REAL-TIME 3D HAND RECONSTRUCTION IN CHALLENGING SCENES FROM A SINGLE COLOR OR DEPTH CAMERA

FRANZISKA MÜLLER

Dissertation zur Erlangung des Grades der

Doktorin der Ingenieurwissenschaften (Dr.-Ing.)

der Fakultät für Mathematik und Informatik
der Universität des Saarlandes

Saarbrücken, 2020

Date of Colloquium:	December 2, 2020
Dean of the Faculty:	Prof. Dr. Thomas Schuster
Chair of the Committee:	Prof. Dr. Jürgen Steimle
Reviewers:	Prof. Dr. Christian Theobalt
	Prof. Dr. Hans-Peter Seidel
	Dr. Shahram Izadi
Academic Assistant:	Dr. Vladislav Golyanik

Für meine Mutter Maria, du bist die Beste.

ABSTRACT

Hands are one of the main enabling factors for performing complex tasks and humans naturally use them for interactions with their environment. Reconstruction and digitization of 3D hand motion opens up many possibilities for important applications. Hands gestures can be directly used for human–computer interaction, which is especially relevant for controlling augmented or virtual reality (AR/VR) devices where immersion is of utmost importance. In addition, 3D hand motion capture is a precondition for automatic sign-language translation, activity recognition, or teaching robots. Different approaches for 3D hand motion capture have been actively researched in the past. While being accurate, gloves and markers are intrusive and uncomfortable to wear. Hence, markerless hand reconstruction based on cameras is desirable. Multi-camera setups provide rich input, however, they are hard to calibrate and lack the flexibility for mobile use cases. Thus, the majority of more recent methods uses a single color or depth camera which, however, makes the problem harder due to more ambiguities in the input. For interaction purposes, users need continuous control and immediate feedback. This means the algorithms have to run in real time and be robust in uncontrolled scenes. These requirements, achieving 3D hand reconstruction in real time from a single camera in general scenes, make the problem significantly more challenging. While recent research has shown promising results, current state-of-the-art methods still have strong limitations. Most approaches only track the motion of a single hand in isolation and do not take background-clutter or interactions with arbitrary objects or the other hand into account. The few methods that can handle more general and natural scenarios run far from real time or use complex multi-camera setups. Such requirements make existing methods unusable for many aforementioned applications. This thesis pushes the state of the art for real-time 3D hand tracking and reconstruction in general scenes from a single RGB or depth camera. The presented approaches explore novel combinations of generative hand models, which have been used successfully in the computer vision and graphics community for decades, and powerful cutting-edge machine learning techniques, which have recently emerged with the advent of deep learning. In particular, this thesis proposes a novel method for hand tracking in the presence of strong occlusions and clutter, the first method for full global 3D hand tracking from in-the-wild RGB video, and a method for simultaneous

pose and dense shape reconstruction of two interacting hands that, for the first time, combines a set of desirable properties previously unseen in the literature.

ZUSAMMENFASSUNG

Hände sind einer der Hauptfaktoren für die Ausführung komplexer Aufgaben, und Menschen verwenden sie auf natürliche Weise für Interaktionen mit ihrer Umgebung. Die Rekonstruktion und Digitalisierung der 3D-Handbewegung eröffnet viele Möglichkeiten für wichtige Anwendungen. Handgesten können direkt als Eingabe für die Mensch-Computer-Interaktion verwendet werden. Dies ist insbesondere für Geräte der erweiterten oder virtuellen Realität (AR / VR) relevant, bei denen die Immersion von größter Bedeutung ist. Darüber hinaus ist die Rekonstruktion der 3D Handbewegung eine Voraussetzung zur automatischen Übersetzung von Gebärdensprache, zur Aktivitätserkennung oder zum Unterrichten von Robotern. In der Vergangenheit wurden verschiedene Ansätze zur 3D-Handbewegungsrekonstruktion aktiv erforscht. Handschuhe und physische Markierungen sind zwar präzise, aber aufdringlich und unangenehm zu tragen. Daher ist eine markierungslose Handrekonstruktion auf der Basis von Kameras wünschenswert. Multi-Kamera-Setups bieten umfangreiche Eingabedaten, sind jedoch schwer zu kalibrieren und haben keine Flexibilität für mobile Anwendungsfälle. Daher verwenden die meisten neueren Methoden eine einzelne Farb- oder Tiefenkamera, was das Aufgabe jedoch schwerer macht, da mehr Ambiguitäten in den Eingabedaten vorhanden sind. Für Interaktionszwecke benötigen Benutzer kontinuierliche Kontrolle und sofortiges Feedback. Dies bedeutet, dass die Algorithmen in Echtzeit ausgeführt werden müssen und robust in unkontrollierten Szenen sein müssen. Diese Anforderungen, 3D-Handrekonstruktion in Echtzeit mit einer einzigen Kamera in allgemeinen Szenen, machen das Problem erheblich schwieriger. Während neuere Forschungsarbeiten vielversprechende Ergebnisse gezeigt haben, weisen aktuelle Methoden immer noch Einschränkungen auf. Die meisten Ansätze verfolgen die Bewegung einer einzelnen Hand nur isoliert und berücksichtigen keine alltäglichen Umgebungen oder Interaktionen mit beliebigen Objekten oder der anderen Hand. Die wenigen Methoden, die allgemeinere und natürlichere Szenarien verarbeiten können, laufen nicht in Echtzeit oder verwenden komplexe Multi-Kamera-Setups. Solche Anforderungen machen bestehende Verfahren für viele der oben genannten Anwendungen unbrauchbar. Diese Dissertation erweitert den Stand der Technik für die Echtzeit-3D-Handverfolgung und -Rekonstruktion in allgemeinen Szenen mit einer einzelnen RGB- oder Tiefenkamera. Die vorgestellten Algorithmen erforschen neue Kombinationen aus genera-

tiven Handmodellen, die seit Jahrzehnten erfolgreich in den Bereichen Computer Vision und Grafik eingesetzt werden, und leistungsfähigen innovativen Techniken des maschinellen Lernens, die vor kurzem mit dem Aufkommen neuronaler Netzwerke entstanden sind. In dieser Arbeit werden insbesondere vorgeschlagen: eine neuartige Methode zur Handbewegungsrekonstruktion bei starken Verdeckungen und in unkontrollierten Szenen, die erste Methode zur Rekonstruktion der globalen 3D Handbewegung aus RGB-Videos in freier Wildbahn und die erste Methode zur gleichzeitigen Rekonstruktion von Handpose und -form zweier interagierender Hände, die eine Reihe wünschenwerter Eigenschaften komibiniert.

ACKNOWLEDGMENTS

This journey took more than four years overall and I am deeply grateful to all the amazing people who I have met along the way.

First of all, I would like to thank Christian Theobalt for his support and guidance and countless insightful discussions. You are a great supervisor who is really invested in every project and the research group you have built is a wonderful place for pursuing a PhD. I would like to thank Hans-Peter Seidel and Shahram Izadi for being part of my thesis committee as well as my proofreaders, Christian Richardt, Jiayi Wang, Marc Habermann, Vladislav Golyanik, Gereon Fox, Dushyant Mehta, and Edgar Tretschk, for their valuable comments. I am thankful to my postdocs Dan Casas and Florian Bernard for their advice and the effort they put into our projects. It was a pleasure working with you. A big thank you goes to Srinath Sridhar who first introduced me to hand tracking when supervising my Bachelor Thesis and from whom I inherited the job of the *GVV hand tracking person*. Another big thank you goes to the person who will inherit (or has already inherited ☺) this job from me, Jiayi Wang. It was great sharing my knowledge with you and I think I have learned at least as much from you. You will always be one of the best office mates I could have wished for and of course the Master of Boardgames. I would like to thank Dushyant Mehta for explaining me many things from neural networks over Indian culture to book and podcast suggestions and for always having a sympathetic ear... and a cup of coffee or tea. I am also grateful to Oleksandr Sotnychenko for all the hours spent recording or programming together, even when it was shortly before a deadline, it was fun. A big round of thanks goes to all members of the Graphics, Vision and Video group and the Computer Graphics department for creating an excellent and inspiring working atmosphere, but also for all the great experiences apart from work. I want to explicitly mention here the bouldering group around Marc Habermann and our D4 Calisthenics crew Jozef Hladký, Ayush Tewari, Krzysztof Wolski, Mojtaba Bemana, and Gereon Fox.

Last but not least, I want to thank my friends and family for their support and love.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

1.1 MOTIVATION

It is the most natural thing for humans to use their hands for everyday interactions with others or with the environment. Also when interacting with machines, and especially computers, the hands are used to press buttons, keys on a keyboard or operate a mouse. However, hands are capable of performing far more dexterous and expressive motions than these input modalities allow. In addition, with the ever increasing popularity and presence of smart and mobile devices, and especially augmented and virtual reality (AR/VR) headsets, the use of conservative input setups is simply not feasible. Technologies for hand tracking and reconstruction enable the use of hands directly as input devices, which is more natural and immersive, removes the need for separate input devices, and can potentially use all degrees of freedom in the hand (see Figure 1.1). Furthermore, these technologies can be employed for automatic sign-language recognition, translation, and generation to increase accessibility and ease communication. Another large area of application is robotics. With automatic hand reconstruction, robots can learn how to interact with their environment and manipulate objects by observing a human teacher, significantly decreasing the manual programming effort. This applies to all kinds of robots, ranging from manufacturing robots in industry to assistive robots that can take care of elderly people.

To make hand tracking and reconstruction usable for the aforementioned applications and to a wide set of users, the methods need to work accurately and robustly in real time for general scenes and with a simple and flexible hardware setup. Since intrusive physical sensors, like data gloves, are impractical or uncomfortable to wear, it is desirable to perform full articulated hand tracking with cameras, but without the use of optical markers on the body as needed by some motion capture systems. For many applications, a single camera is preferable. Multi-camera setups cannot be easily calibrated by non-expert users and they lack the flexibility necessary for the use with mobile devices. For interaction purposes, users need continuous control and immediate feedback. Methods need to run robustly and in real time in uncontrolled environments, for example in a cluttered living room and not only in a research lab. This necessity is even made more crucial by the recent advances in virtual and augmented

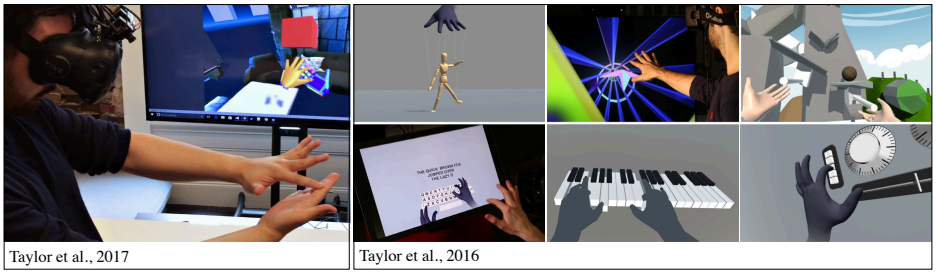


Figure 1.1: Real-time 3D hand motion capture and reconstruction enables diverse applications in virtual reality and gaming. ©The respective copyright owners.

reality, where users are carrying their wearable devices wherever they like. Without a robust, accurate and temporally consistent reconstruction and feedback in real time, the immersion immediately breaks.

Based only on optical sensors, hand tracking and reconstruction is a challenging task, especially in general scenes. Complex hand poses often result in self-occlusions and ambiguities that are a lot more severe than in full-body pose estimation. When restricted to a single camera view, these problems become even harder. In particular, ambiguities in pose could be caused by the self-similarity between fingers or by depth ambiguities. Whereas the color channel often helps when tracking human bodies or objects, e.g. to discriminate different parts, the information is ambiguous for tracking hands due to their mostly uniform color. This, together with the expressiveness of the human hand, leads to a highly underdetermined pose estimation problem. When considering general scenes with clutter and arbitrary objects, it becomes a challenging task to segment hands in the first place. Furthermore, occlusions are stronger and, if objects are tracked at the same time, physical constraints need to be taken into account.

While recent research has shown promising results, current state-of-the-art methods still have limitations. Many approaches only track the motion of a single hand, sometimes in conjunction with a single known object, in an isolated environment. They rarely tackle egocentric views, i.e., where the camera is body-mounted, due to more complicated self-occlusions, and rather focus on exocentric views, i.e., where the user is standing or sitting in front of the camera. The majority of existing approaches also does not take clutter or interactions with arbitrary objects or the other hand into account. The few methods that can handle more general and natural scenarios have runtimes that are far from real time or use complex calibrated multi-camera setups. Such requirements make existing methods unusable in unconstrained contexts, where the user's environment cannot be controlled, or priors on how and with which

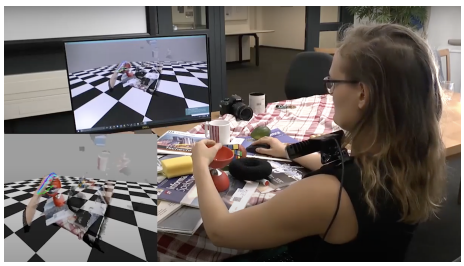
objects users are interacting cannot be imposed. However, having reliably working methods in such contexts is essential for many applications as discussed before.

Therefore, this thesis pushes the state of the art for real-time hand tracking and reconstruction in general scenes from a single camera with unconstrained viewpoint. In particular, this thesis proposes novel methods for hand tracking in the presence of strong occlusions and clutter, hand tracking from in-the-wild RGB video, and simultaneous pose and dense shape reconstruction of two interacting hands.

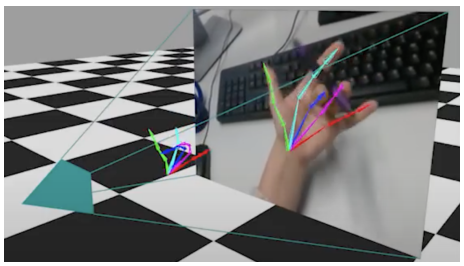
1.2 OVERVIEW

The goal of this thesis is to explore real-time methods to reconstruct a single hand or two hands in general scenes from a single RGB or RGB-D camera (see Figure 1.2). The methods should work for an unconstrained camera view, i.e., for both egocentric and exocentric viewpoints. In this thesis, first-person or egocentric viewpoints are defined as those that would typically be imaged by cameras mounted on the head, shoulder, or chest. Third-person or exocentric viewpoints are recorded by a camera that is standing at a fixed location in front of the user. The obtained hand reconstructions should contain the 3D articulated motion of the hand at a minimum, but might be extended to include the dense 3D hand surface, if possible. The presented methods introduce new ways to combine optimization-based fitting of kinematic hand models with machine learning components to ensure a robust, temporally smooth, and biomechanically plausible result.

Since data is key to train accurate and robust machine learning components, this thesis first introduces several approaches to generate annotated datasets of realistic and complex hand motion and interactions (Chapter 4). A novel merged reality capture setup is introduced to produce the fully annotated synthetic hand dataset *SynthHands* with plausible hand–object interactions, while only needing motion data of one unoccluded hand. To reduce the domain gap between synthetic and real images, a novel geometrically consistent generative adversarial network is used to perform unpaired image-to-image translation, thus creating more realistic, so called *GANerated*, data. Next, an approach for automatic labeling of real depth data with per-pixel segmentation or hand part labels is presented. The method leverages colored gloves or painted hands in combination with a calibrated RGB camera. The concept of discrete hand part labels is generalized to continuous dense surface correspondences for which it is, however, impossible to annotate real data. Hence, a sophisticated motion-capture-driven physical simulation framework

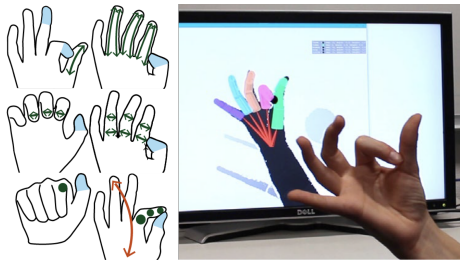


(a) Chapter 5: 3D hand motion capture in presence of strong occlusions and clutter.



(b) Chapter 6: 3D hand motion capture from in-the-wild RGB video.

(c) Chapter 7: Simultaneous pose and dense shape reconstruction of two interacting hands.



(d) Chapter 8: 3D hand motion capture and accurate touch point estimation for thumb-to-finger microgesture recognition.

Figure 1.2: All methods presented in this thesis run in real time using a single color or depth camera.

is introduced to create the synthetic *DenseHands* dataset. It extends the previous datasets by offering realistic two-hand interactions while only requiring a tracking approach for a single hand.

After the datasets have been introduced, the thesis presents several approaches for 3D hand reconstruction that tackle different challenging aspects of the problem. All the methods presented in this thesis push the state of the art of real-time hand tracking and reconstruction.

First, this thesis tackles real-time estimation of the full articulated motion of a single hand *under occlusion and in cluttered scenes* from a single egocentric RGB-D camera (Chapter 5). A first neural network locates the possibly occluded hand in the complex scene while a second neural network regresses joint positions. The articulation parameters of a kinematic hand skeleton, rigid transform and joint angles, are estimated by fitting it to the regressed positions.

Next, the employed sensor setup is further simplified to a *standard RGB camera* with unrestricted viewpoint (Chapter 6). The full parameters of a kinematic skeleton are estimated by minimizing the discrepancy between the skeleton's 2D and 3D joint locations as predicted by a neural network. It is shown that using the enhanced *GANerated Hands* training data is

key to obtain accurate results on real monocular RGB images. This is because the RGB-only network, due to the missing depth modality, is more sensitive to the domain gap between synthetic and real RGB data. As a result, the method is applicable to more general in-the-wild videos, like legacy videos found on YouTube.

Subsequently, this thesis explores real-time reconstruction of *two interacting hands* from a single depth camera (Chapter 7). The method does not only estimate the full articulated motion of both hands but also reconstructs dense 3D shape of both hands. A neural network is used to regress segmentation masks and dense vertex correspondences to a hand model from a depth image. The pose and shape parameters of the two interacting hands are estimated in an energy minimization framework which uses the regressed correspondences.

Finally, a new application of state-of-the-art real-time 3D hand tracking is demonstrated by *FingerInput*, a system for thumb-to-finger microgesture recognition. Leveraging hand part segmentation by a neural network in conjunction with a fully articulated hand model, the system supports a more extensive and richer gesture set compared to any previous work.

1.3 STRUCTURE

This thesis is divided into nine chapters:

- Chapter 1 provides motivation for the topic of this thesis, gives an overview of the work, explains the structure of the thesis and emphasizes the main contributions.
- Chapter 2 discusses previous work in the field of 3D hand reconstruction.
- Chapter 3 introduces the concept of a kinematic hand model and the hand models used throughout the thesis.
- Chapter 4 presents several new datasets that were created for building robust and accurate machine-learning components to be used in the reconstruction approaches.
- Chapters 5, 6, and 7 propose novel methods that tackle challenging aspects of real-time 3D hand reconstruction in general scenes, and provide extensive experimentation and results.
- Chapter 8 presents a novel application of real-time 3D hand tracking, namely a recognition system for thumb-to-finger microgestures.
- Chapter 9 discusses important insights and core contributions of this thesis as well as opportunities for future work.

1.4 CONTRIBUTIONS

This section summarizes the main contributions of this thesis.

The contributions of Chapter 4 are:

- A data generation framework for synthesizing an extensive annotated RGB-D dataset, *SynthHands*, of hands in natural interaction with objects and clutter (published as part of Mueller et al., 2017).
- An enhanced synthetic RGB hand image dataset, the *GANerated Hands* dataset, whose statistical distribution resembles real-world hand images. This is achieved by a novel geometrically consistent generative adversarial network that performs image-to-image translation while preserving poses during translation (published as part of Mueller et al., 2018).
- A new depth-based dataset for per-pixel left/right hand segmentation as well as a novel per-pixel hand part dataset which were automatically annotated using hands colored with body paint and a calibrated RGB camera (published as parts of Mueller et al., 2019 and Soliman et al., 2018, respectively).
- The first two-hand tracking dataset, *DenseHands*, that includes both pose and dense shape annotations. The creation process leverages a single-hand tracker in conjunction with a live physical simulation system to obtain realistic interactions while avoiding inter-hand penetrations (published as part of Mueller et al., 2019).

The contributions of Chapter 5 (published as Mueller et al., 2017) are:

- A novel method that localizes the hand and estimates, in real time, the 3D joint locations from egocentric viewpoints, in clutter, and under strong occlusions using two convolutional neural networks. A kinematic pose tracking energy further refines the pose by estimating joint angles of a temporally smooth tracking.
- Extensive evaluation on a new annotated real benchmark dataset *EgoDexter* featuring egocentric cluttered scenes, interaction with objects, and a diverse set of users.

The contributions of Chapter 6 (published as Mueller et al., 2018) are:

- The first real-time hand tracking system that tracks global 3D joint positions from unconstrained monocular RGB-only images and video.

- Experiments about the influence of the domain gap between synthetic and real images for 3D hand pose estimation from monocular RGB.

The contributions of Chapter 7 (published as Mueller et al., 2019) are:

- The first method that can track two interacting hands in real time with a single depth camera, while at the same time being able to estimate the hand shape automatically and taking collisions into account.
- Contrary to existing methods, the presented approach is more robust and reliable in involved hand–hand interaction settings.

The contributions of Chapter 8 (published as Soliman et al., 2018) are:

- A real-time method for 3D hand tracking based on the combination of the generative Sum of Gaussians hand model and a neural network for hand part classification.
- An approach for fast and accurate detection and precise localization of on-skin touch points for thumb-to-finger microgesture recognition.

1.5 PUBLICATIONS

All the work presented in this thesis was also published in the following publications:

- Franziska Mueller et al. (2017). "Real-time Hand Tracking under Occlusion from an Egocentric RGB-D Sensor." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 1163–1172
- Franziska Mueller et al. (2018). "GANerated Hands for Real-Time 3D Hand Tracking from Monocular RGB." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 49–59
- Mohamed Soliman et al. (2018). "FingerInput: Capturing Expressive Single-Hand Thumb-to-Finger Microgestures." In: *Proceedings of the International Conference on Interactive Surfaces and Spaces (ISS)*. ACM, pp. 177–187 **[Best Academic Paper Award]**

- Franziska Mueller et al. (2019). "Real-time Pose and Shape Reconstruction of Two Interacting Hands with a Single Depth Camera." In: *ACM Transactions on Graphics (TOG)* 38.4, pp. 1–13

In addition, contributions were made to the following publications which are, however, not part of this thesis:

- Dushyant Mehta et al. (2018). "Single-Shot Multi-Person 3D Pose Estimation From Monocular RGB." In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, pp. 120–130
- Abhishake Kumar Bojja et al. (2019). "HandSeg: An Automatically Labeled Dataset for Hand Segmentation from Depth Images." In: *Proceedings of the Conference on Computer and Robot Vision (CRV)*. IEEE, pp. 151–158
- Tarun Yenamandra et al. (2019). "Convex Optimisation for Inverse Kinematics." In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, pp. 318–327
- Dushyant Mehta et al. (2020). "XNect: Real-time Multi-Person 3D Motion Capture with a Single RGB Camera." In: *ACM Transactions on Graphics (TOG)* 39.4
- Jiayi Wang et al. (2020). "Generative Model-Based Loss to the Rescue: A Method to Overcome Annotation Errors for Depth-Based Hand Pose Estimation." In: *Proceedings of the International Conference on Automatic Face and Gesture Recognition (FG)*. IEEE, pp. 93–100
- Neng Qian et al. (2020). "HTML: A Parametric Hand Texture Model for 3D Hand Reconstruction and Personalization." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer

2 RELATED WORK

Hand pose estimation techniques have a rich history due to many possible applications, e.g., in human–computer interaction, AR/VR interfaces, motion control, and activity recognition. Markers or gloves were used to reconstruct hand poses in earlier work or methods that focus solely on high-quality results (Glauser et al., 2019; Han et al., 2018; Wang and Popović, 2009). These setups are inflexible and not usable for ubiquitous or mobile settings. Some used a multi-camera setup to deal with occlusions and ambiguities while losing flexibility and processing speed which are both essential for interactive techniques (Ballan et al., 2012; Sridhar et al., 2013; Wang et al., 2011). Most of the recent approaches refrain from using markers for flexibility, aim at real-time frame rates and use a single RGB or RGB-D camera to enable adaptability for mobile setups. The following review of related work focuses on such methods since they are most similar in spirit to the approaches proposed in this thesis.

2.1 TYPES OF HAND RECONSTRUCTION ALGORITHMS

The first class of methods, the so-called *generative* methods, assumes the availability of a generative model of the hand, ranging from meshes, collections of geometric primitives, to implicit functions, as depicted in Figure 2.1 (Heap and Hogg, 1996; Oikonomidis et al., 2011a; Tagliasacchi et al., 2015; Taylor et al., 2016, 2017; Tkach et al., 2016). During pose optimization, the image formation model is employed to compare the hand model at its current pose to the input image and this discrepancy is minimized. Such hand models are usually personalized to individual users and are obtained manually, e.g., by laser scans or simple scaling of a base model. Only few methods estimate a detailed hand shape automatically. Khamis et al., 2015 build a shape model of a hand mesh from sets of depth images acquired from different users. A method for efficiently fitting this model to a sequence of a new actor was subsequently presented by Tan et al., 2016. Tkach et al., 2017 jointly optimize pose and shape of a sphere mesh online, and accumulate shape information over time to minimize uncertainty. In contrast, Remelli et al., 2017 fit a sphere mesh directly to the whole image set by multi-stage calibration with local anisotropic scalings. Romero et al., 2017 introduce a parametric

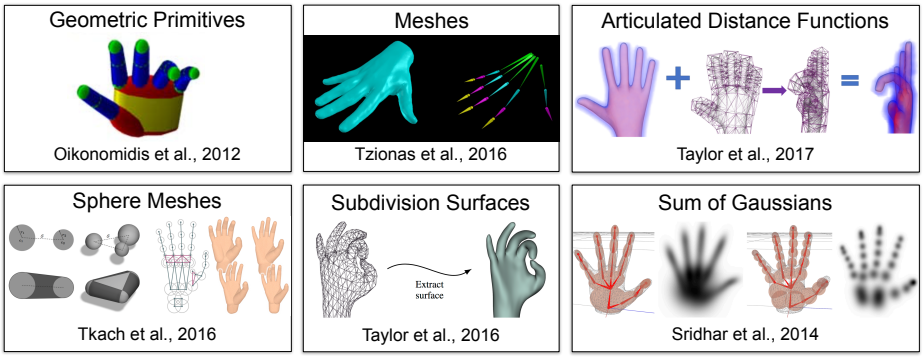


Figure 2.1: Different hand models used in the literature. ©The respective copyright owners.

model of hand shape and pose which can be used for generative model fitting. Generative methods usually enforce temporal consistency but are therefore prone to both propagating errors over time and getting stuck in poor local optima. They do not have a training stage and are hence independent of any biases that might be present in large training data corpora.

On the other end of the spectrum, there are *discriminative* data-driven methods that often perform independent per-frame pose estimations. They are based on machine learning techniques and usually depend on huge pose databases for training or retrieval (Athitsos and Sclaroff, 2003; Tompson et al., 2014; Wang and Popović, 2009; Zhou et al., 2016). Random forests have been a popular choice (Keskin et al., 2012; Li et al., 2015; Sun et al., 2015; Tang et al., 2014; Wan et al., 2016; Xu and Cheng, 2013) but most of the more recent methods resort to using neural networks because they promise large learning capacities for hand pose estimation (Baek et al., 2018; Ge et al., 2016, 2018; Oberweger et al., 2015; Sinha et al., 2016; Wan et al., 2017; Ye et al., 2016). Some of these approaches run an inverse kinematics step to fit a model to the predictions. While these methods do not propagate errors over frames and can exploit the knowledge priors they built during training at test time, they suffer from temporal jitter and might be impacted by data biases.

In general, generative and discriminative approaches have complementary advantages and disadvantages, for example regarding temporal stability, recovery from failures, or dependence on large high-quality data corpora. Thus, the idea to combine these two paradigms is natural. Such *hybrid* approaches have been successfully explored in the context of hand tracking (Qian et al., 2014; Sharp et al., 2015; Sridhar et al., 2015a; Taylor et al., 2016; Ye et al., 2016). For example, they use machine learning components to initialize pose hypotheses in the optimization of the

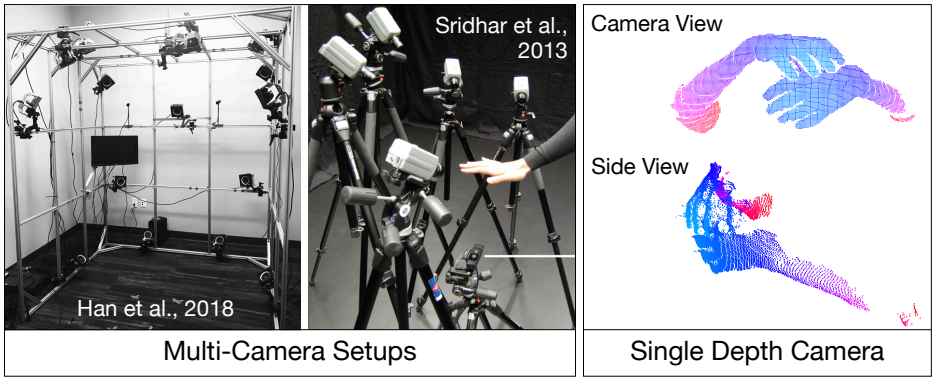


Figure 2.2: Left: Multi-camera setups are complex and inflexible (©The respective copyright owners). Right: A single depth camera can provide partial 3D information (color coding indicates distance from the camera).

generative model or the predicted information is directly integrated into the objective function.

2.2 INPUT MODALITIES

Earlier approaches or approaches solely focusing on high quality employed calibrated multi-camera setups to have more constraints for recovering the full 3D hand pose and to alleviate the challenge of strong self-occlusions (Ballan et al., 2012; Oikonomidis et al., 2011b; Sridhar et al., 2013). Most recent approaches focus on single-camera setups due to the complexity and inflexibility of calibrated multi-camera setups (see Figure 2.2, left). Since a depth image has several advantages over an RGB image, a single RGB-D or depth sensor is a popular choice (survey given by Supančič et al., 2018; Yuan et al., 2018). First, it provides partial 3D information (see Figure 2.2, right) whereas an RGB image contains scale and depth ambiguities. In addition, it is agnostic to lighting and hand appearance variation, making algorithms generalize more easily to unseen scenarios. However, depth sensors are more expensive and not ubiquitous in contrast to RGB cameras. Furthermore, they have a higher power consumption and might not work in outdoor scenes due to interference with the infrared radiation of the sun.

Comparably few methods have focused on hand reconstruction from monocular RGB input. Some of the first methods for this problem did not produce metrically accurate 3D pose as they only fetched the nearest 3D neighbor for a given input or assume that the z-coordinate is fixed (Heap and Hogg, 1996; Romero et al., 2010; Stenger et al., 2006). More recently, Simon et al., 2017 proposed an RGB-based method for hand joint

position regression. However, the method only estimates 2D positions from a monocular image and again requires multi-view triangulation to obtain 3D results. Panteleris and Argyros, 2017 proposed to use a short-baseline stereo RGB camera for hand pose estimation without the need for a disparity map. Nevertheless, stereo cameras are not readily available to everyday users. Zimmermann and Brox, 2017 proposed a learning-based method to predict the 3D hand pose from monocular data. However, their 3D joint predictions are *relative* to a canonical frame, i.e., the absolute coordinates are unknown and any global motion of the hand relative to the camera is lost. Furthermore, their method is not able to distinguish 3D poses with the same 2D joint position projection since their 3D predictions are merely based on the abstract 2D heatmaps and do not directly take the image into account. In contrast, this thesis proposes the first method for real-time *full 3D* hand pose estimation from monocular RGB input (Chapter 6). By jointly learning 2D and 3D joint positions from image evidence, the method is able to correctly estimate poses with ambiguous 2D joint positions. Since the input is unconstrained monocular RGB video, it can directly be employed on community videos, e.g., from YouTube. The method proposed in this thesis, together with other pioneering works, have spurred significant interest in monocular 3D hand reconstruction. This has led to many new works tackling single hand reconstruction from monocular RGB (Baek et al., 2019; Boukhayma et al., 2019; Cai et al., 2018; Iqbal et al., 2018; Panteleris et al., 2018; Spurr et al., 2018; Yang et al., 2019; Zhang et al., 2019), or even hand and object reconstruction (Hasson et al., 2019; Tekin et al., 2019).

2.3 HANDS IN INTERACTION

Whereas most of the aforementioned approaches track a single hand in isolation, there has been much less research on how to reconstruct hands in interaction, i.e., in cluttered environments and while interacting with objects or a second hand. This is due to additional challenges such as segmentation of the hands and severe occlusions, which made researchers focus on simplified scenarios first.

HANDS AND OBJECTS. Some works estimate the pose of one interacting hand without simultaneously tracking the object. They employ large databases (Romero et al., 2010), part-based trackers (Hamer et al., 2009) or formulate the pose estimation as classification problem with pose classes (Rogez et al., 2014). On the one hand they do not need to optimize more parameters than for a single hand, on the other hand they cannot exploit mutual constraints between hands and the manip-

ulated object which could provide valuable information. Methods that incorporate such constraints often use computationally expensive physics simulation (Tzionas et al., 2016) or multiple calibrated viewpoints (Ballan et al., 2012; Oikonomidis et al., 2011b). Together with the additional number of model parameters that need to be estimated for the object this leads to slow offline runtimes of the aforementioned approaches. As in single hand tracking, more recent work is leaning towards the use of a single RGB-D camera (Kyriazis and Argyros, 2014; Tzionas et al., 2014, 2016) to yield a more flexible and mobile setup. This thesis proposed a novel method for real-time 3D hand tracking under strong occlusions in cluttered environments which compares favorably to previous works (Chapter 5).

TWO INTERACTING HANDS. Some methods try to overcome the challenges of two interacting hands, namely the inherent problem to distinguish the two hands and the more severe occlusions, by using marker gloves (Han et al., 2018) or multi-view setups (Ballan et al., 2012). Other approaches tackle the problem from a single RGB-D camera to achieve more flexibility and practical usability. An analysis-by-synthesis approach is employed by Oikonomidis et al., 2012 who minimize the discrepancy of a rendered depth image and the input using particle swarm optimization. Kyriazis and Argyros, 2014 apply an ensemble of independent trackers, where the per-object trackers broadcast their state to resolve collisions. Tzionas et al., 2016 use discriminatively detected salient points and a collision term based on distance fields to obtain an intersection-free model fit. Nevertheless, the aforementioned single-camera methods do not achieve real-time rates, and operate at 0.2 to 4 frames per second. There exist some methods that track two hands in real time, albeit without being able to deal with close hand-hand interactions. Taylor et al., 2016 jointly optimize pose and correspondences of a subdivision surface model but the method fails when the hands come close together, making it unusable for capturing any hand-hand interaction. Taylor et al., 2017 employ machine learning techniques for hand segmentation and palm orientation initialization, and subsequently fit an articulated distance function. They use a custom-built high frame-rate depth camera to minimize the motion between frames, thus being able to fit the model with very few optimizer steps. However, they do not resolve collisions and they do not estimate hand shape, so that they require a given model for every user. While they show some examples of hand-hand interactions, they do not show very close and elaborate interactions, e.g., with tight grasps. In contrast to previous two-hand tracking solutions, this thesis proposes an approach that (i) runs in real time with a commodity camera, (ii) is marker-less,

(iii) uses a single (depth) camera only, (iv) handles hand collisions, and (v) automatically adjusts to the user's hand shape (Chapter 7).

3 PREREQUISITES

This chapter introduces the parametric hand models that are used throughout the thesis. First, the anatomical structure of the hand joints, as depicted in Figure 3.1, is modeled using a kinematic skeleton (Section 3.1). The kinematic hand skeleton is then further extended with different hand surface representations, namely the Sum of Gaussians formulation (Section 3.2) and the surface mesh formulation (Section 3.3).

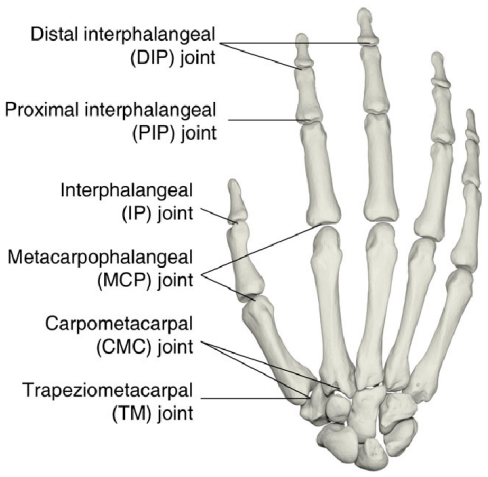


Figure 3.1: The anatomical joints of the hand. (Figure from Bullock et al., 2012)

3.1 KINEMATIC SKELETONS

In general, a kinematic skeleton is a hierarchy of rigid transforms $\in \mathrm{SE}(3)$ which can be used to model articulated motion. The hierarchy is a tree structure, i.e., all transforms have exactly one parent transform except for one root transform. The transforms are usually represented locally. They specify how a point in the local coordinate system of the child transform can be mapped to the local coordinate system of the parent transform. Hence, a mapping from the local coordinate system of transform i to the global coordinate system, or world coordinate system, can be computed by iteratively multiplying the local transforms along the path from i to the root in the hierarchy

$$T_i^g = \left(\prod_{j \in \mathrm{anc}(i)} T_j^l \right) \cdot T_i^l \,. \tag{3.1}$$

Here, $\mathrm{anc}(i)$ is the list of ancestors of transform i, ordered from $\mathrm{parent}(i)$ to the root. In the following, T_i^g is referred to as *local-to-global transform* i whereas T_i^l is denoted as *local transform* i.

For the skeleton of a hand, the transforms correspond to the joints and the hierarchy is approximately given by the bones as shown in Figure 3.2.

The goal of hand pose estimation with a kinematic skeleton is to estimate the parameters of all transforms in the kinematic hand skeleton. The root transform contains the global translation and rotation of the hand. For all other transforms, the translations are described by the bone lengths and might be given depending on the availability of a personalized hand model. For the rotational part of the rigid transforms, a full 3D rotation $\in \mathrm{SO}(3)$ could be estimated. However, the joints in the hand anatomically exhibit fewer degrees of freedom (DOF) as shown in Figure 3.2. Using the full 3 DOF per joint results in a highly over-parameterized model that needs stronger regularization for successfully optimizing the hand pose. For example, Romero et al., 2017 employ a low-dimensional subspace representation for the pose obtained by linear dimensionality reduction via principal component analysis (PCA) (Jolliffe, 1986). A more compactly parameterized and hence popular parameterization for the local rotations in a kinematic hand skeleton is the axis-angle formulation. A corresponding rotation axis is defined for each DOF in the hand according to the hand anatomy. Subsequently, only a single rotation angle needs to be estimated per DOF. A local rotation matrix can be calculated per DOF using the respective rotation axis and angle. The local rotation matrices of all DOF that belong to the same joint i are then concatenated to obtain the rotational part of the rigid transform T_i^l. The parameter vector of a kinematic hand model, consisting of global translation and rotation as well as joint rotation angles, is denoted as $\boldsymbol{\theta} \in \mathbb{R}^{26}$.

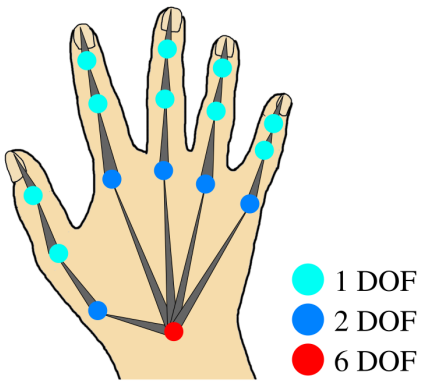


Figure 3.2: The kinematic skeleton hand model. The root transform is located at the wrist and has 6 degrees of freedom (DOF) which correspond to the global position and rotation of the hand. Every other transform corresponds to a rotational joint in the hand and either has 1 or 2 DOF (assuming fixed bone lengths).

