

T.C.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**HAND POSE ESTIMATION AND RENDERING FOR
AUGMENTED REALITY**

IKRAM KOURBANE
A THESIS SUBMITTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
DEPARTMENT OF COMPUTER ENGINEERING

GEBZE
2022

T.C.
GEBZE TECHNICAL UNIVERSITY
GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

**HAND POSE ESTIMATION AND
RENDERING FOR AUGMENTED REALITY**

IKRAM KOURBANE
A THESIS SUBMITTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY
DEPARTMENT OF COMPUTER ENGINEERING

THESIS SUPERVISOR
ASSIST. PROF. DR. YAKUP GENC

GEBZE
2022

T.C.
GEBZE TEKNİK ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ARTIRILMIŞ GERÇEKLİK İÇİN
İNSAN ELİ TAHMİNİ VE
GÖRSELLEŞTİRME

İKRAM KOURBANE
DOKTORA TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

DANIŞMANI
DR. ÖĞR. ÜYESİ YAKUP GENÇ

GEBZE
2022

GEBZE TEKNİK ÜNİVERSİTESİ	DOKTORA JÜRİ ONAY FORMU
----------------------------------	--------------------------------

GTÜ Fen Bilimleri Enstitüsü Yönetim Kurulu'nun/...../..... tarih ve/..... sayılı kararıyla oluşturulan jüri tarafından/...../..... tarihinde tez savunma sınavı yapılan'ın tez çalışması Anabilim Dalında DOKTORA tezi olarak kabul edilmiştir.

JÜRİ

ÜYE

(TEZ DANIŞMANI) : Dr. Öğr. Üyesi Yakup Genç

ÜYE : Prof. Dr. Yusuf Sinan Akgül

ÜYE : Doç. Dr. Ayşe Betül Oktay

ÜYE : Prof. Dr. Behçet Uğur Töreyn

ÜYE : Prof. Dr. Erchan APTOULA

ONAY

Gebze Teknik Üniversitesi Fen Bilimleri Enstitüsü Yönetim Kurulu'nun/...../..... tarih ve/..... sayılı kararı.

İMZA/MÜHÜR

SUMMARY

Monocular RGB-based 3D hand pose estimation is crucial for numerous augmented reality applications. This task is challenging due to occlusion and depth ambiguities. Recent methods use GCN-based (Graph Convolutional Networks) to model kinematic dependencies between hand joints. These techniques estimate a scale-normalized root-relative 3D pose, and the absolute coordinates are unknown. Also, they do not enforce any constraints to restrict the regression model. Meanwhile, they employ large models with inefficient loss functions requiring high computation time. This thesis tackles the issues and presents efficient 2D and 3D hand pose estimation methods. The first method is a multi-scale heatmap regression approach for 2D hand pose estimation. The second study is a GCN-based framework that classifies the joints based on their 2D/3D spatial locality. It learns per-input joint relation constraints that guide the 3D regression task. Our third method exploits the cropped hand and the global scene image, which provides clues about the hand scale and location in the camera space. Also, it presents a new loss function to simplify the regression and speed up the running time. Extensive experiments on five public datasets show that our efficient models estimate accurate global 3D hand poses and outperform the state-of-the-art. The last study presents a GANs-based (generative adversarial network) method that translates a real hand image to a synthetic one and renders it with an object image. Qualitative and quantitative results show that our approach generates visually appealing rendered hand images and outperforms different GANs-based frameworks.

Keywords: 3D hand pose estimation, Constrained learning, Deep learning, Graph convolution networks, Generative adversarial networks, Hand-object rendering.

ÖZET

Renkli (RGB) bir imgeden üç boyutlu (3B) elin yeri ve doğrultusunun tahmini, birçok artırılmış gerçeklik (AG) uygulaması için çok elzemdir. Elin tam görünmemesi ve sahne derinlik belirsizlikleri nedeniyle doğru tahmin zordur. Güncel yöntemler, el eklemleri arasındaki kinematik bağımlılıkları modellemek için çizge evrimsel ağ (ÇGN) kullanmaktadır. Bununla birlikte, bu teknikler, ölçeğe göre normalleştirilmiş bir referansa görece 3B poz tahmin eder ve mutlak koordinatları tahmin etmez. Ayrıca, regresyon problemini sınırlandırmak için herhangi bir kısıtlamadan getirmezler. Bu arada, yüksek hesaplama süresi gerektiren verimsiz kayıp fonksiyonlarına sahip büyük modeller kullanırlar. Bu tezde, bahsedilen sorunlar ele alınmakta ve etkili 2B ve 3B el poz tahmin yöntemleri sunulmaktadır. İlk yöntem, 2B el poz tahmini için çok ölçekli bir sıcaklık haritası regresyon yaklaşımıdır. İkinci çözüm, eklemleri 2B/3B uzamsal konumlarına göre sınıflandıran ÇGN tabanlı bir çerçevedir. Model regresyon görevine rehberlik eden girdi başına ortak ilişkiler kısıtlamasını öğrenir. Üçüncü metodumuz, kamera alanındaki el ölçeği ve konumu hakkında ipuçları sağlayan kırpılmış eli ve küresel sahne görüntüsünü kullanır. Ayrıca regresyonu basitleştirmek ve çalışma süresini hızlandırmak için yeni bir kayıp fonksiyonu sunar. Beş genel veri kümesi ve yeni oluşturulan veri kümemiz üzerinde yapılan kapsamlı deneyler, verimli modellerimizin doğru küresel 3B el pozisyona ve oryantasyonunu tahmin ettiğini ve literatürden daha iyi performans sağladıklarını göstermektedir. Son olarak doğrudan AG görselleştirme için girdi gerçek el görüntüsünü sentetik bir görüntüye çeviren ve onu nesne görüntüsüyle birleştiren çekişmeli üretici ağ (GAN) tabanlı bir yöntem sunmaktadır. Niteliksel ve niceliksel sonuçlar, yaklaşımımızın görsel olarak gerçekçi işlenmiş el görüntüleri oluşturduğunu ve literatürdeki GAN tabanlı çözümlerden daha iyi başarımla sergilediğini gösteriyor.

Anahtar Kelimeler: 3B el poz tahmini, Kısıtlı öğrenme, Derin öğrenme, Çizge evrimsel ağlar, Çekişmeli üretici ağlar, El-nesne görselleştirme.

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude to my advisor Dr. Yakup Genç whose expertise was invaluable in formulating the research questions and methodology. His extensive knowledge and valuable criticism have deeply inspired me. I will always appreciate his limitless patience and continuous support throughout my Ph.D. studies.

Besides my advisor, I would like to thank the rest of my thesis committee: Prof. Yusuf Sinan Akgül and Dr. Betül Oktay. Their insightful comments and constructive suggestions incited me to improve the thesis contributions and widen my research from various perspectives.

My gratitude extends to the professors of the GTU computer engineering department. Their wonderful lectures and plentiful experience helped me to expand my knowledge.

My sincere thanks also go to my research colleagues for their collaboration, encouragement, and social meetings.

I would also thank the Turkish Scholarships organization (YTB) for its financial support during my Ph.D. study.

Last but not the least, I would like to thank my lovely parents and all the rest of my family for their unconditional love, wise counsel, and for supporting me spiritually throughout writing this thesis and my life in general.

TABLE OF CONTENTS

	<u>Page</u>
SUMMARY	v
ÖZET	vi
ACKNOWLEDGEMENTS	vii
TABLE OF CONTENTS	viii
LIST OF ABBREVIATIONS	xii
LIST OF FIGURES	xiv
LIST OF TABLES	xvii
1. INTRODUCTION	1
1.1. Motivations	4
1.2. Contributions	6
1.2.1. Robust Hand Detection and 2D Pose Estimation	6
1.2.2. Accurate Root-relative 3D Hand Pose	7
1.2.3. Accurate Absolute 3D Hand Pose Estimation	7
1.2.4. GANs-based Approach for Hand-object Rendering	8
1.3. Dissertation Outline	8
2. BACKGROUND	10
2.1. Graph Convolution Networks	10
2.2. Generative Adversarial Networks	11
2.3. Hand Detection	13
2.4. 2D Hand Pose Estimation	13
2.5. Modality-based Classification of 3D Hand Pose Estimation	14
2.5.1. Depth-based Methods	15
2.5.2. Multi-views-based Methods	15
2.5.3. Monocular RGB-based Methods	16
2.6. Root-relative 3D Hand Pose Estimation	16
2.6.1. Discriminative Methods	17
2.6.2. Generative Methods	18
2.7. Absolute 3D Hand Pose Estimation	18
2.8. Hand-object Rendering and Image Composition Methods	19

3. A MULTI-SCALE HEATMAP REGRESSION FOR 2D HAND POSE ESTIMATION	21
3.1. Introduction	21
3.2. Skeleton Detection and Hand Bounding Box Localization	23
3.3. Multi-scale Heatmaps Regression for 2D Hand Pose Estimation	24
3.4. Datasets and Evaluation Metrics	27
3.5. Experiments and Results	28
3.5.1. Implementation Details	28
3.5.2. Hand Detection and Bounding Box Estimation Results	29
3.5.3. Pose Estimation Results	32
3.6. Summary	34
4. A HYBRID CLASSIFICATION-REGRESSION APPROACH FOR 3D HAND POSE ESTIMATION	35
4.1. Introduction	35
4.2. Methodology	37
4.2.1. Feature Maps Extraction	38
4.2.2. Classification Modules	39
4.2.3. Relations Function	40
4.2.4. Regression Modules	41
4.2.5. Refinement Module	42
4.2.6. Loss Functions	43
4.3. Experimental Settings	44
4.3.1. Implementation Details	44
4.3.2. Datasets and Evaluation Metrics	45
4.3.3. Hand Model Representation	46
4.4. Results and Discussions	46
4.4.1. Effect of our Per-pose Adjacency Matrix	47
4.4.2. Effect of the Number of Classes	49
4.4.3. Regression-regression vs. Classification-regression	50
4.4.4. Effect of the Refinement Network	52
4.4.5. Qualitative Results	53
4.4.6. Comparisons with State-of-the-art	57
4.4.7. Computation Complexity	58
4.5. Summary	58

5. A GRAPH-BASED ABSOLUTE 3D HAND POSE ESTIMATION USING A SINGLE RGB IMAGE	59
5.1. Introduction	59
5.2. Methodology	61
5.2.1. Architecture	61
5.2.2. Hand Pose Regression using Graph Convolution Network	63
5.2.3. Loss Functions	64
5.2.4. Features Matching	65
5.2.5. Optimization	65
5.3. Experimental Setting	66
5.3.1. Datasets	66
5.3.2. Metrics	67
5.3.3. Implementation Details	68
5.4. Experimental Results	68
5.4.1. Analysis of the Global and Local Modules	69
5.4.2. Effect of the Feature Matching Loss	71
5.4.3. Comparison with the State-of-the-art Methods	75
5.4.4. Qualitative Results	77
5.4.5. Computation Complexity	79
5.4.6. Absolute 3D Hand Pose Estimation for Hands in the Outdoor	80
5.5. Summary	81
6. GAN-BASED HAND-OBJECT RENDERING	82
6.1. Introduction	82
6.2. Real hand-object rendering	82
6.2.1. Challenges of Rendering Module Development	82
6.2.2. Single Class Object Rendering	84
6.2.3. Multi-class Object Rendering	85
6.2.4. Composer-Decomposer architecture for hand-object rendering	94
6.3. Synthetic Object- hand Rendering	95
6.4. Synthetic Object- real Hand Rendering	98
6.4.1. Overview	98
6.4.2. Hand Segmentation	98
6.4.3. Translation between Synthetic and Real Hands	99
6.4.4. Architecture for Hand-object Rendering in AR Applications	100

6.5. Summary	102
7. COMPARISON OF THE PROPOSED 3D HAND POSE ESTIMATION APPROACHES	103
7.1. Introduction	103
7.2. Root-relative 3D Hand Pose Estimation Comparison	103
7.3. Absolute 3D Hand Pose Estimation Comparison	104
7.4. Combination of HCRA and GAAH	104
7.5. Summary	106
8. CONCLUSIONS	107
8.1. Summary of the Contributions	107
8.2. Limitation and Future Work	109
REFERENCES	111
BIOGRAPHY	120
APPENDICES	121

LIST OF ABBREVIATIONS

<u>Abbreviations</u>	<u>Descriptions</u>
A	: The adjacency matrix of graph
ANN	: Adaptative Nearest Neighbor
AUC	: Area Under the Curve
AR	: Augmented Reality
BN	: Batch Normalization
CE	: Cross-Entropy loss function
CNNs	: Convolutional Neural Networks
Conv2D	: 2D Convolution
CPU	: Central Processing Unit
DIP	: Distal Interphalangeal
DL	: Deep Learning
DNNs	: Deep Neural Networks
D	: Degree matrix of graph
DoF	: Degree of Freedom
EPE	: End-Point Error
FM	: Feature Matching
FPHA	: First-Person Hand Actions Benchmark
GANs	: Generative Adversarial Networks
GFLOPS	: Giga Floating-point Operations
GPU	: Graphic Processing Unit
GCNs	: Graph Convolution networks
GCL	: Graph Convolution Layer
GTHD	: GTU Benchmark
G	: Graph
GAAH	: Graph-based Approach for Absolute Hand pose estimation
HCI	Human Computer Interaction
HCRA	: Hybrid Classification-Regression Approach
I	: Identity matrix
IoU	: Intersection over Union
KNN	: K Nearest Neighbor
LeakyReLU	: Leaky Rectified Linear Unit

M	: Million
MAE	: Mean Absolute Error
Ms	: Millisecond
MSE	: Mean Squared Error
MCP	: Metacarpophalangeal
MHP	: Multiview Hand Pose Benchmark
MJPE	: Mean Joint Pixel Error
MUNIT	: Multi-modal Unsupervised Image-to-image Translation
PIP	: Proximal Interphalangeal
PCK	: Percentage of Correct Keypoints
PSO	: Particle Swarm Optimization
ReLU	: Rectified Linear Unit
RHD	: Rendered Hand Pose Benchmark
Res-block	: ResNet Block
RoI	: Region of Interest
STB	: Stereo Hand Pose Tracking Benchmark
TConv2D	: Transposed 2D Convolution
UNIT	: Unsupervised Image-to-image Translation
VR	: Virtual Reality
VAEs	: Variational Auto-Encoders

LIST OF FIGURES

<u>Figure No:</u>	<u>Page</u>
1.1: Description of the two models in hand pose estimation.	1
1.2: Two proposed approaches for hand-object rendering in AR.	3
1.3: Example of the image composition task.	6
2.1: Example of adjacency matrix normalization in graphs.	11
2.2: Information propagation in graphs to manipulate irregular structures.	11
2.3: Alternative training between the generator and the discriminator.	12
2.4: Classification of vision-based 3D hand pose estimation methods.	14
3.1: Faster-RCNN vs. our proposed hand detection and bounding box localization approach.	23
3.2: The architecture of the proposed 2D hand pose estimation approach.	26
3.3: Examples of our hand dataset images. It includes the hand bounding boxes and 21 joint annotations in 2D and 3D spaces.	28
3.4: The results of the skeleton estimation and the bounding box localization on the GTHD dataset using thick and thin skeleton representations.	29
3.5: The results of the skeleton estimation and the bounding box localization on the LSMV dataset using thin and thick skeleton representations.	30
3.6: The impact of the threshold selection on hand detection results.	31
3.7: Qualitative results for the 2D hand pose estimation task.	32
3.8: Quantitative comparison of the proposed 2D hand pose estimation using PCK metric for GTHD (left) and LSMV (right).	34
4.1: Schematic overview of our 3D hand pose estimation framework.	37
4.2: We classify each joint position into a predefined number of classes.	40
4.3: Algorithm of the relations function. It identifies adjacent joints belonging to the same class.	41
4.4: Algorithm of the adaptative nearest neighbor algorithm that identifies adjacent joints.	43
4.5: The description of our hand model.	46
4.6: Qualitative comparison of 3D hand pose estimation between the predefined, the global, and our per-pose adjacency matrices.	48
4.7: Effect of the number of classes on the joint relationship constraints.	50

4.8:	Qualitative comparison of regression-regression and our classification-regression approaches.	51
4.9:	The intermediate outputs of our method on STB and RHD datasets.	52
4.10:	3D hand pose estimation results on the STB dataset.	53
4.11:	3D hand pose estimation results on the RHD dataset.	54
4.12:	Comparison with the state-of-the-art methods on the STB dataset	54
4.13:	Comparison with the state-of-the-art methods on the RHD dataset.	55
5.1:	Schematic overview of our proposed approach for monocular RGB-based absolute 3D hand pose estimation.	61
5.2:	Illustration of two samples with the same relative 3D pose. Unlike the cropped hand image, the global scene preserves the size of the hands that gives a clue about the distance from the camera.	62
5.3:	Our feature matching loss measures the distance between the pairwise distance matrices of the cropped hand image feature maps and root-relative 3D hand pose.	64
5.4:	Ablation studies of the proposed approach.	70
5.5:	The impact of the proposed FM loss on the STB, MHP, and GTHD datasets.	72
5.6:	Comparison with the state-of-the-art methods on the STB dataset.	74
5.7:	Comparison with the state-of-the-art methods on the MHP dataset.	74
5.8:	Comparison with the state-of-the-art methods on the RHD dataset using 3D PCK.	75
5.9:	Comparison with the state-of-the-art methods on the FPHA dataset.	75
5.10:	The intermediate outputs of the proposed method.	78
5.11:	Absolute 3D hand pose estimation results on STB, MHP, and GTHD datasets.	79
5.12:	Absolute 3D hand pose estimation for two hands in outdoor setting.	81
6.1:	Failure cases of the unsupervised image-to-image translation between an empty hand and a hand holding an object.	85
6.2:	Failure cases of the multi-modal unsupervised image-to-image translation task between hand and rendered domains.	86
6.3:	Our training and test schemas are different from MUNIT since we use two inputs. Also, our target distribution has different geometry and appearance than the source.	86

6.4:	Unsupervised real dataset for hand-object composition task. It includes three subsets: object, empty hand, and interacted hand.	87
6.5:	Our MUNIT-based approach for real hand-object rendering.	88
6.6:	Reconstruction loss within the same domain.	89
6.7:	Cycle consistency loss for rendered and hand domains.	91
6.8:	We optimize L1 loss between our input hand image and the model output to preserve the hand pose.	92
6.9:	Results of the hand-to-rendered translation using our proposed MUNIT-based architecture.	93
6.10:	Results of the rendered-to-hand translation using our proposed MUNIT-based architecture.	93
6.11:	Composition-decomposition method for real hand-object rendering.	94
6.12:	Results of real hand-object rendering using composition-decomposition architecture.	95
6.13:	The proposed GANs-based architecture for synthetic hand-object rendering.	96
6.14:	Results of the proposed GANs-based synthetic hand-object rendering approach on the ObMan dataset.	97
6.15:	Hand segmentation results using synthetic and real hand images.	99
6.16:	Image-to-image translation between real and synthetic hand images using Cycle-GANs.	100
6.17:	Image-to-image translation between real and synthetic using Cycle-GANs and the proposed loss functions.	100
6.18:	Hand-object rendering approach for AR application.	101
6.19:	Hand-object rendering results for AR application.	102
7.1:	Absolute 3D hand pose estimation results on the MHP dataset.	105
8.1:	Additional results for hand detection and bounding box localization.	121
8.2:	Direct regression vs. our multi-stage heatmaps regression method.	122
8.3:	2D hand pose estimation results on STB and RHD datasets.	122
8.4:	Algorithm represents the creation of the 2D ground-truth classes of each joint in a cropped hand.	123

LIST OF TABLES

<u>Table No:</u>	<u>Page</u>
3.1: Bounding box evaluation on LSMV and our GTHD dataset with IOU.	31
3.2: Comparison with the state-of-the-art methods on the LSMV datasets with PCK metric.	32
3.3: Comparison with the state-of-the-art methods on GTHD and LSMV datasets with MPE metric.	33
4.1: The architecture of the modified ResNet-10 network.	38
4.2: Comparison of different adjacency matrices in the coarse stage on STB and RHD datasets with EPE [mm] and AUC metrics.	48
4.3: Ablation studies of different baselines with EPE [mm] and AUC metrics on the RHD dataset.	49
4.4: Effect of the number of classes on the 3D hand pose estimation results with EPE [mm] and AUC metrics.	50
4.5: Effect of the number of classes on our 2D/3D classifiers. Each triplet in the classification metrics represents the accuracy, recall, and precision.	50
4.6: Comparison with the state-of-the-art methods on RHD and STB datasets using Mean EPE [pixel].	55
4.7: Comparison with the state-of-the-art methods on RHD and STB datasets using AUC.	56
4.8: Comparison with the state-of-the-art methods on RHD and STB datasets using Mean EPE [mm].	56
5.1: The architecture of the customized ResNet-10 network.	63
5.2: Ablation studies of 3D hand pose estimation on STB, MHP, RHD, and GTHD datasets with EPE [mm] and AUC metrics.	69
5.3: Ablation studies of the root-relative and the absolute pose estimation stages on STB, MHP, and GTHD datasets with EPE [mm] metric.	70
5.4: Comparison with the state-of-the-art methods on STB, MHP, RHD, and FPHA datasets using Mean EPE [mm].	73
5.5: Comparison with the state-of-the-art methods on RHD and STB datasets using Mean EPE [pixel].	73

5.6:	Running time and performance comparison between ResNet-50 and our customized ResNet-10	80
6.1:	Quantitative comparison between the GANs-based baseline and our approach for hand-object rendering.	97
6.2:	Quantitative results for hand segmentation task using synthetic and real images.	99
7.1:	Quantitative comparison between the proposed monocular RGB-based 3D hand pose estimation methods for the root-relative.	103
7.2:	Quantitative comparison between the proposed monocular RGB-based 3D hand pose estimation methods for the root-relative.	104
7.3:	Running time comparison of the proposed monocular RGB-based 3D hand pose estimation methods on NVidia Titan XP GPU.	106

1. INTRODUCTION

Augmented reality (AR) inserts digital elements into our real physical world. These computer-generated objects provide additional knowledge about the scene to enhance the experience of the user. With the recent progress of display technologies, AR applications are becoming popular. They are used in many domains, including marketing, education, and games. These applications usually use mice and keyboards as interaction tools, which may restrict object manipulation.

The hand is the most important and intuitive tool that allows humans to interact with the environment. Thus, hand pose estimation is one of the fundamental well-stated problems in computer vision. It plays an essential role in many applications, such as robotics, AR, virtual reality (VR), gesture recognition, and human-computer interaction (HCI).

Hand pose estimation finds the key points (finger joints and the palm) from a depth or an RGB image. There are two main ways to define the hand pose: 16 key points and 21 key points (Figure 1.1). The first representation consists of one key point on the palm center and three key points for each of the five fingers (thumb, index, middle, ring, and pinky). The key points are located at the centers of three connected bones. In the second representation, the hand is described by 21 joints. The wrist represents the hand palm and four joints (tip, dip, pip, and mcp) per finger. Our thesis follows 21 key points representations since it is usually used in monocular RGB-based hand pose estimation methods. We define each joint position by $joint_{2D} = (x, y)$ and $joint_{3D} = (x, y, z)$ for 2D and 3D coordinates, respectively.

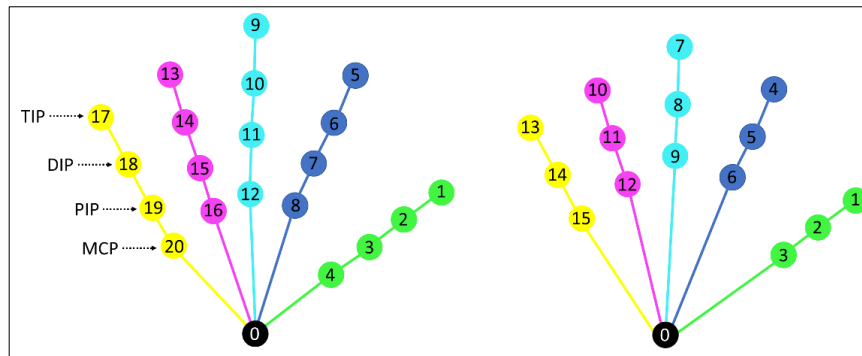


Figure 1.1: Description of the two models in hand pose estimation.

Although studied extensively in the literature, accurate 3D hand pose estimation remains challenging due to several problems, including:

- The high degree of freedom (DoF) of the hand,
- Self-similarity: homogeneous color distribution of the hand skin,
- Self-occlusions and object-occlusion,
- Fast hand motion,
- Different shapes and sizes of the hand.

Early works in hand pose estimation use magnetic sensing devices to track the hand using gloves and visual markers [3]. But these types of solutions are expensive and require complex calibration and setup. Commercial systems are moving to vision-based 3D hand pose estimation methods since they are natural and easy to use.

The rapid development of deep learning techniques solves complex computer vision problems and outperforms conventional methods in many challenging tasks, including object classification [4], object segmentation [5], and object detection [6]. Hand pose estimation is not an exception and deep convolution neural networks (CNNs) [7] have been applied successfully in [8–10]. These studies address the scenarios where the hand is tracked via an RGB-D camera. However, depth-enhanced data is not available everywhere, and they need an overhead setup to utilize. These sensors require high-power consumption and are limited to indoor environments restricting many applications. Other approaches alleviate occlusion problems using multiple RGB cameras located at different viewpoints [9,11,12]. These techniques are difficult to set up and cannot operate in unconstrained scenes (e.g., community videos). As a result, 3D hand pose estimation from a single RGB image has gained considerable attention since it is more accessible (cheaper) and does not need any setup overhead. This task is more challenging than the abovementioned approaches because it exhibits depth and scale ambiguities.

Learning-based methods address the 3D hand pose estimation problem by designing effective models trained on large-scale annotated datasets. We can divide them mainly into two major approaches: generative and discriminative. The former one adopts variational autoencoders (VAEs) [13] or generative adversarial networks (GANs) [14] to model the 3D hand pose distribution [9,15,16]. The methods of the second category use CNNs [7] to extract high-level features from the cropped hand

image to estimate the 3D pose [11,17–21]. After getting a reliable 3D hand pose, we can render it with the 3D object pose using AR techniques (Figure 1.2).

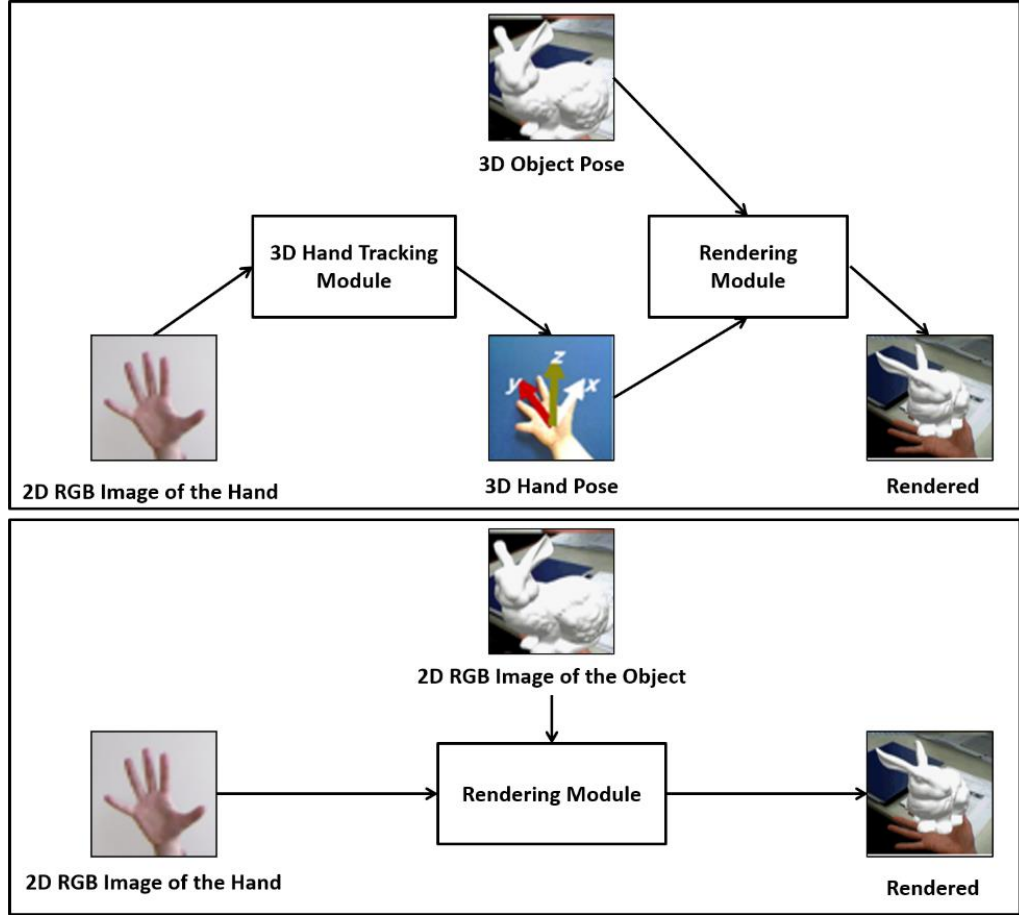


Figure 1.2: Two proposed approaches for hand-object rendering in AR.

Another alternative solution is to directly render the hand image and the virtual object image to output the final image showing the interaction. To the best of our knowledge, there is no current learning-based approach that performs this task. This is mainly due to the lack of datasets that include real hands interacting with virtual objects. Also, this task presents an ill-posed problem since the joint image should model the composition of multiple objects interacting in a 3D visual world. Existing GANs-based image composition methods were designed to achieve realistic rendering either from geometry space [22] or appearance space [23] but not in both. In particular, they involve transforming either a single object of interest (apples to oranges, horses to zebras, label to image, etc.) or changing the style and texture of the input image (day to night, etc.).

1.1. Motivations

Initially, learning-based 3D hand pose estimation methods require a large amount of accurate training data. Due to the difficulty of annotation, there is a limitation in the currently available datasets. Specifically, they are small-scale and do not have a large variation in hand poses, lighting, and backgrounds. Therefore, one of the first aims of this thesis is to build a large-scale dataset that includes RGB images with the corresponding hand bounding boxes, 2D and 3D annotations.

2D hand pose estimation also plays an essential role in sign language understanding and HCI. Existing RGB-based 2D hand pose estimation methods learn the joint locations from a single resolution, which is not suitable for different hand sizes. Thus, our primary study will focus on hand detection and 2D pose estimation tasks.

Current learning-based 3D hand pose estimation methods assume that the model can implicitly learn kinematic relationships between the joints. However, this task is challenging and requires explicit constraints to guide the model optimization. Inspired by the natural graph representation of the hand, recent studies use graph convolution networks (GCN) [24] to model skeleton constraints between joints [25–29]. These techniques use predefined or globally learned joint relationship constraints (adjacency matrix), which may fail to capture pose dependent constraints. Specifically, the predefined representation interprets joint relationships based on the hand skeleton [25–29]. Hence, despite the changes in the input images, this representation remains the same for all images across all datasets. Also, they do not exploit spatial relations between physically disconnected joints. The global learned joint relationship constraints remain the same for the images of the same dataset. Besides, it does not exploit spatial information between adjacent joints.

Most existing monocular RGB-based 3D hand pose estimation methods estimate a scale-normalized root-relative pose. They predict the positions of hand joints relative to a reference point located on the hand (e.g., the wrist). Thus, the global joint coordinates in the world coordinate system are unknown. They assume that the depth of the root joint and the global hand scale is provided at test time [9,11]. In contrast, absolute hand pose estimation finds the hand joint positions in camera-centered coordinates. This extends the application scope to AR/VR, where headsets require camera distance-aware hand pose estimation approaches to perform hand-object

interactions. Also, the 3D absolute pose is important in multi-hand scenes, where one 3D root-relative hand pose may represent different absolute 3D poses. Thus, the global joint coordinates should be identified to recover the spatial layout of the whole scene.

Current methods require expensive computation hardware limiting many applications in resource-constrained environments. Their high computation cost is mainly due to the usage of large models with inefficient loss functions. They only apply the reconstruction supervision during optimization. They do not add any loss function to simplify the 3D hand pose regression task.

Finally, we observe that all the studies aim to estimate an accurate 3D hand pose and render it with the 3D object pose to output the interacted hand image. To the best of our knowledge, there is no current vision-based method that composes the hand and the object images. Image composition is one of the challenging problems in the computer vision and graphics fields. It refers to overlaying objects from different images such that the new image has a realistic appearance, overlay semantic, and correct geometry. Recently GANs have shown great potential in realistic image synthesis [30], whereas most existing works address synthesis realism in either appearance space [23,31,32] or geometry space [22]. Hand-object rendering is more challenging than the traditional image composition problem. We can explain this by the fact that most of the image composition approaches are fully supervised. They have the triplet foreground/background /rendered since they use synthetic data. In our case, the input hand image is real and the object is virtual. Building such a dataset is impractical and requires enormous human supervision. Also, their optimization process is more restricted since the output is more constrained. For example, the glass in Figure 1.3 should be on the top of the eyes. In contrast, the hand can hold the same object in different ways. Finally, the background is fixed, it does not have many degrees of freedom like the hand. In the second part of our thesis, we aim to develop a robust GANs-based rendering of the hand (background object) holding an object (foreground object).

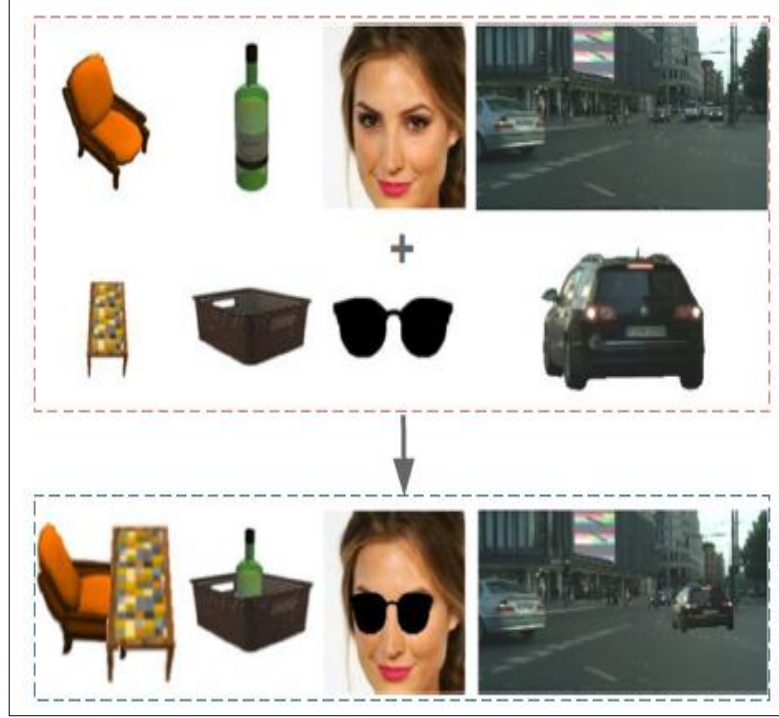


Figure 1.3: Example of the image composition task.

1.2. Contributions

In this thesis, we conduct several studies to estimate accurate 2D and 3D hand poses from a single RGB image. We also propose a generative-based approach that composes the hand and the object images to output the combination hand and the object. To achieve these aims, we build a new large-scale dataset of annotated RGB hand images. Meanwhile, we present new network architecture, optimization constraints, loss functions, and evaluation metrics. Our contributions can be summarized as follows:

1.2.1. Robust Hand Detection and 2D Pose Estimation

Our first study proposes a new deep learning-based framework that consists of two main modules. The first one presents a segmentation-based approach to detect the hand skeleton and localize the hand bounding box. The second module regresses the 2D joint locations through a multi-scale heatmap regression architecture to ensure accurate poses for different hand image sizes. Besides, we exploit the predicted hand skeleton as a constraint to guide the model.

Also, as already mentioned, one of the major challenges in this area is the restricted availability of joint annotations of RGB data. This study also constructs a new dataset that is suitable for both hand detection and pose estimation tasks. It includes the hand bounding boxes, the 2D keypoints, the 3D poses, and the corresponding RGB images. Our dataset presents different lighting conditions and high pose variations.

1.2.2. Accurate Root-relative 3D Hand Pose

To estimate an accurate 3D hand pose from a single RGB camera, our second study searches for constraints that can restrict the 3D regression problem. As we stated previously, current GCN-based approaches model skeleton relationships between joints. However, different hand poses should not share the same relationship constraints (adjacency matrix) since they do not have the same joint configurations (e.g; open and closed hands). Furthermore, we observe that current methods do not exploit physical distance between joints to determine their dependencies. Motivated by these observations, we introduce a coarse-to-fine GCN-based framework to estimate 3D hand pose from a single RGB image. Unlike previous methods that either employ predefined joint relationship constraints [25,27,28] or learn a global one for the entire dataset [29], we propose per-pose (per-image) relationship constraints. For each input image, we express dependencies between the joints based on their spatial locality (positions in 2D or 3D spaces). In particular, we divide our 2D/3D space into multiple classes such that neighboring joints belong to the same group. We utilize the classification probabilities to obtain rough results that supervise the regression stage in the coarse phase. Also, we explore the role of physical distance between joints in our refinement module. It identifies the adjacent joints based on a proposed adaptative nearest neighbor algorithm (ANN).

1.2.3. Accurate Absolute 3D Hand Pose Estimation

The third study addresses the absolute 3D hand pose estimation problem that refers to predicting the specific location of hand joints in the camera coordinate system. This task is less studied in the literature because it is challenging. One difficulty comes from the fact that it presents an ill-posed problem due to the irreversible geometry. In particular, a 2D point in the image plane can correspond to multiple 3D points in world

space. Besides, recovering the global scale of the hand is challenging due to depth ambiguity. Finally, the absolute pose resides in a larger space than the root-relative pose, which changes the underlying optimization process.

We observe that the input of the previous approaches is the cropped hand image. They do not exploit the global scene image that contains valuable information, such as the size of the hand that provides a clue about its depth (e.g. big hands means small depth and vice-versa). Also, the scale of the hand relative to the other motionless objects in the scene reduces the depth and scale ambiguities. Motivated by this observation, we propose a GCN-based method that consists of three stages: 2D key points, 3D root-relative, and 3D absolute pose estimation. To achieve better performance, we present a new loss function. It separates the extracted image features based on 3D joint locations to simplify the regression task and speed up the running time.

