Efficient Tracking of the 3D Articulated Motion of Human Hands

Iason Oikonomidis
January 2015

University of Crete
Computer Science Department

Thesis submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

Doctoral Thesis Committee:
Professor Antonis Argyros, University of Crete (Advisor)
Professor Panos Trahanias, University of Crete
Dr. Manolis Lourakis, Principal Researcher, ICS-FORTH
Professor Georgios Tziritas, University of Crete
Professor Panagiotis Tsakalides, University of Crete
Professor Nikos Paragios, Ecole Central Paris
Professor Danica Kragic, Royal Institute of Technology (KTH)

The work reported in this thesis has been conducted at the Computational Vision and Robotics (CVRL) laboratory of the Institute of Computer Science (ICS) of the Foundation for Research and Technology–Hellas (FORTH) and has been financially supported by a FORTH-ICS scholarship, including funding by the European Commission through projects GRASP (FP7-215821), RoboHow.cog (FP7-288533) and WEARHAP (FP7-ICT-2011-9).

Efficient Tracking of the 3D Articulated Motion of Human Hands

Iason Oikonomidis

January 29, 2015

Acknowledgements

Studying for a doctorate degree has the potential to become a tedious and lonely process. Thanks to the people I had the luck to be surrounded, my studies felt definitely exciting. Furthermore, although there were moments that felt lonely, there were always people I could count on for support and help.

First and foremost, my supervisor, Prof. Antonis Argyros has supervised the course of my studies with great care and thoroughness. He achieved a great balance of simultaneously letting me free to explore, while always being in touch and carefully guiding our work.

Throughout the course of my studies, Dr. Lourakis and Prof. Trahanias were the two additional members of the supervisory committee. Both of them aided my efforts in any way they could and I want to thank them for this. I want to sincerely thank all the members of the examination committee for accepting the invitation and devoting the necessary time and effort, positively contributing in the examination process. I thank individually each one of the four members additionally to the supervisory committee: prof. Tziritas, prof. Tsakalides, prof. Paragios, especially since he travelled all the way from Paris for my defense and last but not least prof. Kragic.

I want to thank the people at the Computer Science Department of the University of Crete, both academic and administrative, for accepting me in the postgraduate program and supporting every step of my studies. My research has been inspired and financially supported by the Foundation for Research and Technology – Hellas and particularly the Institute of Computer Science, and the EU integrated projects GRASP, RoboHow.cog and WEARHAP. Many thanks to FORTH-ICS and the numerous collaborators in these projects.

The Computational Vision and Robotics Laboratory of ICS-FORTH has been an ideal environment in my opinion. I was very fortunate to be part of the lab for the last several years, and I want to sincerely thank all the present and past members. Apart from collaborators, I am glad to call most of them friends. Amongst all of them, I want to especially thank my close friend and collaborator Nikolaos Kyriazis.

Last but not least, I want to thank my friends outside the narrow working environment, and of course my family. My parents have always supported me in any way they could, financially and otherwise. Finally, I want to thank my Eleftheria for believing in me and supporting every step of this course.

I thank you all sincerely.

Abstract

The problem of hand pose estimation and tracking is both theoretically and practically interesting. It is a challenging problem that hasn't been solved in its full generality despite the significant amount of effort that has been devoted to it. This thesis presents methods to track the position, orientation and full articulation of human hands in various everyday scenarios.

Investigated scenarios include tracking one or two hands and tracking the hand(s) in isolation or in interaction with the environment. Design choices for the various presented methods regard the type of input, the selection of appropriate visual cues and furthermore the way they are synthesized and evaluated, as well as the optimization algorithms used to solve the formulated optimization problems. All scenarios use markerless visual observations of the scene as input. We explore the visual cues of skin color, edges, depth map, and visual hull. These observations can come either from a network of cameras or from an RGB-D sensor. The choice of input type partially mandates the visual cues that are employed.

We follow a model-based approach to the problem, formulating the pose estimation task for each frame as an optimization problem. The search space of this problem uses the adopted representation for the hand kinematics. For the case of single hand, the search space is this set of kinematics parameters, whereas for hand-object or hand-hand interaction, this search space is appropriately augmented to include all the tracked entities. This joint consideration, while resulting in optimization problems with tens of parameters, has the advantage that the interaction between the tracked objects can be effortlessly modeled and evaluated. The temporal continuity assumption is used by initializing the search for a frame near the solution for the previous frame.

Joint modeling of the observed entities in the scene allows for effortlessly treating scenarios of complex interaction between these entities. For the case of hand-object interaction, we show how the observed occlusions can provide useful information instead of being an obstacle. For the case of two hands in strong interaction, to the best of our knowledge, the presented results involve the most complex hand-hand interaction attempted so far in the relevant literature.

For the task of optimizing the objective functions that result from the adopted formulation of the problem, we use black-box optimization algorithms. Specifically, variants of Particle Swarm Optimization (PSO) are employed in most scenarios. PSO is an evolutionary optimization algorithm that is derivative-free and easily parallelizable. It is suitable for our task, since it is well-suited to multi-modal, non-differentiable objective functions. A novel evolutionary optimization algorithm is also presented in this thesis, and applied to two of the examined scenarios. This algorithm exploits the useful properties of quasi-random sampling, as well as the power of evolutionary computing.

The various computational steps of all presented methods are carefully designed so that they include parallelizable computations. It is then possible to make use of modern hardware such as the GPU architecture, resulting in practical systems that achieve real-time or interactive frame-rates.

Περίληψη

Το πρόβλημα της τρισδιάστατης παρακολούθησης του ανθρώπινου χεριού έχει τόσο θεωρητικό όσο και πρακτικό ενδιαφέρον. Είναι ένα απαιτητικό πρόβλημα που δεν έχει λυθεί στην πλήρη γενικότητά του, παρά τη σημαντική ερευνητική προσπάθεια που έχει αφιερωθεί σε αυτό. Αυτή η διατριβή αντιμετωπίζει αυτό το πρόβλημα και παρουσιάζει μεθόδους για την παρακολούθηση της 3Δ θέσης της παλάμης του χεριού και των δακτύλων σε ένα ευρύ φάσμα από ενδιαφέροντα σενάρια.

Τέτοια σενάρια περιλαμβάνουν την παρακολούθηση ενός ή δύο χεριών, καθώς και την παρακολούθηση του χεριού(-ών) μεμονωμένα ή σε αλληλεπίδραση με το περιβάλλον. Επιλογές σχετικές με τη σχεδίαση των διάφορων παρουσιαζόμενων μεθόδων αφορούν στην επιλογή κατάλληλων χαρακτηριστικών εικόνας συμπεριλαμβάνοντας τον τρόπο με τον οποίο αυτά μπορούν να συντεθούν και να αποτιμηθούν, καθώς και αλγόριθμους για την επίλυση των προβλημάτων βελτιστοποίησης που προκύπτουν. Όλα τα σενάρια προβλέπουν σαν είσοδο οπτική παρατήρηση της σκηνής χωρίς χρήση υποβοηθητικών σημαδιών. Τα χαρακτηριστικά εικόνας που χρησιμοποιούμε είναι οι ακμές, οι περιοχές χρώματος δέρματος, η απόσταση από τον αισθητήρα και το τρισδιάστατο οπτικό περίγραμμα (visual hull). Οι παρατηρήσεις μπορούν να προέρχονται είτε από ένα δίκτυο συμβατικών καμερών, είτε από μία κάμερα που επιπρόσθετα με το χρώμα καταγράφει και την απόσταση του κάθε σημείου της σκηνής από τον αισθητήρα (RGB-D sensor). Η επιλογή του τύπου εισόδου καθορίζει μερικώς και τα χρησιμοποιούμενα χαρακτηριστικά εικόνας.

Ακολουθούμε την προσέγγιση μεθόδων που βασίζονται σε μοντέλο, διατυπώνοντας το πρόβλημα της εκτίμησης πόζας σε κάθε εικόνα εισόδου σαν ένα πρόβλημα βελτιστοποίησης. Ο χώρος αναζήτησης αυτού του προβλήματος βασίζεται στη χρησιμοποιούμενη παραμετροποίηση της κινηματικής του χεριού. Για την περίπτωση του ενός χεριού, ο χώρος αναζήτησης ταυτίζεται με αυτή την παραμετροποίηση, ενώ για τις περιπτώσεις αλληλεπίδρασης χεριού-χεριού ή χεριού-αντικειμένου, αυτός ο χώρος προσαυξάνεται κατάλληλα ώστε να συμπεριλάβει όλες τις παρακολουθούμενες οντότητες. Αυτή η από κοινού θεώρηση, παρότι οδηγεί σε προβλήματα βελτιστοποίησης με δεκάδες παραμέτρων, έχει το πλεονέκτημα ότι επιτρέπει την μοντελοποίηση της αλληλεπίδρασης των παρακολουθούμενων οντοτήτων με άμεσο τρόπο. Η υπόθεση της χρονικής συνέχειας χρησιμοποιείται μέσω της αρχικοποίησης της αναζήτησης σχετικά με κάποια εικόνα στην περιοχή της εκτίμησης λύσης για την προηγούμενη χρονικά εικόνα.

Η από κοινού θεώρηση των παρατηρούμενων οντοτήτων της σκηνής επιτρέπει την αντιμετώπιση σεναρίων που περιλαμβάνουν πολύπλοκη αλληλεπίδραση ανάμεσα σε αυτές τις οντότητες. Για την περίπτωση της αλληλεπίδρασης χεριού με αντικείμενο, δείχνουμε πώς οι προκύπτουσες αλληλεπικαλύψεις μπορούν να παράσχουν χρήσιμη πληροφορία αντί να αντιμετωπίζονται ως πρόβλημα. Για την περίπτωση των δύο χεριών σε ισχυρή αλληλεπίδραση, οι αλγόριθμοι που προτείνουμε αντιμετωπίζουν την πιο περίπλοκη αλληλεπίδραση χεριών που έχει ως τώρα αναφερθεί στη σχετική βιβλιογραφία.

Για τη βελτιστοποίηση των αντικειμενικών συναρτήσεων, όπως προκύπτουν από την υιοθετούμενη διατύπωση του προβλήματος, χρησιμοποιούμε αλγόριθμους βελτιστοποίησης που δεν απαιτούν γνώση της παραγώγου της αντικειμενικής συνάρτησης. Συγκεκριμένα, στις περισσότερες περιπτώσεις, χρησιμοποιούνται παραλλαγές του αλγορίθμου Βελτιστοποίησης Σμήνους Σωματιδίων (ΒΣΣ) (Particle Swarm Optimization). Ο ΒΣΣ είναι ένας γενετικός αλγόριθμος που δεν απαιτεί γνώση της παραγώγου της αντικειμενικής συνάρτησης

που βελτιστοποιεί, και παραλληλοποιείται εύκολα. Είναι κατάλληλος για το πρόβλημα διότι μπορεί να αντιμετωπίσει μη παραγωγίσιμες συναρτήσεις με πολλά τοπικά βέλτιστα. Παρουσιάζεται επίσης ένας νέος εξελικτικός αλγόριθμος βελτιστοποίησης, και δοκιμάζεται σε δύο από τα εξεταζόμενα σενάρια παρακολούθησης της κίνησης χεριών. Αυτός ο αλγόριθμος εκμεταλλεύεται τις χρήσιμες ιδιότητες της ημι-τυχαίας δειγματοληψίας, συνδυάζοντάς τις με την δύναμη των εξελικτικών υπολογισμών.

Τα διάφορα υπολογιστικά βήματα όλων των παρουσιαζόμενων μεθόδων είναι προσεκτικά σχεδιασμένα ώστε να περιλαμβάνουν υπολογισμούς που επιδέχονται παραλλληλοποίηση. Γίνεται έτσι εφικτή η εκμετάλλευση σύγχρονων αρχιτεκτονικών όπως οι κάρτες γραφικών, έτσι ώστε τα συστήματα που προκύπτουν να επιτυγχάνουν επιδόσεις οι οποίες, ανάλογα με το πρόβλημα, είναι πραγματικού χρόνου ή κοντά σε αυτές.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Computer vision methods aim to capture, analyze and understand information contained in images. In this context many interesting questions can be posed. A non-exhaustive list of related tasks with both theoretical and practical interest includes: industrial automation in production lines and similar environments especially when other means of sensing are either costly or not applicable, automation of surveillance applications, image database indexing, medical applications such as diagnosis aiding, augmented reality for surgery aiding and patient rehabilitation, enhancing driving safety by visually detecting pedestrians and other obstacles, and gesture recognition for human-computer interaction.

