 3D pose estimations, and not necessarily in 2D estimations. Hands may appear variously within images depending on the illuminating condition, skin color, shades, hairs, wrinkles and more. A fixed set of training data limits its distribution of image space features. We thus utilize random images (not necessarily with hands in them) and synthesize them in a feature space in order to ‘hallucinate’ the network, so that general image encoding can be performed. As mentioned earlier in this section, augmentation in feature space is economical and efficient for training data-hungry neural networks. Enlarge the distribution of input images allow our network to generalize over various image inputs. For automation of the augmentation, we sample a hand pose vector from a pre-defined distribution that span a physically plausible human hand pose space. The distribution is trained to evolve to sample better pose hallucination features in an adversarial manner against the training loss.

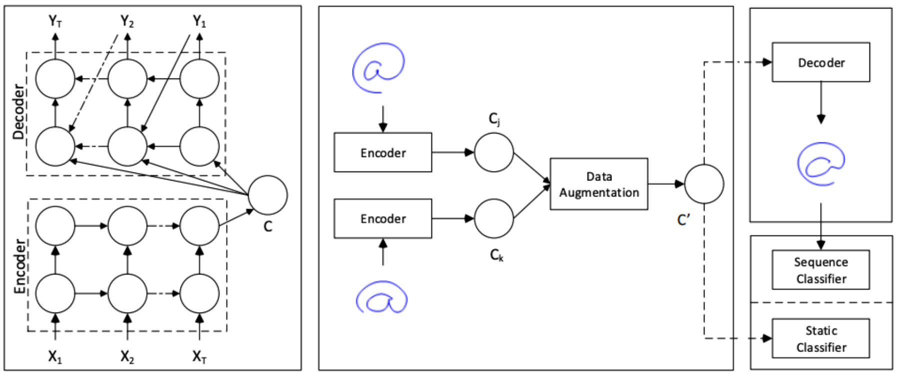
To this end, our contribution in this section are as follows:

- We propose to augment hand pose data without explicit augmented images, but implicitly in feature space for variations of both RGB values and hand postures.
- To do so, we propose *Feature Synthesizer* module that incorporates human hand pose feature and any random RGB images (not necessarily with hands) to augment features of a network.
- The augmentation is automatically performed and learned to sample more effective pose in an adversarial manner against training loss.
- Our method reaches a state-of-the-art performance on competitive hand pose estimation benchmarks.

5.2 Related Works

5.2.1 Feature-level Augmentation

An early approach of dataset augmentation in feature space is performed in a VAE framework with sequence type of data [18].



(a) Sequence autoencoder (b) Encode and apply data transform (c) Decode and/or classify

Figure 5.1: System architecture composed of three steps. (a) A sequence autoencoder learns a feature space from unlabeled data, representing each sequence by a context vector (C). (b) Data is encoded to context vectors and augmented by adding noise, interpolating, or extrapolating (here we depict interpolation). (c) The resulting context vectors can either be used directly as features for supervised learning with a static classifier, or they can be decoded to reconstruct full sequences for training a sequence classifier.

In order to augment a dataset, each example is projected into feature space by feeding it through the sequence encoder, extracting the resulting context vector, and then applying a transformation in feature space (Figure 5.1b). The simplest transform is to simply add noise to the context vectors, however, there is a possibility with this method that the resulting vector may not resemble the same class as the original, or even any of the known classes. In experiments, they generate noise by drawing from a Gaussian distribution with zero mean and per-element standard deviation calculated across all context vectors in the dataset. They include a γ parameter to globally scale the noise:

$$c_i^{'} = c_i + \gamma X, X \mathcal{N}\{0, \sigma_i^2\} \tag{5.1}$$

where i indexes the elements of a context vector which corresponds to data points from the training set. A more directed approach for data augmentation follows the techniques introduced by the work of [12]. For each sample in the dataset, we find its K nearest neighbours in feature space which share its class label. For each pair of neighbouring context vectors, a new context vector can then be generated using interpolation:

$$\mathbf{c}^{'} = (\mathbf{c}_K - \mathbf{c}_j)\lambda + \mathbf{c}_j \tag{5.2}$$

where $\mathbf{c}^{'}$ is the synthetic context vector, $\mathbf{c}_i$ and $\mathbf{c}_j$ are neighbouring context vectors, and λ is a variable in the range $\{0, 1\}$ that controls the degree of interpolation. In our experiments, we use $\lambda = 0.5$ so that the new sample balances properties of both original samples. In a similar fashion, extrapolation can also be applied to the context vectors with a similar degree of the λ value:

$$\mathbf{c}_j^{'} = (\mathbf{c}_j - \mathbf{c}_K)\lambda + \mathbf{c}_j. \tag{5.3}$$

Once new context vectors have been created, they can either be used directly as input for a learning task, or they can be decoded to generate new sequences (Figure 5.1c). When interpolating between two samples, the resulting decoded sequence is set to be the average length of the two inputs. When extrapolating between two samples the length of the new sequence is set to be the same as that of c_j .

5.2.2 Auto-Augment

Our method largely follows the studies of automatic augmentation strategies (Auto-Augment) [15, 49, 30, 97, 86] although our fundamental approach of augmentation differs from the works in many aspects. The original work of Auto-Augment formulates the problem of finding the best augmentation policy as a discrete search problem, as illustrated in Figure 5.2. Their method consists of two components: A search algorithm and a search space. At a high level, the search algorithm (implemented as a controller RNN) samples a data augmentation policy S, which has information about what image processing operation to use, the probability of using the operation in each batch, and the magnitude of the operation. Key to our method is the fact that the policy S will be used to train a neural network with a fixed architecture, whose validation accuracy R will be sent back to update the controller. Since R is not differentiable, the controller will be updated by policy gradient methods.