The keypoint locations in a kinematic hand skeleton, e.g., joint positions or fingertip positions, depend on the pose parameters $\boldsymbol{\theta}$. They can be calculated using the local-to-global transforms of the joints. The (homo-

geneous) 3D position of the j-th joint of model $\mathcal{M}$ in global coordinates is given as

$$\mathcal{M}(\boldsymbol{\theta})_j = T_j^g(\boldsymbol{\theta}) \cdot \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = (\, T_j^g(\boldsymbol{\theta}) \,)_{*,4} \,, \tag{3.2}$$

since the joint is the origin in its own local coordinate system. Analogously, the fingertip position of finger f can be formulated as

$$\mathcal{M}(\boldsymbol{\theta})_f = T^g_{\text{parent}(f)}(\boldsymbol{\theta}) \cdot \text{bone}_f \,, \tag{3.3}$$

where bone_f is the local bone vector connecting the fingertip to its parent joint in the parent's coordinate system.

While the kinematic skeleton hand model can be used to model the anatomical bone structure and motion of a hand, it does not model the surface or volume of the hand. However, when images or a video are provided for hand pose estimation, the hand surface is what is actually visible in the observation. Hence, it is crucial for generative hand models to describe the hand surface or volume in order to compare the current model parameter hypothesis to the input.

3.2 SUM OF GAUSSIANS MODEL

The Sum of Gaussians (SoG) model extends the kinematic skeleton model of the hand with a collection of 3-dimensional Gaussian density functions. It was originally proposed for full-body pose estimation (Stoll et al., 2011) and subsequently adapted for hand tracking (Sridhar et al., 2013, 2014, 2016). The Gaussians are rigidly attached to the bones of the kinematic model, i.e. their position depends on the hand model parameters θ and can be calculated using the local-to-global transform of their parent joint. Their standard deviation σ is set such that the isosurface at 1σ coincides with the surface of the hand (see Figure 3.3 (a)). Whereas the surface approximation is coarser than a surface mesh (see Section 3.3), the SoG model offers high computational efficiency due to the low number of primitives and does not require a separate step for explicit computation of correspondences before model fitting since the Gaussian functions have infinite spatial extent.

To fit the SoG model to images, the following procedure is commonly used:

1. Image regions of similar color or depth are clustered by quadtree clustering as demonstrated in Figure 3.3 (c).

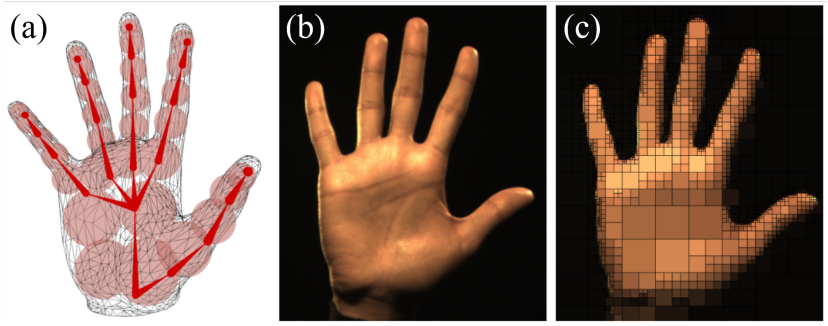


Figure 3.3: (a) The Sum of Gaussians (SoG) hand model. (b,c) Quadtree clustering of an input image. (Figure from Sridhar et al., 2014)

2. A 2D or 3D Gaussian, depending on the input modality, is created for each leaf of the quadtree. The standard deviation is set such that the 1σ isosurface circle or sphere fits the quad. This yields a Sum of Gaussians representation for the input.
3. The similarity of the input and model SoG is then computed as their overlap and can be maximized.

The overlap of two d-dimensional Sums of Gaussians $\{G_p\}_{p=1}^{N_p}, \{G_q\}_{q=1}^{N_q}$ is formulated as

$$\sum_{p=1}^{N_p} \sum_{q=1}^{N_q} \int_{\mathbb{R}^d} G_p(x) \cdot G_q(x)\, dx\,. \tag{3.4}$$

As described by Sridhar et al., 2014, an integral over a product of two un-normalized isotropic Gaussians $G_p(x; \mu_p, \sigma_p)$ and $G_q(x; \mu_q, \sigma_q)$ of dimension d is given as

$$\begin{aligned} &\int_{\mathbb{R}^d} G_p(x; \mu_p, \sigma_p) \cdot G_q(x; \mu_q, \sigma_q)\, dx \\ =&\frac{\sqrt{(2\pi)^d (\sigma_p^2 \sigma_q^2)^d}}{\sqrt{(\sigma_p^2 + \sigma_q^2)^d}} \exp\left(-\frac{||\boldsymbol{\mu}_p - \boldsymbol{\mu}_q||_2^2}{2(\sigma_p^2 + \sigma_q^2)} \right). \end{aligned} \tag{3.5}$$

This term is differentiable with respect to $\boldsymbol{\mu}$ and σ. Furthermore, the derivative $\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\theta}}$ is given by the transforms of the joints. If a hand model, in addition to pose parameters $\boldsymbol{\theta}$, has shape parameters $\boldsymbol{\beta}$, e.g., specifying bone lengths or hand thickness, the derivatives $\frac{\partial \boldsymbol{\mu}}{\partial \boldsymbol{\beta}}$, $\frac{\partial \sigma}{\partial \boldsymbol{\beta}}$ can also be calculated. These analytical derivatives allow fast fitting of the SoG model in an optimization-based framework.

Note that the computation of the overlap considers all possible pairs of Gaussians, since they have infinite spatial extent, and thus does not require any explicit correspondence search like mesh-based models. Furthermore, the SoG model offers a smooth and differentiable way to avoid

collisions within a hand model or between multiple models. By penalizing the overlap of the SoG model with itself, i.e., considering all pairs of Gaussians within the SoG hand model, intersections can be effectively resolved. It should be emphasized that this way to avoid collisions in model fitting is continuous and significantly more efficient compared to binary intersection tests for hand mesh models.

3.3 HAND MESH MODEL

In contrast to the Sum of Gaussians model, hand mesh models describe the hand surface explicitly using a surface mesh. A surface mesh is a piecewise-linear approximation of a 3D surface using a collection of connected surface primitives like triangles or quadrangles. While they usually exhibit a higher computational cost, e.g., compared to the SoG model, they model the hand surface with more detail. Some methods use a personalized hand mesh obtained from a laser scanner (Ballan et al., 2012; Tzionas et al., 2016), which is not readily available to everyday users. Hence, a parametric hand model, which is called the MANO model, was published by Romero et al., 2017.

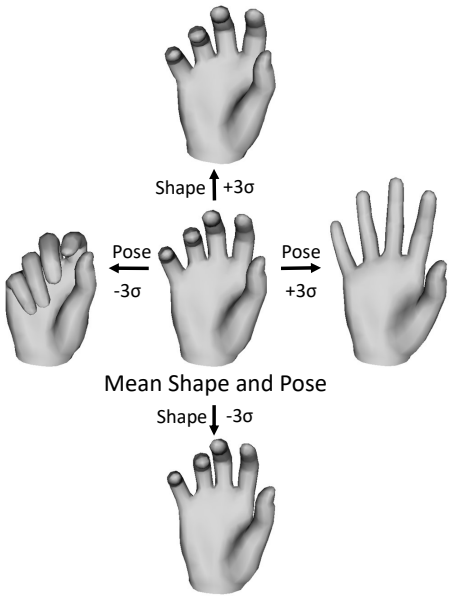


Figure 3.4: Illustration of the parametric MANO hand shape and pose space.

It was built from about 1000 3D hand scans of 31 persons in wide range of different hand poses. The parametric hand shape and pose space of MANO was obtained by fitting a template hand mesh to all scans and subsequently performing Principal Component Analysis (PCA) (see Figure 3.4). The hand surface is represented by a 3D mesh with vertices $\mathcal{V}$, where $N_V := |\mathcal{V}| = 778$. The MANO model defines a function $\mathbf{v} : \mathbb{R}^{N_S} \times \mathbb{R}^{N_P} \to \mathbb{R}^{3N_V}$, that computes the 3D positions of all of the mesh's N_V vertices, given a shape parameter vector $\boldsymbol{\beta} \in \mathbb{R}^{N_S}$ and pose parameter vector $\boldsymbol{\theta} \in \mathbb{R}^{N_P}$, with $N_S = 10$ and $N_P = 51 = 45 + 6$ (for $\boldsymbol{\theta}$ including global translation and rotation).

The function $\mathbf{v}$ is formulated as

$$\mathbf{v}(\boldsymbol{\beta}, \boldsymbol{\theta}) = \mathrm{LBS}(M(\boldsymbol{\beta}, \boldsymbol{\theta}), J(\boldsymbol{\beta}), \mathbf{W}) \,. \tag{3.6}$$

Here, $M(\cdot)$ is a parametric hand template in rest pose, $J(\cdot)$ computes the 3D position of the hand joints, and $\mathbf{W}$ are the skinning weights used by the linear blend skinning function LBS (Alias | Wavefront, 1998; Lewis et al., 2000; Softimage, 1992). Note that the parametric hand template $M(\boldsymbol{\beta}, \boldsymbol{\theta}) = T + S(\boldsymbol{\beta}) + C(\boldsymbol{\theta})$ consists of a fixed template mesh T, identity-specific shape offsets $S(\cdot)$, and pose-dependent corrective offsets $C(\cdot)$. The pose-dependent correctives are used to reduce skinning artifacts, please refer to Romero et al., 2017 for further details.

The shape parameters $\boldsymbol{\beta} \in \mathbb{R}^{N_S}$ and the pose parameters excluding the global rigid transform $\boldsymbol{\theta} \in \mathbb{R}^{N_{P'}}$ are coefficients of the low-dimensional shape and pose spaces that were obtained by performing PCA. The full dimensionality is given as $N_S = 10$ and $N_{P'} = 45$ but any subset of the first x components could be used to obtain less fine-scale but more regularized results. Note that the dimensionality of the MANO pose space (without global translation and rotation) is significantly higher than for the axis-angle-based kinematic skeleton (45 vs. 20, see Section 3.1). This is the case since the MANO model uses the full 3 DOF for each joint irrespective of the anatomically plausible number of DOF for each hand joint. For example, the DIP and PIP joints of the fingers (see Figure 3.1) can be well approximated by a single DOF since they only allow flexion–extension motion. Hence, the use of pose regularizers is inevitable, at least when using all 45 pose PCA components, to make the model fitting problem less ill-posed. Fortunately, since the MANO model was built using PCA, it naturally allows for a statistical regularization by simply imposing that the parameters are close to zero, which corresponds to a Tikhonov regularizer.

4

DATASETS

Data is key for training accurate and robust machine learning systems. To enable generalizability of the system to unseen test cases, the data should capture diverse and challenging scenes. For training supervised methods, the data requires annotations which are often hard to obtain. For example, for manual annotation of keypoint locations in images, an annotator needs to look at every single image and mark all points. This is time-consuming and manual annotations are always noisy. For some tasks, manual annotation might even be impossible (see Section 4.4).

This thesis explores smart and novel ways to generate annotated training data for various tasks related to hand reconstruction. Examples for how this data is used in full 3D hand reconstruction systems are given in the following chapters. The contributions of this chapter can be summarized as:

- A data generation framework for synthesizing an extensive RGB-D dataset, *SynthHands*, with full 3D annotation of 21 hand keypoints. Natural hand–object interactions are captured using a novel merged reality setup (Section 4.1, published as part of Mueller et al., 2017, prerequisite for the method presented in Chapter 5).
- A novel geometrically consistent GAN that performs image-to-image translation while preserving poses during translation. Based on this network, the RGB images of the *SynthHands* dataset are enhanced such that the statistical distribution resembles real-world hand images. The resulting *GANerated Hands* dataset overcomes existing datasets in terms of size (>260k frames), image fidelity, and annotation precision (Section 4.2, published as part of Mueller et al., 2018, prerequisite for the method presented in Chapter 6).
- A depth-based dataset for per-pixel left/right segmentation as well as a novel per-pixel hand part dataset which is automatically annotated using painted hands and a calibrated RGB camera (Section 4.3, published as parts of Mueller et al., 2019 and Soliman et al., 2018, prerequisites for the methods presented in Chapter 7 and Chapter 8, respectively).
- The *DenseHands* dataset which includes both pose and dense shape annotations for two interacting hands. Live user-driven physical

simulation is leveraged to create natural two-hand motions without the need of a robust two-hand tracking system (Section 4.4, published as part of Mueller et al., 2019, prerequisite for the method presented in Chapter 7).

4.1 SYNTHHANDS DATASET

Supervised learning methods, including CNNs, require large amounts of training data in order to learn all the variation exhibited in real hand motion. Fully annotated real data would be ideal for this purpose but it is time consuming to manually annotate data and annotation quality may be unreliable (Oberweger et al., 2016). Automatic annotation based on markers is also unsuitable since it changes the appearance of the hand. To circumvent these problems, existing methods (Rogez et al., 2014, 2015) have used synthetic data. Despite the advances made, existing datasets are constrained in a number of ways: they typically show unnatural mid-air motions, no complex hand-object interactions, and do not model realistic background clutter and noise.

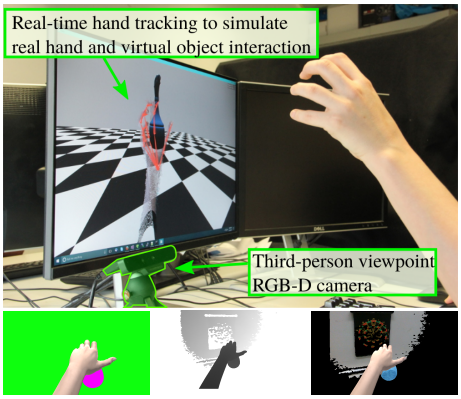


Figure 4.1: The *SynthHands* dataset has accurate annotated data of a hand interacting with objects. A merged reality framework is used to track a real hand, where all joint positions are annotated, interacting with a virtual object (top). Synthetic images are rendered with chroma key-ready colors, enabling data augmentation by composing the rendered hand with varying object texture and real cluttered backgrounds (bottom).

This thesis proposes a new dataset, *SynthHands*, with full 3D annotations for 21 keypoints in the hand, namely the 5 fingertips and all 16 joints (as defined in Section 3.1). The dataset combines real captured hand motion (retargeted to a virtual hand model) with natural backgrounds and virtual objects to sample all important dimensions of variability at previously unseen granularity. It captures the variations in natural hand motion such as pose, skin color, shape, texture, background clutter, camera viewpoint, and hand-object interactions. The *SynthHands* dataset has unique features that make it well suited for supervised training of learning-based methods.

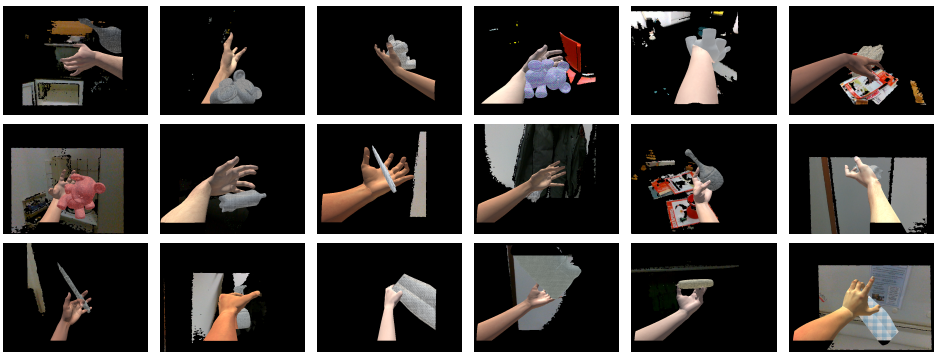


Figure 4.2: An example set of random samples taken from the *SynthHands* dataset. See Table 4.1 for description of dataset variability.

NATURAL HAND MOTIONS. Instead of using static hand poses (Rogez et al., 2015), real, non-occluded, hand motion was captured in mid-air from a third-person viewpoint, with a state-of-the-art real-time markerless tracker (Sridhar et al., 2015a). These motions were subsequently re-targeted onto a photorealistic synthetic hand rigged by an artist. Because the synthetic hand is posed using the captured hand motion, it mimics real hand motions and increases dataset realism.

HAND SHAPE AND COLOR. Hand shape and skin color exhibit large variation across users. To simulate real world diversity, *SynthHands* uses skin textures randomly sampled from 12 different skin tones. Furthermore, it contains variation in other anatomical features (e.g., male hands are typically bigger and may contain more hair). Finally, hand shape variation is modeled by randomly applying a scaling parameter $\beta \in [0.8, 1.2]$ along each dimension of a default hand mesh.

ARBITRARY VIEWPOINT. Synthetic data has the unique advantage that it can be rendered from arbitrary camera viewpoints. In order to support difficult egocentric views, five virtual cameras that mimic different egocentric perspectives are used in addition to five different third-person viewpoints. The virtual cameras generate RGB-D images while also simulating sensor noise and camera calibration parameters.

HAND-OBJECT INTERACTIONS. Hand-object interactions are realistically simulated by using a merged reality approach to track real hand motion interacting with virtual objects. This is achieved by leveraging the real-time capability of existing hand tracking solutions (Sridhar et al., 2015a) to show the user's hand interacting with a virtual on-screen object.

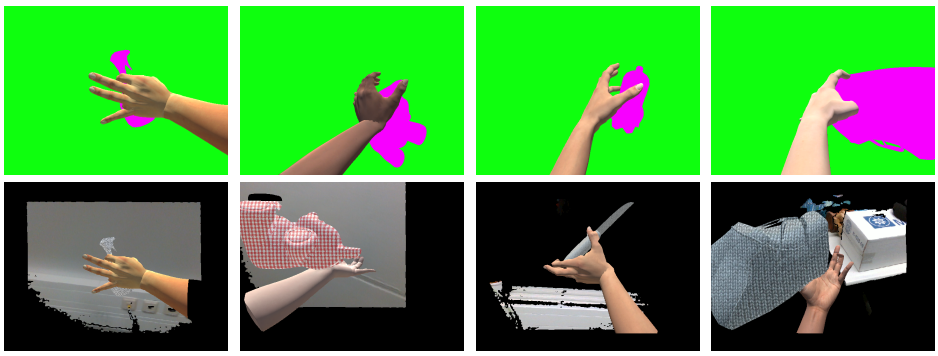


Figure 4.3: The *SynthHands* dataset is created by posing a detailed 3D hand model with real hand motion data. Virtual objects are incorporated into the 3D scenario. To allow data augmentation, scene background and object foreground are rendered as constant plain colors (top row) while the shading information for the object is output in a separate image. In a post-processing step, randomized textures are chosen as object appearance and then composed with the shading details (bottom row). Furthermore, real RGB-D backgrounds are used for background augmentation.

Users perform motions such as object grasping and manipulation, thus simulating real hand-object interactions (see Figure 4.1).

OBJECT SHAPE AND APPEARANCE. *SynthHands* contains interactions with a total of seven different virtual objects in various locations, rotations and scale configurations. To enable augmentation of the object appearance to increase dataset variance, the object albedo (i.e., pink in Figure 4.3) and shading layers are rendered separately. Chroma keying is used to replace the pink object albedo with a texture randomly sampled from a set of 145 textures and combining it with the shading image. Figure 4.3 shows some examples of the data before and after augmentation.

REAL BACKGROUNDS. Finally, cluttered scenes and backgrounds are simulated by compositing the synthesized hand-object images with real RGB-D captures of real backgrounds, including everyday desktop scenarios, offices, corridors and kitchens. Chroma keying is used to replace the default background (green in Figure 4.3) with the captured backgrounds.

The proposed data generation framework is built using the Unity Game Engine (Unity, 2005) and uses a rigged hand model distributed by Leap-Motion, 2016. In total, *SynthHands* contains roughly 220,000 RGB-D images exhibiting large variation seen in natural hands and interactions. Table 4.1 shows the modes of variation in the *SynthHands* dataset. Representative frames are shown in Figure 4.2.

Table 4.1: Details about the variations explored in the *SynthHands* dataset.

Mode of Variation	Amount of Variation
Pose	63,530 frames of real hand motion, sampled every 5th frame
Wrist+Arm Rotation	wrist: sampled from a 70 deg. range arm: sampled from a 180 deg. range
Shape	x, y, z scale sampled uniformly in [0.8, 1.2]; female + male mesh
Skin Color	2 x 6 hand textures (female/male)
Camera Viewpoints	5 egocentric + 5 third-person
Object Shapes	7 objects
Object Textures	145 textures
Background Clutter	10,000 real images, uniform random u,v offset in [-100, 100]

4.2 ENHANCING SYNTHETIC DATA

While the main advantage of synthetic images is that the ground-truth 3D joint positions are known, a significant disadvantage is that they usually lack realism. Such discrepancy between real and synthetic images limits the generalization ability of a CNN trained only on the latter. The problem is more severe when considering RGB images instead of depth images. Rendering realistic depth images is easier since many variations, such as lighting or appearance, are not present. Therefore, synthesizing realistic RGB images of hands is significantly harder.

4.2.1 *Related Techniques*

Techniques like domain adaptation (Ganin and Lempitsky, 2015; Peng and Saenko, 2018; Tzeng et al., 2017) aim to bridge the gap between real and synthetic data by learning features that are invariant to the underlying differences. Other techniques use real–synthetic image pairs (Chen and Koltun, 2017; Isola et al., 2017; Sangkloy et al., 2017) to train networks that can generate images that contain many features of real images. Because it is hard to obtain real–synthetic image pairs, Shrivas-

tava et al., 2017 recently proposed a synthetic-to-real refinement network requiring only *unpaired* examples. However, the extent of refinement is limited due to pixel-wise similarity constraints to the input. In contrast, the unpaired image-to-image translation work of Zhu et al., 2017 relaxes these constraints to finding a bijection between the two domains.

4.2.2 *Proposed Dataset GANerated Hands*

In order to account for the disparity between synthetic and real images, this thesis proposes an image-to-image translation network, the *GeoConGAN*, with the objective to translate synthetic to real images. Most importantly, this network only requires *unpaired* real and synthetic images for training, and employs a novel geometric consistency constraint to ensure valid annotation transfer. Note that for both the real and the synthetic data only foreground-segmented images that contain a hand on white background are used. This facilitates training and focuses the network capacity on the hand region.

REAL HAND IMAGE ACQUISITION. A green-screen setup is used to capture real hand images with varying poses and camera extrinsics from 7 different subjects with different skin tones and hand shapes. In total, 28,903 real hand images were captured using a desktop webcam with image resolution 640×480 pixels.

SYNTHETIC HAND IMAGE GENERATION. The synthetic hand image dataset is based on the *SynthHands* dataset. For each image, a version without the object is rendered to facilitate the task for the *GeoConGAN*. Furthermore, the background is set to plain white.

GEOMETRICALLY CONSISTENT CYCLEGAN (GEOCONGAN). Training a hand joint regression network based on synthetic images alone has the strong disadvantage that the so-trained network has limited generalization to real images (see Section 6.5.1 for the corresponding experiment).

To tackle this problem, this thesis proposes a network that translates synthetic images to "real" (or *GANerated*) images. The translation network is based on CycleGAN (Zhu et al., 2017), which uses adversarial discriminators (Goodfellow et al., 2014) to simultaneously learn cycle-consistent forward and backward mappings. Cycle-consistency means that the composition of both mappings (in either direction) is the identity mapping. In particular, mappings from synthetic to real images (*synth2real*), and from real to synthetic images (*real2synth*) are learned. In contrast to many existing image-to-image or style transfer networks (Isola et al., 2017; Sangkloy

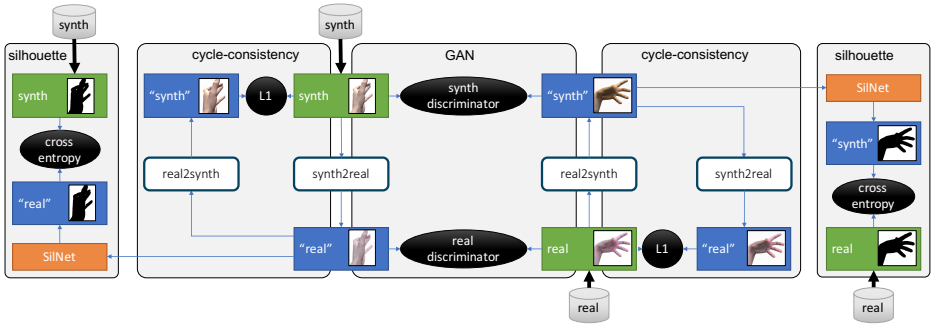


Figure 4.4: Network architecture of the *GeoConGAN*. The trainable part comprises the *real2synth* and the *synth2real* components, where both components are shown twice for visualization purposes. The loss functions are shown in black, images from the database in green boxes, images generated by the networks in blue boxes, and the existing *SilNet* in orange boxes.

et al., 2017), CycleGAN has the advantage that it does not require paired images, i.e., there does not need to exist a real image counterpart for a given synthetic image, which is crucial for the purpose chosen in this thesis due to the unavailability of such pairs.

The architecture of the proposed *GeoConGAN* is illustrated in Figure 4.4. The input to this network are (cropped) synthetic and real images of the hand on a white background in conjunction with their respective silhouettes, i.e., foreground segmentation masks. In its core, the *GeoConGAN* resembles CycleGAN (Zhu et al., 2017) with its discriminator and cycle-consistency loss, as well as the two trainable translators *synth2real* and *real2synth*. However, unlike CycleGAN, an additional geometric consistency loss *GeoCon* is incorporated that ensures that the *real2synth* and *synth2real* components produce images that *maintain the hand pose* during image translation. In particular, the geometric consistency loss enforces the silhouette of the generated image $f(X)$ to be the same as the silhouette of the original real or synthetic image X using the cross-entropy (CE) classification loss

$$GeoCon(X) = \mathrm{CE}(sil(X), sil(f(X))) \ , \ f \in \{real2synth, synth2real\} \ . \tag{4.1}$$

Enforcing consistent hand poses is of utmost importance in order to ensure that the ground-truth joint locations of the synthetic images are also valid for the "real" images produced by *synth2real*. Figure 4.5 shows the benefits of adding this new loss term.

The silhouettes of the original synthetic or real images are obtained from either the synthetic rendering or green-screen segmentation. In order to extract the silhouettes of the images that are produced by both *real2synth* and *synth2real* (blue boxes in Figure 4.4), a binary classification

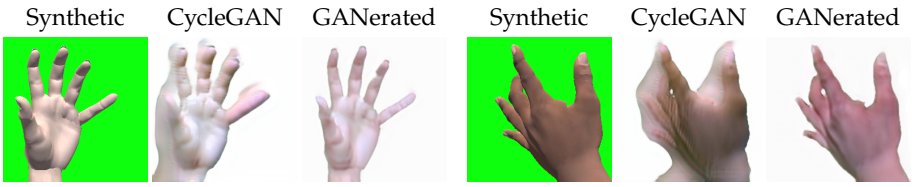


Figure 4.5: The *GeoConGAN* translates from synthetic to real images by using an additional geometric consistency loss.

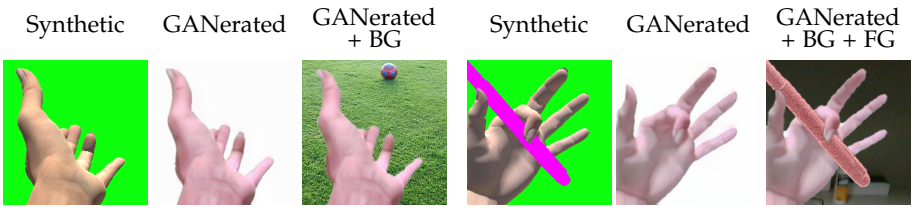


Figure 4.6: Two examples of synthetic images with background/object masks in green/pink, GANerated images, and GANerated images with background (and foreground) augmentation.

network, the *SilNet*, is used. It is based on a simple UNet (Ronneberger et al., 2015) that has three 2-strided convolutions and three deconvolutions. Note that this is a relatively easy task as the images have white background. However, choosing a *differentiable* network over naïve thresholding makes the training of *GeoConGAN* more well-behaved. The *SilNet* is trained beforehand on a small disjoint subset of the data and is fixed while training *synth2real* and *real2synth*. Details can be found in Appendix A.1.

DATA AUGMENTATION. Once the *GeoConGAN* is trained, all synthetically generated images are fed into the *synth2real* component to obtain the *GANerated Hands* dataset, a dataset of "real" images that have associated ground truth 3D joint locations.

By using the background masks from the original synthetic images, background augmentation can be performed by compositing GANerated images (foreground) with random images (background) (Mehta et al., 2017b; Rhodin et al., 2016; Varol et al., 2017). Similarly, the data can be augmented with a randomly textured object by leveraging the object masks produced when rendering the original synthetic sequences (see Section 4.1). Training on images without background or objects and hence employing data augmentation as post processing significantly eases the task for the *GeoConGAN*. Figure 4.6 shows some GANerated images.

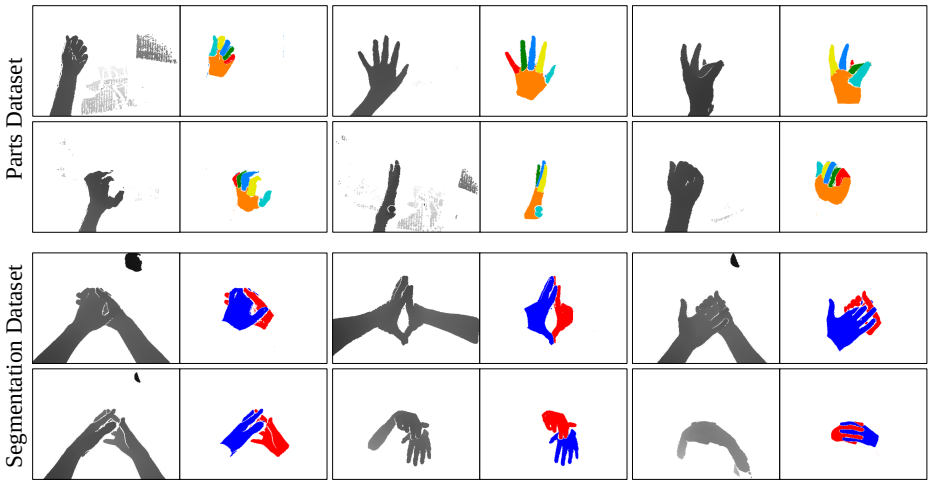


Figure 4.7: Example depth frames and automatically generated labels from the proposed paint-based hand part classification (top) and hand segmentation (bottom) datasets.

4.3 HAND SEGMENTATION AND PART CLASSIFICATION

The previous two sections have focused on obtaining data with 3D keypoint annotations that are key to develop the hand pose estimation methods presented in Chapters 5 and 6. A different, but similarly important, task that often precedes hand pose estimation is hand segmentation or part classification. Given an input image, the desired output is a segmentation mask which specifies for each pixel which hand or hand part is visible. This thesis explores creation and automatic annotation of depth-based segmentation datasets for different use cases.

The automatic annotation is based on simultaneous recording with a calibrated depth and RGB camera. Then, the hands or hand parts can be color-coded using body paint or colored gloves. By exploiting the RGB image stream in combination with the calibration information, the pixels in the depth image can be automatically annotated using color thresholding. More formally, let $I_c, I_d \in \mathbb{R}^{3\times3}$ be the intrinsic camera matrices for the RGB and depth camera, respectively, and let $E_{d2c} \in \mathbb{R}^{3\times4}$ be the extrinsic matrix transforming from the depth camera coordinate system to the RGB camera coordinate system. For each pixel (u, v) in the depth image $\mathcal{D}$, a color value can be assigned based on the corresponding pixel in the RGB image $\mathcal{C}$, resulting in a colored depth image $\mathcal{B}$. In the following calculation, the conversions of vectors to and from homogeneous coordinates are usually dropped for brevity. They can be

inferred from the matrix dimensions. First, the pixel in the depth image is back-projected to a ray in the 3D depth camera coordinate system

$$\text{ray}(u,v) = I_d^{-1} \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} . \tag{4.2}$$

Since the depth of the pixel in camera space is known, the corresponding 3D point $\boldsymbol{p}^{u,v}$ can be determined by scaling the back-projected ray such that its z coordinate corresponds to the depth

$$\Pi_d^{-1}(u,v) = \mathcal{D}(u,v) \cdot \text{unit}_z(\text{ray}(u,v)) , \tag{4.3}$$

where the operator $unit_z$ scales a vector such that its z coordinate is 1. The obtained 3D point $\boldsymbol{p}^{u,v}$ is then tranformed to the coordinate system of the RGB camera and projected to the color image plane

$$\Pi_c(\boldsymbol{p}^{u,v}) = \text{unit}_z(I_c \cdot E_{d2c} \cdot \boldsymbol{p}^{u,v}) . \tag{4.4}$$

Hence, the colored depth image $\mathcal{B}$ is computed as

$$\mathcal{B}(u,v) = \mathcal{C}(\Pi_c(\Pi_d^{-1}(u,v))) . \tag{4.5}$$

The classification label for pixel (u,v) is then obtained by thresholding $\mathcal{B}(u,v)$ in HSV color space. The HSV color space has the advantage that it exhibits a better separation of shading effects in comparison to the RGB color space. The hands or hand parts of the user can either be colored using body paint or colored gloves. Both choices have unique advantages and disadvantages. Gloves can be re-used for different users, however the fit might not be tight, leading to distorted hand shapes in the depth image or sliding of the hand part boundaries. Body paint is more time-consuming to apply but it does preserve the original shape and supports more accurate boundaries for hand parts.

During the course of this thesis, two paint-based segmentation datasets were recorded and successfully employed for hand tracking tasks. The first one focuses on left/right hand segmentation for tracking of close two-hand interactions (used in Chapter 7). The second one contains hand part labels and enables hand tracking for recognizing various microgestures (used in Chapter 8). The paint-based capture setup is shown in Figure 4.8. Please see Figure 4.7 for example frames from both datasets and Table 4.2 for more details.

In addition to the two datasets presented in this thesis, contributions were also made to a more large-scale glove-based two-hand segmentation dataset and a novel segmentation method using a lightweight neural network architecture (Bojja et al., 2019).

Figure 4.8: Setup (left) and recording (right) of the paint-based hand part classification dataset.

Table 4.2: Details about depth-based segmentation datasets.

Property	**Segmentation Dataset**	**Parts Dataset**
Task	two-hand interactions	single-hand microgestures
Annotation	3 classes: left, right, background	7 classes: one per finger, palm, background
Users	3 (1 female, 2 male)	5 (2 female, 3 male)
Viewpoints	egocentric (shoulder), frontal	egocentric (shoulder and head)
Frames	19,926	66,662

4.4 DENSEHANDS DATASET

While hand segmentation and part classification assign discrete labels to pixels, dense correspondence regression extends these concepts to continuous hand surface locations. Given an input image, the desired output is a dense correspondence image. For each pixel, it contains a k-dimensional feature vector that uniquely determines the 3D hand surface point that is visible at this pixel. These correspondences can greatly help in fitting a hand model to an input image since they are significantly more accurate and robust compared to naïvely using closest points as correspondence. The corresponding experiment will be presented in a later chapter of the thesis (Section 7.5.2).

4.4.1 *Dense Correspondence Encoding*

The dense correspondence encoding is the function $\eta : \mathcal{S} \rightarrow [0,1]^k$ that maps every 3D surface location of the hand model to a unique value in $[0,1]^k$. Dimensionality $k = 3$ is used in the thesis.

NAÏVE ENCODING. A naïve encoding can be obtained by placing the hand surface mesh into the 3D unit cube and using the 3D location directly for η. This leads to similar encoding values for spatially close locations with respect to Euclidean distance. For example, adjacent fingertips would have similar values hence leading to easy confusion. Instead of Euclidean distance, the geodesic distance on the surface should be used for constructing η.

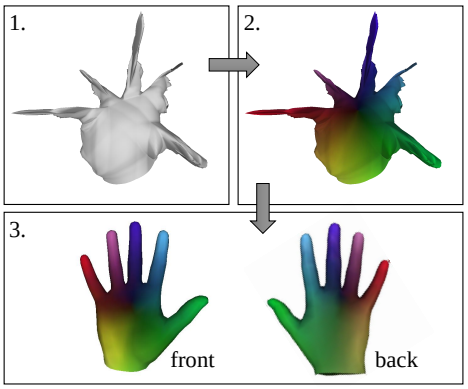


Figure 4.9: (1) The hand mesh is first transformed to a space where Euclidean and geodesic distances are approximately the same. (2) Then, this embedded mesh is colored using an HSV color cylinder. (3) The vertex colors are transferred back to the original hand mesh to be used as dense correspondence encoding. Notice that front and back color assignments differ in saturation, especially in the palm area.

GEODESICS-BASED ENCODING. Figure 4.9 shows the different steps to obtain a geodesics-based encoding. First, multi-dimensional scaling (Bronstein et al., 2006) is used to transform the 3D hand mesh into a space where Euclidean distance (approximately) resembles geodesic distance. Subsequently, an HSV color space cylinder is mapped onto the embedded hand mesh such that different hues are mapped onto different fingers. The resulting values are transferred to the original mesh to obtain the dense corresponding encoding η. Later in the thesis (Section 7.5.2), it is demonstrated that the proposed geodesic HSV embedding leads to improved results compared to the naïve embedding.

EXTENSION TO TWO HANDS. The dense correspondence encoding η can be easily extended to two hands by increasing the dimensionality to $k + 1$, where the last dimension indicates handedness (0: left hand, 0.5: right hand).

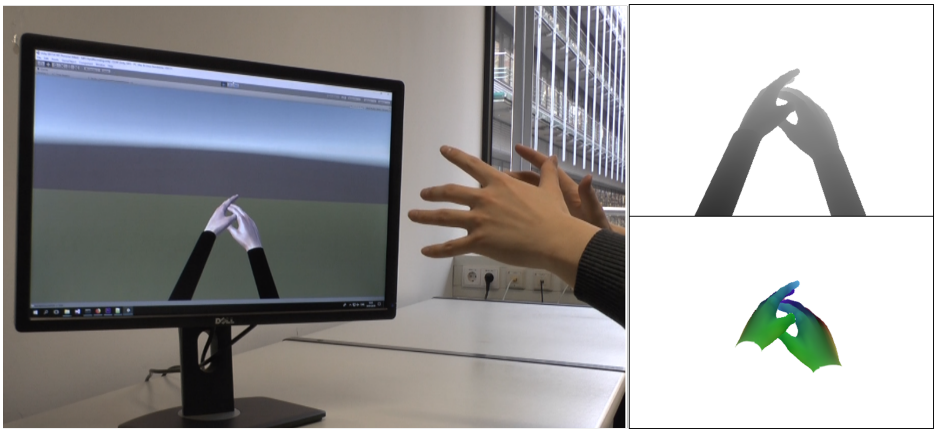


Figure 4.10: The synthetic *DenseHands* dataset is generated by tracking two hands, separated by a safety distance, which are used to control in real time a physically-based simulation of two interacting hands in the virtual scenario (left). The output are depth maps (top right) and dense surface annotations (bottom right) of the resulting simulation.

4.4.2 *Data Generation*

In the following, the generation of the *DenseHands* dataset is described. The goal is to provide dense correspondence annotation for two naturally interacting hands. Since this annotation cannot be obtained manually, the data is generated synthetically in a simulation system.

The simulation is driven by skeletal hand motion capture (mocap) data (LeapMotion, 2016) to maximize natural hand motion. However, existing hand motion capture solutions cannot robustly deal with close and complex hand-hand interactions. Hence, the actors move both hands at a safety distance in the real world while this offset is subtracted in the simulator. An extension of the work of Verschoor et al., 2018 is used for physics simulation. It enables simultaneous two hand simulation and guarantees plausible and intersection-free hand poses. Note that the proposed motion capture and simulation framework runs live such that the user can accomodate to driving the hand models which are virtually moved closer together. The actor receives immediate visual feedback and is thus able to simulate natural interactions. Figure 4.10 depicts a live session of this data generation step.