The problem of effectively recovering the pose (3D position and orientation) of human body parts using visual markerless observations is interesting because of its theoretical importance and its diverse applications. The human visual system exhibits a remarkable ability to seemingly effortlessly solve it. It is interesting to understand the specific mechanisms of the human brain that are involved in this process. Furthermore, a wide range of useful applications can be implemented provided that this fundamental problem is robustly and efficiently solved [7]. Impressive motion capture systems that employ visual markers [8, 9] or other specialized hardware have been developed. However, there is intense interest in developing markerless computer-vision based solutions, because they are non-invasive and potentially cheaper than solutions based on other technologies such as electromagnetic tracking or inertial measurement units [10, 11].

1.1 Description of the Problem

In the general category of problems regarding human articulation tracking, the particular problem of 3D hand pose estimation is of special interest. Specifically, there is significant interest for practical systems that can estimate the full, time-varying pose of one or more human hands in real time. The observed hands are potentially in interaction with the environment, for example while manipulating objects. The description of the hand pose must be as detailed as possible to enable capturing the high versatility and dexterity of human hands. Observations of the scene are in the form of image sequences, either from regular color cameras, or from the most recent RGB-D sensors [12]. In all cases the observations are markerless, avoiding unnecessary interference with the scene. Furthermore it is assumed that the visual sensors are fully calibrated, i.e. both the intrinsics and extrinsics parameters are known.

1.2 Importance of the Problem

By understanding the configuration of human hands we are in a position to build systems that can capture human activities and potentially understand important aspects of the interaction of a human with her/his environment, both physical and social. Furthermore, practical solutions to the problem enable human-computer interaction in a more natural way than the usual keyboard-mouse combination. The touch-screen enabled devices that have recently gained popularity are such an example of natural user interfaces. Nevertheless, the currently widely used input methods, including touch-screens, are still lacking in scenarios that involve the manipulation of three-dimensional objects on-screen, since the input is fundamentally two-dimensional. Robust solutions to hand pose estimation and tracking enable natural ways to interact with computer in such scenarios, by effortlessly conveying three-dimensional information [9]. Human-computer interaction can benefit from robust hand tracking solutions in many different areas: natural user interfaces, sign language understanding and interpretation, robot learning by demonstration and patient rehabilitation.

Apart from practical utility, hand pose estimation and tracking are interesting in a theoretical level as well. The human visual system exhibits the ability to seemingly effortlessly perform the task, along with human body pose estimation and related problems. Algorithmic solutions to hand pose estimation may aid in advancing our understanding of the inner workings of the human brain.

1.3 Problem Difficulties

A significant amount of literature has been devoted to the problem of pose recovery of articulated objects using visual input. The problems of recovering the pose of the human body and the human hand present similarities such as the tree-like connectivity and the size variability of the articulated parts. On the other hand, a human hand usually has consistent appearance statistics (skin color), whereas the appearance of humans is much more diverse because of clothing. Moeslund et al. [7] provide a thorough review covering the general problem of visual human motion capture and analysis. Four different subproblems in the problem area of human body pose estimation are identified in [7]. These are: initialization, tracking, pose estimation and recognition. In this context, for the respective problem of hand pose estimation, the present work focuses on the middle two subproblems.

Despite the significant amount of work in the field [2], the problem of hand pose estimation remains open and presents several theoretical and practical challenges due to a number of related issues. Fundamentally, the kinematics of the human hand is complicated. Complicated kinematics is hard to accurately represent and also results in a high dimensional configuration space. Extended self-occlusions further complicate the problem by resulting in incomplete and/or ambiguous observations. The problem of tracking a hand in interaction with its environment is further complicated since the occlusions in that case also occur by the interaction with other objects.

A variety of methods have been proposed to visually capture human hand motion. Erol et al. [2] present a review of such methods. Based on the completeness of the computed pose, they differentiate between partial and full pose estimation methods. They further

divide the class of full pose estimation methods into *appearance-based* and *model-based* ones. A final distinction is made for methods that estimate a sequence of hand poses based on video input: methods that maintain a set of possible alternative solutions for each time step are called *Multiple Hypotheses* ones, whereas methods that keep only the best of the considered ones are called *Single Hypothesis*.

As listed in [2], the problem exhibits numerous difficulties. These difficulties range from issues that are encountered in most computer vision applications such as occlusions, to issues specific to the problem such as the high dimensionality of the hand configuration space:

- *High dimensionality:* The human hand exhibits more than 20 DOF for articulation alone. These can be reduced with appropriate dimensionality reduction techniques, but experiments show that no less than six can be used without significant information loss. This estimation does not even consider the case of object manipulation– the hand can take otherwise unnatural poses in this case. This high dimensionality renders naive search algorithms inefficient, giving rise to the need for sophisticated and/or specialized searching techniques.

- *Self occlusions:* Since the human hand is very versatile, some parts of it can occlude others, complicating the observation process. Unless a high-quality depth map is available, segmentation into parts is unreliable since low-level information is not sufficient in the case of occlusions. Furthermore, hierarchical approaches fail since visual evidence about parts (such as fingers or phalanges) cannot be considered in isolation to the other parts.

- *Uncontrolled environments:* Many practical applications require the use of hand pose estimation and tracking systems in cluttered background and arbitrary lighting conditions. Such environments make feature extraction processes unreliable: skin color detection is sensitive to lighting conditions, and cluttered background may hinder the performance of foreground/background segmentation methods as well as methods based on edge features.

- *Rapid hand motion:* Hand motion speeds involve $5m/s$ for translation and $300°/s$ for wrist rotation. These speeds give rise to problems such as motion blur (compromising the performance of edge detectors) and the invalidation of the temporal continuity hypothesis. Sturman in [13] recommends at least 100Hz to overcome these problems.

- *Computational complexity:* A video stream at normal frame rates (around $30Hz$) and standard resolutions (such as 640×480 pixels) has a significant rate of data. This rate is higher for multiple streams (stereo systems or more cameras), larger image resolutions and higher frame rates.

More difficulties not listed in [2] include:

- *Inaccuracies in modeling:* The human hand is composed of numerous bones held together and mobilized by an intricate system of muscles and tendons. The usually employed hand kinematics models are merely approximating the degrees of freedom of hand bones (for the complexity of the thumb base joint, see [14]). Furthermore,

the soft tissues surrounding the bones are deformed during hand motions, thus imposing difficulties in modeling the appearance of the hand.

- *Variability across humans:* Human hands are not identical: among different individuals there is a deviation in the sizes and shapes of bones, as well as the angles and limits of the joints. Ideally a system must be able to adjust to the observed hands' dimensions. Two reasons for this kind of per-person adjustments are: improved accuracy of the estimation (based on a better representation of the observations) and better correspondence of candidate poses to the actual hand configuration.

- *Interaction with the environment:* In real world scenarios it may be desirable to track the human hand while it interacts with the surroundings. This may include manipulated objects or even other hands. In such scenarios, extra difficulties are imposed because occlusions can now occur in more ways than the self-occlusions of the isolated-hand scenario.

Most of the listed difficulties are addressed in the presented work. Specifically, the high degree of dimensionality is effectively handled by powerful evolutionary optimization algorithms. Furthermore, the high level of self occlusions is inherently tackled by the adopted model-based approach, by jointly considering the observed scene. Interaction with objects of known geometry is also similarly treated. On the other hand, the variability of hand size and appearance across humans is not treated in this work.

1.4 Thesis Outline

The main claim of this thesis is that careful design and implementation of the steps of a model-based approach can lead to robust, full DoF hand tracking systems that perform close to real-time achieving accuracy in the order of millimeters. The resulting, highly parallelizable computation pipeline, based on an appropriate approximation of the hand shape, allows the efficient generation of image features and the exploitation of powerful evolutionary optimization algorithms. Careful design allows to tap into the power of modern hardware and more specifically the parallel computation platform of modern GPUs. The underlying computational framework, within which the presented methods were implemented, is presented in detail in [15].

The rest of this thesis is organized as follows: an outline of the relevant literature is presented in Chapter 2. The problem formulation of the proposed methodology is described in Chapter 3 along with necessary tools. In Chapter 4, specific instances built using this general approach are presented. Chapter 5 concludes the thesis.

Chapter 2

Literature Overview

The recovery of the full 3D configuration of articulated objects such as human bodies and human hands presents a lot of challenges. Several approaches have been proposed that address various aspects of the problem such as its dimensionality, the incomplete and/or ambiguous observations due to scene clutter, its computational requirements, etc. Moeslund et al. [7] provide a review of research to the general problem of visual human motion capture and analysis. A review that is specific to the problem of human hand motion estimation is provided in [2].

Before the overview of hand tracking methods, a few issues relevant to it are discussed. Namely, the anatomy and modeling of the human hand as well as optimization methods and more specifically black-box optimization are briefly discussed.

2.1 Human Hand Modeling

Hand pose estimation methods use representations of certain aspects of the human hand, such as its appearance and kinematics. In the following sections we first provide a brief overview of the human hand anatomy, followed by commonly adopted models of hand kinematics and appearance. The problems of modelling natural hand motions as well as the fitting of a generic kinematics model to the metrics of a specific user are also posed in this context. In this thesis these two problems are not systematically tackled.

2.1.1 Human Hand Anatomy

To build human hand models, knowledge regarding the human hand anatomy is required. Based on this knowledge, standard tools for kinematics modeling can be applied. Natural human hand motion can be modeled using this knowledge of hand anatomy, as well as statistics of natural motions. Generating the respective appearance induced by the resulting kinematics model is relevant to the area of computer graphics.

As described in [2], the human hand is formed of 27 bones (8 bones for the wrist and 19 bones for the palm and fingers), shown in Figure 2.1(a). Each of the fingers consists of three bones. Four bones form the palm, each one of them corresponding to one of the four fingers except the thumb which is directly attached to the wrist. The names of the bones are descriptive of their position: wrist parts are called carpals, palm parts are called metacarpals and finger parts are called phalanges. Phalanges are further distinguished in

proximal, middle and distal ones in the order from the finger base to the fingertip. Three joint types (shown in Figure 2.1(b)) are distinguished: carpometacarpal (CMC, joining wrist bones to bones of the palm), metacarpophalangeal (MCP, joining palm bones to finger bones) and interphalangeal (IP, joining parts of the finger). The CMC joint of the thumb is also called trapeziometacarpal (TM) and is the most complex one to analyze and model.

2.1.2 Hand Kinematics Modeling

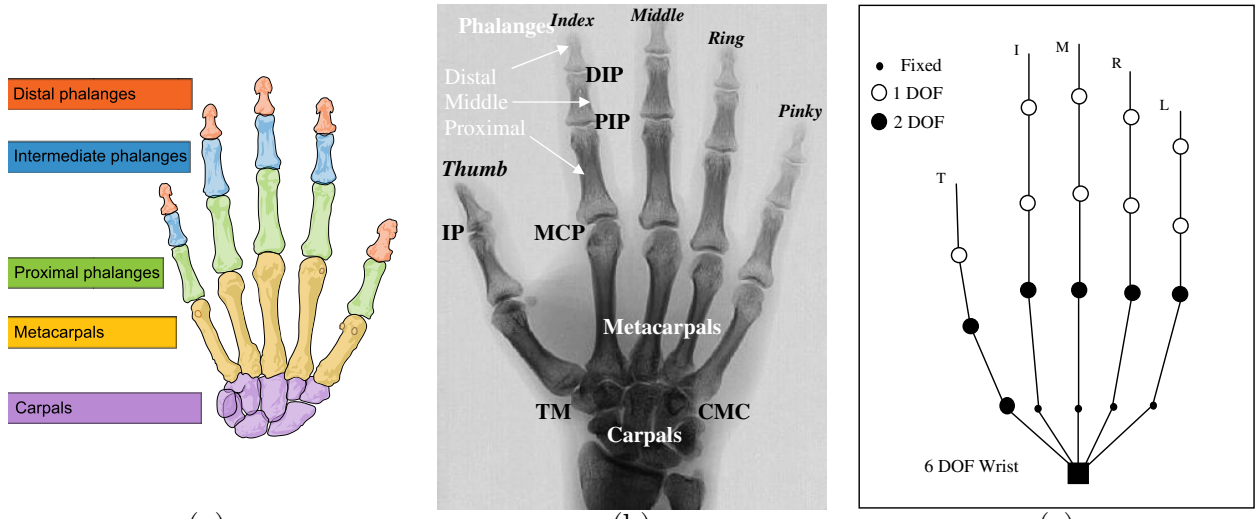
With respect to the kinematics modeling of the joints, all the IP joints can be accurately modeled as single DOF rotational joints, and all five MCP joints as saddle joints, i.e. joints with two rotational degrees of freedom. The CMC of the index and the middle finger are almost rigidly joined, thus forming the larger part of the palm. The CMC of the ring and pinky finger have a small capability of motion, however this DOF is usually discarded resulting in a rigid palm. As already stated, the CMC of the thumb is a special case. It has been shown [14] that this joint has two non-perpendicular and non-intersecting axes of rotation. However, since this modeling is difficult to handle and fine-tune in practice, the model of a saddle joint is usually adopted for this case as well. The described kinematics model is graphically depicted in Figure 2.1(c).