1.2.4. GANs-based Approach for Hand-object Rendering

Although the second part of the thesis is dedicated to AR application, it is vision-based since it aims to employ generative models to render the hand and the object images. No existing dataset includes real hand images holding synthetic objects. For that reason, our fourth study converts the input real hand image to a synthetic one using a GANs-based unsupervised image-to-image translation framework. This task is challenging since the pose of the input hand should be preserved. After that, we use another GANs-based network to render a synthetic hand and virtual object in an end-to-end manner. Finally, we segment the hand from the generated rendered image and translate it back to the real domain. To improve our qualitative and quantitative results, we propose a new loss function that enforces the segmentation maps of the object and the hands in the joint image to be like the ground truth.

1.3. Dissertation Outline

We organize the remainder of this thesis as follows. Chapter 2 introduces some core concepts about GANs and GCNs. Also, it reviews the most important traditional and deep learning-based hand detection and pose estimation approaches. Finally, it presents the related studies for hand-object rendering. We propose skeleton-aware

multi-scale heatmap regression for 2D hand pose estimation in Chapter 3. A hybrid classification-regression approach for 3D hand pose estimation using graph convolutional networks is introduced in Chapter 4. We present a graph-based approach for absolute 3D hand pose estimation using a single RGB image in Chapter 5. The proposed GANs-based approach for hand-object rendering is described in Chapter 6. We compare our proposed RGB-based 3D hand pose estimation approaches against each other, and we perform a combination of them in Chapter 7. Finally, we conclude this thesis, discuss the limitations of the presented studies, and describes future works in Chapter 8. The Appendix includes supplementary materials for Chapter 3, Chapter 4 and Chapter 7.

2. BACKGROUND

This chapter presents an overview of GCNs and GANs since our proposed 3D hand pose estimation and hand-object rendering approaches employ them. We explain the concepts of the graph adjacency matrix and the impact of the normalization process in Section 2.1. We also present the GANs architecture and loss functions in Section 2.2. Traditional and current learning-based hand detection methods are discussed in Section 2.3. Section 2.4 presents the most effective 2D hand pose estimation studies. We review the most important 3D hand pose estimation studies based on modality (depth, multi-view, or RGB) in Section 2.5. Also, we classify them based on the followed approach (generative or discriminative) in Section 2.6. Absolute 3D hand pose estimation methods are discussed in Section 2.7. Since no current learning-based approach performs hand-object rendering, we review the most important GANs-based image composition methods because they have shared challenges in Section 2.8.

We note that the reader needs to have a basic understanding of computer vision problems and terminologies. Also, we assume that deep learning techniques, such as feed-forward, convolutional neural networks, activation functions, optimizers, basic loss functions, and other terminologies are known. We encourage the reader to learn the fundamentals of deep learning in [23].

2.1. Graph Convolution Networks

GCNs are introduced in [24] to perform semi-supervised classification on graph-structured data. Unlike CNNs that can only handle fixed structures, GCNs treat irregular structures in a non-Euclidian space. Let $G = (V, E)$ denote a graph, where V and E represent a set of M nodes and L edges, respectively. Let $A \in [0,1]^{M \times M}$ be the adjacency matrix of G . Let I be the identity matrix and $\tilde{A} = A + I$ is the new self-loop adjacency matrix (Figure 2.1). Since the graph may include low/high-degree nodes a normalization process is required to avoid vanishing/exploding gradients. [24] addressed this problem by scaling rows and columns of $\hat{A}$ in (2.1).

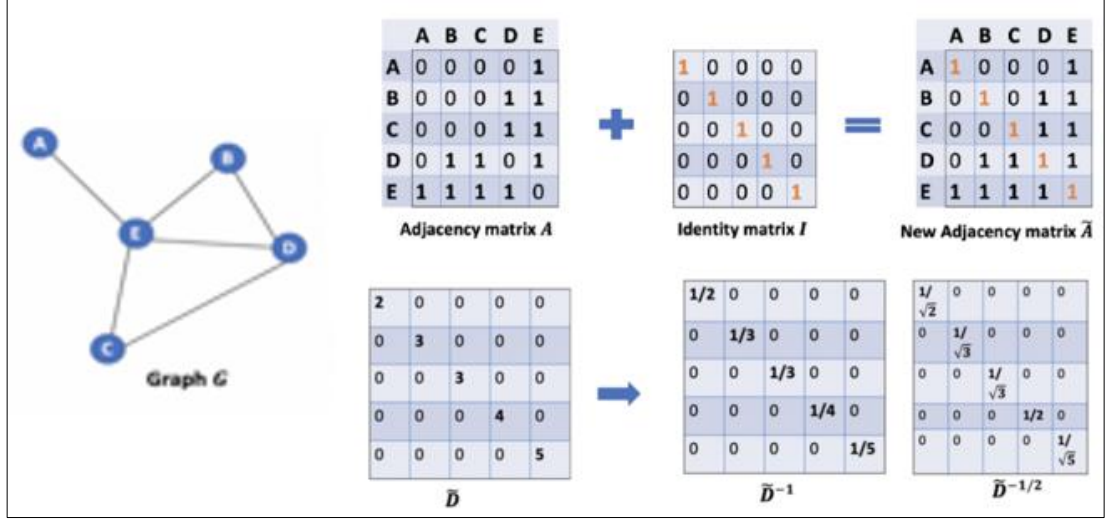


Figure 2.1: Example of adjacency matrix normalization in graphs.

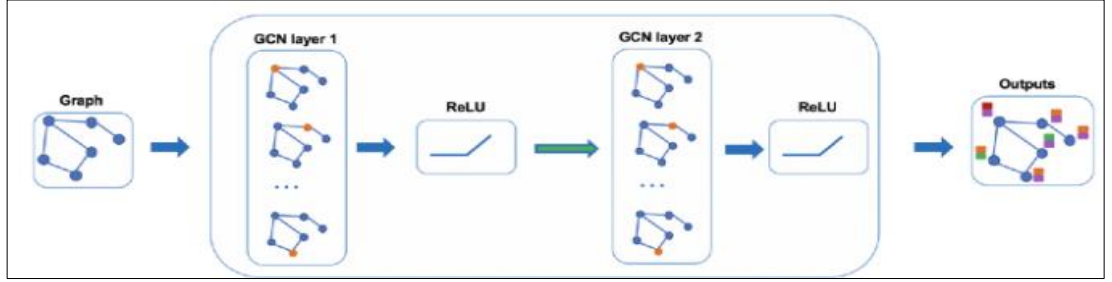


Figure 2.2: Information propagation in graphs to manipulate irregular structures.

$$\hat{A} = \tilde{D}^{1/2} \tilde{A} \tilde{D}^{1/2} \quad (2.1)$$

Where: $\tilde{D}$ is the degree matrix of $\tilde{A}$.

GCNs propagate information between nodes to update their representations. The adjacency matrix serves as a mask to select the aggregated nodes (Figure 2.2). The propagation rule in GCN is computed in (2.2):

$$H^{k+1} = \partial(\hat{A}H^k W^k) \quad (2.2)$$

Where: $\partial = ReLU$ is the activation function, H^k and W^k are the node features and the weights in the k^{th} layer, respectively.

2.2. Generative Adversarial Networks

GANs [14] achieve great success in modeling complex data distribution. They have been used in many complex computer vision problems and built new state of art in many tasks, including generating high-resolution images [31], performing image-

to-image translation tasks [28] (e.g., image inpainting and style transfer), and video generation [32].

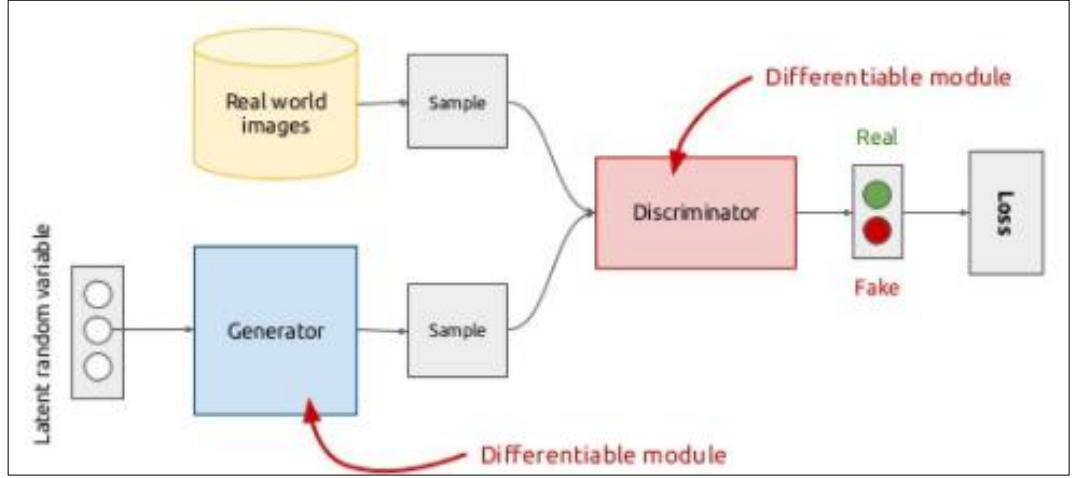


Figure 2.3: Alternative training between the generator and the discriminator.

GANs have two networks playing a min-max game against each other (Figure 2.3). The generator sample Z vector from a random distribution and performs up-sampling operations to build a fake image and fool the discriminator. The latter takes real and synthesized data and tries to distinguish between them. It estimates the probability that a sample came from the training data (data distribution) or G (model distribution). This competition will enforce the generator to output more realistic images. Training GANs is very hard due to the simultaneous optimization of different objectives and parameters. The discriminator and the generator update their parameters based on the loss functions described in (2.3) and (2.4), respectively.

$$L_{discriminator} = \min_D E_{x \sim \mathbb{P}_r} [-\log (D(x))] + E_{z \sim \mathcal{N}(0,1)} [-\log (1 - D(G(z)))] \quad (2.3)$$

$$L_{generator} = \min_G E_{z \sim \mathcal{N}(0,1)} [-\log (D(G(z)))] \quad (2.4)$$

Where:

- $\mathbb{P}_r$ is the real data distribution.
- $\mathcal{N}(0, 1)$ is the unit Gaussian distribution.
- D is the discriminator network (differentiable) function.
- G is the generator network (differentiable) function

2.3. Hand Detection

The hand detection task identifies the hand region and distinguishes it from the background. It has many applications including, gesture recognition [33] hand segmentation, and hand tracking [34]. Traditional computer vision methods follow a feature extraction and classification scheme for hand detection. They extract skin color features, shape features, or combine the two types of features to represent the image [35]. Following, they utilize a classifier to check each pixel, whether it belongs to the hand or not [36].

Deep learning-based methods circumvent such bottlenecks by unifying feature extraction and classification phases. This combined strategy has been outperforming conventional methods for the last five years. For instance, [37] employs two streams Faster RCNN [6] for hand detection. The first stream extracts feature maps from depth video while the second one extracts it from RGB video. After that, they use an alignment stage to connect the two features and they run a region proposal network to classify the pixels. Another method [38] applies multi-scale FasterRCNN to avoid missing the small hands.

2.4. 2D Hand Pose Estimation

Estimating the 2D hand pose has been an active and challenging area of research in computer vision. Recently, deep learning-based methods achieve competitive performance and outperform traditional methods. We can classify these based on the input modality into two broad categories as depth-based and RGB-based. In the former class, several studies achieve accurate 2D pose estimation results for images containing single hands [10,34–36]. Also, [42] handles multi-hands using pictorial structure models and Mask-RCNN. RGB-based methods are more challenging and less studied in the literature. Early studies give the cropped hand image as input to the ResNet-based model to directly regress the 2D joint by minimizing the mean square error (MSE) between the predicted 2D joint annotations and their ground truth [43]. Recently, [44] employs a graph-based framework to allow features at each node to be represented by 2D spatial confidence maps. Also, [45] proposes an adaptive graphical model network that includes two branches of CNNs computing unary and pairwise potential functions and a graphical model to integrate the calculated information. [46]

employs a cascaded CNN to predict the silhouette information (mask) and the 2D key-points in an end-to-end manner after localizing the hand region. To perform efficient small hand 2D pose estimation, [47] simultaneously regresses the hand region of interests and hand key-points. Subsequently, it iteratively uses the hand region of interest (RoI) as feedback information for boosting the hand keypoints estimation performance. [48] proposes the limb probabilistic mask, which uses a Gaussian distribution mask rather than the one-hot mask. The 2D pose regression task employs the synthesized hand mask to model the structural relationship between the 2D keypoints. To address the self-occlusion issue, it splits the whole hand mask into five fingers and the palm. There are several datasets for 2D hand pose estimation, including LSMV [43], OneHand10K [47], CMU and MPII+NZSL [49]. However, they are small-scale and do not show high pose variations.

2.5. Modality-based Classification of 3D Hand Pose Estimation

Until the early 2000s, almost all hand tracking methods relied on magnetic sensing devices (chips). These methods are cumbersome, expensive to set up, and hard to calibrate. In contrast, vision-based techniques provide a more natural way to perform 3D hand pose estimation. We classify them based on the input modality into three categories: depth-based, multi-view RGB-based, and monocular RGB-based (Figure 2.4).

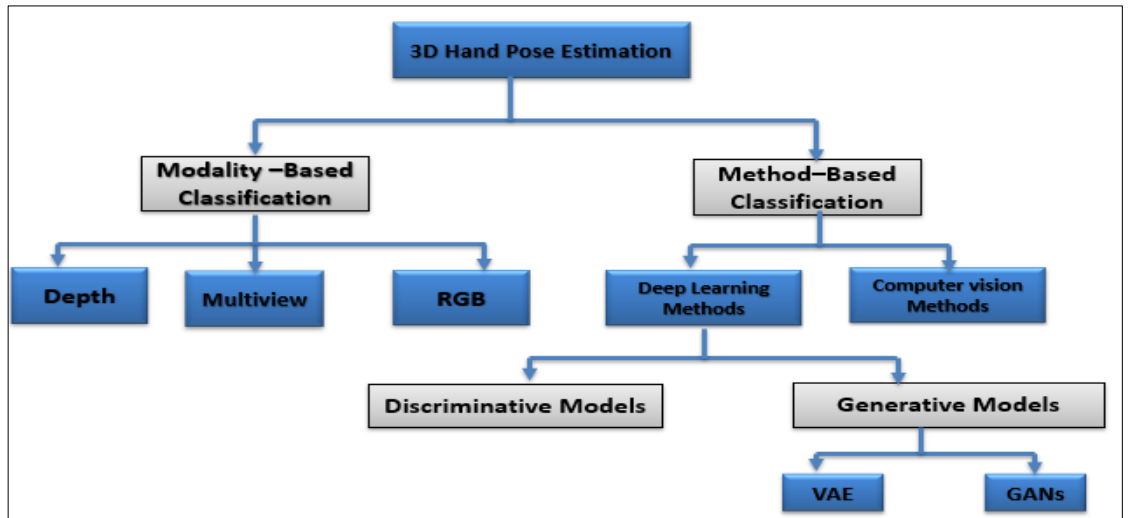


Figure 2.4: Classification of vision-based 3D hand pose estimation methods.

2.5.1. Depth-based Methods

With the introduction of depth sensors, new advances in the 3D hand pose estimation domain spurred, and various vision-based approaches quickly emerged in the literature. [8] adopts CNN to estimate the 3D hand pose from depth images for the first time. After this study, many methods have been proposed to increase the speed and robustness of pose estimation. They handle different types of inputs, such as 2D pixels [39,50], a set of 3D points [51,52], or voxels [53]. [54] presents a real-time hand tracking system based on a single depth camera. Their pipeline has three main steps: a hand RoI extraction stage, a reinitialization stage that employs a multi-layered discriminative strategy to do per-frame pose estimation, and a generative model-fitting stage. [55] comes with a fully articulated hand tracking system. [56] tried to increase the speed and robustness of pose estimation using a randomized decision forest that classifies pixels into parts of the hand in the detection phase. In the optimization step, their proposed objective function combines a Gaussian mixture representation with the results of the detection phase to estimate a pose that best fits the depth. Also, [57] can estimate the 3D hand pose without using GPU. It models the hand using several spheres and defines a fast cost function. Recently, [39] presents a new generative model that builds a new state-of-the-art in 3D hand pose estimation from a depth camera. Their method combines two deep generative models (VAEs [13] and GANs [14]) to learn a shared latent space. However, RGB-D sensors are not as available as monocular RGB cameras. Also, they only work in indoor environments and require an overhead setup.

2.5.2. Multi-views-based Methods

In the second category, several studies use many RGB cameras on different angles to alleviate the occlusion problem [49,58]. [59] proposes to project the query depth image onto three orthogonal planes and utilize these multi-view projections to regress 2D heatmaps. These multi-view heatmaps are then fused to produce final 3D hand pose estimation with learned pose priors. Their experiments show that the proposed method largely outperforms the state-of-the-art on challenging datasets. Inspired by this work, recent works aim to estimate the 3D hand pose from an RGB image using a multiview setup. [60] propose using a short-baseline stereo calibrated

camera for hand pose estimation without the need for a disparity map. Recently [49] can estimate the 3D hand pose even in the wild and with occlusion cases. They used a multiview bootstrapping to detect hand keypoints in real-time from RGB images. Since each view gives the truth annotation of a subset of key points, they can label images that are difficult or impossible to annotate due to occlusion using a panoptic camera setup, which restricts natural motion and appearance variation. The multi-view category also has some limitations, such as the difficulty of setup and datasets development. Also, they do not work well in unconstrained scenes (e.g., community videos).

2.5.3. Monocular RGB-based Methods

RGB cameras are more available compared to the two-mentioned categories. Also, they operate in all environments, and they do not need a setup overhead. There are several attempts to solve this problem with traditional methods. We can classify them mainly into two groups: appearance-based and model-based [60–62]. The former one extracts features from the input image, including color, texture, and shape. They may combine different computer vision techniques, such as image differencing, cross-correlation, skin detection, and active contours. For instance, [63] uses a multi-modal approach that combines optical flow and color cues to obtain a stable hand tracking system. The model-based techniques search the closest configuration that fits the hand model. They apply an optimization process using Particle Swarm Optimization (PSO). However, it is difficult to get diverse initial particles to address the fast motion problems. In other words, they require powerful prior knowledge about many dynamic and physics hypotheses, which leads to poor performances. Learning-based methods outperform traditional ones and achieve promising results. They can be classified into two main groups: root-relative 3D hand pose estimation and absolute 3D hand pose estimation. We review these approaches in the next two sections.

2.6. Root-relative 3D Hand Pose Estimation

Most existing methods estimate a scale-normalized root-relative pose. In particular, they predict the positions of hand joints relative to a reference point located on the hand (e.g., the wrist). Mainly, we can classify learning-based root-relative 3D

hand pose estimation methods into two categories: the first one includes discriminative models, which need a large-scale realistic hand dataset to extract features and estimate the 3D pose. The second category presents generative models that search for the best 3D hand configuration that will be evaluated against real data. These methods have been demonstrated to be quite effective. However, their development cost is expensive

2.6.1. Discriminative Methods

Discriminative methods act as a function that maps directly between the RGB image and the 3D pose using large-scale annotated datasets. They mainly use classification or regression techniques using learning-based techniques including, randomized decision forests, regression forests, boosted classifier trees, and CNNs. The classification-based methods can only classify a definite number of known discrete hand poses and are less suitable for continuous and accurate pose estimation. The regression-based approach needs a large amount of training data and involves a challenging optimization problem due to depth ambiguity.

[11] is one of the first attempts that estimate the 3D hand pose from a monocular RGB image using CNNs. It employs synthetic data and three concatenated networks: hand segmenter, 2D keypoints, and 3D hand pose predictors. After this work, many deep-based studies have been proposed to improve 3D hand pose estimation. [17] estimates 3D hand poses by combining CNN with a kinematic 3D hand model. [18] leverage depth information to supervise the training by having a pair of RGB-RGBD images. [19] uses a two-stack hourglass with latent 2.5D heatmaps and lifts them to 3D. Other approaches achieve reliable 3D poses by regressing the hand mesh and estimating the 3D pose from it [20,21,64,65]. However, these methods require additional supervision since the datasets must include the meshes and the 3D poses. More recently, [66] proposed a knowledge distillation and generalization framework for RGB-based pose estimation. GCN [24] became a hot topic of research in the computer vision field since there exist several problems that have graph-like structures as input. [25] applies GCNs [24] to estimate the 3D pose with the help of a weakly supervised training strategy that employs depth regularizer. [27] exploits spatial and temporal relationships for 3D human and hand pose estimation tasks. [26] introduces a self-supervised module that uses 2D relationships and 3D geometric knowledge to reduce the gap between 2D and 3D spaces. [29] proposes a UNet-based GCNs to

estimate the 3D hand pose and the 6D object pose. It learns global geometric relationship constraints for all hand poses. This thesis introduces a GCN-based framework for 3D hand pose estimation using a single RGB image. It exploits the spatial dependencies between joints to learn per-pose (per-image) relationship constraints yielding better performance than the global one. Unlike the discussed methods, our proposed approaches do not require depth or meshes supervision.

2.6.2. Generative Methods

Deep generative-based approaches exhibit promising performances and generate accurate 3D poses. They generate hand configuration hypotheses and evaluate them on available visual observations. They differ on several factors, including kinematic constraints, optimization techniques, and the cost function.

[9] uses VAEs [13] to learn a shared latent space across RGB and depth modalities to remedy the missing depth ambiguity. However, these methods model black-box latent representations, which synthesize a single 3D pose for a given RGB image. To address this issue, [15,16] disentangle the diverse factors that affect the hand visualization, including camera viewpoint, scene context, and background. Other studies employ generative models to build new datasets for training effective models. [12] uses generative adversarial networks (GANs) [14] to translate synthetic labeled data to realistic images to reduce the domain gap.

2.7. Absolute 3D Hand Pose Estimation

Although absolute 3D hand pose estimation has more application in AR/VR and multi-hand scenes, it is less studied in the literature compared to the root-relative 3D pose. We can explain this by the fact that estimating the global coordinates is too challenging due to irreversible geometry and scale ambiguities. To the best of our knowledge, there are only three studies [12,19,67] that work around this problem. [12] employs GANs [14] to translate synthetic labeled data to realistic images. This data augmentation helps to reduce the domain gap and estimate a reliable scale-normalized root-relative 3D hand pose. To restore an absolute 3D hand pose using kinematic hand model fitting. It recovers the 3D pose by optimizing different loss functions in a kinematic hand model, namely, a 3D loss term, an angle constraint loss, a reprojection

of the target 3D pose back to the 2D space using the camera intrinsic, and a temporal smoothness loss. However, optimizing loss functions with dissimilar objectives is prone to errors, e.g.: failures in the previous frames may affect the temporal smoothness loss of the current one. [19] reconstructs the global 3D hand pose from the latent 2.5D heatmaps that include 2D heatmaps and a depth map. In particular, it estimates the 2D key points and depth map to ensure the scale and translation invariance. This method assumes that the intrinsic camera parameters are known. To recover the hand scale, they conduct statistics about the dataset in a post-processing step (e.g average bone lengths). Recently, [67] estimate the global pose for the two hands using four main stages: hand segmentation, 2D canonical pose estimation, 3D canonical hand pose estimation, and 3D global hand pose estimation. It uses the spherical coordinate system to overcome the depth and rotation ambiguities encountered in the cartesian system. To eliminate the scale ambiguity, it fixes the distance between the finger Meta Carpo Phalangeal joints (MCP) and the palm to a fixed length. Our approach estimates the absolute 3D pose using a GCN-based framework that learns the hand scale and location without using pre-calculated statistics about the datasets or requiring the camera intrinsic parameters.

2.8. Hand-object Rendering and Image Composition Methods

Current methods estimate an accurate 3D hand pose from a single RGB image and render it with the object 3D pose. No existing vision-based approach renders the object and the hand images for AR applications. This task is close to image composition that aims to create a realistic-looking image by taking the foreground object of one image and combining it with the background from another image. This task is usually performed using professional image editing software such as Photoshop. However, it requires human supervision to adjust details, including geometry and color. Recently, deep generative models have achieved promising results in several computer vision problems and generate realistic images using GANs. This inspired many researchers to present new image composition methods that do not need human experts. [68] modified the GAN framework by restricting the range of the generator to a geometric manifold using a spatial transformer network. This method generates realistic composite images that are geometrically consistent. However, it

only operates in cases where the foreground appearance is already consistent with the background image. [69] address this problem by simultaneously learning the geometric and color correction with adversarial learning. Their framework contains four sub-networks: a transformation network, a refinement network, a discriminator network, and a segmentation network. The transformation and refinement networks act together as the generative compositing model, which aims to generate a realistic composite image while considering geometric, color, and boundary consistency. Meanwhile, the discriminator and segmentation networks help to increase the realism of the composite image through adversarial learning. [70] propose to use deep neural networks to learn the location and size of the foreground object for compositing humans into the background image. [71] propose to model object composition in a GAN framework as a self-consistent composition-decomposition network. Their model is conditioned on the object images from their marginal distributions and can generate a realistic image from their joint distribution. [72] adopts GANs to harmonize the foreground object within the background image and generates shadow effects by learning from real shadows in natural images. It disentangles the style transfer and shadow generation of the foreground object.

3. A MULTI-SCALE HEATMAP REGRESSION FOR 2D HAND POSE ESTIMATION

3.1. Introduction

Solving the 2D hand pose estimation problem is essential for many applications, including HCI and sign language recognition. The earlier works in hand tracking use special hardware to track the hand, such as gloves and visual markers [73]. But these types of solutions are expensive and restrict the applications to limited scenarios. Tracking hands without any device or markers is desirable. To this end, several works have been proposed in the literature to tackle this problem [74]. However, markerless hand pose estimation is very challenging due to strong articulations and self-occlusions. Also, the hands have a huge variation in shape, size, skin texture, and color.

We can classify RGB-based hand pose estimation methods into two broad categories as regression-based and detection-based. The former approach uses CNNs as an automatic feature extractor to directly estimate the joint locations [11,43,75]. Although the regression-based approach is fast at inference time, it remains a difficult optimization problem due to its non-linear nature requiring many iterations and a lot of data for convergence [76].

To overcome these limitations, recent solutions to human and hand pose estimation problems use probability density maps such as the heatmap [12,19,77]. They divide the pose estimation problem into two steps. The first one finds a dense pixel-wise prediction for each joint while the second step infers the joint locations by finding the maximum pixel in each heatmap. The heatmap representation helps the neural network to estimate the joint locations robustly and has a fast convergence property.

In this work, we focus on the 2D hand pose estimation from a single RGB image. This task is also challenging due to the many DOF and the self-similarity of the hand. The proposed approach has two principal components; The first one estimates the hand skeleton using the UNet-based architecture [78]. The hand bounding boxes are extracted in a post-processing step from the predicted skeleton. The second part presents a new multi-scale heatmap regression approach to estimate joint locations from multiple resolutions. Specifically, the network output is supervised on different scales to ensure accurate poses for different hand image sizes. This strategy helps the

model for better learning of the contextual and the location information. Besides, our method uses the predicted hand skeleton as additional information to guide the network to predict the 2D hand pose.

We validate the proposed method on a common existing Large-Scale Multi-View hand pose dataset (LSMV) [43]. Furthermore, we create a new dataset suitable for hand detection and 2D pose estimation tasks using leap motion sensors. This dataset includes 60 thousand samples, such that each one contains the hand bounding box, the 2D keypoints, the 3D pose, and the corresponding RGB image. We extended our experiments to our newly created dataset (GTHD). Results demonstrate that our method generates accurate poses and outperforms three state-of-the-arts [43,45,79]. In summary, our contributions are the following:

- We propose a segmentation-based approach for skeleton detection and hand bounding box localization.
- We propose a multi-scale heatmap regression architecture that uses the hand skeleton as additional information to constrain the 2D hand pose estimation task. The reported qualitative and quantitative results demonstrate the competitiveness of the proposed method.
- We introduce a new dataset to validate the hand detection and the 2D pose estimation methods.

We organize the rest of this chapter as the following. Sections 3.2 and 3.3 describe in detail our hand detection and pose estimation approaches. Section 3.4 defines the required steps to build our hand pose dataset. Section 3.5 discusses the conducted experiments and the obtained qualitative and quantitative results. Finally, Section 3.6 provides a summary of this study.

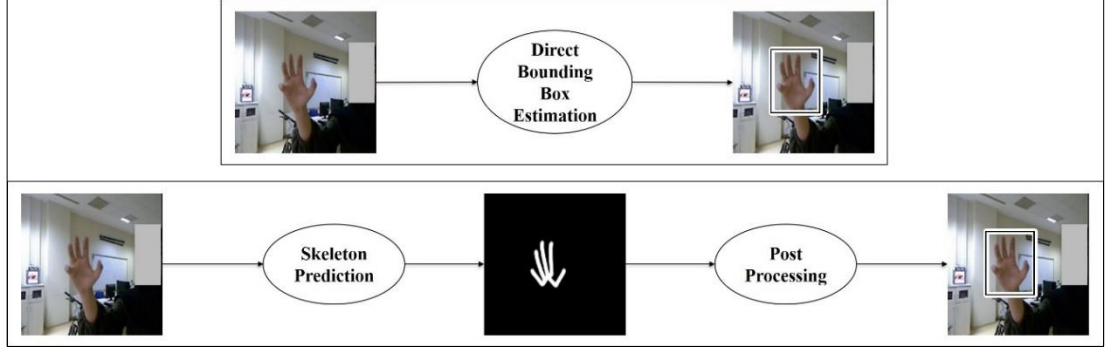


Figure 3.1: Faster-RCNN vs. our proposed hand detection and bounding box localization approach.

3.2. Skeleton Detection and Hand Bounding Box Localization

Our proposed approach for 2D hand pose estimation uses a skeleton-based approach to detect the hand and extract the bounding boxes. The second part uses the predicted skeleton as a constraint to guide the proposed multi-scale heatmap regression approach to predict the 2D joint locations of the cropped hand.

We represent the detected hand location in an image by a rectangular region with four corners. Faster-RCNN [6] type of deep network models directly regresses the four corner coordinates from the given hand image. Alternatively, we can predict the 2D hand skeleton and extract the bounding box in a post-processing step (Figure 3.1). Direct regression of the bounding box is useful for hand cropping but cannot be further exploited for other tasks. In contrast, estimating the hand skeleton includes useful information that guides the 2D pose estimation. Also, the segmentation task is less challenging than predicting the bounding box. Of course, one needs to have the training data with corresponding skeletons. We can obtain this type of data using a 3D hand tracker and an RGB camera to provide the 2D keypoints. We create the ground truth data for the skeleton by connecting the joints in each finger and attaching the palm to the ends of each finger. Also, we represent each joint location by the standardized Gaussian blob.

We can treat hand skeleton data as a segmentation mask. Thus, we use the well-known UNet architecture [78] since it is one of the best encoder-decoder architectures for semantic segmentation. It has two major properties. The first one is the skip connections between the encoder and the decoder layers that enable the network to

learn the location and the contextual information. The second property is its symmetry, leading to better information transfer and performance.

The model outputs single feature maps on which we apply a sigmoid activation function to bound the prediction values between 0 (background) and 1 (hand). We localize the bounding box using a post-processing step, in which we identify the foreground pixels, and then we apply a region-growing algorithm. In our case, the horizontal and vertical boundaries of the recovered regions are reported as the location of the detected hand. Our model robustly differentiates between the skin of the hand and that of the face. Also, it can detect the hand even in cluttered images or different lightning conditions.

Concerning the loss function, we did two experiments. In the first one, we only used the L_1 loss function, which cannot robustly localize the skeleton and adversely affects the bounding box localization results. In contrast, using the combination of L_1 loss (3.1) and a SoftDice (3.2) loss with their empirical weights can robustly localize the hand (3.3). We trained the model for 20 epochs using a batch size of 8.

$$L_1(x, \hat{x}) = \|x - \hat{x}\|_1 \quad (3.1)$$

$$SoftDice(x, \hat{x}) = 1 - \frac{2\hat{x}^T x}{\|\hat{x}\|_2^2 + \|x\|_2^2} \quad (3.2)$$

$$Total(x, \hat{x}) = \lambda_1 \times L_1(x, \hat{x}) + \lambda_2 \times SoftDice(x, \hat{x}) \quad (3.3)$$

Where: x , $\hat{x}$, λ_1 and λ_2 represent the ground truth skeleton, the predicted skeleton, and the two hyperparameters of the loss function (set to 0.4 and 0.6, respectively).

3.3. Multi-scale Heatmaps Regression for 2D Hand Pose Estimation

Most of the existing hand pose estimation methods predict the heatmaps at a single scale. However, the hand in the original image can have several sizes (close/far hands). Hence, when we use a single scale image, the cropped hand image size cannot be suitable for all the dataset samples.

To address this limitation, we propose a multi-scale heatmaps regression architecture that performs the back-propagation process for many resolutions providing better joint learning for both large and small hands. Moreover, the cropped hand image would include some parts of the background. To overcome this problem, we employ the predicted hand skeleton to act as an attention mechanism for the network to focus on hand pixels. This makes the 2D pose regression task less challenging to optimize.

Figure 3.2 shows our skeleton-aware multi-scale heatmaps network approach for 2D hand pose estimation. We feed the concatenation of the cropped hand image and the predicted skeleton to the first convolution layer. The latter is followed by two down-sampling ResNet blocks, two up-sampling ResNet blocks, and a final transposed convolution layer that recovers the input resolution. After each down-sampling (similarly up-sampling), we apply a 1×1 convolution layer followed by a sigmoid activation function to output 21 or 20 feature maps representing the heatmaps in GTHD or LSMV datasets, respectively. The heatmaps resolution is divided/multiplied by two after each down-sampling/up-sampling.

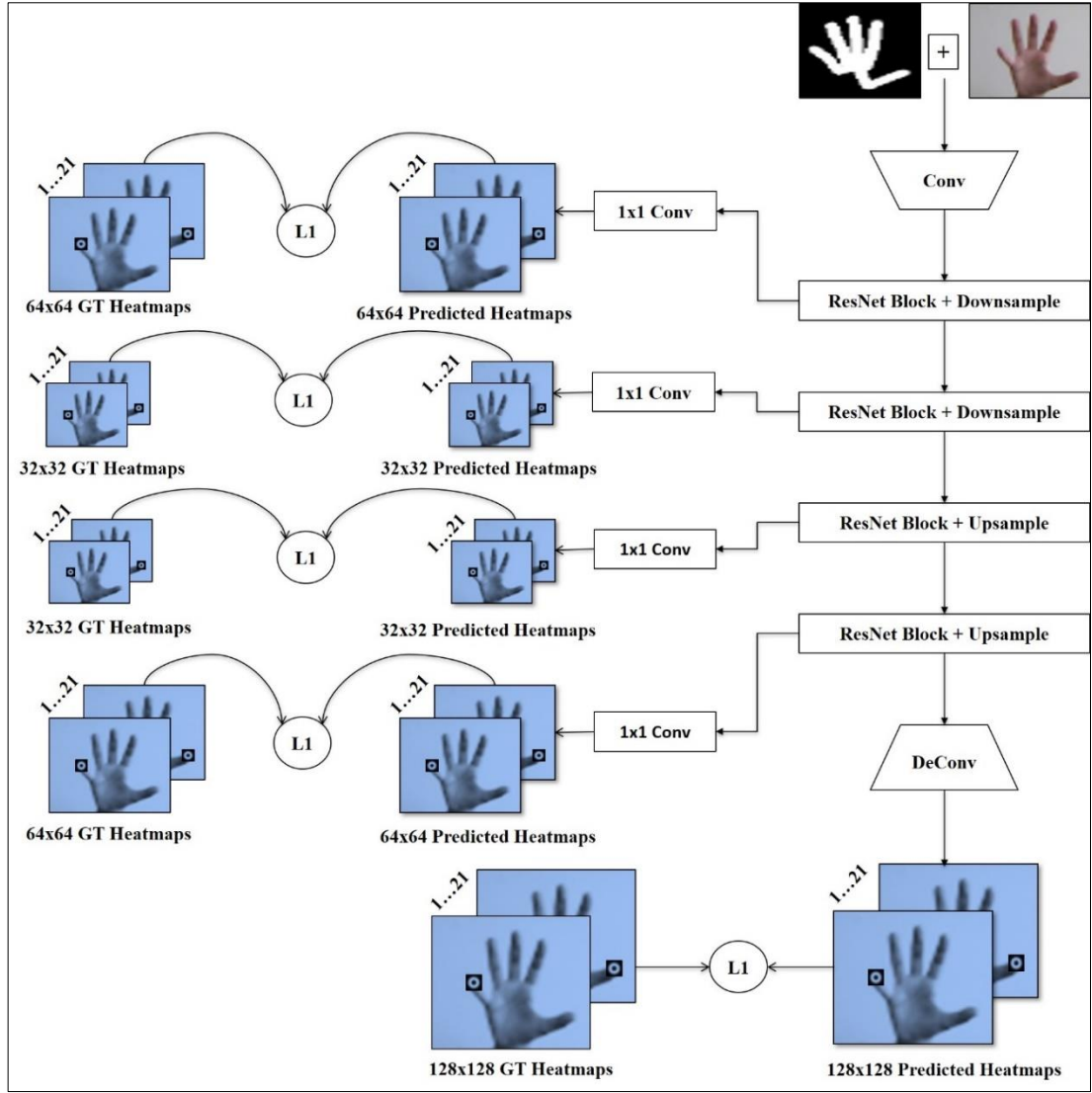


Figure 3.2: The architecture of the proposed 2D hand pose estimation approach.

In test time, we calculate a weighted average of the predicted five heatmaps to find the coordinate of the 2D keypoints. We formulate the loss function in (3.4) as:

$$L_2(x, \hat{x}) = \sum_{i=1}^k \delta_i \|x_i - \hat{x}_i\|_2 \quad (3.4)$$

Where: k is the number of scales including the full resolution output, and δ_i is the weight given for each scale. In our experiments, we choose $k = 5$ and δ_i is set to be 1, 1/2, and 1/4 for scales 128, 64, and 32 respectively.

3.4. Datasets and Evaluation Metrics

Deep learning methods require a large number of labeled data for training. There is a lack of datasets that has RGB hand images with their 2D annotations that we can use to train our proposed approaches. For example, [80] has RGB images with their 2D annotations, but they are both small scale and do not describe the hand by joint annotations.

Our method has been implemented and tested on two different datasets. The first one is LSMV [43], which is one of the large-scale datasets that provide the hand bounding boxes, the 2D key-points as well as the 3D pose. We split the data into 60000, 15000, and 12760 samples for the training set, validation set, and test set, respectively. While LSMV [43] can be used to train and validate the 2D hand pose estimators, it cannot be used for hand detection since it does not have images without hands.