The hand simulator consists of an articulated skeleton surrounded by a low-resolution finite-element soft tissue model. The hands of the actor are tracked using LeapMotion, 2016, and the mocap skeletal configuration is linked through viscoelastic springs (a.k.a. PD controller) to the articulated skeleton of the hand simulator. In this way, the hand closely follows

the mocap input during free motion, yet it reacts to contact. The hand simulator resolves inter-hand collisions using a penalty-based frictional contact model, which provides smooth soft tissue interactions at minimal computational cost. The soft tissue layer is particularly helpful at allowing smooth and natural motions in highly constrained situations such as interlocking fingers. As the hands are commanded by the mocap input, their motion is inherently free of intra-hand collisions. However, the simulated inter-hand interaction may produce finger motions that lead to intra-hand collisions. These are usually small enough to be negligible and thus self-collision handling was avoided to maintain real-time interaction at all times.

After obtaining the simulated two-hand motion data driven by 5 actors, the scenes are re-rendered from different virtual viewpoints to create the *DenseHands* dataset with a total of 80,000 frames. The output of this data generation step is a depth image for each simulated frame as well as the dense correspondence image of the hand meshes colored with the encoding η. The fixed value $\mathbf{1}^4$ is used for non-hand pixels. Additionally, the generated depth images are post-processed to mimic typical structured-light sensor noise at depth discontinuities.

4.5 CONCLUSION

After having introduced diverse hand datasets and smart ways to automatically generate annotations in this chapter, the following chapters present various real-time 3D hand reconstruction methods that make use of the presented datasets.

5

REAL-TIME 3D HAND TRACKING UNDER OCCLUSION

This chapter presents an approach for real-time, robust and accurate 3D hand pose estimation from a moving egocentric RGB-D camera in cluttered real environments (published as Mueller et al., 2017). Existing methods typically fail for hand-object interactions in cluttered scenes imaged from egocentric viewpoints—common for virtual or augmented reality applications. The proposed approach uses two subsequently applied Convolutional Neural Networks (CNNs) to localize the hand and regress 3D joint locations. Hand localization is achieved by using a CNN to estimate the 2D position of the hand center in the input, even in the presence of clutter and occlusions. The localized hand position, together with the corresponding input depth value, is used to generate a normalized cropped image that is fed into a second CNN to regress relative 3D hand joint locations in real time. For added accuracy, robustness and temporal stability, pose estimates are refined using a kinematic pose tracking energy. To train the CNNs, the *SynthHands* dataset (see Section 4.1) was introduced. It was captured using a merged reality approach and large amounts of annotated data were synthesized, depicting natural hand interaction in cluttered scenes. Through quantitative and qualitative evaluation, it is shown that the method is robust to self-occlusion and occlusions by objects, particularly in moving egocentric perspectives.

5.1 INTRODUCTION

Estimating the full articulated 3D pose of hands is becoming increasingly important due to the central role that hands play in everyday human activities. Applications in activity recognition (Rohrbach et al., 2012), motion control (Zhao et al., 2013), human–computer interaction (Sridhar et al., 2015b), and virtual/augmented reality (VR/AR) require real-time and accurate hand pose estimation under challenging conditions. Spurred by recent developments in commodity depth sensing, several methods that use a single RGB-D camera have been proposed (Ge et al., 2016; Qian et al., 2014; Sridhar et al., 2015a; Tagliasacchi et al., 2015; Taylor et al., 2016; Wan et al., 2017). In particular, methods that use convolutional neural networks, possibly in combination with model-based hand tracking, have been shown to work well for static, third-person viewpoints in

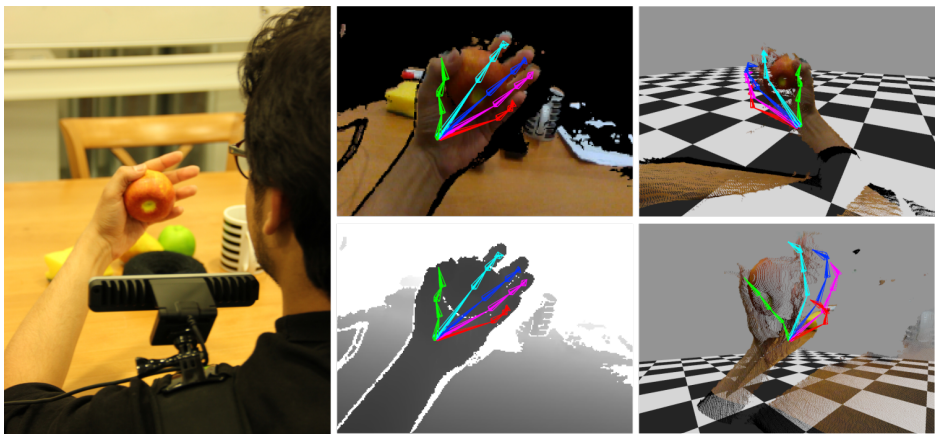


Figure 5.1: This thesis presents an approach to track the full 3D pose of the hand from egocentric viewpoints (left), a challenging problem due to additional self-occlusions, occlusions from objects and background clutter. The method can reliably track the hand in 3D even under such conditions using only RGB-D input. This figure shows tracking results overlaid with color and depth map (center), and visualized from virtual viewpoints (right).

uncluttered scenes (Oberweger et al., 2015; Sinha et al., 2016; Tompson et al., 2014), i.e., mostly for free hand motion in mid-air, a setting that is uncommon in natural hand interaction.

However, real-time hand pose estimation from *moving*, *first-person* camera viewpoints in *cluttered real-world scenes* where the hand is often occluded as it naturally interacts with objects, remains an unsolved problem. Occlusions, cluttered backgrounds, manipulated objects, and field-of-view limitations make this scenario particularly challenging. CNNs are promising to tackle this problem but typically require large amounts of *annotated* data which is hard to obtain since markerless hand tracking (even with multiple views), and manual annotation on a large scale is infeasible in egocentric settings due to (self-)occlusions, cost, and time. Even semi-automatic annotation approaches (Oberweger et al., 2016) would fail if large parts of the hand are occluded. Synthetic data, on the other hand, is inexpensive, easier to obtain, removes the need for manual annotation, and can produce accurate ground truth even under occlusion.

This thesis presents the first method that supports *real-time* egocentric hand pose estimation in real scenes with cluttered backgrounds, occlusions, and complex hand-object interactions using a single commodity RGB-D camera (see Figure 5.1). The task of per-frame hand pose estimation is divided into: (1) hand localization, and (2) 3D joint location regression. Hand localization, an important task in the presence of scene clutter, is achieved by a CNN that estimates the 2D image location of the center of the hand. Further processing results in an image-level bounding

box around the hand and the 3D location of the hand center (or of the occluding object directly in front of the center). This output is fed into a second CNN that regresses the relative 3D locations of the 21 hand joints. Both CNNs are trained with the *SynthHands* dataset, a large dataset of fully annotated data which was obtained by combining hand-object interactions with real cluttered backgrounds using a new *merged reality* approach (see Section 4.1). This increases the realism of the training data since users can perform motions to mimic manipulating a virtual object using live feedback of their hand pose. These motions are rendered from novel egocentric views using a framework that realistically models RGB-D data of hands in natural interaction with objects and clutter.

The 3D joint location predictions obtained from the CNN are accurate but suffer from kinematic inconsistencies and temporal jitter expected in single frame pose estimation methods. To overcome this, the estimated 3D joint locations are refined using a fast inverse kinematics pose tracking energy that uses 3D and 2D joint location constraints to estimate the joint angles of a temporally smooth skeleton. Together, this results in the first real-time approach for smooth and accurate hand tracking even in cluttered scenes and from moving egocentric viewpoints. The accuracy, robustness, and generality of the proposed approach are demonstrated on the new benchmark dataset *EgoDexter* with moving egocentric cameras in real cluttered environments. In summary, the contributions of this chapter are:

- A novel method that localizes the hand and estimates, in real time, the 3D joint locations from egocentric viewpoints, in clutter, and under strong occlusions using two CNNs. A kinematic pose tracking energy further refines the pose by estimating joint angles of a temporally smooth tracking.
- Extensive evaluation on the new annotated real benchmark dataset *EgoDexter* featuring egocentric cluttered scenes, diverse users, and interaction with objects.

5.2 OVERVIEW

The goal is to estimate the full 3D articulated pose of the hand imaged with a single commodity RGB-D sensor. The RGB and depth channels from the Intel RealSense SR300 (IntelRealSenseSR300, 2016) are used, both with a resolution of 640×480 pixels and captured at 30 Hz. The hand pose estimation is formulated as an energy minimization problem that incorporates per-frame pose estimates into a temporal tracking framework. The goal is to find the joint angles of a kinematic hand

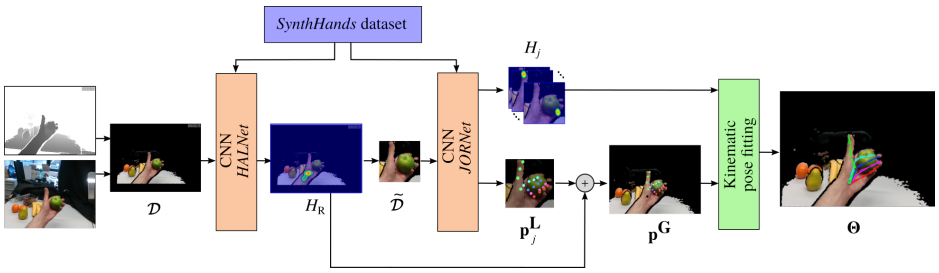


Figure 5.2: Overview of the proposed method. Starting from an RGB-D frame, the 2D hand position heatmap is regressed using the CNN *HALNet* and a cropped frame is computed. A second CNN, *JORNet*, is used to predict root-relative 3D joint positions as well as 2D joint heatmaps. Both CNNs are trained with the new *SynthHands* dataset. Finally, the joint angles of a kinematic skeleton are obtained via a pose tracking step.

skeleton (Section 3.1) that best represent the input observation. Similar strategies have been shown to be successful in state-of-the-art methods (Qian et al., 2014; Sridhar et al., 2015a, 2016; Taylor et al., 2016) that use per-frame pose estimation to initialize a tracker that refines and regularizes the joint angles of a kinematic skeleton for free hand tracking. However, the per-frame pose estimation components of these methods struggle under strong occlusions, hand-object interactions, scene clutter, and moving egocentric cameras. The proposed approach overcomes this limitation by combining a CNN-based 3D pose regression framework, that is tailored for this challenging setting, with a kinematic skeleton tracking strategy for temporally stable results.

The task of hand pose estimation is divided into several subtasks (Figure 5.2). First, *hand localization* (Section 5.3.1) is achieved by a CNN that estimates an image-level heatmap (that encodes position probabilities) of the *root* — a point which is either the hand center (knuckle of the middle finger, shown with a star shape in Figure 5.3) or a point on an object that occludes the hand center. The 2D and 3D root positions are used to extract a normalized cropped image of the hand. Second, *3D joint regression* (Section 5.3.2) is achieved with a CNN that regresses root-relative 3D joint locations from the cropped hand image. Both CNNs are trained with the *SynthHands* dataset which contains large amounts of annotated data which were generated with a novel framework to automatically generate 3D hand joint motion with natural hand interaction (Sections 4.1 and 5.3.3). Finally, the regressed 3D joint positions are used in a kinematic pose tracking framework (Section 5.4) to obtain temporally smooth tracking of the hand motion.

5.3 SINGLE FRAME 3D POSE REGRESSION

The goal of 3D pose regression is to estimate the 3D joint locations of the hand at each frame of the RGB-D input. Note that the regressed 3D location (as shown in Figure 5.3) coincide with the joints and fingertips of the kinematic hand skeleton (Section 3.1), hence guaranteeing a consistent representation for the pose fitting step. For the regression of 3D joint locations, first, a *colored depth map* $\mathcal{D}$ is created from the raw input produced by commodity RGB-D cameras (e.g., Intel RealSense SR300). $\mathcal{D}$ is defined as

$$\mathcal{D} = \texttt{colormap}(\mathbf{R}, \mathbf{G}, \mathbf{B}, \mathbf{Z}) \,, \tag{5.1}$$

where $\texttt{colormap}(\cdot)$ is a function, that depends on the camera calibration parameters, to obtain the corresponding color pixel for each pixel in the depth map $\mathbf{Z}$. Computing $\mathcal{D}$ allows to ignore camera-specific variations in extrinsic parameters. $\mathcal{D}$ is also downsampled to a resolution of 320×240 to aid real-time performance. Next, the pose regression approach that is robust even in challenging cluttered scenes with notable (self-)occlusions of the hand is described. As shown in the evaluation (Section 5.5), using a two-step approach to first localize the hand in full-frame input and subsequently estimate 3D pose outperforms using a single CNN for both tasks.

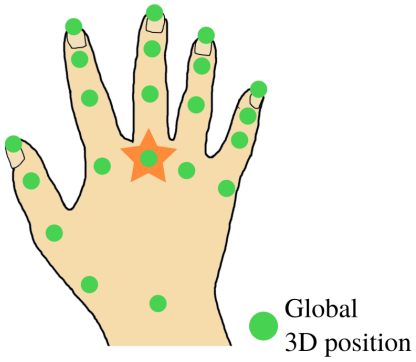


Figure 5.3: The 3D joint regression step outputs $J = 21$ global 3D joint locations, shown in green, which are later used to estimate the joint angles of a kinematic skeleton hand model. The orange star depicts the joint used as a hand root.

5.3.1 *Hand Localization*

The goal of the first part of pose regression is to localize the hand in challenging cluttered input frames resulting in a bounding box around the hand and 3D root location. Given a colored depth map $\mathcal{D}$, a crop is computed as

$$\widetilde{\mathcal{D}} = \texttt{imcrop}(\mathcal{D}, H_{\mathrm{R}}) \,, \tag{5.2}$$

where H_{R} is a heatmap encoding the position probability of the 2D hand root and $\texttt{imcrop}(\cdot)$ is a function that crops the hand area of the input frame. In particular, H_{R} is estimated using a CNN which is called *HALNet* (HAnd Localization Net). The $\texttt{imcrop}(\cdot)$ function picks the image-level

heatmap maximum location $\phi(H_R) = (u, v)$ and uses the associated depth z in $\mathcal{D}$ to compute a depth-dependent crop, the side length of which is inversely proportional to the depth and contains the hand. Additionally, `imcrop`$(\cdot)$ also normalizes the depth component of the cropped image by subtracting z from all pixels.

HALNet uses an architecture derived from *ResNet50* (He et al., 2016) which has been shown to have a good balance between accuracy and computational cost (Canziani et al., 2016). The number of residual blocks was reduced to 10 to achieve real-time framerate while maintaining high accuracy. This network is trained using *SynthHands*, a new synthetic dataset with ample variance across many dimensions such as hand pose, skin color, objects, hand-object interaction and shading details (Section 4.1). See Section 5.3.3 and Appendix A.2 for training and architecture details.

POST PROCESSING. To make the estimation of the root maximum location robust over time, an additional step is added to prevent outliers from affecting 3D joint location estimates. A history of maximum locations is maintained and they are labeled as *confident* or *uncertain* based on the following criterion. If the likelihood value of the heatmap maximum at a frame t is < 0.1 and it occurs at > 30 pixels from the previous maximum then it is marked as uncertain. If a maximum location is uncertain, it is updated as

$$\phi_t = \phi_{t-1} + \delta^k \frac{\phi_{c-1} - \phi_{c-2}}{||\phi_{c-1} - \phi_{c-2}||}, \tag{5.3}$$

where $\phi_t = \phi(H_R^t)$ is the updated 2D maximum location at the frame t, ϕ_{c-1} is the last *confident* maximum location, k is the number of frames elapsed since the last confident maximum, and δ is a decay factor to progressively downweight uncertain maxima. The decay factor is empirically set $\delta = 0.98$ and used in all results.

5.3.2 *3D Joint Regression*

Starting from a cropped and normalized input $\widetilde{\mathcal{D}}$ that contains a hand, potentially partially occluded, the goal is to regress the global 3D hand joint position vector $\mathbf{p}^{\mathbf{G}} \in \mathbb{R}^{3\times J}$. A CNN, referred to as *JORNet* (JOint Regression Net), is used to predict per-joint 3D root-relative positions $\mathbf{p}^{\mathbf{L}} \in \mathbb{R}^{3\times J}$ in $\widetilde{\mathcal{D}}$. Additionally, *JORNet* also regresses per-joint 2D position likelihood heatmaps $\mathbf{H} = \{H_j\}_{j=1}^{J}$, which will be used to regularize the predicted 3D joint positions in a later step. Global 3D joint positions are obtained as $\mathbf{p}_j^{\mathbf{G}} = \mathbf{p}_j^{\mathbf{L}} + \mathbf{r}$, where $\mathbf{r}$ is the global position of the hand center (or a point on an occluder) obtained by backprojecting the 2.5D hand root

position (u, v, z) to 3D. *JORNet* uses the same architecture as *HALNet* and is trained with the same data (Section 4.1). See Section 5.3.3 for training details and Appendix A.2 for architecture details.

5.3.3 *Training*

Both *HALNet* and *JORNet* are trained on the egocentric data from the *SynthHands* dataset. Importantly, note that *SynthHands* does not contain 3D scans of the real test objects nor 3D models of similar objects used for evaluation in Section 5.5. This demonstrates that the proposed approach generalizes to unseen objects.

The CNNs are trained using the Caffe framework (Jia et al., 2014), and the AdaDelta solver (Zeiler, 2012) with a momentum of 0.9 and weight decay factor of 0.0005. The learning rate is tapered down from 0.05 to 0.000025 during the course of the training. For training *JORNet*, the ground-truth (u, v) and z of the hand root is used to create the normalized crop input. To improve robustness, uniform noise ($\in [-25, 25]$ mm) is added to the backprojected 3D root position in the *SynthHands* dataset. *HALNet* is trained for 45,000 iterations and *JORNet* for 60,000 iterations. The final networks were chosen as the ones with the lowest loss values. The layers in the networks that are similar to *ResNet50* are initialized with weights of the original *ResNet50* architecture trained on ImageNet (Russakovsky et al., 2015). For the other layers, the weights are initialized randomly. For details of the loss weights used and the taper scheme, please see Appendix A.2.

5.4 HAND POSE OPTIMIZATION

The estimated per-frame global 3D joint positions $\mathbf{p}^{\mathbf{G}}$ are not guaranteed to be temporally smooth nor do they have consistent inter-joint distances (i.e., bone lengths) over time. This is mitigated by fitting a kinematic skeleton parameterized by joint angles $\boldsymbol{\theta}$, as introduced in Section 3.1, to the regressed 3D joint positions. Additionally, the fitting is refined by leveraging the 2D heatmap output from *JORNet* as an additional contraint and is regularized using joint limit and smoothness constraints. In particular, the goal is to minimize

$$\mathcal{E}(\boldsymbol{\theta}) = E_{\text{data}}(\boldsymbol{\theta}, \mathbf{p}^{\mathbf{G}}, \mathbf{H}) + E_{\text{reg}}(\boldsymbol{\theta}) \,, \tag{5.4}$$

where E_{data} is the data term that incorporates both the 3D positions and 2D heatmaps

$$E_{\text{data}}(\boldsymbol{\theta}, \mathbf{p}^{\mathbf{G}}, \mathbf{H}) = w_{\text{p3}} E_{\text{pos3D}}(\boldsymbol{\theta}, \mathbf{p}^{\mathbf{G}}) + w_{\text{p2}} E_{\text{pos2D}}(\boldsymbol{\theta}, \mathbf{H}) \,. \tag{5.5}$$

The first term E_{pos3D} minimizes the 3D distance between each predicted joint location $\mathbf{p}_j^{\mathbf{G}}$ and its corresponding position $\mathcal{M}(\boldsymbol{\theta})_j$ in the kinematic skeleton set to pose $\boldsymbol{\theta}$

$$E_{\text{pos3D}}(\boldsymbol{\theta}) = \sum_{j=1}^{J} ||\mathcal{M}(\boldsymbol{\theta})_j - \mathbf{p}_j^{\mathbf{G}}||_2^2 \,. \tag{5.6}$$

The second data term, E_{pos2D}, minimizes the 2D distance between each joint heatmap maximum $\phi(H_j)$ and the projected 2D location of the corresponding joint in the kinematic skeleton

$$E_{\text{pos2D}}(\boldsymbol{\theta}) = \sum_{j=1}^{J} ||\Pi(\mathcal{M}(\boldsymbol{\theta})_j) - \phi(H_j))||_2^2 \,, \tag{5.7}$$

where Π projects the joint onto the image plane. The weights for the different terms were empirically tuned as: $w_{p3} = 0.01$ and $w_{p2} = 5 \times 10^{-7}$.

The data terms are regularized by enforcing joint limits and temporal smoothness constraints

$$E_{\text{reg}}(\boldsymbol{\theta}) = w_{\text{l}} E_{\text{lim}}(\boldsymbol{\theta}) + w_{\text{t}} E_{\text{temp}}(\boldsymbol{\theta}) \tag{5.8}$$

where

$$E_{\text{lim}}(\boldsymbol{\theta}) = \sum_{\theta_i \in \boldsymbol{\theta}} \begin{cases} 0 & \text{, if } \theta_i^l \leq \theta_i \leq \theta_i^u \\ (\theta_i - \theta_i^l)^2 & \text{, if } \theta_i < \theta_i^l \\ (\theta_i^u - \theta_i)^2 & \text{, if } \theta_i > \theta_i^u \end{cases} \tag{5.9}$$

is a soft prior to enforce biomechanical pose plausibility, with $\boldsymbol{\theta}^l, \boldsymbol{\theta}^u$ being the lower and upper joint angle limits, respectively, and

$$E_{\text{temp}}(\boldsymbol{\theta}) = ||\nabla\boldsymbol{\theta} - \nabla\boldsymbol{\theta}^{(t-1)}||_2^2 \tag{5.10}$$

enforces constant velocity to prevent dramatic pose changes. The weights for the regularizers were empirically chosen as: $w_l = 0.03$ and $w_t = 10^{-3}$. The objective is optimized using 20 iterations of conditioned gradient descent.

5.5 RESULTS AND EVALUATION

Several experiments were conducted to evaluate the proposed method and different components of it. To facilitate evaluation and to test the generalization capability to real data, the new real benchmark dataset *EgoDexter* was captured. In addition, to enable evaluation of the different components of the proposed method, a synthetic test set consisting of 5120 fully annotated frames from the *SynthHands* dataset was held out.

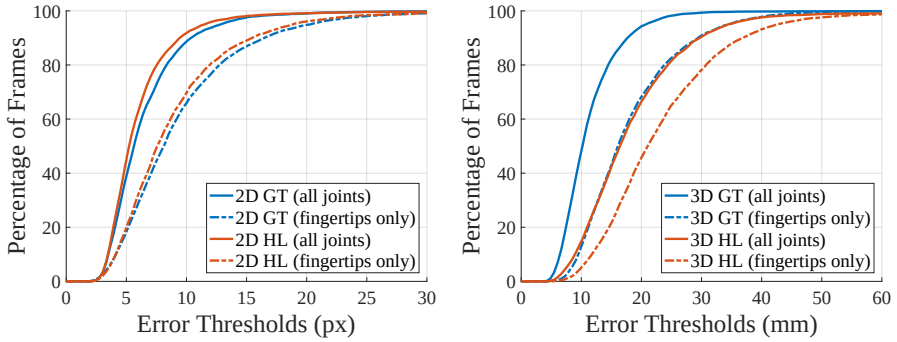


Figure 5.4: Comparison of 2D (left) and 3D (right) error of the joint position estimates of *JORNet*. *JORNet* was initialized with either the ground truth ("GT", blue ■) or with the proposed hand localization step ("HL", orange ■). It can be observed that HL initialization does not substantially reduce the performance of *JORNet*. As expected, fingertips-only errors (dashed lines) are higher than the errors for all joints.

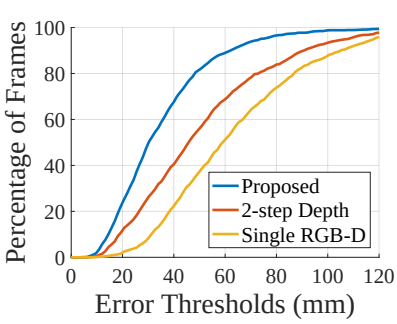
5.5.1 *Benchmark Dataset EgoDexter*

EgoDexter consists of 3190 frames of natural hand interactions with objects in real cluttered scenes, moving egocentric viewpoints, complex hand-object interactions, and natural lighting. Of these, 1485 frames were manually annotated using an annotation tool to mark 2D and 3D fingertip positions, a common approach used in free hand tracking (Ballan et al., 2012; Sridhar et al., 2013). In total 4 sequences were gathered (`Rotunda`, `Desk`, `Kitchen`, `Fruits`) featuring 4 different users (2 female), skin color variation, background variation, different objects, and camera motion. Note that the objects in *EgoDexter* are distinct from the objects in the *SynthHands* training data to show the ability of the approach to generalize.

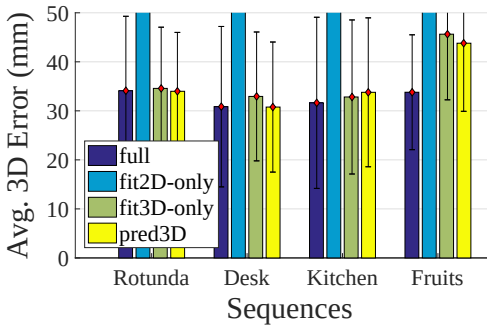
5.5.2 *Component Evaluation: HALNet and JORNet*

First, the performance of *HALNet* and *JORNet* on the synthetic test set is analyzed. The main goal of *HALNet* is to accurately localize the 2D position of the root (which either lies on the hand or on an occluder in front) accurately. Thus, the 2D Euclidean pixel error between the ground truth root position and the predicted position is used as the evaluation metric. On average, *HALNet* produces an error of **2.2 px** with a standard deviation of 1.5 px on the test set. This low average error ensures that reliable crops are obtained for *JORNet*.

To evaluate *JORNet*, the 3D Euclidean distance between ground-truth joint positions (of all hand joints) and the predicted position is used as



(a) Comparison of the proposed two-step RGB-D CNN architecture, the corresponding depth-only version and a single combined CNN which is trained to directly regress global 3D pose. The proposed approach achieves the best performance on the real test sequences.



(b) Using only the 2D fitting energy leads to catastrophic tracking failure on all sequences. The version restricted to the 3D fitting term achieves a similar error as the raw 3D predictions while it ensures biomechanical plausibility and temporal smoothness. The full formulation that combines 2D as well as 3D terms yields the lowest error.

Figure 5.5: Ablative analysis of (a) different CNN architectures and (b) the proposed kinematic pose tracking on the real annotated dataset *EgoDexter*.

the error metric. For comparison, also the errors for only the 3D fingertip positions are reported which are a stricter measure of performance. Since the output of *JORNet* is dependent on the crop estimated in the hand localization step, two conditions are evaluated: (1) using ground-truth crops ("GT", blue ■), (2) using crops from the hand localization step ("HL", orange ■). This helps evaluate how hand localization affects the final joint positions. Figure 5.4 shows the percentage of the test set that produces a certain 2D or 3D error for all joints and fingertips only. For 3D error, using ground-truth crops is better than using the crops from the hand localization. The difference is not substantial which shows that the hand localization step does not lead to catastrophic failures of *JORNet*. For 2D error, however, *JORNet* initialized with HL results in marginally better accuracy. The hypothesis is that this is because *JORNet* is trained on noisy root positions (Section 5.3.3) while the ground truth lacks any such noise.

5.5.3 *Ablation Study*

In the following, different design choices for the proposed method are evaluated: the structure of the CNN regressor, the type of input data, and the kinematic model fitting energy.

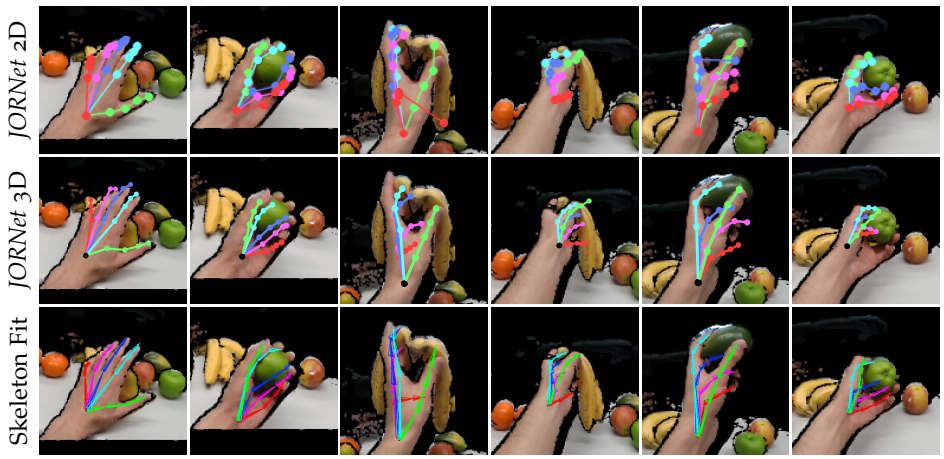


Figure 5.6: 2D predictions (top row), 3D predictions (middle row) and tracked skeleton (bottom row) on the real annotated sequence `Fruits`. The combination of 2D and 3D predictions in the tracking framework leads to better results than either of the predictions in isolation.

CNN STRUCTURE EVALUATION. It is shown that, on the real annotated benchmark *EgoDexter*, the proposed approach that uses two subsequently applied CNNs is better than a single CNN to directly regress joint positions in cluttered scenes. The CNN for comparison was trained with the same architecture as *JORNet* but with the task of directly regressing 3D joint positions from full frame RGB-D images which often have large occlusions and scene clutter. Figure 5.5a shows that the 3D fingertip error plot for this CNN ("single RGB-D", yellow ■) is worse than for the proposed two-step approach (blue ■). This demonstrates that learning to directly regress 3D pose in cluttered scenes with occlusion is a harder task, which can be simplified by breaking it into two steps.

INPUT DATA EVALUATION. Next, it is shown, on the *EgoDexter* dataset, that using both RGB and depth input (RGB-D) is superior to using only depth, even when using both of the proposed CNNs. Figure 5.5a compares the 3D fingertip error of a variant of the proposed two-step approach trained with only depth data ("2-step Depth", orange ■). The result indicates that additional color cues help the approach perform significantly better.

GAIN BY KINEMATIC MODEL. Figure 5.6 depicts the predicted joints by each of the key components of the proposed pipeline — 2D predictions, 3D predictions, and final tracked skeleton — on the test sequence `Fruits`. Note that the modes of failure for the 2D and 3D predictions are different which leads to accurate skeleton tracking even if one kind of prediction

is incorrect. Thus, the combination of 2D and 3D predictions with the tracking framework consistently produces the best results. This is also quantitatively shown in Figure 5.5b, which provides an ablative analysis of the energy terms as well as the effect of kinematic pose tracking on the final pose estimate. Because the pose tracking energy enforces joint angle limits, temporal smoothness, and consistent bone lengths, the combined approach produces the lowest average error of **32.6 mm**.

5.5.4 *Comparison to the State of the Art*

It was not possible to quantitatively evaluate on the only other existing egocentric hand dataset at the time (Rogez et al., 2015) due to a different sensor currently unsupported by the proposed approach. Using the proposed method with the *Senz3D* camera requires adaptation of intrinsic camera parameters and noise characteristics for training. However, to provide a visual comparison to evaluate the proposed method, sequences that mimic sequences used by Rogez et al., 2015 were recorded, and a qualitative evaluation is shown in Figure 5.8. The proposed method achieves significantly more accurate hand tracking, while running in real time.

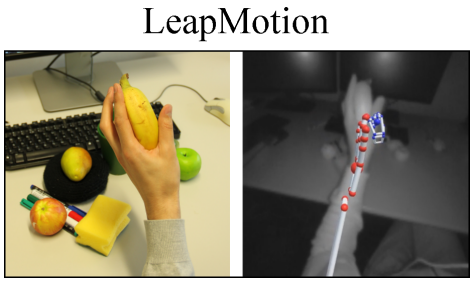


Figure 5.7: In contrast to the proposed method, LeapMotion Orion fails under large occlusions.

To show the completely different nature of the problem tackled by the proposed approach, which cannot be solved by employing state-of-the-art methods for hand tracking in free air, the method of Sridhar et al., 2015a was applied to the real test sequences from *EgoDexter*. Figure 5.9 demonstrates catastrophic failures of the aforementioned approach. On the other hand, Figure 5.10 shows how the proposed method successfully tracks sequences mimicked from the work of Sridhar et al., 2015a. The proposed approach achieves comparable results, with improved stability of the hand root position. Figure 5.7 also shows that the commercial solution LeapMotion, 2016 does not work well under severe occlusions caused by objects in contrast to the proposed approach.

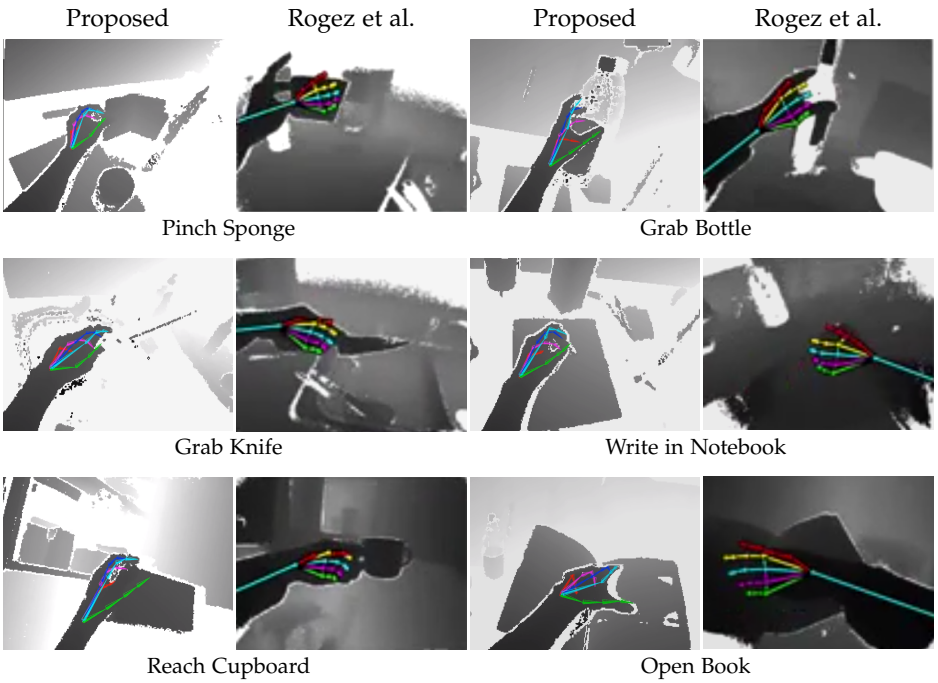


Figure 5.8: Qualitative evaluation of the proposed method and Rogez et al., 2015. The motions originally used by Rogez et al., 2015 are mimicked because, due to sensor differences (i.e., lens intrinsics, etc.), the trained CNNs cannot directly run on their data.

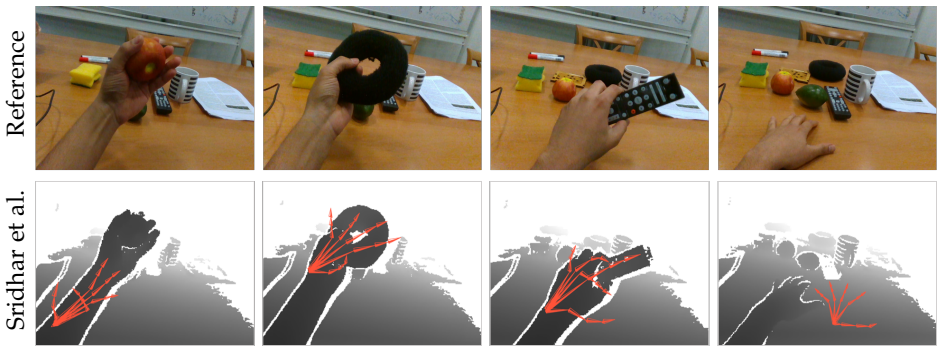


Figure 5.9: Qualitative results produced by the approach of Sridhar et al., 2015a on the benchmark dataset *EgoDexter*. The close proximity of the arm to the camera and the interaction with objects and the environment leads to catastrophic failures.

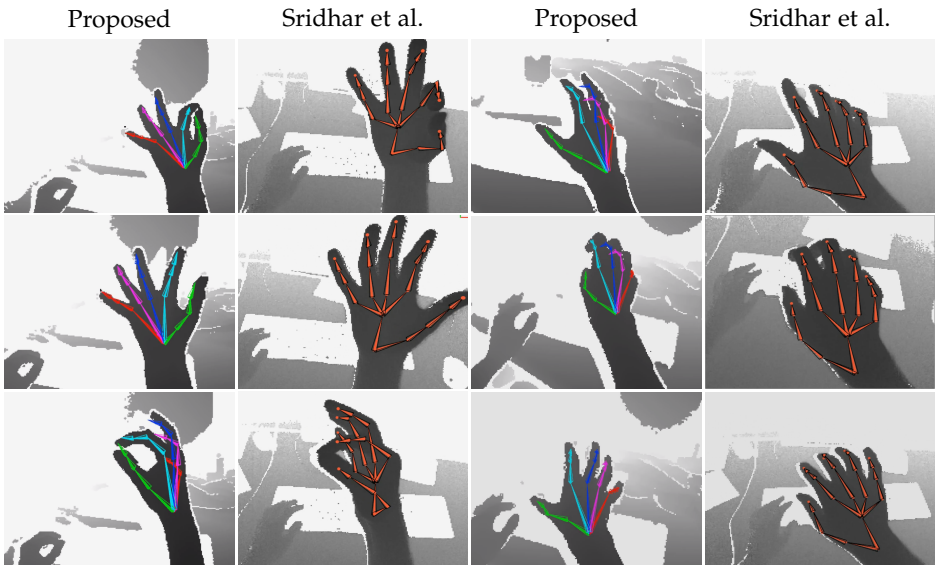


Figure 5.10: Qualitative comparison between the proposed method and Sridhar et al., 2015a, mimicking some of their sequences.

5.5.5 *Qualitative Results*

Figure 5.11 shows qualitative results from the proposed approach which works well for challenging real world scenes with clutter, hand-object interactions, and different hand shapes. Even though the machine learning components were only trained on the egocentric data from *SynthHands*, Figure 5.12 demonstrates the generalizability to 3rd-person views. Note that the hand localization step is robust to other skin-colored parts, like faces, in the input.

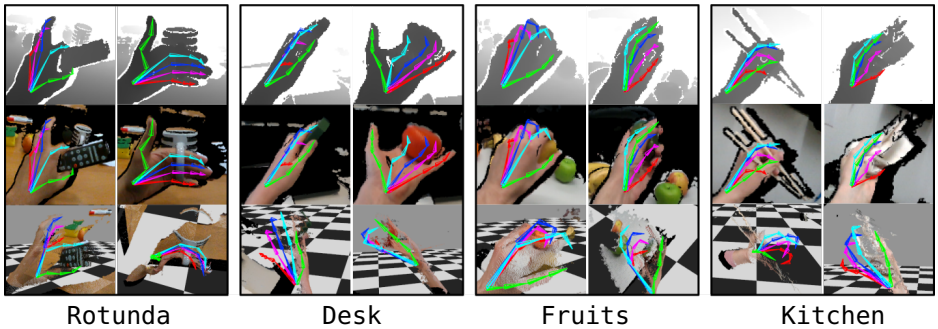


Figure 5.11: Qualitative results on the real annotated test sequences from the *EgoDexter* benchmark dataset. The results overlayed on the input images and the corresponding 3D view from a virtual viewpoint (bottom row) show that the proposed approach is able to handle complex object interactions, strong self-occlusions and a variety of users and backgrounds.