For the purpose of hand tracking, the adopted kinematic model essentially acts as a set of constraints. The enforcement of the adopted kinematic model is usually based on standard robotics tools. Specifically, the position and orientation of the palm are deemed free (i.e. 6 DOF), and the DOFs of finger parts are encoded as joint angles, measured using a predefined convention. It is worth noting that this type of model is common but not the only one employed. For an exception, the reader is referred to [16]. In that work, the kinematic model is imposed by a prior distribution regarding the relative position of finger parts (this prior is derived from the adopted kinematics), so the kinematics is not a hard constraint, but rather softly imposed.

2.1.3 Hand Appearance Modeling

Visual methods for hand pose recovery extract information from the observed image in order to compute the estimated position, orientation and finger articulation of the hand. This is performed by extracting relevant features from the input image. Comparable synthetic features must then be available from the employed hand model in different candidate hand poses, allowing the identification of the most suiting hand pose. Therefore, the appearance of the modeled hand must be synthesized, or specifically, at least the appearance aspects needed to extract the relevant features. This gives rise to different shape and appearance approximations based on the computational constraints of each method, and the types of employed features.

As a general trend, the modeling of the hand appearance is increasingly improved over time (see Figure 2.2). A first approximation was to assume that each rigid part can be modeled by a cylinder, employed by [3] (Figure 2.2(a)). The only type of feature used in that work is image edges, and hence the approximation is sufficient. On the other end of the spectrum, the single employed "feature" in [5] is the full appearance of the hand, and hence a photorealistic hand model is required (Figure 2.2(c)). Albrecht et al.



(a) (b) (c)

Figure 2.1: Anatomy and kinematic modeling of the human hand retrieved from [1] and [2]. (a) Anatomy of the human hand with annotated bones (b) and joints. (c) A commonly adopted model of hand kinematics. The wrist is depicted with the filled square, and serves as the global frame of reference for the hand (6 DOF). The legend at the top-left of the figure shows symbols for joints of zero (fixed link), one and two degrees of freedom (see text for more details).

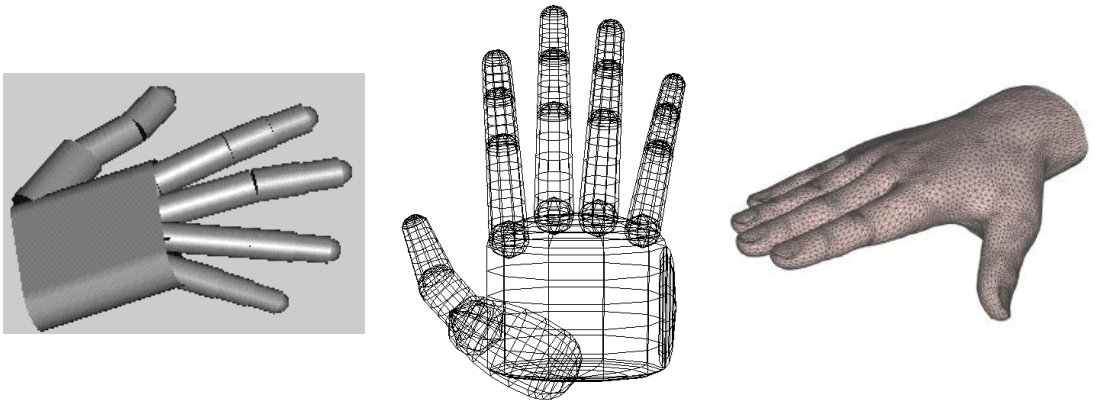
in [17] study the anatomic properties of the hand in order to build a realistic hand model for use in computer graphics. They present a detailed hand model capable of realistic hand motion based on an elaborate skeleton, muscles and skinning (a computer graphics technique for the computation of deformations of non-rigid objects).

2.2 Optimization

Optimization is defined as the task of finding the best fitting element in a set S, called the *search space*, given a fitness criterion f, termed the *objective function*. The most usual scenario involves a subset of $\mathbb{R}^n$ as the search space S, with the objective function f mapping elements of the search space to $\mathbb{R}$, denoted as $f : S \subseteq \mathbb{R}^n \to \mathbb{R}$. For this scenario, the optimization problem is called a minimization if the optimum value is the lowest one: for the optimum x_0, $\forall x \in S : f(x) \geq f(x_0)$. In the opposite case, where $\forall x \in S : f(x) \leq f(x_0)$, the problem is called a maximization.

A solution $x_0 \in S$ is called a local minimum if there is a radius $r > 0$ such that $\forall x \in S$ with $|x - x_0| < r$, it holds that $f(x) \geq f(x_0)$. Furthermore, the element x_0 with the lowest objective function value over all elements in S is called the global minimum: $\forall x \in S$ it holds that $f(x) \geq f(x_0)$. The solution to a minimization problem is defined to be the global minimum.

Many tasks in computer vision are formulated as optimization problems [18–21]. Optimization is a powerful framework that allows the transformation of hard estimation problems to potentially easier, evaluation ones. More specifically, in many parameter estimation problems that occur in computer vision and other disciplines, instead of at-



(a) (b) (c)

Figure 2.2: Employed hand appearance models. (a) In [3], fifteen cylinders compose the hand fingers (the palm and the spheres at the fingertips are not used for feature extraction). (b) In [4], the shape of the hand is built from 39 truncated quadrics. (c) A very accurate model is employed in [5]. Images (a) and (b) are retrieved from the respective publications and (c) from [6].

tacking the parameter estimation problem directly, the optimization framework allows to solve the easier problem of scoring candidate solutions by defining an appropriate objective function. After the implementation of the objective function, a suitable optimization method can be used to solve the original parameter estimation problem.

There has been a significant amount of work dedicated to developing strategies and techniques that efficiently solve optimization problems. For analytical, differentiable objective functions the problem has been extensively studied [22]. For these cases, the key property to exploit is the observation that the local optimum is necessarily located in a position where the gradient vanishes. This leads to the main idea of gradient descent: by iteratively following the direction opposite to the gradient it can be shown that one asymptotically approaches a local minimum.

However, in many real world problems it is difficult or even impossible to employ differentiable objective functions. Furthermore, even for differentiable functions it may be very difficult to compute the gradient. For this reason, recently there has been an increasing amount of work regarding so called *black-box* optimizations methods, i.e. methods that need only the domain and the objective function of the problem [23, 24].

Many black-box optimization methods fall under the general approach of *evolutionary computing*. Methods that follow the approach of evolutionary computing are called *evolutionary optimization*. Evolutionary optimization is widely regarded as a powerful strategy to optimize objective functions with significant amounts of noise, discontinuities and even uncertain values [25, 26]. As identified in [25], four key ideas govern the design of evolutionary computing methods:

- one or more populations of individuals competing for limited resources,
- the notion of dynamically changing populations due to the birth and death of individuals,
- a concept of fitness, reflecting the ability of individuals to survive and reproduce,

- a concept of variational inheritance: offspring closely resemble their parents, but are not identical.

In the hand pose estimation and tracking problem, the parameter estimation problem can be transformed into the relatively easier one of evaluating the fitness of a vector of parameters. More specifically, the parameter estimation problem is formulated: given a set of visual observations of a hand, estimate a set of hand pose parameters that match these observations. On the other hand, the parameter evaluation problem is formulated as: given a set of visual observation of the hand along with a human hand pose parameterization, quantify how well the given parameterization matches the visual observations. A solution for the second problem can serve as an objective function in an optimization problem that essentially solves the first one: optimize over the hand pose configuration space for a given visual input, thus effectively solving the direct, parameter estimation problem.

This formulation leads to a scoring function that quantifies the discrepancy between a hypothesized hand pose and the available visual observations of it, the properties of which must be studied for suitability in an optimization problem. In practice this function will be multimodal, difficult to differentiate or even discontinuous and hence non-differentiable. Specifically, if a graphics pipeline is involved as in the case of this thesis, the inherently discrete nature of the rendered images imposes a significant difficulty in estimating the derivative of the objective function. Because of these reasons, it is desirable to avoid the necessity to compute the derivative of the objective function, thus excluding gradient-based optimization approaches. Black-box optimization techniques on the other hand solely rely on the availability of the objective function, and can also potentially overcome the multiple local minima that occur.

In this thesis, hand pose estimation and tracking is formulated as an optimization problem as described above. All the objective functions that are employed for the various hand tracking scenarios have the aforementioned properties that inhibit the use of gradient-based optimizers. For this reason, all the adopted optimization techniques are black-box. More specifically, variants of Particle Swarm Optimization (PSO) [27] are employed in most cases. Additionally, a novel evolutionary optimization algorithm is proposed and compared to PSO. This evolutionary technique 3.5.2 is specifically designed for the problem of hand tracking. The algorithm is tailored to the local optimization problem that occurs under the assumption that consecutive observations of the scene depict the human hand in states that are close to one another in the hand configuration space.

2.3 Categorizations of Hand Tracking Approaches

Erol et al. in [2] categorize methods as partial or full pose estimation ones, depending on the level of detail they provide regarding the observed hand. Another categorization within the full pose estimation class identifies appearance-based and model-based methods. Appearance-based methods estimate hand configurations by establishing a direct mapping of image features to the hand configuration space [28–41]. Model-based approaches employ a 2D or a 3D hand model. In the case of 3D hand models, the hand pose is estimated by matching the projection of the model to the observed image fea-

tures. The task is then formulated as a search problem in a high dimensional configuration space, which typically induces a high computational cost [3–5, 16, 42–45]. Most methods for hand pose estimation can be characterized as either appearance-based or model-based ones. Few methods fall between these two categories by adopting elements of both. For example, Qian et al. in [45] propose a model-based method which uses a set of appropriately placed spheres to model the volume of the hand. They propose an algorithm that combines Particle Swarm Optimization and Iterative Closest Point to optimize an appropriately formulated objective function. Although model-based, the authors propose fingertip detection to efficiently search for candidate poses, thus borrowing elements from appearance-based approaches.

2.3.1 Appearance-based versus Model-based Methods

Appearance-based Methods

Appearance-based methods estimate hand configurations directly from images using a pre-computed mapping from the image feature space to the hand configuration space.

A common strategy to the problem of hand pose estimation is to discretize the space H of all possible hand configurations. In this approach, a reference database D is built out of selected hand poses $\{h_1, h_2, \ldots\} \subseteq H$, along with their corresponding appearance. Each entry in the database D is a pair of a hand configuration h and an image I of the hand in this configuration: $D = \{(h_1, I_1), (h_2, I_2), \ldots\}$. The problem can be reduced in this way to one of image retrieval: the database image I_n that best matches the observed one is paired with the hand configuration h_n which comprises the response of the method. Thus, by discretizing the target domain, the original regression problem is essentially reduced to a classification one. Alternatively, this database can be used as a training set for a regression learning technique, or more generally for any method capable of interpolating between provided samples, where input is the set of images (processed in order to extract features of interest) and output the respective hand poses. Methods that pre-process a large set of examples as just outlined in order to produce a regressor or a classifier are called *appearance-based*.