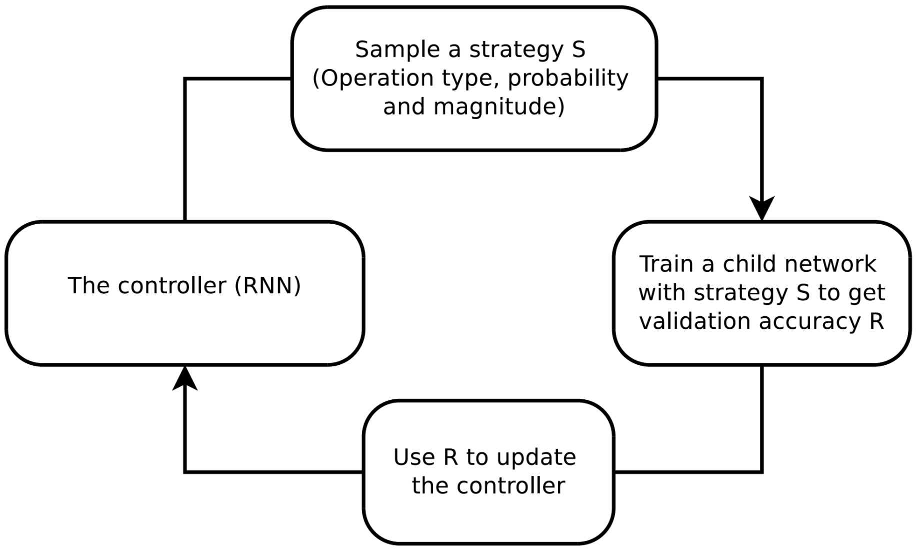

Sample a strategy S
(Operation type, probability and magnitude)
Train a child network with strategy S to get validation accuracy R
Use R to update the controller
The controller (RNN)

Figure 5.2: Overview of Auto-Augmentation framework of using a search method (e.g., Reinforcement Learning) to search for better data augmentation policies. A controller RNN predicts an augmentation policy from the search space. A child network with a fixed architecture is trained to convergence achieving accuracy R. The reward R will be used with the policy gradient method to update the controller so that it can generate better policies over time. The image is from the original work [15]

Augmentation policies from the original work are updated every epoch after all batches of training data samples are 'seen' by the child model, which makes the whole training progress takes a massive computation hours to reach convergence. The policy controller that is structured as RNN requires large number of epochs to learn. The works of [49, 30] try to overcome the large computation burden.

Unlike Auto-Augmentation methods that accompany with reinforcement updates, the augmentation strategy can be learned in an adversarial manner. For example as in the work of [97], the augmentation data sampling distribution is learned are sampled in the direction of increasing training loss of the child model. The method not only allows great reduction of training hours, but also more stable learning of the whole automatic augmenting system. The framework is retrieved from the original work and illustrated in Figure 5.3

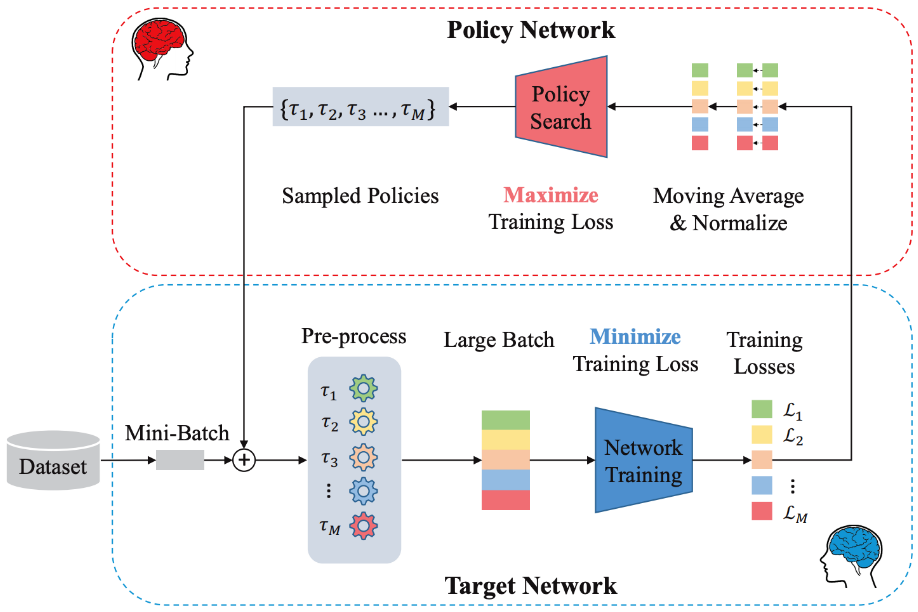


Figure 5.3: The work of [97] formulates the automatic augmentation policy learning with an adversarial update setting. The data of each batch is augmented by multiple pre-processing components with sampled policies $\{\tau_1, \tau_1, ..., \tau_M,\}$, respectively. Then, a target network is trained to minimize the loss of a large batch, which is formed by multiple augmented instances of the input batch. We extract the training losses of a target network corresponding to different augmentation polices as the reward signal. Finally, the augmentation policy network is trained with the guideline of the processed reward signal, and aims to maximize the training loss of the target network through generating adversarial policies.

5.2.3 RandAugment

Recent work has shown that data augmentation has the potential to significantly improve the generalization of deep learning models. As mentioned in a previous section, automated augmentation strategies have led to state-of-the-art results in image classification and object detection. While these strategies were optimized for improving validation accuracy, they also led to state-of-the-art results in semi-supervised learning and improved robustness to common corruptions of images. However such methods still require a separate optimization procedure, which significantly increases the computational cost and complexity of training a machine learning model. One of the sharing outcomes of auto-augmentation and relevant methods is that the learned augmentation policy is more likely to output large magnitude of augmentation such as brightness and contrast. Motivated by such issue, The work of [16] proposes to randomly choose augmentation policy instead of learning one. This has a significantly reduced search space which allows it to be trained on the target task with no need for a separate proxy task. RandAugment yields better performance for CIFAR-10/100, SVHN, and ImageNet datasets than those of automatic augmentation strategies.

```
transforms = [
'Identity', 'AutoContrast', 'Equalize',
'Rotate', 'Solarize', 'Color', 'Posterize',
'Contrast', 'Brightness', 'Sharpness',
'ShearX', 'ShearY', 'TranslateX', 'TranslateY']

def randaugment(N, M):
"""Generate a set of distortions.

  Args:
    N: Number of augmentation transformations to
        apply sequentially.
    M: Magnitude for all the transformations.
"""

 sampled_ops = np.random.choice(transforms, N)
 return [(op, M) for op in sampled_ops]
```

Figure 5.4: An Illutration of the work of [16] that formulates their augmentation policy to be random instead of learning automatic augmentation policy learning with an adversarial update setting. The motivation of their work comes from the issue that auto-augmentation outputs more complex augmentation policies as training progresses.