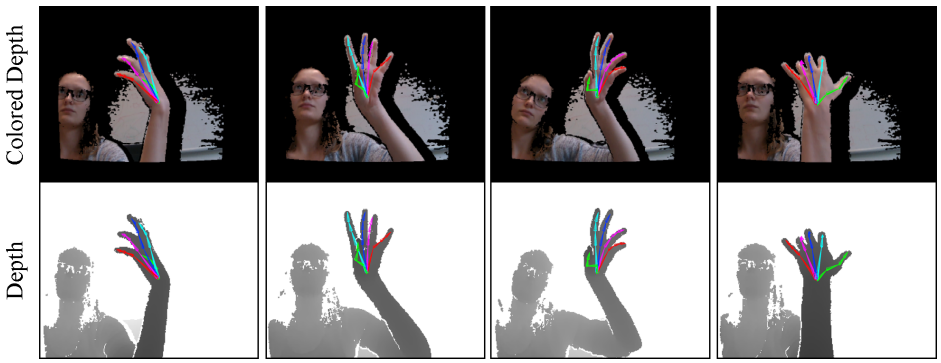


Figure 5.12: Despite being trained for egocentric views, the proposed approach in fact generalizes to 3rd-person views. In addition, it is robust to the presence of other skin-colored body parts.

5.5.6 *Runtime Performance*

The entire method runs in real-time on an Intel Xeon E5-2637 CPU (3.5 GHz) with an Nvidia Titan X (Pascal). Hand localization takes 11 ms, 3D joint regression takes 6 ms, and kinematic pose tracking takes 1 ms.

5.6 LIMITATIONS AND FUTURE WORK

The proposed method works well even for challenging egocentric viewpoints and notable occlusions. However, there are some failure cases which are shown in Figure 5.13, e.g., due to very strong occlusions or color–depth misalignment. Figure 5.14 depicts the results from several intermediate steps of the method in case of failure. In particular, the figure shows errors due to extreme self occlusions, severe hand and object occlusions, and when the hand is located out of the camera field of view.

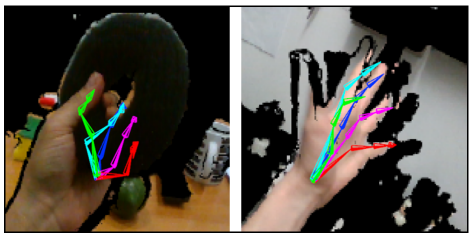


Figure 5.13: Failures in the hand localization step (left) or fast motion that leads to misalignment in the colored depth image (right) can cause incorrect pose estimates.

Large amounts of synthetic data were used for training the proposed CNNs and simulated sensor noise for a specific camera preventing generalization. In the future, the application of deep domain adaptation (Ganin and Lempitsky, 2015) could be explored which offers a way to jointly make use of labeled *synthetic* data together with unlabeled or partially labeled *real* data (see Section 4.2 and Chapter 6).

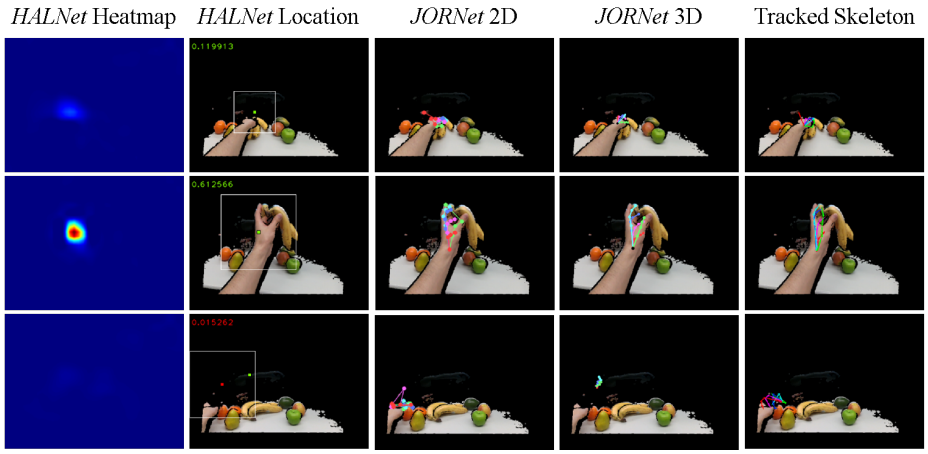


Figure 5.14: The failure cases of the proposed method can be caused by different intermediate steps. The first two columns show the output from *HALNet*, a heatmap and its maximum location with corresponding confidence. Note that the root stabilization step improved the location in the last row (from green to red) but did not succeed. The third, fourth and fifth column shows 2D predictions, 3D predictions, and the tracked kinematic skeleton, respectively.

5.7 CONCLUSION

This chapter of the thesis presented a method for hand pose estimation in challenging first-person viewpoints with large occlusions and scene clutter. The proposed method uses two CNNs to localize and estimate, in real time, the 3D joint locations of the hand. A pose tracking energy further refines the pose by estimating the joint angles of a kinematic skeleton for temporal smoothness. To train the CNNs, the *SynthHands* dataset was used (Section 4.1) which leverages a merged reality approach to capture natural hand interactions, hand shape, size and color variations, object occlusions, and background variations from egocentric viewpoints. Furthermore, a new benchmark dataset *EgoDexter* was introduced, containing annotated real sequences of challenging cluttered scenes as seen from egocentric viewpoints. Quantitative and qualitative evaluation showed that the proposed approach is capable of achieving low errors and consistent performance even under difficult occlusions, scene clutter, and background changes.

As mentioned in the limitations, the presented approach was trained on fully synthetic data. The next chapter explores how to use enhanced training data which reduces the domain gap between synthetic and real hand images to build a method for hand tracking from monocular RGB video, removing the need for depth sensors.

6 REAL-TIME 3D HAND TRACKING FROM MONOCULAR RGB VIDEO

The previous chapter introduced a new approach for real-time hand tracking under occlusion and in cluttered scenes from an egocentric RGB-D sensor. This chapter (published as Mueller et al., 2018) aims to remove the need for depth sensing by the camera, solely relying on a monocular RGB camera, e.g., a commodity webcam. The proposed tracking method combines a convolutional neural network with a kinematic 3D hand model, such that it generalizes well to unseen data, is robust to occlusions and varying camera viewpoints, and leads to anatomically plausible as well as temporally smooth hand motions. While the approach from Chapter 5 used purely synthetic data to train the machine learning components, the method presented in this chapter leverages the enhanced *GANerated Hands* dataset which was created using domain adaptation techniques (Section 4.2). The *GANerated Hands* dataset was created by a geometrically consistent image-to-image translation network that translates synthetic images to "real" images, such that the so-generated images resemble the statistical distribution of real-world hand images. It is demonstrated that the proposed hand tracking system outperforms the current state-of-the-art on challenging RGB-only footage and that using GANerated data indeed improves the performance on real test sequences.

6.1 INTRODUCTION

Estimating the 3D pose of the hand is a long-standing goal in computer vision with many applications such as in virtual/augmented reality (VR/AR) (Lee and Hollerer, 2009; Piumsomboon et al., 2013) and human–computer interaction (Markussen et al., 2014; Sridhar et al., 2015b). While there is a large body of existing works that consider marker-free image-based hand tracking or pose estimation, many of them require depth cameras (Ge et al., 2016; Qian et al., 2014; Sharp et al., 2015; Sridhar et al., 2015a; Tagliasacchi et al., 2015; Taylor et al., 2016; Wan et al., 2017) or multi-view setups (Ballan et al., 2012; Sridhar et al., 2013; Wang et al., 2011). However, in many applications these setups are unfavorable since the required hardware is less ubiquitous, more expensive, and does not work in all scenes.

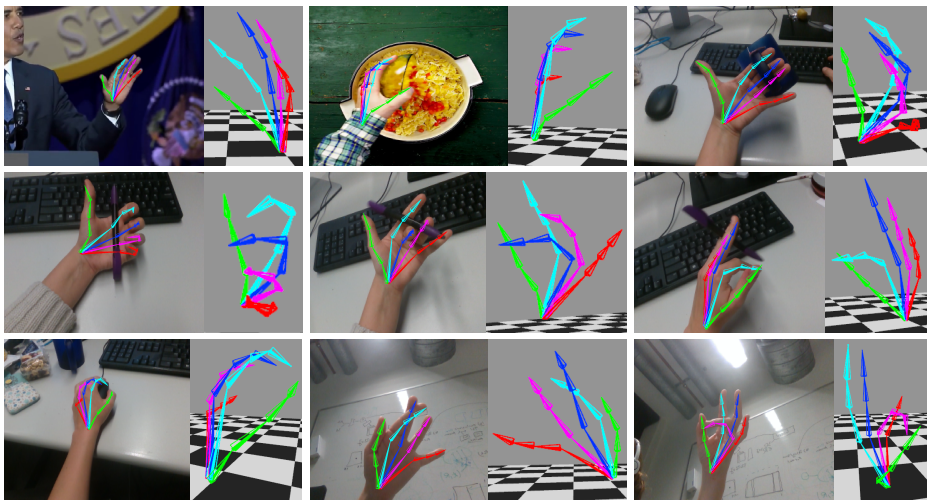


Figure 6.1: This chapter presents an approach for real-time 3D hand tracking from monocular RGB-only input. The proposed method is compatible with unconstrained video input such as community videos from YouTube (top left, top center), and robust to occlusions. It achieves real-time 3D hand tracking results using an off-the-shelf RGB webcam in unconstrained setups (top right, middle row, bottom row).

In contrast, this thesis addresses these issues and proposes a new algorithm for *real-time skeletal 3D hand tracking* with a *single color camera* that is robust under object *occlusion and clutter*. Recent developments that consider RGB-only markerless hand tracking (Gomez-Donoso et al., 2017; Simon et al., 2017; Zimmermann and Brox, 2017) come with clear limitations. For example, the approach by Simon et al., 2017 achieves the estimation of 3D joint locations within a multi-view setup; however in the monocular setting only 2D joint locations are estimated. Similarly, the method by Gomez-Donoso et al., 2017 is also limited to 2D. Recently, Zimmermann and Brox, 2017 presented a 3D hand pose estimation method based on monocular RGB data, however, they only obtain relative 3D positions and struggle with occlusions.

Inspired by recent work on hand and body tracking (Mehta et al., 2017b; Mueller et al., 2017; Tompson et al., 2014), the proposed method combines CNN-based 2D and 3D hand joint predictions with a kinematic fitting step to track hands in global 3D from monocular RGB. The major issue of such (supervised) learning-based approaches is the requirement of suitable *annotated* training data. While it has been shown to be feasible to manually annotate 2D joint locations in single-view RGB images (Johnson and Everingham, 2010), it is impossible to accurately annotate in 3D due to the inherent depth ambiguities. One way to overcome this issue is to leverage existing multi-camera methods for tracking hand motion in 3D (Ballan et

al., 2012; Gomez-Donoso et al., 2017; Sridhar et al., 2013; Wang et al., 2011). However, the resulting annotations would lack precision due to inevitable tracking errors. Some works render synthetic hands for which the perfect ground truth is known (e.g., Chapter 5 or Zimmermann and Brox, 2017). However, CNNs trained on synthetic data may not always generalize well to real-world images. Hence, the *GANerated Hands* dataset, as introduced in Section 4.2, is used as training data. The *GANerated Hands* dataset was created by performing image-to-image translation between synthetic and real images. The creation procedure fulfills two strong requirements. First, it works on *unpaired images* so that a large-scale real hands dataset can easily be collected. Second, the algorithm preserves the pose of the hand with a geometric consistency loss such that the annotations of the synthetic images are still valid for the translated images. Throughout the rest of this chapter, images are denoted as "real" (in quotes), or *GANerated*, to refer to synthetic images after they have been processed by the translation network, the *GeoConGAN* (Section 4.2), such that they resemble the statistical distribution of real-world images.

Finally, using annotated RGB images produced by the *GeoConGAN*, a CNN that jointly regresses image-space 2D and root-relative 3D hand joint positions is trained. While the skeletal hand model in combination with the 2D predictions is sufficient to estimate the global translation of the hand, the relative 3D positions resolve the inherent ambiguities in global rotation and articulation which occur in the 2D positions. In summary, this chapter contributes:

- The first real-time hand tracking system that tracks *global 3D hand pose* from unconstrained monocular RGB-only images.
- Experiments about the influence of the domain gap between synthetic and real images for 3D hand pose estimation from monocular RGB.

6.2 OVERVIEW

The main goal of this chapter is to present a real-time system for monocular RGB-only hand tracking in 3D. The overall system is outlined in Figure 6.2. Given a live monocular RGB-only video stream, a CNN hand joint regressor, *RegNet*, is used to predict 2D joint heatmaps and 3D joint positions (Section 6.3). *RegNet* is trained with images from the *GANerated Hands* dataset that were generated by an image-to-image translation network, the *GeoConGAN*, that enriches synthetic hand images. These images are better suited to train a CNN that should work on real footage, as demonstrated in Section 6.5.1. After joint regression, a kinematic skeleton

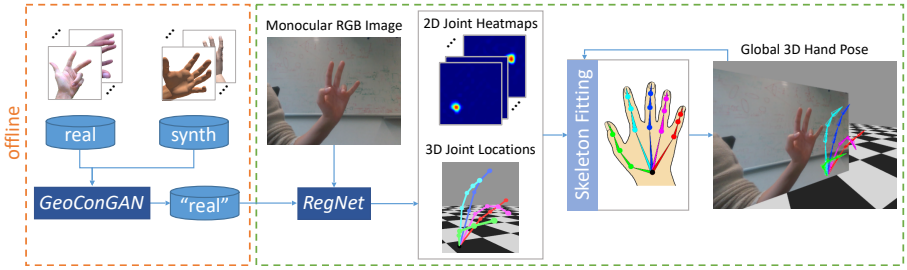


Figure 6.2: Overview of the proposed real-time system for monocular RGB hand tracking in 3D.

(as introduced in Section 3.1) is fitted to both the 2D and 3D predictions by minimizing a fitting energy (Section 6.4). The energy formulation has several key advantages for achieving a robust 3D hand pose tracking:

1. Anatomical plausibility is enforced.
2. The *absolute* 3D positions are retrieved.
3. Pose ambiguities that arise in monocular RGB images are resolved.
4. Temporal stability is imposed across multiple frames.

6.3 HAND JOINTS REGRESSION

In order to regress the hand pose from a (cropped) RGB image of the hand, a CNN that predicts 2D and 3D positions of 21 hand joints is trained, which is in the following referred to as *RegNet*. The 2D joint positions are represented as heatmaps in image space, and the 3D positions are represented as 3D coordinates relative to the root joint normalized by the hand size. Note that regressing both 2D and 3D joints is complementary to each other, as the 2D heatmaps are able to represent uncertainties and describe the position relative to the camera, whereas the 3D positions resolve the depth ambiguities in the hand articulations.

6.3.1 *Network Architecture*

The *RegNet*, shown in Figure 6.3, is based on a residual network consisting of 10 residual blocks that is derived from the *ResNet50* architecture (He et al., 2016), as done in Chapter 5. Additionally, a (differentiable) refinement module based on a projection layer (*ProjLayer*) is incorporated to better coalesce the 2D and 3D predictions. The idea of the *ProjLayer* is to perform an orthographic projection of (preliminary) intermediate 3D predictions, from which 2D Gaussian heatmaps are created (within the layer). These

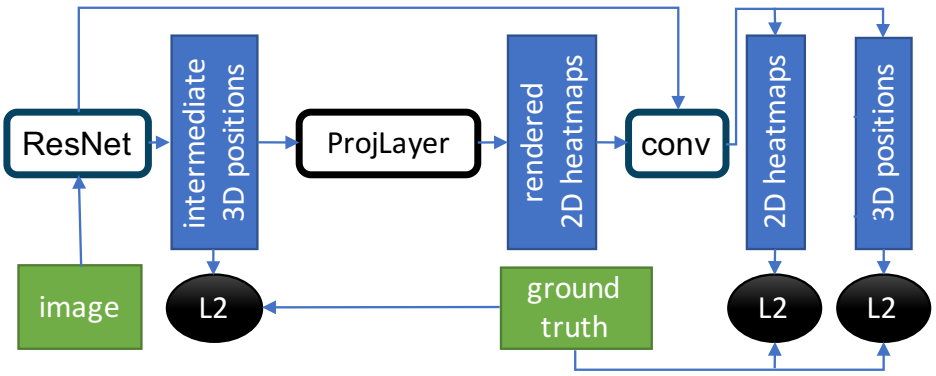


Figure 6.3: Architecture of *RegNet*. While only *ResNet* and *conv* are trainable, errors are still back-propagated through the proposed *ProjLayer*. Input data is shown in green, data generated by the network in blue, and the losses are shown in black.

heatmaps are then leveraged in the remaining part of the network (*conv*) to obtain the final 2D and 3D predictions. Figure 6.4 shows that this leads to improved results.

6.3.2 *Network Training*

The training is based on a mixture of *GANerated* (Section 4.2) and synthetic images, in conjunction with corresponding 3D ground-truth joint positions. The training set contains approximately 440,000 samples in total out of which 60% are *GANerated*. It was empirically found that the performance on real test data does not further improve by increasing this percentage. *RegNet* is trained with *relative* 3D joint positions, which are computed by normalizing the absolute 3D ground-truth joint positions such that the middle finger metacarpophalangeal (MCP) joint is at the origin and the distance between the wrist joint and the middle MCP joint is 1. Details can be found in Appendix A.3.

During test time, i.e., for hand tracking, the input to *RegNet* is a cropped RGB image, where the (square) bounding box is derived from the 2D detections of the previous frame. In the first frame, the square bounding box is located at the center of the image, with size equal to the input image height. Also, the outputs of *RegNet* are filtered with the 1€ filter (Casiez et al., 2012) to obtain temporally smoother predictions.

6.4 KINEMATIC SKELETON FITTING

After obtaining the 2D joint predictions in the form of heatmaps in image space, and the 3D joint coordinates relative to the root joint, a kinematic skeleton model is fitted to this data. This ensures an anatomically plausible hand pose, while at the same time allowing to retrieve the *absolute* hand pose, as described below. Note that we use the camera coordinate system as global coordinate frame. Moreover, when processing a sequence of images, i.e., performing hand *tracking*, temporal smoothness can be imposed additionally.

6.4.1 *Hand Model Adaptation*

To account for bone length variability across different users, *per-user skeleton adaptation* is performed. The user-specific bone lengths are obtained by averaging relative bone lengths of the 2D prediction over 30 frames while the users hold their hand parallel to the camera image plane. Alternatively, when this short adaptation sequence is not available, e.g., for community videos, the relative bone lengths of the 3D joint predictions are used. Up to a single factor due to the inherent scale ambiguity in RGB data, global 3D results can be determined which is important for many applications and not supported by previous work (Zimmermann and Brox, 2017). In addition, metrically accurate 3D results are obtained when provided with the metric length of a single bone.

6.4.2 *Fitting Energy*

For model fitting, the goal is to minimize the energy

$$\mathcal{E}(\boldsymbol{\theta}) = E_{\text{2D}}(\boldsymbol{\theta}) + E_{\text{3D}}(\boldsymbol{\theta}) + E_{\text{lim}}(\boldsymbol{\theta}) + E_{\text{temp}}(\boldsymbol{\theta}) \,, \tag{6.1}$$

where the individual energy terms are described below.

2D FITTING TERM. The purpose of E_{2D} is to minimize the distance between the hand joint position projected onto the image plane and the heatmap maxima. It is given by

$$E_{\text{2D}}(\boldsymbol{\theta}) = \sum_j \omega_j \| \Pi(\mathcal{M}_j(\boldsymbol{\theta}))) - \boldsymbol{u}_j \|_2^2 \,, \tag{6.2}$$

where $\boldsymbol{u}_j \in \mathbb{R}^2$ denotes the heatmap maximum of the j-th joint, $\omega_j > 0$ is a scalar confidence weight derived from the heatmap, and $\Pi : \mathbb{R}^3 \mapsto \mathbb{R}^2$ is the projection from 3D space to the 2D image plane, which is based on

the camera intrinsics. Note that this 2D term is essential for retrieving *absolute* 3D positions since the 3D fitting term E_{3D} takes only root-relative articulation into account, as described next.

3D FITTING TERM. The purpose of the term E_{3D} is to obtain an accurate estimate of the 3D hand articulation by using the predicted *relative* 3D joint positions. Moreover, this term resolves depth ambiguities that are present when using 2D joint positions only. E_{3D} is defined as

$$E_{3D}(\boldsymbol{\theta}) = \sum_j \|(\mathcal{M}_j(\boldsymbol{\theta}) - \mathcal{M}_{\text{root}}(\boldsymbol{\theta})) - \boldsymbol{z}_j\|_2^2 \,. \tag{6.3}$$

The variable $\boldsymbol{z}_j \in \mathbb{R}^3$ is the user-specific position of the j-th joint relative to the root joint, which is computed from the output of the *RegNet*, $\boldsymbol{x}_j$, as

$$\boldsymbol{z_j} = \boldsymbol{z}_{p(j)} + \frac{\|\mathcal{M}_j(\boldsymbol{\theta}) - \mathcal{M}_{p(j)}(\boldsymbol{\theta})\|_2}{\|\boldsymbol{x}_j - \boldsymbol{x}_{p(j)}\|_2}(\boldsymbol{x}_j - \boldsymbol{x}_{p(j)}) \,, \tag{6.4}$$

where $p(j)$ is the parent of joint j and $\boldsymbol{z}_{\text{root}} = \mathbf{0} \in \mathbb{R}^3$. The idea of using user-specific positions is to avoid poor local minima caused by bone length inconsistencies between the hand model and the 3D predictions.

JOINT ANGLE CONSTRAINTS. The term E_{lim} penalizes anatomically implausible hand articulations by enforcing that joints do not bend too far. Note that the limit constraint is not used for the global rigid translation and rotation parameters, but only for the part of the pose parameters $\boldsymbol{\theta}$ that describe the articulation angles, further denoted as $\boldsymbol{\theta}^{\text{artc}} \in \mathbb{R}^{20}$. Mathematically, it is defined as

$$E_{\text{lim}}(\boldsymbol{\theta}) = \|\max(\left[\mathbf{0}, \boldsymbol{\theta}^{\text{artc}} - \boldsymbol{\theta}^{\max}, \boldsymbol{\theta}^{\min} - \boldsymbol{\theta}^{\text{artc}}\right])\|_2^2 \,, \tag{6.5}$$

where $\boldsymbol{\theta}^{\max}, \boldsymbol{\theta}^{\min}$ are the upper and lower limits for joint angles and the function $\max : \mathbb{R}^{\odot \times 3} \mapsto \mathbb{R}^{\odot}$ computes the row-wise maximum.

TEMPORAL SMOOTHNESS. The term E_{temp} penalizes deviations from constant velocity in $\boldsymbol{\theta}$. It is formulated as

$$E_{\text{temp}}(\boldsymbol{\theta}) = \|(\nabla\boldsymbol{\theta}^{\text{prev}} - \nabla\boldsymbol{\theta})\|_2^2 \,, \tag{6.6}$$

where the gradients of the pose parameters $\boldsymbol{\theta}$ are determined using finite (backward) differences.

6.4.3 *Optimization*

In order to minimize the energy in Equation (6.1) a gradient-descent strategy is used. Let $\boldsymbol{t}, \boldsymbol{r} \in \mathbb{R}^3$ be the parts of the pose parameters $\boldsymbol{\theta}$

that express the global rigid translation and rotation, respectively, and $\boldsymbol{\theta}^{\text{artc}} \in \mathbb{R}^{20}$ be the remaining parameters, namely the articulation angles, as defined for the joint angle constraint. For the first frame, $\boldsymbol{\theta}$ is initialized to represent a flat hand that is centered in the image and 45cm away from the camera plane. For the remaining frames the translation and articulation parameters $\boldsymbol{t}$ and $\boldsymbol{\theta}^{\text{artc}}$ from the previous frame are used as initialization. While conducting experiments, it was found that fast global hand rotations may lead to a poor optimization result corresponding to a local minimum in the non-convex energy landscape. In order to deal with this problem, the initialization for the global rotation matrix $\text{rotmat}(\boldsymbol{r}) = \boldsymbol{R} \in \text{SO}(3)$ does not rely on the previous value $\boldsymbol{R}^{\text{prev}}$, but instead it is based on the relative 3D joint predictions. Specifically, it can be observed that in the human hand the wrist joint and its four direct child joints of the non-thumb fingers (the respective MCP joints) are (approximately) rigid (see Figure 6.2, *Skeleton Fitting* block). Thus, the global rotation $\boldsymbol{R}$ can be found by solving the problem

$$\min_{\boldsymbol{R} \in \text{SO}(3)} \|\boldsymbol{R}\bar{\boldsymbol{Z}} - \tilde{\boldsymbol{Z}}\|_F^2 \,, \tag{6.7}$$

where $\bar{\boldsymbol{Z}}$ contains (fixed) direction vectors derived from the *hand model*, and $\tilde{\boldsymbol{Z}}$ contains the corresponding direction vectors that are derived from the current *RegNet predictions*. Both have the form $\boldsymbol{Z} = \left[\boldsymbol{y}_{j_1}, \boldsymbol{y}_{j_2}, \boldsymbol{y}_{j_3}, \boldsymbol{y}_{j_4}, n\right] \in \mathbb{R}^{3\times 5}$, where the $\boldsymbol{y}_{j_k} = \frac{1}{\|\boldsymbol{x}_{j_k} - \boldsymbol{x}_{\text{root}}\|}(\boldsymbol{x}_{j_k} - \boldsymbol{x}_{\text{root}}) \in \mathbb{R}^3$ are (normalized) vectors that point from the wrist joint to the respective non-thumb MCP joints $j_1, \ldots, j_4$, and $n = \boldsymbol{y}_{j_1} \times \boldsymbol{y}_{j_4}$ is the (approximate) normal vector of the "palm-plane". To obtain $\bar{\boldsymbol{Z}}$ we compute the $\boldsymbol{y}_j$ based on the $\boldsymbol{x}_j$ of the 3D *model points* in world space, which is done only once for a skeleton at the beginning of the tracking when the global rotation of the model is the identity rotation. To obtain $\tilde{\boldsymbol{Z}}$ in each frame, the $\boldsymbol{x}_j$ are set to the *RegNet predictions* for computing the $\boldsymbol{y}_j$. While problem (6.7) is non-convex, it still admits the efficient computation of a global minimum as it is an instance of the *Orthogonal Procrustes Problem* (Schönemann, 1966; Ten Berge, 2006): for $\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$ being the singular value decomposition of $\tilde{\boldsymbol{Z}}\bar{\boldsymbol{Z}}^T \in \mathbb{R}^{3\times 3}$, the global optimum of (6.7) is given by $\mathbf{R} = \boldsymbol{U}\,\text{diag}(1, 1, \det(\boldsymbol{U}\boldsymbol{V}^T))\boldsymbol{V}^T$.

6.5 EXPERIMENTS

The proposed method is quantitatively and qualitatively evaluated and compared to other state-of-the-art methods on a variety of publicly available datasets. For that, the Percentage of Correct Keypoints (PCK) score is used, a popular criterion to evaluate pose estimation accuracy.

PCK defines a candidate keypoint to be correct if it falls within a circle (2D) or sphere (3D) of given radius around the ground truth.

6.5.1 *Ablation Study*

Figure 6.4 shows a comparison of the accuracies when training the joint regressor *RegNet* with different types of training data. Specifically, the following versions are compared:

(1) using synthetic images only (blue ■),

(2) using synthetic images plus color augmentation (orange ■),

(3) using synthetic images in combination with *GANerated* images (yellow ■),

(4) in addition to (3) the *ProjLayer* is used in *RegNet* (purple ■).

For color augmentation, gamma correction was employed with random $\gamma \in [0.25, 2]$ sampled uniformly. While the *RegNet* is evaluated on the entire *Stereo Dataset* (Zhang et al., 2016) comprising 12 sequences, it was *not trained on any* frame of the dataset for this test. Training on purely synthetic data leads to poor accuracy (3D PCK@50mm $\approx$ 0.55). While color augmentation on synthetic images improves the results, training on *GANerated* images significantly outperforms standard augmentation techniques, achieving a 3D PCK@50mm $\approx$ 0.80. This test validates the argument for using *GANerated* images.

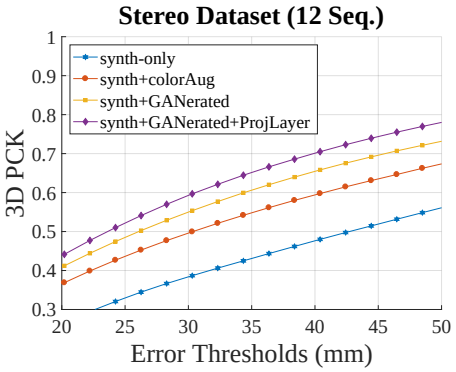
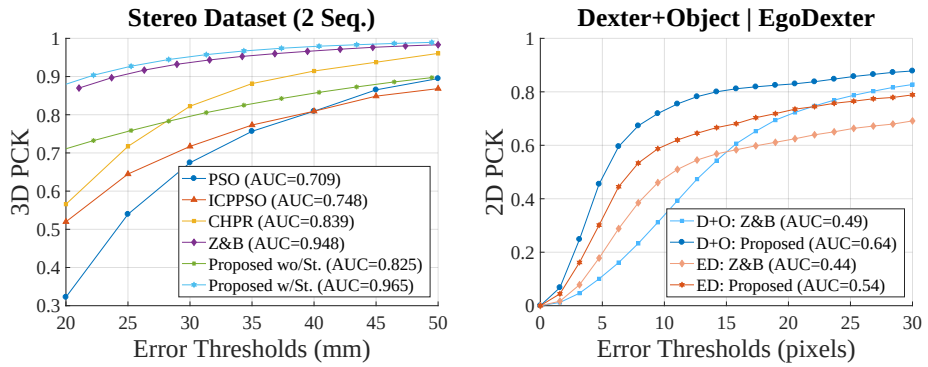


Figure 6.4: Ablative study analyzing different training options using the 3D PCK metric on the *Stereo Dataset* (Zhang et al., 2016). Using *GANerated* images (yellow) outperforms using only synthetic images (blue). Our projection layer (purple) to enforce 2D/3D consistency further improves accuracy.

6.5.2 *Comparison to the State of the Art*

Figure 6.5a evaluates the detection accuracy of the proposed approach on the *Stereo Dataset*, and compares it to existing methods (Zhang et al., 2016; Zimmermann and Brox, 2017). The same evaluation protocol as in Zimmermann and Brox, 2017 was used, i.e., training is performed



(a) 3D PCK on the *Stereo Dataset* (Zhang et al., 2016). 10 sequences are used for training and 2 for testing, as in previous works (Zhang et al., 2016; Zimmermann and Brox, 2017). The proposed method (light blue) achieves the best results.

(b) 2D PCK on the *Dexter+Object (D+O)* (Sridhar et al., 2016) and *EgoDexter (ED)* (Chapter 5) datasets. The proposed method (saturated blue/orange) outperforms Zimmermann and Brox, 2017 (Z&B) on both datasets.

Figure 6.5: Quantitative comparison with state-of-the-art methods on publicly available datasets.

on 10 sequences and testing on the other 2. Furthermore, Zimmermann and Brox, 2017 align their 3D prediction to the ground-truth wrist which is also done for the results of the proposed approach for fairness. The proposed method (light blue ■) outperforms all existing methods. Additionally, even *without* training on any sequence of the *Stereo Dataset* the proposed method still outperforms some of the existing works (green line ■ in Figure 6.5a). This demonstrates the generalization of the presented approach.

Figure 6.5b shows the 2D PCK, in pixels, on the *Dexter+Object* (Sridhar et al., 2016) and *EgoDexter* (Chapter 5) datasets. The proposed method (saturated blue ■/orange ■) significantly outperforms Zimmermann and Brox, 2017 (pale blue ■/orange ■), which fails under difficult occlusions. Note that the 3D PCK cannot be reported since the method by Zimmermann and Brox, 2017 only outputs root-relative 3D, and these datasets do not have root joint annotations.

Figure 6.6 presents qualitative results of the proposed method and the method by Zimmermann and Brox, 2017 on three datasets: the *Stereo Dataset* (Zhang et al., 2016), *Dexter+Object* (Sridhar et al., 2016), and *EgoDexter* (Chapter 5). The proposed method is able to provide robust tracking of the hand even under severe occlusions, and significantly improves over Zimmermann and Brox, 2017 in these cases. While the proposed approach already outperformed Zimmermann and Brox, 2017 in the quantitative evaluation (see Figure 6.5b), it should be emphasized that this is not the full picture, since the *Dexter+Object* and *EgoDexter*

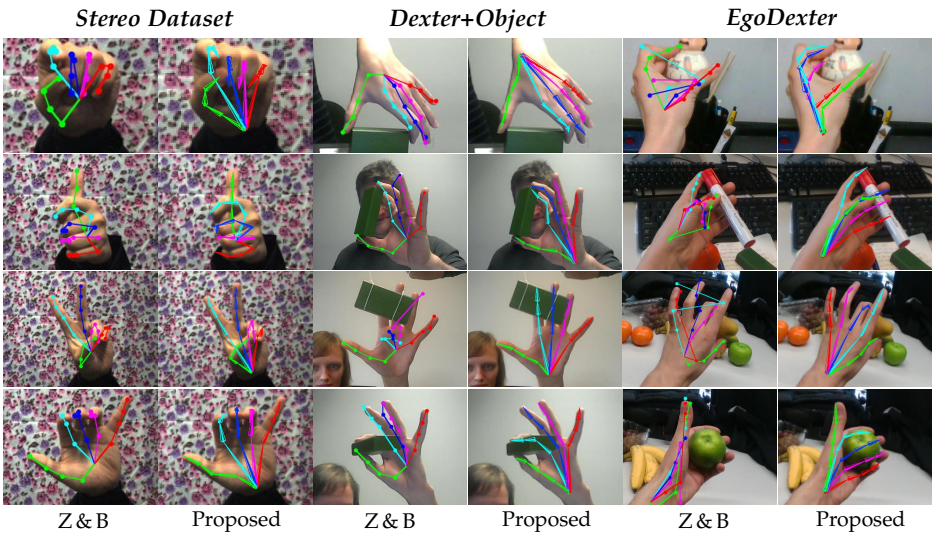


Figure 6.6: Qualitative comparison to Zimmermann and Brox, 2017 (Z & B) on three different datasets. The proposed method is more robust in cluttered scenes and correctly retrieves the hand articulation when fingers are hidden behind objects.

datasets only provide annotations for *visible* finger tips due to the manual annotation process. Thus, the error of occluded joints is not at all reflected in the quantitative analysis. Since the proposed method is explicitly trained to deal with occlusion, in contrast to Zimmermann and Brox, 2017, the qualitative analysis in columns 3–6 of Figure 6.6 highlights the superiority of the presented method in such scenarios.

6.5.3 *Comparison to RGB-D Methods*

The proposed method for RGB-only 3D hand reconstruction is compared to the RGB-D method by Sridhar et al., 2016. The 3D tracking of hands in purely RGB images is an extremely challenging problem due to inherent depth ambiguities of monocular RGB images. While the proposed method advances the state-of-the-art of RGB-only hand tracking methods, there is still a gap between RGB-only and RGB-D methods (e.g., Chapter 5 or Sharp et al., 2015; Sridhar et al., 2015a). A quantitative analysis of this accuracy gap is shown in Figure 6.7, where the results of the proposed method (dark blue ■) are compared with the RGB-D method from Sridhar et al., 2016 (orange ■). The mean error for the proposed RGB approach is ≈5cm, whereas the RGB-D method of Sridhar et al., 2016 achieves ≈2cm on their dataset *Dexter+Object*.

In order to better understand the source of errors, an additional experiment is performed where the global z-position of the RGB-only results is translated to best match the depth of the ground truth. In Figure 6.7 these depth-normalized results (light blue ■) are compared with the original results (dark blue ■). It can be seen that a significant portion of the gap between methods based on RGB and RGB-D is due to inaccuracies in the estimation of the hand root position. Reasons for an inaccurate hand root position include a skeleton that does not perfectly fit the user's hand (in terms of bone lengths), as well as inaccuracies in the 2D predictions.

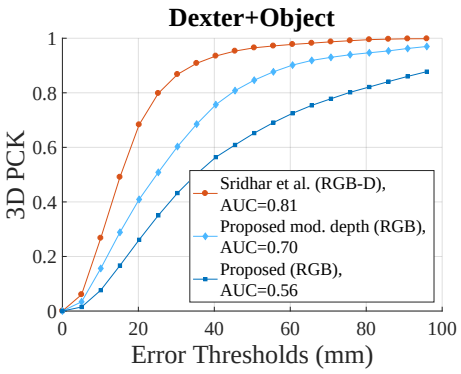


Figure 6.7: 3D PCK on *Dexter+Object*. Note that the approach by Sridhar et al., 2016 requires RGB-D input, while the proposed method uses RGB only.

6.5.4 *Qualitative Evaluation*

Qualitative results of the method on RGB videos from different sources are shown in Figure 6.1. It is demonstrated that the proposed method is compatible with community or vintage RGB video. In particular, 3D hand tracking in YouTube videos is presented, which demonstrates the generalization of the method. Other sequences were tracked live with a regular desktop webcam in an office environment. The proposed method accurately recovers the full 3D articulated pose of the hand.

Figure 6.8 and Figure 6.9 show qualitative evaluation of each of the intermediate stages along the proposed tracking solution as well as the final result. In particular, Figure 6.8 contains results on the *EgoDexter* dataset (Chapter 5) where a subject grabs different objects in an office environment, and Figure 6.9 shows results on community videos downloaded from YouTube. In both figures, various visualizations are provided: heatmap maxima of the 2D joint detections (first row); root-relative 3D joint detections (second row); global 3D tracked hand projected into camera plane (third row); and global 3D tracked hand visualized in a virtual scenario with the original camera frustum (fourth and fifth rows).

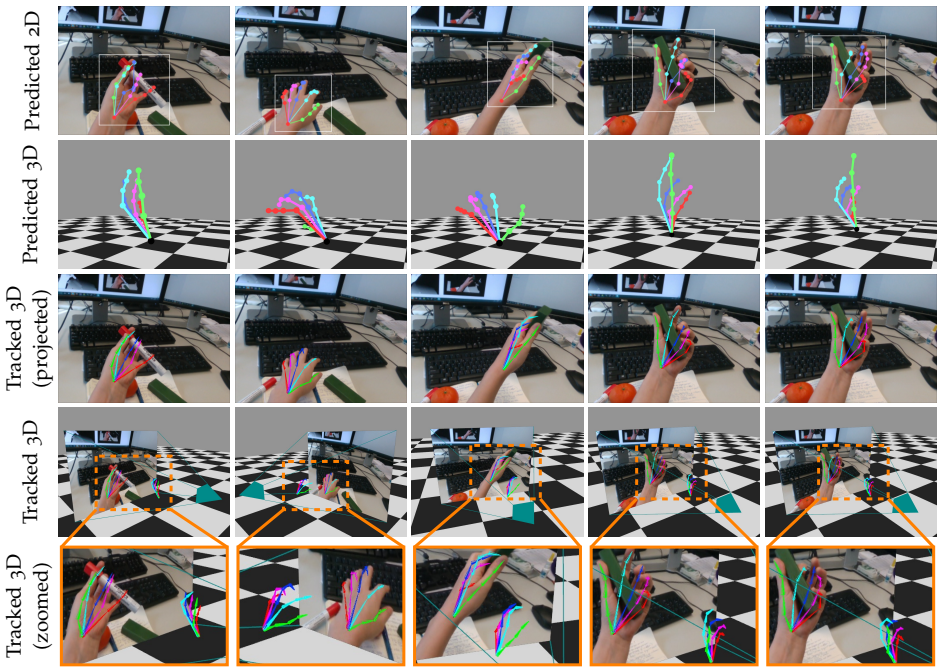


Figure 6.8: Qualitative results of different stages of the method on the *Desk* sequence from *EgoDexter* (Chapter 5). *RegNet* output (rows 1,2) and final tracking.