Appearance-based methods attempt to solve a difficult problem since the mapping from images to hand poses is highly nonlinear due to the variation of hand appearances under different views. Further difficulties are posed by the requirement of collecting large training data sets and the requirement for accuracy of pose estimation. On the positive side, appearance based methods are usually fast at runtime, require only a single camera and have been successfully employed for gesture recognition and grasp categorization tasks [33, 34]. A particular property of appearance-based methods is that they can be easily specialized to specific hand motions (e.g. specific types of object grasps or a set of selected gestures): the provided examples need only be sampled from the poses of interest. This is simultaneously a strong and weak point of this approach: such methods are easily applied to specialized cases, however in order to cover the full set of possible hand poses, a very large dataset must be employed. As a bottom line, the distinguishing feature of appearance-based methods is that the required mapping is computed/generated during the training/engineering phase and remains fixed thereafter. After this stage, this computed mapping is applied to the input images.

Model-based Methods

Model-based methods follow a different approach. Methods of this type are employing a hand model that is used during the estimation process to compute image features which are compared to respective features extracted from the observed image. More specifically, features such as edges, skin color, or even full appearance estimation are constructed using the employed hand model. These features are compared to respective features extracted from the observed image(s). A quantification of this comparison for varying poses of the hand model serves as an objective function to an optimization routine. Thus, the original problem is effectively reduced to an optimization one with search space the parameterization of the hand model.

The formulation as a search problem in a high dimensional configuration space undermines the computational performance of model-based methods. On the other hand, these methods, unlike appearance-based ones, can be easily adapted to different situations such as varying lighting conditions or object manipulation. The research areas of model-based methods include the construction of efficient and realistic 3D hand models, the dimensionality reduction of configuration space and the development of fast and reliable tracking algorithms to estimate the hand posture.

2.3.2 Disjoint and Joint Evidence Methods

One more categorization is based on how partial evidence regarding the individual rigid parts of the articulated object contributes to the final solution [46]. *Disjoint evidence methods* [3,16,42,47] consider individual parts in isolation prior to evaluating them against observations. *Joint evidence methods* [4, 5, 28, 29, 31, 35, 36, 46, 48, 49] consider all parts in the context of complete articulated object hypotheses. By construction, joint-evidence methods treat part interactions effortlessly, but their computational requirements are rather high. Disjoint evidence methods usually have lower computational requirements than joint-evidence ones, but need to explicitly handle part interactions such as collisions and occlusions. Since such issues are pronounced in the problem of tracking a hand in interaction with objects and in the problem of two hands tracking, joint evidence methods are more suitable than disjoint evidence methods for such tasks.

Another advantage of joint-evidence methods over disjoint-evidence ones is the allocation of search resources: since the articulation configuration of the object of interest is evaluated in whole, instead of partially, configurations that are implausible because of physics, or other a-priori known constraints regarding the scene. Therefore the joint-evidence methods typically search in spaces of higher dimensionality that disjoint-evidence ones, however the search resources can be better allocated.

2.3.3 Comparison of Approaches

Appearance-based and model-based methods exhibit different strengths and weaknesses. Comparing them is not straightforward, since there are a lot of aspects and parameters to consider. As already mentioned, the computational complexity of appearance-based methods is usually lower than that of model-based ones. Nevertheless, the gap of computational efficiency between the two approaches is narrowing because of algorithmic

improvements and the advance of hardware. Scalability and versatility favor model-based methods, since the design phase for these methods is usually less demanding. As a comparative example, the per-person adjustment of hand metrics is in most cases a straightforward task for model-based methods whereas for appearance-based ones, either a new mapping must be computed from scratch for each set of measurements, or the training must include samples of differently sized hands. Another aspect that favors model-based methods is the expected accuracy. In this respect, the appearance-based methods have essentially a fixed accuracy which is fully determined from the computed mapping. On the other hand, model-based methods are limited only by computational time, since the expected accuracy of the result is in theory directly related to the amount of performed computations.

2.3.4 Categorization of the Presented Work

In terms of the previously described classifications, this thesis presents model-based, joint-evidence methods for tracking the full articulation of hands. The common underlying methodology is versatile enough to handle different observation modes as well as varying scenarios regarding the observed scene. More specifically, observations from multicamera systems and RGB-D sensors can be used. The treated scenarios range from a single hand in isolation, to a hand manipulating an object, to two hands in strong interaction with each other.

The appearance of the human hand is modeled in this work as a set of appropriately transformed cylinders and spheres. This achieves a good approximation of the hand shape that yields a high degree of computational parallelism (by reusing the geometric primitives of a cylinder and a sphere) enabling fast synthesis of features. The kinematics are modeled as a simple kinematic tree similar to 2.2(c) and are strongly enforced, enabling the compact representation of the finger positions by only 20 Degrees of Freedom. For more details see Section 3.2.

Chapter 3

Problem Formulation and Tools

This Chapter provides an overview of the problem formulation, the visual cues and the principles along which they are utilized, as well as necessary tools required for the implementation of the actual methods presented in Chapter 4.

The problem of 3D hand tracking is formulated as a sequence of optimization problems. The input to the system is generally assumed to be multi-view, a set of fully calibrated and synchronized image streams that capture the hand motion from different viewpoints. This trivially includes the case where the input consists of only a single view of the scene. The image streams can either consist of regular color images, or alternatively they may include depth information, captured from RGB-D sensors such as the Kinect [12]. Each multi-frame, defined as a set of simultaneously captured images, is used to instantiate an optimization problem (Section 3.1). The search space of this problem is the hand configuration space (Section 2.1.2 and 3.2). The objective function of the problem quantifies the discrepancy between a hypothesized hand pose and the observed image(s), including also self-consistency terms (Section 3.4). Appropriate visual cues are employed in each scenario, permitting the efficient computation of the objective function, while retaining the essential information of the input stream(s) (Section 3.3). For each such optimization problem, the best scoring hand configuration that is found with a suitable optimization algorithm (Section 3.5) is deemed the solution. This solution is then used to initialize the search for the next frame (Section 3.6).

3.1 Formulation

The model-based, estimation-by-synthesis approach for hand tracking mandates that the task is formulated as an optimization problem. Under this formulation, the unknown parameters of the observed hand pose that are to be estimated become parameters of the objective function to be optimized. The objective function must be designed so that its optimum coincides with the sought-after solution. For our specific problem, the objective function quantifies the discrepancy between a candidate hand pose and the observations regarding the scene. This is achieved in practice by comparing visual cues extracted from the observation and computed from candidate hand poses. Apart from attaining the global optimum at the appropriate hand pose, the objective function must be carefully designed to aid the optimization process. The behavior of the objective function around the optimum is crucial for the accuracy of the estimated solution, while the behavior

further away determines the robustness of the resulting method.

The formulation as an optimization problem allows to solve a difficult parameter estimation problem by converting it to one of parameter scoring/validation. In practice, for the problem at hand, this means that instead of having to estimate the pose parameters directly from the visual input, we are in a position to solve the problem only by quantifying how well a candidate pose fits the observations. The employed optimization algorithm handles the task of finding the optimum of this objective function, thus simultaneously solving the original problem of pose estimation.

The design choices regarding components of the presented methods affect the overall computational performance of the resulting system. Such components include the employed visual cues, the optimization algorithms as well as the form of the objective function. The selection of the miscellaneous components that comprise the computational pipeline visual cues, along with the way the comparison between hypothesis and observation is performed, determines the computational cost of the objective function. Apart from the direct effect on computational cost, the selected cues can indirectly impact the overall runtime by the degree to which the required computations are parallelizable. The implementation of this tracking framework is described in detail and as a unified approach to scene understanding in [15], including issues on parallelization of the involved computations.

3.2 Hand Model

We base the employed hand appearance and kinematics model on that proposed in [17] (see also Section 2.1.2). The global position of the hand requires 3 Degrees of Freedom (DoFs) that specify the center of the palm in the global frame of reference. The rotation of the palm is encoded with 4 more DoFs using the quaternion representation of 3D rotations. Rotations in 3D space have 3 degrees of freedom, and hence the quaternion representation is redundant. Nevertheless, we employ this redundant representation instead of others that require only 3 parameters, because this representation has a smooth mapping to 3D rotations that does not suffer from discontinuities such as the problem of gimbal lock [50]. In sum, the global position and orientation of the palm are encoded using a total of 7 parameters.

For the 5 fingers of the hand, 20 DoFs are used to fully determined the articulation pose, specifying the state of finger joints in the form of angles. Each of the fingers requires 4 DoFs: 2 degrees of freedom for the base, determining flexion-extension and abduction-adduction, and 2 more DoFs corresponding to the two remaining joints of the finger. The actual kinematics of the human thumb are more complex than this simple model (see [14]), however in practice the thumb motion is adequately approximated with a model similar to the other fingers, with the same number of DoFs. This parameterization yields a total of 27 parameters that describe the position, orientation and finger articulation of a human hand in 3D. Throughout this thesis, such a 27-dimensional vector will be denoted as h, an element of the hand parameterization space H: $h \in H$. Written out in detail $h = (x, y, z, q_w, q_x, q_y, q_z, \theta_{thumb}, \theta_{index}, \theta_{middle}, \theta_{ring}, \theta_{pinky})$ where for each finger, θ_{finger} groups the four angles that parameterize it: $\theta_{finger} = (\theta_{1,base}, \theta_{2,base}, \theta_{mid\,joint}, \theta_{tip\,joint})$.

Visually, the adopted kinematics model is depicted in Figure 3.1. The figure is adapted

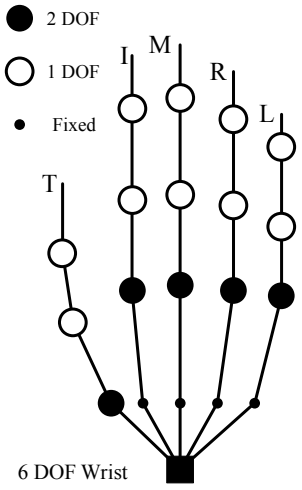


Figure 3.1: Illustration of the adopted model of hand kinematics. Adapted from [2]. As described in the text, the thumb kinematics are approximated using only 4 DoFs, similar to the other four fingers.

from [2], showing the Degrees of Freedom used throughout this work.

In order to reduce the complexity of the hand kinematics, rules derived by biomechanics studies [2] have been proposed. One such common rule links the angle of the proximal and distal interphalangeal joint: $\theta_{tip\,joint} = \frac{2}{3}\theta_{mid\,joint}$, where $\theta_{mid\,joint}$ and $\theta_{tip\,joint}$ respectively denote the angle of the proximal and distal interphalangeal joint. This rule is a good approximation when the fingers move freely, however it breaks when the fingers apply force, e.g. when grasping an object or even pressing the index against the thumb. As described in the introduction, we are interested in scenarios where the hands interact with objects, therefore we do not adopt this simplifying rule. Even more complex static and dynamic constraints of the human hand motion have been proposed and used throughout the literature (see [2]), however they are not adopted in this work. The rationale behind this choice is to firstly aim to solve the fully unconstrained problem, and then add appropriate constraints where necessary. Enforcing such constraints is not always an easy task, however the adopted model-based approach for hand tracking is highly extensible, leaving more room for modifications than appearance-based approaches.

Given specific values for these 27 parameters, the position and orientation of each part of the hand is fully determined. Our goal is to optimize these parameters, searching for the pose that best matches the observed visual input. Towards this end, under the adopted model-based approach, it is necessary to synthesize comparable visual cues that can then be compared with the observed ones, thus providing a score of the candidate pose and driving the optimization process.