5.3 Hand Pose Auto-Augment (HPAA)

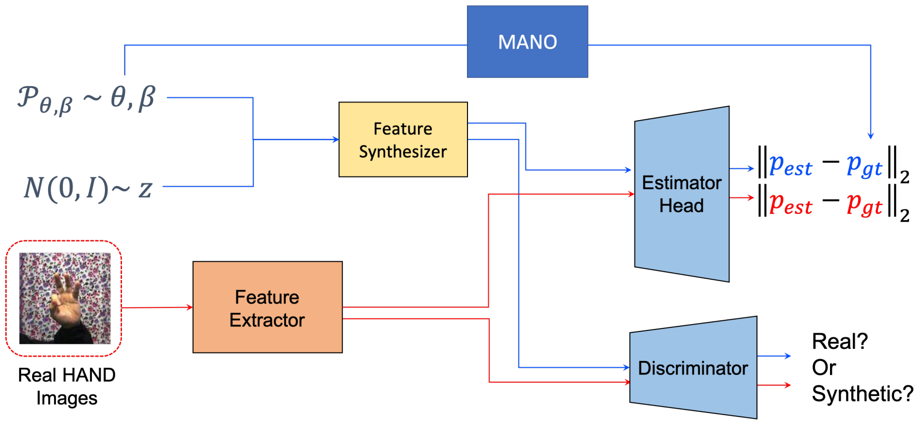


Figure 5.5: This figure illustrates the overall framework of Hand Pose Auto-Augmentation (HPAA). HPAA augments visual features in a feature space. The augmented features are generated based on two random variables; one from a pose distribution that determines pose θ and shape β, and another random variable that generates fake feature. Each variable distribution is learned with two different losses. First variable that determines pose and shape of hands based on MANO is learned with L2-loss between the estimated poses and the poses that are determined by the sampled MANO parameters. Another random variable is trained by the discriminator loss while aligning the domains of fake and real features. *Estimator Head* is structured with 3 FC layers that output MANO pose θ and shape β parameters along with camera parameters R, t, s, as depicted in Figure 5.6. And, the discriminator is structured with 2 FC layers that outputs values in range of [0, 1].

In this section, we propose a novel augmentation method that is performed in an automatic end-to-end manner while regular training of a hand pose estimation model is performed. Our framework for training is largely structured with a *Feature Extractor* (FE), *Feature Synthesizer* (FE-Synth) and *Estimator Head* (Estimator). FE is responsible for encoding images into compacted features that connote RGB image features and pose features. The features from FE are strictly from real images. In order to effectively perform automatic augmentation with synthetic sample generations, we create synthetic features that hallucinate the Estimator head. Feature Synthesizer “synthesizes” the MANO parameters β, θ into fake feature samples to confuse the estimator head. The synthesizer is learned to generate realistic fake features through GAN-like updates. One iteration of update is performed with F-synth and without FE, and another is done vice versa. This learning is inspired by domain adversarial alignment learning [1] We perturb the image features by inputting random variables to F-synth so that Estimator Head is able to learn about other image features. Using variations within the distribution of images seen by the network is only interpolating among its training distribution, easily yielding over-fitting. We thus apply adversarial updates of the distribution against the training loss. The method is thus updated three times: one with only FE fixed, one with only F-synth fixed and the last one only with sampling distribution.

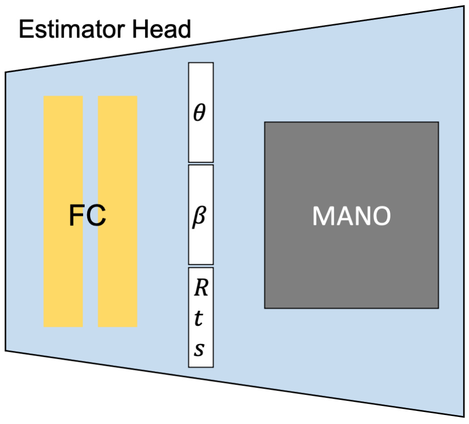


Figure 5.6: The structure of Estimator Head.

5.4 Experiments

FreiHand Dataset In order to present the effectiveness of HPAA, the method is evaluated with FreiHand dataset [103], details of which is visualized in Figures 2.6 and 2.7. The training set of the dataset is re-visualized with random images projected on the chromakey part of training image samples. The distribution of FreiHand datsaet spans much wider space then that of STB dataset. STB dataset is collected so that data samples are visualized under various lighting conditions while FreiHand samples are collected for various poses (in both 3rd-person and egocentric perspectives) with various lighting conditions and backgrounds replaced with chromakey background. Such factor of STB is reviewed in its original work [95] We therefore perform our main experiment on FreiHand dataset since our method proves of its conributions based on augmenting poses.

Comparisons We firstly set our baseline as the work of [6], since our network's architectural framework is inspired by the method. The performance of the baseline is acquired without any form of augmentation. Then we perform an experiment with the baseline with random augmentation (RandAugment) [16]. The random augmentation is executed on the original training images. Since our groundtruths included 2D keypoint locations for the finger joints, we exclude shear transformations and vertical/horizontal flips to avoid mismatches of 2D keypoint labels. We consider RandAugment represents the performance of other auto-augmentation methods (e.g. adversarial auto-augmentation [97]) since it outperforms other standard automatic augmentation methods. We conduct experiments with two variants of our proposing method; one method with random pose sampling from a fixed distribution of random (PoseRandAugment) and another with learning pose sampling distribution (PoseAugment).