To overcome this limitation, and train both the hand detector and the hand pose estimator, we have built our dataset (GTHD) using an RGB camera and a Leap Motion sensor [81]. It is composed of two subsets; The first one has 60 thousand RGB images with their corresponding hand bounding boxes, 2D keypoints, and 3D pose. The second set has 15 thousand RGB images that present either the background or people who do not show their hands. The new dataset has a large variation in hand poses, backgrounds, skin color, and texture

The RGB camera provides an image with a resolution of 640×480 pixels. The leap motion controller is a combination of hardware and software that senses the fingers of the hand to provide the 3D joint locations. Hence, a projection process from 3D space to the 2D image plane is necessary. We achieve this goal in two steps. In the first one, we use OpenCV to estimate specific intrinsic parameters of the camera. In the second step, we estimate the extrinsic parameters between the leap motion controller and the camera. To get the correct pose with its corresponding image, we synchronize the two sensors in time.

Finally, to find the rotation and translation matrices, we manually mark one keypoints in a set of hand images and solve the PnP problem by computing the 3D-2D correspondences [82]. Figure 3.3 illustrates the results of the calibration process. We randomly split the GTHD dataset into a training set (75%), a validation set (10%), and a test set (15%).

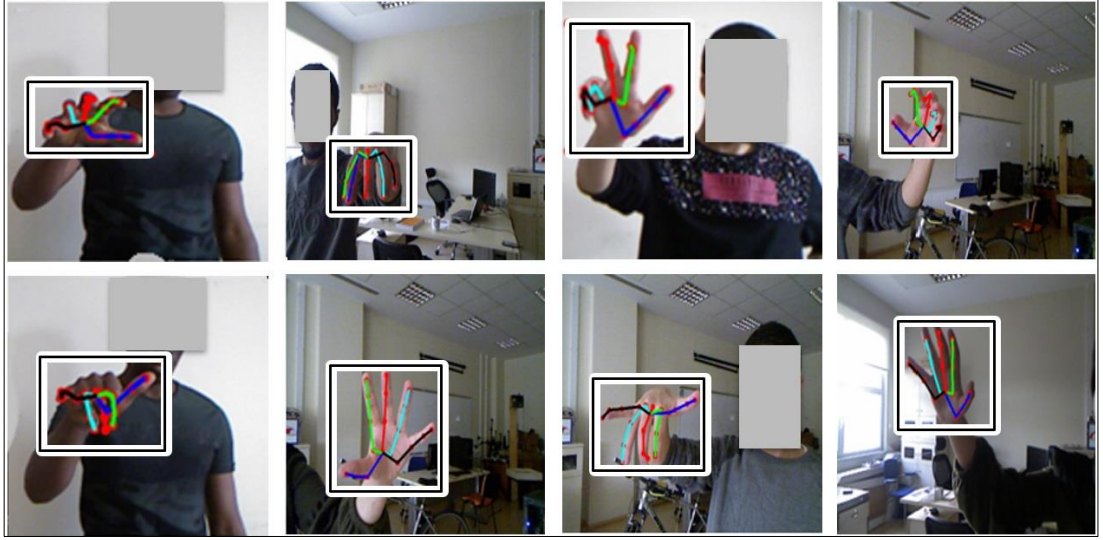


Figure 3.3: Examples of our hand dataset images. It includes the hand bounding boxes and 21 joint annotations in 2D and 3D spaces.

We report the performance of the hand skeleton detection module using famous classification metrics, such as accuracy, precision, recall, and F1. Furthermore, we calculate the Area Under ROC Curve (AUC) for GTHD datasets since it measures how well the two classes (Hand and NoHand) are separable. It calculates the trade-off between the true positive rate and the false positive rate. Also, we report the Intersection over Union (IOU) metric to quantify our model performance in the hand bounding box detection task. It evaluates the predicted bounding boxes by comparing them against the ground truth.

To quantitatively evaluate the performance of the proposed 2D hand pose regression methods, we use the Probability of Correct Keypoints (PCK) metric [49] as it is used frequently in human and hand pose estimation tasks. We use a normalized threshold by dividing all the joint values by the size of the hand bounding box. Also, for additional quantification of the performance of the proposed method, we report the mean joint pixel error (MJPE) over the input hand image with 128×128 resolution.

3.5. Experiments and Results

3.5.1. Implementation Details

We train the models for 30 epochs using a batch size of 8 and Adam as an optimizer. We initialize the learning rate to 0.01, and we decrease it after every eight

epochs by 10%. We conduct all experiments on NVIDIA GTX 1080 GPU using Pytorch v1.6.0.

Before giving the RGB image as input to the model, we resize and normalize the datasets by subtracting the mean from all the images. The number of heatmaps is the number of joints where we represent each one with a Gaussian blob in a map of the same size of the image. The coordinate of the joint is the location of the highest value in the heatmap. We find them by applying the argmax function.

To validate our hand detection approach, we use different degrees of skeleton thickness (1, 3, and 6). In the first case, the skeleton is simply composed of lines. In other cases, it has thicker connections and regions around the joints. Furthermore, we select a threshold that represents the number of foreground pixels to be the criterion to separate the two classes (Hand and NoHand).

3.5.2. Hand Detection and Bounding Box Estimation Results

Our approach for hand bounding box localization can robustly estimate the hand skeleton and localize the hand bounding box for the two datasets. It does not produce any false positives for background images or images with people who do not show their hands (see Figure 3.4 and Figure 3.5). The rows from top to down show: the input image, the ground truth skeleton, the predicted skeleton, and the obtained bounding boxes.

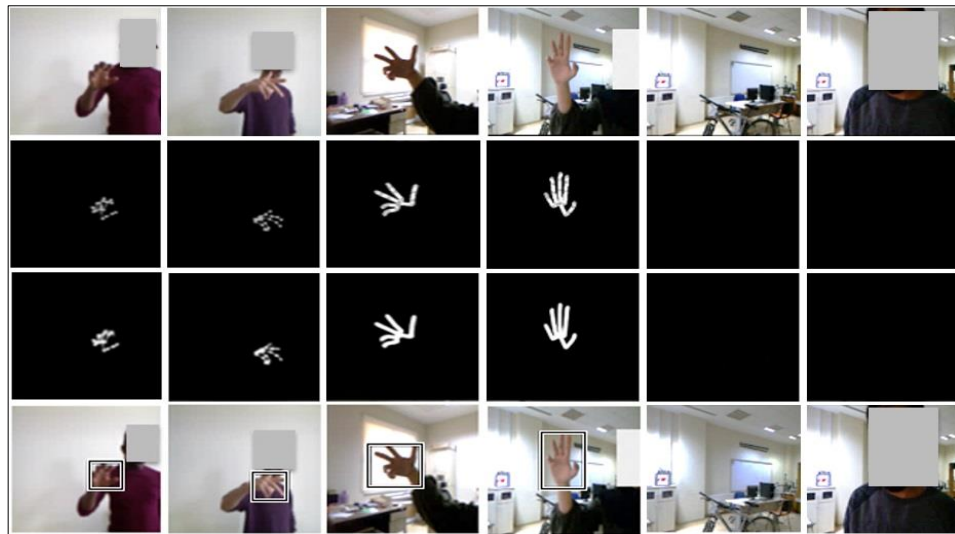


Figure 3.4: The results of the skeleton estimation and the bounding box localization on the GTHD dataset using thick and thin skeleton representations.

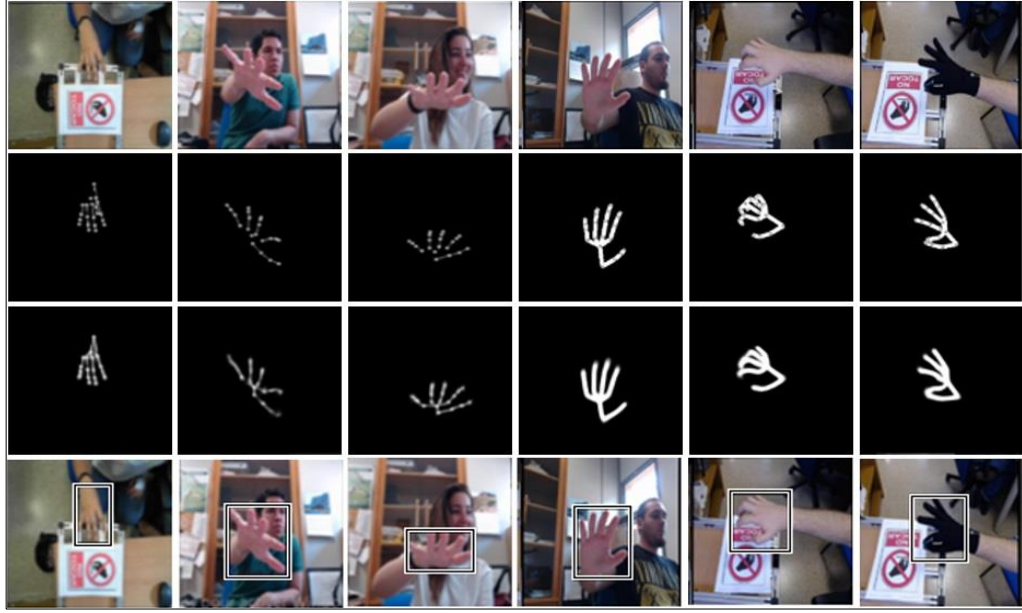


Figure 3.5: The results of the skeleton estimation and the bounding box localization on the LSMV dataset using thin and thick skeleton representations.

The correct threshold for selecting Hand from NoHand depends on the data. A robust threshold should eliminate the noise and be in an interval that does not miss samples from the dataset distribution. In other words, the selected threshold should decrease both the false-negative rate (adding samples from the NoHand class) and the false positive rate (missing samples from the Hand class) to achieve high performance and robustly detect the hand. Figure 3.6. shows that selecting a threshold from the interval $[200,400]$ is the best choice for our dataset. Also, the thickest skeleton representation seems to be more robust to the noise. It outperforms the other representations and achieves a higher performance ($AUC = 0.99$). Finally, our approach records high scores of $Accuracy = 0.99$, $Precision = 0.97$, $Recall = 0.99$, and $F1 = 0.98$.

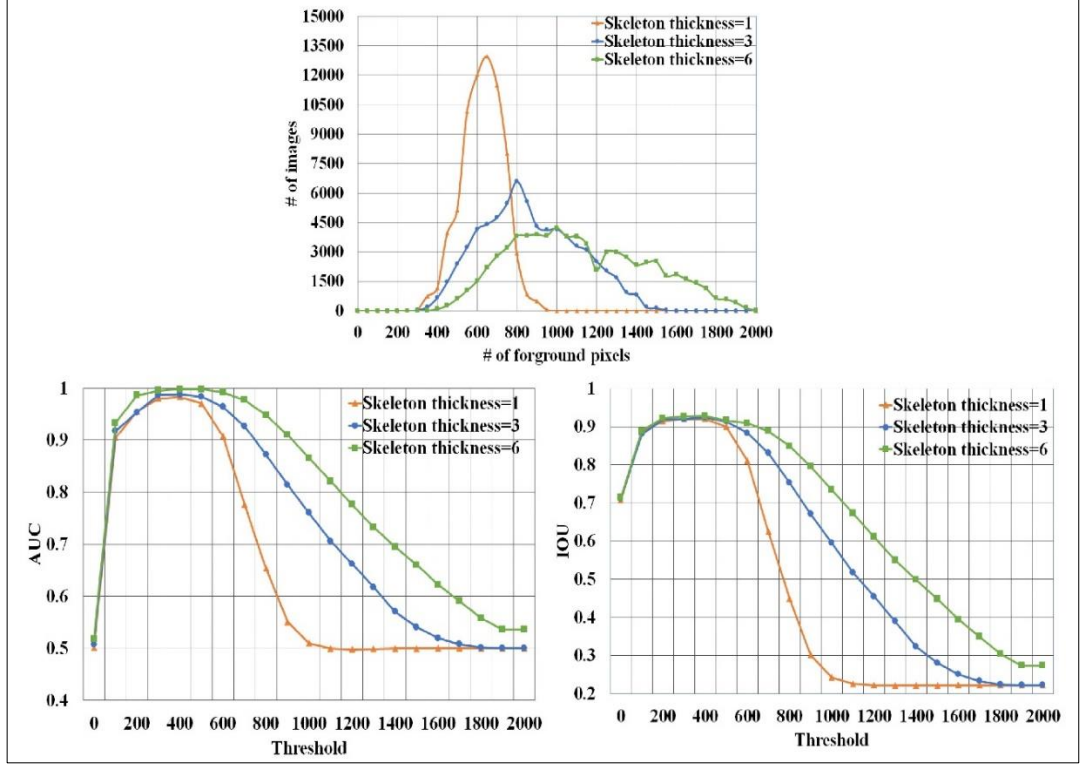


Figure 3.6: The impact of the threshold selection on hand detection results.

We do not report the AUC for the LSMV dataset because it does not have images without hands. Nevertheless, we predict the skeleton representation to extract the hand bounding boxes and perform our proposed 2D hand pose estimation method (Figure 3.5). Also, we report IOU in Table 3.1 showing that the proposed method outperforms Faster-RCNN [6].

Table 3.1: Bounding box evaluation on LSMV and our GTHD dataset with IOU.

Dataset	Faster-RCNN	Ours
GTHD	0.912	0.923
LSMV	0.895	0.917

To show the influence of the hand detection step on the bounding box localization performance, we record AUC and IOU metrics for different thresholds on the GTHD dataset. Figure 3.6 shows that the performance of the bounding box localization is strongly related to skeleton detection. Also, the thickest skeleton

representation seems to be more robust to the noise. It outperforms the other representations and achieves a higher performance (0.99 in AUC and 0.92 in IOU).

3.5.3. Pose Estimation Results

The proposed method can robustly estimate the 2D hand pose even in the cases of complex poses and cluttered images. Figure 3.7 shows some randomly selected test images on LSMV and GTHD datasets. The columns from left to right in each image show: the direct regression proposed in [43], our proposed skeleton aware multi-scale heatmaps regression, and the ground truth joints.

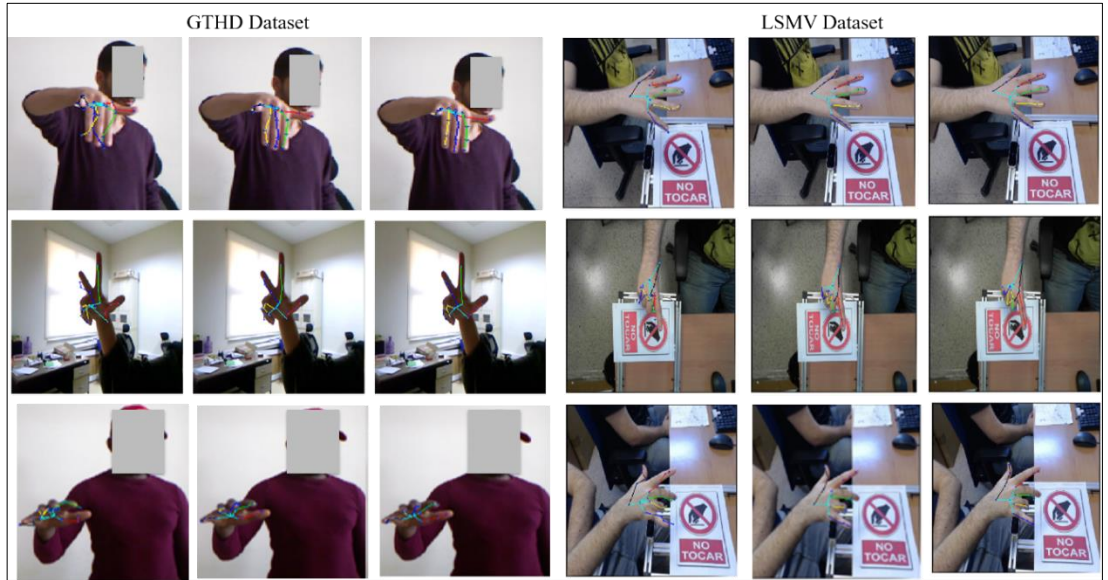


Figure 3.7: Qualitative results for the 2D hand pose estimation task.

Table 3.2: Comparison with the state-of-the-art methods on the LSMV datasets with PCK metric.

Threshold of PCK	0.01	0.02	0.03	0.04	0.05	0.06	Mean PCK
Gomez et al [43]	39.27	71.12	90.43	93.56	95.69	80.74	80.74
Kong et al [45]	41.38	85.67	93.96	96.61	97.77	98.42	85.63
Kong et al [44]	41.27	85.89	93.82	96.43	97.61	98.29	85.56
Ours	51.02	88.91	95.30	97.54	98.27	98.63	88.27

Table 3.3: Comparison with the state-of-the-art methods on GTHD and LSMV datasets with MPE metric.

Methods	GTHD	LSMV
Gomez et al [43]	13.20	10.00
Lie et al. [83]	6.25	8.05
Single-scale [78]	7.33	5.87
Our w/o skeleton	5.89	4.95
Ours	5.51	4.67

We compare the proposed pose estimation approach against three-deep learning-based methods [43,45,79] on the LSMV dataset. Our baseline is [43] uses ResNet-50 architecture [84] to directly regress the 2D joints from RGB images. The other deep-based methods [45,79] are two of the existing state-of-the-art in 2D hand pose estimation. The proposed skeleton-aware multi-scale heatmaps regression method outperforms [43,45,79] since it learns the joint location from many resolutions. It reports the highest PCK across all the thresholds (Table 3.2).

To further demonstrate the effectiveness of the proposed approach, we conduct additional experiments on the GTHD dataset. In the first one, we perform state-of-the-art methods [43,83]. The second experiment applies single-scale heatmap regression using UNet architecture [78] on 128×128 resolution images. The third experiment performs our multi-scale heatmaps regression without the skeleton information. In the last experiment, we perform our skeleton aware multi-stage heatmap regression architecture shown in Figure 3.2. We can see from Figure 3.8 that our method achieves a high PCK score (0.98) with a small threshold in LSMV and GTHD datasets. Furthermore, the hand skeleton representation improves the proposed multi-scale heatmaps regression method since it constrains the 2D pose estimation task (Table 3.3).

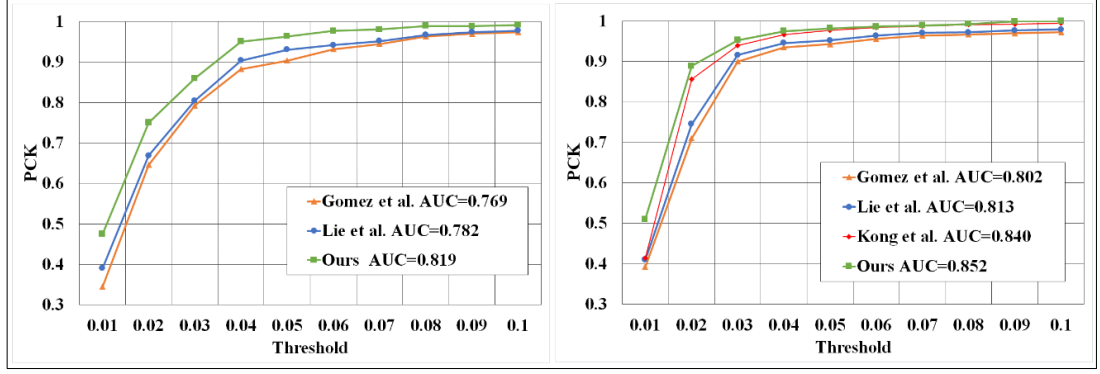


Figure 3.8: Quantitative comparison of the proposed 2D hand pose estimation using PCK metric for GTHD (left) and LSMV (right).

Estimating the 2D hand pose using the single scale heatmaps regression outperforms the direct regression since the detected heatmaps help CNNs to learn better the joint locations and converge faster (Figure 3.8). Finally, our proposed method for 2D hand pose estimation provides more improvement for our dataset since it has more complex poses, face occlusion cases, and lighting condition variations (Figure 3.8 and Table 3.3).

3.6. Summary

Existing RGB-based 2D hand pose estimation methods learn the joint locations from a single resolution, which is not suitable for different hand sizes. To tackle this problem, we propose a new deep learning-based framework that consists of two main modules. The first one presents a segmentation-based approach to detect the hand skeleton and localize the hand bounding box. The second module regresses the 2D joint locations through a multi-scale heatmap regression approach that exploits the predicted hand skeleton as a constraint to guide the model. Moreover, we construct a new dataset that is suitable for both hand detection and pose estimation tasks. It includes the hand bounding boxes, the 2D keypoints, the 3D poses, and their corresponding RGB images. We conduct extensive experiments on two datasets to validate our method. Qualitative and quantitative results demonstrate that the proposed method outperforms the state-of-the-art and recovers the pose even in cluttered images and complex poses.

4. A HYBRID CLASSIFICATION-REGRESSION APPROACH FOR 3D HAND POSE ESTIMATION

4.1. Introduction

Hand pose estimation is one of the well-known and fast-growing topics in the computer vision community. It plays a significant role in multiple application domains, such as HCI, AR, and robotics. Although studied extensively in the literature, 3D hand pose estimation remains challenging due to the similarity among fingers, self-occlusion, and the complexity of the hand poses.

With the introduction of affordable commodity depth sensors and the rapid development of deep learning techniques, RGBD-based approaches achieve accurate 3D hand pose estimation results [39,50–53,85,86]. Other methods employ calibrated multi-view cameras to reduce the depth ambiguity [49,58]. However, both depth and multi-view solutions are not readily available and work only in constrained environments. RGB-based methods are preferable since they are more accessible and do not require controlled settings. Due to the absence of depth information and occlusion, this task becomes an ill-posed problem. Furthermore, collecting 3D annotations for real images requires complex multi-view setups and manual annotations.

Learning-based methods tackle RGB-based 3D hand pose estimation by creating effective models trained on large-scale datasets [12,43,87]. We can divide them mainly into two categories. The first one adopts generative models to find the closet configuration (model-based) [9,12,15,16]. The second category uses discriminative models to extract features and estimate the 3D pose (appearance-based) [11,17–21]. Despite the promising results, most learning-based methods assume that the model can implicitly learn kinematic relationships between the joints. However, this task is challenging and requires explicit constraints to guide the model optimization.

Inspired by the natural graph representation of the hand, recent studies use GCNs [24] to model skeleton constraints between joints [25–29]. [27] exploits annotated video frames to enforce spatial and temporal relationships between the joints based on predefined semantic meanings. This approach is limited to consecutive images, where temporal information is present. [28] estimates the 3D human pose by learning weighted relationships between body joints based on a predefined human skeleton.

However, this approach misses potential connections between physically disconnected joints, which fails to capture pose-dependent relationships. To address this limitation, [29] replaces the skeleton-based representation with global learnable constraints between all hand poses. But different hand poses should not share the same relationship constraints since they do not have the same joint configurations (e.g.: open and closed hands). Furthermore, current methods do not exploit physical distance between joints to determine their dependencies.

Motivated by these observations, we introduce a two-stage GCN-based framework to estimate 3D hand pose from a single RGB image. Unlike previous methods that either employ predefined joint relationship constraints [25,27,28] or learn a global one for the entire dataset [29], we propose per-pose (per-image) relationship constraints. For each input image, it expresses dependencies between the joints based on their spatial locality (positions in 2D or 3D spaces). Our first stage is a hybrid classification-regression model that employs a lightweight feature extractor (ResNet-10) [84] and GCN modules [24] to classify each 2D/3D joint into its corresponding 2D/3D blocks. In particular, we divide our 2D/3D space into multiple classes such that neighboring joints belong to the same class. We utilize the classification probabilities to construct per-pose geometric constraints for 2D/3D joints. Afterward, we regress an initial (coarse) 3D hand pose using another GCN module that benefits from rough classification results to guide the pose estimation task (Figure 4.1). Identifying neighboring joints for each input helps the model to estimate accurate 3D hand poses. Finally, we further improve the coarse pose using ANN-based (Adaptative Nearest Neighbors) GCN. Instead of fixing K neighbors for all the nodes (KNN), we calculate the distance between all joints to learn a differentiable threshold that determines a different number of neighbors for each node. The proposed approach speeds up the running time since it includes a lightweight feature extractor [69] and GCN-based modules [24]. The latter proved to be more computationally efficient than CNN [29]. We summarize our contributions as follows:

- We propose a two-stage GCN-based approach that combines classification and regression to estimate an accurate 3D hand pose from a single RGB image. The classification branch finds spatial dependencies between neighboring joints to learn per-pose (per-image) relationship constraints that guide the regression branch.

- Instead of KNN-based GCN, we propose an adaptive nearest neighbor algorithm that learns a different number of joint relationships improving the 3D hand pose estimation task.
- To evaluate our approach, we conduct extensive experiments on synthetic and real-world datasets. The reported quantitative and qualitative results demonstrate that our efficient method outperforms several baselines and state-of-the-art and estimates accurate 2D/3D hand poses.

The rest of this chapter is organized as follows: Section 4.2 explains the proposed method in-depth and describes the most important modules of our framework. Section 4.3 describes the experimental setting. Section 4.4 analyzes the obtained results on two public datasets and compares the proposed approach against the state-of-the-art. Finally, Section 4.5 presents the summary of the study.

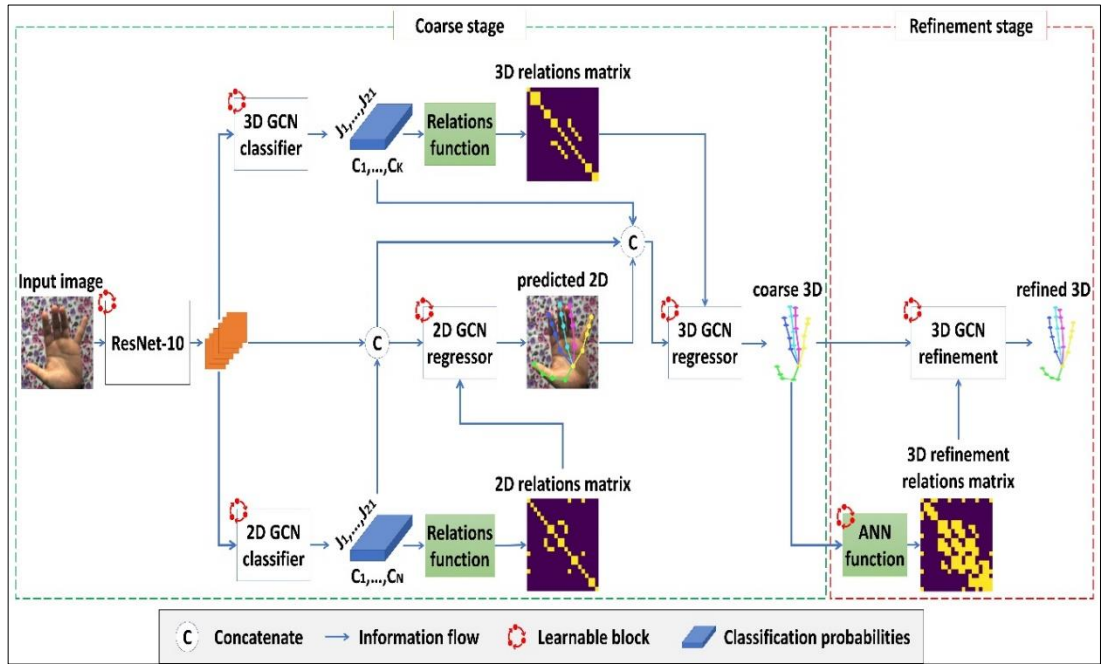


Figure 4.1: Schematic overview of our 3D hand pose estimation framework.

4.2. Methodology

Due to the absence of depth information, it is challenging to estimate accurate 3D hand poses from a monocular RGB image. We observe that one of the main obstacles for learning-based 3D hand pose estimation approaches is the non-

exploitation of spatial dependencies between adjacent joints to constrain the 3D pose estimation task and reduce the depth ambiguity. Furthermore, [27,29] shows that coarse-to-fine techniques improve the 3D pose estimation accuracy. Motivated by these observations, we introduce an effective approach for 3D hand pose estimation from a monocular RGB image (Figure 4.1).

Table 4.1: The architecture of the modified ResNet-10 network.

Layer	In	Out	Kernel	Stride	Padding
Conv2D	3	32	7×7	2	3
BN	32	32	-	-	-
ReLU	32	32	-	-	-
Max-pool	32	32	3	2	1
Res-Block	32	32	3×3	1	1
Res-Block	32	64	3×3	2	1
Conv2D	32	64	1×1	2	0
BN	64	64	-	-	-
Res-Block	64	128	3×3	2	1
Conv2D	64	128	1×1	2	0
BN	128	128	-	-	-
Res-Block	128	256	3×3	2	1
Conv2D	128	256	1×1	2	0
BN	256	256	-	-	-
Conv2D	256	21	3×3	1	1

4.2.1. Feature Maps Extraction

We use a modified ResNet-10 architecture [69] as a backbone for feature extraction from an RGB hand image. We reduce the number of feature maps in each convolution layer (width) to improve efficiency without affecting the performances. We describe the details of our architecture in Table 4.1. We feed the cropped hand into the network, and we flatten the last two dimensions of the feature maps from $(B, 21, W, H)$ to $(B, 21, W \times H)$, where B is the batch size, W and H are the width and the height of the feature maps, respectively. The reason for this transformation is to

get the correct input format for the 2D/3D GCN classifiers, where each joint has a feature vector of size $W \times H$.

4.2.2. Classification Modules

We feed the output feature maps of the ResNet-10 network to 2D/3D GCN-based classifiers [24] to assign each 2D/3D joint to a specific 2D/3D block, respectively, based on its locality. This task is less challenging than regression since the output dimension is restricted. Furthermore, it exploits spatial dependencies between the joints (nodes) to learn 2D/3D per-pose relationship constraints that serve as additional information during pose estimation.

To train our classifiers, we need the ground truth classes for 2D and 3D joints. We classify each 2D joint into one of the 2D blocks based on its (x, y) coordinates. Specifically, we divide the 2D space into $N = n \times n$ blocks where n is the number of splits in the horizontal and the vertical axis of the image. The shape of 2D classes is $(B, N, 21)$, where B is the batch size and 21 is the number of joints. Figure 4.2 illustrates the quantization process of the joints in 2D and 3D spaces. The top and the bottom rows demonstrate two examples of 2D and 3D quantization. Due to visualization difficulties, we represent the 3D classes with cubes.

In the 3D joint classification, we divide the 3D space into $K = k \times k \times k$ blocks where k is the number of splits in the x, y and z axis. We classify each 3D joint into one of these 3D blocks based on its (x, y, z) coordinates. The shape of 3D classes is $(B, K, 21)$ and the output of the 2D/3D classifier in Figure 4.1 is a tensor of class probabilities for each joint.

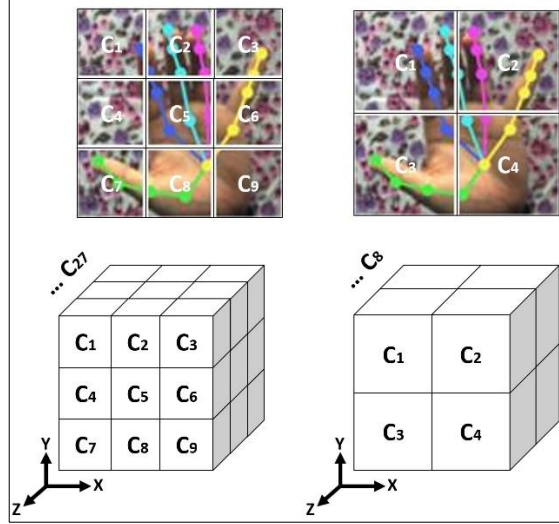


Figure 4.2: We classify the joint positions into a predefined number of classes.

In this stage, we use the propagation rule in [29] to classify 2D/3D joints. Instead of feeding the predefined adjacency matrix [24], our GCNs learn a global one $\hat{A}$. The latter is a 21×21 matrix of weights declared in the GCN and optimized during training. Their values are used as input to the GCN-based regressors. At test time, these weights become constant regardless of the input image. We empirically set the number of graph convolution layers of our network to 2. We feed the estimated values to the Relations function in Figure 4.3, which outputs a per-pose 2D/3D geometric relationship constraints to express spatial dependencies between each pair of joints.

We find that the GCN-based classifier exhibits better performance than Fully Connected (FC) networks. Note that we quantized the 2D/3D interval into overlapped blocks to consider the problem of joints residing in borders. However, we do not notice any gain over the proposed classification approach. We can explain this by that using overlapped blocks increases the number of classes reducing accuracy performances. Also, a joint can belong to different classes, which confuses the classifier.

4.2.3. Relations Function

We employ the Relations Function described in Figure 4.3 to express geometric constraints between the joints. It constructs a 2D/3D per-pose adjacency matrix that serves as an extra weighting for the GC layers. The algorithm takes the output of the 2D/3D classifier, applies a SoftMax activation function to get class probabilities of each joint and an Argmax function to find the class corresponding to the highest

probability. Finally, adjacent joints are identified by comparing their labels. The output matrix is of size 21×21 where each (i, j) coordinate exhibits the relation between $joint_i$ and $joint_j$. Note that the proposed function can identify relations between pairs, which are physically disconnected in the hand skeleton model.

Algorithm 1: Relations function.	
1	function RelationsFunction (<i>data</i>);
	Input : Classifier outputs for each joint.
	Output : 21×21 relations matrix.
2	<i>matrix</i> $\leftarrow []$;
3	<i>probabilities</i> $\leftarrow \text{softmax}(\text{data}, \text{dim} = 1)$;
4	<i>labels</i> $\leftarrow \text{argmax}(\text{probabilities}, \text{dim} = 1)$;
5	for $i=0:21$ do
6	<i>values</i> $\leftarrow []$;
7	for $j=0:21$ do
8	<i>a</i> $\leftarrow \text{labels}[:, i]$;
9	<i>b</i> $\leftarrow \text{labels}[:, j]$;
10	<i>values</i> [<i>j</i>] $\leftarrow (a == b)$;
11	end
12	<i>matrix</i> [<i>i</i>] $\leftarrow \text{values}$;
13	end
14	return <i>matrix</i> ;

Figure 4.3: Algorithm of the relations function. It identifies adjacent joints belonging to the same class.

4.2.4. Regression Modules

This module aims to estimate 21 2D/3D hand joints from a monocular RGB image. [29] estimates the 3D poses using ((2.2) as a propagation rule. It learns a global adjacency matrix between all the poses in the dataset. Although [29] outperforms the limitation of [25,27] that does not include relations of physically disconnected joints, it presents another problem. In particular, learning only a global adjacency matrix is a potential limitation since different hand poses in the dataset share a single relationship constraint (e.g., open and close hands have the same adjacency matrix despite being physically dissimilar). Our 2D/3D GCN-based regressors exploits the estimated 2D/3D per-pose adjacency matrices R_{2D} and R_{3D} as local constraints for each hand pose (4.1) and (4.3). Besides, it keeps the learnable adjacency matrix $\hat{A}$ [29] that

expresses global relationships. This combination guides the 2D/3D hand pose regressors to provide an initial promising pose using (4.2) and (4.4).

$$R_{2D} = \text{RelationsFunction}(L_{2D}) \quad (4.1)$$

$$H_{2D}^{k+1} = \text{ReLU}(\hat{A}H_{2D}^k W_A^k R_{2D} H_{2D}^k W_R^k) \quad (4.2)$$

Where: L_{2D} is the output of the 2D classifier, H_{2D}^0 is the concatenation of the ResNet-10 features and the 2D classifier probabilities. W_A^k , and W_R^k are the weights for the global and the local adjacency matrices, respectively.

Recent works demonstrate that 2D pose information improves the 3D pose estimation task [24,73,74]. We feed the concatenation of the estimated 2D poses, the image feature maps of the ResNet-10 network, the 2D classification probabilities, and the 3D classification probabilities (H_{3D}^0) to the 3D regressor. Furthermore, we exploit the 3D relations matrix (4.3) as an additional weighting mechanism during neighborhood aggregation to express the spatial dependencies between the 3D joints (4.4).

$$R_{3D} = \text{RelationsFunction}(L_{3D}) \quad (4.3)$$

$$H_{3D}^{k+1} = \text{ReLU}(\hat{A}H_{3D}^k W_A^k R_{3D} H_{3D}^k W_R^k) \quad (4.4)$$

Where: L_{3D} is the output of the 3D classifier.

4.2.5. Refinement Module

After getting reliable 2D/3D poses from the coarse stage, we freeze the network weights, and we improve the estimated 3D pose through a refinement module. The latter applies the proposed ANN algorithm to construct a per-pose adjacency matrix for the GCN. Previous works employ KNN-based graphs for other computer vision tasks. [56,75] create graphs from image features and apply residual KNN-based GCN for face clustering. [34] connects KNN 3D points cloud for RGBD semantic segmentation. However, fixing a unique number of nearest neighbors K for all the hand poses of the dataset does not capture the underlying structure of joints, which may affect information aggregation in the GCN. Moreover, the same hand pose may

include high-degree nodes (joints) as well as low-degree ones. Hence, the number of nearest neighbors should be different for each joint based on its distance to the others.

To alleviate this problem, we first calculate the distance matrix $D \in R^{21 \times 21}$ among 21 joints. Subsequently, we learn a global threshold T to be the criteria to select the related joints. Following the algorithm in Figure 4.4, if $D(i, j) \leq T$ then joint i and joint j are considered as adjacent and $R_{ANN}(i, j) = 1$, where R_{ANN} is the adjacency matrix. We feed R_{ANN} and the prior estimated pose H_{3D}^0 to GCN refinement to produce a more accurate pose (4.5). Note that the gain of the adaptative ANN-based GCN module appears when the coarse pose is predicted from our hybrid classification-regression approach. In other words, estimating 3D poses without using the proposed local constraints (2D/3D per-pose adjacency matrix) produces less accurate coarse poses. Thus, ANN may identify the wrong nearest neighbors in the adjacency matrix.

$$H_{3D}^{k+1} = ReLU(\hat{A}H_{3D}^k W_A^k R_{ANN} H_{3D}^k W_R^k) \quad (4.5)$$

Algorithm 2: Adaptative nearest neighbour algorithm.	
1	function ANN ($3D, \theta$);
	Input : $3D$: coarse predictions, θ : learnable threshold
	Output : 21×21 ANN relations matrix.
2	$matrix \leftarrow []$;
3	for $i=0:21$ do
4	$values \leftarrow []$;
5	for $j=0:21$ do
6	$values[j] \leftarrow MSE(3D[i], 3D[j])$;
7	end
8	$matrix[i] \leftarrow values$;
9	end
10	$matrix = float(matrix \leq \theta)$;
11	return $matrix$;

Figure 4.4: Algorithm of the adaptative nearest neighbor algorithm that identifies adjacent joints.