The process of synthesizing visual cues uses a description of the hand structure in the form of 3D meshes. We build this description out of mesh representations of two basic geometric primitives, a cylinder and a sphere as illustrated in Figure 3.2. Appropriate homogeneous transformations of these two shapes form the 3D structure of the employed human hand model, similar to [4]. Each transformation performs two different tasks. First, it appropriately transforms primitives to more general quadrics and, second, it

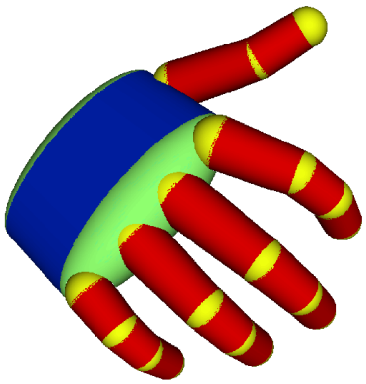


Figure 3.2: Hand model with colored parts. Each color denotes a different type of geometric primitive: blue for elliptic cylinders, green for ellipsoids, yellow for spheres and red for cones.

applies the required kinematics. Using the shape transformation matrix

$$T_s = \begin{pmatrix} e \cdot s_x & 0 & 0 & 0 \\ 0 & e \cdot s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 1-e & e \end{pmatrix}, \tag{3.1}$$

spheres can be transformed to ellipsoids and cylinders to elliptic cylinders or cones. The parameters of T_s, s_x, s_y and s_z are scaling factors along the respective axes. The parameter e is used only in the case of cones, representing the ratio of the small to the large radius of the cone before scaling. The convention we adopt is that the axis of symmetry of the cylinder is parallel to the z axis before this transformation. If not transforming to a cone, e is fixed to 1. The final homogeneous transformation T for each primitive (sphere or cylinder) is

$$T = T_k \cdot T_s, \tag{3.2}$$

where T_k is the rigid transformation matrix computed from the kinematics model.

The palm and each of the phalanges are considered rigid parts, appropriately placed in 3D space according to the kinematics described previously. The palm is built out of an ellipsoid cylinder and two ellipsoids serving as caps. Each joint has a sphere centered around it, serving the purpose of smoothing the discontinuity between consecutive rigid parts of the hand. Each of the phalanges consists of a cone, except for the base of the thumb which is modeled as an ellipsoid.

3.3 Feature Computation and Evaluation

The raw visual input is processed in order to remove known types of noise that may be present: regular color cameras exhibit varying amounts of white noise whereas depth sensors also exhibit Gaussian noise in the depth measurements, as well as more complex types of noise such as shot noise and bands of missing measurements around objects. Apart from noise reduction, the raw input is further processed to compute visual features that are then used for hypothesis evaluation. The computation involved with each feature

is composed of two different stages: input preprocessing and hypothesis comparison. The former includes both computing specific visual cues from the raw input, but additionally to perform preprocessing steps that allow to speed-up computations in the upcoming hypothesis comparison stage. Additional computations in the first stage can potentially speed up computations at later stages: the input preprocessing is performed once per input image, whereas the evaluation of the objective function is performed hundreds or thousands of times per frame. Therefore, careful choice of preprocessing computations can potentially speed up the overall process.

In order to synthesize features that are directly and efficiently comparable to the observed ones, the camera calibration information is required. More specifically, we resort to visual cues that can be represented as pixel maps, which are computed in direct correspondence to the input pixels. This is achieved by extracting the view frustum information contained in the camera calibration information, and simulating it in the graphics pipeline. Although possible, the radial distortion information is not taken into account in the graphics pipeline. Instead, an "undistort map" is computed during the camera calibration process. During acquisition, this map is applied in real time on the input observations, effectively removing the radial distortion.

The employed visual cues are: image edges, skin color, depth map and the visual hull. All features are computed on regular grids, the first three as pixel maps in one-to-one correspondence with the input and the visual hull as a voxel occupancy map. This choice facilitates comparison of candidate solutions with the observation, and allows for significant speed up in a GPU implementation. Additionally (and complementary) to the issue of computational efficiency, the choice of visual cues is followed by design choices regarding the quantification of the discrepancy between hypothesized and observed features. The average difference between the observed and hypothesized maps is such a quantification, however this is not always suitable as an objective function in the subsequent optimization process. Special preprocessing steps in the observed cues, as described in the previous paragraph, allows the use of more elaborate score fuctions without sacrificing computational efficiency.

3.3.1 Camera Calibration and Computer Graphics

For the problem of camera calibration we resort to the solution implemented in the OpenCV library [51]. The result of this calibration process is two matrices for each camera, one related to the intrinsic and one to the extrinsic parameters of the camera. The extrinsic matrix describes a rotation and translation, capturing the relation between the global coordinate frame and the camera one. The intrinsic matrix models the projection from the camera coordinate frame to the projection plane. For computer vision applications it suffices to be able to map points of the 3D space to the camera plane. This is modeled as a homogeneous matrix transformation that maps $4D$ vectors to $3D$ vectors, representing respectively $3D$ points to $2D$ points in homogeneous coordinates.

For computer graphics applications however it is also required to perform occlusion testing of rendered objects, allowing the ones closer to the camera to occlude others further away. This requirement mandates that depth information is not lost from the projection operation. Towards this end, computer graphics exchange the projection plane with a *projection cube*: the first two coordinates are the same as the projection plane,

and the last serves for occlusion testing. Because of the projective transformation, the last value cannot be directly the depth value, at least not without non-linear operations. Instead, a linear transformation of the inverse depth is used in practice. The boundaries of the rendered volume are defined by the faces of this cube. Apart from boundaries on the projection plane, computer graphics define the notions of *clipping planes*, near and far: objects outside this depth range are not rendered, thus hiding objects behind the virtual camera, avoiding projective distortions of objects very close to it, and hiding objects far away from it. The calibration information is used below to derive this augmented projective transformation.

The intrinsic parameters matrix is

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.3}$$

where f_x and f_y denote the focal length expressed in pixel units scaled by the aspect ratio of the pixels, and c_x, c_y denote the principal point in pixels. This matrix projects points in the camera coordinate system on the camera plane. This projection is denoted as

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} = \begin{bmatrix} f_x X + c_x Z \\ f_y Y + c_y Z \\ Z \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{3.4}$$

where $(X, Y, Z)^\intercal$ denotes a point in $3D$ whereas $(x, y, w)^\intercal$ denotes the respective $2D$ point using homogeneous coordinates.

As already outlined, the goal is to adapt the projection matrix so that it retains the necessary depth information. This can be achieved by augmenting it with an extra line and column. The additional values determine the positions of the near and far plane. All values are set to 0 except for the ones that regard the Z coordinate because the near-far plane clipping will be decided based only on the value of this coordinate. The projection on the right-hand side of Equation 3.4 can be equivalently written as

$$\begin{bmatrix} f_x X + c_x Z \\ f_y Y + c_y Z \\ aZ + b \\ Z \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & a & b \\ 0 & 0 & 1 & 0 \end{bmatrix} . \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \tag{3.5}$$

where a and b are parameters of the operation. This operation results in the same values as Equation 3.4, with the additional third value of $(aZ + b)/Z$. The vector on the left-hand-side of this operation corresponds to a point in $\mathbb{R}^3$ which, after dividing the value in the first 3 components by the value in the last one, yields:

$$(\frac{f_x X + c_x Z}{Z}, \frac{f_y Y + c_y Z}{Z}, \frac{aZ + b}{Z}, 1)^\intercal. \tag{3.6}$$

This last value is used for occlusion handling and clipping according to depth. The value $(aZ + b)/Z$ is a monotonic function of Z since the origin of the camera coordinate frame coincides with the center of projection. Therefore, the values of the parameters a and b can be fully determined by the two constraints regarding the near and far plane. Assuming

that these two clipping planes are respectively defined by $Z = z_n$ and $Z = z_f$ in the camera coordinate frame, and that the clipping cube defines as near and far respectively the values of 0 and 1, we can impose these constrains in a system of two equations:

$$\begin{aligned} \frac{az_n + b}{z_n} &= 0 \\ \frac{az_f + b}{z_f} &= 1. \end{aligned} \tag{3.7}$$

Solving this system for a and b yields

$$\begin{aligned} a &= \frac{z_f}{z_f - z_n} \\ b &= \frac{z_f z_n}{z_f - z_n}. \end{aligned} \tag{3.8}$$

In total, given a 3×3 intrinsic parameters matrix from a camera calibration process, and the values z_n, z_f defining the near and far plane, the projective transformation that appropriately transform the depths and clips them in the range $[0, 1]$ is:

$$\begin{bmatrix} f_x & 0 & c_x & 0 \\ 0 & f_y & c_y & 0 \\ 0 & 0 & \frac{z_f}{z_f - z_n} & \frac{z_f z_n}{z_f - z_n} \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.9}$$

3.3.2 Cropping the Region of Interest

The implementation of the presented work relies heavily in GPU computations, to synthesize and compare cues of candidate solutions. In [48] we used the calibration information as described previously in Section 3.3.1. From [46] and on, we added an extra step in the calibration information to crop both the observation and the synthesized cues in a region of interest (ROI). This achieves higher accuracy, faster runtime or both, by allocating computations to the parts of the scene that are relevant to the task. In practice, since in all the presented works the tackled problem involves tracking one or more visual entities, it is safe to assume that the area of interest for the current frame is close to the location of the tracked entities in the previous frame.

3.3.3 Image Edges

Image edges are a visual cue with numerous applications in computer vision [52, 53], including hand tracking [4, 16]. Edges constitute a useful visual cue for the task of hand pose estimation because they are robust to white noise and illumination conditions. Since the hand is uniformly colored and mostly texture-less, edges can be used to provide information regarding self-occluding poses: edges can be observed when a finger occludes another part of the hand such as the palm, providing useful information regarding the hand pose that is not contained in other visual cues such as the skin color map of the image. We use this visual cue in all the methods that work without depth information (works [48, 49, 54]), as an alternative means to acquire detailed information regarding the scene structure.

For the hand tracking pipeline, RGB input images are processed using the Canny edge detector [55]. The parameters of the method, namely the Gaussian blur variance and the threshold values for the Hysteresis thresholding process are experimentally determined. Indicative parameter values for the low and high threshold of Hysteresis thresholding are 0.25 and 0.35 respectively, yielding edge maps than only retain strong edges, thus filtering out background clutter. Gaussian blur should be tailored to each camera individually, based on the light conditions and the observed sensor noise. We experimented with high quality cameras (see Chapter 4) that did not require any blur to yield satisfactory results. The result is a binary map in pixelwise correspondence to the input image. By convention, the set values of the map signify edge pixels. For the computation of the objective function, the edge map of the input image must be compared to the hypothesized hand pose.

We are in a position to synthesize a comparable edge map for a candidate hypothesis by means of graphics rendering. The observed and the synthesized map are in pixel-wise correspondence since we are using the available calibration information, as described in Section 3.3.1. Given this correspondence, these two maps will be identical for the appropriate hand pose, except for inaccuracies in the modeling of the hand appearance.

The goal now is to quantify the discrepancy between these two maps, formulating a comparison function. This function will serve as a term in the objective function of an optimization process. As outlined in the introduction of this Chapter 3, the discrepancy between these two maps must be quantified in a way that aids the optimization process. With this goal in mind, the ideal behavior of this comparison would be to attain its global optimum when these maps coincide, and gradually fade away with increasing map differences. The sum of differences of these two maps does not meet this constraint since it exhibits a very abrupt spike close to the sought-after solution, and is practically flat everywhere else. Any optimization strategy would essentially be reduced to random sampling of the search space, and the desired solution would be found only by chance.

A scoring function that exhibits this desired behavior is the average of minimum distances between edge pixels of the maps. More specifically, for each edge pixel of the hypothesized edge map, the closest edge point of the observed map is located, and their distance is computed. The mean value of these distances is a function that exhibits behavior suitable for optimization. Since we can invest time for the preprocessing of the input image, as described in the introduction of this chapter, we apply the Distance Transform [56] to the observed edge map. By adopting this strategy, the potentially costly computation of minimum distances is reduced to simple lookups. During hypothesis evaluation, it suffices to mask the Distance Transform of the observed edges using the hypothesized edge map. The resulting distances are then summed and divided by the total number of hypothesized edge pixels, yielding the mean distance of hypothesized edge pixels to the closest observed ones. This last normalization is necessary because otherwise hypotheses that render few edge pixels will be favored, hindering the search process.

3.3.4 Skin Color

Skin color is another visual cue that encodes useful information for hand tracking. Since we are interested in markerless hand tracking, we assume that the observed hand is skin

colored, and therefore we can exploit this information, which can, in cases, be complementary to image edges. For example, such a case can arise when background edges appear close to the observed hand. In such a situation, the additional information from the skin color segmentation can aid the disambiguation of hand edges and unrelated ones. We put this visual cue to use in all the presented methods of Chapter 3.

To detect skin colored regions in the input images, we resort to color segmentation in the YUV color space. Specifically, we employ the skin color detection module of the methodology for skin colored blob tracking described in [57]. The parameters of this method are experimentally determined. The result of this process is a binary map, again in pixelwise correspondence to the input image, and with the convention that set values signify detected skin color.