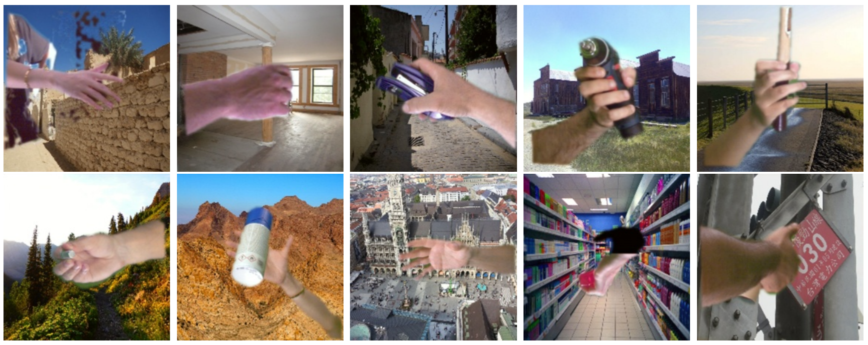


Figure 5.7: FreiHand data images with random images on chromakey part of images.

Table 5.1: Experimental results on FreiHand datset [103] based on various settings of HPAA. Two of settings of HPAA are performed; one with prior distribution of MANO parameters $M = \{\theta, \beta\}$ learned during training baseline, and one without any prior distribution.

	Base	Base+RA	Base+PRA		Base+PA	
			w/ $P_{base}(M)$	w/out $P_{base}(M)$	w/ $P_{base}(M)$	w/out $P_{base}(M)$
2D/3D AUC	0.725/0.921	0.715/0.909	0.762/0.952	0.687/0.876	0.739/0.939	**0.782/0.958**

Table 5.1 analyzes our contributions upon baseline. RA, PRA and PA respectively represent RandAugment [16], PoseRandAugment and PoseAugment. $P(M)$ represents the distribution of MANO parameters $M = \{\theta, \beta\}$ and P_{base} means the distribution learned from baseline. "With $P_{base}(M)$" the pose sampling distribution is initialized with the baseline distribution. Sampling without $P_{base}(M)$ represents initialization of poses and shapes to be sampled from ranges of [-2, 2] and [-0.1, 0.1], respectively.

As it can be shown in Table 5.1, RandAugment does not improve the baseline's performance, but degrades it. We believe that this is caused because, unlike object detection problems where objects are present as lumps in an image, hands are highly deformable and its structure must be meticulously reviewed by a pose estimator. Random lighting condition upon images of hands may result in loss of visual information of fingers. Few of such cases are presented in Figure 5.8. The magnitude of augmentation does not affect evenly to all fingers. The best performance is acquired by our method without the use of prior pose distribution of baseline.

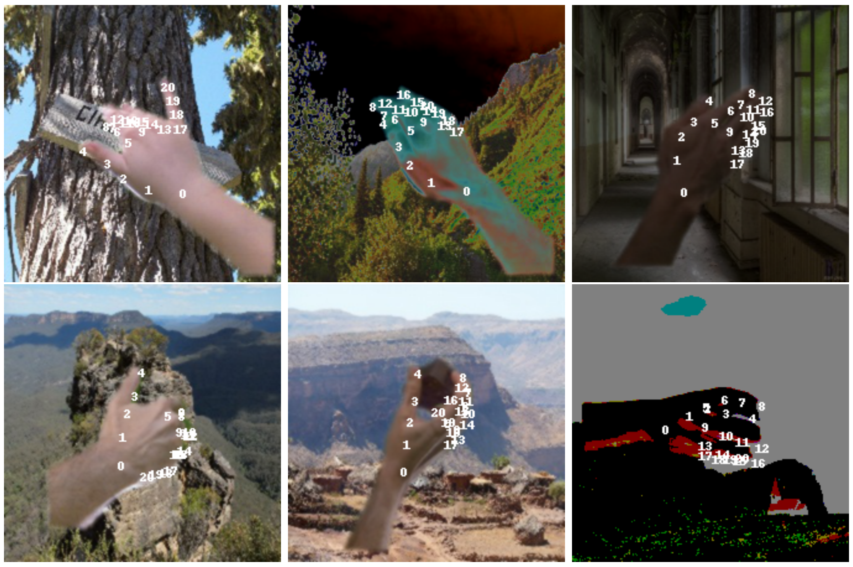


Figure 5.8: Examples of FreiHand samples [103] augmented by a random policy [16].

SOTA Comparison Figure 5.10 shows our results along with other contemporary RGB-based 3D hand pose estimation methods. Our final result outperforms all the conventional methods. Extrapolating from trianing distribution with a pose distribution clearly affects the network's performance. Also, qualitative results of STB dataset are presented in Figure 5.10.