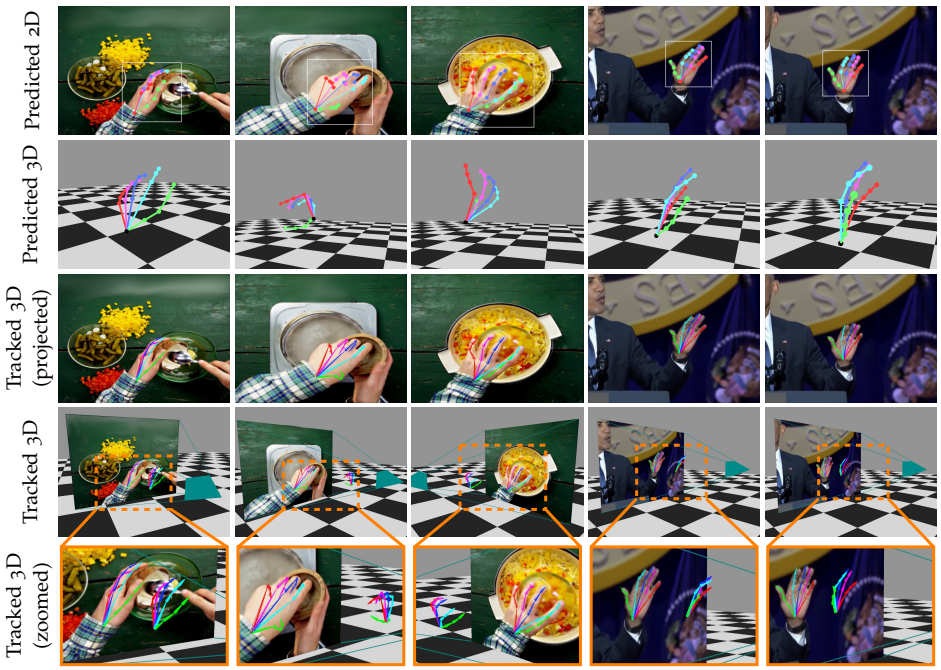
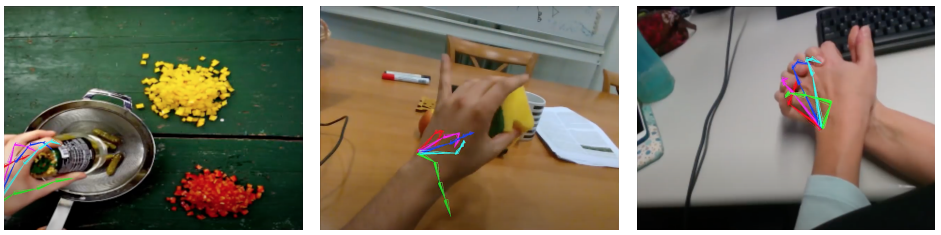


Figure 6.9: Qualitative results of different stages of the method on community videos from YouTube. *RegNet* output (rows 1,2) and final tracking.



(a) The hand leaves the field of view of the camera.

(b) The background color is similar to the skin color.

(c) Two hands are interacting.

Figure 6.10: Failure cases of the proposed method.

6.6 LIMITATIONS AND DISCUSSION

Figure 6.10 shows failure cases of the proposed method. When the hand is partially cut off by the image boundary due to leaving the field of view of the camera, there should ideally not be any sufficiently confident 2D predictions for the invisible keypoints. However, when these out-of-view keypoints are incorrectly predicted on the image plane, the estimated hand skeleton is pushed inside the field of view. Another difficult scenario for the presented method is when the background has similar appearance as the hand, as *RegNet* struggles to obtain good predictions and thus tracking becomes unstable. Note that this problem is especially severe when using monocular RGB input since there is no depth or multi-view information available to help the disambiguation between hand and background. This can be addressed by using an explicit segmenter, similar to Zimmermann and Brox, 2017. Moreover, when multiple hands are close in the input image, detections may be unreliable. While the proposed approach can handle sufficiently separate hands—due to the bounding box tracker—tracking of interacting hands, or hands of multiple persons, is an interesting direction for follow-up work.

The 3D tracking of hands in purely 2D images is an extremely challenging problem. While the proposed real-time method for 3D hand tracking outperforms state-of-the-art RGB-only methods, there is still an accuracy gap between the achieved results and existing RGB-D methods, as discussed in Section 6.5.3. Nevertheless, the proposed method is an important step towards democratizing RGB-only 3D hand tracking.

6.7 CONCLUSION

Most existing works either consider 2D hand tracking from monocular RGB, or they use additional inputs, such as depth images or multi-view RGB, to track the hand motion in 3D. While the recent method by

Zimmermann and Brox, 2017 tackles monocular 3D hand tracking from RGB images, the proposed approach addresses the same problem but goes one step ahead with regards to several dimensions: the proposed method obtains the *absolute* 3D hand pose by kinematic model fitting, is *more robust* to occlusions, and *generalizes better* due to enrichment of synthetic data such that it resembles the distribution of real hand images. The experimental evaluation demonstrates these benefits as the proposed method significantly outperforms Zimmermann and Brox, 2017, particularly in difficult occlusion scenarios.

Despite impressive results in many cases, the method proposed in this chapter only works for a single hand and fails when two hands overlap in the image as shown in the limitations (Figure 6.10c). Chapter 7 specifically tackles the challenging problem of reconstructing two strongly interacting hands in real time.

7 REAL-TIME POSE AND SHAPE RECONSTRUCTION OF TWO INTERACTING HANDS

Chapter 5 and Chapter 6 have tackled different aspects of challenging scenes for 3D hand reconstruction, namely strong occlusion and cluttered environments, and the inherent ambiguity of monocular RGB input data. Whereas the previously presented methods only work for a single hand, this chapter focuses on a novel method for two strongly interacting hands, even enabling simultaneous reconstruction of pose and shape in real time (published as Mueller et al., 2019).

The proposed approach is the first two-hand tracking solution that combines the following favorable properties: it is marker-less, uses a single consumer-level depth camera, runs in real time, handles inter- and intra-hand collisions, and automatically adjusts to the user's hand shape. In order to achieve this, a recent parametric hand pose and shape model (Romero et al., 2017) and dense correspondence predictions based on a deep neural network are embedded into a suitable energy minimization framework. The correspondence prediction network is trained on the *DenseHands* dataset (Section 4.4), a two-hand dataset which was synthesized based on physical simulations and includes both hand pose and shape annotations while at the same time avoiding inter-hand penetrations. To achieve real-time rates, the model fitting is phrased in terms of a nonlinear least-squares problem so that the energy can be optimized based on a highly efficient GPU-based Gauss-Newton optimizer. State-of-the-art results are shown in scenes that exceed the complexity level demonstrated by previous work, including tight two-hand grasps, significant inter-hand occlusions, and gesture interaction.

7.1 INTRODUCTION

The marker-less estimation of hand poses is a challenging problem that has received a lot of attention in the vision and graphics communities. The relevance of the problem is owed to the fact that hand pose recognition plays an important role in many application areas such as human-computer interaction (Kim et al., 2012), augmented and virtual reality (AR/VR) (Höll et al., 2018), sign language recognition (Koller et al., 2016), as well as body language recognition relevant for psychology. Depending on the particular application, additional requirements are

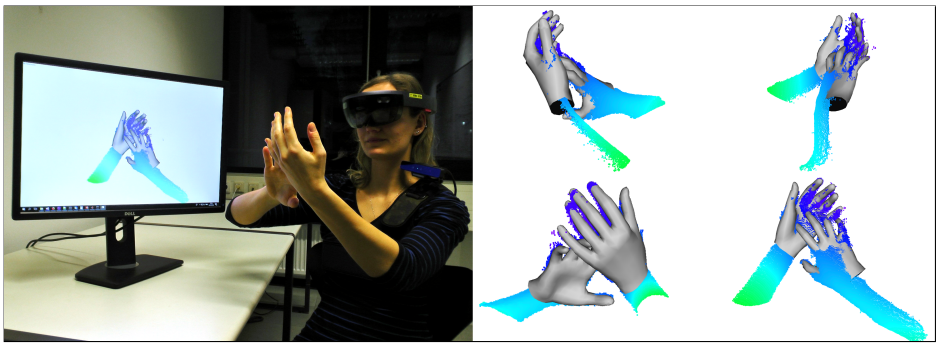


Figure 7.1: The proposed method estimates the pose and shape of two interacting hands in real time from a single depth camera. The picture on the left depicts an AR setup with a shoulder-mounted depth camera. On the right, the depth data and the estimated 3D hand pose and shape from four different views are shown.

frequently imposed on the method, such as performing hand tracking in real time, or dynamically adapting the tracking to person-specific hand shapes for increased accuracy. Ideally, reconstruction should be possible with a simple hardware setup and therefore methods with a single color or depth camera are widely researched. Existing marker-less methods for hand pose estimation typically rely on either RGB (Cai et al., 2018; Mueller et al., 2018; Zimmermann and Brox, 2017), depth images (Sridhar et al., 2015a; Supančič et al., 2018; Taylor et al., 2017; Yuan et al., 2018), or a combination of both (Oikonomidis et al., 2011a; Rogez et al., 2014). The major part of existing methods considers the problem of processing a single hand only (e.g. Oberweger et al., 2015; Qian et al., 2014; Ye and Kim, 2018). Some of them, including the method introduced in Chapter 5, are even able to handle object interactions (Sridhar et al., 2016; Tzionas et al., 2016), which is especially challenging due to potential occlusions.

As humans naturally use both their hands during daily routine tasks, many applications require to track both hands simultaneously (see Figure 7.1), rather than tracking a single hand in isolation. While there are a few existing works that consider the problem of tracking two hands at the same time, they are limited in at least one of the following points: (i) they only work for rather simple interaction scenarios (e.g., no tight two-hand grasps, significant inter-hand occlusions, or gesture interaction), (ii) they are computationally expensive and not real-time capable, (iii) they do not handle collisions between the hands, (iv) they use a person-specific hand model that does not automatically adapt to unseen hand shapes, or (v) they heavily rely on custom-built dedicated hardware. In contrast to existing methods, the proposed approach can handle two hands in interaction while not having any of the limitations (i)-(v), see Table 7.1.

Table 7.1: The proposed method is the first to combine several desirable properties.

	[Oikon. 2012]	[Tzionas 2016]	[Tkach 2017]	[Taylor 2017]	Proposed
Interacting Hands	✓	✓	✗	✓	✓
Shape Estimation	✗	✗	✓	✗	✓
Real Time	✗	✗	✓	✓	✓
Commodity Sensor	✓	✓	✓	✗	✓
Collision Avoidance	✓	✓	✓	✗	✓

This thesis presents for the first time a marker-less method that can track two hands with complex interactions in real time with a single depth camera, while at the same time being able to estimate the person's hand shape. From a technical point of view, this is achieved thanks to a novel learned dense surface correspondence predictor that is combined with a recent parametric hand model (Romero et al., 2017). These two components are combined in an energy minimization framework to find the pose and shape parameters of both hands in a given depth image. Inspired by the recent success of deep learning approaches, especially for image-based prediction tasks (Alp Güler et al., 2017, 2018; Badrinarayanan et al., 2017; Zhang et al., 2017b), a correspondence regressor based on deep neural networks is employed. Compared to ICP-like local optimization approaches, using such a global correspondence predictor is advantageous, as it is less prone to the failures caused by wrong initialization and can easily recover even from severe tracking errors. Since it is not feasible to obtain reliable dense correspondence annotations in real data, synthetic data has to be used. Here, it is crucial to obtain *natural* interactions between the hands, which implies that simply rendering a model of the left and of the right hand (in different poses) into the same view is not sufficient. Thus, the synthetic *DenseHands* dataset, as introduced in Section 4.4, is used. The creation process of *DenseHands* makes use of an extension of the motion capture-driven physical simulation (Verschoor et al., 2018) that leads to faithful, collision-free, and physically plausible simulated hand-hand interactions.

The main contributions of the presented approach are summarized as follows:

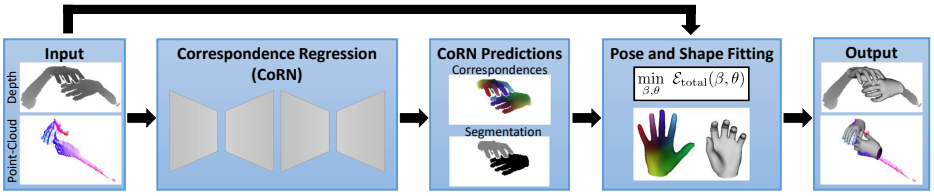


Figure 7.2: Overview of the proposed two-hand pose and shape estimation pipeline. Given only a depth image as input, the dense correspondence regression network (CoRN) computes a left/right segmentation and a vertex-to-pixel map. To obtain the hand shape estimation and pose tracking this data is used in an energy minimization framework, where a parametric hand pose and shape model is fit so that it best explains the input data.

- The first method that can track two interacting hands in real time with a single depth camera, while at the same time being able to estimate the hand shape and taking collisions into account.
- Contrary to existing methods, the proposed approach is more robust and reliable in involved hand-hand interaction settings.

7.2 OVERVIEW

Figure 7.2 shows an overview of the pipeline for performing real-time hand pose and shape reconstruction of two interacting hands from a single depth sensor. The first step is training a neural network that regresses dense correspondences between the hand model and a depth image that depicts two (possibly interacting) hands. In order to disambiguate between pixels that belong to the left hand, and pixels that belong to the right hand, the dense correspondence map also encodes the segmentation of the left and right hand. To ensure realistic training data of hand interactions, the *DenseHands* dataset (Section 4.4) is used, which makes use of motion capture-driven physical simulation to generate (synthetic) depth images along with ground-truth correspondence maps. The *DenseHands* data is additionally augmented with real depth data that is used for training the segmentation channel of the correspondence map. The so-obtained correspondence maps are then used to initialize an energy minimization framework, where a parametric hand model, as described in Section 3.3, is fitted to the given depth data. During fitting, the proposed method uses statistical pose and shape regularizers to avoid implausible configurations, a temporal smoothness regularizer, as well as a collision regularizer in order to avoid interpenetration between both hands and within each hand.

In order to achieve real-time performance, the energy minimization step is phrased in terms of a nonlinear least-squares formulation, and makes use of a highly efficient data-parallel GPU implementation based on a Gauss-Newton optimizer.

The remainder of this section describes the generalization of the single hand MANO model (see Section 3.3) to two hands. Subsequently, a detailed explanation of the dense correspondence regression is provided in Section 7.3, followed by a description of the pose and shape estimation in Section 7.4.

7.2.1 *Two Hand Model*

The MANO model (Romero et al., 2017) is used as 3D hand representation. It is a low-dimensional parametric hand surface model that captures hand shape variation as well as hand pose variation.Details are provided in Section 3.3. Let $\mathcal{V}$ be the 3D mesh vertices of a single hand model where $N_V := |\mathcal{V}| = 778$. Furthermore, let $\mathbf{v} : \mathbb{R}^{N_S} \times \mathbb{R}^{N_P} \rightarrow \mathbb{R}^{3N_V}$ be the function that computes the 3D positions of all of the mesh's N_V vertices, given a shape parameter vector $\boldsymbol{\beta} \in \mathbb{R}^{N_S}$ and pose parameter vector $\boldsymbol{\theta} \in \mathbb{R}^{N_P}$, with $N_S = 10$ and $N_P = 51$. The pose parameter vector contains both the global rigid body pose, as well as the individual articulation parameters. The notation $\mathbf{v}_i(\boldsymbol{\beta}, \boldsymbol{\theta}) \in \mathbb{R}^3$ denotes the 3D position of the i-th vertex.

For tracking two hands that can move independently, independent hand models of the left and right hand are used, which are denoted by $\mathcal{V}_{\text{left}}$ and $\mathcal{V}_{\text{right}}$ with vertices $\mathbf{v}_{\text{left}}(\boldsymbol{\beta}_{\text{left}}, \boldsymbol{\theta}_{\text{left}})$ and $\mathbf{v}_{\text{right}}(\boldsymbol{\beta}_{\text{right}}, \boldsymbol{\theta}_{\text{right}})$, respectively. For notational convenience, the parameters of the left and right hand are stacked so that $\boldsymbol{\beta} = (\boldsymbol{\beta}_{\text{left}}, \boldsymbol{\beta}_{\text{right}})$ and $\boldsymbol{\theta} = (\boldsymbol{\theta}_{\text{left}}, \boldsymbol{\theta}_{\text{right}})$, and $\mathcal{V}$ with $N_v := |\mathcal{V}| = 2{\cdot}778$ denotes the combined vertices of the left and the right hand.

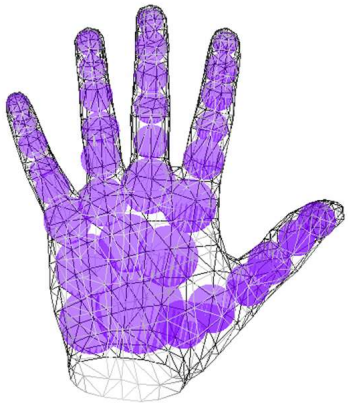


Figure 7.3: Illustration of hand model mesh with the collision proxies.

To resolve interpenetrations at high computational efficiency, collision proxies are added to the hand model. The volumetric extent of the hand is approximated with a set of spheres that are modeled with 3D Gaussians as employed by previous work (Sridhar et al., 2015a) and explained in Section 3.2. Using this formulation, interpenetrations can then be avoided by penalizing the overlap of the Gaussians during pose optimization. Note that the

overlap between Gaussians is differentiable and can be computed in closed form—in contrast to naïve binary collision checking. The Gaussians are combined with the existing MANO model by rigging their positions to the hand joints and coupling their standard deviations to pairs of manually selected vertices. Doing this ensures that the position and size of the Gaussians vary in accordance with the pose and shape parameters $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$. For each hand 35 3D Gaussians are added, which leads to a total number of $N_C = 70$ for the combined two-hands model. A visualization of the isosurface at 1 standard deviation of the Gaussians is shown in Figure 7.3. Next, the correspondence regressor is described that is eventually coupled with the two-hands model in order to perform pose and shape reconstruction.

7.3 DENSE CORRESPONDENCE REGRESSION

Let $\mathcal{I}$ be the input depth image of pixel-dimension h by w defined over the image domain Ω. The goal is to learn a vertex-to-pixel correspondence map $c : \mathcal{V} \to \bar{\Omega}$ that assigns to each vertex of the model $\mathcal{V}$ a corresponding pixel of $\mathcal{I}$ in the image domain Ω. In order to allow the possibility to not assign an image pixel to a vertex (i.e. a vertex currently not visible), the set Ω is extended to also include $\emptyset$, which is denoted by $\bar{\Omega}$.

7.3.1 *Obtaining Vertex-to-Pixel Mappings*

To obtain the vertex-to-pixel correspondence map c, a dense correspondence image $\mathcal{N} : \Omega \to [0, 1]^4$, as defined in Section 4.4, is used. For each image pixel, it contains a 4-channel feature value, the dense correspondence encoding, that uniquely encodes the 3D surface point of the hand model, which is visible at this pixel. To obtain a vertex-to-pixel correspondence map c for a given depth image $\mathcal{I}$, the image pixels are first mapped to the dense correspondence encoding space using $\mathcal{N}$ (a function over the image domain). Subsequently, the per-pixel values obtained through $\mathcal{N}$ are compared with the fixed dense correspondence encoding η defined over the hand model surface (see Section 4.4.1). The vertex-to-pixel maps are constructed by a thresholded nearest-neighbor strategy for all vertices $i \in \mathcal{V}$ as follows

$$\hat{c}(i) = \underset{j \in \Omega}{\operatorname{argmin}} \quad \|\mathcal{N}(j) - \eta(i)\|_2 \,, \text{ and} \tag{7.1}$$

$$c(i) = \begin{cases} \hat{c}(i) & \text{if } \|\mathcal{N}(\hat{c}(i)) - \eta(i)\|_2 < t_c \\ \emptyset & \text{otherwise} \end{cases} . \tag{7.2}$$

If the closest predicted value for some vertex i is larger than the empirically chosen threshold t_c=0.04, this vertex is likely to be invisible in the input depth image.

7.3.2 *Training Data*

Since it is not possible to annotate dense correspondences on real data, the synthetically generated *DenseHands* dataset is used. This dataset contains realistic two-hand interactions that were generated by a live motion-capture-driven physical simulation framework (see Section 4.4). Nevertheless, when only trained with synthetic data, neural networks tend to overfit and hence may not generalize well to real test data. To overcome this, real depth camera footage of hands was integrated into the so far synthetically generated training set. Since it is infeasible to obtain dense correspondence annotations on real data, the annotation on real data is restricted to the left/right hand segmentation task. Because the proposed approach only uses the depth input channel as input data, the color channel can be instrumented for automatic segmentation annotation. As body paint (Soliman et al., 2018; Tompson et al., 2014) has less influence on the observed hand shape, in contrast to colored gloves (Taylor et al., 2017), body paint is used to obtain reliable annotations by color segmentation in the RGB image provided by the depth camera. Please refer to Section 4.3 for more details on automatic color-based segmentation annotation in depth images. In total, 3 users (1 female, 2 male) with varying hand shapes (width: 8–10cm, length: 17–20.5cm) were captured. Approximately $3,000$ images were recorded per subject and viewpoint (shoulder-mounted camera and frontal camera), resulting in a total number of $19,926$ images.

7.3.3 *Neural Network Regressor*

Based on the mixed real and synthetic training data described in Section 7.3.2, a neural network is trained to learn the dense correspondence image mapping $\mathcal{N}$, as depicted in Figure 7.4. Inspired by recent architectures used for per-pixel predictions (Newell et al., 2016; Ronneberger et al., 2015), the proposed network comprises two stacked encoder-decoder processing blocks. The first block is trained to learn the segmentation task, i.e. it outputs per-class probability maps in the original input resolution for the three possible classes $\{\text{left}, \text{right}, \text{non-hand}\}$. These class probability maps are concatenated with the input depth image $\mathcal{I}$ and fed into the second encoder-decoder to regress the 3-channel per-pixel hand surface correspondence information. The final mapping $\mathcal{N} : \Omega \rightarrow [0,1]^4$

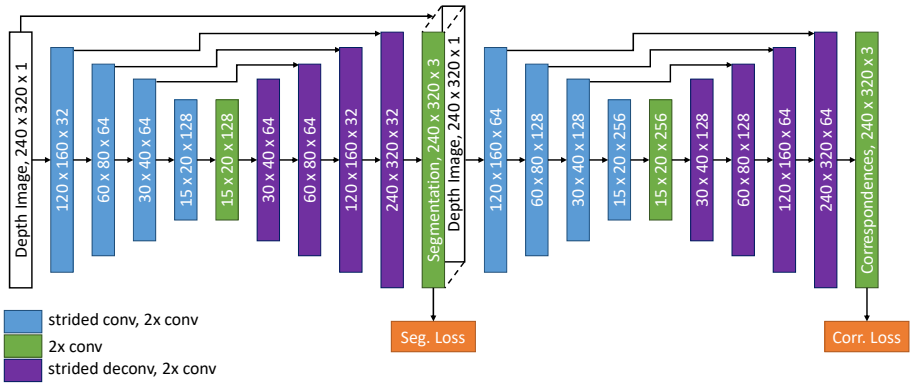


Figure 7.4: The proposed correspondence regression network (CoRN) consists of two stacked encoder-decoder networks. The output sizes of the layer blocks are specified as height $\times$ width $\times$ number of feature channels. In addition, the colors of the layer blocks indicate which operations are performed (best viewed in color).

is then obtained by concatenating the correspondence output with the label of the most likely class for each pixel. Note that the class labels are scaled to also match the range $[0, 1]$ by setting left $= 0$, right $= 0.5$, and non-hand $= 1$. Both encoder-decoder subnetworks share the same architecture. The resolution is downsampled using convolutions with stride 2 and upsampled with the symmetric operation, deconvolutions with stride 2. Note that every convolution and deconvolution is followed by batch normalization and rectified linear unit (ReLU) layers. In addition, skip connections are used to preserve spatially localized information and enhance gradient backpropagation. Since the second subnetwork needs to learn a harder task, the number of feature maps are doubled in all layers. The segmentation loss is formulated as the softmax cross entropy, a standard classification loss. For the correspondence loss, the squared Euclidean distance is used as is common in regression tasks. The complete network is trained end-to-end, with mixed data batches containing both synthetic and real samples in one training iteration. For the latter, only the segmentation loss is active. For more details on the training procedure, please refer to Appendix A.4.

7.4 POSE AND SHAPE ESTIMATION

The pose and shape of the hands present in image $\mathcal{I}$ are estimated by fitting the hand surface model (see Section 7.2.1) to the depth image data. In the first step, the foreground point-cloud $\{\mathbf{d}_j \in \mathbb{R}^3\}_{j=1}^{N_I}$ in the

depth image $\mathcal{I}$ is extracted, along with the respective point-cloud normals $\{\mathbf{n}_j \in \mathbb{R}^3\}_{j=1}^{N_I}$ obtained by Sobel filtering. Based on the assumption that the hands and arms are the objects closest to the camera, the foreground is extracted using a simple depth-based thresholding strategy, where N_I denotes the total number of foreground pixels (of both hands together). Subsequently, this point-cloud data is used in conjunction with the learned vertex-to-pixel correspondence map c within an optimization framework. By minimizing a suitable nonlinear least-squares energy function, which will be defined next, the hand model parameters that best explain the point-cloud data are determined.

The total energy for both the left and the right hand is defined as

$$\mathcal{E}_{\text{total}}(\boldsymbol{\beta}, \boldsymbol{\theta}) = \mathcal{E}_{\text{data}}(\boldsymbol{\beta}, \boldsymbol{\theta}) + \mathcal{E}_{\text{reg}}(\boldsymbol{\beta}, \boldsymbol{\theta}) \,, \tag{7.3}$$

where $\boldsymbol{\beta}$ are the shape parameters and $\boldsymbol{\theta}$ are the hand pose parameters, as described in Section 7.2.1. The data term $\mathcal{E}_{\text{data}}$ measures for a given parameter tuple $(\boldsymbol{\beta}, \boldsymbol{\theta})$ how well the hand model explains the depth image $\mathcal{I}$, and the term $\mathcal{E}_{\text{reg}}$ is a regularizer that accounts for temporal smoothness, plausible hand shapes and poses, as well as avoiding interpenetrations within and between the hands.

7.4.1 *Data Term*

The data term is based on a combination of a point-to-point and a point-to-plane term as

$$\mathcal{E}_{\text{data}}(\boldsymbol{\beta}, \boldsymbol{\theta}) = \omega_{\text{point}} E_{\text{point}}(\boldsymbol{\beta}, \boldsymbol{\theta}) + \omega_{\text{plane}} E_{\text{plane}}(\boldsymbol{\beta}, \boldsymbol{\theta}) \,, \tag{7.4}$$

where $\omega_{\odot}$ is used to denote the relative weights of the terms.

POINT-TO-POINT. Let γ_i be the visibility indicator for the i-th vertex, which is defined to be 1 if $c(i) \neq \emptyset$, and 0 otherwise. The point-to-point energy measures the distances between all visible model vertices $\mathbf{v}_i(\boldsymbol{\beta}, \boldsymbol{\theta})$ and the *corresponding* 3D point at pixel $c(i)$, denoted as $\mathbf{d}_{c(i)}$, and is defined as

$$E_{\text{point}}(\boldsymbol{\beta}, \boldsymbol{\theta}) = \sum_{i=1}^{N_\mathcal{V}} \gamma_i ||\mathbf{v}_i(\boldsymbol{\beta}, \boldsymbol{\theta}) - \mathbf{d}_{c(i)}||_2^2 \,. \tag{7.5}$$

POINT-TO-PLANE. The point-to-plane energy is used to penalize the deviation from the model vertices $\mathbf{v}_i(\boldsymbol{\beta}, \boldsymbol{\theta})$ and the point-cloud surface tangent, and is defined as

$$E_{\text{plane}}(\boldsymbol{\beta}, \boldsymbol{\theta}) = \sum_{i=1}^{N_\mathcal{V}} \gamma_i \left\langle \mathbf{v}_i(\boldsymbol{\beta}, \boldsymbol{\theta}) - \mathbf{d}_{c(i)}, \mathbf{n}_{c(i)} \right\rangle^2 \,, \tag{7.6}$$

where $\mathbf{n}_{\odot}$ denotes the input normal at a specific pixel obtained by Sobel filtering.

7.4.2 *Regularizer*

The regularizer $\mathcal{E}_{\text{reg}}$ comprises statistical pose and shape regularization terms, a temporal smoothness term, as well as a collision term. It is defined as

$$\begin{aligned}\mathcal{E}_{\text{reg}}(\boldsymbol{\beta},\boldsymbol{\theta}) =& \omega_{\text{shape}} E_{\text{shape}}(\boldsymbol{\beta}) + \omega_{\text{pose}} E_{\text{pose}}(\boldsymbol{\theta}) && (7.7)\\ & \omega_{\text{temp}} E_{\text{temp}}(\boldsymbol{\beta},\boldsymbol{\theta}) + \omega_{\text{coll}} E_{\text{coll}}(\boldsymbol{\beta},\boldsymbol{\theta})\,. && (7.8)\end{aligned}$$

STATISTICAL REGULARIZERS. The MANO model is parameterized in terms of a low-dimensional linear subspace obtained via PCA. Hence, in order to impose a plausible pose and shape at each captured frame, the Tikhonov regularizers are employed as

$$E_{\text{shape}}(\boldsymbol{\beta}) = ||\boldsymbol{\beta}||_2^2 \quad \text{and} \quad E_{\text{pose}}(\boldsymbol{\theta}) = ||\boldsymbol{\theta}||_2^2\,. \tag{7.9}$$

Note that E_{pose} is only employed for the part of $\boldsymbol{\theta}$ that contains the pose PCA coefficients, not for the global rigid transform.

TEMPORAL REGULARIZER. In order to achieve temporal smoothness, a zero velocity prior is used on the shape parameters $\boldsymbol{\beta}$ and the pose parameters $\boldsymbol{\theta}$, i.e.

$$E_{\text{temp}}(\boldsymbol{\beta},\boldsymbol{\theta}) = ||\boldsymbol{\beta}^{(t)} - \boldsymbol{\beta}^{(t-1)}||_2^2 + ||\boldsymbol{\theta}^{(t)} - \boldsymbol{\theta}^{(t-1)}||_2^2\,. \tag{7.10}$$

COLLISION REGULARIZER. In order to avoid interpenetration within individual hands, as well as interpenetrations between the left and the right hand, a collision energy term is added. As described in Section 7.2.1, spherical collision proxies are placed inside each hand mesh, and then overlaps between these collision proxies are penalized. Mathematically, this is phrased based on the overlap of (isotropic) Gaussians (Sridhar et al., 2015a), which results in soft collision proxies defined as smooth occupancy functions. The energy reads

$$E_{\text{coll}}(\boldsymbol{\beta},\boldsymbol{\theta}) = \sum_{p=1}^{N_C} \sum_{q=p+1}^{N_C} \int_{\mathbb{R}^3} G_p(x;\boldsymbol{\beta},\boldsymbol{\theta}) \cdot G_q(x;\boldsymbol{\beta},\boldsymbol{\theta})\, dx\,. \tag{7.11}$$

Here, G_p, G_q denote the Gaussian collision proxies whose mean $\boldsymbol{\mu}$ depends on the pose and shape parameters $\boldsymbol{\beta},\boldsymbol{\theta}$, whereas the standard deviation σ only depends on the shape $\boldsymbol{\beta}$. Please refer to Section 3.2 for more details about the volumetric Gaussian hand model, including the closed-form expression for E_{coll}.

7.4.3 *Optimization*

The energy $\mathcal{E}_{\text{total}}$ is phrased in terms of a nonlinear least-squares formulation, so that it is amenable to be optimized based on the Gauss-Newton algorithm. All derivatives of the residuals can be computed analytically, so that all entries of the Jacobian can be computed efficiently on the GPU with high accuracy. Please note that the derivative $\frac{\partial E_{\text{coll}}(\boldsymbol{\beta},\boldsymbol{\theta})}{\partial \boldsymbol{\beta}}$ is not used in the optimization since this encourages shrinking of the hand models when they are interacting. Instead, the shape $\boldsymbol{\beta}$ is optimized using all other energy terms and the Gaussian parameters are updated according to $\boldsymbol{\beta}$ in every optimizer iteration. More details on the GPU implementation can be found in Appendix B.

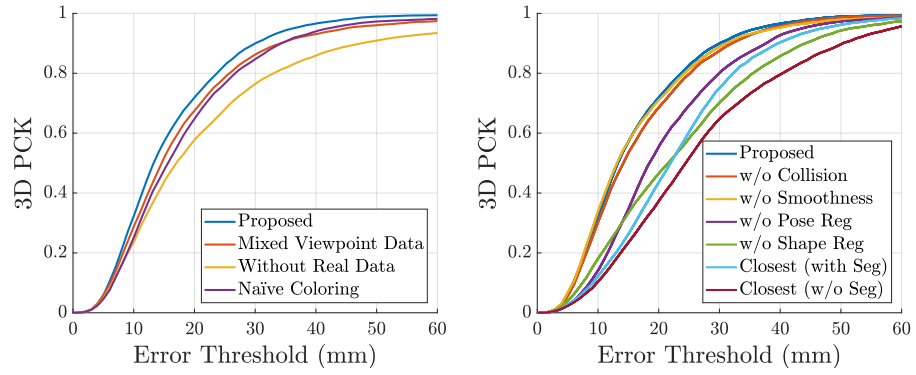
Note that although in principle it would be sufficient to optimize for the shape parameter once per actor and then keep it fixed throughout the sequence, the shape optimization is performed in each frame of the sequence. This has the advantage that a poorly chosen frame for estimating the shape parameter does not have a negative impact on the tracking of subsequent frames. Empirical findings indicate that the hand shape is robust and does not significantly change throughout a given sequence.

7.5 EVALUATION

In this section the proposed two-hand tracking approach is thoroughly evaluated. Section 7.5.1 includes additional implementation details. Subsequently, in Section 7.5.2, an ablation study is presented, followed by a comparison to state-of-the-art tracking methods in Section 7.5.3. Eventually, in Section 7.5.4, additional results are provided, including an experiment that demonstrates the ability of the method to adapt to user-specific hand shapes.

7.5.1 *Implementation*

The implementation runs on two GPUs of type NVIDIA GTX 1080 Ti. One GPU runs the correspondence regression network CoRN, as well as the per-vertex correspondence matching for frame $t + 1$, while the other GPU runs the model optimization for frame t. Overall, 30 fps are achieved using an implementation based on C++, CUDA, and the Tensorflow library. A depth camera Intel RealSense SR300 is used for the real-time results and evaluation. Section 7.5.3 also demonstrates results when using a publicly available dataset that was captured with a different sensor.



(a) (b)

Figure 7.5: Results of the ablation study. (a) shows different configurations regarding the correspondence regressor (CoRN). (b) shows configurations regarding the optimizer.

Unless stated otherwise, CoRN training always uses synthetic and real images (cf. Section 7.3.2) rendered and recorded from a *frontal* viewpoint. It should be emphasized that it is reasonable to use view-specific correspondence regressors as for a given application it is usually known from which view-point the hands are to be tracked.

7.5.2 *Ablation Study*

A detailed ablation study was conducted, where the effects of the individual components of the proposed approach are analyzed. For these evaluations the dataset provided by Tzionas et al., 2016 is used, which comes with annotations of the joint positions on the depth images. Figure 7.5 shows quantitative results of the analysis for a range of different configurations. To this end, the *percentage of correct keypoints* (PCK) is used as measure, where the horizontal axis shows the error, and the vertical axis indicates the percentage of points that fall within this error. To compute the PCK, the same set of keypoints as in Tzionas et al., 2016 is considered. Notice that despite using Tzionas et al.'s dataset, Figure 7.5 does not show their results because they do not provide 3D PCK values. Qualitative results of the ablation study are shown in Figure 7.6.

CORRESPONDENCE REGRESSION NETWORK. In Figure 7.5a four settings of different configurations for training the correspondence regressor (CoRN) are presented:

(1) The proposed CoRN network as explained in Section 7.3 (blue ■, "Proposed").

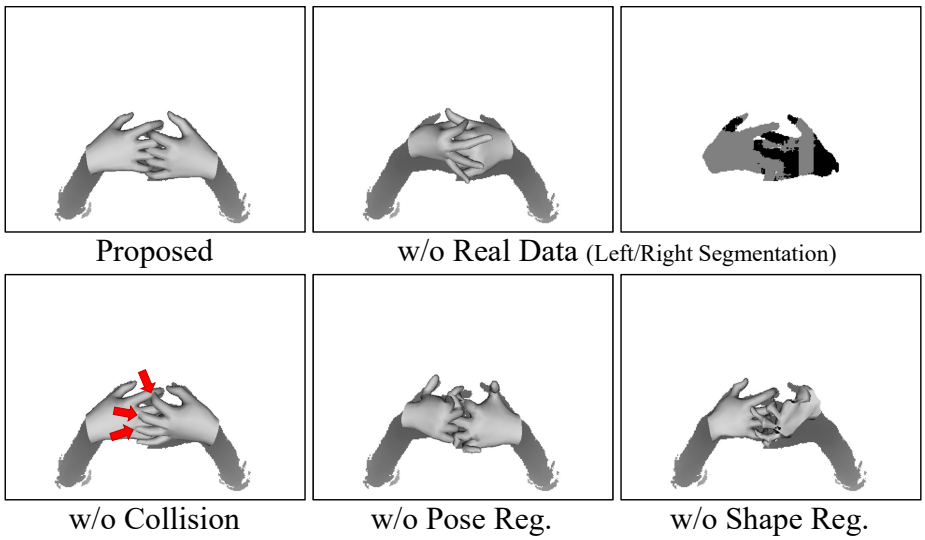


Figure 7.6: Qualitative examples from the ablation study.

(2) The CoRN network but trained based on data from two viewpoints, egocentric as well as frontal (orange ■, "Mixed Viewpoint Data").

(3) The CoRN network that is trained only with synthetic data, i.e., without using any real data as described in Section 7.3.2 in order to train the segmentation sub-network (yellow ■, "Without Real Data").

(4) Instead of using the proposed geodesic HSV embedding as dense correspondence encoding (cf. Section 4.4.1), the naïve encoding by mapping the original mesh onto the RGB cube is used (purple ■, "Naïve Coloring").

It can be seen that the proposed training setting outperforms all other settings.

POSE AND SHAPE ESTIMATION. In Figure 7.5b different optimizer configurations are shown. Five versions of the energy are evaluated:

(1) The complete energy $\mathcal{E}_{\text{total}}$ that includes all terms (blue ■, "Proposed").

(2) The energy without the collision term E_{coll} (orange ■, "w/o Collision").

(3) The energy without the temporal smoothness term E_{temp} (yellow ■, "w/o Smoothness").

(4) The energy without the pose regularizer E_{pose} (purple ■, "w/o Pose Reg").

(5) The energy without the pose regularizer E_{shape} (green ■, "w/o Shape Reg").

In addition, to demonstrate the importance of CoRN, comparisons to two configurations using closest point correspondences instead are performed:

(6) Finding the vertex correspondence as the closest input point that was classified with the same handedness (light blue ■, "Closest (with Seg)").

(7) Finding the vertex correspondence as the closest input point in the whole point cloud (dark red ■, "Closest (w/o Seg)").

For these two configurations, the hand models were manually initialized as close as possible in the first frame to enable a fair comparison. Note that this is not necessary with CoRN which provides global correspondences across the full frame.

It can be observed that the complete energy performs best, compared to leaving individual terms out. Moreover, removing the pose regularizer or the shape regularizer worsens the outcome significantly more compared to dropping the collision or the smoothness terms when looking at the PCK. Note that the smoothness term removes temporal jitter that is only marginally reflected by the numbers. Similarly, while removing the collision term does not affect the PCK significantly, it severely worsens the results perceptually. Using naïve closest points instead of predicted CoRN correspondences results in significantly higher errors, this holds for both versions, with and without segmentation information. Additionally, Figure 7.6 shows qualitative examples from the ablation study that further validate that each term of the complete energy formulation is essential to obtain high quality tracking of hand-hand interaction.