4.2.6. Loss Functions

Our training schema is composed of two stages coarse and refinement. The former involves two loss terms for the classification and the regression. (4.6) and (4.7) classify each joint into one of the blocks in the 2D/3D space, where $CE(\cdot)$ is a multi-

class Cross-entropy loss function. In (4.8) and (4.9), we use an L_2 regression loss between the predicted and the ground truth 2D/3D joints.

$$L_{classification2D} = \sum_{i=1}^{21} CE_i(Predictedclass2D_i, GT2D_i) \quad (4.6)$$

$$L_{classification3D} = \sum_{i=1}^{21} CE_i(Predictedclass3D_i, GT3D_i) \quad (4.7)$$

$$L_{regression2D} = L_2(Predicted2DPose, GT2DPose) \quad (4.8)$$

$$L_{regression3D} = L_2(Predicted3DPose, GT3DPose) \quad (4.9)$$

The coarse loss in (4.10) is a linear combination of the regression and the classification loss, where $\alpha_1 = 100$ and $\alpha_2 = 1$ for the regression and the classification terms, respectively. We empirically find the best hyperparameters by trying different configurations.

$$L_{coarse} = \alpha_1(L_{regression2D} + L_{regression3D}) + \alpha_2(L_{classification2D} + L_{classification3D}) \quad (4.10)$$

In the refinement stage, we use the L_2 loss between the ground truth 3D and the predicted one (4.11).

$$L_{refinement} = L_2(Fine3DPose, GT3DPose) \quad (4.11)$$

4.3. Experimental Settings

4.3.1. Implementation Details

We implement the proposed method using Pytorch v1.6 [76], CUDA v10.1, and cuDNN v7.6.4. We train our model on images of size 256×256 for 400 epochs using a batch size of 64. We initialize the weights of the ResNet-10 network using a normal distribution, where the $mean = 0$ and $std = 0.02$. We initialize the weights of the GCN networks using Xavier [77] where we set the gain to the square root of 2. We use NVIDIA's Apex mixed-precision training of 16-bits to speed up the training. It takes three days to converge on a single NVIDIA TITAN X GPU. We empirically found

that AdaDelta optimizer [78] is better than ADAM [79] for our task. Since the weights updating formula does not include a learning rate, we use the default optimizer parameters.

4.3.2. Datasets and Evaluation Metrics

To train and evaluate our proposed method, we need datasets that include RGB images with their 2D and 3D annotations. We conduct our experiments using the Stereo Hand Pose Tracking Benchmark (STB) [80] and Rendered Hand Pose (RHD) datasets [11]. These benchmarks are widely used for 3D hand pose estimation from a monocular RGB image.

STB [80] contains 30k and 6K real-world images for training and test, respectively. This dataset provides the RGB images with their corresponding 3D annotations. To obtain the 2D keypoints, we perform a projection operation from the 3D space using the given camera intrinsic parameters. This dataset is less challenging since it has fewer lighting, viewpoint, and background variations. Also, the images are not noisy and have high resolution (640×480).

RHD [11] is a very challenging hand dataset that contains synthetic images of resolution 320×320 collected from twenty persons performing 39 actions. The images are rendered from several viewpoints and using different illumination conditions. RHD dataset contains 41K images for training and 3K images for testing covering several backgrounds and hand shapes. Each sample in the dataset includes an RGB image, a hand mask image, the corresponding depth image, the 2D key-points, and the 3D hand pose.

To crop the hand from the image, we find the min_x , min_y , max_x and max_y of the 2D points. The top left corner (min_x, min_y) and the bottom right corner (max_x, max_y) are subtracted/added to a small threshold of 10 and 20 for the RHD and STB datasets, respectively, ensuring that the cropped hand image includes all the joints.

To evaluate our approach and quantitatively compare it against the state-of-the-art methods, we report the common metric mean end-point-error (EPE), which measures the average Euclidean distance between the ground truth and the estimated keypoints. The distances are expressed in pixels (px) and millimeters (mm) for 2D and 3D hand pose estimation, respectively. Furthermore, we show the area under the curve (AUC)

on the percentage of correct keypoints (PCK), which is widely used to evaluate human/hand pose estimation approaches with different thresholds. Although the main aim of the proposed method is to estimate the 3D hand pose, we also validate the classification performance using common metrics, including accuracy, precision, and recall.

4.3.3. Hand Model Representation

We follow the hand model of [9,80] (Figure 4.5), such that the hand is described by 21 joints, namely the wrist representing the hand palm and four joints (tip, dip, pip, and mcp) per finger (thumb, index, middle, ring, and pinky). Since we aim to identify the spatial relationships between each pair of joints in 2D/3D space, the order of the joints and fingers is important to understand the proposed per-pose constraints (adjacency matrices). We define each joint position by $joint_{2D} = (x, y)$ and $joint_{3D} = (x, y, z)$ for 2D and 3D coordinates, respectively.

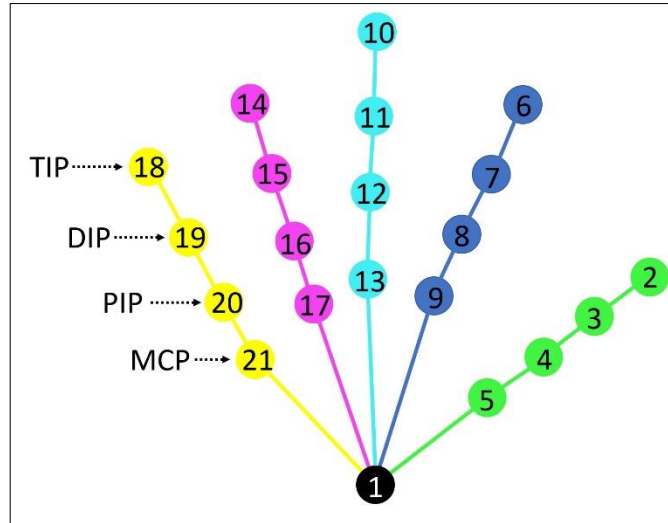


Figure 4.5: The description of our hand model.

4.4. Results and Discussions

To evaluate the proposed approach and find the best design choice, we conduct extensive experiments on STB and RHD datasets. We first compare our per-pose joint relationship constraints against the predefined and the global adjacency matrices. Also, we investigate the impact of classification probabilities on the 3D hand pose estimation

performances, and we compare our method against a regression-regression approach. Finally, we analyze the effect of the number of classes, and we show the benefit of the proposed refinement module. We select the best model to compare against state-of-the-art methods.

4.4.1. Effect of our Per-pose Adjacency Matrix

To validate the proposed hybrid classification-regression method, we compare it against several baselines using the aforementioned metrics. We define Baseline A as the simplest model that employs only the global adjacency matrix to estimate the 2D/3D hand pose. It uses neither classification probabilities and per-pose adjacency matrices nor the refinement module. In particular, we remove the classification modules that provide 2D/3D per-pose constraints from the framework in Figure 4.1. We can see from Table 4.2 that our proposed method (Full) outperforms Baseline A that provides lower scores in both AUC and EPE metrics proving the superiority of the proposed per-pose adjacency matrix over the global shared one. We incrementally add different modules to investigate the impact of the refinement module.

The second experiment compares our Full model against the predefined adjacency matrix on the STB and RHD datasets. Qualitative and quantitative results in Figure 4.6 and Table 4.3 show that the proposed classification-regression approach performs favorably against the predefined and global adjacency matrices. We can explain this by that the predefined matrix does not include relations between physically disconnected joints since it is obtained from the hand skeleton. Also, despite the changes in the input images, it remains the same for all images across all datasets. Although the global learnable adjacency matrix [29] changes from one dataset to another, it remains identical for the images which belong to the same dataset, which fails to capture pose dependent constraints. On the other hand, our Relations function outputs a specific adjacency matrix for each pose to determine joint relationships based on their spatial locality. Besides, it creates relations between physically disconnected joints. In other words, for each image, the location of each joint to the others in the low-level features (image pixels) is encoded in the high-level features. This process infers relationships between distant patches in the image by sliding the convolution filters over the output of the last layers.

Table 4.2: Comparison of different adjacency matrices in the coarse stage on STB and RHD datasets with EPE [mm] and AUC metrics.

	Method	Mean EPE	AUC (20-50)
STB	Ours	7.19	0.995
	Global adj. matrix	9.75	0.976
	Predefined adj. matrix	12.66	0.955
	Regression-regression	11.92	0.958
RHD	Ours	14.76	0.927
	Global adj. matrix	19.16	0.858
	Predefined adj. matrix	22.72	0.829
	Regression-regression	21.55	0.837

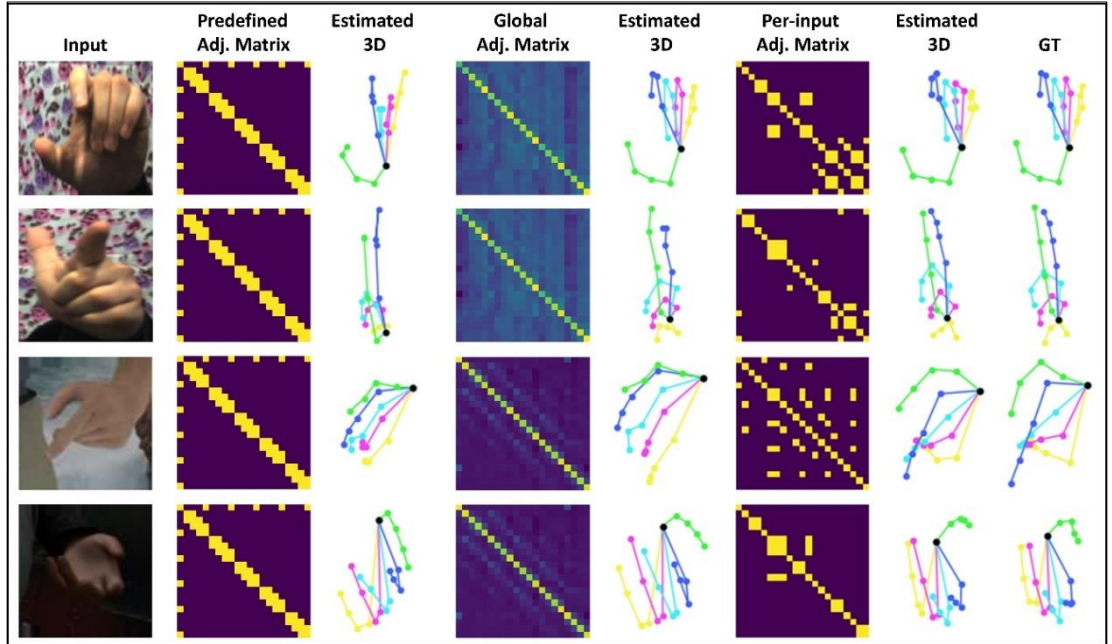


Figure 4.6: Qualitative comparison of 3D hand pose estimation between the predefined, the global, and our per-pose adjacency matrices.

Table 4.3: Ablation studies of different baselines with EPE [mm] and AUC metrics on the RHD dataset.

Method	EPE	AUC
Baseline A: full w/o classification and refinement modules	19.16	0.858
Baseline B: full w/o refinement (coarse)	14.76	0.927
Baseline C: full w/ FC refinement	14.68	0.928
Baseline D: full w/ KNN refinement	13.43	0.935
Baseline E: full w/o classification probabilities	16.02	0.895
Full	12.81	0.939

4.4.2. Effect of the Number of Classes

We conduct a set of experiments to find the best number of 2D/3D blocks (classes) that provide the best 3D pose estimation performances on the RHD [11] dataset. We compare the effect of 4, 9, 16, and 25 classes by dividing the 2D space into (2×2) , (3×3) , (4×4) , and (5×5) blocks, respectively. Besides, we compare the effect of 8, 27, 64, and 125 classes by dividing the 3D space into $(2 \times 2 \times 2)$, $(3 \times 3 \times 3)$, $(4 \times 4 \times 4)$, and $(5 \times 5 \times 5)$ blocks, respectively. We investigate several combinations of 2D and 3D class numbers (Table 4.4). We also illustrate 2D and 3D adjacency matrices for small and large classes. Figure 4.7 shows that having a small 2D/3D class number (e.g., 4 and 8) increases the adjacent joints and the matrix will not effectively express relations since most joints will be related to each other. Meanwhile, having a large number of classes (e.g., 25 and 125) produces an identity-like adjacency matrix. Consequently, it cannot discover the relation between physically disconnected joints. Also, all the poses will have the same adjacency matrix (same problem as the predefined and the global adjacency matrices). We empirically fix the number of classes to 16 and 27 for the 2D and 3D classifiers, respectively (Table 4.4). This configuration achieves good regression performances and constructs meaningful per-pose relation matrices (Figure 4.7). Table 4.5 shows that there exist other combinations that report better detection results (E.g., 4 and 8 classes for 2D and 3D, respectively). But they do not outperform the selected configuration in the 3D pose estimation performance, which is our target task.

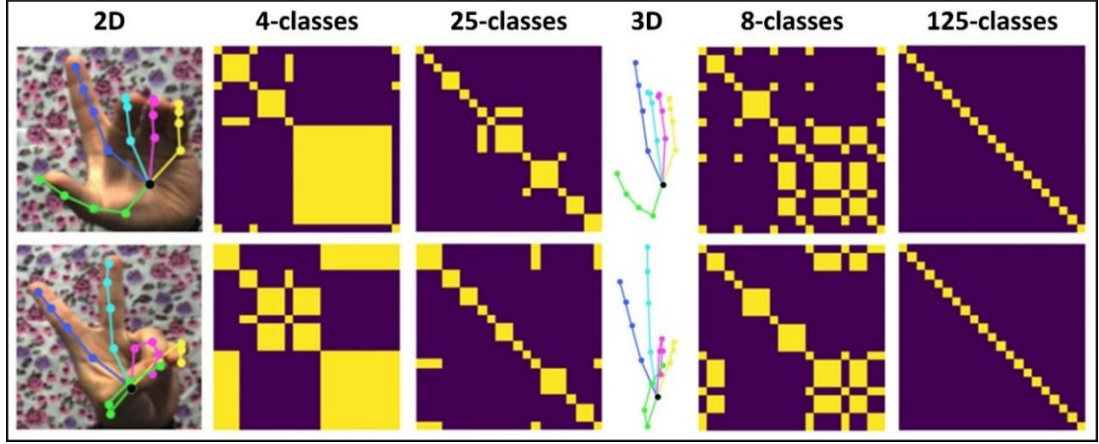


Figure 4.7: Effect of the number of classes on the joint relationship constraints.

Table 4.4: Effect of the number of classes on the 3D hand pose estimation results with EPE [mm] and AUC metrics.

Combinations 2D and 3D	4 and 8	9 and 27	9 and 64	16 and 27	16 and 64	25 and 128
Mean EPE	16.72	14.95	15.44	12.81	15.26	20.15
AUC (20-50)	0.887	0.925	0.916	0.936	0.912	0.842

Table 4.5: Effect of the number of classes on our 2D/3D classifiers. Each triplet in the classification metrics represents the accuracy, recall, and precision.

2D classes	2D classification metrics	3D classes	3D classification metrics
4	99.98/99.94/99.88	8	99.97/99.92/99.82
9	99.96/99.90/99.81	27	99.83/99.78/99.7
16	99.91/99.87/99.78	64	95.15/94.92/94.53
25	99.86/99.82/99.73	125	88.11/86.74/85.23

4.4.3. Regression-regression vs. Classification-regression

In this section, we compare our approach against the regression-regression approach. Specifically, we remove classification modules, and we estimate initial 2D and 3D hand joint locations using 2D heatmap and simple 3D regression, respectively. Subsequently, we calculate the distance matrix $D \in R^{21 \times 21}$ among 21 joints. We empirically select a threshold of 40mm to construct a per-pose adjacency matrix in the

3D space. We can see from Table 4.3 that our model exhibits better performance than the regression-regression approach. Also, we qualitatively compare the two methods by showing the estimated 3D poses and the constructed ANN matrices, which are used in the refinement stage. Figure 4.8 shows that our model estimates more accurate 3D coarse poses, and it exhibits lower errors since our ANN matrix is more similar to the ground truth one. We can explain this by that using the regression-regression approach in the coarse stage may identify wrong nearest neighbors because the initially estimated poses are not sufficiently accurate. Furthermore, in addition to the constructed per-pose adjacency matrices, our approach concatenates the classification probabilities with the extracted image features for each joint. This operation gives information about the confidence of the classifiers on each joint to the 2D/3D regressor. In contrast, the regression-regression approach uses only the adjacency matrix to guide the regressors. To investigate the impact of the classification probabilities, we perform Baseline E. Specifically, we give only the image features as input to the regressors, and we use our per-pose adjacency matrices from the classifiers. Table 4.2 shows that Baseline E exhibits inferior performance compared to our Full model, which proves that the classification probabilities are an essential component of the proposed approach. We note that our refinement stage is a regression-regression approach. After getting a reliable 3D coarse pose in the first stage, we improve it using another GCN-based module. The input to the latter is the constructed per-pose ANN adjacency matrix and the 3D coarse pose. The impact of the refinement module is investigated in the next section.

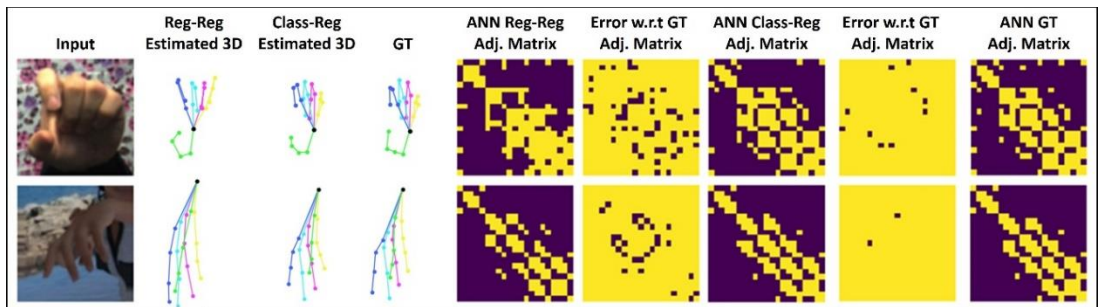


Figure 4.8: Qualitative comparison of regression-regression and our classification-regression approaches.

4.4.4. Effect of the Refinement Network

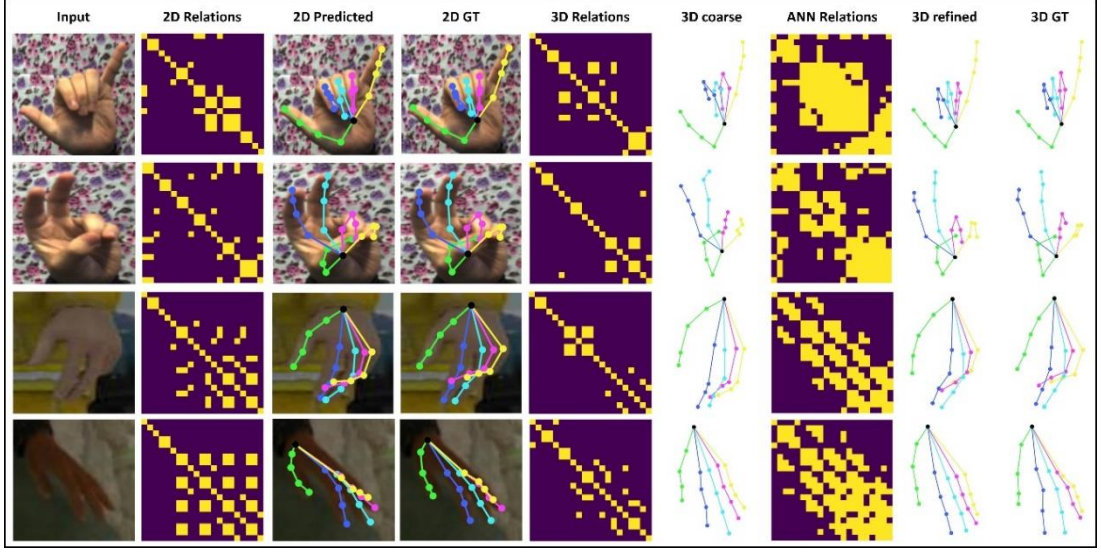


Figure 4.9: The intermediate outputs of our method on STB and RHD datasets.

As mentioned previously, our approach estimates an initial 3D hand pose in the coarse stage. In the second stage, the refinement network takes as inputs the estimated coarse pose and the learned ANN adjacency matrix to obtain a more accurate pose. To investigate the impact of this module, we report the quantitative results of the coarse prediction (Baseline B) in Table 4.2.

Also, Figure 4.9 shows the qualitative results of the coarse estimation, the refined estimation, and the learned per-pose adjacency matrices. We can see that the refinement module improves the coarse prediction and achieves more reliable qualitative and quantitative results. We can explain this by that the proposed ANN technique uses the estimated coarse 3D poses to provide a precise per-pose adjacency matrix. Hence, exploiting this information at this stage yields more accurate spatial dependencies and improves performance. Table 4.2 shows a significant margin between the full model and Baseline A. The latter estimates the coarse pose without exploiting per-pose constraints to identify spatial relations. Therefore, applying the proposed ANN on Baseline A identifies wrong related joints and leads to poor performance.

The GCN-based refinement modules outperform the simple FC refinement module (Baseline C) demonstrating the superiority of GCN to fit the problem of hand pose estimation. To validate the proposed ANN algorithm, we compare it against KNN

(Baseline D), where we empirically fix K to 5. Results show that the ANN-based refinement module outperforms the KNN-based version (Table 4.2). We can explain this by that the KNN adjacency matrix learns the same number of adjacent joints regardless of the pose and the joint locations. On the other hand, we can see from (Figure 4.9) that the number of neighboring joints in the ANN adjacency matrix is based on the joint distribution in the pose. We note that after training, the model learns the threshold parameter $T_{STB} = 52mm$ and $T_{RHD} = 44mm$ for the two datasets.

4.4.5. Qualitative Results

In addition to the quantitative validation, we show qualitative results of the proposed 2D/3D hand pose estimation method on STB and RHD datasets. Figure 4.9 presents all the intermediate outputs, including the predicted poses and the learned per-pose adjacency matrices. We can see that the estimated 2D and 3D poses are very close to the ground truth, demonstrating that the proposed approach achieves accurate results for both 2D and 3D hand pose estimation. Furthermore, 2D, 3D, and ANN adjacency matrices are interpretable, and they reflect spatial relationships between the predicted joints that can be physically connected or disconnected in the hand skeleton.

Figure 4.10 and Figure 4.11 show more qualitative results of the 3D hand pose estimation task on STB and RHD datasets with different views. We can see that the estimated 3D poses are reliable even in the case of self-occlusion and complex background textures. We also report more qualitative results for 2D hand pose estimation in the appendix.

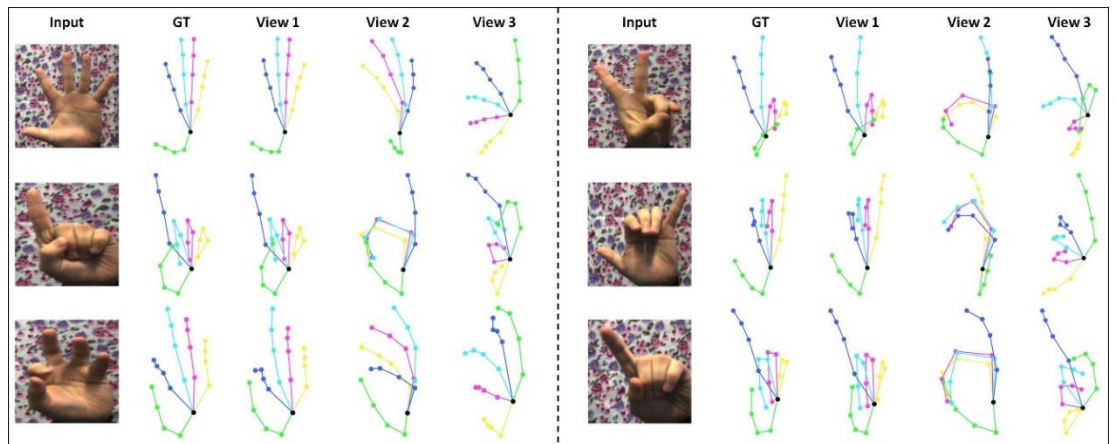


Figure 4.10: 3D hand pose estimation results on the STB dataset.

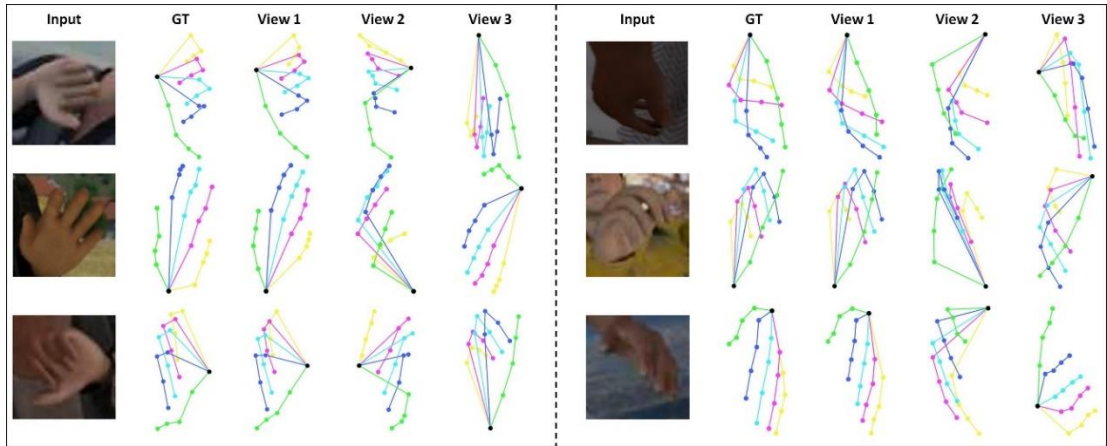


Figure 4.11: 3D hand pose estimation results on the RHD dataset.

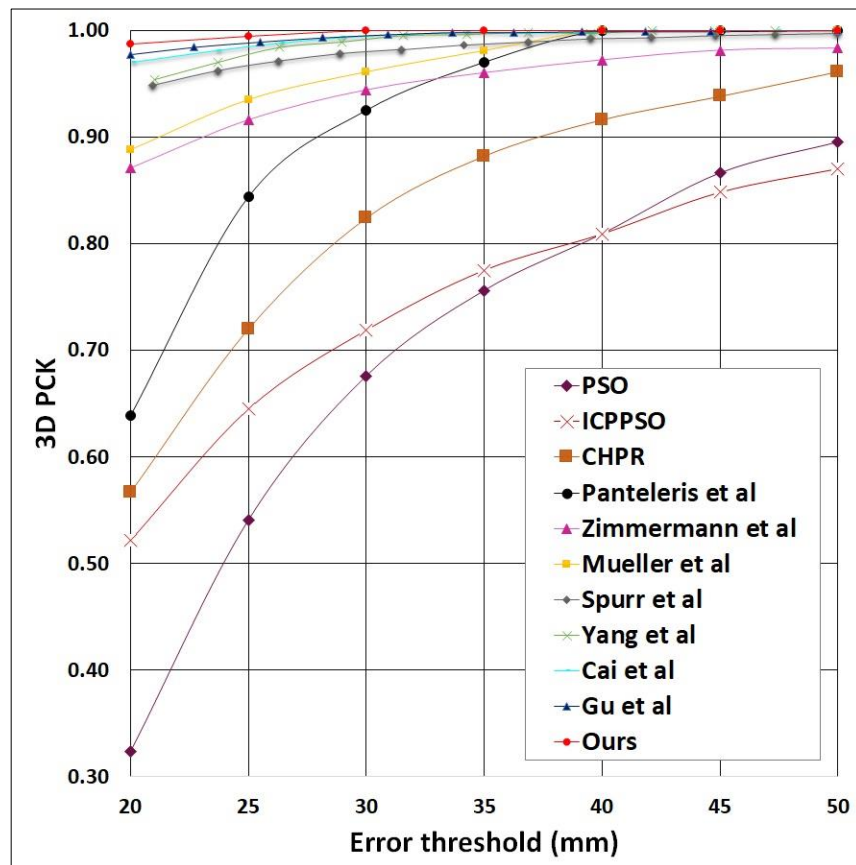


Figure 4.12: Comparison with the state-of-the-art methods on the STB dataset

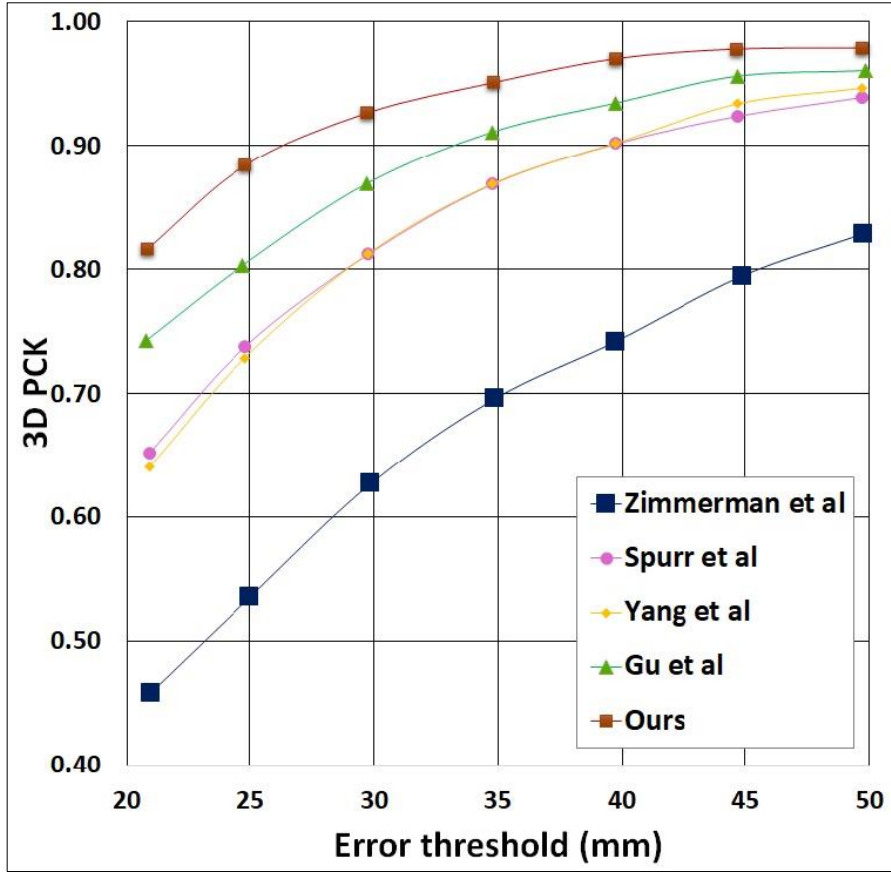


Figure 4.13: Comparison with the state-of-the-art methods on the RHD dataset.

Table 4.6: Comparison with the state-of-the-art methods on RHD and STB datasets using Mean EPE [pixel].

Methods	RHD	STB
Zimmermann et al	9.14	5
Iqbal et al	3.57	-
Ours	2.91	1.56

Table 4.7: Comparison with the state-of-the-art methods on RHD and STB datasets using AUC.

Methods	RHD	STB
Zimmermann et al [16]	0.675	0.948
Muller et al [17]	-	0.965
Spurr et al [19]	0.849	0.983
Cai et al [20]	0.887	0.994
Iqbal et al [38]	-	0.994
Yang et al [24]	0.849	0.991
Boukhyama et al [22]	-	0.994
Baek et al [34]	0.926	0.995
Ge et al [21]	0.920	0.998
Zhang et al [27]	0.901	0.995
Cai et al [30]	-	0.995
Guo et al [29]	0.933	0.998
Zhao et al [36]	0.872	0.987
Gu et al [25]	0.887	0.996
Ours	0.939	0.998

Table 4.8: Comparison with the state-of-the-art methods on RHD and STB datasets using Mean EPE [mm].

Methods	RHD	STB
Zimmermann et al [16]	30.42	8.68
Spurr et al [19]	19.73	8.56
Yang et al [24]	19.95	8.66
Gu et al [25]	17.11	7.27
Zhao et al [27]	-	8.18
Ours	12.81	6.58

4.4.6. Comparisons with State-of-the-art

We report the quantitative results of the 2D hand pose estimation task with EPE (pixel), and we compare it against [11,19]. We can see from Table 4.6 that our approach achieves low errors in the two datasets and outperforms the state-of-the-art, proving that the 2D GCN-based regressor benefits from the learned per-pose constraint (2D relations matrix).

Since our main aim is to estimate the 3D hand pose, we compare the proposed method against several competitive state-of-the-art on RHD and STB datasets with all the aforementioned metrics. For a fair comparison, we follow previous works preprocessing of the RHD dataset [11], and we validate the proposed method under the same evaluation conditions and standards and using the same data split. We use a range of 20 – 50 (*mm*) for the 3D PCK metric since it is a well-known standard criterion for RHD and STB datasets. Figure 4.12 shows the 3D PCK results comparison on the STB dataset of our approach and several state-of-the-art methods [7,9,10,13–15,23,51–53]. Our method outperforms [7,9,10,13,15,51–53] and achieves competitive performances on the STB dataset to [16,27]. To further evaluate our approach, we report the 3D PCK curves of the RHD dataset. Figure 4.13 shows that our method outperforms the compared state-of-the-art methods [9,11,15,16] on all the PCK thresholds. We present the results of the two datasets with additional state-of-the-art methods [16–19,21,22,54–56]. We can see that the proposed approach has the highest score in the RHD dataset, improving the AUC value to 0.939. Also, there is a small improvement in the STB dataset (0.998). Note that the advantage of our approach appears in the RHD dataset that includes low-resolution images with complex backgrounds and poses. STB dataset is less challenging since it contains simple backgrounds with a restricted number of poses. Finally, Table 4.8 lists the EPE metric comparison against five state-of-the-art methods [7,9,13,14,56]. We can see that the proposed method has superior performances providing 12.81 and 6.58 mean EPE on RHD and STB datasets, respectively. This is mainly due to the learned per-pose geometric constraint (adjacency matrix) and the ANN-based refinement module. We note that choosing a suitable number of classes has a significant impact on the 3D hand pose estimation accuracy. Using the proposed approach with a very small/large number of classes yields inferior performance.

4.4.7. Computation Complexity

Although the proposed framework includes six modules, it is computationally efficient, where the running time on NVIDIA TITAN X GPU is 7ms. We can explain this by that the network consists of a lightweight feature extractor (ResNet10) and five GCN-based modules, which are significantly faster than convolutional neural networks [29]. While the classification modules (per-pose constraints) notably contribute to improving the 3D hand pose estimation accuracy, it necessitates only an additional 1ms per image, which does not speed down the inference time.

4.5. Summary

Hand pose estimation is a crucial part of a wide range of augmented reality and human-computer interaction applications. Predicting the 3D hand pose from a single RGB image is challenging due to occlusion and depth ambiguities. GCN-based (Graph Convolutional Networks) methods exploit the structural relationship similarity between graphs and hand joints to model kinematic dependencies between joints. These techniques use predefined or global learned joint relationships, which may fail to capture pose dependent constraints. To address this problem, we propose a two-stage GCN-based framework that learns per-pose (per-image) relationship constraints. Specifically, the first phase quantizes the 2D/3D space to classify the joints into 2D/3D blocks based on their locality. This spatial dependency information guides the regression branch to estimate reliable 2D and 3D poses. The second stage further improves the 3D estimation through a GCN-based module that uses an adaptative nearest neighbor algorithm to determine joint relationships. Extensive experiments show that our two-stage GCN approach yields an efficient model that produces accurate 2D/3D hand poses and outperforms the state-of-the-art on two public datasets.

5. A GRAPH-BASED ABSOLUTE 3D HAND POSE ESTIMATION USING A SINGLE RGB IMAGE

5.1. Introduction

Hand pose estimation is one of the fundamental tasks in computer vision. Although studied extensively in the literature, accurate 3D hand pose estimation remains challenging due to strong articulation, self-similarity, and self-occlusions of the hand.

With the fast advancement of deep learning techniques and the widespread availability of commodity depth sensors, RGB-D-based 3D hand pose estimation methods achieve reliable results [39,50–53]. However, this hardware requires high-power consumption and is limited to indoor environments restricting many applications. Other approaches alleviate occlusion problems using multiple RGB cameras located at different viewpoints [42,50,81]. These techniques are difficult to setup and cannot operate in unconstrained scenes (e.g., community videos). As a result, 3D hand pose estimation from a single RGB image has gained considerable attention since it is more accessible and does not pose any setup overhead. However, this task is more challenging than the aforementioned approaches because it exhibits depth and scale ambiguities. Recently, several learning-based methods tackle this problem by designing effective models trained on large-scale annotated datasets. Inspired by the natural graph representation of the hand, recent discriminative studies use GCNs [24] to model kinematic relationships between joints [25–27,29].

Most existing methods estimate a scale-normalized root-relative pose. In particular, they predict the positions of hand joints relative to a reference point located on the hand (e.g., the wrist). The global joint coordinates in the world coordinate system are unknown. They assume that the depth of the root joint and the global hand scale is provided at test time [9,11]. In contrast, absolute hand pose estimation finds the hand joint positions in camera-centered coordinates [10,17,57]. This extends the application scope to AR/VR, where headsets require camera distance-aware 3D hand pose estimation approaches to perform hand-object interactions. Also, the absolute pose is crucial in multi-hand images because a root-relative pose may represent different global poses. Thus, estimating the absolute pose is essential to recover the spatial layout of the whole scene. However, this task is too challenging. One difficulty

comes from the fact that it presents an ill-posed problem due to the irreversible geometry. In particular, a 2D point in the image plane can correspond to multiple 3D points in world space. Besides, recovering the global scale of the hand is challenging due to depth ambiguity. Finally, the absolute pose resides in a larger space than the root-relative pose, which changes the underlying optimization process.