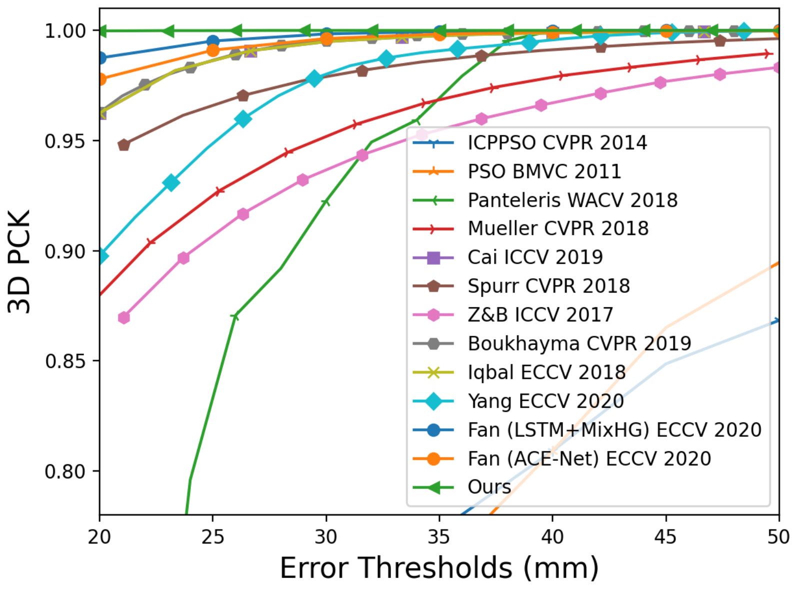


Figure 5.9: HPAA result of 3D AUC on STB Dataset

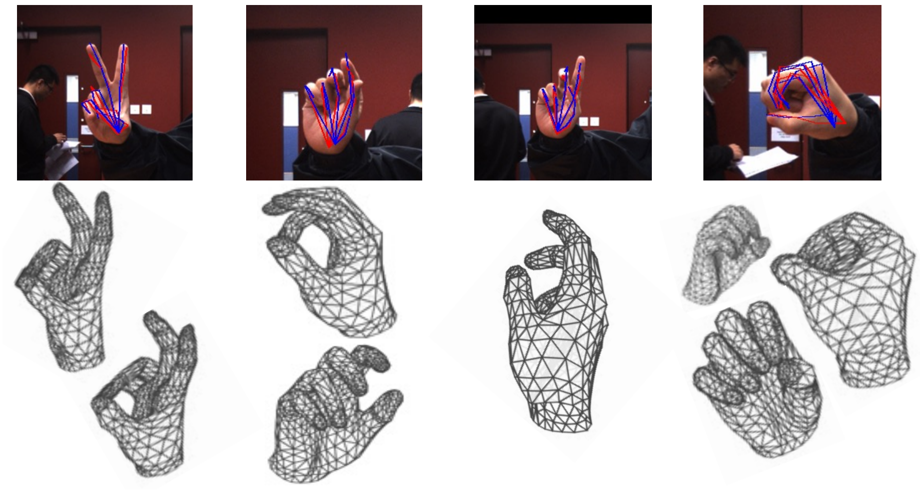


Figure 5.10: Qualitative results of HPAA on STB Dataset.

Chapter 6

Conclusion

In my dissertation, we have addressed and tackled the scarcity of sequential RGB dataset which limits conventional methods from exploiting temporal features for 3D HPE. We have proposed a novel method to generate SeqHAND dataset, a dataset with sequential RGB image frames of synthetic hand poses in motions that are interpolated from existing static pose annotations. We have then also proposed a framework that exploits visuo-temporal features for 3D hand pose estimations in a recurrent manner. We have implemented cost functions considering the temporal smoothness of sequential hand pose estimations. However, such framework would still be limited due to lack of large scale sequential RGB hand images, so to achieve a stable and efficient solution, we uniquely designed a new large-scaled hand motion dataset generation framework that generates synthetic images with human-like hand movements while preserving realistic appearances of hands as conventionally studied by other works in the literature. Our proposed method outperforms other existing approaches that take RGB-only inputs that are based on solely appearance-based methods, and consequently produces pose-flow estimations that mimic natural movements of human hands. We plan to enable the framework to solve (self-)occlusion problems more robustly.

With sequential inputs, we were able to witness possibility of overcoming conventional struggle against occlusion problems in the literature of 3D hand pose estimations. We plan to further the project towards unifying an external hand detector that localizes hands from raw images into a single 3D hand pose estimator since both tasks correlate closely. In addition, such framework would avoid tracking in-continuity, which is the reason we had to experience errors caused by pester-some hand tracking problems.

In this work, we consider augmentation not by a domain-specific transformation, but by perturbing, interpolating, or extrapolating between existing examples. However, we choose to operate not in input space, but in a learned feature space. Higher level representations are claimed to expand the relative volume of plausible data points within the feature space, conversely shrinking the space allocated for unlikely data points. As such, when traversing along the manifold it is more likely to encounter realistic samples in feature space than compared to input space. Unsupervised representation learning models offer a convenient way of learning useful feature spaces for exploring such transformations. By manipulating the vector representation of data within a learned feature space a dataset can be augmented in a number of ways. One of the most basic transformations that can be applied to the data is to simply add random noise to the context vector. In the context of class-imbalanced data, we believe that extrapolation between samples could also be applied. We investigate some of these methods to see which is most effective for improving the performance of supervised learning models when augmented data is added to the dataset within the feature space.

Bibliography

[1] H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, and M. Marchand. Domain-adversarial neural networks. *arXiv preprint arXiv:1412.4446*, 2014.

[2] G. Alain, Y. Bengio, L. Yao, J. Yosinski, E. Thibodeau-Laufer, S. Zhang, and P. Vincent. Gsns: generative stochastic networks. *Information and Inference: A Journal of the IMA*, 5(2):210–249, 2016.

[3] S. Baek, K. I. Kim, and T.-K. Kim. Pushing the envelope for RGB-based dense 3D hand pose estimation via neural rendering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1067–1076, 2019.

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

[5] Y. Bengio, G. Mesnil, Y. Dauphin, and S. Rifai. Better mixing via deep representations. In *International conference on machine learning*, pages 552–560. PMLR, 2013.

[6] A. Boukhayma, R. d. Bem, and P. H. Torr. 3d hand shape and pose from images in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10843–10852, 2019.

[7] Y. Cai, L. Ge, J. Cai, and J. Yuan. Weakly-supervised 3D hand pose estimation from monocular RGB images. pages 666–682, 2018.

[8] Y. Cai, L. Ge, J. Liu, J. Cai, T.-J. Cham, J. Yuan, and N. M. Thalmann. Exploiting spatial-temporal relationships for 3D pose estimation via graph convolutional networks. 2019.

[9] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017.