INDEPENDENCE OF INITIALIZATION. Figure 7.7 shows that the proposed hand tracker is able to recover from severe errors that occur when the hand motion is extremely fast, so that the depth image becomes blurry. In this scenario, as soon as the hand moves with a normal speed again, the tracker is able to recover and provide an accurate tracking. Note that this is in contrast to local optimization approaches (e.g., based on an ICP-like procedure for pose and shape fitting) that cannot recover from bad results due to severe non-convexity of the energy landscape.

7.5.3 *Comparison to the State of the Art*

Next, the proposed method is compared to state-of-the-art methods.

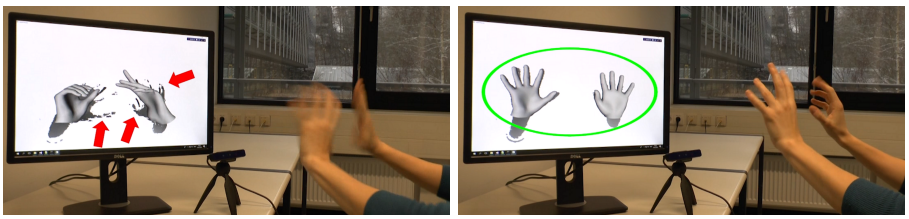


Figure 7.7: The proposed method may fail, e.g., when the depth image is severely blurred (left). However, due to the per-frame CoRN predictions, it instantly recovers when the image quality improves again.

COMPARISON TO TZIONAS ET AL., 2016. Table 7.2 presents results of the quantitative comparison to the work of Tzionas et al., 2016. The evaluation is based on their two-hand dataset that comes with joint annotations. As shown, the relative 2D pixel error is very small in both methods. While it is slightly higher with the proposed approach, it has to be emphasized that it achieves a 150× speed-up and does not require a user-specific hand model. Furthermore, Figure 7.8 qualitatively shows that the precision error difference does not result in any noticeable visual quality gap. Moreover, it should be pointed out that the finger tip detection method of Tzionas et al., 2016 is ad-hoc trained for their specific camera, whereas the proposed correspondence regressor has never seen data from the depth sensor used in this comparison.

Table 7.2: The proposed method is compared to the method by Tzionas et al., 2016 on their provided dataset. The table shows the average and standard deviation of the 2D pixel error (relative to the diagonal image dimension), as well as the per-frame runtime. Note that the pixel errors of both methods are very small, and that the proposed method is 150× faster. Moreover, the proposed approach automatically adjusts to the user-specific hand shape, whereas Tzionas et al. require a 3D scanned hand model.

	2D Error	**Runtime**	**Shape Estimation**
Proposed	1.35±0.28 %	33ms	✓
Tzionas *et al.*	0.63±0.12 %	4960ms	✗

COMPARISON TO LEAPMOTION, 2016. While the commercial solution Leap Motion successfully tracks two hands when they are spatially separated by a significant offset, it struggles and fails for complex hand-hand interactions and does not recover well from errors (see Figure 7.9).

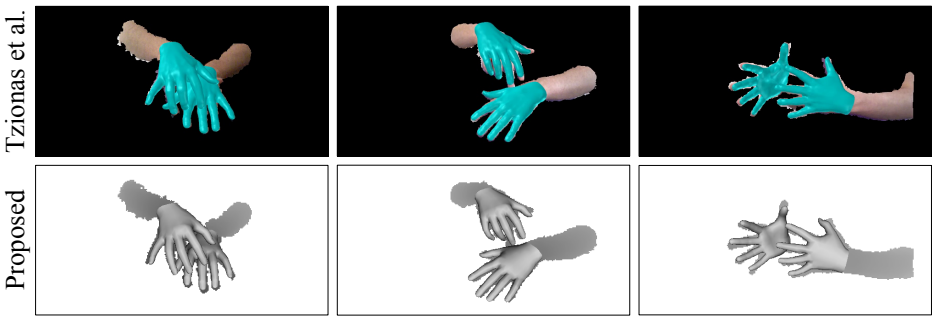


Figure 7.8: Qualitative comparison with Tzionas et al., 2016. The proposed method achieves results with comparable visual quality while running multiple orders of magnitude faster.

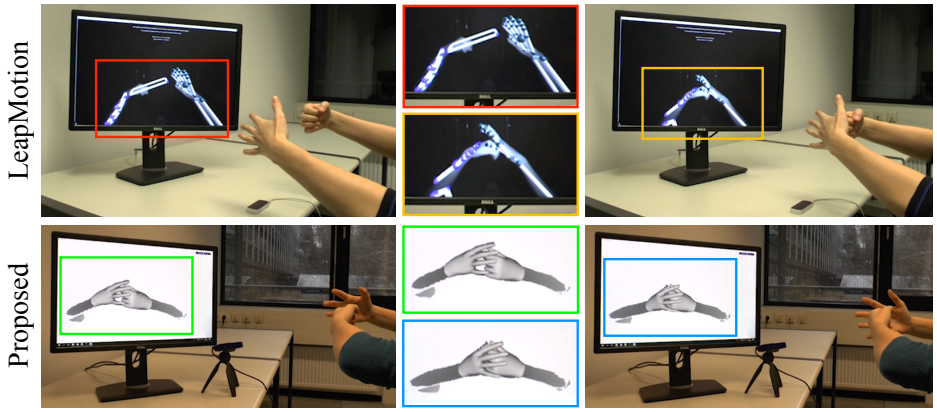


Figure 7.9: Qualitative comparison with LeapMotion, 2016. LeapMotion (top) fails when the hands are interacting. Note that it does not recover from the error, namely the flipped hand model, when the hands move apart again (top left). In contrast, the proposed method (bottom) successfully tracks the interaction.

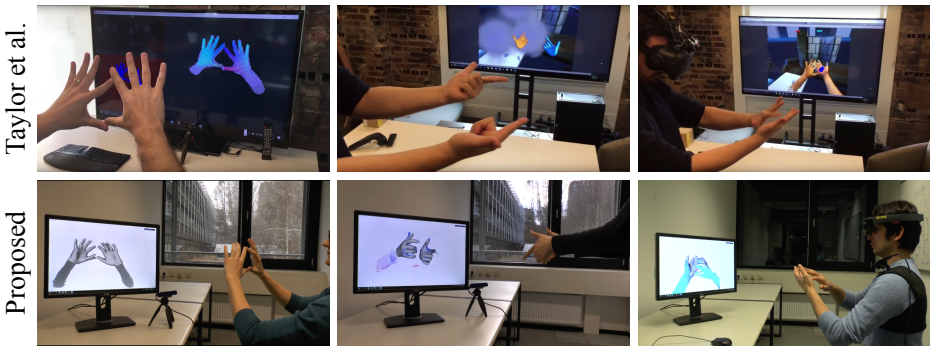


Figure 7.10: Qualitative comparison to Taylor et al., 2017. The proposed method is able to track two hands in similar poses while at the same time reconstructing shape automatically and avoiding collisions.

In contrast, the proposed approach is able to not only successfully track challenging hand-hand interactions, but also estimate the 3D hand shape.

OTHER METHODS. Since the authors of Taylor et al., 2017 did not release their dataset, the presented method could not directly be compared with their results. Nevertheless, the proposed method shows accurate tracking results on similar scenes (Figure 7.10), as well as some settings that are arguably more challenging than theirs (Figure 7.12).

7.5.4 *More Results*

This section presents additional results on hand shape adaption as well as additional qualitative results.

HAND SHAPE ADAPTATION. Here, the adaptation to user-specific hand shapes is investigated. Figure 7.11 shows the obtained hand shape when running the proposed method for four different persons with varying hand shapes. It can be seen that the proposed method is able to adjust the geometry of the hand model to the users' hand shapes.

Due to the severe difficulty in obtaining reliable 3D ground-truth data and disentangling shape and pose parameters, direct quantitative evaluation of the hand shape is not possible. Instead, the consistency of the estimated bone lengths on the sequences of Tzionas et al., 2016 is evaluated in addition. The average standard deviation is 0.6 mm, which indicates that the shape estimation is stable over time.

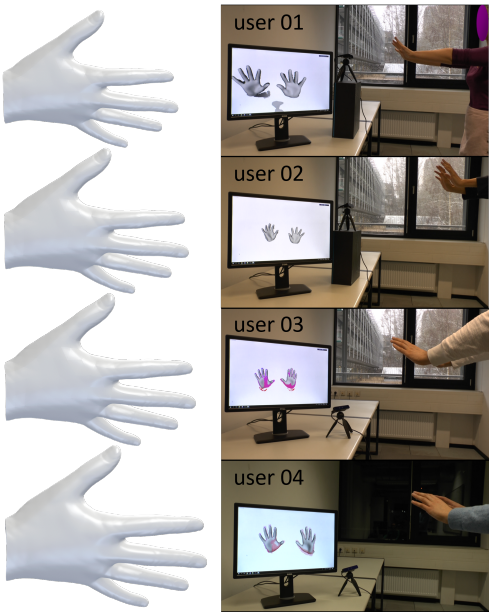


Figure 7.11: The presented 3D hand models (left) were obtained from fitting the model to different users with varying hand shape using the proposed approach. Small to large hand shapes are shown from top to bottom. Note that all four hand shapes on the left are shown in the same pose in order to allow for a direct comparison.

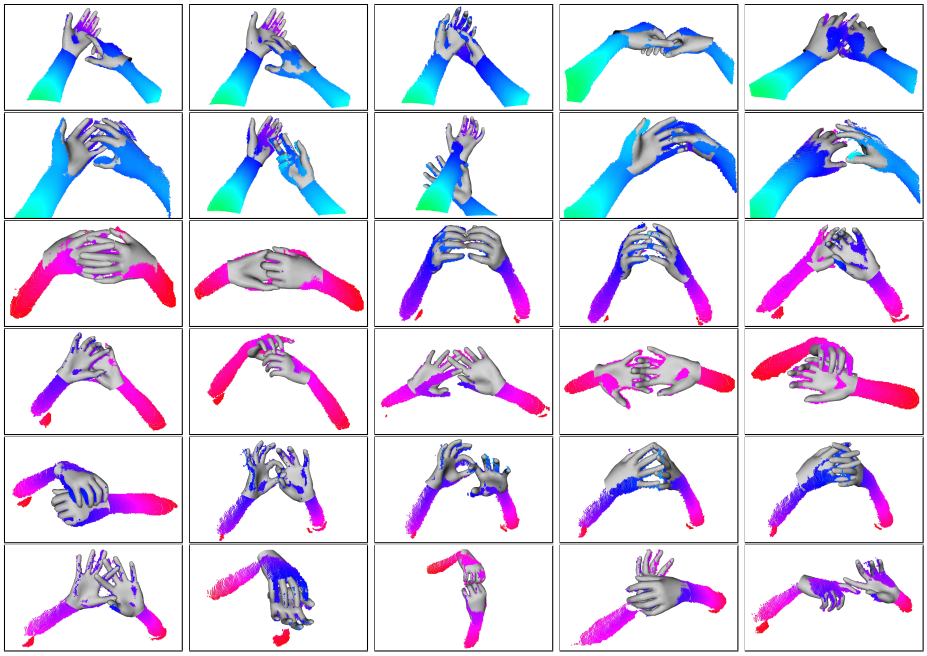


Figure 7.12: Qualitative results for the proposed method. Note that the different colors of the depth image are due to different absolute depth values. The top two rows depict egocentric viewpoints, whereas the bottom four rows show 3rd-person viewpoints.

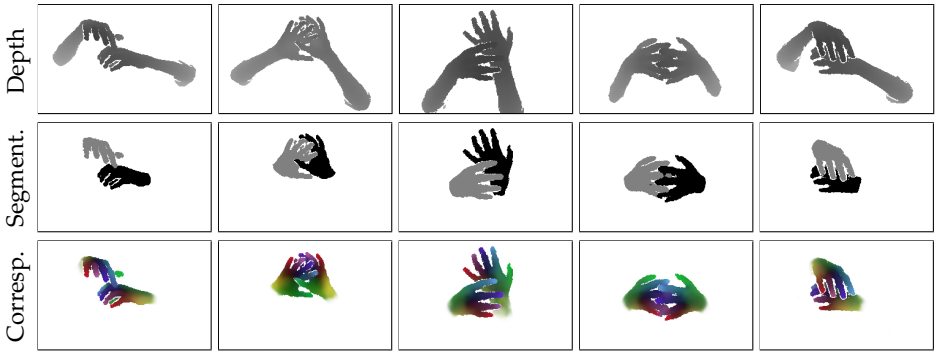


Figure 7.13: Given a depth image (top) as input, CoRN produces accurate segmentation (middle) and dense correspondences (bottom).

QUALITATIVE RESULTS. Figure 7.12 presents qualitative results of the proposed pose and shape estimation method. The first two rows show frames for an egocentric viewpoint, where CoRN was also trained for this setting, whereas the remaining rows show frames for a frontal viewpoint. It can be seen that in a wide range of complex hand-hand interactions the proposed method robustly estimates the hand pose and shape. CoRN is an essential part of the method and is able to accurately predict segmentation and dense correspondences for a variety of inputs (see Figure 7.13). However, wrong predictions may lead to errors in the final tracking result as demonstrated in Figure 7.14.

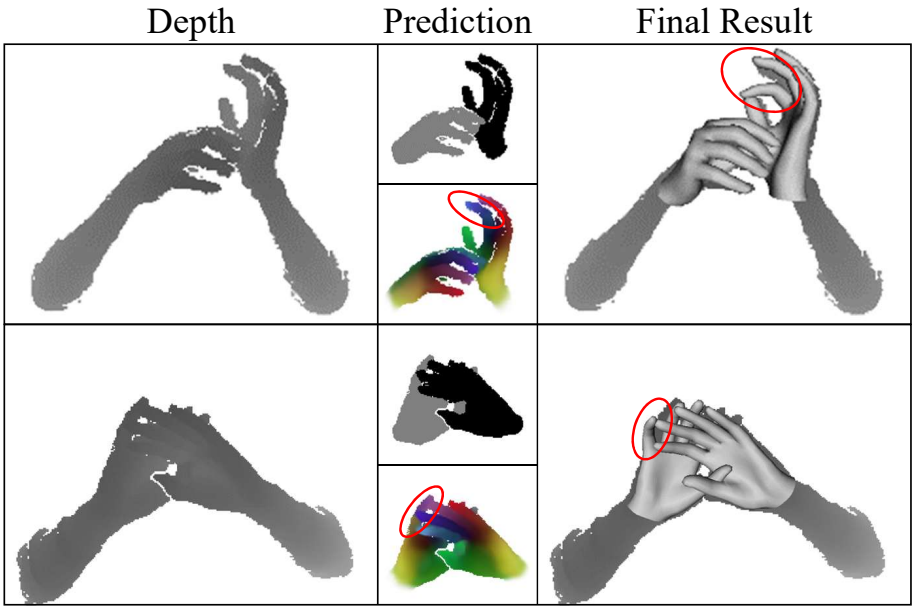


Figure 7.14: Erroneous CoRN predictions, e.g., wrongly classified fingers, negatively impact the final tracking result (see Figure 4.9 for the reference coloring).

7.6 LIMITATIONS AND DISCUSSION

Although overall compelling results for the estimation of hand pose and shape in real-time were demonstrated, there are several points that leave room for further improvements. While the presented approach better handles complex hand-hand interactions compared to previous real-time approaches, the method may still struggle in very challenging situations. For example, this may happen when the user performs extremely fast hand motions that lead to a severely blurred depth image (see Figure 7.7), or when one of the hands is mostly occluded.

TEMPORAL COHERENCE. In case of strong occlusions, temporal jitter may occur due to the insufficient information in the depth image. This could be mitigated by a more involved temporal smoothness term, e.g., stronger smoothing when occlusions are happening, or a temporal network architecture for correspondence prediction. Also, the current temporal smoothness prior may cause a delay in the tracking for large inter-frame motion.

CORRESPONDENCE MATCHING. To further improve the quality of the results, in the future one can use more elaborate strategies for finding correspondences, e.g., by using matching methods that are more advanced than nearest-neighbor search, or by incorporating confidence estimates in the correspondence predictor.

DATA GENERATION. Although the data generation scheme for *DenseHands* (as described in Section 4.4) has proven successful for training CoRN, some generated images might not be completely realistic. This is due to the LeapMotion tracker's limitations and the hence mandatory distance between the two real hands. In future work, the proposed method could drive the simulation, and the data could be iteratively refined by bootstrapping.

DETAILED HAND SHAPE. While the proposed approach is the only real-time approach that can automatically adjust to user-specific hand shapes, the obtained hand shapes are not as detailed as high-quality laser scans. On the one hand, this is because the MANO model (Romero et al., 2017) is rather coarse with its 778 vertices per hand, and on the other hand the depth image is generally of lower resolution compared to laser scans.

HANDS AND OBJECTS. One relevant direction for future works is to deal with two hands that manipulate an object. Particular challenges are that one additionally needs to separate the object from the hands, as well as being able to cope with more severe occlusions due to the object. In addition, estimated hand-object configurations should result in physically plausible and stable grasps.

PHYSICS SIMULATION FOR MODEL FITTING. Another point that is left for future work is to also integrate a physics simulation step directly into the tracker, so that at run-time one can immediately take fine-scale collisions into account. Currently, slight intersections may still happen due to the computationally efficient but coarse collision proxies.

COMPUTATIONAL COST. In terms of computational cost, currently the setup depends on two high-end GPUs, one for the regression network and one for the optimizer. In order to achieve a computationally more light-weight processing pipeline, one could consider lighter neural network architectures, such as CNNs tailored towards mobile platforms (e.g. Howard et al., 2017).

7.7 CONCLUSION

This chapter presented a method for real-time pose and shape reconstruction of two interacting hands. The main features that distinguish the proposed method from previous two-hand tracking approaches is that it combines a wide range of favorable properties, namely it is marker-less, relies on a single depth camera, handles collisions, runs in real time with a commodity camera, and adjusts to user-specific hand shapes. This is achieved by combining a neural network for the prediction of correspondences with an energy minimization framework that optimizes for hand pose and shape parameters. For training the correspondence regression network, the *DenseHands* dataset was used. It was created by leveraging physics-based simulation for generating (annotated) synthetic training data that contains physically plausible interactions between two hands. The addition of real depth data, annotated for the segmentation task only, further improved the performance. Due to a highly efficient GPU-based implementation of the energy minimization based on a Gauss-Newton optimizer, the approach is real-time capable. It was experimentally shown that the approach achieves results that are qualitatively similar and quantitatively close to the two-hand tracking solution by Tzionas et al., 2016, while at the same time being two orders of magnitude faster. Moreover, it was qualitatively demonstrated that the proposed method can handle more complex hand-hand interactions compared to recent state-of-the-art hand trackers.

This chapter and the previous two chapters have focused on developing state-of-the-art 3D hand motion capture and reconstruction methods that run in real time and only require a single color or depth camera. The availability of such methods enables many potential applications, for example in robotics, activity recognition, or human–computer interaction. The next chapter will put emphasis on one important application and present a novel prototype system for the accurate recognition of thumb-to-finger microgestures.

8 FINGERINPUT

Whereas the previous chapters have presented methods for accurate 3D hand motion capture and reconstruction in real time, this chapter shifts the focus towards applications enabled by these methods. In particular, this chapter introduces *FingerInput*, an application of a 3D hand tracking method for fine-grained thumb-to-finger microgesture detection (published as Soliman et al., 2018). Please note that the contributions of this thesis lie in the development of the hand tracking system and the precise reconstruction of touch points from a single body-mounted depth sensor, which is described in Section 8.4. It is the first system that accurately detects the touch points between fingers as well as the finger flexion. For proper context, additional information about the gesture design space is included in Section 8.3 but that part of the research is not a contribution of this thesis.

8.1 INTRODUCTION

There has been a growing interest in thumb-to-finger gestures, which take advantage of the inherent fine motor skills of fingers to allow users to expressively control digital systems (Chan et al., 2016; Chan et al., 2013; Huang et al., 2016; Lien et al., 2016). By touching one or multiple fingers with the thumb, the user can perform touch input directly on the skin. This promises to be a very direct, fast, and discreet type of input, even more as it supports one-handed and eyes-free interactions. However, sensing such thumb-to-finger gestures is hard because these gestures involve small movements and are performed at a body location that is difficult to instrument.

Pioneering research has demonstrated a variety of gestures and presented various approaches to sensing (Huang et al., 2016; Weigel et al., 2017). Considering the recency of the field, it is not surprising that it is characterized by point explorations, focusing on a specific and rather small subset of thumb-to-finger gestures. As a result, these recognition systems are typically developed to demonstrate novel interactions and therefore limited to specific instances, such as tapping on a finger segment or sliding along a finger. While viable for the purpose, such restricted gesture sets limit the scope of possible mappings in real-world applications. So far, it remained unclear whether the conceptual space of possible

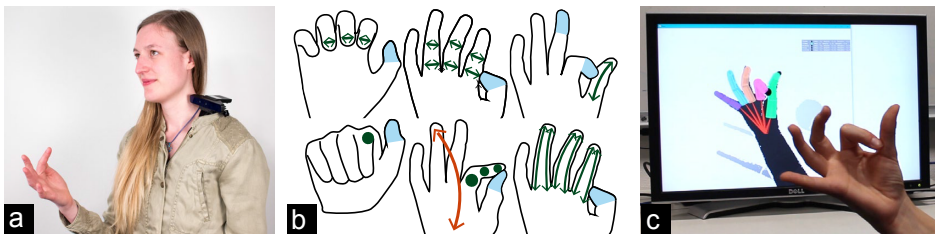


Figure 8.1: (a) FingerInput enables detection of versatile thumb-to-finger interactions using a body-worn depth camera. (b) By detecting finger flexion as well as touch locations, it supports a broad set of microgestures. (c) This is achieved thanks to a hybrid method that combines a convolutional neural network, a fully articulated hand model, and continuous collision detection based on geometric primitives.

gestures has been fully covered, what are common design dimensions, and—most important from a technical perspective—how the various gestures can be integrated in one system. This chapter presents a method which supports the set of thumb-to-finger gestures to a fuller extent and focuses on expressive, multidimensional thumb-to-finger interaction (see Figure 8.1). The supported gesture space provides a broader list of microgestures than previous work: the primitives cover existing gestures from the literature on hand-free microgrestures (Hrabia et al., 2013; Huang et al., 2016; Tsukadaa and Yasumurab, 2001; Wolf et al., 2011) while also demonstrating opportunities for novel gestures. The gesture design space enables to directly derive a set of technical requirements for the proposed recognition system in order to support a broader set of gestures. It is the first system that can capture all primitives of the design space and hence significantly extends the set of gestures that can be detected in an interactive system, adding to the expressiveness of input. The system is capable of identifying fingers and finger segments, tracking their 3D pose, and detecting linear and rotary touch contact between fingers, all with a high accuracy and in real time. The system uses a body-worn depth sensor mounted on the user's head or shoulder. As such, it does not require any instrumentation of the hand, while not being affected by bad lighting as only depth information is used. Three pilot studies are presented to validate the functionality of the algorithm. Results from a fourth technical evaluation with users show a high accuracy of 91% on thumb-to-finger gesture recognition, for a rich variety of eight gesture classes.

8.2 RELATED WORK FOR ON-BODY TOUCH INPUT

For related approaches for 3D hand pose reconstruction, please refer to Chapter 2. Most importantly, all existing hand tracking approaches estimate flexion angles for all fingers and some run in real time, however, no method for precise continuous touch point estimation has been proposed. The remainder of this section discusses related work on detecting on-body touch input.

Detecting touch input on the body has been approached using different sensing techniques, including acoustic sensing (Harrison et al., 2010), inertial and magnetic sensing (Chan et al., 2013; Chen et al., 2016; Hrabia et al., 2013; Huang et al., 2016), photo-reflective sensing (Ogata et al., 2013), radar (Wang et al., 2016) or capacitive sensing (Weigel et al., 2015, 2017). However, these approaches have some limitations: capacitive approaches have relatively low resolution, and the hand needs to be instrumented. Magnetic approaches have high resolution, but they do not provide accurate temporal touch detection. All the approaches are able to detect touch, but do not measure finger flexion for fingers other than the ones involved in the touch action (Yoon et al., 2014, 2015).

Another widely used approach consists of using a body-mounted camera. Possible camera locations include the head (Colaço et al., 2013; Funk et al., 2016; Tamaki et al., 2009), shoulder (Harrison et al., 2011; Winkler et al., 2014), chest (Chan et al., 2015a; Loclair et al., 2010; Mistry et al., 2009), and wrist (Dementyev and Paradiso, 2014; Kim et al., 2012; Prätorius et al., 2014; Sridhar et al., 2017). OmniTouch (Harrison et al., 2011) uses a depth camera and a projector mounted on the shoulder to turn the inside of the palm into a touch surface. Sridhar et al., 2017 use a depth camera mounted on the wrist to enable 3D input on the back of the hand. PinchWatch (Loclair et al., 2010) allows microinteractions by mounting a depth camera on the chest, which tracks the hand wearing a display. These approaches do not require instrumentation of the hand and generally tend to work robustly for larger touch or free-hand gestures that are based on the hand shape. However, it is a hard problem to accurately detect touch contact between fingers from a distant camera.

Finer finger gestures are addressed by Cyclopsring (Chan et al., 2015b) by mounting a fish-eye camera on a ring, which is used to detect different touch gestures based on the hand shape. For detecting touch, some approaches use image based techniques, such as flood filling (Harrison et al., 2011; Sridhar et al., 2017). While flood filling is suitable for detecting touch on a constrained touch area like a flat surface (Harrison et al., 2011), or the back of the hand (Sridhar et al., 2017), it is not suitable for a wider class of touch interactions on more general and complex surfaces, like

different finger segments. Even when geometric shapes have been used for hand tracking (e.g. Qian et al., 2014), none of the prior work used this approach for detecting touch points.

Extending beyond prior work, the proposed system is able to detect fine finger-to-finger interaction, involving the continuous rotational angle and relative position along the touched finger segment and an accurate detection of touch contact. Moreover, it provides continuous estimation of the flexion angles of all fingers and exact relative finger positions in 3D. As explained in the next section, this is sufficient for accurate detection of thumb-to-finger microgestures.

8.3 DESIGN SPACE OF THUMB-TO-FINGER GESTURES

This section describes the design space of thumb-to-finger gestures, introduces the example gesture set for the prototype system, and the implied technical requirements. The contents of this section are not contributions of this thesis but are rather provided as necessary context information.

8.3.1 *Dimensions of Variation*

A thumb-to-finger gesture can be defined by four dimensions: ① which finger is touching, ② what location on another finger is touched, ③ what touch action is performed, and ④ how fingers are flexed. The dimensions and their possible values are illustrated in Figure 8.2.

① *Touch Initiator*

The first defining factor for microgestures is the finger that triggers the touch action, that is, either the thumb or one of the remaining fingers.

THUMB-TO-FINGER. The vast majority of prior work focused on gestures that are initiated by the thumb, which is touching another finger. Work investigating such thumb-to-finger input includes Chan et al., 2016; Chan et al., 2013; Huang et al., 2016; Tsai et al., 2016a, 2017; Weigel et al., 2017; Whitmire et al., 2017; Zhang et al., 2017a. This space is comparably well-covered, including systematic empirical studies that investigated the comfort regions for touch interactions initiated by the thumb (Huang et al., 2016).

FINGER-TO-THUMB. The reverse interaction of a finger initiating a touch *on* the thumb has rarely been investigated. In the following, this is called *finger-to-thumb* input. This form of input has been implicitly

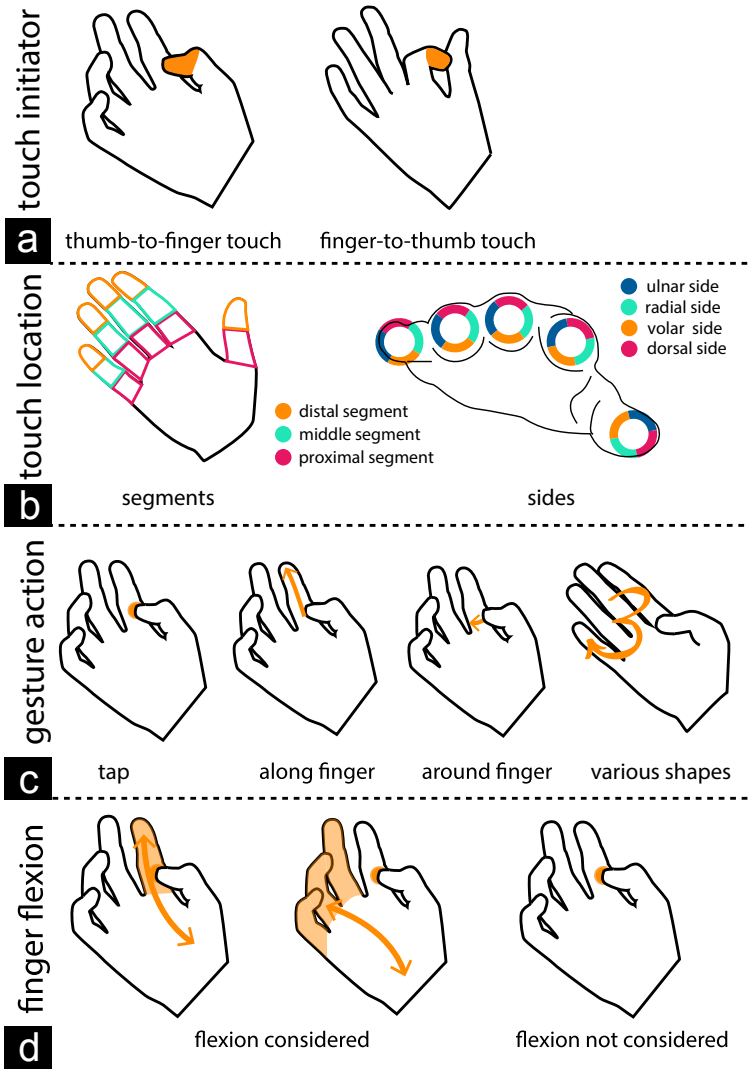


Figure 8.2: Thumb-to-finger gestures are defined by four dimensions: (a) The finger initiating the touch, (b) the touch location, (c) the gesture action, and (d) finger flexion.

used to extend the input area of finger sliding gestures (Loclair et al., 2010) or to enable sliding gestures while holding an object (Wolf et al., 2011). So far, this form of input has been limited to linear sliding with the index finger on the thumb. There is the opportunity to extend to a wider variety of gestures that include tapping, rotational sliding, and use of other fingers.

A third class, which would include touch contact between any two fingers other than the thumb, is explicitly excluded as it lays out of the scope of thumb-and-finger interaction. Such touch contact is particularly challenging to detect correctly, as the contact is frequent and the involved areas are not only points but whole surfaces at the time.

② *Touch Location*

The touched location is defined by the touched finger segment (*proximal*, *middle* or *distal*) and the rotary side of the finger (*radial*, *ulnar*, *dorsal* or *volar*). A tapping gesture contains only one touch location, while a continuous gesture contains multiple sequential locations.

FINGER SEGMENT. Each finger can be divided into two segments (thumb) or three segments (other fingers). Because of the tactile and visual cues generated by knuckles and wrinkles, each segment is clearly delimited. This makes them a natural choice as touch targets, as seen in prior studies (Prätorius et al., 2014; Tsai et al., 2016a; Whitmire et al., 2017). Some work has subdivided the segments even further (Huang et al., 2016; Tsai et al., 2017).

FINGER SIDE. The location of touch input on a finger is also defined by the rotary angle around the finger's longitudinal axis. The vast majority of work has focused on input performed on only one side of the finger: either the radial side (Tsai et al., 2017; Wang et al., 2016; Yoon et al., 2015) (i.e., the side closer to the thumb) or the volar (i.e., palmar) side (Huang et al., 2016; Prätorius et al., 2014; Seo and Cho, 2014). Only very few studies have investigated input on other sides. Notable exceptions include work that investigates the likability of touch input on two sides of the fingers (radial, palmar) (Tsai et al., 2016a). Other work applied tap on two sides of one segment (Huang et al., 2014) and pioneered sliding input around the finger segment (Ogata et al., 2012; Tsai et al., 2016b), finger nail (Kao et al., 2015), or finger wrinkles (Weigel et al., 2017). Overall, gestures that involve the different rotary sides of the fingers are still underexplored and present new opportunities for interaction.

③ *Gesture Action*

The gesture action defines what form of touch input the user is performing: either a tap, a continuous longitudinal or rotary sliding movement, or a specific shape that is drawn with the touch initiator.

TAPPING. The touch initiator is touching a finger at a discrete location. The touch locations explored include the different segments and sides of the fingers (Chan et al., 2013; Gustafson et al., 2013; Huang et al., 2014, 2016; Tsai et al., 2016a,b; Wang et al., 2016; Yoon et al., 2015), as well as the fingernails (Kao et al., 2015).

SLIDING ALONG THE FINGER. The touch initiator slides along the touched finger's longitudinal axis. The slide can be performed along the entire finger (Chan et al., 2015b; Gustafson et al., 2013; Loclair et al., 2010; Whitmire et al., 2017; Wolf et al., 2011; Yoon et al., 2014; Zhang et al., 2017a), or on a segment of the finger (Chan et al., 2013; Kao et al., 2015; Seo and Cho, 2014; Tsai et al., 2016b; Weigel et al., 2017). This set of actions is typically used to manipulate continuous values (Chan et al., 2015b; Loclair et al., 2010; Whitmire et al., 2017), but it has also been used for discrete gestures (Seo and Cho, 2014; Tsai et al., 2016b; Weigel et al., 2017).

SLIDING AROUND THE FINGER. The touch initiator slides perpendicular to the lateral axis of the touched finger (Chan et al., 2013; Kao et al., 2015; Ogata et al., 2012; Tsai et al., 2016b; Weigel et al., 2017). The action can be also be performed on multiple fingers (Chan et al., 2016; Gustafson et al., 2013; Wolf et al., 2011; Zhang et al., 2017a).

DRAWING SHAPES ON THE FINGERS. The initiator is used to draw a shape on one or more fingers. The action is completed once the shape is fully drawn and the touch contact released. Different shapes have been investigated, including circles (Loclair et al., 2010), characters (Huang et al., 2016), and digits (Zhang et al., 2017a). The drawing action can be performed on a single segment of one finger (Huang et al., 2016; Seo and Cho, 2014), or on multiple fingers (Chan et al., 2016).

④ *Finger Flexion*

The flexion of the different fingers of the hand can be considered as a part of the performed gesture. This property adds an additional dimension to the touch gesture performed on the fingers. Each finger can be *open*, *folded* or *moving*. The terms open and folded are used similarly to Krupka et al.,

2017 who defined a finger or the thumb as folded when its tip resides in a certain area in front of the palm. Finger flexion can be a discrete property. It can also be a continuous feature when one or more fingers are moving from an open to a folded state or from a folded to an open state during a gesture. There is only one prior study that included finger flexion of the touched finger to execute different actions (Yoon et al., 2015). Combining thumb-to-finger touch with the expressive capabilities of free-hand gestures (Wang et al., 2016) opens up a promising direction for novel gestures.

8.3.2 *Example Gesture Set*

A representative set of demanding microgestures for implementation in the proposed prototype system is shown in Figure 8.3. This selection includes discrete tapping as well as continuous movement along and around fingers, and also circular shapes. It further includes gesturing at many different finger locations, using the thumb or the index finger as initiator, and includes finger flexion. Interactions on and with the pinkie finger were excluded, since the largest part of the pinkie finger lays outside the comfort region of interaction (Huang et al., 2016).

This set of gestures is considerably more versatile than the thumb-to-finger microgestures presented in any single prior system. The gesture set is comprised of the following:

(a) **Finger Tap:** This set of gestures includes taps on nine segments on index, middle and ring finger. These gestures, known from prior work, are quick and easy to perform.

(b) **Fist Tap:** The hand forms a fist, all fingers being folded, while the thumb taps on a segment on the outer side of the index finger.

(c) **Tap-and-Flap:** Tapping based input can also be combined with dynamic finger poses. The thumb is tapping on the outer side of one of the index finger segments, while the other fingers move either from open to folded or vice versa.

(d) **Linear Thumb-to-Finger Slide:** The thumb is the touch initiator and performs a linear slide along the index, middle or ring finger. This set includes sliding on the inner and outer side of each finger, as well as in both directions, sliding from the root of the finger to the tip and vice versa, resulting in 12 gestures.

(e) **Linear Finger-to-Thumb Slide:** A similar sliding gesture can also be performed by the index or middle finger on the thumb.

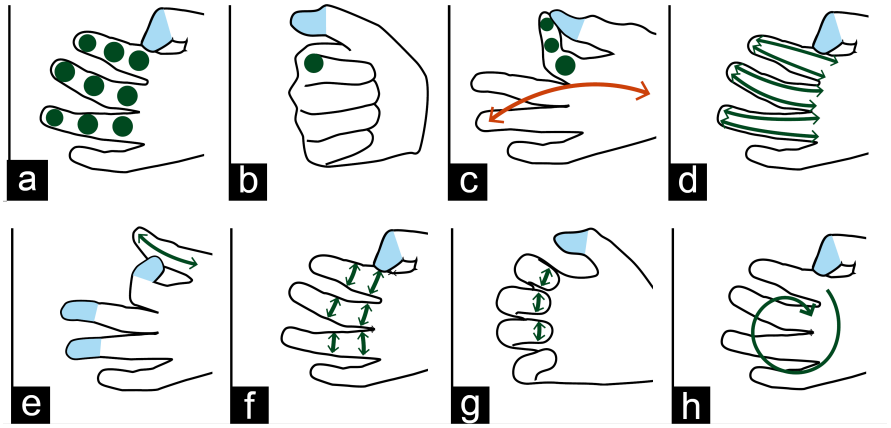


Figure 8.3: The 8 different classes of the evaluation gesture set: (a) finger taps, (b) fist tap, (c) tap-and-flap, (d) linear thumb-to-finger slides, (e) linear finger-to-thumb slides, (f) rotational thumb-to-finger slides, (g) fingertip slides, and (h) drawing a circle.

(f) **Rotational Thumb-to-Finger Slide:** The thumb is the touch initiator and performs a rotational slide around one of the six inner and middle finger segments of the index, middle and ring finger while all fingers are open. Sliding in both directions is included, resulting in overall 18 gestures.

(g) **Fingertip Slide:** The thumb can perform a rotational slide around the fingertip following the curve of the fingernail. This gesture applies to the index, middle, and ring fingers and can be performed in both directions.

(h) **Draw Circle:** The thumb starts from the lower segment of the index and draws a circle on the fingers.

8.3.3 *Resulting Technical Requirements*

Accurate detection of the gestural primitives of all four dimensions poses a set of demanding technical requirements for gesture recognition systems. The following six main requirements are identified:

1. Hand and finger segmentation
2. Identification of the touch initiator and of the touched finger
3. Estimation of the touch location, including linear (segment) and rotational (finger side) position
4. Temporal detection of touch contact (touch down vs. touch up), to identify the onset and offset of a gesture

5. Estimation of the flexion angles of all fingers
6. Real-time performance.

Based on these requirements, a gesture recognition approach, that is the main contribution of this thesis to the project, was developed and is described in the following section.

8.4 GESTURE RECOGNITION APPROACH

This section describes the proposed depth-camera-based gesture recognition approach for supporting versatile and expressive thumb-to-finger interactions which constitutes the contribution of the thesis to this project. The approach works in real time, with a single body-mounted depth sensor, and is able to reconstruct fine-grained thumb-to-finger interactions with high accuracy without requiring any instrumentation of the hand. The approach combines the real-time reconstruction of a fully-articulated hand pose (based on a fully convolutional neural network, a kinematic skeleton and a Sum of Gaussians model) with real-time detection of thumb-to-finger touch contact to accurately classify input gestures.

To capture input gestures, the user mounts a depth camera on the head or either shoulder, as shown in Figure 8.4. This placement follows strategies from prior work and ensures compatibility with AR/VR devices. For instance, future head-mounted displays will likely include a forward-facing depth camera. For the proposed prototype, an Intel RealSense SR300 camera (IntelRealSenseSR300, 2016) with a sensing range of 20 cm to 120 cm is used to capture the depth images.