We observe that the input of the previous approaches is the cropped hand image. They do not exploit the global scene image that contains valuable information, such as the size of the hand that provides a clue about its depth (e.g., big hands mean small depth and vice-versa). Also, the interaction of hands with other commonly seen objects, specifically other body parts such as shoulders and heads, can help to learn the hand position. Motivated by this observation, we propose a GCN-based method that consists of three stages: the first one extracts high-level features from the original scene and the cropped hand images to estimate the global and the local 2D keypoints. The second stage combines the extracted image features with the predicted 2D local hand pose to estimate a root-relative non-scale normalized 3D hand pose. Finally, we estimate the absolute 3D hand pose using the previous stage outputs (Figure 5.1). Besides, we propose a feature matching (FM) loss function to simplify the regression task and improve the performance. Specifically, we correlate the extracted image features to the root-relative 3D joints to make them sufficiently separable.

We conduct extensive experiments on four public benchmarks [9,34,36,80]. Quantitative and qualitative results demonstrate that the proposed method estimates accurate 2D keypoints and 3D hand poses in challenging scenarios, such as complex texture backgrounds, self-occluded hands, and object-occluded hands. Furthermore, it achieves competitive results against several state-of-the-art methods with an efficient running time. We extend our experiments to our newly built dataset for 2D/3D hand pose estimation. It includes 30 thousand samples collected using the leap motion sensor [81] with different lighting conditions and high pose variation. Each sample contains the hand bounding box, the 2D keypoints, the 3D pose, and the corresponding RGB image. We can summarize our contributions as follows:

- We propose a multi-stage GCN-based approach that exploits the global scene and the cropped hand images to estimate an accurate absolute 3D hand pose from a single RGB image.

- We propose a feature matching loss to separate the extracted features of the hand image and simplify the 3D pose regression task.
- To evaluate the proposed approach, we conduct extensive experiments on synthetic and several real-world datasets. The reported quantitative and qualitative results demonstrate that our method outperforms state-of-the-art and estimates accurate 2D/3D hand poses.

The rest of this paper is organized as follows: Section 5.2 explains the proposed method in-depth and describes the most important modules of our framework. Section 5.3 describes the experimental setting. Section 5.4 presents experimental results on public datasets and compares the proposed approach against the state-of-the-art methods. Finally, Section 5.5 presents the summary of this study.

5.2. Methodology

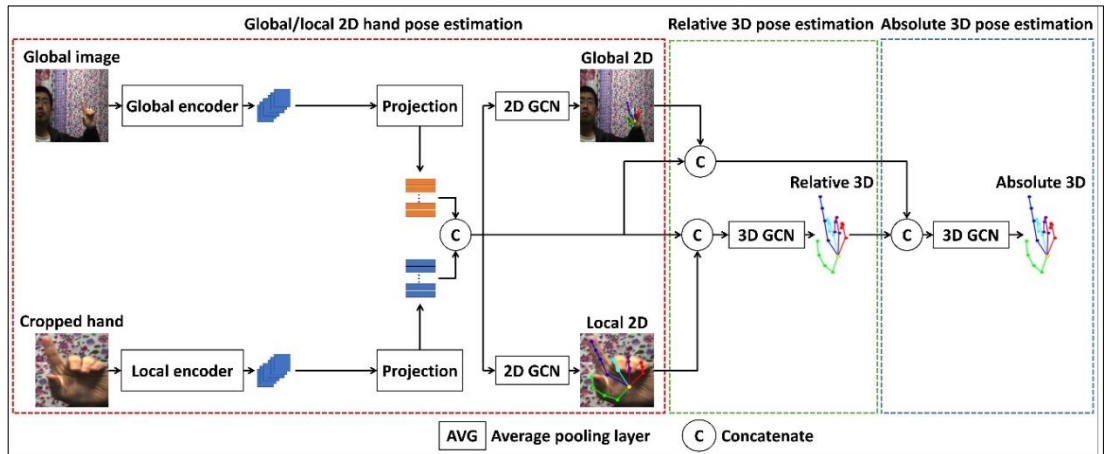


Figure 5.1: Schematic overview of our proposed approach for monocular RGB-based absolute 3D hand pose estimation.

5.2.1. Architecture

Estimating an absolute 3D hand pose from a single RGB image is challenging due to depth and scale ambiguities. Most existing 3D hand pose estimation methods crop the hand area in an input image with a ground truth bounding box or the bounding box that is predicted from a hand detection model. The cropped hand image is fed into the 3D pose estimation module to predict a scale-normalized root-relative pose. As their models take a single cropped image, estimating the absolute camera-centered

coordinate of each key point is difficult. To handle this issue, we exploit the original scene image that includes relevant information that can constrain the problem, such as the size of the hand that provides a clue about its distance from the camera (depth). More specifically, a small hand indicates being far from the camera (large depth value) and vice-versa. (Figure 5.2). Also, the scale of the hand relative to the other objects in the scene helps to reduce the depth and scale ambiguities. In particular, when the hand pose, scale, and location change from one image to another, other objects in the background remain motionless. Their scale invariance gives information about the distance between the hand and the camera (z coordinate).



Figure 5.2: Illustration of two samples with the same relative 3D pose. Unlike the cropped hand image, the global scene preserves the size of the hands that gives a clue about the distance from the camera.

Our framework involves three training stages: global/local 2D pose, root-relative 3D pose, and absolute 3D pose estimation (Figure 5.1). In the first stage, we use two customized ResNet-10 networks (Table 5.2) as a backbone to extract image feature maps from the original scene image (global encoder) and the cropped hand image (local encoder). We feed their feature maps to projection layers (Convolution layer) that reduce the dimensionality of the feature maps from $(256, 7, 7)$ and $(256, 4, 4)$ to $(21, 7, 7)$ and $(21, 4, 4)$ for the global and local image, respectively. We flatten the last two dimensions of the feature maps from $(B, 21, W, H)$ to $(B, 21, W \times H)$, where B is the batch size, W and H are the width and the height of the feature maps, respectively. The reason for this transformation is to get the correct input format for the GCN-based regressors, where each joint has a feature vector of size $W \times H$. We concatenate the feature vectors from the global and the local hand image. The resulting tensor of size $(21, 65)$ is given to global and local GCNs to estimate 2D keypoints location in the original scene, and the cropped hand images

Recent works demonstrate that 2D key points are crucial for the 3D pose estimation task [28,88,89]. Thus, the second stage feeds the concatenation of the estimated local 2D poses and the global/local image feature vectors to the root-relative 3D regressor. Instead of solving quadratic equations or putting assumptions about the hand, our absolute 3D regressor learns depth from the cropped hand image and the global scene features. Meanwhile, it exploits the predicted root-relative 3D pose and the global 2D key points to estimate (x, y) coordinates in the 3D space.

Table 5.1: The architecture of the customized ResNet-10 network.

Layer	In	Out	Kernel	Stride	Padding
Conv2D	3	64	3×3	2	1
BN	64	64	-	-	-
ReLU	64	32	-	-	-
Max-pool	64	64	3	2	1
Res-Block	64	64	3×3	1	1
Res-Block	64	128	3×3	1	1
Res-Block	128	256	3×3	1	1
Res-Block	256	256	3×3	1	1

5.2.2. Hand Pose Regression using Graph Convolution Network

Unlike CNNs that only handle fixed structures, GCNs treat irregular ones in a non-euclidian space. Recent 3D hand pose estimation methods adapt GCNs since the hand joints have a basic graph data structure with proper geometry details. Inspired by these studies [25–27,29], we apply Chebyshev spectral GCN [24] to estimate the 3D hand joint coordinates from a single RGB image.

There are two joint relationships representation approaches (adjacency matrices) in 3D hand pose estimation. The first technique uses a predefined hand skeleton-based adjacency matrix to model kinematic dependencies between joints. However, this approach misses potential relationships of physically disconnected joints. Also, it remains the same for all datasets. [29] addresses this limitation by replacing the skeleton relationships with global learnable joint relationships constraints. More specifically, the GCN network contains a weight matrix of size 21×21 that is updated during training, and the values are fed as input to the GC

layers. The final weights in the matrix define the adjacency matrix learned from the dataset. In our 2D/3D GCN-based regressor, we use this propagation rule to estimate 2D local, 2D global, 3D root-relative pose, and 3D absolute pose. Our GCNs contains two graph convolution layers, where the number of features is set to 128. The output layer yields 2 and 3 features for 2D and 3D hand pose estimation, respectively. Since we concatenate outputs of different ranges from the previous stages, we use layer normalization before feeding the inputs to the GCNs.

5.2.3. Loss Functions

Our network is supervised by FM loss and the 2D/3D regression losses. Let $P_{absoute} = (X, Y, Z)$ be the absolute 3D pose of a hand joint. Also, let the global 2D pose p_{global} be the projection of $P_{absoute}$ on the image plane using intrinsic and extrinsic camera parameters K , where: $p_{global} = K \times P$. To create the local 2D pose p_{local} , we crop p_{global} using the minimum and the maximum values of the x and y axis, respectively. To get the root-relative 3D pose, we subtract P_{palm} from $P_{absoute}$: $P_{relative} = P_{absoute} - P_{palm}$, where P_{palm} is the absolute 3D position of the root joint.

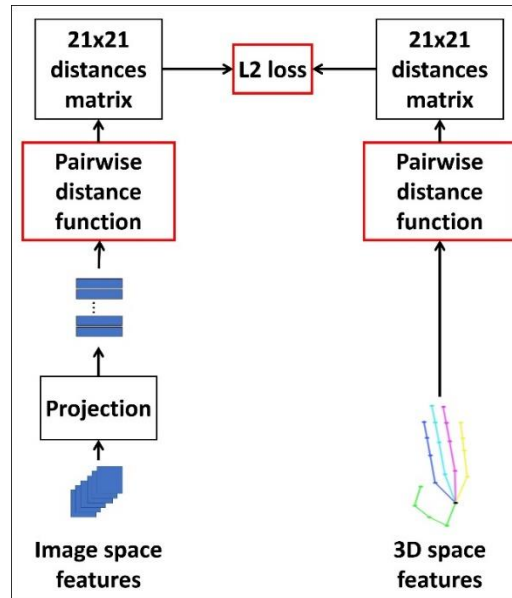


Figure 5.3: Our feature matching loss measures the distance between the pairwise distance matrices of the cropped hand image feature maps and root-relative 3D hand pose.

5.2.4. Features Matching

Figure 5.3 describes our proposed feature matching loss. We feed the projected feature maps of the cropped hand image into a pairwise distance function, where we calculate the distance between each feature vector pair in the $(21, 16)$ tensor. Meanwhile, we calculate the distance between each joint pair of the ground truth root-relative 3D hand pose. We minimize the distance between the two matrices to enforce agreements between image and 3D features in an intermediate space. This correlation simplifies the regression task to produce an accurate root-relative 3D hand pose. We name our features matching loss as $FM(.)$.

5.2.5. Optimization

In this section, we explain our loss terms for each training stage. In the first stage, (5.1) and (5.2) calculates the L_2 loss between the predicted and the ground truth global/local 2D poses, respectively.

$$L_{global} = \|\hat{p}_{global} - p_{global}\|_2 \quad (5.1)$$

$$L_{local} = \|\hat{p}_{local} - p_{local}\|_2 \quad (5.2)$$

Furthermore, we employ the FM loss (5.3) between the projected features of the cropped hand image (P_L) and the ground truth relative 3D pose ($P_{relative}$).

$$L_{FM} = FM(P_L, P_{relative}) \quad (5.3)$$

The overall loss of the first stage is the sum of all losses.

In the second stage, we calculate the loss between the estimated $\hat{P}_{relative}$ and the ground truth $P_{relative}$ root-relative 3D hand pose using (5.4).

$$L_{relative} = \|\hat{P}_{relative} - P_{relative}\|_2 \quad (5.4)$$

In the final stage, we measure the loss between the estimated $\hat{P}_{absolute}$ and the ground truth $P_{absolute}$ absolute 3D hand pose using (5.5).

$$L_{absolute} = \|\hat{P}_{absolute} - P_{absolute}\|_2 \quad (5.5)$$

5.3. Experimental Setting

5.3.1. Datasets

We conduct our experiments using four different datasets: STB [95], Multiview 3D Hand Pose dataset (MHP) [43], RHD [11] and First-Person Hand Action (FPHA) [41]. We also create the GTHD dataset to extend our experiments. The hand is represented by 21 joints: four keypoints (mcp, pip, dip, and tip) per finger (thumb, index, middle, ring, and little) and the hand palm.

The STB dataset is widely used to train and validate RGB-based 3D hand pose estimation methods. It contains 640×480 hand images with different illumination and background conditions. It includes 18K samples with their corresponding 3D poses covering random and counting gestures. To obtain the 2D keypoints, we perform a projection operation from the 3D space using the given camera intrinsic and extrinsic parameters.

MHP is a large-scale dataset that holds 21 hand motion videos collected from multiple views and divided into 60K, 15K, and 12760 images for the training set, validation set, and test set, respectively. Each sample includes RGB hand images, bounding boxes, 2D hand keypoints, and the 3D pose for different hand sizes and colors.

RHD is one of the most challenging datasets since it presents low-resolution noisy synthetic images (320×320) containing diverse hand textures with heavily occluded fingers. The images are rendered from varying camera angles and collected from twenty characters performing 39 actions in front of random backgrounds. RHD contains 41,258 and 2,728 samples for training and test set, respectively. Each one includes the RGB image, 2D key-points, 3D pose, depth map, and segmentation mask.

FPHA presents first-person videos of complex hand actions corresponding to daily human activities (open, close, and put) performed on different objects (milk, juice bottle, liquid soap, and salt). This dataset can be used for various tasks since it includes 3D hand pose labels, 6D object pose, and action categories. FPHA is a large and diverse dataset including 105,459 annotated frames obtained from 1175 videos performed by six actors.

To enrich our experiments, we build our new dataset (GTHD) using an RGB camera and a Leap Motion sensor [81]. It contains 60 thousand RGB images with the corresponding hand bounding boxes, 2D keypoints, and 3D poses. To estimate an absolute 3D hand pose, we use a single RGB camera. The latter provides an image with a resolution of 640×480 pixels. The leap motion controller senses the fingers to provide the 3D joint locations. Hence, a projection process from 3D space to the 2D image plane is necessary. We achieve this goal in two steps. In the first one, we use OpenCV to estimate specific intrinsic parameters of the camera. In the second step, we estimate the extrinsic parameters between the leap motion controller and the camera. To get the correct pose with its corresponding image, we synchronize the two sensors in time. Finally, to find the rotation and translation matrices, we manually mark one joint in a set of hand images and solve the *PnP* problem by computing the 3D-2D correspondences [82]. We capture images with high pose and lighting variations. We capture the images in an indoor environment using six distinct backgrounds. To obtain a vast variety in the pose space, three subjects with different skin tones and hand shapes perform several gestures, such as grasping, pointing, and counting. The motion of the hand was natural and unrestricted. To ensure sufficient lighting variations, we collect images at different times during the day. This allows various sampling of sunlight in the space. Our dataset contains 60 thousand RGB images with the corresponding hand bounding boxes, 2D keypoints, and 3D poses. We randomly split the GTHD dataset into a training set (75%), a validation set (10%), and a test set (15%).

5.3.2. Metrics

To evaluate the accuracy of the proposed method and compare it against the state-of-the-art, we report the three most common metrics in 3D hand pose estimation:

- EPE: End-Point Error measures the average Euclidean distance between the ground truth and the estimated keypoints. The distances are expressed in millimeters (mm) and pixels (px) for 3D and 2D hand pose estimation, respectively.
- PCK: considers the predicted joints as correct if the distance to the ground truth joint is within a given threshold. It is widely used to evaluate RGB-based 3D hand pose estimation approaches between 20-50mm on STB, MHP, and RHD datasets.

- AUC: instead of computing PCK for a single threshold, we show the area under the curve on PCK for different error thresholds to give a more accurate evaluation of the model performances.

5.3.3. Implementation Details

We implement the proposed method using Pytorch v1.8 [91], CUDA v10.1, and cuDNN v7.6.4. We use a batch size of 128, and we train our model for 150, 75, and 50 epochs per stage, respectively. We resize the original scene and the cropped hand images to 224×224 and 128×128 , respectively. Note that we keep the original aspect ratio during resizing of the cropped hand images. We initialize the weights of the ResNet-10 network using a normal distribution, where the *mean* = 0 and *std* = 0.02. We use the Adam optimizer [94] with a learning rate of 0.001 decayed using a Cosine learning rate scheduler. We initialize the weights of the GCN networks using Xavier [92] where we set the gain to the square root of 2. We use NVIDIA's Apex mixed-precision training of 16-bits to speed up the training. Our whole training process takes two days on average to converge using an NVIDIA TITAN X GPU.

5.4. Experimental Results

To evaluate the proposed approach and find the best design choice, we conduct extensive experiments on the aforementioned datasets. We investigate the impact of the original scene (global) and the cropped hand (local) image on the 3D hand pose estimation accuracy. We also analyze the effect of the proposed FM loss. We report qualitative and quantitative results of different baselines, and we select the best model to compare against the state-of-the-art methods.

Table 5.2: Ablation studies of 3D hand pose estimation on STB, MHP, RHD, and GTHD datasets with EPE [mm] and AUC metrics.

	Method	Mean EPE	AUC (0-50)	AUC (20-50)
STB	Full	6.21	0.716	0.998
	Full w/o FM loss	9.62	0.716	0.982
	Full w/o global	15.07	0.612	0.948
MHP	Full	10.23	0.771	0.954
	Full w/o FM loss	14.42	0.712	0.906
	Full w/o global	21.36	0.617	0.824
RHD	Full	12.09	0.748	0.915
	Full w/o FM loss	16.73	0.679	0.887
	Full w/o global	29.57	0.503	0.835
GTHD	Full	9.03	0.783	0.972
	Full w/o FM loss	13.23	0.720	0.924
	Full w/o global	19.71	0.625	0.861

5.4.1. Analysis of the Global and Local Modules

To validate the proposed approach, we perform Baseline A which employs only the cropped hand image as input and does not exploit any knowledge about the scene (global image). In particular, we remove the global feature extractor and the global 2D pose estimator modules from the framework in Figure 5.1. We can see from Table 5.2 and Figure 5.4 that the proposed method (Full) outperforms Baseline A that provides lower scores in both AUC and EPE metrics on all the datasets. We can explain this by that feeding the hand in its original scene gives a clue about its position in the camera coordinate system.

Table 5.3: Ablation studies of the root-relative and the absolute pose estimation stages on STB, MHP, and GTHD datasets with EPE [mm] metric.

	Method	STB	MHP	GTHD
Absolute pose	Full	6.21	10.23	9.03
	Baseline A: Full w/o global module	15.07	21.36	19.71
	Baseline B: Baseline A + hand bounding box	14.56	20.47	18.54
	Baseline C: Full w/o face pixels	11.08	-	12.81
	Baseline D: Full w/o background pixels	-	17.56	16.02
	Baseline E: Full w/o local module	9.53	15.34	14.96
Root-relative pose	Baseline A: Full w/o global	10.43	13.78	12.76
	Baseline E: Full w/o local	17.50	20.16	18.43

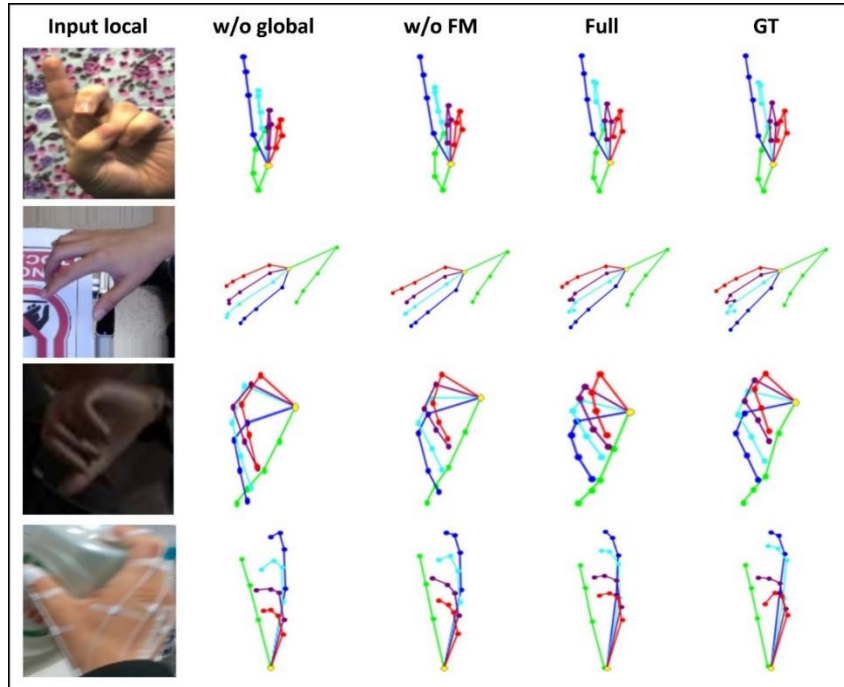


Figure 5.4: Ablation studies of the proposed approach.

To support our claim, we conduct three additional experiments. Initially, to investigate the impact of the hand scale in the original scene, we perform Baseline B, where the inputs are the cropped hand image and the hand bounding boxes. We can

see from Table 5.3 that Baseline B outperforms Baseline A confirming the benefits of the hand scale information. Meanwhile, it shows that other motionless objects in the scene help to reduce the depth ambiguities since it reports inferior performance compared to our Full model. To verify this claim, we perform Baseline C that examines the impact of the face on the performance of the Full model. More specifically, we use the face detector [97] to remove the face pixels from the scene image, and then we apply our proposed architecture in Figure 5.1. We conduct this experiment using STB and GTHD datasets since they include frontal faces with backgrounds and poses variation. The third experiment is Baseline D which investigates the impact of the background on MHP and GTHD datasets. In particular, it replaces the background pixel values with white color and performs the proposed architecture. Experimental results show the benefit of the face and background in improving the estimation accuracy. These results can be explained by that while the hand scale and position change, static objects provide clues about the changed scale and depth.

Finally, we show the impact of local and global modules that use the cropped hand and the original scene image, respectively. As seen from Table 5.3, the local module is more crucial than the global for the root-relative 3D hand pose estimation stage. We can explain this by that the network focuses on hand pixels and will not be affected by the background. In contrast, for the absolute 3D hand pose estimation stage, other objects in the scene and the original scale of the hand provide clues about its distance to the camera. We note that combining local and global modules yields the most effective baseline in both root-relative and absolute 3D pose estimation stages.

5.4.2. Effect of the Feature Matching Loss

To investigate the impact of the proposed FM loss, we report quantitative and qualitative results of Baseline Full w/o FM that performs the proposed approach using only L_2 loss between the ground truth and the predicted joint locations in 2D and 3D spaces. We can see from Table 5.2 and Figure 5.4 that the FM loss improves the model performance and achieves more accurate results. To explain this, we visualize the extracted features of the cropped hand image w/wo FM loss. Specifically, we project the feature vectors of each joint into the 2D space using t-SNE [98]. We can see from Figure 5.5 that the FM loss separates the extracted image features and more organized joint clusters appear in STB, MHP, and GTHD datasets. Having large margins between

the joints in the intermediate space facilitates the regression and improves the estimation accuracy. In contrast, Baseline Full w/o FM exhibits overlapped features, which makes the optimization task more challenging.

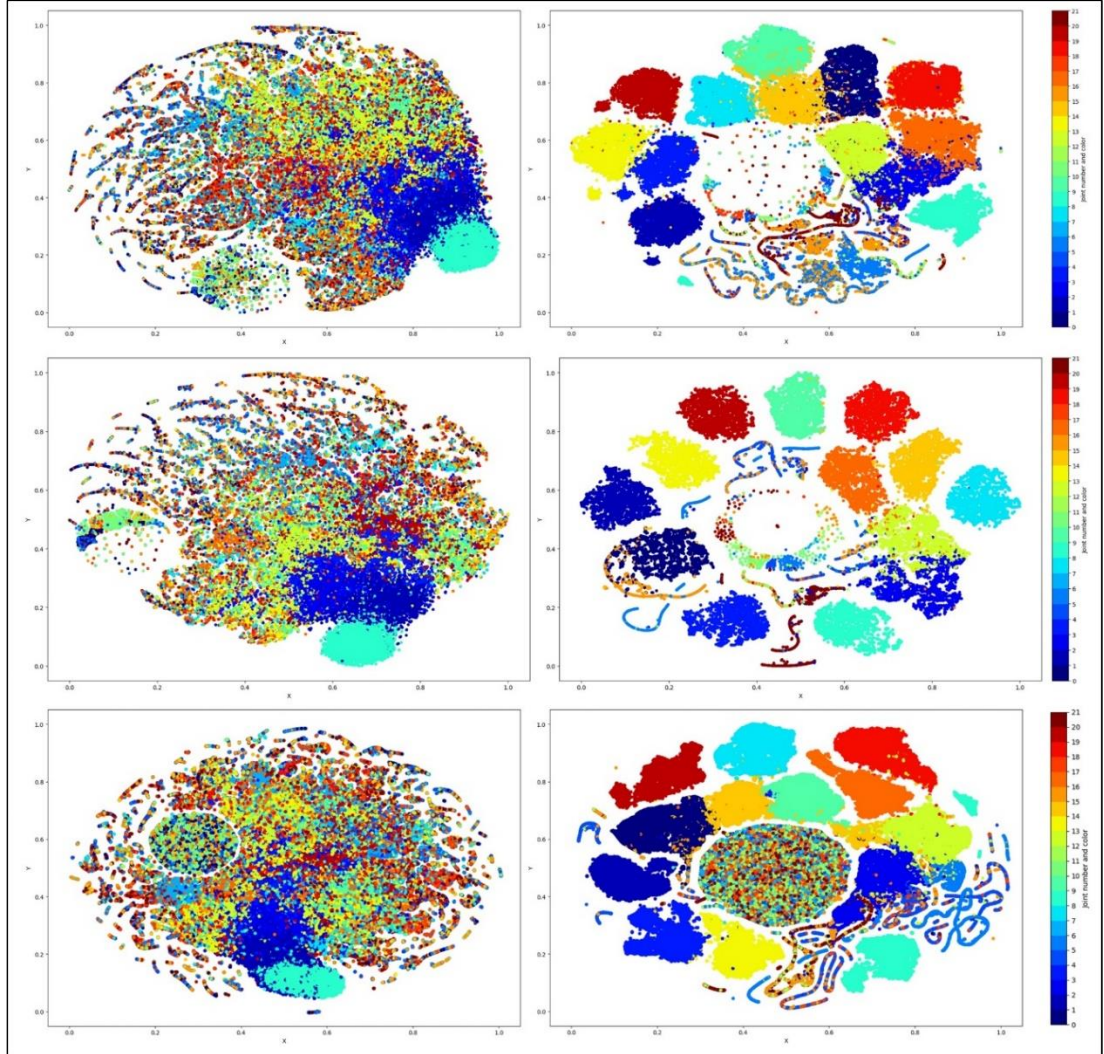


Figure 5.5: The impact of the proposed FM loss on the STB, MHP, and GTHD datasets.

Table 5.4: Comparison with the state-of-the-art methods on STB, MHP, RHD, and FPHA datasets using Mean EPE [mm].

Methods	RHD	STB	MHP	FPHA
Zimmermann et al [11]	30.42	8.68	-	-
Spurr et al [9]	19.73	8.56	-	-
Yang et al [18]	19.95	8.66	-	-
Gu et al [16]	17.11	7.27	-	-
Chen et al [48]	18	10.05	13.12	-
Chen et al [99]	-	11.3	16.2	-
Boukhayma et al [20]	-	9.76	-	-
Hernando et al [41]	-	-	-	11.25
Tekin et al [100]	-	-	-	16.15
Doosti et al [29]	-	-	-	6.81
Ours	12.09	6.21	10.23	6.64

Table 5.5: Comparison with the state-of-the-art methods on RHD and STB datasets using Mean EPE [pixel].

Methods	RHD	STB
Zimmermann et al [11]	9.14	5
Iqbal et al [19]	3.57	-
Ours	2.83	1.49

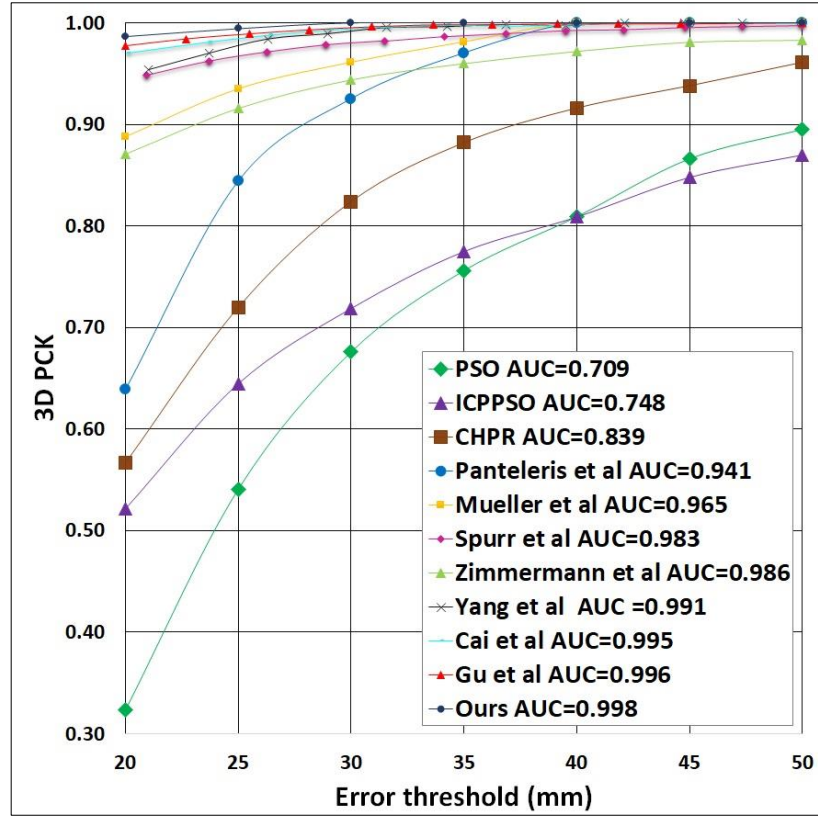


Figure 5.6: Comparison with the state-of-the-art methods on the STB dataset.

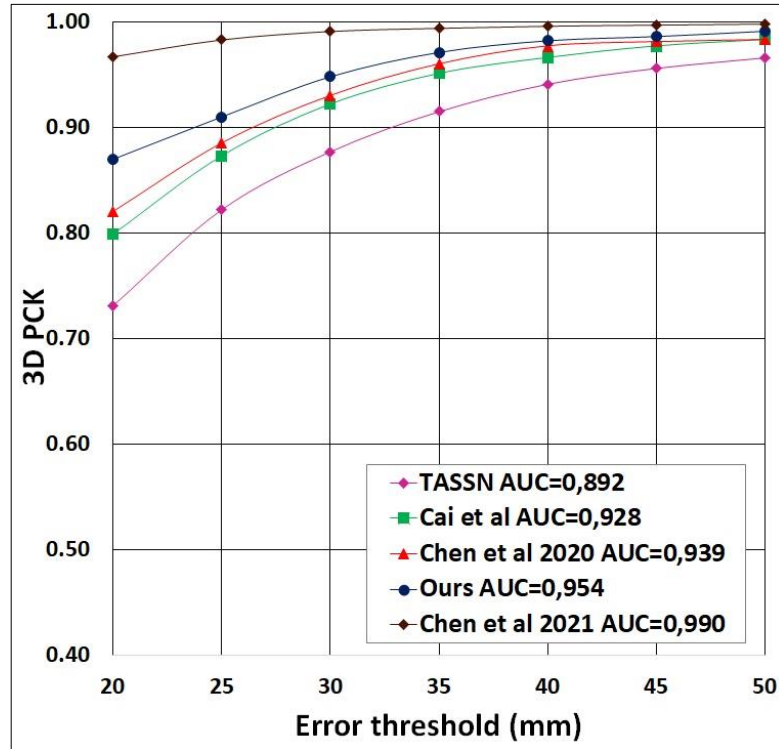


Figure 5.7: Comparison with the state-of-the-art methods on the MHP dataset.

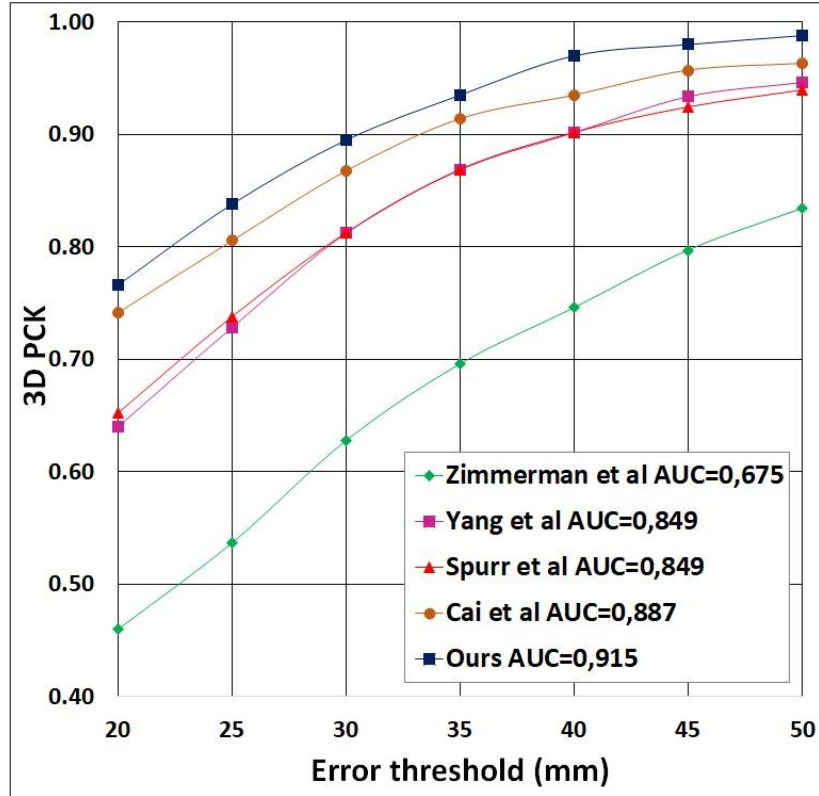


Figure 5.8: Comparison with the state-of-the-art methods on the RHD dataset using 3D PCK.

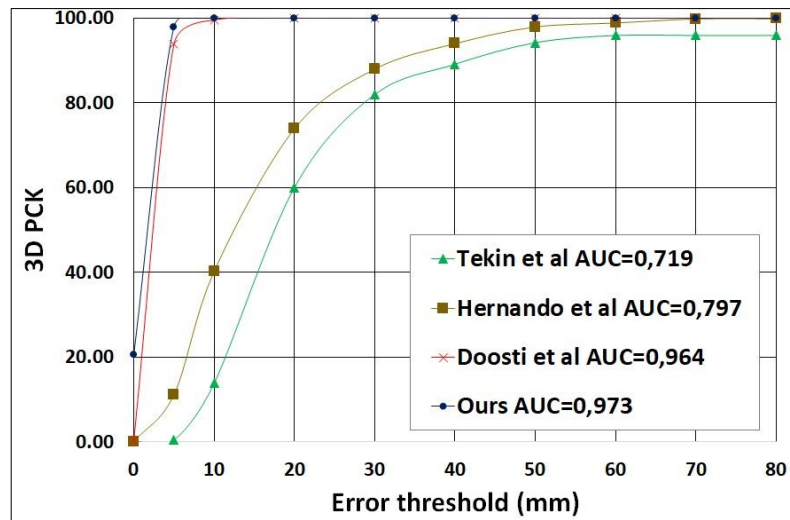


Figure 5.9: Comparison with the state-of-the-art methods on the FPFA dataset.

5.4.3. Comparison with the State-of-the-art Methods

We compare the proposed approach against recent state-of-the-art RGB-based 3D hand pose estimation methods using the aforementioned metrics and datasets. Note

that we validate our model under the same evaluation standards of [11,12] and using the same data splits.

We compare our method against [9,11,15,16] with the EPE (mm) metric. Experimental results in Table 5.4 demonstrate the superiority of the proposed method that reports the lowest error on the four datasets STB, MHP, RHD, and FPHA. More specifically, it archives improvements up to 1.06 Mean EPE in STB (14.5%), 2.89 Mean EPE in MHP (22.02%), 5.02 Mean EPE in RHD (29.33%) and 0.17 Mean EPE in FPHA (2.49%). We note that our method exhibits an impressive gain on the RHD dataset, which has more occluded fingers and complex backgrounds. We also report the quantitative results of the 2D hand pose estimation task with the EPE [pixel] metric on STB and RHD datasets. We can see from Table 5.5 that our approach reports the lowest error outperforming [11,19].

Figure 5.6 shows the 3D PCK results comparison on the STB dataset of our approach and several state-of-the-art methods [9,11,12,15–17,27,60–62]. Our method outperforms [9,11,12,15,17,60–62] and achieves competitive performances compared to [16,27].

In Figure 5.7, we present the 3D PCK results comparison of the MHP dataset. Our method performs favorably against three single-view-based state-of-the-art methods [16,41,84]. However, it exhibits inferior performance compared to [81] that uses a multi-view-based approach. More specifically, it uses images collected from cameras located at different angles to reduce the depth ambiguity.

To further evaluate our approach, we report the 3D PCK curves on the RHD dataset, which have varying backgrounds and viewpoints. Figure 5.9 shows that our method surpasses all the stated methods [9,11,15,18] with a significant margin on all the PCK thresholds. We note that STB and MHP datasets contain fewer hand poses and background variations. As such, the advantages of our approach are more apparent on the RHD dataset.

Finally, to confirm the scalability of the proposed approach, we report the 3D PCK curve on the FPHA dataset [41] that includes hands holding objects of different sizes and shapes. Figure 5.9 shows that our method performs favorably against the three reported state-of-the-art approaches [25,34,85].

5.4.4. Qualitative Results

In addition to the quantitative evaluation, we report the qualitative results on different datasets. We visually evaluate the effectiveness of our model to estimate reliable poses in all the intermediate stages, including 2D local, 2D global, and the root-relative 3D pose. Figure 5.11 shows some randomly selected test images on STB, MHP, RHD, FPHA, and GTHD datasets. The proposed method can robustly estimate the 2D global and local hand poses that confirm the advantage of multi-task learning of the two branches. Furthermore, the root-relative 3D hand poses are accurate even in complex texture backgrounds and hand poses. That is mainly due to the proposed FM loss that enhances the extracted features to learn better representations.

We also report qualitative results for the global 3D hand pose estimation task. Seen from Figure 5.11 that the proposed approach estimates reliable 3D hand poses on STB, MHP, and GTHD datasets even in self-occluded hands and diverse lighting conditions. We can explain this by that combining the cropped hand and global scene reduces depth and scale ambiguities.