[10] H. J. Chang, G. Garcia-Hernando, D. Tang, and T.-K. Kim. Spatio-temporal hough forest for efficient detection–localisation–recognition of fingerwriting in egocentric camera. *Computer Vision and Image Understanding*, 148:87–96, 2016.

[11] H. J. Chang, G. Garcia-Hernando, D. Tang, and T.-K. Kim. Spatio-temporal hough forest for efficient detection–localisation–recognition of fingerwriting in egocentric camera. *Computer Vision and Image Understanding*, 148:87 – 96, 2016.

[12] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[13] W. Chen, C. Yu, C. Tu, Z. Lyu, J. Tang, S. Ou, Y. Fu, and Z. Xue. A survey on hand pose estimation with wearable sensors and computer-vision-based methods. *Sensors*, 20(4):1074, 2020.

[14] C. Cooper, E. M. Dennison, H. G. Leufkens, N. Bishop, and T. P. van Staa. Epidemiology of childhood fractures in britain: a study using the general practice research database. *Journal of Bone and Mineral Research*, 19(12):1976–1981, 2004.

[15] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 113–123, 2019.

[16] E. D. Cubuk, B. Zoph, J. Shlens, and Q. V. Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

[17] T. E. de Campos and D. W. Murray. Regression-based hand pose estimation from multiple cameras. 1:782–789, 2006.

[18] T. DeVries and G. W. Taylor. Dataset augmentation in feature space. *arXiv preprint arXiv:1702.05538*, 2017.

[19] C. Doersch and A. Zisserman. Sim2real transfer learning for 3d human pose estimation: motion to the rescue. *arXiv preprint arXiv:1907.02499*, 2019.

[20] B. Doosti, S. Naha, M. Mirbagheri, and D. J. Crandall. Hope-net: A graph-based model for hand-object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6608–6617, 2020.

[21] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes

Challenge 2012 (VOC2012) Results. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[22] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 409–419, 2018.

[23] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 10833–10842, 2019.

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

[25] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[26] S. Hampali, M. Rad, M. Oberweger, and V. Lepetit. Honnotate: A method for 3d annotation of hand and object poses. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3196–3206, 2020.

[27] Y. Hasson, B. Tekin, F. Bogo, I. Laptev, M. Pollefeys, and C. Schmid. Leveraging photometric consistency over time for sparsely supervised hand-object reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 571–580, 2020.

[28] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid. Learning joint reconstruction of hands and manipulated

objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11807–11816, 2019.

[29] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid. Learning joint reconstruction of hands and manipulated objects. In *CVPR*, 2019.

[30] R. Hataya, J. Zdenek, K. Yoshizoe, and H. Nakayama. Faster autoaugment: Learning augmentation strategies using backpropagation. In *European Conference on Computer Vision*, pages 1–16. Springer, 2020.

[31] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[32] S. Hinterstoisser, V. Lepetit, P. Wohlhart, and K. Konolige. On pre-trained image features and synthetic images for deep learning. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, pages 0–0, 2018.

[33] Z. Hu, Y. Hu, J. Liu, B. Wu, D. Han, and T. Kurfess. A crnn module for hand pose estimation. *Neurocomputing*, 333:157–168, 2019.

[34] U. Iqbal, P. Molchanov, T. Breuel Juergen Gall, and J. Kautz. Hand pose estimation via latent 2.5 d heatmap regression. pages 118–134, 2018.

[35] A. Jaimes and N. Sebe. Multimodal human–computer interaction: A survey. *Computer vision and image understanding*, 108(1-2):116–134, 2007.

[36] Y. Jang, S. Noh, H. J. Chang, T. Kim, and W. Woo. 3D finger cape: Clicking action and position estimation under self-occlusions in egocentric

viewpoint. *IEEE Transactions on Visualization and Computer Graphics*, 21(4):501–510, April 2015.

[37] Y. Jang, S.-T. Noh, H. J. Chang, T.-K. Kim, and W. Woo. 3d finger cape: Clicking action and position estimation under self-occlusions in egocentric viewpoint. *IEEE Transactions on Visualization and Computer Graphics*, 21(4):501–510, 2015.

[38] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349(6245):255–260, 2015.

[39] K. Karunratanakul, J. Yang, Y. Zhang, M. Black, K. Muandet, and S. Tang. Grasping field: Learning implicit representations for human grasps. *arXiv preprint arXiv:2008.04451*, 2020.

[40] S. Khamis, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon. Learning an efficient model of hand shape variation from depth images. pages 2540–2548, 2015.

[41] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[42] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

[43] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[44] T. H. N. Le, K. G. Quach, C. Zhu, C. N. Duong, K. Luu, and M. Savvides. Robust hand detection and classification in vehicles and in the

wild. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1203–1210. IEEE, 2017.

[45] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[46] M. Lee, S. Lee, S. Son, G. Park, and N. Kwak. Motion feature network: Fixed motion filter for action recognition. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 387–403, 2018.

[47] T. Lee and T. Hollerer. Multithreaded hybrid feature tracking for markerless augmented reality. *IEEE transactions on visualization and computer graphics*, 15(3):355–368, 2009.

[48] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[49] S. Lim, I. Kim, T. Kim, C. Kim, and S. Kim. Fast autoaugment. *arXiv preprint arXiv:1905.00397*, 2019.

[50] J. Liu, H. Ding, A. Shahroudy, L.-Y. Duan, X. Jiang, G. Wang, and A. C. Kot. Feature boosting network for 3d pose estimation. *IEEE transactions on pattern analysis and machine intelligence*, 42(2):494–501, 2019.

[51] L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[52] M. Madadi, S. Escalera, A. Carruesco, C. Andujar, X. Baró, and J. Gonzàlez. Top-down model fitting for hand pose recovery in sequences of depth images. *Image and Vision Computing*, 79:63–75, 2018.

[53] A. Makris, N. Kyriazis, and A. A. Argyros. Hierarchical particle filtering for 3D hand tracking. pages 8–17, 2015.