Figure 8.5 depicts an overview of the proposed algorithm. Figure 8.6 shows examples of different touch poses processed by the algorithm pipeline. The algorithm consists of four main steps: (1) hand part classi-

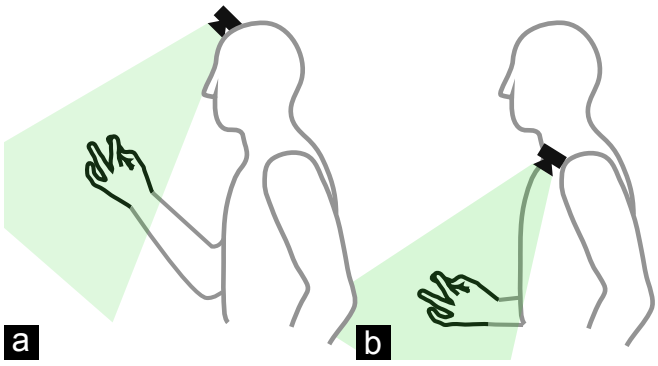


Figure 8.4: Thumb-to-finger touch gestures are captured using a depth camera that can be mounted on (a) the head or (b) the shoulder.

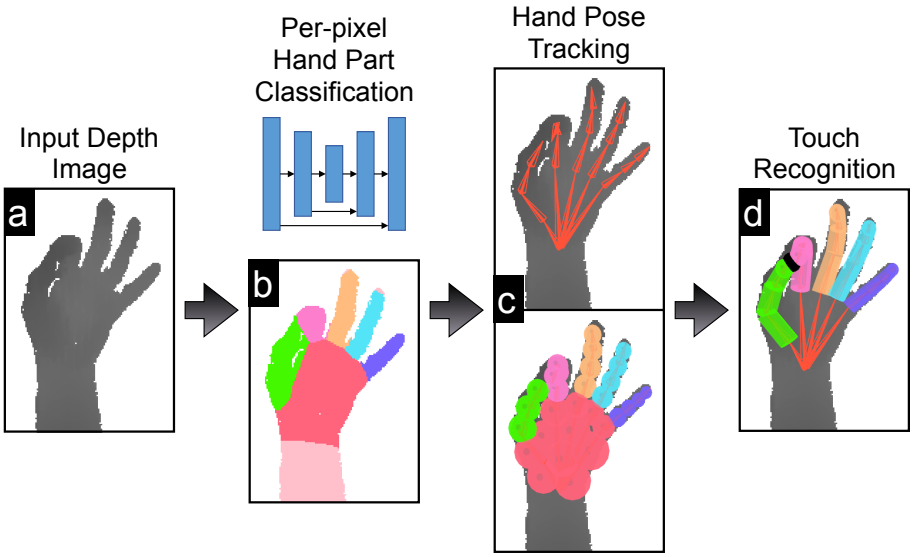


Figure 8.5: Algorithm processing steps: (a) Input depth image. (b) Per-pixel hand part labels are obtained using a CNN classifier. (c) The full pose of the hand is estimated using a coarse generative model. (d) Touch proxies –which approximate the surface more accurately– are attached to the kinematic skeleton and used for continuous 3D touch recognition.

fication in the image, (2) 3D hand pose estimation, (3) touch detection, and (4) gesture classification. The steps are described in more detail in the following.

8.4.1 *Hand Part Classification*

A per-pixel classifier is used to segment the hand from the background and the arm, and to identify different fingers (see Figure 8.5 (b)). The classifier is a fully convolutional neural network inspired by the U-Net architecture (Ronneberger et al., 2015). The encoder part transforms the input depth image at resolution 240×320 to 256 feature maps of size 15×20 using 5 convolution layers and 1 max-pooling layer. Afterwards, the decoder generates a class label image at the original image resolution with as many channels as number of classes using 4 deconvolution layers. At each pixel, the softmax function is applied over all channels to obtain a probability distribution encoding the likelihood of this pixel belonging to either class. The part label for this pixel is extracted as the top-scoring class. During training, the CNN tries to minimize the multinomial logistic loss with information gain matrix. The weights in the information gain matrix are set according to the class frequencies to

handle class imbalance. The CNN is trained using the Caffe framework (Jia et al., 2014). The training is performed for 175,000 iterations with an input batch size of 16 using the AdaDelta (Zeiler, 2012) solver with momentum 0.95 and 0.0005. The base learning rate is lowered from 1.0 to 0.1 after 110,000 iterations.

To train the classifier, real annotated training data was collected with automatic color-based labeling from a body-mounted RGB-D camera at resolution 480×640. The data includes 5 participants (3 males, 2 females; 22–27 years old) with the camera mounted at two different locations (head and shoulder), to provide an egocentric view of the hand for eyes-free input (see Figure 8.4). To collect ground-truth information, a color coding was applied to fingers, palm, back-of-the-hand, and arm, using finger paint, which is not visible in the IR image. This is the only step that required the color channels (RGB). Per-pixel hand part labels for the depth image are obtained by HSV color segmentation. For more information on the data collection, please refer to Section 4.3. In total, 66,662 images were collected and automatically annotated. Of this set, 60,000 images were used for training the classifier. The remaining images were held out as a test set for classifier evaluation.

8.4.2 *Hand Pose and Fingertip Estimation*

Recognizing finger-to-finger touch is a harder problem than identifying touch between a finger and a constrained, planar surface (Harrison et al., 2011; Sridhar et al., 2017). It requires not only knowledge of fingertip positions but of the location of all bones in every finger. This full articulation of the hand is commonly described by 26 degrees of freedom (DOF). For estimating all these DOF, a generative model consisting of a kinematic skeleton and a 3D Sum of Gaussians (SoG) model (as introduced in Sections 3.1 and 3.2) is used. The SoG coarsely models the volumetric extent of the hand (see Figure 8.5 (c)) and has been successfully employed for full body as well as hand motion tracking (Sridhar et al., 2015a; Stoll et al., 2011).

For every input depth image, the hand pose $\boldsymbol{\theta}$ is found as the minimizer of the energy

$$\mathcal{E}(\boldsymbol{\theta}) = E_{\text{data}}(\boldsymbol{\theta}) + E_{\text{coll}}(\boldsymbol{\theta}) + E_{\text{lim}}(\boldsymbol{\theta}) + E_{\text{temp}}(\boldsymbol{\theta}) \tag{8.1}$$

that consists of a data term, which describes the discrepancy of the generative model and the input observation, and several regularizers. Let $\{G_m\}_{m=1}^{N_m}, \{G_i\}_{i=1}^{N_i}$ be the SoG representation of the hand model and the

input depth image, respectively. The data term is then formulated based on the 3D Gaussian overlap

$$E_{\text{data}}(\boldsymbol{\theta}) = -\sum_{m=1}^{N_m}\sum_{i=1}^{N_i}\Delta(m,i)\int_{\mathbb{R}^3} G_m(x;\boldsymbol{\theta})\cdot G_i(x)\,dx\,, \tag{8.2}$$

where only the model Gaussians depend on the hand pose $\boldsymbol{\theta}$ and the overlap is negated since $\mathcal{E}$ is minimized. The weight $\Delta(m,i)$ for each pair of Gaussians takes into account the predicted hand part label l_i obtained in the previous step and compares it to the pre-defined label l_m of the hand model Gaussian

$$\Delta(m,i) = \begin{cases} 0 & \text{if } (l_m \neq l_i) \text{ or } (d_{m,i} > r_{max}) \\ (1 - \frac{d_{m,i}}{r_{max}}) & \text{else} \end{cases}, \tag{8.3}$$

where $d_{m,i} = \|\mu_m - \mu_i\|_2$ is the distance of the Gaussians and $r_{max} = 30$ cm is the influence radius. The part label l_i of the input Gaussian i is obtained by majority vote over the predicted per-pixel labels of the pixels that are clustered to Gaussian i.

The collision term E_{coll} prevents self-intersections by penalizing the overlap of the hand model SoG with itself as introduced in Chapter 7, Equation (7.11). Furthermore, limits of joint angles and temporal smoothness are incorporated into the energy to ensure physical plausibility, as described in Chapter 5, Equations (5.9) and (5.10).

This objective function is fast to optimize and has shown accurate and temporally stable results (for details see Sridhar et al., 2016). The fingertip positions and 3D location of every bone in the hand can be read from the posed kinematic skeleton after optimization, as described in Section 3.1.

8.4.3 *Touch Detection*

To reliably recognize finger-to-finger touch contact based on the hand pose, touch proxies are attached to the hand model (see Figure 8.5 (d)). These approximate the surface more accurately than the isospheres of the Gaussians. The three segments of each finger are modeled as elliptical cylinders, the fingertips are modeled as spheres. These proxies are attached to the bones of the kinematic model so that they can easily be moved according to the tracked hand pose. Calculating a 3D touch point $\mathbf{p}$ can hence be formulated as sphere/cylinder and sphere/sphere intersection tests. The touch point $\mathbf{p}$ is then transformed to the local coordinate system (LCS) of the touched segment: $\mathbf{p}_{loc} = T \cdot \mathbf{p}$, where T is the transformation matrix from the global to the local coordinate system. Note that the LCS of a segment is setup s.t. the y-axis aligns

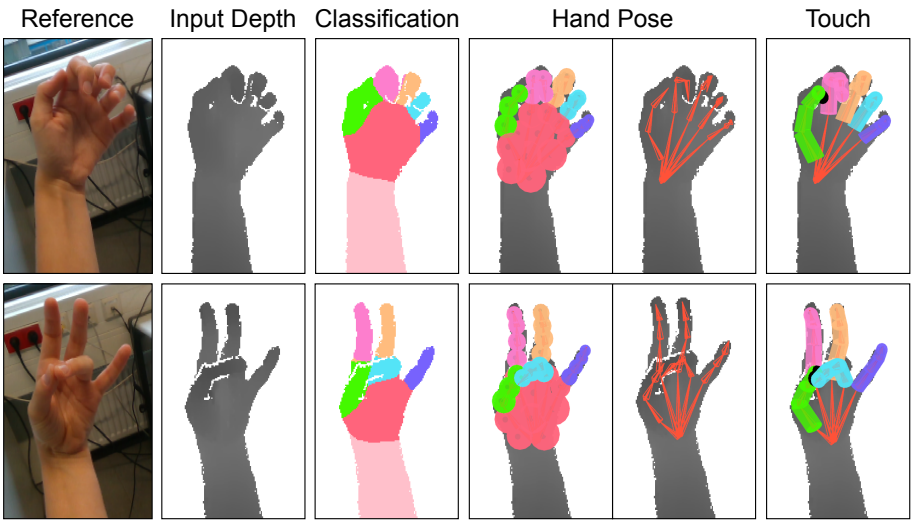


Figure 8.6: Different touch poses through the system pipeline. Touch position is indicated by a black circle.

with the bone and the x-z-plane is perpendicular to the bone. Thus, the exact relative geometric location of the touch point within the segment can be recovered: (1) the rotational angle around the bone is determined by the position of the touch point in the x-z-plane of the LCS, i.e. the x- and z-coordinate of $\mathbf{p}_{loc}$, and (2) the position along the bone is derived from the y-coordinate of $\mathbf{p}_{loc}$. The use of geometric primitives for touch recognition is computationally more efficient than other surface representations (e.g. meshes) while still allowing to model surfaces that are more complex than a single planar object.

8.4.4 *Gesture Classification*

The touch information and the estimated hand pose are fed into a continuous gesture classifier to enable rich user interactions. The ability of *FingerInput* to detect exact touch instances provides a straightforward approach to recognizing gestures using voting-based discriminative classification in a continuous manner. Since all gestures involve touch contact, the classifier is activated when a touch instance starts and runs as long as it is present. The definition of gestures through the dimensions of the design space presented above allows for representing and defining gestures as a combination of values of these dimensions. A dictionary of gestures, defined by the values of the given dimensions, is stored in the system. For recognition, and on each frame, the values of each of the four dimensions are observed: touch initiator, touch location, gesture

action, and finger flexion. A vote is then added to the corresponding combination of the dimensions, which maps to one of the defined gestures in the dictionary. A time sliding window across frames is then used to determine the performed gesture through majority voting, where the gesture with the most votes in the current window is selected. Empirical results, presented in Section 8.5, demonstrate that the system is able to detect a versatile set of gestures with a high accuracy.

Note that the proposed gesture recognition system is not restricted to those gestures, but can be easily trained to detect other gestures that are based on the dimensions of the design space.

8.4.5 *Implementation and Runtime*

The prototype system runs at 40 frames per second on a desktop computer, using an Nvidia GTX 1080 GPU. The system recognizes gestures performed and sends events to clients through a WebSocket connection. Clients such as smart watches, smart phones, wall displays, or head-mounted displays can receive this information over a wireless connection.

8.5 SYSTEM EVALUATION

The contribution of this chapter is a tracker that accurately detects an expressive set of thumb-to-finger microgestures. To validate this claim, a series of pilot studies is conducted first to assess the correct and accurate functioning of the individual processing steps of the algorithm. The main evaluation study then investigates the end-to-end functionality of the approach by assessing the system's accuracy to detect demanding microgestures that were performed by users. The findings confirm the functionality and accuracy of the proposed system.

8.5.1 *Pilot Study 1: Finger Classification*

The first requirement for gesture detection is the correct segmentation of regions in the camera image. As described in Section 8.4.1, 66,662 depth images were collected and automatically annotated per-pixel for 6 semantic hand part classes and a non-hand class. 60,000 of these images were used for classifier training and 6,662 images were held out as test set. Table 8.1 shows the classification accuracies per class. The proposed CNN classifier achieves an average accuracy of **90.2%**, which shows its ability

Table 8.1: Per-class accuracies achieved by the proposed classifier for the classes.

Region	Palm	Thumb	Index	Middle	Ring	Pinkie	Non-hand
Accuracy	96.0%	93.5%	91.6%	87.3%	79.1%	84.8%	99.0%

to accurately classify different fingers and to detect the exact fingers involved in a gesture.

8.5.2 *Pilot Study 2: 3D Fingertip Localization*

To characterize the system's accuracy in detecting the positions of the fingertips in 3D space, two participants (1 female, 1 male; 23 and 27 years old) with average hand size (hand lengths: 201 mm, 192 mm; hand widths: 90 mm, 89 mm) (Poston, 2000) interacted with the system by performing gestures from the gesture set (see Figure 8.3) as indicated by a video reference. In total, 3,573 frames were collected and the fingertip positions were annotated in each frame to serve as ground truth for evaluation. As a baseline reference, an egocentric tracker that uses a random decision forest (RDF) (Sridhar et al., 2015a) instead of a CNN is used. Then the average fingertip localization error is calculated by computing the Euclidean error of the 5 fingertip positions averaged over all frames.

The proposed tracking system outperforms the egocentric RDF-based tracker by a large margin (**13.92 mm** and **16.37 mm** vs. 22.5 mm and 38.0 mm). The results show that the system has the capability to localize fingertips of different fingers with an error less than the average finger segment size. Note that the average error of the egocentric RDF-based tracker is higher than the error achieved by the corresponding third-person tracker proposed by Sridhar et al., 2015a on common third-person datasets. This seems to indicate that egocentric sequences are more challenging than third-person settings due to frequent self-occlusion of the hand and additional camera motion.

8.5.3 *Pilot Study 3: Touch Contact Between Fingers*

To measure touch accuracy, a second dataset is used to compare the tracker output with ground-truth data acquired via self-capacitive touch sensing. Capacitive touch is very accurate as it does not confuse finger touch down events with finger hover state and it is unaffected by occlusions. To capture capacitive ground-truth data, a custom-built very thin and flexible sensor was used that participants wore on the thumb during

this pilot study (see Figure 8.7). Two participants (1 female, 1 male; 23 and 27 years old) performed the same gestures as in Pilot Study 2. This dataset (5,174 frames) was automatically annotated with ground-truth information.

The accuracy is calculated using a frame-by-frame comparison as well as a time window comparison. For the frame-by-frame comparison, the number of frames where a touch is detected both by the system and in the ground truth data is calculated, yielding an average accuracy of **87.5%**. Most of the misclassified frames lie either directly before or after a performed touch gesture. These one-frame errors can be addressed with a time window: at a window length of 500 ms, the system has an average accuracy of **95.8%**. Lengths of 600 and 700 ms result in **96.2%** and **97.1%**, respectively.

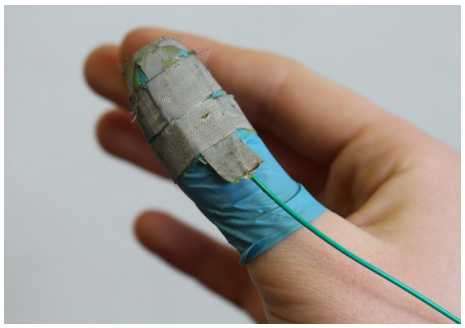


Figure 8.7: Automatic ground-truth annotation of thumb-to-finger touch using a capacitive finger glove, built out of nitrile rubber with conductive fabric affixed around the tip of the finger.

8.5.4 *Main Evaluation Study: Gesture Detection Accuracy*

To formally evaluate the accuracy of end-to-end gesture recognition, a user study was performed. 10 volunteers were recruited (6 males, 4 females, 19–29 years old). Their hand lengths varied from 155–212 mm, and the hand widths from 74–92 mm (mean dimensions were 180 mm by 86 mm), providing a representative sample of hand size distribution (Poston, 2000). The task consisted of performing all gestures in the 8 gesture classes presented in Figure 8.3, with 4 trials each. Per gesture, two trials were performed while sitting, and the two others in a standing

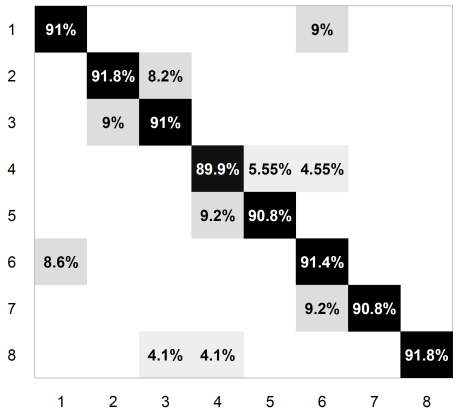


Figure 8.8: Confusion matrix for the 8 different classes of the gesture set: (1) finger taps, (2) fist tap, (3) tap-and-flap, (4) linear thumb-to-finger slides, (5) linear finger-to-thumb slides, (6) rotational thumb-to-finger slides, (7) fingertip slides, and (8) drawing a circle. The proposed system has a classification accuracy of **91.06%**.

position. For each trial, participants started with an open-hand pose with no touching fingers, and then performed the touch gesture shown on a screen. Participants were instructed on how to perform the gestures, and were given several minutes to practice the gestures until they were comfortable with them. Gestures were recorded with an Intel RealSense SR300 depth camera, mounted on the participant's shoulder using a shoulder mount. Each recording session lasted ≈1 hour, with a break after performing 25 gestures. The proposed classifier evaluated the performed gestures for each frame.

The overall accuracy of the system for gesture classification was **91.06%**. The confusion matrix is shown in Figure 8.8. As the results reveal, the fist tap and the circle drawing has the highest accuracy of 91.8%. This validates the ability of the proposed system to avoid false activations of primary actions. Linear thumb-to-finger slides have the lowest accuracy of 89.9%. Rotational slides are at times confused with finger taps, as they both occur inside one segment.

8.6 DISCUSSION AND LIMITATIONS

SELECTED GESTURE SET. The presented gesture set is a selected subset from the possibilities present in the design space. These gestures were chosen to highlight the flexibility of the tracker with an heterogeneous gesture set, yet they are in no way exhaustive. Given that the tracker accurately recognizes the hand pose and touch points, other microgestures could be easily supported; a notable example of this would be to take into consideration the hand position and orientation, which are computed as part of the tracking process.

OCCLUSIONS. The results from the evaluation show that *FingerInput* achieves thumb-to-finger gesture classification with low false positives and can detect the fingertip location and classify fingers accurately. However, since the technique is based on an optical approach, it is limited by line-of-sight requirements: some occlusions resulting from crossed fingers, overly bent fingers, or rotated hands can be problematic for gesture classification. For this reason, the current setup assumes the user is holding the hand in a relaxed pose in front of the chest.

HAND MODEL. The current version of the system requires that the measurements of the hand are manually indicated once for each user. Future implementations will include an initialization step.

MISCLASSIFICATION. It was empirically found that tap gestures can be misclassified as rotational slides when the user is slightly moving the finger while tapping. While this demonstrates the high spatial resolution of rotational slide, which even detects minimal movement, it may result in an undesired command. A straightforward solution consists of increasing the threshold for a minimal sliding movement. The trade-off between spatial resolution of rotational slides and robust detection of finger taps is an important question that should be further investigated in future work. An alternative design solution consists of spatial multiplexing: Drawing from the many different finger locations supported by the proposed system, the designer can reserve some segments or finger sides for tap gestures, while others are used for sliding input.

MOBILITY. The current prototype uses a desktop computer. The mobility could be extended by connecting the depth camera to a body-worn microcontroller (e.g., Raspberry Pi), and streaming depth data to a server for gesture classification.

8.7 CONCLUSION

This chapter presented *FingerInput*, a system for versatile thumb-to-finger touch gestures, based on a 3D hand pose estimation method using a depth sensor. The main contribution of this thesis to the project was the gesture recognition approach (Section 8.4), whereas the exploration of the gesture design space (Section 8.3) was provided for better context. The gesture recognition approach covers the dimensions of the design space, and recognizes discrete and continuous thumb-to-finger touch gestures. The results demonstrate that it is possible to implement a gesture recognition system that considerably extends the types of gestures that are supported in a single interface. The example applications showed that thumb-to-finger gestures are rich and versatile and offer strong support for direct and single-handed interaction.

9 CONCLUSION

This thesis explored various challenging directions to push the state of the art in real-time 3D hand reconstruction and tracking from a single camera.

All presented methods largely profited from the proposed novel datasets that were created in a smart way, combining different techniques to generate data for new and often harder tasks (Chapter 4). The first method dropped the assumption of observing a single isolated hand in free air, which has been common in related work (Chapter 5). The proposed dataset *SynthHands*, which was created using a novel merged reality setup, enabled training of a two-stage neural network regressor for sparse keypoints. In combination with a kinematic skeleton fitting step, the method achieved robustness to strong occlusions caused by arbitrary objects and in cluttered environments. Next, the sensor assumption was relaxed to a monocular RGB camera while still aiming at a full reconstruction in global 3D space (Chapter 6). To accomplish generalization from synthetic RGB training data to real RGB test sequences, it was crucial to enhance the training data using a novel geometrically consistent GAN, yielding the *GANerated Hands* dataset. The last method tackled simultaneous pose and shape reconstruction of two strongly interacting hands from a single depth sensor (Chapter 7). Instead of predicting sparse keypoint locations, a neural network was trained to regress dense correspondences between the input image and the hand model surface, enabled by the new *DenseHands* dataset. These correspondences were subsequently used to fit the hand models in a generative optimization-based framework.

In addition, this thesis also presented a hand and gesture tracker enabling the application of 3D hand tracking for recognition of thumb-to-finger microgestures for human–computer interaction (Chapter 8).

9.1 INSIGHTS

Besides the individual contributions of each chapter, there are several broader insights that arise from the thesis as a whole.

THE IMPORTANCE OF DATA. All presented methods for 3D hand motion capture and reconstruction introduced their own dataset to achieve the respective goal, namely *SynthHands* (Section 4.1, used in Chapter 5),

GANerated Hands (Section 4.2, used in Chapter 6), and *DenseHands* (Section 4.4, used in Chapter 7). The creation of *SynthHands* and *DenseHands* demonstrated how to leverage existing methods in a smart way to obtain automatically annotated data for a more challenging scenario. For the first one, the merged reality capture only required tracking of a single hand in free air to generate realistic interactions with virtual objects. For the second one, the two hands of a user were tracked separately at a safety distance while the live physical simulation framework enabled creation of natural two-hand interactions. The *GANerated Hands* dataset showed how to use machine learning techniques to transform existing data for a new task while ensuring annotation consistency without requiring paired examples. These datasets proposed in this thesis were a crucial advantage over related work in order to tackle underexplored or new challenges in 3D hand tracking.

HANDLING THE DOMAIN GAP. The datasets used for the presented methods were mostly based on synthetic data. To enable generalization to real test sequences, two different strategies emerged in this thesis. First, the domain gap can be reduced by adding real data either in augmentation or annotated for a subset of the tasks. The *SynthHands* dataset consists of rendered hands, however, the object textures and RGB-D backgrounds for augmentation were real data. Depth images can be automatically annotated for the hand segmentation task by coloring the hands. Adding this real depth data to the synthetic *DenseHands* dataset for segmentation and correspondence regression, significantly increased the generalizability as shown in this thesis. If no real data can be added, the second strategy for reducing the domain gap is to use image-to-image translation techniques to make the data distributions and image features more similar. Generative adversarial networks can be trained to enhance synthetic images such that they mimic real images better. As shown for the *GANerated Hands* dataset, this can be done with unpaired examples while ensuring valid annotation transfer with a geometric consistency loss. To easily generate large annotated training corpora, the methods proposed in this thesis based their training data generation on synthetic data. For the successful application to real test cases, it was key to explore ways to reduce the domain gap.

COMBINING MACHINE LEARNING AND MODEL FITTING. Whereas many of the more recent related methods have solely focused on using machine learning techniques, especially neural networks, for hand reconstruction, the approaches proposed in this thesis combine machine learning components with optimization-based model fitting in novel

ways. The proposed new combinations of these two paradigms usually achieve results that are better than any of the two components applied separately. In the first two presented methods, the kinematic skeleton fitting step combines the 2D and 3D predictions by jointly fitting to both, leading to a full 3D result with improved reprojection error in the input image. Furthermore, imposing bio-mechanical constraints and temporal regularization is straightforward in the optimization of the model fitting step without the need for a temporal neural network architecture. In the two-hand reconstruction method, the commonly used closest point correspondences were replaced with regressed correspondences from a neural network. Therefore, the model fitting step became independent of the initialization and exhibits improved accuracy as demonstrated experimentally. In addition, the task for the neural network can be formulated as 2D image-to-image translation hence not requiring capacity for explicit lifting to the 3D space.

In conclusion, this thesis has shown that efficient and innovative strategies to combine machine learning and optimization-based model fitting bring many advantages. On one hand, availability of model knowledge or a separate model fitting step means that simpler machine learning methods are often sufficient or that they do not need to learn everything from scratch. On the other hand, guidance from machine learning techniques significantly improves the robustness and accuracy of conventional optimization-based model fitting algorithms.

9.2 OUTLOOK

This thesis significantly advanced the state of the art in real-time 3D hand reconstruction and tracking from a single RGB or depth camera. Nevertheless, there are still many open directions for future work. In the following, some of them are presented in more detail.

TWO HANDS AND OBJECTS. While this thesis tackled 3D reconstruction of a single strongly occluded hand and two hands interacting with each other, it did not consider simultaneous reconstruction of the hands and the objects that they are interacting with. However, this is an important setting for various applications, for example in robotics. The inherent segmentation problem is harder, there are more occlusions, and the dimensionality of the optimization problem for model fitting significantly increases, especially in the case of non-rigidly deforming objects. An advantage of simulatenous reconstruction is that mutual constraints like touch points and interaction forces can be exploited. Hence, exploration and integration of real-time physical simulation techniques could be

an avenue for future work. A method for reconstructing two hands in interaction with an object should ideally work with arbitrary unknown objects. Thus, building a model of the object while tracking it in 3D is a promising approach. Requirements for operation in real time and a single camera setup make these future directions even more challenging.

APPEARANCE MODEL AND MORE DETAILED GEOMETRY. Methods that rely on scanned per-user hand meshes (e.g., from a 3D laser scanner) cannot be employed by everyday users. Instead, having parametric hand models (Khamis et al., 2015; Romero et al., 2017; Tkach et al., 2017) is desirable because they enable adaptation to unseen users by optimizing in a comparably low-dimensional parameter space. The MANO model (Romero et al., 2017) was the first publicly available parametric model for hand pose and shape, and future models will improve over it in many ways. First, there is not yet any parametric hand appearance model, which is crucial for many applications such as personalized avatars in AR/VR. In addition, model estimates could be directly compared to an RGB input image, both as a loss in a neural network as well as in a generative model fitting framework. This enables simultaneous appearance reconstruction and could potentially increase the accuracy of the pose and shape estimates. The complexity of such an appearance model is variable and could for example also include pose-dependent appearance changes or sophisticated material models.

A second direction for improving parametric hand models is to aim for representing more fine-grained geometry. In general, estimating the hand shape of the user more accurately translates to a more accurate 3D pose tracking result. The mesh of the MANO model is comparably coarse, it consists of 778 vertices for a single hand. In order to achieve a high-quality reconstruction that includes fine-grained geometry details, such as pose-dependent wrinkles, a higher mesh resolution is necessary. A prerequisite for building such a high-resolution model is the availability of high-quality 3D hand scan data. Once built, such a high-quality model could even be fitted to low-resolution or noisy data (e.g., from a commodity depth sensor). Overfitting to the noise could be easily prevented by regularization in the low-dimensional parametric space hence yielding plausible fine-grained geometry details from low-quality data.

A

NEURAL NETWORK DETAILS

This appendix provides network architecture and training details for the various neural networks used throughout the thesis.

A.1 GEOCONGAN (CHAPTER 4)

This section provides details of the *GeoConGAN* network (Chapter 4).

A.1.1 *Network Design*

The architecture of *GeoConGAN* is based on the CycleGAN (Zhu et al., 2017), i.e., two conditional generator and two discriminator networks are trained for synthetic and real images, respectively. Recently, also methods using only one generator and discriminator for enrichment of synthetic images from unpaired data have been proposed. Shrivastava et al., 2017 and Liu et al., 2019 both employ an L1 loss between the conditional synthetic input and the generated output (in addition to the common discriminator loss) due to the lack of image pairs. This loss forces the generated image to be similar to the synthetic image in all aspects, i.e., it might hinder the generator in producing realistic outputs if the synthetic data is not already close. Instead, the proposed network uses the combination of cycle-consistency and geometric consistency loss to enable the generator networks to move farther from the synthetic data thus approaching the distribution of real world data more closely while preserving the pose of the hand. The *GeoConGAN* contains *ResNet* generator and Least Squares *PatchGAN* discriminator networks.

A.1.2 *Training Details*

The *GeoConGAN* is trained in Tensorflow (Abadi et al., 2016) for 20,000 iterations with a batch size of 8. The Adam optimizer (Kingma and Ba, 2014) is initialized with a learning rate of 0.0002, $\beta_1 = 0.5$, and $\beta_2 = 0.999$.

A.2 HALNET AND JORNET (CHAPTER 5)

This section explains the network architecture used for *HALNet* and *JORNet* and provides training details. Furthermore, the experiments which lead to the specific design decisions are presented.

A.2.1 *Network Design*

The ResNet architecture (He et al., 2016) has been successfully used for full body pose estimation in previous work (Mehta et al., 2017a). While ResNet50 offers a good trade-off between speed and accuracy, hand motion is fast and exhibits rapid directional changes. Further, the egocentric camera placement leads to even faster relative motion in the scene. Therefore, experiments were performed based on recent investigations into the nature of representations learned by ResNets (Greff et al., 2016) to get a faster architecture without significantly affecting the accuracy.

CORE ARCHITECTURE. Starting from ResNet50, a residual block is removed from level 3, and only 4 residual blocks are kept at level 4. Level 5 is replaced with two 3×3 convolution layers with 512 (conv4e) and 256 (conv4f) features and no striding. Both of these layers also use batch normalization (Ioffe and Szegedy, 2015). From *conv4f*, A 3×3 convolutional stub followed by bilinear upsampling produces the joint location heatmaps, and a fully-connected layer with 200 nodes followed by another fully-connected layer predict the joint location vector. See Figure A.1 for details.

The resulting architecture needs 10 ms for a forward pass at resolution 320 x 240 (*HALNet*) and 6 ms at resolution 128 x 128 (*JORNET*) on a Nvidia Pascal Titan X GPU. This is a significant speed-up compared to the ResNet50 version which needs 18 ms and 11 ms, respectively. Evaluation on the *SynthHands* test set shows that the drop in accuracy is only marginal. ResNet50 trained on the hand localization task achieves 2.1 px average error whereas *HALNet* achieves 2.2 px.

INTERMEDIATE SUPERVISION. For *HALNet* and *JORNet*, a subset of the feature maps at each of *res3a*, *res4a* and *conv4e* in the networks is treated as the predicted heatmaps, for intermediate supervision (Lee et al., 2015). For *JORNet*, the feature maps at the aforementioned stages are additionally used to predict joint positions for intermediate supervision (see Figure A.1).

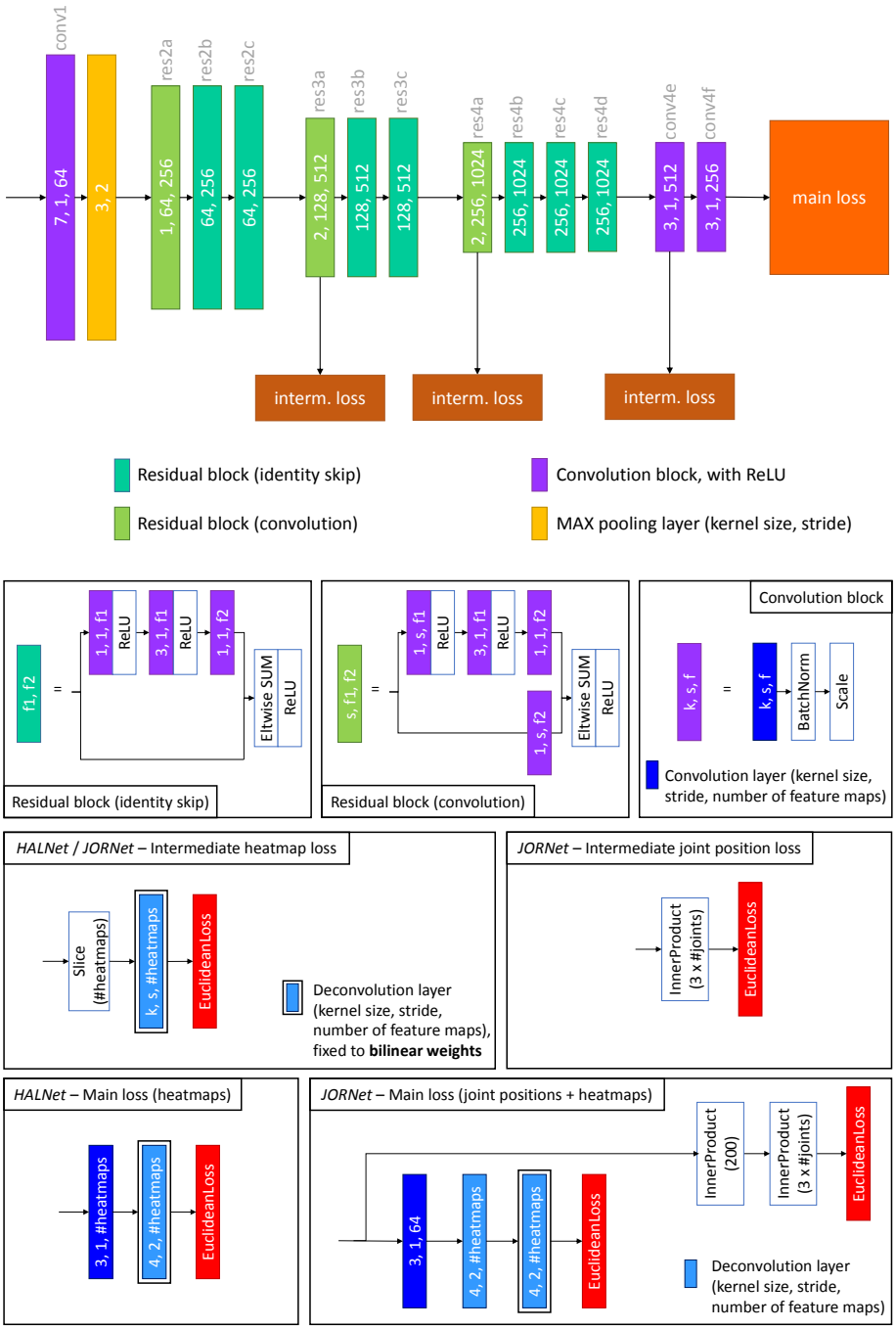


Figure A.1: The proposed network architecture for *HALNet* and *JORNet*.

AUXILIARY TASK IN HALNET. Predicting heatmaps for all joints as auxiliary task helps *HALNet* to learn the structure of the hand. This leads to a better performance compared to a version only trained to regress the root heatmap. On the *SynthHands* test set, regressing heatmaps for all joints instead of only for the root improves the 2D pixel error by **6.4%** (from 2.35 px to 2.2 px).

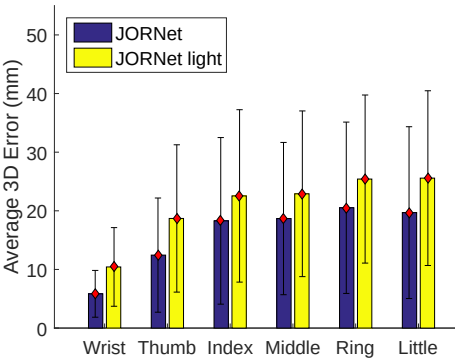


Figure A.2: Training *JORNet* to regress heatmaps and local joint positions for all joints instead of only for fingertips and the wrist (*JORNet light*) reduces the error for fingertips and wrist on the *SynthHands* test set.

REGRESSING ALL JOINTS IN JORNET. Although fingertips and wrist location alone provide a strong constraint for the pose of the hand, training *JORNet* to regress the heatmaps and local 3D positions for all joints improves the accuracy. Figure A.2 shows the average error for the wrist and all fingertips in 3D on the *SynthHands* test set. The full *JORNet* version yields a significant increase in performance compared to *JORNet light* which was only trained for wrist and fingertips.

A.2.2 *Training Details*

The proposed network are trained within the Caffe (Jia et al., 2014) framework, using the AdaDelta scheme (Zeiler, 2012) with momentum set to 0.9 and weight decay to 0.005. Both networks are trained with an input batch size of 16. *HALNet* uses a base learning rate of 0.05 and is trained for 45k iterations. The input has a spatial resolution of 320x240 px, and the output heatmaps have a resolution of 40x30 px. The main heatmap loss has a loss weight of 1.0, and all intermediate heatmap losses have loss weights of 0.5. For *JORNet*, the input has a spatial resolution of 128x128 px. The base learning rate is 0.05, with main heatmap loss weight set at 1.0 and joint position loss weight at 2500. The intermediate heatmap losses have their loss weights set to 0.5 and intermediate joint position loss weights set to 1250. After 45k iterations, the base learning rate is lowered to 0.01, the intermediate heatmap loss weights lowered to 0.1 and the intermediate joint position loss weights lowered to 250, and trained for a further 15k iterations.

A.3 REGNET (CHAPTER 6)

This section provides details of the *RegNet* network (Chapter 6). A forward pass of *RegNet* in takes 13 ms on a GTX 1080 Ti.

A.3.1 *Projection Layer ProjLayer*

Recent work in 3D body pose estimation has integrated projection layers to leverage 2D-only annotated data for training 3D pose prediction (Brau and Jiang, 2016). Since *GANerated Hands* training dataset provides perfect 3D ground truth, the proposed projection layer merely acts as refinement module to link the 2D and 3D predictions. The intermediate relative 3D joint position prediction are projected using orthographic projection where the origin of the 3D predictions (the middle MCP joint) projects onto the center of the rendered heatmap. Hence, the rendered heatmaps are also relative and not necessarily in pixel-correspondence with the ground truth 2D heatmaps. Therefore, the rendered heatmaps are further processed before feeding them back into the main network branch. Note that the rendered heatmaps are differentiable with respect to the 3D predictions which makes backpropagation of gradients through the *ProjLayer* possible.