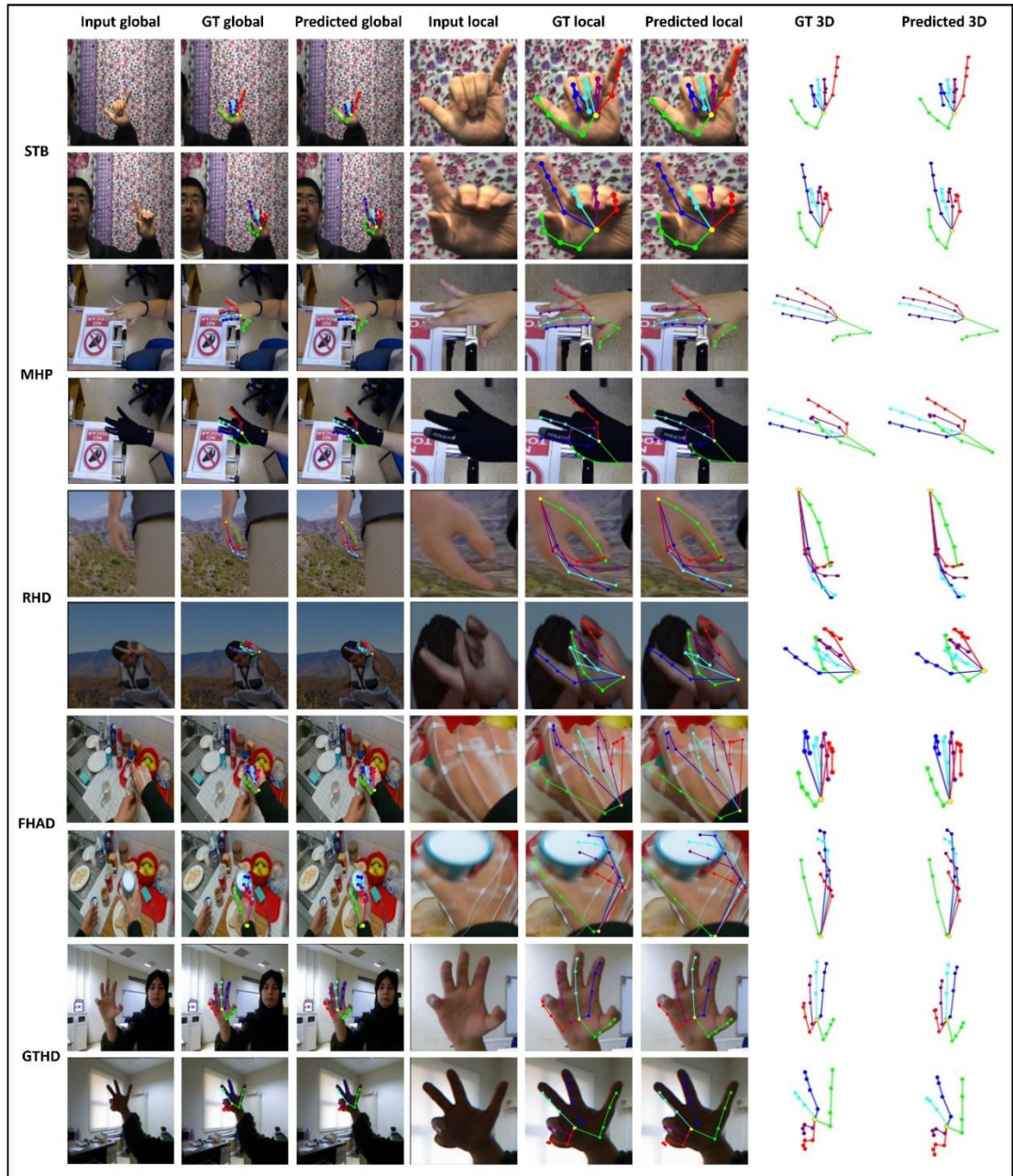


Figure 5.10: The intermediate outputs of the proposed method.

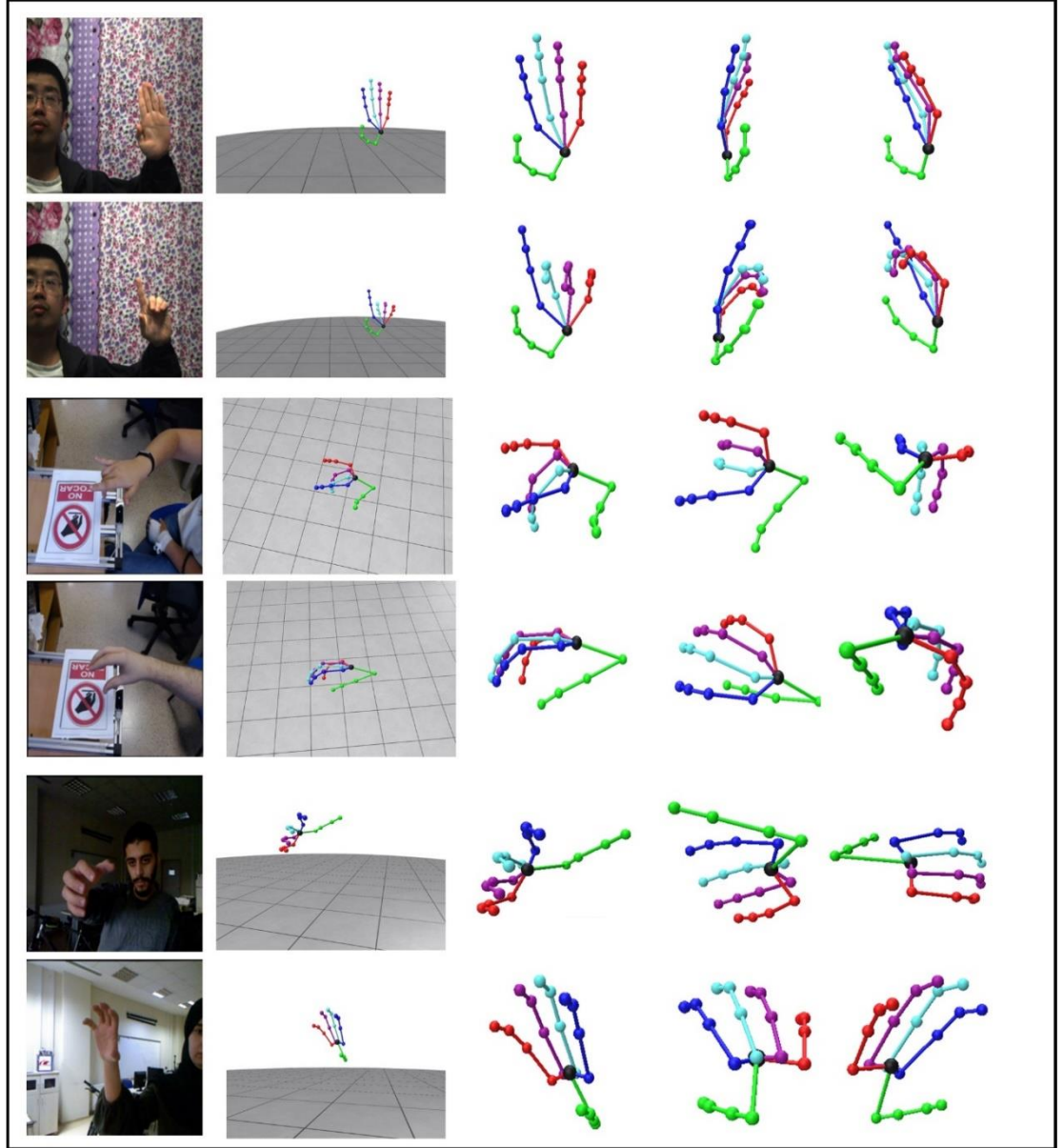


Figure 5.11: Absolute 3D hand pose estimation results on STB, MHP, and GTHD datasets.

5.4.5. Computation Complexity

The proposed approach is computationally efficient to state-of-the-art methods [25] as the running time on NVIDIA TITAN X GPU is 12ms. We can explain this by that the network consists of a lightweight feature extractor (ResNet10) and four GCN-based modules, which are significantly faster than CNNs [29]. While the 2D global pose estimator module notably contributes to improving the performance, it necessitates only an additional 1.5ms per image, which does not speed down the inference time. We train and test our model using two different backbones,

specifically, ResNet-50 and our customized ResNet-10. Table 5.6 shows the used architecture, the number of parameters, and the running time. Also, we report the EPE metric to evaluate the 3D hand pose estimation performance on the STB dataset. We can see that the model that uses our customized ResNet-10 as a feature extractor achieves competitive results in an efficient running time. In contrast, ResNet-50-based baseline significantly speeds down the computational time without noticeable improvements. Thus, we reduce the number of feature maps in each convolution layer to improve the model efficiency without affecting the performance.

Table 5.6: Running time and performance comparison between ResNet-50 and our customized ResNet-10

Backbone	Number of parameters	Running time	EPE
ResNet-50	43.6M	55ms	6.07
Customized ResNet-10	3.8M	12ms	6.21

5.4.6. Absolute 3D Hand Pose Estimation for Two Hands in the Outdoor

In addition to the mentioned indoor datasets, we show the effectiveness of our method in an outdoor environment. We use the EgoHand dataset [57] that includes synthetic images with the corresponding absolute 3D annotations of the two hands. It contains 50K and 5K samples for training and test, respectively. It also has segmentation masks and the 2D key points for the two hands from an egocentric view. Since we have left and right hands, we apply small changes in our architecture. In particular, we share the local image encoder in Figure 5.1 between the cropped left and right images. We keep the global image encoder to get original scene image features. Since the two hands have different articulations, we horizontally flip the right-cropped hand and its corresponding 2D and 3D ground truth data before feeding it to the shared image encoder to remain consistent for the model. Also, we share all the GCNs for the local 2D, the root-relative, and the absolute 3D pose estimation. Our model yields competitive performances (AUC=0.914) and estimates promising results for two hands in outdoor images (Figure 5.12). In contrast, root-relative 3D hand pose estimation methods cannot work for multi-hand scenes since the predicted pose are relative to a single point (wrist) located in one hand.

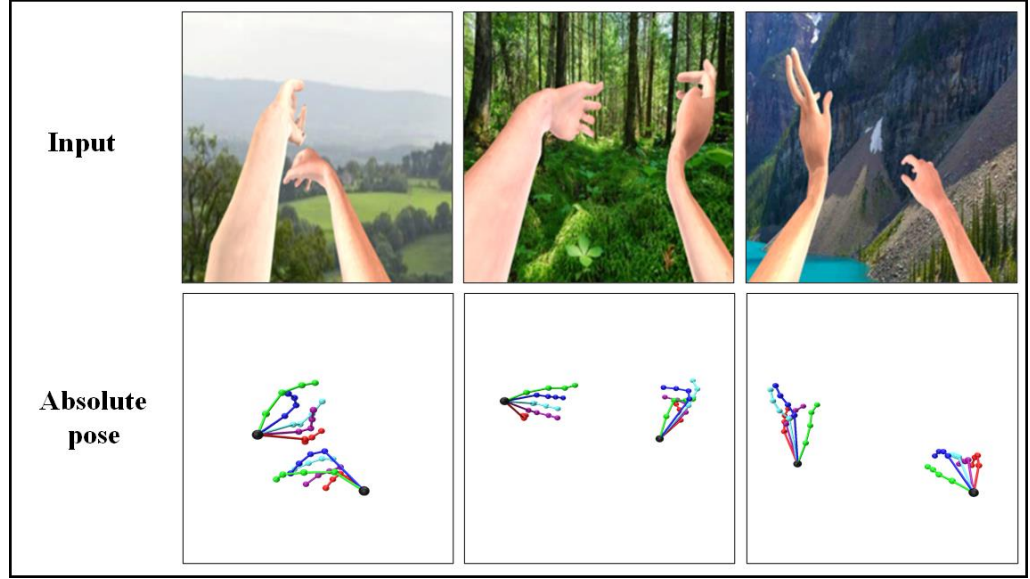


Figure 5.12: Absolute 3D hand pose estimation for two hands in outdoor setting.

5.5. Summary

Most of the existing methods mainly focus on estimating a scale-normalized root-relative 3D pose from the cropped hand image. In this work, we propose a multi-stage GCN-based (Graph Convolutional Networks) approach to estimate the absolute 3D hand pose from a single RGB image. We exploit both the cropped hand and the global scene image, which provides clues about the hand scale and location in the camera space. Our network consists of three main stages: 2D key-points, 3D root-relative, and 3D absolute pose estimation. To achieve better performance, we propose a new loss function that separates the extracted image features based on 3D joint locations to simplify the regression task. Extensive experiments on four public datasets show that our efficient model estimates accurate global 3D hand poses and performs favorably against several baselines and state-of-the-art methods. Also, we validate the proposed approach on a newly created dataset. It contains RGB hand images with accurate 3D pose annotations and high lighting and poses variations.

6. GAN-BASED HAND-OBJECT RENDERING

6.1. Introduction

Most existing hand-object rendering methods try to build an efficient 3D hand pose estimator from RGB images or depth images. After that, they render the estimated 3D hand pose with virtual object 3D pose using AR techniques. Our second part of the thesis aims to develop a single rendering module that takes the hand and the object RGB images to compose them and output the interacted hand image. This approach will allow us to interact with virtual objects using cheap RGB cameras in AR applications.

Although this study is dedicated to AR applications, it is vision-based since it employs generative models to render the hand and the object images. This task is not studied in the literature, but it is close to image composition which is one of the challenging problems of computer vision and graphics fields. It refers to overlying objects from different images to be in one single image. The latter should have a realistic appearance, overlay semantic, and correct geometry. Generally, taking the foreground object of one image and combining it with the background from another image semantic layout color using professional image editing software (E.g., Photoshop). Recently GANs have shown great potential in realistic image synthesis [28,31,32]. However, most existing works address synthesis realism in either appearance space or geometry space.

This chapter presents a new method for hand-object composition. It is divided into two main parts: Section 6.2 proposes a GANs-based image composition framework of the real hand (background object) and the real object (foreground object) images. Section 6.3 presents a GANs-based method to render the real hand image with a synthetic object. Finally, Section 6.5 presents the summary of the study.

6.2. Real hand-object rendering

6.2.1. Challenges of Rendering Module Development

We consider the task of composing two input images into a joint image that captures their realistic interactions. We learn a function that maps images of different

objects sampled from their marginal distributions (e.g., hand and pen) into a combined sample (hand holding a pen). This task includes several challenges, such as

Geometry harmony: current GAN-based image-to-image translation methods learn the style of images of the target domain, whereas geometric realism is largely ignored [21]. They assume that the target and source distributions share the geometry content. However, in our task, the input and the output have different geometry. A natural image is a 2D projection of a composition of multiple objects interacting in a 3D visual world. Modeling compositionality in 2D images is a challenging problem that requires knowledge of the complex interaction between the background and foreground objects, including:

- Relative scaling (relative camera position).
- Spatial layout.
- Occlusion.
- View-point transformations.
- Semantic understanding of realistic object layouts.

Also, the appearance harmony should be preserved. The model adjusts the color, brightness, and style of the foreground object to harmoniously embed them into the background image. Usually, this overlaying produces various artifacts due to appearance conflicts between the foreground objects and the background images. For instance, given an image of a hand and an image of a pen, the model should deal with several challenges, such as lighting, white balance, and shading differences to generate a realistic interacted hand image.

The goal of the current GANs-based approaches is to learn a mapping function that translates a given sample from the source distribution to the output distribution [29,101]. This primarily involves transforming either a single object of interest (apples to oranges, horses to zebras, label to image, etc.) or changing the style and texture of the input image (day to night, etc.). In other words, most existing GANs were designed to achieve synthesis realism either from geometry space or appearance space but not in both. Also, the input for these methods is either a sampled vector or a single image. Adapting GANs to take two input images (hand + object) may have several problems.

It maps from two different source distributions to output a single sample that should have the geometry content and the appearance of the two images.

The unsupervised learning and dataset problem is also crucial. To perform our task, we must have the triplet empty hand, object, hand-holding object. Building such a dataset is impractical and time-consuming. Hence the supervised learning of the proposed methods is unattainable. Thus, we transform the problem into unsupervised learning to solve the dataset problem. However, performing image composition without supervised learning is too challenging.

6.2.2. Single Class Object Rendering

We first tried to solve the problem using a single object class. In the literature, there are several image-to-image translation approaches. Pix-to-Pix [30] is one of the best and most successful frameworks that archives good results. However, it is fully supervised, and we should have pairs of hand/hand +object images, which is an impractical task requiring considerable human supervision and consuming time. Cycle-GAN [21] and UNIT [101] are unsupervised image-to-image translation that is based on the idea of cycle consistency. It means if we convert a zebra image to a horse image and then back to the zebra image, we should get the very same input image back. However, most of these methods only change the appearance of the input image since they assume there is a shared latent space between the target and the source distributions. We conduct our first experiment using the UNIT framework to investigate whether it can learn the additional object geometry or not. Also, we want to check if it can produce a natural joint image that has a realistic appearance of both the object and the hand. We have two domains: the first includes empty hands covering several poses. The second domain has images of hands holding one specific object and interacting with it using different poses. This dataset is fully unsupervised (we do not have the pair of hand/hand + object images). From Figure 6.1, we can see that the network starts to learn the appearance of the hand and the object, but visual artifacts still appear. Furthermore, it fails to preserve the geometry of the hand and the object, which results in an irrelevance joint image. We can explain this by the fact that the UNIT framework is a function that learns how to synthesize only the appearance of

the target domain. Besides, it does not preserve the hand pose (geometry) during mapping since the dataset does not include all the possible configurations.



Figure 6.1: Failure cases of the unsupervised image-to-image translation between an empty hand and a hand holding an object.

6.2.3. Multi-class Object Rendering

In the second experiment, we want to generalize the approach to handle multiple object classes. The UNIT framework [101] is not suitable for that task since it is Unimodal. To address that, we conduct the second experiment using the MUNIT framework [29] which is a multi-modal unsupervised image-to-image translation approach. It maps the input image to multiple outputs sharing the same content (geometry) and having different appearances. We adopt it to handle our problem by converting empty hand images to rendered hands (a hand holding different objects) and vice versa.



Figure 6.2: Failure cases of the multi-modal unsupervised image-to-image translation task between hand and rendered domains.

From the results of the previous experiment (Figure 6.2), we understand that the original MUNIT framework is not suitable for our task. We can explain this by the fact that MUNIT has a single input, which is not our case. In test time, we have two inputs: the first one is the hand image, and the second one is the object image (Figure 6.3). The output is the composition image of these two inputs. We overcome this limitation by adding a third modality (object) to the previous experiment. Thus, we change the original architecture, the loss function, and the training approach accordingly to handle the new modifications. The second reason is MUNIT is an image-to-image translation framework that ensures that the target image has a realistic appearance. However, it does not care about the geometry since it assumes that the source and target domains share the same content (geometry) and have different styles, which is not the case. Our input and output have different contents (geometry) and styles (appearances).

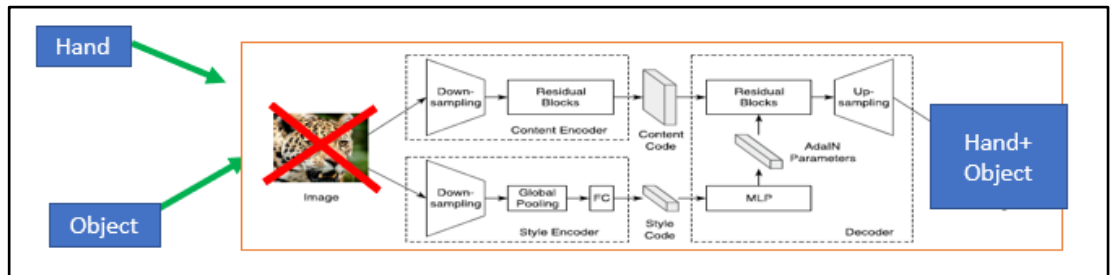


Figure 6.3: Our training and test schemas are different from MUNIT since we use two inputs. Also, our target distribution has different geometry and appearance than the source.

Training dataset for real hand-object rendering deep learning methods require many labeled data. There is a lack of datasets that can be used to train our proposed approach. It should include the triplet hand, object, and rendered (a hand holding an object). Meanwhile, building such a dataset is impractical, time-consuming and should be done with human supervision. To overcome this limitation, we created a new dataset that is fully unsupervised (multi-views, and several rendered hands. We change both the object and the hand pose to ensure diversity. Specifically, the dataset is composed of three subsets: the first one is the style category that presents the images of the objects from several viewpoints. It has 35K images for seven objects (each one has 5K images). The second subset is the rendered class that shows a hand-holding object. It has 35K images for seven objects (each one has 5k images). The last subset is the hand that has 70K empty hand images. We tried to cover all the possible poses. Also, we use a green chroma to provide the segmentation maps of the foreground objects. Figure 6.5 shows the different blocks of the proposed architecture for real hand-object rendering. We explain each block as follows:



Figure 6.4: Unsupervised real dataset for hand-object composition task. It includes three subsets: object, empty hand, and interacted hand.

Domain reconstruction training: each domain hand/rendered has a specific content/style encoder and image decoder networks. For the hand domain, we use the hand image as an input for both the content and the style encoders (6.1) and (6.2). In the rendered domain, we use the rendered image as an input to the content encoder (6.4) and the object image as an input to the style encoder (6.5). To reconstruct the hand, we feed the latent representations of the content and the style to the hand decoder

(6.3), and we use the L1 loss function to measure the error with the ground truth image (6.7). For the rendered domain reconstruction, we decode both the rendered and the object images (6.6), and we measure the L1 loss with their corresponding ground truth images (6.8) and (6.9). We note that Figure 6.6 explains the reconstruction within the domain.

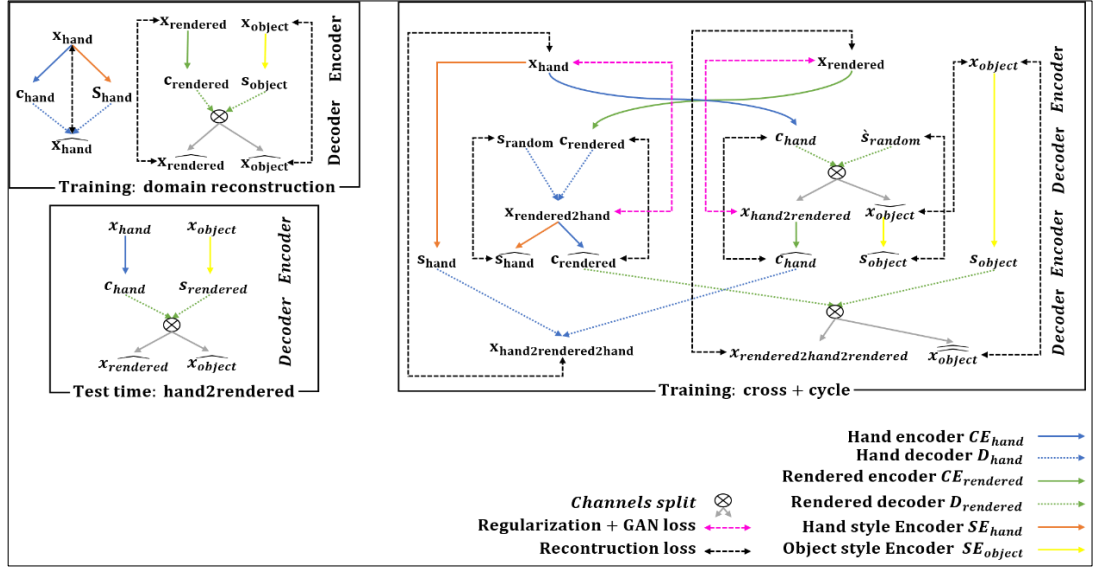


Figure 6.5: Our MUNIT-based approach for real hand-object rendering.

$$c_{\text{hand}} = \text{CE}_{\text{hand}}(x_{\text{hand}}) \quad (6.1)$$

$$s_{\text{hand}} = \text{SE}_{\text{hand}}(x_{\text{hand}}) \quad (6.2)$$

$$\widehat{x}_{\text{hand}} = D_{\text{hand}}(c_{\text{hand}}, s_{\text{hand}}) \quad (6.3)$$

$$c_{\text{rendered}} = \text{CE}_{\text{rendered}}(x_{\text{rendered}}) \quad (6.4)$$

$$s_{\text{object}} = \text{SE}_{\text{object}}(x_{\text{object}}) \quad (6.5)$$

$$x_{\text{rendered}}, \widehat{x}_{\text{object}} = D_{\text{rendered}}(c_{\text{rendered}}, s_{\text{object}}) \quad (6.6)$$

$$L_{\text{hand-rec}} = \|x_{\text{hand}} - \widehat{x}_{\text{hand}}\|_1 \quad (6.7)$$

$$L_{\text{rendered-rec}} = \|x_{\text{rendered}} - x_{\text{rendered}}\|_1 \quad (6.8)$$

$$L_{\text{object-rec}} = \|x_{\text{object}} - \widehat{x}_{\text{object}}\|_1 \quad (6.9)$$

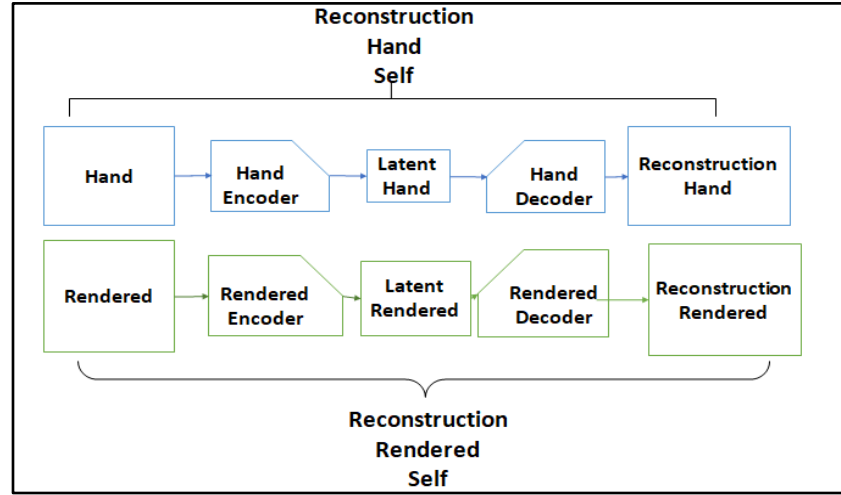


Figure 6.6: Reconstruction loss within the same domain.

Cross-domain training translates from the hand domain to the rendered domain, we feed a sampled latent representation from the Gaussian distribution and the latent representation of the hand to the rendered decoder to predict both the rendered and the object images (6.10). After that, we encode the resulting rendered/object images using the rendered image encoder and the object style encoder. We measure the error

between the Gaussian vector and the encoded style. Also, we measure the error between the encoded rendered image and the hand latent representation. We use the adversarial loss to ensure that the distribution of the fake rendered images is close to the distribution of real rendered images (6.12). We follow the same approach to translate from the rendered domain to the hand domain. We feed a sampled latent representation from the Gaussian distribution and the latent representation of the rendered to the hand decoder to predict both the rendered and the object images (6.11). We use the adversarial loss to ensure that the distribution of the fake hand images is close to the distribution of real hand images (6.13).

$$x_{\text{hand2rendered}}, \widetilde{x_{\text{object}}} = D_{\text{rendered}}(c_{\text{hand}}, \hat{s}_{\text{random}}) \quad (6.10)$$

$$x_{\text{rendered2hand}} = D_{\text{hand}}(c_{\text{rendered}}, s_{\text{random}}) \quad (6.11)$$

$$L_{\text{rendered-adv}} = \text{LSGAN}(x_{\text{rendered}}, x_{\text{hand2rendered}}) \quad (6.12)$$

$$L_{\text{hand-adv}} = \text{LSGAN}(x_{\text{hand}}, x_{\text{rendered2hand}}) \quad (6.13)$$

Cycle domain training: the cycle consistency loss ensures the translation back to the source domain (6.16) and (6.17). In particular, if we translate from the hand to the rendered domain, and after that, we translate the result back to the hand domain (6.14), we should get the original hand image Figure 6.7. We feed the encoded object style and the encoded cross-domain latent representation from the rendered domain to decode back the hand. We use the same approach to reconstruct back the rendered image from the encoded hand style and the cross-domain latent representation of the hand (6.15).

$$x_{\text{hand}2\text{rendered}2\text{hand}} = D_{\text{hand}}(\widehat{c}_{\text{hand}}, s_{\text{hand}}) \quad (6.14)$$

$$x_{\text{rendered}2\text{hand}2\text{rendered}}, \widehat{\widehat{x_{\text{object}}}} = D_{\text{rendered}}(c_{\text{rendered}}, s_{\text{object}}) \quad (6.15)$$

$$L_{\text{hand-cycle-rec}} = \|x_{\text{hand}} - x_{\text{hand}2\text{rendered}2\text{hand}}\|_1 \quad (6.16)$$

$$L_{\text{rendered-cycle-rec}} = \|x_{\text{rendered}} - x_{\text{rendered}2\text{hand}2\text{rendered}}\|_1 \quad (6.17)$$

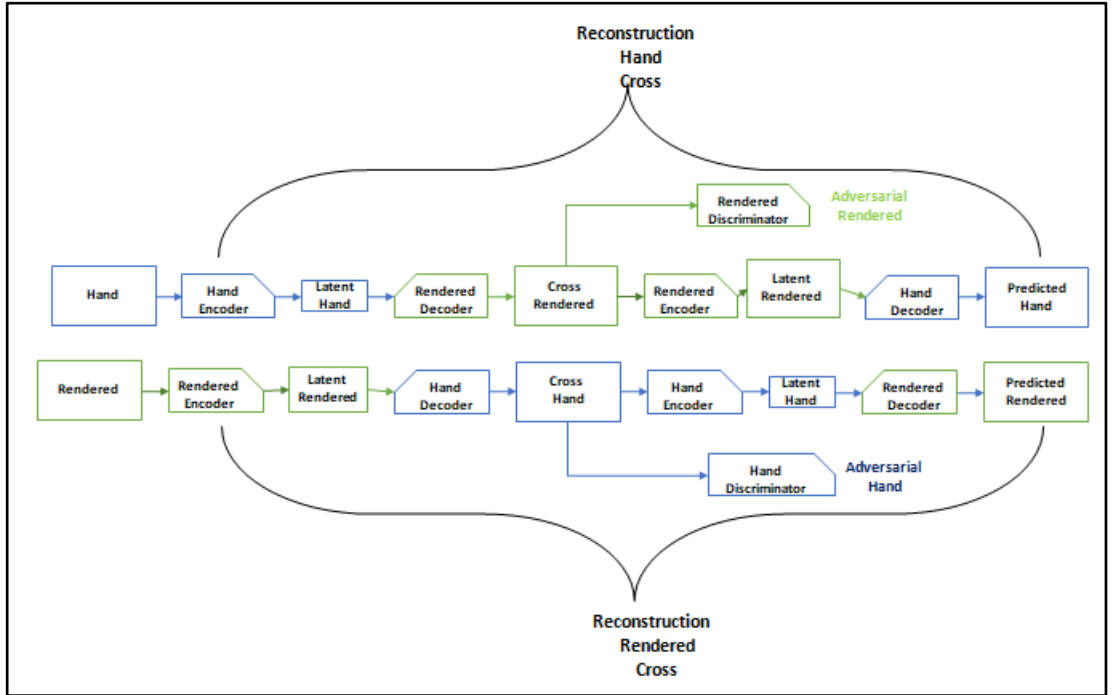


Figure 6.7: Cycle consistency loss for rendered and hand domains.

The last problem is to find a solution to preserve the hand pose during the mapping. Since the proposed method is fully unsupervised and may not include all the possible configurations, our model can output a rendered hand image pose that does not match the input hand pose. To address this limitation, in Figure 6.8 we add a regularization loss that computes the L1 loss between the two images to enforce the same pose (6.18) and (6.19). Meanwhile, the adversarial loss imposes the target image

to be as similar as possible to the images of the source domain. The optimization process finds the best parameters to satisfy the two constraints.

$$L_{\text{rendered-reg}} = \|x_{\text{hand}} - x_{\text{hand2rendered}}\|_1 \quad (6.18)$$

$$L_{\text{hand-reg}} = \|x_{\text{rendered}} - x_{\text{rendered2hand}}\|_1 \quad (6.19)$$

Test time: to get the rendered image, we feed the hand and the object to the hand/style encoders. Their latent representations are given to the rendered decoder to get the rendered image.

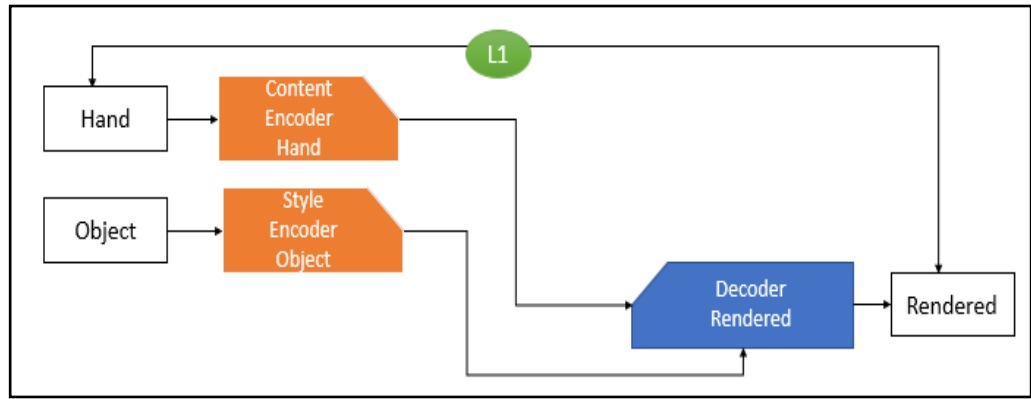


Figure 6.8: We optimize L1 loss between our input hand image and the model output to preserve the hand pose.

We can see from Figure 6.9 that our proposed regularization loss can relatively preserve the hand pose between the input and output images. Besides, our approach generates plausible images and can naturally render the hand image with the given object image. Also, it successfully removes the object and converts a rendered hand to an empty one (Figure 6.10). However, it still suffers from several problems, such as:

- Dynamic objects such as scissors and glasses are very hard to be predicted since they have many degrees of freedom.
- Sometimes it still does not preserve the pose of the hand (usually, it appears in hand-to-rendered translation).
- It still does not perform well in test time (especially when we use new objects that do not exist in the dataset).

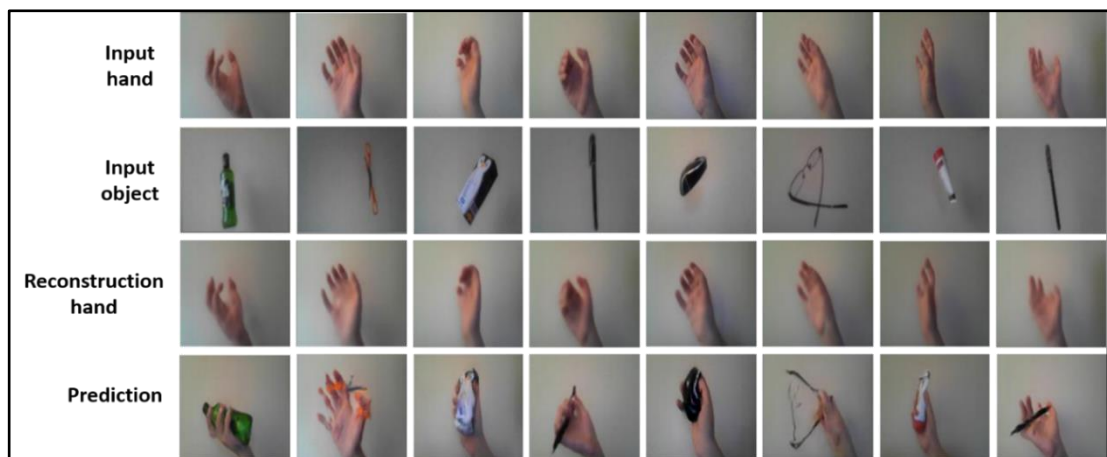


Figure 6.9: Results of the hand-to-rendered translation using our proposed MUNIT-based architecture.

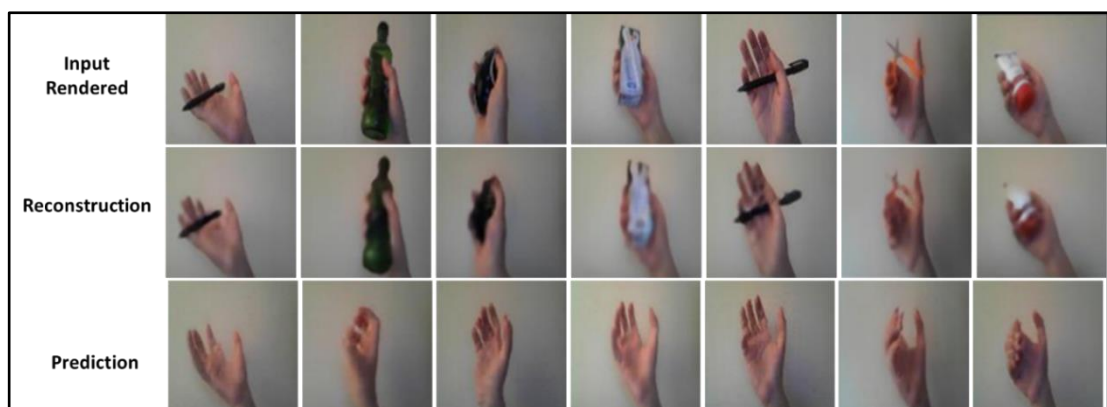


Figure 6.10: Results of the rendered-to-hand translation using our proposed MUNIT-based architecture.