[54] J. Malik, A. Elhayek, F. Nunnari, K. Varanasi, K. Tamaddon, A. Heloir, and D. Stricker. Deephps: End-to-end estimation of 3d hand pose and shape by learning from synthetic depth. In *2018 International Conference on 3D Vision (3DV)*, pages 110–119. IEEE, 2018.

[55] A. Markussen, M. R. Jakobsen, and K. Hornbæk. Vulture: a mid-air word-gesture keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1073–1082, 2014.

[56] 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. pages 49–59, 2018.

[57] F. Mueller, M. Davis, F. Bernard, O. Sotnychenko, M. Verschoor, M. A. Otaduy, D. Casas, and C. Theobalt. Real-time pose and shape reconstruction of two interacting hands with a single depth camera. *ACM Transactions on Graphics (TOG)*, 38(4):1–13, 2019.

[58] 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. pages 1284–1293, 2017.

[59] M. Oberweger, G. Riegler, P. Wohlhart, and V. Lepetit. Efficiently creating 3D training data for fine hand pose estimation. pages 4957–4965, 2016.

[60] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3D tracking of hand articulations using kinect. 1(2):3, 2011.

[61] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. pages 2088–2095, 2011.

[62] S. Ozair and Y. Bengio. Deep directed generative autoencoders. *arXiv preprint arXiv:1410.0630*, 2014.

[63] P. Panteleris and A. Argyros. Back to RGB: 3D tracking of hands and hand-object interactions based on short-baseline stereo. pages 575–584, 2017.

[64] P. Panteleris, I. Oikonomidis, and A. Argyros. Using a single RGB frame for real time 3D hand pose estimation in the wild. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 436–445. IEEE, 2018.

[65] D. Pathak, P. Krahenbuhl, J. Donahue, T. Darrell, and A. A. Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016.

[66] T. Piumsomboon, A. Clark, M. Billinghurst, and A. Cockburn. User-defined gestures for augmented reality. In *IFIP Conference on Human-Computer Interaction*, pages 282–299. Springer, 2013.

[67] R. Remilekun Basaru, G. Slabaugh, E. Alonso, and C. Child. Hand pose estimation using deep stereovision and markov-chain monte carlo. pages 595–603, 2017.

[68] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *2012 IEEE Conference*

on Computer Vision and Pattern Recognition, pages 1194–1201. IEEE, 2012.

[69] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6):245:1–245:17, Nov. 2017.

[70] R. Rosales, V. Athitsos, L. Sigal, and S. Sclaroff. 3D hand pose reconstruction using specialized mappings. 1:378–385, 2001.

[71] J. Schlüter and T. Grill. Exploring data augmentation for improved singing voice detection with neural networks. In *ISMIR*, pages 121–126, 2015.

[72] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, et al. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3633–3642, 2015.

[73] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. pages 1145–1153, 2017.

[74] J. Song, G. Sörös, F. Pece, S. R. Fanello, S. Izadi, C. Keskin, and O. Hilliges. In-air gestures around unmodified mobile devices. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 319–329, 2014.

[75] A. Spurr, J. Song, S. Park, and O. Hilliges. Cross-modal deep variational hand pose estimation. pages 89–98, 2018.

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

33rd Annual ACM Conference on Human Factors in Computing Systems, pages 3643–3652, 2015.

[77] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt. Fast and robust hand tracking using detection-guided optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3221, 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 *European Conference on Computer Vision*, pages 294–310. Springer, 2016.

[79] S. Sridhar, H. Rhodin, H.-P. Seidel, A. Oulasvirta, and C. Theobalt. Real-time hand tracking using a sum of anisotropic gaussians model. 1:319–326, 2014.

[80] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 824–832, 2015.

[81] M. Sundermeyer, R. Schlüter, and H. Ney. Lstm neural networks for language modeling. In *Thirteenth annual conference of the international speech communication association*, 2012.

[82] D. Tang, H. Jin Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3D articulated hand posture. pages 3786–3793, 2014.

[83] J. Taylor, R. Stebbing, V. Ramakrishna, C. Keskin, J. Shotton, S. Izadi, A. Hertzmann, and A. Fitzgibbon. User-specific hand modeling from

monocular depth sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 644–651, 2014.

[84] B. Tekin, F. Bogo, and M. Pollefeys. H+ o: Unified egocentric recognition of 3d hand-object poses and interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4511–4520, 2019.

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

[86] L. Wei, A. Xiao, L. Xie, X. Chen, X. Zhang, and Q. Tian. Circumventing outliers of autoaugment with knowledge distillation. *arXiv preprint arXiv:2003.11342*, 2(8), 2020.

[87] Y. Wu, W. Ji, X. Li, G. Wang, J. Yin, and F. Wu. Context-aware deep spatiotemporal network for hand pose estimation from depth images. *IEEE transactions on cybernetics*, 2018.

[88] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. In *Advances in neural information processing systems*, pages 802–810, 2015.

[89] L. Yang and A. Yao. Disentangling latent hands for image synthesis and pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9877–9886, 2019.

[90] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *IEEE transactions on pattern analysis and machine intelligence*, 35(12):2878–2890, 2012.

[91] F. Yin, X. Chai, and X. Chen. Iterative reference driven metric learning for signer independent isolated sign language recognition. In *European Conference on Computer Vision*, pages 434–450. Springer, 2016.

[92] S. Yuan, B. Stenger, and T.-K. Kim. RGB-based 3D hand pose estimation via privileged learning with depth images. *arXiv preprint arXiv:1811.07376*, 2018.

[93] S. Yuan, Q. Ye, B. Stenger, S. Jain, and T.-K. Kim. Bighand2. 2m benchmark: Hand pose dataset and state of the art analysis. pages 4866–4874, 2017.

[94] 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.

[95] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang. A hand pose tracking benchmark from stereo matching. pages 982–986, 2017.

[96] 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.

[97] X. Zhang, Q. Wang, J. Zhang, and Z. Zhong. Adversarial autoaugment. *arXiv preprint arXiv:1912.11188*, 2019.