A.3.2 *Training Details*

The *RegNet* is trained in the Caffe (Jia et al., 2014) framework for 300,000 iterations with a batch size of 32. The AdaDelta (Zeiler, 2012) solver is used with an initial learning rate of 0.1 which is lowered to 0.01 after 150,000 iterations. All layers which are shared between *RegNet* and *ResNet50* are initialized with the weights obtained from ImageNet pretraining (Russakovsky et al., 2015). Both, the 2D heatmap loss and the local 3D joint position loss, are formulated using the Euclidean loss with loss weights of 1 and 100, respectively.

A.4 CORN (CHAPTER 7)

This section provides training details of the correspondence regression network CoRN (Chapter 7).

A.4.1 *Training Details*

CoRN was trained in Tensorflow using the Adam (Kingma and Ba, 2014) optimizer with the default parameter settings. The training ran for 450,000 iterations using a batch size of 8, where synthetic and real images were sampled with 50% probability each. With a training time of approximately 20 seconds for 100 iterations, the total training process took 25 hours on an Nvidia Tesla V100 GPU.

A.4.2 *Input Data Processing*

The depth values are scaled to meters and the mean value of all valid depth pixels is subtracted. Furthermore, the following image augmentations are applied to the training data, where all augmentation parameters are sampled from a uniform random distribution:

- rotation augmentation with rotation angle $\in [-90, 90]$ degrees,
- translation augmentation in the image plane with offset $\in [-0.25, 0.25] \cdot$ image size, as well as
- scale augmentation with possibly changing aspect ratio in the range of $[1.0, 2.0]$.

Note that all these augmentations are applied on-the-fly while training, i.e., the sampled augmentations for a training sample differ for each epoch, effectively increasing the training set size. In addition to these on-the-fly augmentations, all images are also mirrored (and the respective procedure is applied to the annotations), which however is performed offline.

B

GPU-BASED GAUSS-NEWTON OPTIMIZER

In Chapter 7, an efficient GPU-based Gauss Newton optimizer was used to reconstruct two interacting hands in real time. This appendix provides additional details about the implementation of the optimizer that is used to minimize the nonlinear least-squares energy function.

For the Gauss-Newton optimization steps, the non-constant entries of the Jacobian matrix $J \in \mathbb{R}^{8871 \times 122}$ and the residuals $f \in \mathbb{R}^{8871}$ are computed using CUDA kernels on the GPU. The structure is setup to ensure that all threads in the same block compute derivatives for the same energy term. Subsequently, the matrix-matrix and matrix-vector products $J^\top J$ and $J^\top f$ are computed using an efficient implementation in shared memory. For solving the linear system $J^\top J \cdot \delta = J^\top f$, $J^\top J \in \mathbb{R}^{122 \times 122}$ and $J^\top f \in \mathbb{R}^{122}$ are copied to the CPU and the preconditioned conjugate gradient (PCG) solver of the Eigen library is employed to obtain the parameter update δ.

BIBLIOGRAPHY

Abadi, Martín, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. (2016). "Tensorflow: Large-Scale Machine Learning on Heterogeneous Distributed Systems." In: *arXiv preprint arXiv:1603.04467*.

Alias | Wavefront (1998). *Maya 1.0 Using Maya*.

Alp Güler, Riza, George Trigeorgis, Epameinondas Antonakos, Patrick Snape, Stefanos Zafeiriou, and Iasonas Kokkinos (2017). "DenseReg: Fully Convolutional Dense Shape Regression In-The-Wild." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 6799–6808.

Alp Güler, Riza, Natalia Neverova, and Iasonas Kokkinos (2018). "DensePose: Dense Human Pose Estimation in the Wild." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 7297–7306.

Athitsos, Vassilis and Stan Sclaroff (2003). "Estimating 3D Hand Pose from a Cluttered Image." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 432–442.

Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla (2017). "Segnet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 39.12, pp. 2481–2495.

Baek, Seungryul, Kwang In Kim, and Tae-Kyun Kim (2018). "Augmented Skeleton Space Transfer for Depth-Based Hand Pose Estimation." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 8330–8339.

Baek, Seungryul, Kwang In Kim, and Tae-Kyun Kim (2019). "Pushing the Envelope for RGB-Based Dense 3D Hand Pose Estimation via Neural Rendering." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1067–1076.

Ballan, Luca, Aparna Taneja, Juergen Gall, Luc Van Gool, and Marc Pollefeys (2012). "Motion Capture of Hands in Action using Discriminative Salient Points." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 640–653.

Bojja, Abhishake Kumar, Franziska Mueller, Sri Raghu Malireddi, Markus Oberweger, Vincent Lepetit, Christian Theobalt, Kwang Moo Yi, and Andrea Tagliasacchi (2019). "HandSeg: An Automatically Labeled

Dataset for Hand Segmentation from Depth Images." In: *Proceedings of the Conference on Computer and Robot Vision (CRV)*. IEEE, pp. 151–158.

Boukhayma, Adnane, Rodrigo de Bem, and Philip H.S. Torr (2019). "3D Hand Shape and Pose From Images in the Wild." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 10843–10852.

Brau, Ernesto and Hao Jiang (2016). "3D Human Pose Estimation via Deep Learning from 2D Annotations." In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, pp. 582–591.

Bronstein, Michael M, Alexander M Bronstein, Ron Kimmel, and Irad Yavneh (2006). "Multigrid Multidimensional Scaling." In: *Numerical Linear Algebra with Applications* 13.2-3, pp. 149–171.

Bullock, Ian M, Júlia Borràs, and Aaron M Dollar (2012). "Assessing Assumptions in Kinematic Hand Models: A Review." In: *Proceedings of the IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. IEEE, pp. 139–146.

Cai, Yujun, Liuhao Ge, Jianfei Cai, and Junsong Yuan (2018). "Weakly-Supervised 3D Hand Pose Estimation from Monocular RGB Images." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 1–17.

Canziani, Alfredo, Adam Paszke, and Eugenio Culurciello (2016). "An Analysis of Deep Neural Network Models for Practical Applications." In: *arXiv preprint arXiv:1605.07678*.

Casiez, Géry, Nicolas Roussel, and Daniel Vogel (2012). "1€ Filter: A Simple Speed-Based Low-Pass Filter for Noisy Input in Interactive Systems." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 2527–2530.

Chan, Edwin, Teddy Seyed, Wolfgang Stuerzlinger, Xing-Dong Yang, and Frank Maurer (2016). "User Elicitation on Single-Hand Microgestures." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 3403–3414.

Chan, Liwei, Rong-Hao Liang, Ming-Chang Tsai, Kai-Yin Cheng, Chao-Huai Su, Mike Y Chen, Wen-Huang Cheng, and Bing-Yu Chen (2013). "FingerPad: Private and Subtle Interaction Using Fingertips." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 255–260.

Chan, Liwei, Chi-Hao Hsieh, Yi-Ling Chen, Shuo Yang, Da-Yuan Huang, Rong-Hao Liang, and Bing-Yu Chen (2015a). "Cyclops: Wearable and Single-Piece Full-Body Gesture Input Devices." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 3001–3009.

Chan, Liwei, Yi-Ling Chen, Chi-Hao Hsieh, Rong-Hao Liang, and Bing-Yu Chen (2015b). "Cyclopsring: Enabling Whole-Hand and Context-Aware Interactions Through a Fisheye Ring." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 549–556.

Chen, Ke-Yu, Shwetak N. Patel, and Sean Keller (2016). "Finexus: Tracking Precise Motions of Multiple Fingertips Using Magnetic Sensing." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 1504–1514.

Chen, Qifeng and Vladlen Koltun (2017). "Photographic Image Synthesis with Cascaded Refinement Networks." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 1511–1520.

Colaço, Andrea, Ahmed Kirmani, Hye Soo Yang, Nan-Wei Gong, Chris Schmandt, and Vivek K Goyal (2013). "Mime: Compact, Low Power 3D Gesture Sensing for Interaction with Head Mounted Displays." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 227–236.

Dementyev, Artem and Joseph A Paradiso (2014). "WristFlex: Low-Power Gesture Input with Wrist-Worn Pressure Sensors." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 161–166.

Funk, Markus, Sven Mayer, Michael Nistor, and Albrecht Schmidt (2016). "Mobile In-Situ Pick-by-Vision: Order Picking Support Using a Projector Helmet." In: *Proceedings of the International Conference on PErvasive Technologies Related to Assistive Environments (PETRA)*. ACM, 45:1–45:4.

Ganin, Yaroslav and Victor Lempitsky (2015). "Unsupervised domain adaptation by backpropagation." In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 1180–1189.

Ge, Liuhao, Hui Liang, Junsong Yuan, and Daniel Thalmann (2016). "Robust 3D Hand Pose Estimation in Single Depth Images: from Single-View CNN to Multi-View CNNs." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Ge, Liuhao, Yujun Cai, Junwu Weng, and Junsong Yuan (2018). "Hand PointNet: 3D Hand Pose Estimation Using Point Sets." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Glauser, Oliver, Shihao Wu, Daniele Panozzo, Otmar Hilliges, and Olga Sorkine-Hornung (2019). "Interactive Hand Pose Estimation Using a Stretch-Sensing Soft Glove." In: *ACM Transactions on Graphics (TOG)* 38.4, pp. 1–15.

Gomez-Donoso, Francisco, Sergio Orts-Escolano, and Miguel Cazorla (2017). "Large-Scale Multiview 3D Hand Pose Dataset." In: *arXiv preprint arXiv:1707.03742*.

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative Adversarial Nets." In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2672–2680.

Greff, Klaus, Rupesh K Srivastava, and Jürgen Schmidhuber (2016). "Highway and Residual Networks Learn Unrolled Iterative Estimation." In: *Proceedings of the International Conference on Learning Representations (ICLR)*.

Gustafson, Sean G., Bernhard Rabe, and Patrick M. Baudisch (2013). "Understanding Palm-based Imaginary Interfaces: The Role of Visual and Tactile Cues when Browsing." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 889–898.

Hamer, Henning, Konrad Schindler, Esther Koller-Meier, and Luc Van Gool (2009). "Tracking a Hand Manipulating an Object." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 1475–1482.

Han, Shangchen, Beibei Liu, Robert Wang, Yuting Ye, Christopher D Twigg, and Kenrick Kin (2018). "Online Optical Marker-based Hand Tracking with Deep Labels." In: *ACM Transactions on Graphics (TOG)* 37.4, pp. 1–10.

Harrison, Chris, Desney Tan, and Dan Morris (2010). "Skinput: Appropriating the Body as an Input Surface." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 453–462.

Harrison, Chris, Hrvoje Benko, and Andrew D Wilson (2011). "OmniTouch: wearable multitouch interaction everywhere." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 441–450.

Hasson, Yana, Gül Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J. Black, Ivan Laptev, and Cordelia Schmid (2019). "Learning Joint Reconstruction of Hands and Manipulated Objects." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 11807–11816.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep Residual Learning for Image Recognition." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 770–778.

Heap, Tony and David Hogg (1996). "Towards 3D Hand Tracking using a Deformable Model." In: *Proceedings of the International Conference on Automatic Face and Gesture Recognition (FG)*. IEEE, pp. 140–145.

Höll, Markus, Markus Oberweger, Clemens Arth, and Vincent Lepetit (2018). "Efficient Physics-Based Implementation for Realistic Hand-

Object Interaction in Virtual Reality." In: *Proceedings of the Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, pp. 175–182.

Howard, Andrew G, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam (2017). "Mobilenets: Efficient Convolutional Neural Networks for Mobile Vision Applications." In: *arXiv:1704.04861*.

Hrabia, Christopher-Eyk, Katrin Wolf, and Mathias Wilhelm (2013). "Whole Hand Modeling using 8 Wearable Sensors: Biomechanics for Hand Pose Prediction." In: *Proceedings of the Augmented Human International Conference*. ACM, pp. 21–28.

Huang, Da-Yuan, Ming-Chang Tsai, Ying-Chao Tung, Min-Lun Tsai, Yen-Ting Yeh, Liwei Chan, Yi-Ping Hung, and Mike Y Chen (2014). "TouchSense: Expanding Touchscreen Input Vocabulary using Different Areas of Users' Finger Pads." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 189–192.

Huang, Da-Yuan, Liwei Chan, Shuo Yang, Fan Wang, Rong-Hao Liang, De-Nian Yang, Yi-Ping Hung, and Bing-Yu Chen (2016). "DigitSpace: Designing Thumb-to-Fingers Touch Interfaces for One-Handed and Eyes-Free Interactions." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 1526–1537.

IntelRealSenseSR300 (2016). https://www.intelrealsense.com/coded-light.

Ioffe, Sergey and Christian Szegedy (2015). "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift." In: *Proceedings of the International Conference on Machine Learning (ICML)*, pp. 448–456.

Iqbal, Umar, Pavlo Molchanov, Thomas Breuel Juergen Gall, and Jan Kautz (2018). "Hand Pose Estimation via Latent 2.5D Heatmap Regression." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 118–134.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros (2017). "Image-to-Image Translation with Conditional Adversarial Networks." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1125–1134.

Jia, Yangqing, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell (2014). "Caffe: Convolutional Architecture for Fast Feature Embedding." In: *Proceedings of the International Conference on Multimedia*. ACM, pp. 675–678.

Johnson, Sam and Mark Everingham (2010). "Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation." In: *Proceedings of the British Machine Vision Conference (BMVC)*. Vol. 2. 4, p. 5.

Jolliffe, Ian T (1986). "Principal Components in Regression Analysis." In: *Principal Component Analysis*. Springer, pp. 129–155.

Kao, Hsin-Liu (Cindy), Artem Dementyev, Joseph A. Paradiso, and Chris Schmandt (2015). "NailO: Fingernails as an Input Surface." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 3015–3018.

Keskin, Cem, Furkan Kıraç, Yunus Emre Kara, and Lale Akarun (2012). "Hand Pose Estimation and Hand Shape Classification using Multi-Layered Randomized Decision Forests." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 852–863.

Khamis, Sameh, Jonathan Taylor, Jamie Shotton, Cem Keskin, Shahram Izadi, and Andrew Fitzgibbon (2015). "Learning an Efficient Model of Hand Shape Variation from Depth Images." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 2540–2548.

Kim, David, Otmar Hilliges, Shahram Izadi, Alex D Butler, Jiawen Chen, Iason Oikonomidis, and Patrick Olivier (2012). "Digits: Freehand 3D Interactions Anywhere using a Wrist-Worn Gloveless Sensor." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 167–176.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization." In: *arXiv preprint arXiv:1412.6980*.

Koller, Oscar, O Zargaran, Hermann Ney, and Richard Bowden (2016). "Deep Sign: Hybrid CNN-HMM for Continuous Sign Language Recognition." In: *Proceedings of the British Machine Vision Conference (BMVC)*.

Krupka, Eyal, Kfir Karmon, Noam Bloom, Daniel Freedman, Ilya Gurvich, Aviv Hurvitz, Ido Leichter, Yoni Smolin, Yuval Tzairi, Alon Vinnikov, and Aharon Bar Hillel (2017). "Toward Realistic Hands Gesture Interface: Keeping it Simple for Developers and Machines." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM.

Kyriazis, Nikolaos and Antonis Argyros (2014). "Scalable 3D Tracking of Multiple Interacting Objects." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 3430–3437.

LeapMotion (2016). https://developer.leapmotion.com/orion.

Lee, Chen-Yu, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu (2015). "Deeply-Supervised Nets." In: *Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*. Vol. 2. 3, pp. 562–570.

Lee, Taehee and Tobias Hollerer (2009). "Multithreaded Hybrid Feature Tracking for Markerless Augmented Reality." In: *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 15.3, pp. 355–368.

Lewis, John P, Matt Cordner, and Nickson Fong (2000). "Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation." In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. ACM, pp. 165–172.

Li, Peiyi, Haibin Ling, Xi Li, and Chunyuan Liao (2015). "3D Hand Pose Estimation using Randomized Decision Forest with Segmentation Index Points." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 819–827.

Lien, Jaime, Nicholas Gillian, M Emre Karagozler, Patrick Amihood, Carsten Schwesig, Erik Olson, Hakim Raja, and Ivan Poupyrev (2016). "Soli: Ubiquitous Gesture Sensing with Millimeter Wave Radar." In: *ACM Transactions on Graphics (TOG)* 35.4, pp. 1–19.

Liu, Jian, Hossein Rahmani, Naveed Akhtar, and Ajmal Mian (2019). "Learning Human Pose Models from Synthesized Data for Robust RGB-D Action Recognition." In: *International Journal of Computer Vision (IJCV)* 127.10, pp. 1545–1564.

Loclair, Christian, Sean Gustafson, and Patrick Baudisch (2010). "PinchWatch: A Wearable Device for One-Handed Microinteractions." In: *Proceedings of the Conference on Human-Computer-Interaction with Mobile Devices and Services (MobileHCI)*. ACM.

Markussen, Anders, Mikkel Rønne Jakobsen, and Kasper Hornbæk (2014). "Vulture: A Mid-air Word-gesture Keyboard." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. ACM, pp. 1073–1082.

Mehta, Dushyant, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt (2017a). "Monocular 3D Human Pose Estimation In The Wild Using Improved CNN Supervision." In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, pp. 506–516.

Mehta, Dushyant, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt (2017b). "VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera." In: *ACM Transactions on Graphics (TOG)* 36.4.

Mehta, Dushyant, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Srinath Sridhar, Gerard Pons-Moll, and Christian Theobalt (2018). "Single-Shot Multi-Person 3D Pose Estimation From Monocular RGB." In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, pp. 120–130.

Mehta, Dushyant, Oleksandr Sotnychenko, Franziska Mueller, Weipeng Xu, Mohamed Elgharib, Pascal Fua, Hans-Peter Seidel, Helge Rhodin, Gerard Pons-Moll, and Christian Theobalt (2020). "XNect: Real-time

Multi-Person 3D Motion Capture with a Single RGB Camera." In: *ACM Transactions on Graphics (TOG)* 39.4.

Mistry, Pranav, Pattie Maes, and Liyan Chang (2009). "WUW-wear Ur world: A Wearable Gestural Interface." In: *Extended Abstracts on Human Factors in Computing Systems*. ACM, pp. 4111–4116.

Mueller, Franziska, Dushyant Mehta, Oleksandr Sotnychenko, Srinath Sridhar, Dan Casas, and Christian Theobalt (2017). "Real-time Hand Tracking under Occlusion from an Egocentric RGB-D Sensor." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 1163–1172.

Mueller, Franziska, Florian Bernard, Oleksandr Sotnychenko, Dushyant Mehta, Srinath Sridhar, Dan Casas, and Christian Theobalt (2018). "GANerated Hands for Real-Time 3D Hand Tracking from Monocular RGB." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 49–59.

Mueller, Franziska, Micah Davis, Florian Bernard, Oleksandr Sotnychenko, Mickeal Verschoor, Miguel A Otaduy, Dan Casas, and Christian Theobalt (2019). "Real-time Pose and Shape Reconstruction of Two Interacting Hands with a Single Depth Camera." In: *ACM Transactions on Graphics (TOG)* 38.4, pp. 1–13.

Newell, Alejandro, Kaiyu Yang, and Jia Deng (2016). "Stacked Hourglass Networks for Human Pose Estimation." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 483–499.

Oberweger, Markus, Paul Wohlhart, and Vincent Lepetit (2015). "Training a Feedback Loop for Hand Pose Estimation." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 3316–3324.

Oberweger, Markus, Gernot Riegler, Paul Wohlhart, and Vincent Lepetit (2016). "Efficiently Creating 3D Training Data for Fine Hand Pose Estimation." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

Ogata, Masa, Yuta Sugiura, Hirotaka Osawa, and Michita Imai (2012). "iRing: Intelligent Ring Using Infrared Reflection." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*, pp. 131–136.

Ogata, Masa, Yuta Sugiura, Yasutoshi Makino, Masahiko Inami, and Michita Imai (2013). "SenSkin: Adapting Skin as a Soft Interface." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 539–544.

Oikonomidis, Iason, Nikolaos Kyriazis, and Antonis A Argyros (2011a). "Efficient Model-Based 3D Tracking of Hand Articulations using Kinect." In: *Proceedings of the British Machine Vision Conference (BMVC)*. Vol. 1. 2.

– (2011b). "Full DOF Tracking of a Hand Interacting with an Object by Modeling Occlusions and Physical Constraints." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 2088–2095.

Oikonomidis, Iasonas, Nikolaos Kyriazis, and Antonis A Argyros (2012). "Tracking the Articulated Motion of Two Strongly Interacting Hands." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1862–1869.

Panteleris, Paschalis and Antonis Argyros (2017). "Back to RGB: 3D Tracking of Hands and Hand-Object Interactions Based on Short-Baseline Stereo." In: *Proceedings of the International Conference on Computer Vision (ICCV) Workshops*. IEEE, pp. 575–584.

Panteleris, Paschalis, Iason Oikonomidis, and Antonis Argyros (2018). "Using a Single RGB Frame for Real Time 3D Hand Pose Estimation in the Wild." In: *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 436–445.

Peng, Xingchao and Kate Saenko (2018). "Synthetic to Real Adaptation with Deep Generative Correlation Alignment Networks." In: *Proceedings of the Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 1982–1991.

Piumsomboon, Thammathip, Adrian Clark, Mark Billinghurst, and Andy Cockburn (2013). "User-Defined Gestures for Augmented Reality." In: *Extended Abstracts on Human Factors in Computing Systems*. ACM, pp. 955–960.

Poston, Alan (2000). "Human Engineering Design Data Digest." In: *Washington, DC: Department of Defense Human Factors Engineering Technical Advisory Group*.

Prätorius, Manuel, Dimitar Valkov, Ulrich Burgbacher, and Klaus Hinrichs (2014). "DigiTap: An Eyes-Free VR/AR Symbolic Input Device." In: *Proceedings of the Symposium on Virtual Reality Software and Technology*. ACM, pp. 9–18.

Qian, Chen, Xiao Sun, Yichen Wei, Xiaoou Tang, and Jian Sun (2014). "Realtime and Robust Hand Tracking from Depth." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1106–1113.

Qian, Neng, Jiayi Wang, Franziska Mueller, Florian Bernard, Vladislav Golyanik, and Christian Theobalt (2020). "HTML: A Parametric Hand Texture Model for 3D Hand Reconstruction and Personalization." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer.

Remelli, Edoardo, Anastasia Tkach, Andrea Tagliasacchi, and Mark Pauly (2017). "Low-Dimensionality Calibration Through Local Anisotropic

Scaling for Robust Hand Model Personalization." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 2535–2543.

Rhodin, Helge, Christian Richardt, Dan Casas, Eldar Insafutdinov, Mohammad Shafiei, Hans-Peter Seidel, Bernt Schiele, and Christian Theobalt (2016). "EgoCap: Egocentric Marker-less Motion Capture with Two Fisheye Cameras." In: *ACM Transactions on Graphics (TOG)* 35.6, pp. 1–11.

Rogez, Grégory, Maryam Khademi, JS Supančič III, Jose Maria Martinez Montiel, and Deva Ramanan (2014). "3D Hand Pose Detection in Egocentric RGB-D Images." In: *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*. Springer, pp. 356–371.

Rogez, Grégory, James S Supančič, and Deva Ramanan (2015). "First-Person Pose Recognition using Egocentric Workspaces." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 4325–4333.

Rohrbach, Marcus, Sikandar Amin, Mykhaylo Andriluka, and Bernt Schiele (2012). "A Database for Fine Grained Activity Detection of Cooking Activities." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1194–1201.

Romero, J., H. Kjellström, and D. Kragic (2010). "Hands in Action: Real-time 3D Reconstruction of Hands in Interaction with Objects." In: *Proceedings of the International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 458–463.

Romero, Javier, Dimitrios Tzionas, and Michael J. Black (2017). "Embodied Hands: Modeling and Capturing Hands and Bodies Together." In: *ACM Transactions on Graphics (TOG)* 36.6, 245:1–245:17.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation." In: *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, pp. 234–241.

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei (2015). "ImageNet Large Scale Visual Recognition Challenge." In: *International Journal of Computer Vision (IJCV)* 115.3, pp. 211–252.

Sangkloy, Patsorn, Jingwan Lu, Chen Fang, FIsher Yu, and James Hays (2017). "Scribbler: Controlling Deep Image Synthesis with Sketch and Color." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 5400–5409.

Schönemann, Peter H (1966). "A Generalized Solution of the Orthogonal Procrustes Problem." In: *Psychometrika* 31.1, pp. 1–10.

Seo, Kyeongeun and Hyeonjoong Cho (2014). "AirPincher: A Handheld Device for Recognizing Delicate Mid-Air Hand Gestures." In: *Proceedings of the Adjunct Publication of the Symposium on User Interface Software and Technology (UIST Adjunct)*. ACM, pp. 83–84.

Sharp, Toby, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, et al. (2015). "Accurate, Robust, and Flexible Real-time Hand Tracking." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 3633–3642.

Shrivastava, Ashish, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russ Webb (2017). "Learning from Simulated and Unsupervised Images through Adversarial Training." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 2107–2116.

Simon, Tomas, Hanbyul Joo, Iain Matthews, and Yaser Sheikh (2017). "Hand Keypoint Detection in Single Images using Multiview Bootstrapping." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 1145–1153.

Sinha, Ayan, Chiho Choi, and Karthik Ramani (2016). "DeepHand: Robust Hand Pose Estimation by Completing a Matrix Imputed with Deep Features." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 4150–4158.

Softimage (1992). *Softimage V2.51 User's Guide*.

Soliman, Mohamed, Franziska Mueller, Lena Hegemann, Joan Sol Roo, Christian Theobalt, and Jürgen Steimle (2018). "FingerInput: Capturing Expressive Single-Hand Thumb-to-Finger Microgestures." In: *Proceedings of the International Conference on Interactive Surfaces and Spaces (ISS)*. ACM, pp. 177–187.

Spurr, Adrian, Jie Song, Seonwook Park, and Otmar Hilliges (2018). "Cross-Modal Deep Variational Hand Pose Estimation." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 89–98.

Sridhar, Srinath, Antti Oulasvirta, and Christian Theobalt (2013). "Interactive Markerless Articulated Hand Motion Tracking using RGB and Depth Data." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 2456–2463.

Sridhar, Srinath, Helge Rhodin, Hans-Peter Seidel, Antti Oulasvirta, and Christian Theobalt (2014). "Real-time Hand Tracking Using a Sum of Anisotropic Gaussians Model." In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, pp. 319–326.

Sridhar, Srinath, Franziska Mueller, Antti Oulasvirta, and Christian Theobalt (2015a). "Fast and Robust Hand Tracking Using Detection-

Guided Optimization." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 3213–3221.

Sridhar, Srinath, Anna Maria Feit, Christian Theobalt, and Antti Oulasvirta (2015b). "Investigating the Dexterity of Multi-Finger Input for Mid-Air Text Entry." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 3643–3652.

Sridhar, Srinath, Franziska Mueller, Michael Zollhöefer, Dan Casas, Antti Oulasvirta, and Christian Theobalt (2016). "Real-time Joint Tracking of a Hand Manipulating an Object from RGB-D Input." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 294–310.

Sridhar, Srinath, Anders Markussen, Antti Oulasvirta, Christian Theobalt, and Sebastian Boring (2017). "WatchSense: On- and Above-Skin Input Sensing through a Wearable Depth Sensor." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 3891–3902.

Stenger, Björn, Arasanathan Thayananthan, Philip HS Torr, and Roberto Cipolla (2006). "Model-based Hand Tracking using a Hierarchical Bayesian Filter." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 28.9, pp. 1372–1384.

Stoll, Carsten, Nils Hasler, Juergen Gall, Hans-Peter Seidel, and Christian Theobalt (2011). "Fast Articulated Motion Tracking using a Sums of Gaussians Body Model." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 951–958.

Sun, Xiao, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun (2015). "Cascaded Hand Pose Regression." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 824–832.

Supančič, James Steven, Grégory Rogez, Yi Yang, Jamie Shotton, and Deva Ramanan (2018). "Depth-Based Hand Pose Estimation: Methods, Data, and Challenges." In: *International Journal of Computer Vision (IJCV)* 126.11, pp. 1180–1198.

Tagliasacchi, Andrea, Matthias Schroeder, Anastasia Tkach, Sofien Bouaziz, Mario Botsch, and Mark Pauly (2015). "Robust Articulated-ICP for Real-Time Hand Tracking." In: *Computer Graphics Forum (Symposium on Geometry Processing)*. Vol. 34. 5. Wiley Online Library, pp. 101–114.

Tamaki, Emi, Takashi Miyaki, and Jun Rekimoto (2009). "Brainy Hand: An Ear-worn Hand Gesture Interaction Device." In: *Extended Abstracts on Human Factors in Computing Systems*. ACM, pp. 4255–4260.

Tan, David Joseph, Thomas Cashman, Jonathan Taylor, Andrew Fitzgibbon, Daniel Tarlow, Sameh Khamis, Shahram Izadi, and Jamie Shotton (2016). "Fits Like a Glove: Rapid and Reliable Hand Shape Personal-

ization." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 5610–5619.

Tang, Danhang, Hyung Jin Chang, Alykhan Tejani, and Tae-Kyun Kim (2014). "Latent Regression Forest: Structured Estimation of 3D Articulated Hand Posture." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 3786–3793.

Taylor, Jonathan, Lucas Bordeaux, Thomas Cashman, Bob Corish, Cem Keskin, Toby Sharp, Eduardo Soto, David Sweeney, Julien Valentin, Benjamin Luff, et al. (2016). "Efficient and Precise Interactive Hand Tracking through Joint, Continuous Optimization of Pose and Correspondences." In: *ACM Transactions on Graphics (TOG)* 35.4, pp. 1–12.

Taylor, Jonathan, Vladimir Tankovich, Danhang Tang, Cem Keskin, David Kim, Philip Davidson, Adarsh Kowdle, and Shahram Izadi (2017). "Articulated Distance Fields for Ultra-fast Tracking of Hands Interacting." In: *ACM Transactions on Graphics (TOG)* 36.6, 244:1–244:12.

Tekin, Bugra, Federica Bogo, and Marc Pollefeys (2019). "H+O: Unified Egocentric Recognition of 3D Hand-Object Poses and Interactions." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 4511–4520.

Ten Berge, Jos M F (2006). "The Rigid Orthogonal Procrustes Rotation Problem." In: *Psychometrika* 71.1, pp. 201–205.

Tkach, Anastasia, Mark Pauly, and Andrea Tagliasacchi (2016). "Sphere-Meshes for Real-time Hand Modeling and Tracking." In: *ACM Transactions on Graphics (TOG)* 35.6, 222:1–222:11.

Tkach, Anastasia, Andrea Tagliasacchi, Edoardo Remelli, Mark Pauly, and Andrew Fitzgibbon (2017). "Online Generative Model Personalization for Hand Tracking." In: *ACM Transactions on Graphics (TOG)* 36.6, 243:1–243:11.

Tompson, Jonathan, Murphy Stein, Yann Lecun, and Ken Perlin (2014). "Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks." In: *ACM Transactions on Graphics (TOG)* 33.5, pp. 1–10.

Tsai, Hsin-Ruey, Cheng-Yuan Wu, Lee-Ting Huang, and Yi-Ping Hung (2016a). "ThumbRing: Private Interactions using One-Handed Thumb Motion Input on Finger Segments." In: *Proceedings of the International Conference on Human-Computer Interaction with Mobile Devices and Services Adjunct (MobileHCI Adjunct)*. ACM, pp. 791–798.

Tsai, Hsin-Ruey, Min-Chieh Hsiu, Jui-Chun Hsiao, Lee-Ting Huang, Mike Chen, and Yi-Ping Hung (2016b). "TouchRing: Subtle and Always-Available Input using a Multi-Touch Ring." In: pp. 891–898.

Tsai, Hsin-Ruey, Te-Yen Wu, Da-Yuan Huang, Min-Chieh Hsiu, Jui-Chun Hsiao, Yi-Ping Hung, Mike Y Chen, and Bing-Yu Chen (2017). "SegTouch: Enhancing Touch Input While Providing Touch Gestures on Screens Using Thumb-To-Index-Finger Gestures." In: ACM, pp. 2164–2171.

Tsukadaa, Koji and Michiaki Yasumurab (2001). "Ubi-Finger: Gesture Input Device for Mobile Use." In: *Ubicomp 2001 Informal Companion Proceedings*. Vol. 11.

Tzeng, Eric, Judy Hoffman, Kate Saenko, and Trevor Darrell (2017). "Adversarial Discriminative Domain Adaptation." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. Vol. 1. 2. IEEE, pp. 7167–7176.

Tzionas, Dimitrios, Abhilash Srikantha, Pablo Aponte, and Juergen Gall (2014). "Capturing Hand Motion with an RGB-D Sensor, Fusing a Generative Model with Salient Points." In: *Proceedings of the German Conference on Pattern Recognition (GCPR)*. Springer, pp. 277–289.

Tzionas, Dimitrios, Luca Ballan, Abhilash Srikantha, Pablo Aponte, Marc Pollefeys, and Juergen Gall (2016). "Capturing Hands in Action using Discriminative Salient Points and Physics Simulation." In: *International Journal of Computer Vision (IJCV)* 118.2, pp. 172–193.

Unity (2005). https://unity.com.

Varol, Gül, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid (2017). "Learning from Synthetic Humans." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 109–117.

Verschoor, Mickeal, Daniel Lobo, and Miguel A Otaduy (2018). "Soft Hand Simulation for Smooth and Robust Natural Interaction." In: *Proceedings of the Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, pp. 183–190.

Wan, Chengde, Angela Yao, and Luc Van Gool (2016). "Hand Pose Estimation from Local Surface Normals." In: *European Conference on Computer Vision*. Springer, pp. 554–569.

Wan, Chengde, Thomas Probst, Luc Van Gool, and Angela Yao (2017). "Crossing Nets: Combining GANs and VAEs with a Shared Latent Space for Hand Pose Estimation." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 680–689.

Wang, Jiayi, Franziska Mueller, Florian Bernard, and Christian Theobalt (2020). "Generative Model-Based Loss to the Rescue: A Method to Overcome Annotation Errors for Depth-Based Hand Pose Estimation." In: *Proceedings of the International Conference on Automatic Face and Gesture Recognition (FG)*. IEEE, pp. 93–100.

Wang, Robert Y. and Jovan Popović (2009). "Real-time Hand-Tracking with a Color Glove." In: *ACM Transactions on Graphics (TOG)* 28.3, pp. 1–8.

Wang, Robert, Sylvain Paris, and Jovan Popović (2011). "6D Hands: Markerless Hand-Tracking for Computer Aided Design." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 549–558.

Wang, Saiwen, Jie Song, Jamie Lien, Ivan Poupyrev, and Otmar Hilliges (2016). "Interacting with Soli: Exploring Fine-Grained Dynamic Gesture Recognition in the Radio-Frequency Spectrum." In: *Proceedings of the Symposium on User Interface Software and Technology (UIST)*. ACM, pp. 851–860.

Weigel, Martin, Tong Lu, Gilles Bailly, Antti Oulasvirta, Carmel Majidi, and Jürgen Steimle (2015). "iSkin: Flexible, Stretchable and Visually Customizable On-Body Touch Sensors for Mobile Computing." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 2991–3000.

Weigel, Martin, Aditya Shekhar Nittala, Alex Olwal, and Jürgen Steimle (2017). "SkinMarks: Enabling Interactions on Body Landmarks Using Conformal Skin Electronics." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 3095–3105.

Whitmire, Eric, Mohit Jain, Divye Jain, Greg Nelson, and Ravi Karkar (2017). "DigiTouch : Reconfigurable Thumb-to-Finger Input and Text Entry on Head-mounted Displays." In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.3, pp. 1–21.

Winkler, Christian, Julian Seifert, David Dobbelstein, and Enrico Rukzio (2014). "Pervasive Information Through Constant Personal Projection: The Ambient Mobile Pervasive Display (AMP-D)." In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 4117–4126.

Wolf, Katrin, Anja Naumann, Michael Rohs, and Jörg Müller (2011). "A Taxonomy of Microinteractions: Defining Microgestures based on Ergonomic and Scenario-Dependent Requirements." In: *Proceedings of the IFIP Conference on Human-Computer Interaction*. Springer, pp. 559–575.

Xu, Chi and Li Cheng (2013). "Efficient Hand Pose Estimation from a Single Depth Image." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 3456–3462.

Yang, Linlin, Shile Li, Dongheui Lee, and Angela Yao (2019). "Aligning Latent Spaces for 3D Hand Pose Estimation." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 2335–2343.

Ye, Qi and Tae-Kyun Kim (2018). "Occlusion-aware Hand Pose Estimation Using Hierarchical Mixture Density Network." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 801–817.

Ye, Qi, Shanxin Yuan, and Tae-Kyun Kim (2016). "Spatial Attention Deep Net with Partial PSO for Hierarchical Hybrid Hand Pose Estimation." In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Springer, pp. 346–361.

Yenamandra, Tarun, Florian Bernard, Jiayi Wang, Franziska Mueller, and Christian Theobalt (2019). "Convex Optimisation for Inverse Kinematics." In: *Proceedings of the International Conference on 3D Vision (3DV)*. IEEE, pp. 318–327.

Yoon, Sang Ho, Ke Huo, and Karthik Ramani (2014). "Plex: Finger-worn Textile Sensor for Mobile Interaction During Activities." In: *Proceedings of the International Joint Conference on Pervasive and Ubiquitous Computing: Adjunct Publication*. ACM, pp. 191–194.

Yoon, Sang Ho, Ke Huo, Vinh P. Nguyen, and Karthik Ramani (2015). "TIMMi: Finger-worn Textile Input Device with Multimodal Sensing in Mobile Interaction." In: *Proceedings of the International Conference on Tangible, Embedded, and Embodied Interaction*, pp. 269–272.

Yuan, Shanxin, Guillermo Garcia-Hernando, Björn Stenger, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, Junsong Yuan, Xinghao Chen, Guijin Wang, Fan Yang, Kai Akiyama, Yang Wu, Qingfu Wan, Meysam Madadi, Sergio Escalera, Shile Li, Dongheui Lee, Iason Oikonomidis, Antonis Argyros, and Tae-Kyun Kim (2018). "Depth-Based 3D Hand Pose Estimation: From Current Achievements to Future Goals." In: *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, pp. 2636–2645.

Zeiler, Matthew D (2012). "Adadelta: An Adaptive Learning Rate Method." In: *arXiv preprint arXiv:1212.5701*.

Zhang, Cheng, Anandghan Waghmare, Pranav Kundra, Yiming Pu, Scott Gilliland, Thomas Ploetz, Thad E Starner, Omer T Inan, and Gregory D Abowd (2017a). "FingerSound: Recognizing Unistroke Thumb Gestures using a Ring." In: *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 1.3, pp. 1–19.

Zhang, Jiawei, Jianbo Jiao, Mingliang Chen, Liangqiong Qu, Xiaobin Xu, and Qingxiong Yang (2016). "3D Hand Pose Tracking and Estimation Using Stereo Matching." In: *arXiv preprint arXiv:1610.07214*.

Zhang, Kai, Wangmeng Zuo, Yunjin Chen, Deyu Meng, and Lei Zhang (2017b). "Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising." In: *IEEE Transactions on Image Processing* 26.7, pp. 3142–3155.

Zhang, Xiong, Qiang Li, Hong Mo, Wenbo Zhang, and Wen Zheng (2019). "End-to-End Hand Mesh Recovery From a Monocular RGB Image." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 2354–2364.

Zhao, Wenping, Jianjie Zhang, Jianyuan Min, and Jinxiang Chai (2013). "Robust Realtime Physics-based Motion Control for Human Grasping." In: *ACM Transactions on Graphics (TOG)* 32.6, 207:1–207:12.

Zhou, Xingyi, Qingfu Wan, Wei Zhang, Xiangyang Xue, and Yichen Wei (2016). "Model-based Deep Hand Pose Estimation." In: *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2421–2427.

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A Efros (2017). "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 2223–2232.

Zimmermann, Christian and Thomas Brox (2017). "Learning to Estimate 3D Hand Pose from Single RGB Images." In: *Proceedings of the International Conference on Computer Vision (ICCV)*. IEEE, pp. 4903–4911.