For the comparison of the input skin color map to a hypothesized one, the pixelwise correspondence of these maps is again exploited. In contrast to the case of edges (see Section 3.3.3), the behavior of the average difference between maps is suitable as an objective function. This is because the foreground pixels are comparable in number to the background ones, making the averaged difference an appropriate function for the purpose of optimization. Since we are interested only in the number of pixels that differ between hypothesis and observation, the absolute difference is employed in practice. An efficient way to compute the absolute difference between binary values is to compute their exclusive OR (XOR).

This approach was adopted in [48], whereas from the next work [46], an adapted version of F-measure was employed [58]. F-measure is a function that combines the precision and recall of a classification test in a single quantity. The intuition behind it is that both precision and recall are important when assessing a test. The harmonic mean of these two quantities combines them in a single, balanced value:

$$F = 2\frac{pr}{p+r} \tag{3.10}$$

where p and r respectively denote the precision and recall of the assessed test. For more details regarding the F-measure, the reader is referred to [58].

Inspired from this intuitive balancing between precision and recall, we adapted the F-measure for our case. Based on the F-measure, we formulated a function that combines the number of foreground and background pixels so that they equally contribute to the final score.

3.3.5 Depth Map

When available, the cue of depth map of the scene provides very useful information for the task at hand. This visual cue has been widely used in the relevant literature for articulated pose estimation and tracking, see for example [47] for human body pose and [42] for hand pose. Kinect [12], introduced in the end of 2010, marked a combination of inexpensiveness, ease of use and quality of the estimated depth map that was not available before. In [46, 59] we employ a Kinect sensor as the single source of observation, exploiting both available input modalities, namely the RGB stream and the synchronized depth maps. The RGB stream is processed to extract skin colored areas (Section 3.3.4) while the depth map is minimally preprocessed.

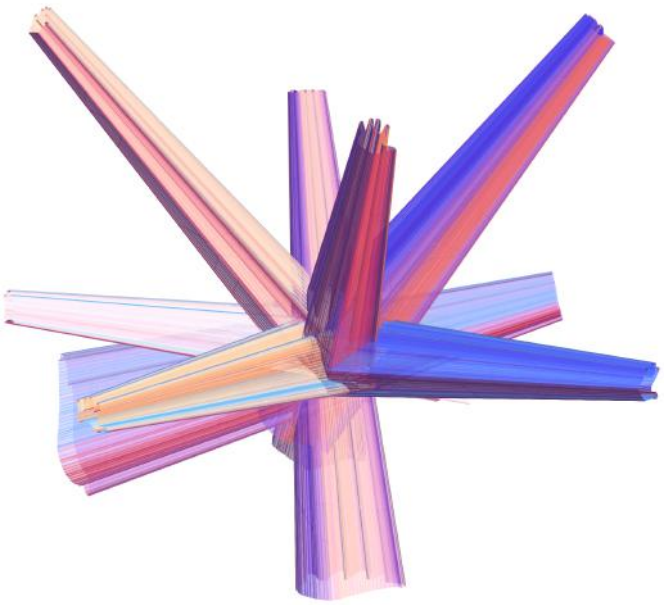


Figure 3.3: Illustration of the definition of the visual hull.

The depth maps produced by Kinect contain noise, that can be reduced with appropriate operations. Specifically, in [46] and [59] the input provided from Kinect is filtered with a median filter of window size 3 by 3. This operation reduces shot noise as well as Gaussian noise, both present in the input from Kinect. This filtered result is then masked with the estimated skin colored areas, yielding the depth map we use as observation.

Having a concrete model of the geometry of the observed scene, it is possible to synthesize the depth map corresponding to a candidate solution. This depth map is in pixel-wise correspondence, making the mean depth difference an appropriate quantification of the divergence between hypothesis and observation. Furthermore, this behaves well for the purpose of optimization, since it forms a smooth peak around the solution. In practice, the depth difference between each hypothesized and observed pixel is clamped, because after a certain threshold the magnitude of the difference provides no further information but instead hinders the search process. After experimentation the specific threshold was set to $4cm$ for all the experiments presented in Chapter 4.

3.3.6 Visual Hull

The visual cues described in the previous Sections are all arranged in $2D$ maps, in one-to-one correspondence with the input image(s). Even the depth map which provides $3D$ information regarding the observed scene is still mapped in a $2D$ grid, capturing essentially partial information regarding the scene. Therefore, all visual cues presented so far are significantly affected by the viewpoint. Aiming to alleviate this dependence to the selected viewpoint, in [54] we also experimented with the cue of visual hull [60].

A practical issue that must be addressed with all the previous visual cues for the case of multi-view input is the balancing between the matching scores for the same visual score across all different views. Specifically, a comparison between a hypothesized and an observed visual cue map yields one similarity/discrepancy score for each view, and

these scores must be somehow combined to yield an overall score. This issue is effortlessly addressed with the visual hull: the cue itself combines the visual input across all available views, and therefore the consequent comparison yields a single matching score, regardless of the number of available views of the scene.

In practice we approximate the foreground mask for each view using the skin color map. This is a valid approximation assuming that the observed hand is not covered or occluded by any other objects such as a glove or a manipulated object. Having a foreground mask for each available view, it is possible to compute the visual hull, defined as the intersection of the generalized frusta, visualized in Figure 3.3. This can be efficiently computed on a regular lattice using GPU acceleration [61].

The visual hull as computed using this method is represented as an occupancy map. The physical $3D$ space of the scene is partitioned in a regular lattice. Each vertex of the lattice, called a voxel, is assigned a binary value, representing occupancy. Qualitatively this representation is similar to the skin color map, since both are binary representations of occupied areas. Therefore, for the purposes of comparison and optimization the same principles apply. Specifically, the F-measure between a hypothesized and an observed occupancy volume is a good choice as a term of an objective function.

3.4 Modeling Non-Visual Information about the Scene

Knowledge regarding the observed scene can provide useful constraints, aiding the optimization process. The kinematics of the human hand offer such a set of constraints: the knowledge of the joint ranges can be directly provided as hard limits of the search space when using joint angles to encode hand poses, as in our adopted representation. Further information regarding the way human hands move can also be utilized. Kyriazis in his PhD thesis [15] presents a framework having as goal scene understanding. In that approach, knowledge regarding the observed scene is exploited in a unified, systematic matter. The key insight is that, in a physical world, the laws of physics must hold, imposing constraints on the observed entities.

For the present thesis, within the adopted model-based framework there are two main ways to impose constraints based on knowledge regarding human hand motion. The first is to generate hand poses that satisfy such constraints, and the second is to freely generate hand poses and afterwards test whether they meet the required constraints. For the first approach, a generative model of natural human hand motion is required. The search is performed on the parameter space of this model, and the resulting hand pose is guaranteed to satisfy the constraints, given that the generative model enforces them strictly. For the second approach, if the human hand motion knowledge is modeled simply as a set of constraints, penalty terms associated with these constraints can be added to the objective function. If a pose violates one or more constraints then a high penalty drives the search away from this area.

In this work we opt for this last approach because of its extensibility: constraints can be added or removed from the objective function independently of each other. This approach proves useful in the case of hand-object interaction since their interaction is harder to model. For hand-object interaction scenarios, all that is required with the adopted approach is to check for and appropriately penalize hand-object collisions in

candidate poses. When adding penalty terms to the objective function, care must be taken to avoid creating abrupt or discontinuous behavior: the penalty term should exhibit gradual transition from the permissible areas to the penalized ones.

3.5 Optimization

As introduced in Section 2.2, the objective functions that occur in the various explored scenarios are multimodal, difficult to differentiate and exhibit discontinuities. For these reasons we choose techniques that are tolerant to such issues, called black-box optimization algorithms.

In most scenarios we employ variants of the PSO algorithm to optimize the occurring objective functions. PSO is shown to efficiently tackle the high-dimensional, non-differentiable, multimodal objective functions that occur for the various scenarios. Another useful property of PSO that is exploited is the fact that the initial population does not have any requirements. This allows us to provide as an initial population, candidate poses close to the previous solution, thus exploiting the temporal continuity assumption.

Apart from PSO, a novel evolutionary optimization technique is also presented in this thesis, proposed in [62]. It is specifically tailored for the local search problem that occurs for consecutive frames when tracking a hand, assuming that the observed motion is not very abrupt. The power of evolutionary techniques is combined with Quasi-random sampling to efficiently search for the best matching hand pose.

Furthermore, as most evolutionary optimization techniques, both employed algorithms are amenable to parallelization: the computation of the objective function for all the individuals of the population can be performed in parallel. This parallelization is exploited using the GPU architecture, thus significantly speeding up the optimization process.

3.5.1 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is an optimization technique that was introduced by Kennedy et al [63]. It is an evolutionary algorithm since it incorporates concepts such as populations, generations and rules of evolution for the atoms of the population (particles). A population is essentially a set of points in the parameter space of the objective function to be optimized. The particles evolve in batches which are called generations according to a policy which emulates "social interaction".

Canonical PSO, the simplest of PSO variants, was preferred among other optimization techniques due to its simplicity and efficiency. More specifically, it depends on very few parameters, does not require extra information on the objective function (e.g., its derivatives) and requires a relatively low number of evaluations of the objective function [64]. Following the notation introduced in [65], every particle holds its current position, a candidate solution, in a vector x_t and its current velocity in a vector v_t. Moreover, each particle i stores in vector p_i the position at which it achieved, up to the current generation t, the best value of the objective function. Finally, the swarm as a whole, stores in vector p_g the best position encountered across all particles of the swarm. p_g is broadcasted to the entire swarm, so that every particle is aware of the global optimum. The update

equations that are applied in every generation t to reestimate each particle's velocity and position are

$$v_t = K(v_{t-1} + c_1 r_1 (p_i - x_{t-1}) + c_2 r_2 (p_g - x_{t-1})) \tag{3.11}$$

and

$$x_t = x_{t-1} + v_t, \tag{3.12}$$

where K is a constant *constriction factor* [66]. In Eqs. (3.11), c_1 is called the *cognitive component*, c_2 is termed the *social component* and r_1, r_2 are random samples of a uniform distribution in the range [0..1]. Finally, $c_1 + c_2 > 4$ must hold [66]. For all the experiments involving PSO, the values $c_1 = 2.8$, $c_2 = 1.3$ and $K = \frac{2}{\left|2-\psi-\sqrt{\psi^2-4\psi}\right|}$ with $\psi = c_1 + c_2$ were used.

Typically, the particles are initialized at random positions and their velocities are initialized to zero. Each dimension of the multidimensional parameter space is bounded in some range. If, during the position update, a velocity component forces the particle to move to a point outside the bounded search space, this component is set to the appropriate limit for this dimension. For the canonical PSO variant, this is the only constraint imposed on the particle positions and implicitly on their velocities.

Apart from this canonical variant, in some scenarios we choose another variant that better explores the parameter space. Specifically, for the scenarios using input from RGB-D sensors, the canonical variant successfully estimates the $6D$ global pose of the hand. However, the estimation of the 20 remaining parameters that are related to finger angles is not equally satisfactory. To overcome this problem and increase accuracy, we employ a PSO variant that performs randomization on the 20 dimensions corresponding to finger joint angles, similar to that suggested in [67].

Regarding the computational complexity of the algorithm, the update rules in Equations 3.11 and 3.12 are essentially simple linear operations. Therefore, usually the computational bottleneck in a PSO run is the computation of the objective function. The number of computations of the objective function is determined by the product of the number of particles multiplied by the number of generations. Unless a termination condition is met, terminating early the search, this is exactly the number of total objective function evaluations.

The accuracy of the estimated hand pose is usually dependent on this number of objective function computations, since the search continuously improves the estimated result. For this reason, in the experimental evaluation of the explored methods we vary these two parameters, namely the particle and generation count, assessing the resulting accuracy for each such configuration.