6.2.4. Composer-Decomposer architecture for hand-object rendering

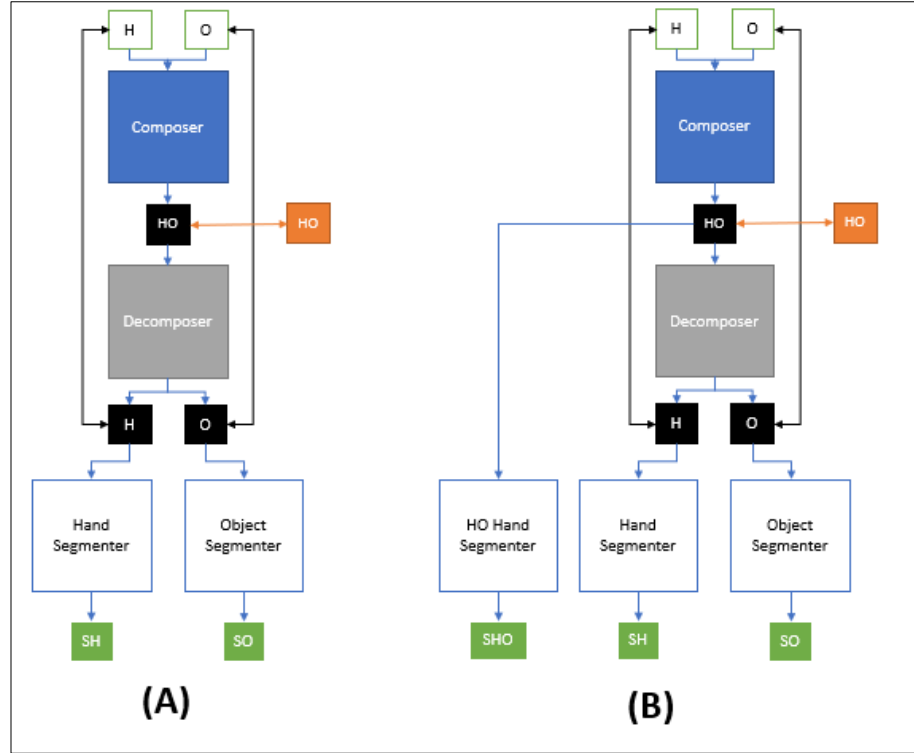


Figure 6.11: Composition-decomposition method for real hand-object rendering.

We create a new dataset that includes the triplet: hand, object, and rendered images with the corresponding segmentation maps. We used a green Chroma to segment the foreground object from the background. Meanwhile, we propose a new architecture that takes the object and the hand images to compose them using autoencoders. The rendered image is given to a decomposition module that predicts the hand and the object images to compare them against the ground truth. Besides, we predict the segmentation maps of the estimated hand and object to ensure that they have the same geometry of ground truths (Figure 6.11). Results in Figure 6.12 show that this approach generates realistic rendered images with appearance harmony. However, the hand pose has not persevered since the learning is unsupervised and the dataset does not include all the possible configurations of the triplet hand, object, and rendered images.

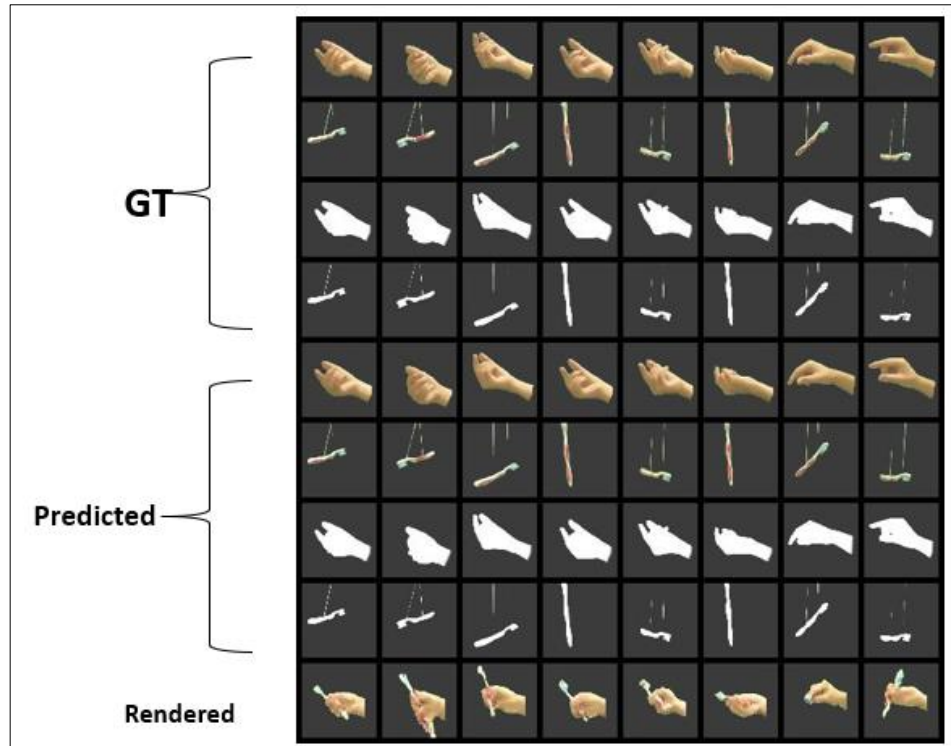


Figure 6.12: Results of real hand-object rendering using composition-decomposition architecture.

6.3. Synthetic Object- hand Rendering

Since hand-object rendering for augmented reality applications is difficult, we start solving the problem using virtual objects and synthetic hands. This is also a challenging problem due to several reasons, including:

- Most of the image composition approaches are fully supervised. They have the triplet foreground/background /rendered (they use synthetic data).
- Their output is more constrained, for example (glass should be on the top of eyes). In contrast, the hand can hold the same object in different ways.
- The background is fixed, it does not have many degrees of freedom like the hand.
- Variant types and shapes of the hands and the objects.

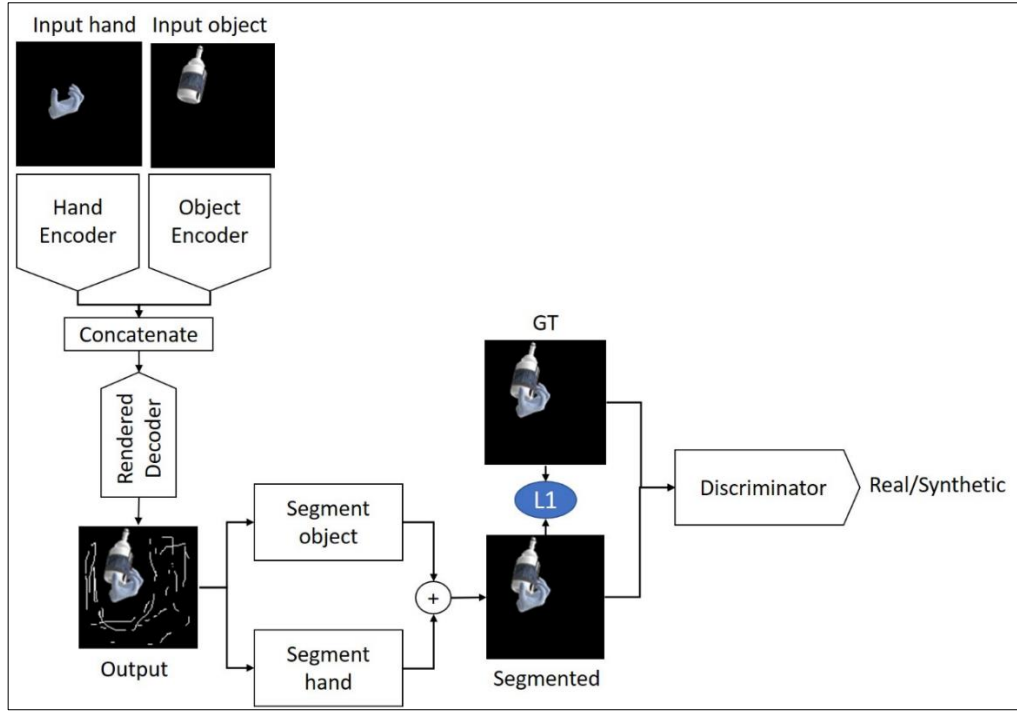


Figure 6.13: The proposed GANs-based architecture for synthetic hand-object rendering.

We conduct our experiments using the large-scale synthetic image ObMan dataset [102]. It includes 150K of hands grasping objects images generated by an automatic robotic grasping software. Also, it provides the corresponding depth map, the 2D/3D hand keypoints, the object mesh, and the segmentation maps of the hand and the object. It has a large variation in pose, background, texture, and lighting. The proposed architecture in Figure 6.13 takes as input the RGB images of the hand and the object and gives them to the corresponding encoders. We concatenate the latent representations to generate the rendered image. We use the discriminator to criticize the realism of the rendered hand image.

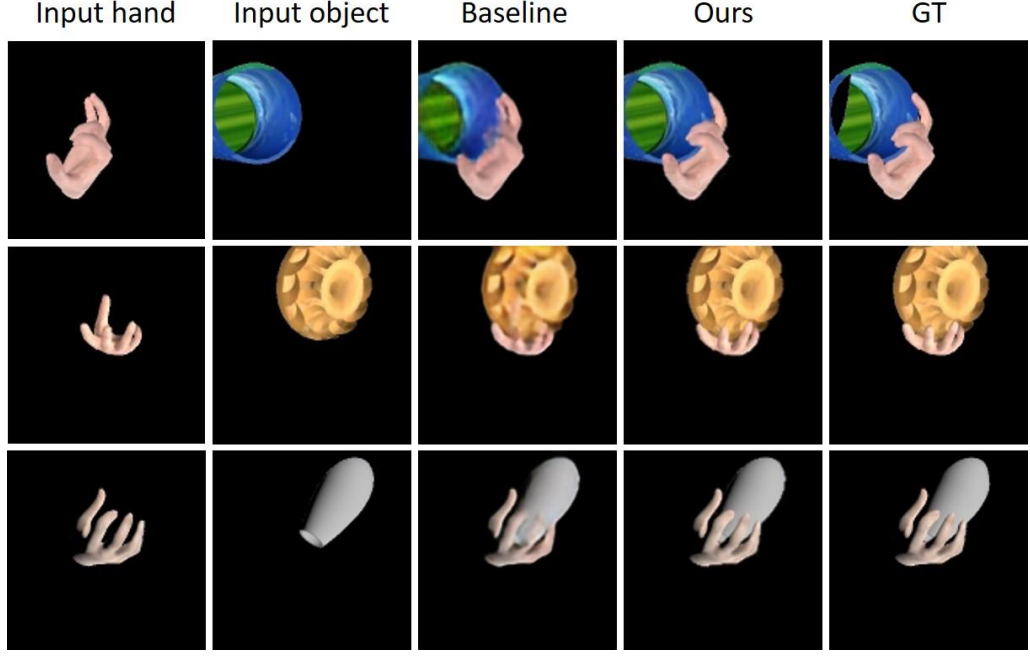


Figure 6.14: Results of the proposed GANs-based synthetic hand-object rendering approach on the ObMan dataset.

Table 6.1: Quantitative comparison between the GANs-based baseline and our approach for hand-object rendering.

Approach	MAE^-	$SSIM^+$	$PSNR^+$
GAN-based baseline	0.0094	0.959	26.67
Ours	0.0048	0.979	28.83

The proposed approach for synthetic hand-object rendering obtains plausible results on the ObMan dataset. However, it still presents some failure cases. In particular, the model does not adjust the appearance and geometry of the foreground object into the hand image. It mixes between them and does not ensure pixel continuity of the object. We address this issue using the following modifications:

- Performs data augmentation, such as vertical flip, horizontal flip, random cropping, resizing and shift scale rotate. This generalizes the model and improves the results.

- Try different architectures for the generator and the discriminator and augment the number of the model parameters.
- Use a loss function that enforces the segmentation maps of the object and the hand of the interacted image to be similar to the ground truth ones. Specifically, we give the output rendered image to hand Segmenter and object Segmenter to recover a non-noisy image.

Qualitative and quantitative results in Figure 6.14 and Table 6.1 show the conducted edits notably improve our GAN-based model and achieve plausible results.

6.4. Synthetic Object- real Hand Rendering

6.4.1. Overview

The architecture proposed in Figure 6.13 takes a synthetic hand image as input. However, the input in AR applications the input should be a real hand image. Therefore, we propose a GAN-based framework that performs real hand-synthetic object rendering into three main stages:

- Convert the input real hand image to a synthetic one using unsupervised image-to-image translation frameworks. This task is challenging since the pose of the hand should be preserved.
- Apply the architecture proposed in Figure 6.13 that renders a synthetic hand and virtual object in an end-to-end manner.
- Segment the hand from the generated rendered image and translate it back to the real domain.

6.4.2. Hand Segmentation

Before performing the translation between synthetic and real hand images, we should segment the hands. We use our proposed UNet-based hand Segmenter of the first study. We can see from Figure 6.15 and Table 6.2 that our model obtains good qualitative and quantitative results. To segment the real hand from the background, we use a green chroma and adobe premiere to find the mask of the ground truth.

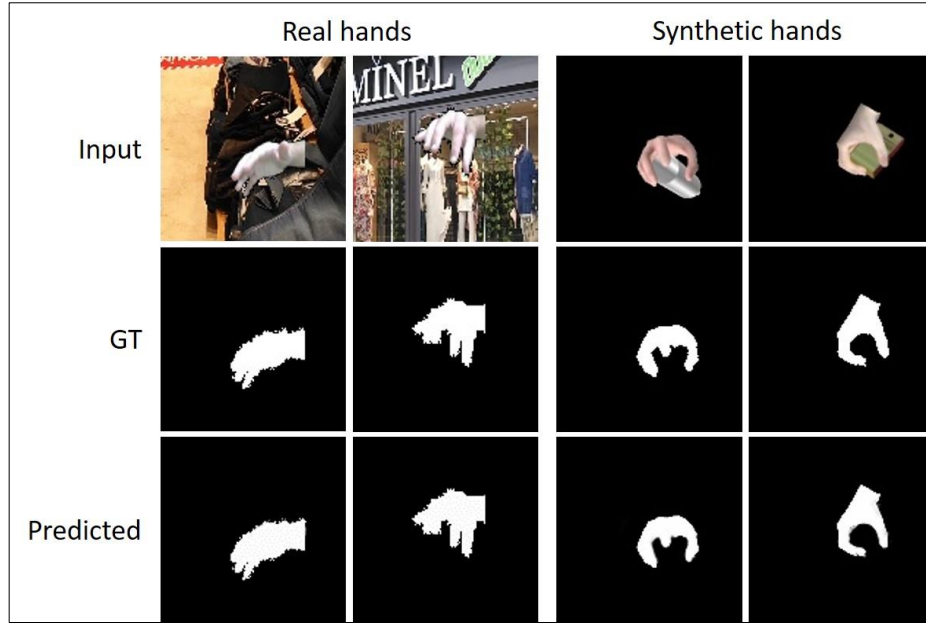


Figure 6.15: Hand segmentation results using synthetic and real hand images.

Table 6.2: Quantitative results for hand segmentation task using synthetic and real images.

	Approach	Accuracy	Precision	Recall	F1
Real hands	Class hand	0.997	0.998	0.996	0.998
	Class background	0.998	0.999	0.998	0.999
Synthetic hands	Class hand	0.998	0.997	0.995	0.998
	Class background	0.999	0.998	0.997	0.999

6.4.3. Unsupervised Image-to-image Translation between Synthetic and Real Hands

In our first experiment, to convert the real image to a synthetic one, we use the Cycle GAN framework [42], which is one of the famous unsupervised image-to-image translation methods. However, this experiment does not provide a plausible generated synthetic image (Figure 6.16). They did not preserve the shape and the pose of the hand in the two translations (real to synthetic and synthetic to real translations). In the second experiment, we added our proposed loss function to ensure hand pose preservation. It enforces the hand mask and the magnitude image of the input image

to be close to the translated one using L2 loss. We can see from Figure 6.17 that this experiment obtains more realistic images and accurate poses.

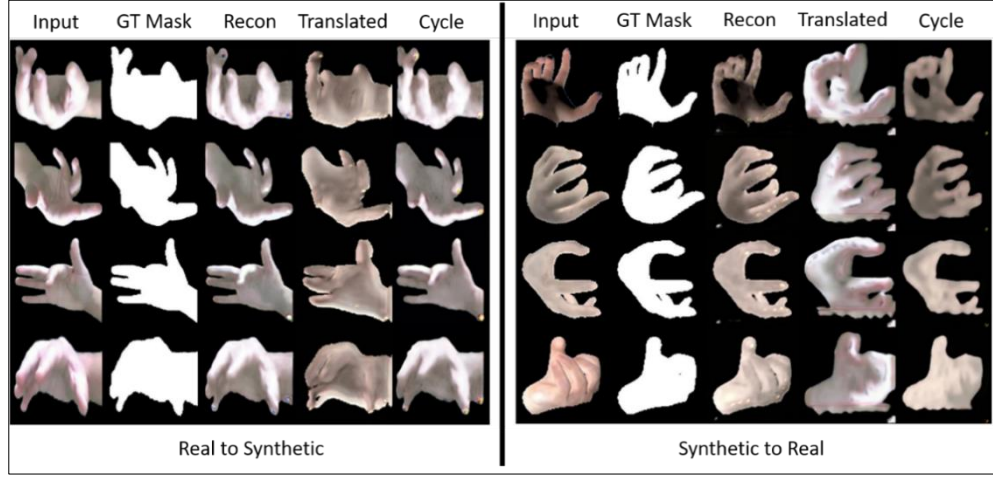


Figure 6.16: Image-to-image translation between real and synthetic hand images using Cycle-GANs.

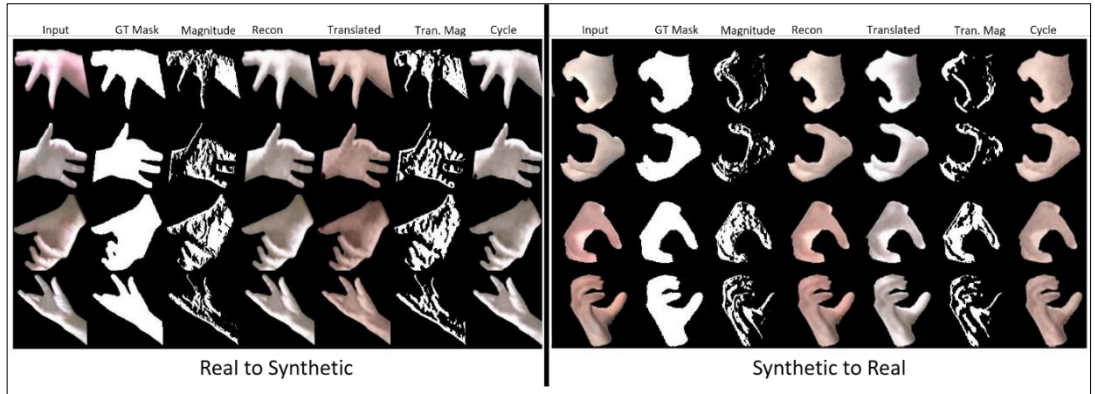


Figure 6.17: Image-to-image translation between real and synthetic using Cycle-GANs and the proposed loss functions.

6.4.4. The Overall Architecture for Hand-object Rendering in AR Applications

The last step of our proposed approach is to combine the hand segmentation, the unsupervised image-to-image translation between the real and synthetic hand images, and the hand-object rendering tasks (Figure 6.18). Specifically, we give the input RGB image to the real hand Segmenter. We multiply the predicted mask with the original image to get the hand image that will be cropped and resized. We convert the real hand image to a synthetic one using the proposed CycleGANs-based architecture. Next, we place the translated hand in its original position in the input image. We give the

synthetic object image and the hand image to our GANs-based hand-object rendering framework. We give the output rendered hand image to another Segmenter to get the mask and replace the hand synthetic pixels with real ones. Finally, we merge the rendered hand image with the original background. Results in Figure 6.19 show that the generated images are visually appealing in all the intermediate stages. We preserve the hand pose, and the rendered hand image has appearance and geometry harmony.

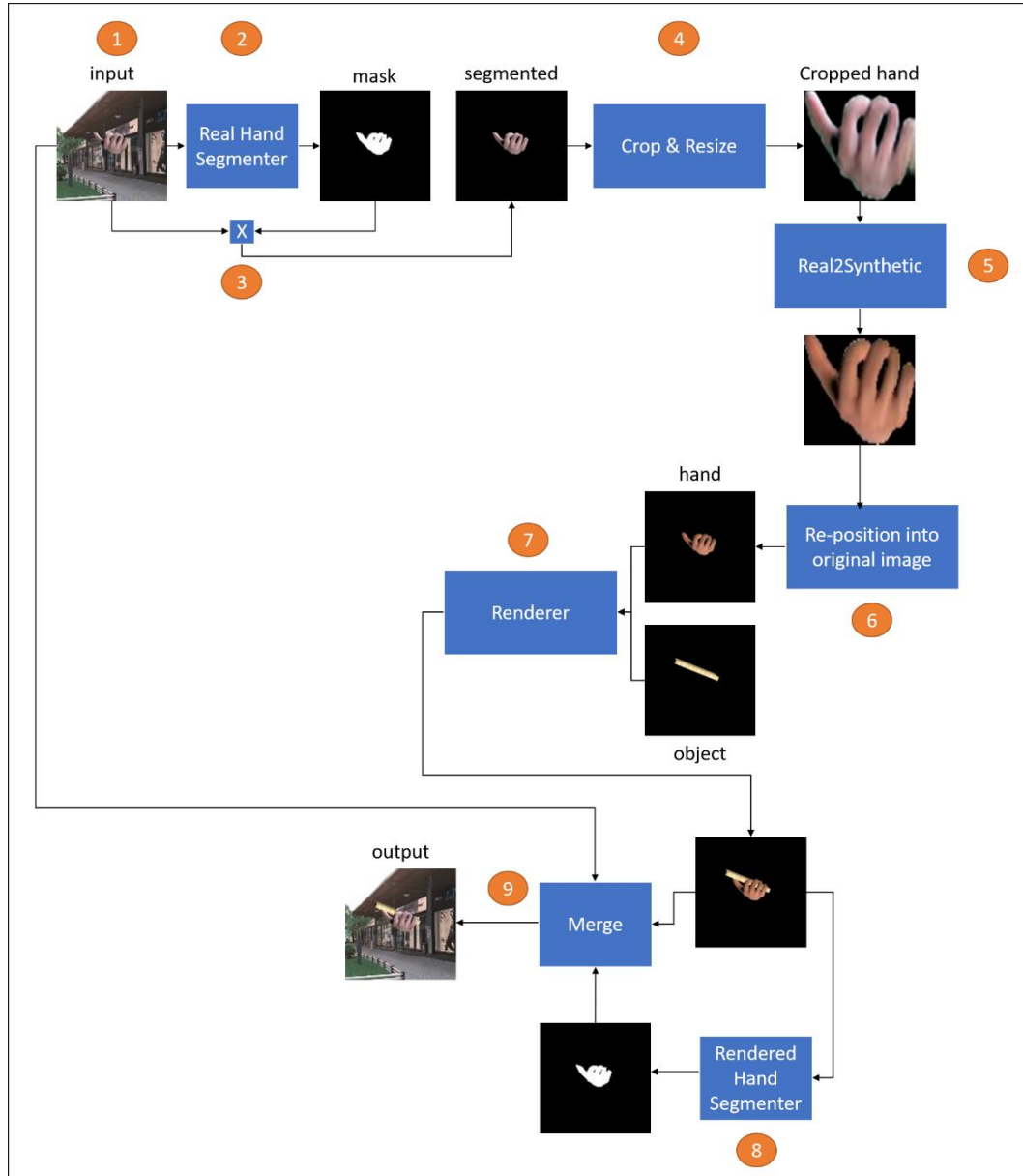


Figure 6.18: Hand-object rendering approach for AR application.

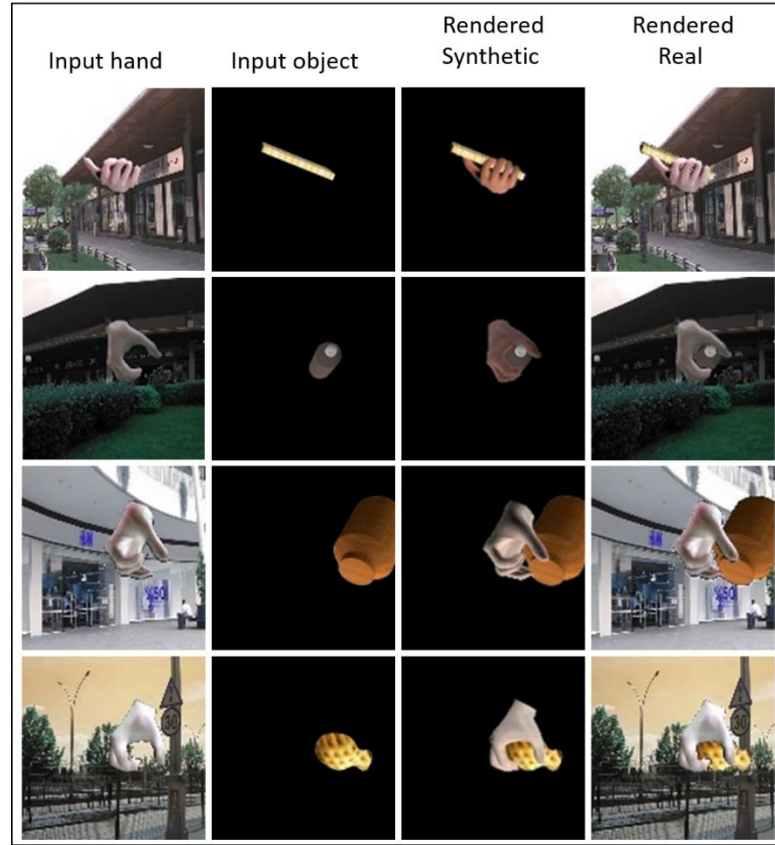


Figure 6.19: Hand-object rendering results for AR application.

6.5. Summary

In this chapter, we present a new generative-based approach for hand-object rendering. Since this task is too challenging, we progressively solve the problem. In the first experiment, we propose a learning-based architecture for real hand-object rendering. It adapts the MUNIT framework to take two input images and generate a composed image with appearance and geometry harmony. The second study proposes a GANs-based architecture for synthetic hand-object rendering. It ensures that the hand naturally grasps the object. In other words, the model should distinguish between the pixels of the hand and the object. Finally, our last experiment applies an unsupervised image-to-image translation between real and synthetic hand images. After that, it composes them and outputs the rendered hand image. Qualitative and quantitative results show that our approach achieves promising results and generates visually appealing rendered hand images.

7. COMPARISON OF THE PROPOSED 3D HAND POSE ESTIMATION APPROACHES

7.1. Introduction

In this chapter, we conduct extensive experiments to compare our two proposed GCNs-based 3D hand pose estimation methods. We report qualitative and qualitative results on the MHP dataset. Additionally, this chapter presents the most effective monocular RGB-based 3D hand pose estimation method. It combines our hybrid classification-regression approach (HCRA) with the graph-based absolute 3D hand pose estimation (GAAH).

7.2. Root-relative 3D Hand Pose Estimation Comparison

In the first experiment, we report comparisons of our monocular RGB-based 3D hand pose estimation proposed methods for the root-relative coordinates. We use the same evaluation standards and data split. To ensure a fair comparison, we aligned the root key points (wrist) of the ground-truth and estimated poses before calculating the metrics of all the methods. Table 7.1 shows that the HCRA technique outperforms the GAAH in the two datasets. It provides lower scores in both AUC and EPE metrics proving the superiority of the proposed per-pose adjacency matrix over the global shared one. In particular, the classification branch finds spatial dependencies between neighboring joints to guide the regression branch. Although the global learnable adjacency matrix changes from one dataset to another, it remains identical for the images which belong to the same dataset, which fails to capture pose dependent constraints.

Table 7.1: Quantitative comparison between the proposed monocular RGB-based 3D hand pose estimation methods for the root-relative.

Approach	EPR_{mean}^-	AUC_{0-50}^+	AUC_{20-50}^+
HCRA	9.58	0.847	0.994
GAAH	11.78	0.791	0.986

7.3. Absolute 3D Hand Pose Estimation Comparison

In the second experiment, we report comparisons of our monocular RGB-based 3D hand pose estimation proposed methods for the global coordinates (camera-relative). We show qualitative and quantitative results for the MHP dataset. Since the MHP datasets provide 3D pose annotations for complete hand skeleton, we report the performance of the proposed HCRA approach by adding the ground-truth depth of the root joint and the global scale of the hand. Table 7.2 shows that the GAAH technique outperforms the HCRA. It provides lower scores in both AUC and EPE metrics. We can explain this by that feeding the scale of the hand in its original scene gives a clue about its position in the camera coordinate system. Besides, other objects in the scene provide clues about the distance to the camera.

Table 7.2: Quantitative comparison between the proposed monocular RGB-based 3D hand pose estimation methods for the root-relative.

Approach	EPR_{mean}^-	AUC_{0-50}^+	AUC_{20-50}^+
HCRA	11.13	0.758	0.945
GAAH	10.23	0.771	0.954
Combined	9.12	0.789	0.967

7.4. Combination of HCRA and GAAH

The last experiment combines the HCRA and the GAAH approaches on the MHP dataset. Specifically, we modify our proposed approach for monocular RGB-based absolute 3D hand pose estimation. The first stage estimates the global 2D pose from the original scene image using the extracted image features and the global learned adjacency matrix. For the local 2D key points estimation, we give the extracted image features to a GCN-based classifier that outputs the classification probabilities in 2D space. The Relation function constructs the per-pose adjacency matrix of the 2D joints based on their spatial alignment such that neighboring joints belong to the same class. This information will be used to guide the 2D local GCN-based regressors to estimate reliable 2D poses.

The second stage estimates the root-relative 3D hand pose using the extracted image features and the local 2D hand pose. Meanwhile, the root-relative 3D regressor also benefits from a per-pose adjacency matrix. It is constructed using another GCN-based classifier that assigns neighboring joints in the 3D space to the same class. Finally, we estimate the absolute 3D hand pose using the previous stage outputs. Namely, the extracted image features of the global and local images, the 2D global key points, the 2D local key points, and the root-relative 3D hand pose.

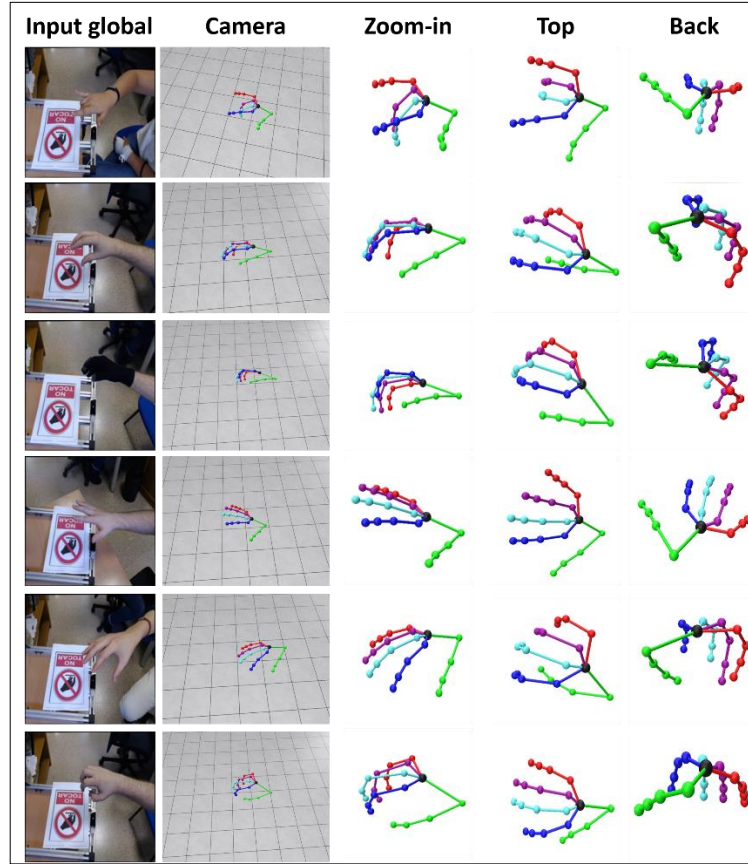


Figure 7.1: Absolute 3D hand pose estimation results on the MHP dataset.

We can see from Figure 7.1 that our new composed approach yields more favorable results compared to all the individual methods. Also, the quantitative results in Table 7.2 show remarkable improvement over HCRA and HCRA models. We can explain this by the fact that the HCRA approach improves the local 2D key points estimation. Also, the root-relative 3D pose estimation becomes more accurate. This is mainly due to the per-pose adjacency matrix that guides the regressor better than the global adjacency matrix. Note that we keep the global scene image features and the global 2D key points to improve the absolute 3D hand pose estimation.

To compare the efficiency of our proposed 3D hand pose estimation methods, we report the number of model parameters and the runtime on CPU and GPU of our models on an Intel(R) Core (TM) i7-2600K CPU @ 3.40GHz and an NVidia Titan XP GPU. Table 7.3 shows that the HCRA model yields the lowest inference time. The reason is that most of the architecture is based on GCNs, which are significantly faster than a convolutional neural network. In contrast, GAAH consists of two ResNet-based feature extractors for the global scene and the cropped hand images. The combination of the two approaches yields the most effective model that achieves the lowest estimation errors. However, the model has a large number of parameters and reports the highest running time.

Table 7.3: Running time comparison of the proposed monocular RGB-based 3D hand pose estimation methods on NVidia Titan XP GPU.

Method	Number of parameters	Running time
HCRA	2.9M	7ms
GAAH	3.8M	12ms
Combined	4.6M	16ms

7.5. Summary

This chapter presents extensive experiments to compare the proposed monocular RGB-based techniques. Qualitative and quantitative results show that the combination of the two methods yields the most effective model. It estimates accurate 3D hand poses even in occluded hands, complex backgrounds, and diverse lighting conditions. Meanwhile, it reports the lowest EPE errors and the highest AUC in the quantitative comparison. The HCRA is computationally efficient than GAAH and exhibits better performance for the root-relative 3D hand pose. Also, it improves the performance of the GAAH in the absolute 3D hand pose estimation method. Note that the global scene features of the GAAH approach proved to be more crucial for the absolute pose. Our models are more computationally efficient than most of the state-of-the-art approaches.

8. CONCLUSIONS

This chapter concludes the thesis and summarizes our main contributions by recalling the principal solved problems, the proposed approaches, and the obtained results. Besides, it discusses the limitations of the proposed methods and suggests the possibilities for future research.

8.1. Summary of the Contributions

In this thesis, we present 2D and 3D hand pose estimation approaches that solve the following problem:

- Build a large-scale hand dataset that is suitable for hand detection, 2D and 3D pose estimation tasks with high lighting and poses variations.
- Accurate hand detection, bounding box localization, and 2D hand pose estimation approach for different hand sizes.
- Enforcing joint relationships constraints about the input hand image to simplify the 3D regression task.
- Estimate accurate absolute 3D hand pose from a single synthetic or real RGB image even in cluttered images and complex poses.
- Speed up the inference time.
- Simplify the regression task using a new loss function that enforces the extracted image feature to be well separated.

Also, the second part of the thesis presents a new approach to solve the hand-object rendering problem. It performs several steps, such as:

- Robust unsupervised image-to-image translation between the synthetic and real hands.
- Ensure that the geometry and the appearance harmonies of the generated interacted hand image are realistic.
- Adapt GANs for hand-object image composition and propose new regularization techniques.

To achieve the mentioned goals, we propose several approaches such that each one focuses on addressing a specific problem.

In the first study, we propose a new learning-based method for 2D hand pose estimation. It performs multi-scale heatmaps regression and uses the hand skeleton as additional information to constrain the regression problem. It provides better results compared with the direct regression and single-scale heatmaps regression. Also, we present a new method for hand bounding box localization that first estimates the hand skeleton and then extracts the bounding box. This approach provides accurate results since it learns more information from the skeleton. This study also introduces a new large-scale dataset that can use both for hand detection and 2D pose estimation tasks.

The second study proposes an efficient GCN-based approach for 2D/3D hand pose estimation from a monocular RGB image. Our method addresses one of the challenges of GCN by learning per-pose adjacency matrix expressing local and global relationships between the hand joints. We obtain this information using a classification step that exploits the spatial dependencies between the joints. We improve the predicted pose through a GCN-based refinement module that employs the proposed ANN algorithm to determine the number of adjacent joints. Experimental results on realistic and synthetic datasets prove the competitiveness of our efficient approach compared to several state-of-the-art methods.

The third study proposes a multi-stage GCN-based method that estimates the global 3D hand pose from a single RGB image. We show the benefit of combining the cropped hand image and original scene image in inferring accurate 3D root-relative pose, and absolute pose, respectively. Also, we present a new loss function that separates the extracted image features based on the 3D joint location to facilitate the regression task. The proposed method demonstrated state-of-the-art results on challenging datasets with highly occluded fingers, complex backgrounds, and object interactions. Besides, it shows competitive performances for two hands in the outdoor. Also, we show that the FM loss has an impressive benefit on the performances. The running time of our model is still efficient compared with state-of-the-art methods.

The last study proposes a new generative-based approach for hand-object rendering. It translates the real hand image to a synthetic one. Next, it composes it with the object image and translates it back to the real domain. Experiments show that our model generates plausible interacted hand images and achieves competitive performances to different GANs-based baselines.

8.2. Limitation and Future Work

The results obtained during this thesis are globally promising and encouraging. However, this section discusses the limitation of the designed approaches and gives a clue about the possible solutions to overcome them. Also, we suggest directions for future research that are not explored in our study.

In this thesis, we have shown that our proposed 3D hand pose estimation methods give very good results for hands in isolation. However, when two hands are closely interacting, or a hand is grasping an object tightly, the accuracy of our methods may decrease. Although we have shown accurate results for joint hand-object pose estimation, the benchmark dataset used for the evaluation did not contain very tight interactions. Also, there exists a large variability of hand shapes and sizes in practice and the test datasets do not cover all the possible combinations. Hence, the accuracy may deteriorate for unknown users, who have different hand shapes and sizes.

As a future work for 3D hand pose estimation, we propose to use temporal data for our absolute 3D hand pose estimation approach. The model recalls information about the hand motion from previous frame estimates, which are stored in a memory bank. Also, we suggest employing a weighted loss function that will give more importance to occluded joints, which are difficult to estimate. This approach requires human annotation for occluded joints.

Also, in our future work, we propose to use the attention mechanism in 3D hand pose estimation. Existing methods ignore the hand texture features and the underlying spatial information in the RGB image. To address this issue, we propose a channel fusion attention mechanism that combines 2D keypoint features and RGB image features at the channel level. In other words, we augment convolution with self-attention to capture long-range dependencies in an RGB image and improve the estimation accuracy.

The main limitation of the proposed GAN-based hand-object rendering approach happens in test time. Specifically, the hand of the user should be aligned with the virtual object. In other words, our approach can only be applied when we know the hand and the object position a priori. Also, due to GANs stability issues and hardware constraints, we train and test our model on a $128 * 128$ resolution. Training GANs on higher resolutions may cause convergence issues and incorporate additional

challenges. To address this problem, a more powerful architecture and loss functions can be used to improve the image quality.