[98] Z. Zhang, S. Xie, M. Chen, and H. Zhu. Handaugment: A simple data augmentation method for depth-based 3d hand pose estimation. *arXiv preprint arXiv:2001.00702*, 2020.

[99] W. Zhao, J. Chai, and Y.-Q. Xu. Combining marker-based mocap and RGB-d camera for acquiring high-fidelity hand motion data. pages 33–42, 2012.

[100] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[101] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[102] C. Zimmermann and T. Brox. Learning to estimate 3D hand pose from single RGB images. pages 4903–4911, 2017.

[103] 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 *Proceedings of the IEEE International Conference on Computer Vision*, pages 813–822, 2019.

초 록

2D 이미지에서 사람의 손 모양과 포즈를 인식하고 구현흐는 연구는 각 손가락 조인트들의 3D 위치를 검출하는 것을 목표로한다. 손 포즈는 손가락 조인트들로 구성되어 있고 손목 관절부터 MCP, PIP, DIP 조인트들로 사람 손을 구성하는 신체적 요소들을 의미한다. 손 포즈 정보는 다양한 분야에서 활용될수 있고 손 제스쳐 감지 연구 분야에서 손 포즈 정보가 매우 훌륭한 입력 특징 값으로 사용된다.

사람의 손 포즈 검출 연구를 실제 시스템에 적용하기 위해서는 높은 정확도, 실시간성, 다양한 기기에 사용 가능하도록 가벼운 모델이 필요하고, 이것을 가능케 하기 위해서 학습한 인공신경망 모델을 학습하는데에는 많은 데이터가 필요로 한다. 하지만 사람 손 포즈를 측정하는 기계들이 꽤 불안정하고, 이 기계들을 장착하고 있는 이미지는 사람 손 피부 색과는 많이 달라 학습에 사용하기가 적절하지 않다. 그러기 때문에 본 논문에서는 이러한 문제를 해결하기 위해 인공적으로 만들어낸 데이터를 재가공 및 증량하여 학습에 사용하고, 그것을 통해 더 좋은 학습성과를 이루려고 한다.

인공적으로 만들어낸 사람 손 이미지 데이터들은 실제 사람 손 피부색과는 비슷할지언정 디테일한 텍스쳐가 많이 달라, 실제로 인공 데이터를 학습한 모델은 실제 손 데이터에서 성능이 현저히 많이 떨어진다. 이 두 데이타의 도메인을 줄이기 위해서 첫번째로는 사람손의 구조를 먼저 학습 시키기위해, 손

모션을 재가공하여 그 움직임 구조를 학스한 시간적 정보를 뺀 나머지만 실제 손 이미지 데이터에 학습하였고 크게 효과를 내었다. 이때 실제 사람 손모션을 모방하는 방법론을 제시하였다.

두번째로는 두 도메인이 다른 데이터를 네트워크 피쳐 공간에서 align시켰다. 그뿐만아니라 인공 포즈를 특정 데이터들로 augment하지 않고 네트워크가 많이 보지 못한 포즈가 만들어지도록 하나의 확률 모델로서 설정하여 그것에서 샘플링하는 구조를 제안하였다.

본 논문에서는 인공 데이터를 더 효과적으로 사용하여 annotation이 어려운 실제 데이터를 더 모으는 수고스러움 없이 인공 데이터들을 더 효과적으로 만들어 내는 것 뿐만 아니라, 더 안전하고 지역적 특징과 시간적 특징을 활용해서 포즈의 성능을 개선하는 방법들을 제안했다. 또한, 네트워크가 스스로 필요한 데이터를 찾아서 학습할수 있는 자동 데이터 증량 방법론도 함께 제안하였다. 이렇게 제안된 방법을 결합해서 더 나은 손 포즈의 성능을 향상 할 수 있다.

주요어: 3차원 손 포즈 인식

학번: 2016-30732

감사의 글

무엇보다 제가 박사학위를 받는동안 기다려주시고 저를 가장 가까이에서 응원해주신 저희 부모님께 감사드립니다. Machine Learning/Computer Vision 분야로 이끌어주시고 논문지도와 연구자의 덕목을 갖게 해주신 곽노준 교수님께 깊은 감사를 드립니다. 덕분에 박사 학위를 공부하는 동안 깊이 있는 탐구와 창의적인 해석을 배울수 있게 되었습니다. 논문의 작성 전략과 완성도를 이끌어 내주시고, 전문 분야를 함께 구축하며 자신감을 키워주신 장형진 교수님도 감사드립니다. 저의 박사학위논문 심사를 맡아주신 이교구, 박재흥, 이원종, 이민식 교수님들 또한 감사드립니다. 209호 룸메이트들 지수형, 김지수, 승의형, 호준이, 지훈이형, 규정이형 같이 방에서 얘기 나누면서 아이디어 토의하고 일상 얘기 많이했었던 덕분에 많이 힘이 되었습니다. 같이 논문 협력하며 썼던 시명이형, 성욱이, 재석이; 항상 생각이 막히면 찾아가서 얘기나누던 재영이형, 효진누나, 장호; 날카로운 논문 리뷰를 하던 상호형; 어리지만 배울점이 많던 자연이, 예지, 기윤이, 인섭이; 커리어와 인생에 대해서 자주 얘기나누던 지은이 누나, 성헌이형, 지혜누나, 서형이; 그 외 MIPAL 연구실에서 함께 동고동락하며 열정적으로 연구하던 친구들 모두 다 감사드립니다. 이 연구실에서 했던 공부와 시간이 정말 값지고 보람찼습니다. 졸업 후에도 함께 했던 친구들과 교수님들께 좋은 귀감이 될수 있도록 최선을 다하며, 항상 공정하게 선한 영향을 끼치는 사람이 되도록 노력하겠습니다. 저의 한국이름 '한열'은 유년기에 없어졌지만, 대학원 생활과 함께 '한열'이라는 이름으로 어릴적부터 동경해오던 한국에서의 삶을 원없이 즐길수 있어서 행복했습니다.