3.5.2 Evolutionary Optimization with Quasi-random Sampling

Sobol [68] introduced a low-discrepancy sequence of n samples x_i in the S-dimensional hypercube $[0..1]^S$ with the aim to approximate the integral

$$\int_{[0,1]^S} f(x)\,\mathrm{d}x \tag{3.13}$$

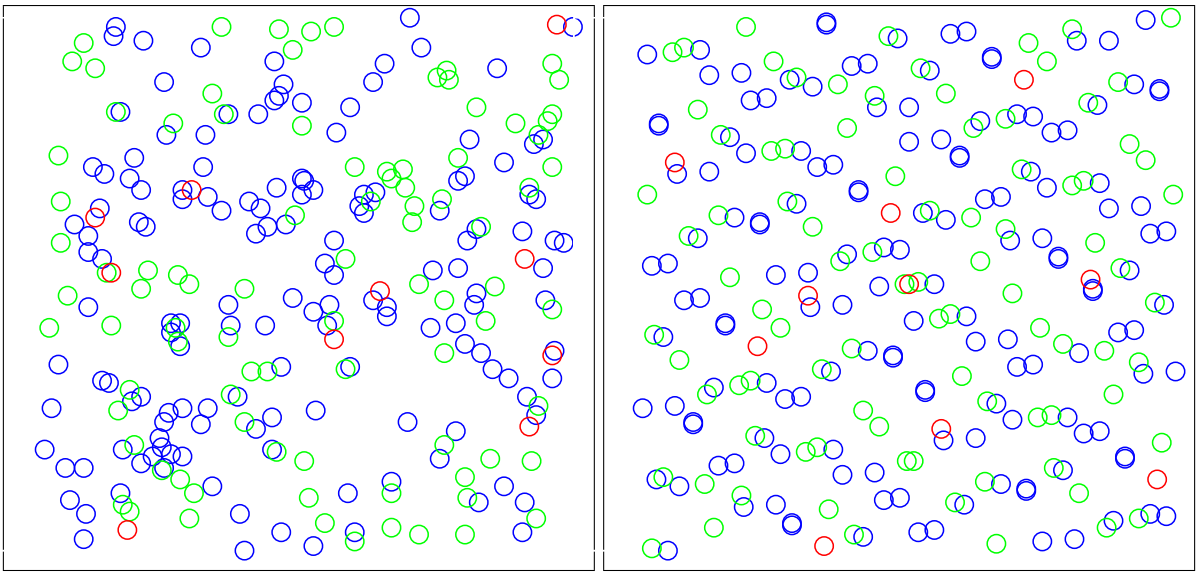


Figure 3.4: Left: 256 points on the $2D$ plane obtained from a pseudo-random number generator. Right: the first 256 samples of the Sobol sequence. Samples 1 to 10, 11 to 100 and 101 to 256 are in red, blue and green colors, respectively. The Sobol sequence covers the space more evenly. In our problem formulation, the Sobol sequence is used to form quasi-random hypotheses of hand configurations in the $27D$ (single hand) and $54D$ (two hands) configuration spaces. Example inspired from `http://en.wikipedia.org/wiki/Sobol_sequence`.

of an arbitrary function f over $[0,1]^S$ by the limit

$$\lim_{n\to\infty} \sum_{i=1}^{n} f(x_i) \tag{3.14}$$

with the fastest possible convergence. Loosely speaking, for small sample sizes, in the order of tens or hundreds of samples, the resulting coverage of the sampled space is more even, leaving smaller gaps than that of a set of points sampled from a uniform distribution. For a visual comparison in $2D$ see Figure 3.4. The comparison favors quasi-random sampling as the number of space dimensions S is increased. For a more detailed and formal presentation of low-discrepancy and Sobol sequences, the reader is referred to [69].

As in most evolutionary optimization algorithms, in our approach there is the notion of a population of candidate solutions or atoms. These atoms are essentially points in the search space, corresponding to candidate solutions. This population evolves in steps, called generations. A high-level outline of the proposed evolutionary Quasi-random search algorithm is presented in Algorithm 1.

The algorithm maintains the full history H of all the search space positions that have been explored so far, along with their corresponding fitness scores w, and runs for a fixed number G of generations. In each generation g, $0 \leq g \leq G-1$, a center position h_C is defined. For the first generation, h_C is set equal to the solution h_{t-1} sought for the previous frame. N atoms h_t^i are then defined around h_C based on the Sobol sequence. This is done so as to take advantage of the way quasi-random sampling can evenly sample

Algorithm 1 The proposed evolutionary Quasi-random search algorithm

Input: The solution h_{t-1} for the previous frame.
Output: The solution h_t for the current frame.

$H \leftarrow \emptyset$; $T \leftarrow \emptyset$;
$h_C \leftarrow h_{t-1}$;
for $g = 0 \dots G-1$ **do**
  // Define atoms h_t^i $(1 \leq i \leq N)$ around h_C (Eq.(3.15))
  $\{h_t^i\} \leftarrow SobolSequence(h_C, N, g)$;
  // compute E for atoms and store fitness
  $H \leftarrow H \cup \{h_t^i\}$;
  $w(h_t^i) = E(h_t^i)$;
  $T \leftarrow TopScoringAtoms(H, N_T)$;
  $h_C \leftarrow WeightedSum(T)$; (Eq.(3.16))
end for
$h_t \leftarrow TopScoringAtoms(H, 1)$
return (h_t);

high-dimensional spaces. Each parameter dimension has different units and range, so a vector s of scales is used to adjust the original range $[0, 1]$ of the Sobol sequence to the appropriate one. More specifically,

$$h_t^i = h_C + s \circ c^g \circ (2x_{r+i} - 1). \tag{3.15}$$

In Eq.(3.15), i iterates over the population count, "$\circ$" denotes the Hadamard or entry-wise product between vectors, x_n is the n-th sample of the Sobol sequence of appropriate dimensions, and r is a large random integer after which we draw samples from the Sobol sequence. c is a vector of contraction coefficients, with entries in the range (0..1]. Raising to g denotes entrywise power. The goal of this operation is to reduce the size of the search space around h_C as a function of the generation count.

All the identified atoms h_t^i are inserted in the history H. The objective function E is consequently evaluated for each of these atoms h_t^i, resulting in corresponding fitness scores $w(h_t^i) = E(h_t^i)$. Next, from the whole history H, the set T containing n_T atoms with the highest fitness scores is computed. A new center in the search space h_C is located as a weighted sum of these n_T atoms, as follows:

$$h_C = \frac{1}{\sum_{h \in T} q(w(h))} \sum_{h \in T} q(w(h)) \cdot h. \tag{3.16}$$

We chose $q(x) = \exp(ax)$ because a can be appropriately chosen to scale the weights so that there is a fixed ratio between the first and second best scoring atoms.

The above procedure is repeated for all G generations. After this computation is completed, the most fit atom among the whole history H is reported as the result h_t of the optimization process.

From a computational complexity point of view, the most expensive part of the algorithm is the evaluation of the objective function for a given atom, as in the case of PSO.

N such evaluations are performed in each generation, thus the product $N \cdot G$ determines the computational budget of the method. It should be noted that within each generation, the computations for each atom are independent of the computations for the other atoms. This inherent computational parallelism can be exploited to achieve very efficient implementations in GPU architectures.

Regarding stability, if each value in the vector of factors c is chosen to be lower than 1 then obviously the algorithm converges at some point in the search space.

3.6 Sequence of Optimizations

The explored scenarios involve tracking the poses of one or more entities. Under the temporal continuity assumption, poses for consecutive frames are close in the parameter space. In order to explore this assumption, we exploit the fact that the employed evolutionary algorithms can use an initial population as the starting point of the search. This initial population effectively determines the search area, since the motion of the particles in the search space is based on their previous positions. Exploiting this property, we use perturbations of the solution for the previous frame to initialize the search for the current one. The perturbations occur by adding Gaussian noise of appropriate variance to the previous solution. For the first frame we assume that an initialization pose is available.

3.7 Performance Evaluation

An issue pertinent to all the presented methods is that of performance evaluation. Apart from empirical assessment, e.g. using visualization tools, it is mandatory to have a consistent, systematic way to quantify the performance of a proposed methodology.

Towards this end, the first issue that is raised is the availability of ground truth. This is not trivial for the involved scenarios since automating the task essentially requires to solve the problem, while manual annotation is a tedious, time consuming and error-prone process. In all the presented methods we solve this problem using synthetic data. Specifically, after capturing a real-world sequence, we track the relevant entities in an offline process using very high computational budgets. This results in a sequence of poses that closely resembles the captured ones. The captured data are discarded, and this sequence becomes the reference that will later serve as the ground truth. Next, a graphics rendering process is utilized to synthesize image sequences that would be observed given this reference sequence of poses. During this step it is possible to directly synthesize the required image features. Furthermore, it is possible to assess the effect on noisy observations by adding artificial noise to the synthetic images.

Given that a sequence of images is available along with corresponding ground truth data, a systematic way to measure the accuracy of a method remains to be defined. A straightforward approach would be to directly compare the ground truth sequence with the recovered one, for example by computing the norm of the difference for each time instant. However this approach suffers from the problem of (depending on the adopted parameterization) evaluating within the same vector values that encode position along with ones that encode joint angles. This results in one type of values to dominate the results compared to the others. Even when only comparing angle values, for example

by computing the norm of the joint angle difference, the results are not representative: joint angles that are closer to the root of the kinematic chain have a greater effect in the final position than ones further away. Furthermore, the final value of the norm of differences in pose space is difficult to intuitively grasp, whereas it would be desirable for the quantitative evaluation of a track to have an intuitive interpretation.

Towards this end, we adopt a common (see [42]) approach: certain key-points are identified on the adopted hand model including the centers of all finger joints, the center of the palm as well as all the fingertips. Given a specific hand pose using the adopted parameterization, it is straightforward to compute the position of these key-points in $3D$ space. The error metric used to assess the performance of all the presented methods is based on this building block: given an estimated pose and a ground truth one, the average distance between corresponding key-points is computed. For a sequence of such poses we resort to averaging again, yielding an overall average distance Δ for the entire sequence.

A final issue the remains is the randomized nature of all the presented methodologies. Because of this fact, even for exactly the same input, the recovered track varies between runs, essentially adding noise to the estimation of the method's accuracy. For this reason we perform each experiment multiple times using exactly the same method configuration, and the average or median of the resulting values Δ_i is computed, yielding the final error metric $\mathcal{D}$: $\mathcal{D} = \overline{\Delta_i}$. The mean and the median have both their weaknesses: averaging is prone to outliers, whereas, since the distances are positive, the median yields a result that intuitively underestimates the actual discrepancy. Indicative numbers of experiment repetitions are in the range of 10 to 20 times.

Chapter 4

Hand Tracking Scenarios

This Chapter presents works that tackle real-world problems involving 3D hand tracking. The explored scenarios include a single hand in isolation (Sections 4.1, 4.2, 4.5 and 4.6), a hand manipulating an object (Section 4.3), as well as two hands in strong interaction (Section 4.4). The principles and techniques presented in Chapter 3 are combined and applied, yielding systems that efficiently and robustly solve real-world tracking problems, achieving real-time or interactive frame rates. The implementation of the presented systems is within the computational framework proposed in [15].

More specifically, this chapter describes six different works [46, 48, 49, 54, 59, 62], each of them proposing a method to tackle a specific hand tracking scenario. For each such scenario, appropriate visual cues are identified, an objective function is formulated, and an appropriate optimization algorithm is proposed to solve the problem. All the methods are experimentally assessed in both real-world and synthetic input.

Firstly, in the work presented in Section 4.1 (see [48]), a multi-camera system is used to acquire images of a hand moving freely in isolation, i.e. not interacting with any objects. An appropriate objective function is formulated using the visual cues of skin color and edges, and Particle Swarm Optimization (PSO) is employed to optimize it.

In the work presented in Section 4.2 (see [46]), the input is acquired using a single RGBD sensor, specifically a Kinect [12]. The scenario again involves a single hand moving freely in isolation. The method uses skin color and depth to segment the area of interest. This input is provided to an appropriately formulated objective function, which is optimized with a variant of PSO.

In the work presented in Section 4.3 (see [49]), a multi-camera system is again employed, observing a hand that manipulates an object of known geometric shape, but with unknown dimensions. The method uses the visual cues of skin color and edges, and employs canonical PSO to solve the resulting optimization problem.

A demanding scenario is handled in the work presented in Section 4.4 (see [59]), where a Kinect is used to acquire observations of two hands in strong interaction. The objective function uses information from the visual cues of skin color and depth, and a variant of PSO is used to optimize it.

The method that is presented in Section 4.5 (see [54]) explores the visual hull as a way to balance the input information across different views. This approach is compared to the similar one in [48] on multi-camera as well as stereo input.