REFERENCES

- [1] El-Sawah A., Georganas N. D., Petriu E. M., (2008), “A prototype for 3-D hand tracking and posture estimation”, *IEEE Transactions on Instrumentation and Measurement*, 57 (8), 1627–1636.
- [2] He K., Zhang X., Ren S., Sun J., (2016), “Deep residual learning for image recognition”, *International Conference on Computer Vision and Pattern Recognition*, 770-778, Las Vegas, NV, USA, 26 June-1 July.
- [3] Shelhamer E., Long J., Darrell T., (2017), “Fully convolutional networks for semantic segmentation”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39 (4), 640–651.
- [4] Ren S., He K., Girshick R., Sun J., (2017), “Faster R-CNN: Towards real-time object detection with region proposal networks”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39 (6), 1137–1149.
- [5] Krizhevsky A., Sutskever I., Hinton G. E., (2017), “ImageNet classification with deep convolutional neural networks”, *Communications of the ACM*, 60 (6), 84–90.
- [6] Tompson J., Stein M., Lecun Y., Perlin K., (2014), “Real-time continuous pose recovery of human hands using convolutional networks”, *ACM Transactions on Graphics*, 33 (5), 1-10.
- [7] Spurr A., Song J., Park S., Hilliges O., (2018), “Cross-modal deep variational hand pose estimation”, *International Conference on Computer Vision and Pattern Recognition*, 89–98, Salt Lake City, UT, USA, 18-22 June.
- [8] Wan C., Probst T., Gool L. van, Yao A., (2018), “Dense 3D regression for hand pose estimation”, *International Conference on Computer Vision and Pattern Recognition*, 5147–5156, Salt Lake City, UT, USA, 18-22 June.
- [9] Zimmermann C., Brox T., (2017), “Learning to estimate 3D hand pose from single RGB images”, *International Conference on Computer Vision*, 4913–4921, Venice, Italy, 22-29 October.
- [10] Mueller F., Bernard F., Sotnychenko O., Mehta D., Sridhar S., Casas D., Theobalt C., (2018), “GANerated hands for real-time 3D hand tracking from monocular RGB”, *International Conference on Computer Vision and Pattern Recognition*, 49–59, Salt Lake City, UT, USA, 18-22 June.
- [11] Kingma D. P., Welling M., (2014), “Auto-encoding variational bayes”, *International Conference on Learning Representations*, Banff, AB, Canada, April 14-16.

- [12] Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., Bengio Y., (2014), “Generative adversarial nets”, International Conference on Neural Information Processing Systems, 2672–2680, Montreal, Canada, 8-13 December.
- [13] Yang L., Yao A., (2019), “Disentangling latent hands for image synthesis and pose estimation”, International Conference on Computer Vision and Pattern Recognition, 9869–9878, Long Beach, CA, USA, 16-20 June.
- [14] Gu J., Wang Z., Ouyang W., Zhang W., Li J., Zhuo L., (2020), “3D hand pose estimation with disentangled cross-modal latent space”, Winter Conference on Applications of Computer Vision, 380–389, Snowmass Village, CO, USA, 1-5 March.
- [15] Panteleris P., Oikonomidis I., Argyros A., (2018), “Using a single RGB frame for real time 3D hand pose estimation in the Wild”, Winter Conference on Applications of Computer Vision, Lake Tahoe, NV, USA, 12-15 March.
- [16] Cai Y., Ge L., Cai J., Yuan J., (2018), “Weakly-supervised 3D hand pose estimation from monocular RGB images”, Lecture Notes in Computer Science, 678–694.
- [17] Iqbal U., Molchanov P., Breuel T., Gall J., Kautz J., (2018), “Hand pose estimation via latent 2.5D heatmap regression”, Lecture Notes in Computer Science, 125–143.
- [18] Boukhayma A., de Bem R., Torr P. H. S., (2019), “3D hand shape and pose from images in the wild”, International Conference on Computer Vision and Pattern Recognition, 10835–10844, Long Beach, CA, USA, 16-20 June.
- [19] Zhou Y., Habermann M., Xu W., Habibie I., Theobalt C., Xu F., (2020), “Monocular real-time hand shape and motion capture using multi-modal data”, International Conference on Computer Vision and Pattern Recognition, 5345–5354, Virtual, 14-19 June.
- [20] Emami H., Aliabadi M. M., Dong M., Chinnam R. B., (2021), “SPA-GAN: spatial attention GAN for image-to-image translation”, IEEE Transactions on Multimedia, 23, 391–401.
- [21] Zhu J. Y., Park T., Isola P., Efros A. A., (2017), “Unpaired image-to-image translation using cycle-consistent adversarial networks”, International Conference on Computer Vision, 2223-2232, Venice, Italy, 22-29 October.
- [22] Kipf T. N., Welling M., (2017), “Semi-supervised classification with graph convolutional networks”, International Conference on Learning Representations, Toulon, France, 24-26 April.
- [23] Ge L., Ren Z., Li Y., Xue Z., Wang Y., Cai J., Yuan J., (2019), “3D hand shape and pose estimation from a single RGB image”, International Conference on

Computer Vision and Pattern Recognition, 10825–10834, Long Beach, CA, USA, 16-20 June.

- [24] Guo S., Rigall E., Qi L., Dong X., Li H., Dong J., (2021), “Graph-based CNNs with self-supervised module for 3d hand pose estimation from monocular RGB”, *IEEE Transactions on Circuits and Systems for Video Technology*, 31 (4), 1514–1525.
- [25] Cai Y., Ge L., Liu J., Cai J., Cham T. J., Yuan J., Thalmann N. M., (2019), “Exploiting spatial-temporal relationships for 3D pose estimation via graph convolutional networks”, *International Conference on Computer Vision*, 2272–2281. Seoul, Korea, 27 October-2 November.
- [26] Zhao L., Peng X., Tian Y., Kapadia M., Metaxas Di. N., (2019), “Semantic graph convolutional networks for 3D human pose regression”, *International Conference on Computer Vision and Pattern Recognition*, 3420–3430, Long Beach, CA, USA, 16-20 June.
- [27] Doosti B., Naha S., Mirbagheri M., Crandall D. J., (2020), “Hope-Net: A graph-based model for hand-Object pose estimation”, *International Conference on Computer Vision and Pattern Recognition*, 6607–6616, Virtual, 14-19 June.
- [28] Yu J., Lin Z., Yang J., Shen X., Lu X., Huang T., (2019), “Free-form image inpainting with gated convolution”, *International Conference on Computer Vision*, 4470–4479, Seoul, Korea, 27 October-2 November.
- [29] Huang X., Liu M. Y., Belongie S., Kautz J., (2018), “Multimodal unsupervised image-to-image translation”, *Lecture Notes in Computer Science*, 11207, 179–196.
- [30] Isola P., Zhu J. Y., Zhou T., Efros A. A., (2017), “Image-to-image translation with conditional adversarial networks”, *International Conference on Computer Vision and Pattern Recognition*, 5967–5976, Honolulu, Hawaii, USA, 22-25 July.
- [31] Karras T., Aittala M., Laine S., Härkönen E., Hellsten J., Lehtinen J., Aila T., (2021), “Alias-free generative adversarial networks”, *International Conference on Neural Information Processing Systems*, 34, Virtual, 6-14 December.
- [32] Tulyakov S., Liu M. Y., Yang X., Kautz J., (2018), “MoCoGAN: Decomposing motion and content for video generation”, *International Conference on Computer Vision and Pattern Recognition*, 1526–1535, Salt Lake City, UT, USA, 18-22 June.
- [33] Yang H., Zhang J., (2017), “Hand pose regression via a classification-guided approach”, *Lecture Notes in Computer Science*, 10113, 452–466.
- [34] Qi X., Liao R., Jia J., Fidler S., Urtasun R., (2017), “3D graph neural networks for RGBD semantic segmentation”, *International Conference on Computer Vision*, 5209–5218. Venice, Italy, 22-29 October.

- [35] Kumar A., Zhang D., (2006), “Personal recognition using hand shape and texture”, *IEEE Transactions on Image Processing*, 15 (8), 2454–2461.
- [36] Ong E. J., Bowden R., (2004), “A boosted classifier tree for hand shape detection”, *International Conference on Automatic Face and Gesture Recognition*, 889–894, Seoul, South Korea, 19-19 May.
- [37] Liu Z., Chai X., Liu Z., Chen X., (2017), “Continuous gesture recognition with hand-oriented spatiotemporal feature”, *International Conference on Computer Vision Workshops*, 3056–3064, Venice, Italy, 22-29 October.
- [38] Le T. H. N., Zheng Y., Zhu C., Luu K., Savvides M., (2016), “Multiple scale Faster-RCNN approach to driver’s cell-phone usage and hands-on steering wheel detection”, *International Conference on Computer Vision and Pattern Recognition Workshops*, 46–53, Las Vegas, NV, USA, 26 June-1 July.
- [39] Wan C., Probst T., van Gool L., Yao A., (2017), “Crossing nets: Combining gans and vaes with a shared latent space for hand pose estimation”, *International Conference on Computer Vision and Pattern Recognition*, 1196–1205, Honolulu, Hawaii, USA, 22-25 July.
- [40] Yang Y., Feng C., Shen Y., Tian D., (2018), “FoldingNet: Point cloud auto-encoder via deep grid deformation”, *International Conference on Computer Vision and Pattern Recognition*, 206–215, Salt Lake City, UT, USA, 18-22 June .
- [41] Garcia-Hernando G., Yuan S., Baek S., Kim T. K., (2018), “First-person hand action benchmark with RGB-D videos and 3D hand pose annotations”, *International Conference on Computer Vision and Pattern Recognition*, 409–419, Salt Lake City, UT, USA, 18-22 June.
- [42] Duan L., Shen M., Cui S., Guo Z., Deussen O., (2019), “Estimating 2D multi-hand poses from single depth images”, *Lecture Notes in Computer Science*, 11134, 257–272.
- [43] Gomez-Donoso F., Orts-Escolano S., Cazorla M., (2019), “Large-scale multiview 3D hand pose dataset”, *Image and Vision Computing*, 81, 25–33.
- [44] Defferrard M., Bresson X., Vandergheynst P., (2016), “Convolutional neural networks on graphs with fast localized spectral filtering”, *International Conference on Neural Information Processing Systems*, 3844–3852, Barcelona, Spain, 5-10 December.
- [45] Kong D., Chen Y., Ma H., Yan X., Xie X., (2020), “Adaptive graphical model network for 2D hand pose estimation”, *British Machine Vision Conference*, Virtual, 7-10 September.
- [46] Wang Y., Peng C., Liu Y., (2019), “Mask-pose cascaded CNN for 2D hand pose estimation from single color image”, *IEEE Transactions on Circuits and Systems for Video Technology*, 29 (11), 3258–3268.

- [47] Wang Y., Zhang B., Peng C., (2020), “SRHandNet: Real-time 2D hand pose estimation with simultaneous region localization”, *IEEE Transactions on Image Processing*, 29, 2977–2986.
- [48] Chen L., Lin S. Y., Xie Y., Lin Y. Y., Fan W., Xie X., (2020), “DGGAN: Depth-image guided generative adversarial networks for disentangling RGB and depth images in 3D hand pose estimation”, *Winter Conference on Applications of Computer Vision*, Snowmass Village, CO, USA, 1-5 March.
- [49] Simon T., Joo H., Matthews I., Sheikh Y., (2017), “Hand keypoint detection in single images using multiview bootstrapping”, *International Conference on Computer Vision and Pattern Recognition*, 4645–4653, Honolulu, Hawaii, USA, 22-25 July.
- [50] Oberweger M., Lepetit V., (2017), “DeepPrior++: Improving fast and accurate 3D hand pose estimation”, *International Conference on Computer Vision Workshops*, 585–594, Venice, Italy, 22-29 October.
- [51] Ge L., Ren Z., Yuan J., (2018), “Point-to-point regression point net for 3D hand pose estimation”, *Lecture Notes in Computer Science*, 11217, 489–505.
- [52] Li S., Lee D., (2019), “Point-to-pose voting based hand pose estimation using residual permutation equivariant layer”, *International Conference on Computer Vision and Pattern Recognition*, 11919–11928, Long Beach, CA, USA, 16-20 June.
- [53] Chang J. Y., Moon G., Lee K. M., (2018), “V2V-PostNet: Voxel-to-voxel prediction network for accurate 3D hand and human pose estimation from a single depth map”, *International Conference on Computer Vision and Pattern Recognition*, 5079–5088, Salt Lake City, UT, USA, 18-22 June.
- [54] Sharp T., Keskin C., Robertson D., Taylor J., Shotton J., Kim D., Rhemann C., Leichter I., Vinnikov A., Wei Y., Freedman D., Kohli P., Krupka E., Fitzgibbon A., Izadi S., (2015), “Accurate, robust, and flexible realtime hand tracking”, *Conference on Human Factors in Computing Systems*, 3633–3642, Seoul, South Korea, 18-23 April.
- [55] Taylor J., Bordeaux L., Cashman T., Corish B., Keskin C., Sharp T., Soto E., Sweeney D., Valentin J., Luff B., Topalian A., Wood E., Khamis S., Kohli P., Izadi S., Banks R., Fitzgibbon A., Shotton J., (2016), “Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences”, *ACM Transactions on Graphics*, 35 (4), 1-12.
- [56] Tang D., Yu T. H., Kim T. K., (2013), “Real-time articulated hand pose estimation using semi-supervised transductive regression forests”, *International Conference on Computer Vision*, 3224–3231, 1-8 December.
- [57] Qian C., Sun X., Wei Y., Tang X., Sun J., (2014), “Realtime and robust hand tracking from depth”, *International Conference on Computer Vision and Pattern Recognition*, 1106–1113, Columbus, OH, USA, 23-28 June.

- [58] Wang R., Paris S., Popović J., (2011), “6D hands: markerless hand-tracking for computer-aided design”, Annual ACM Symposium on User Interface Software and Technology, 549–558, Santa Barbara, CA, USA, 16-19 October.
- [59] Ge L., Liang H., Yuan J., Thalmann D., (2018), “Robust 3D hand pose estimation from single depth images using multi-view CNNs”, IEEE Transactions on Image Processing, 27 (9), 4422–4436.
- [60] Oikonomidis I., Kyriazis N., Argyros A., (2011), “Efficient model-based 3D tracking of hand articulations using Kinect”, British Machine Vision Conference, 101.1-101.11, Dundee, UK, 29 August- 2 September.
- [61] Zhang J., Jiao J., Chen M., Qu L., Xu X., Yang Q., (2016), “3D Hand pose tracking and estimation using stereo matching”, ArXiv Preprint ArXiv:1610.07214.
- [62] Qian C., Sun X., Wei Y., Tang X., Sun J., (2014), “Realtime and robust hand tracking from depth”, International Conference on Computer Vision and Pattern Recognition, 1106–1113, Columbus, OH, USA, 23-28 June.
- [63] Yikai F., Kongqiao W., Jian C., Hanqing L., (2007), “A real-time hand gesture recognition method”, International Conference on Multimedia and Expo, ICME 2007, 995–998, Beijing, China, 2-5 July.
- [64] Baek S., Kim K. I., Kim T. K., (2019), “Pushing the envelope for RGB-based dense 3D hand pose estimation via neural rendering”, International Conference on Computer Vision and Pattern Recognition, 1067–1076, Long Beach, CA, USA, 16-20 June.
- [65] Zhang X., Li Q., Mo H., Zhang W., Zheng W., (2019), “End-to-end hand mesh recovery from a monocular RGB image”, International Conference on Computer Vision, 2354–2364. Seoul, Korea, 27 October-2 November.
- [66] Zhao L., Peng X., Chen Y., Kapadia M., Metaxas D. N., (2020), “Knowledge as priors: cross-modal knowledge generalization for datasets without superior knowledge”, International Conference on Computer Vision and Pattern Recognition, 6527–6536. , Virtual, 14-19 June.
- [67] Lin F., Wilhelm C., Martinez T., (2021), “Two-hand global 3d pose estimation using monocular RGB”, Winter Conference on Applications of Computer Vision, 2372–2380, Virtual, 5-9 January.
- [68] Lin C. H., Yumer E., Wang O., Shechtman E., Lucey S., (2018), “ST-GAN: spatial transformer generative adversarial networks for image compositing”, International Conference on Computer Vision and Pattern Recognition, 9455–9464, Salt Lake City, UT, USA, 18-22 June.
- [69] Chen B. C., Kae A., (2019), “Toward realistic image compositing with adversarial learning”, International Conference on Computer Vision and Pattern Recognition, 8407–8416, Long Beach, CA, USA, 16-20 June.

- [70] Tan F., Bernier C., Cohen B., Ordonez V., Barnes C., (2018), “Where and who? Automatic semantic-aware person composition”, Winter Conference on Applications of Computer Vision, 1519–1528, Lake Tahoe, NV, USA, 12-15 March.
- [71] Azadi S., Pathak D., Ebrahimi S., Darrell T., (2018), “Compositional GAN: learning conditional image composition”, CoRR, abs/1807.0.
- [72] Zhan F., Lu S., Zhang C., Ma F., Xie X., (2021), “Adversarial image composition with auxiliary illumination”, Lecture Notes in Computer Science, 12623, 234–250.
- [73] Bianchi M., Salaris P., Bicchi A., (2013), “Synergy-based hand pose sensing: Optimal glove design”, International Journal of Robotics Research, 32 (4), 407–424.
- [74] Huang L., Zhang B., Guo Z., Xiao Y., Cao Z., Yuan J., (2021), “Survey on depth and RGB image-based 3D hand shape and pose estimation”, Virtual Reality and Intelligent Hardware, 3 (3), 207–234.
- [75] Carreira J., Agrawal P., Fragkiadaki K., Malik J., (2016), “Human pose estimation with iterative error feedback”, International Conference on Computer Vision and Pattern Recognition, 4733–4742, Las Vegas, NV, USA, 26 June-1 July.
- [76] Bulat A., Tzimiropoulos G., (2016), “Human pose estimation via convolutional part heatmap regression”, Lecture Notes in Computer Science, 9911, 717–732.
- [77] Papandreou G., Zhu T., Kanazawa N., Toshev A., Tompson J., Bregler C., Murphy K., (2017), “Towards accurate multi-person pose estimation in the wild”, International Conference on Computer Vision and Pattern Recognition, 3711–3719, Honolulu, Hawaii, USA, 22-25 July.
- [78] Ronneberger O., Fischer P., Brox T., (2015), “U-net: Convolutional networks for biomedical image segmentation”, International Conference on Medical Image Computing and Computer-Assisted Intervention, 234-241, Munich, Germany, 5-9 October.
- [79] Kong D., Ma H., Xie X., (2020), “SIA-GCN: A Spatial information aware graph neural network with 2D convolutions for hand pose estimation”. ArXiv Preprint ArXiv:2009.12473.
- [80] Kumar P. P., Vadakkepat P., Loh A. P., (2010), “Hand posture and face recognition using a fuzzy-rough approach”, International Journal of Humanoid Robotics, 7 (3), 331–356.
- [81] Potter L. E., Araullo J., Carter L., (2013), “The leap motion controller: A view on sign language”, Australian Computer-Human Interaction Conference, 175–178, Adelaide, Australia, 25-29 November.

- [82] Beardsley P., Murray D., Zisserman A., (1992), “Camera calibration using multiple images”, *Lecture Notes in Computer Science*, 588, 312–320.
- [83] Li S., Chan A. B., (2015), “3D human pose estimation from monocular images with deep convolutional neural network”, *Lecture Notes in Computer Science*, 9004, 332–347.
- [84] He K., Zhang X., Ren S., Sun J., (2016), “Deep residual learning for image recognition”, *International Conference on Computer Vision and Pattern Recognition*, 770-778, Las Vegas, NV, USA, 26 June-1 July.
- [85] Li X., Zhou Y., Sun Y., Lin X., Ma X., (2020), “A multi-branch hand pose estimation network with joint-wise feature extraction and fusion”, *Signal Processing: Image Communication*, 81, 115692.
- [86] Chen Z., Du K., Sun Y., Lin X., Ma X., (2020), “Hierarchical neural network for hand pose estimation”, *Signal Processing: Image Communication*, 87, 115909.
- [87] Zimmermann C., Ceylan D., Yang J., Russell B., Argus M. J., Brox T., (2019), “FreiHAND: A dataset for markerless capture of hand pose and shape from single RGB images”, *International Conference on Computer Vision*, 813–822. Seoul, Korea, 27 October-2 November.
- [88] Martinez J., Hossain R., Romero J., Little J. J., (2017), “A simple yet effective baseline for 3d human pose estimation”, *International Conference on Computer Vision*, 2659–2668. Venice, Italy, 22-29 October.
- [89] Fang H. S., Xu Y., Wang W., Liu X., Zhu S. C., (2018), “Learning pose grammar to encode human body configuration for 3D pose estimation”, *AAAI Conference on Artificial Intelligence*, 6821–6828, New Orleans, Louisiana, USA, February 2-7.
- [90] Wang Z., Zhang L., Li Y., Wang S., (2019), “Linkage based face clustering via graph convolution network”, *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019-June, 1117–1125, Long Beach, CA, USA, 16-20 June.
- [91] Paszke A., Gross S., Chintala S., Chanan G., Yang E., (2017), *Automatic differentiation in PyTorch*, *International Conference on Neural Information Processing Systems*, 0-0, Long Beach, CA, USA, 4-9 December.
- [92] Glorot X., Bengio Y., (2010), “Understanding the difficulty of training deep feedforward neural networks”, *Journal of Machine Learning Research*, 9, 249–256.
- [93] Zeiler M. D., (2012), “ADADELTA: An adaptive learning rate method”, *ArXiv Preprint ArXiv:1212.5701*.

- [94] Kingma D. P., Ba J. L., (2015), “Adam: A method for stochastic optimization”, International Conference for Learning Representations, 0-0, Banff, Canada, 14-16 April.
- [95] Zhang J., Jiao J., Chen M., Qu L., Xu X., Yang Q., (2018), “A hand pose tracking benchmark from stereo matching”, International Conference on Image Processing, 982–986, Athens, Greece, 7-10 October.
- [96] Chen L., Lin S. Y., Xie Y., Lin Y. Y., Xie X., (2021), “MVHM: A large-scale multi-view hand mesh benchmark for accurate 3D hand pose estimation”, Winter Conference on Applications of Computer Vision, 836–845, Virtual, 5-9 January.
- [97] Deng J., Guo J., Zhou Y., Yu J., Kotsia I., Zafeiriou S., (2019), “RetinaFace: single-stage dense face localization in the Wild”, Arxiv:1905.00641.
- [98] van der Maaten L., Hinton G., (2008), “Visualizing data using t-SNE”, Journal of Machine Learning Research, 9 (11), 2579–2625.
- [99] Chen L., Lin S. Y., Xie Y., Lin Y. Y., Xie X., (2021), “Temporal-aware self-supervised learning for 3D hand pose and mesh estimation in videos”, Winter Conference on Applications of Computer Vision, 1049–1058, Virtual, 5-9 January.
- [100] Tekin B., Bogo F., Pollefeys M., (2019), “H+O: Unified egocentric recognition of 3D hand-object poses and interactions”, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June, 4506–4515, Long Beach, CA, 16-20 June.
- [101] Liu M. Y., Breuel T., Kautz J., (2017), “Unsupervised image-to-image translation networks”, International Conference on Neural Information Processing Systems, 701–709, Long Beach, CA, USA, 4-9 December.
- [102] Hasson Y., Varol G., Tzionas Di., Kalevatykh I., Black M. J., Laptev I., Schmid C., (2019), “Learning joint reconstruction of hands and manipulated objects”, In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 11799–11808, Long Beach, CA, 16-20 June.

BIOGRAPHY

Ikram Kourbane received her B.Sc. and M.S degrees in Computer Engineering from the University of Batna (Algeria) in 2013 and 2015, respectively. She is currently pursuing her Ph.D. degree in Computer Engineering at Gebze Technical University (Turkey). Her research interests are in the areas of computer vision and machine learning.

APPENDICES

Appendix A: Publications within the Scope of the Thesis Study

- A hybrid classification-regression approach for 3D hand pose estimation using graph convolutional networks, in the Journal of Signal Processing Image Communication (2021).
- Skeleton-aware Multi-scale Heatmap Regression for 2D Hand Pose Estimation, in the Journal of Informatica (2021).
- A graph-based approach for absolute 3D hand pose estimation using a single RGB image. Minor revision submitted to the journal of Applied Intelligence (2022).

Appendix B: Additional Results of 2D Hand Pose Estimation

Figure B1.1 and Figure B1.2 show additional results for hand detection and 2D pose estimation tasks. We segment the hand skeleton to use it in our multi-scale heatmaps regression approach.



Figure B1.1: Additional results for hand detection and bounding box localization.



Figure B1.2: Direct regression vs. our multi-stage heatmaps regression method

Appendix C: 2D Qualitative Results

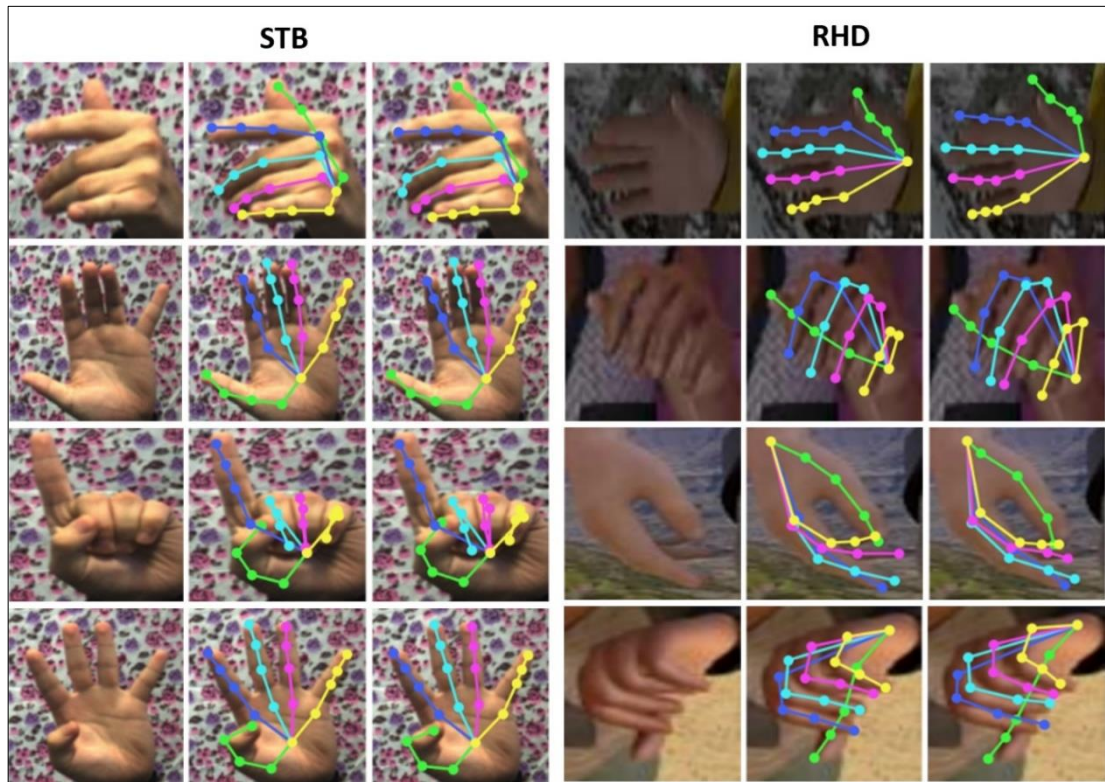


Figure C1.1: 2D hand pose estimation results on STB and RHD datasets.

Figure C1.1 provides qualitative results for 2D hand pose estimation using the HCRA approach. This task also has several applications, such as sign language and gesture recognition. We can see that the proposed approach produces accurate 2D poses on STB and RHD datasets.

Appendix D: Ground Truth of 2D/3D Joints Classification Task

In this section, we explain the creation of the ground truth 2D/3D join locations. The function `CreateClasses2D` in Figure D1.1 creates an array of size 21, where each value represents the class of a specific 2D joint. The inputs are the (x, y) coordinates of the 2D keypoints, the number of splits, and the image size (the width and height must be equal). Following, we explain the algorithm by specifying the line number and a brief description of its functionality. Note that our algorithms follow a Python style pseudo-code.

Algorithm 3: Creation of the 2D ground truth classes of each joint in a cropped hand.	
1	function <code>CreateClasses2D (2D, splits, size)</code> ;
	Input : (x, y) points, splits number and image size
	Output : An array containing the class of each 2D joint
2	$block \leftarrow \text{int}(\text{size}/\text{splits})$;
3	$\text{parts}, \text{data}, \text{output} \leftarrow []$;
4	$\text{classes} \leftarrow \{\}$;
5	$k \leftarrow 0$;
6	for i in $\text{range}(0, \text{size} + \text{block}, \text{block})$ do
7	$\text{parts}[i] \leftarrow i$;
8	end
9	for $j=0:\text{split}$ do
10	for $i=0:\text{split}$ do
11	$\text{data}[k] \leftarrow [\text{parts}[j], \text{parts}[j + 1], \text{parts}[i], \text{parts}[i + 1]]$;
12	$k \leftarrow k + 1$;
13	end
14	end
15	for $i=0:21$ do
16	$\text{classes}[i] \leftarrow [0, ..., 0]$;
17	end
18	for $\text{key}, (a_x, b_x, a_y, b_y)$ in $\text{data.items}()$ do
19	$r \leftarrow (a_x \leq 2D[:, 0]) \& (b_x \geq 2D[:, 0]) \& (a_y \leq 2D[:, 1]) \& (b_y \geq 2D[:, 1])$;
20	for $i=0:21$ do
21	if $r[i] = \text{True}$ then
22	$\text{classes}[i][\text{key}] \leftarrow 1$;
23	end
24	end
25	end
26	$i \leftarrow 0$;
27	for key, val in $\text{classes.items}()$ do
28	$\text{output}[i] \leftarrow \text{argmax}(\text{val})$;
29	$i \leftarrow i + 1$;
30	end
31	return output ;

Figure D1.1: Algorithm represents the creation of the 2D ground-truth classes of each joint in a cropped hand.

We follow the algorithm in Figure D1.2 to create the ground truth classes of each joint in the 3D space.

- Line 2 calculates the size of each block.

- Line 6 to 8 initializes the *parts* array, which represents the beginning and the end of the blocks in the x and y axis, e.g.: if the number of splits is 4, *parts* = [0, 64, 128, 192, 256], where $C_0 \in [0, 64]$, $C_1 \in [64, 128]$, etc.
- Line 9 to 14: for each block (class), *data* contains the x and y intervals, where the joints are located, e.g.: *data*[0] = [0, 32, 64, 128] if $joint_0$ is in the interval [0, 32] and [64, 128] in the x and y axis, respectively.
- In Line 15 to 17, *classes* is a dictionary, where the key is the joint number, and the value is a zero vector. To fill in the latter, we iterate over all the (*key*, *value*) pairs in *data* to evaluate the boolean expression in Line 19. The latter creates a boolean vector of size 21 such that each value is True/False if the joint is inside/outside the x and y intervals.
- Line 20 to 24: we get the one-hot vector of each joint by assigning the position of the correct class in the zero-vector to 1.
- From Line 27 to 30: *output* is an array of size 21, where we get the class of each joint using the Argmax function on the one-hot vectors.

Algorithm 4: Creation of the 3D ground truth classes of each joint in a cropped hand.	
1	function Create3DClasses ($3D, splits$);
	Input : (x, y, z) points and splits number
	Output : An array containing the class of each 3D joint
2	$start_{x_{3d}}, start_{y_{3d}}, start_{z_{3d}} \leftarrow \min(3D[:, 0]), \min(3D[:, 1]), \min(3D[:, 2]);$
3	$end_{x_{3d}}, end_{y_{3d}}, end_{z_{3d}} \leftarrow \max(3D[:, 0]), \max(3D[:, 1]), \max(3D[:, 2]);$
4	$interval_{x_{3d}} \leftarrow end_{x_{3d}} - start_{x_{3d}};$
5	$interval_{y_{3d}} \leftarrow end_{y_{3d}} - start_{y_{3d}};$
6	$interval_{z_{3d}} \leftarrow end_{z_{3d}} - start_{z_{3d}};$
7	$step_x \leftarrow \text{float}(interval_{x_{3d}}/splits);$
8	$step_y \leftarrow \text{float}(interval_{y_{3d}}/splits);$
9	$step_z \leftarrow \text{float}(interval_{z_{3d}}/splits);$
10	$parts_x, parts_y, parts_z \leftarrow [];$
11	$data, output \leftarrow [];$
12	$classes \leftarrow \{ \};$
13	$k \leftarrow 0;$
14	$range_x = \text{arange}(start_{x_{3d}}, end_{x_{3d}} + step_x, step_x);$
15	$range_y = \text{arange}(start_{y_{3d}}, end_{y_{3d}} + step_y, step_y);$
16	$range_z = \text{arange}(start_{z_{3d}}, end_{z_{3d}} + step_z, step_z);$
17	for i, j, k in $range_x, range_y, range_z$ do
18	$parts_{x[i]} \leftarrow i, parts_{y[j]} \leftarrow j, parts_{z[k]} \leftarrow k;$
19	end
20	for $i=0:21$ do
21	$classes[i] \leftarrow [0, ..., 0];$
22	end
23	for $j=0:n_splits$ do
24	for $i=0:n_splits$ do
25	for $p=0:n_splits$ do
26	$data[k] \leftarrow [parts_x[j], parts_x[j+1], parts_y[i], parts_y[i+1], parts_z[p], parts_z[p+1]];$
27	$k \leftarrow k+1;$
28	end
29	end
30	end
31	for $key, (a_x, b_x, a_y, b_y, a_z, b_z)$ in $data.items()$ do
32	$r \leftarrow (a_x \leq 3D[:, 0]) \& (b_x \geq 3D[:, 0]) \& (a_y \leq 3D[:, 1]) \& (b_y \geq 3D[:, 1]) \& (a_z \leq 3D[:, 2])$ $\& (b_z \geq 3D[:, 2]);$
33	for $i=0:21$ do
34	if $r[i] = \text{True}$ then
35	$classes[i][key] \leftarrow 1;$
36	end
37	end
38	end
39	$i \leftarrow 0;$
40	for key, val in $classes.items()$ do
41	$output[i] \leftarrow \text{argmax}(val);$
42	$i \leftarrow i+1;$
43	end
44	return $output;$

Figure D1.2: Algorithm represents the creation of the 3D ground truth of each joint in a cropped hand.

Appendix E: Additional Results of Absolute 3D Hand Pose Estimation

Figure E1.1 shows additional results of the absolute 3D hand pose estimation task on the STB dataset.

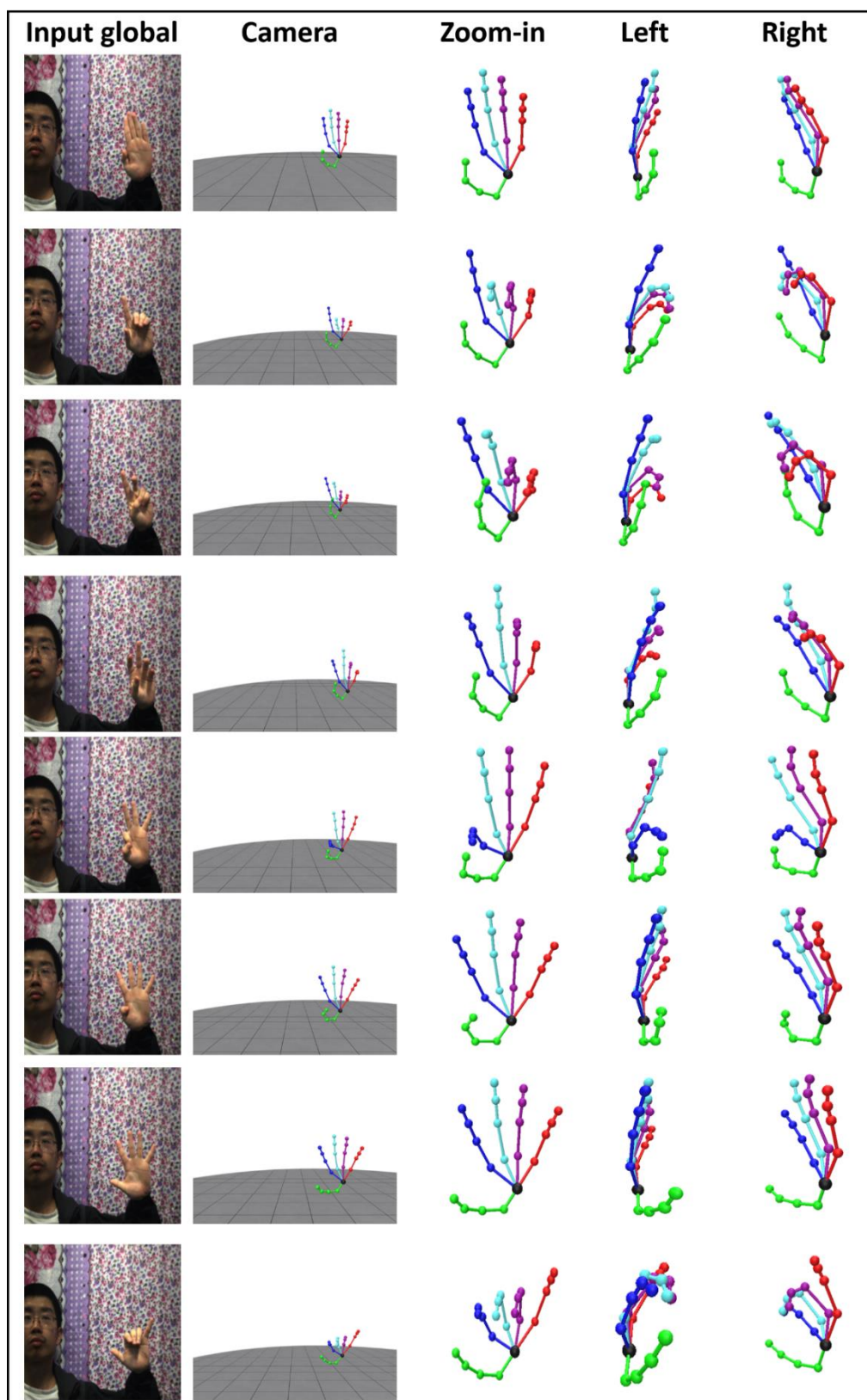


Figure E1.1: Absolute 3D hand pose estimation results on the STB dataset.