Finally, in the work presented in Section 4.6 (see [62]), the evolutionary optimiza-

Table 4.1: Overview of the various hand tracking scenarios handled by works presented in this thesis.

Input type	Multi-camera	RGB-D
Hand in isolation	Sections 4.1 and 4.5	Sections 4.2 and 4.6
Hand-object interaction	Section 4.3	
Hand-hand interaction		Sections 4.4 and 4.6

Table 4.2: Computational performance measurements. Number of multiframes per second processed for a number of PSO generations and camera views. The cases of 16 and 128 particles per generation are presented.

Generations	2 views	4 views	8 views
10	7.69/2.48	4.22/1.26	2.14/0.63
15	7.09/1.91	3.65/0.97	1.85/0.49
20	**6.23**/1.55	3.19/0.79	1.62/0.39
25	5.53/1.31	2.85/0.67	1.44/0.33
30	5.00/1.13	2.59/0.57	1.30/0.29
35	4.55/1.00	2.34/0.50	1.18/0.25
40	4.18/0.89	2.15/0.45	1.09/0.23

tion technique presented in 3.5.2 is tested in the single hand and two hands in strong interaction scenarios. Meta-optimization is used to fine-tune the free parameters of the algorithm, and the performance is compared to that of the PSO variant in the respective works above.

Table 4.1 summarizes the different scenarios and input types used throughout the different works. The rows group the different scenarios and the columns separate the works by input type.

4.1 Hand Tracking using Multi-view Input

Our first work [48] on hand tracking uses input from a multi-camera system and proposes a method to solve the problem in the case of a hand in isolation. The utilized visual cues are skin color (Section 3.3.4) and edges (Section 3.3.3). The formulated objective function combines this visual information with a finger collision term. Canonical PSO is utilized to solve the resulting optimization problem. The resulting method is tested in real-world and synthetic image sequences for varying computational budgets of PSO, varying levels of image noise as well as different numbers of available views.

4.1.1 Method Outline

The input to the method is a sequence of simultaneously acquired images, captured with a fully calibrated multi-camera system. A multiframe M is defined to be a set of simultaneously captured images. As already introduced, the utilized visual cues are skin color and edges. The formulated objective function $E(h, M)$ measures the discrepancies

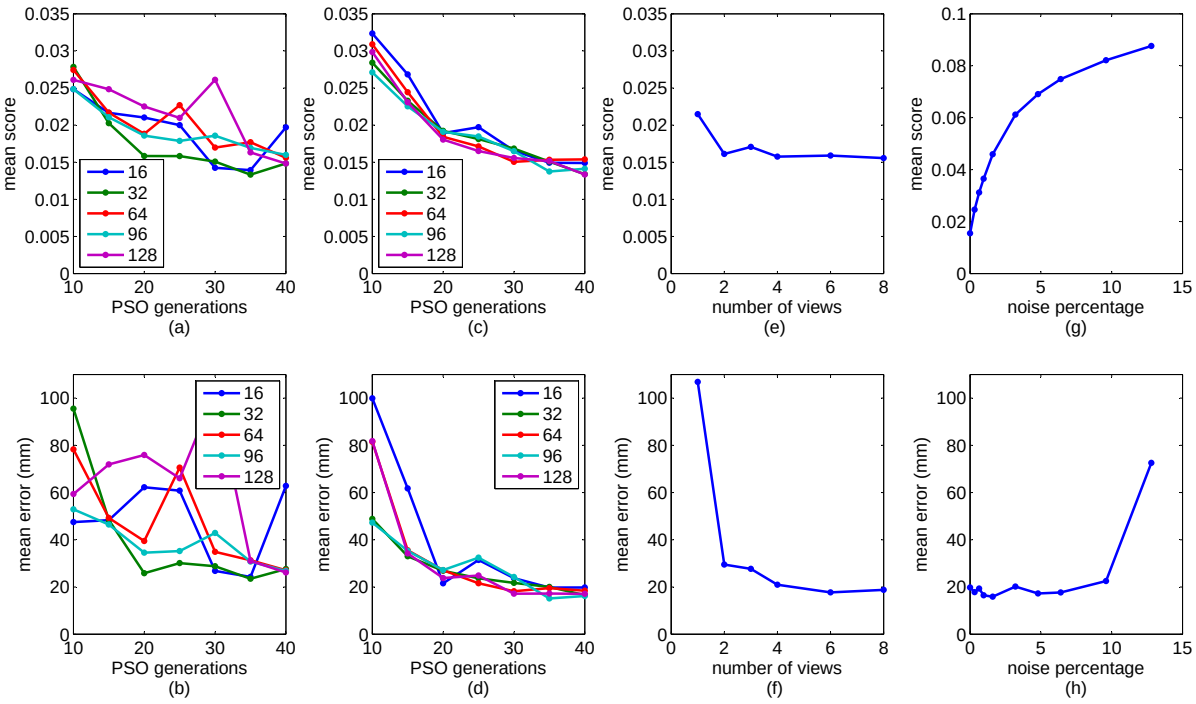


Figure 4.1: Performance of the presented method for different values of selected parameters. For plots of the top row, the vertical axis represents the mean score E. For plots of the bottom row, the vertical axis represents mean error in mm (see text for additional details). (a),(b): Varying values of PSO parameters particles and generations for 2 views. (c),(d): Same as (a),(b) but for 8 views. (e),(f): Increasing number of views. (g),(h): Increasing amounts of segmentation noise.

between skin and edge maps, computed in a multiframe and the skin and edge maps that are rendered for a given hand pose hypothesis h:

$$E(h, M) = \sum_{I \in M} D(I, h, \mathrm{C}(I)) + \lambda_k \cdot kc(h). \tag{4.1}$$

In Equation 4.1, h is the hand pose hypothesis using the adopted parameterization, M is the corresponding observation multiframe, I is an image in M, $\mathrm{C}(I)$ is the set of camera calibration parameters corresponding to image I and λ_k is a normalization factor. The function D of Equation 4.1 is defined as

$$D(I, h, c) = \frac{\sum o_s(I) \otimes r_s(h, c)}{\sum o_s(I) + \sum r_s(h, c) + \epsilon} + \lambda \frac{\sum o_d(I) \cdot r_e(h, c)}{\sum r_e(h, c) + \epsilon}, \tag{4.2}$$

where $o_s(I), o_d(I), r_s(h, c), r_e(h, c)$ respectively denote the observed skin color map, the distance transform of the observed edges, and the rendered skin color and edge maps generated by h. A small term ${}_2h\epsilon$ is added to the denominators of Equation 4.2 to avoid divisions by zero. The symbol $\otimes$ denotes the logical XOR (exclusive disjunction) operator. Finally, λ is a constant normalization factor. The sums are computed over entire feature maps.

The two terms of Equation 4.2 quantify the discrepancy between the observed and hypothesized skin color maps and edge maps. For the first term, the XOR operation in the numerator serves to count the pixels with differing values between observation and hypothesis (respectively o_s and r_s). This count is divided by the sum of set pixels in either the hypothesis or the observation, in order to balance the overall scores among observed skin maps with varying pixel counts across the available views I in the multiframe M. The numerator of the second term sums the point-wise multiplication between the distance transform of the observed edge map and the hypothesized edge map (respectively o_d and r_e). Since the convention for the edge maps is that a value of 0 indicates no edge whereas a value of 1 signifies the presence of an edge, the numerator is essentially the sum of the distance of each edge pixel in r_e to the closest observed edge pixel, as captured by the distance transform o_d.

The function kc adds a penalty to kinematically implausible hand configurations. Only adjacent finger inter-penetration is penalized. Therefore, kc is defined as

$$kc(h) = \sum_{p \in \mathrm{Q}} \begin{cases} -\phi(p) & \phi(p) < 0 \\ 0 & \phi(p) \geq 0 \end{cases}, \tag{4.3}$$

where Q denotes the three pairs of adjacent fingers, excluding the thumb, and ϕ denotes the difference between the abduction-adduction angles of those fingers. In all experiments the values of λ and λ_k were both set to 10.

As described in Chapter 3, the observed edge maps are further processed, and the resulting distance transform is the map that is actually involved in the objective function computations (second term of Equation 4.2). This strategy allows the computation of an objective function with a suitable behavior with only the overhead of computing the distance transform for each input edge map. On the other hand, the observed skin color is directly participating in the computations: the binary XOR operation is computed between the observed and hypothesized skin color maps (first term of Equation 4.2).

4.1.2 Experimental Evaluation

The quantitative and qualitative experimental validation of the presented method is performed based on both synthetic and real-world sequences of multiframes. A system of eight synchronized, fully calibrated RGB cameras was used to acquire real-world data.

The quantitative evaluation of the presented method is based on synthetic sequences of multiframes. The hand model presented in Section 3.2 is animated so as to perform motions as simple as waving and as complex as object grasping. A synthetic sequence of 360 poses of the moving hand occurs from this process. Each pose is observed by eight virtual cameras surrounding the hand.

The quantitative evaluation assesses the influence of several factors such as PSO parameters, number of available views (i.e., multiframe size) and segmentation noise, over the performance of the presented method. Figure 4.1 illustrates the obtained results.

For each multiframe of the sequence, the best scoring hand pose h_{best} using the specified parameter values was estimated. Figures 4.1(a), (c), (e) and (g) provide plots of the score $E(h_{best}, M)$ (averaged for all multiframes M) as a function of various experimental conditions. Similarly, Figures 4.1(b), (d), (f) and (h) illustrate the actual error in

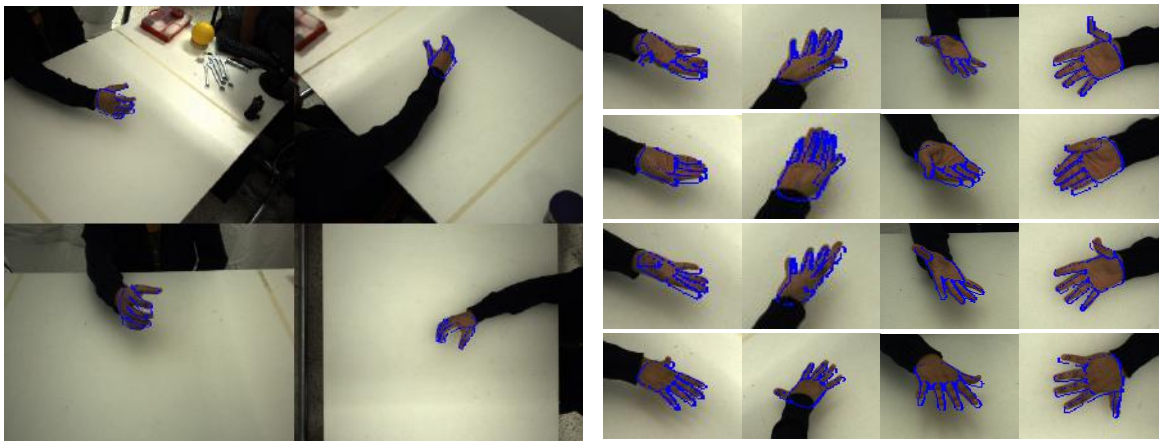


Figure 4.2: Sample frames from real-world experiments. Left: four views of a multiframe of a cylindrical grasp. Right: Zoom on hands in several multiframes. Quadruples of thumbnails are from the same multiframe; columns correspond to the same camera view.

3D hand pose recovery in millimeters, in the experimental conditions of Figures 4.1(a), (c), (e) and (g), respectively. This error was computed in accordance to the outline in Section 3.7 as follows. The five fingertips as well as the center of the palm are selected as reference points. For each such reference point, the Euclidean distance between its estimated position and its ground truth position is first calculated. These distances are averaged across all multiframes, resulting in a single error value $\mathcal{D}$ for the whole sequence.

Figures 4.1(a) and (b) show the behavior of the presented method as a function of the number of PSO generations and particles per generation. In this experiment, each multiframe consisted of 2 views with no noise contamination. It can be verified that varying the number of particles per generation does not affect considerably the error in 3D hand pose recovery. Thus, the number of generations appears to be more important than the number of particles per generation. Additionally, it can be verified that the accuracy gain for PSO parameterizations with more than 16 particles and more than 25 generations was insignificant. Figures 4.1(c), (d) are analogous to those of Figures 4.1(a),(b), except the fact that each multiframe consisted of 8 rather than 2 views. The error variance is even smaller in this case as a consequence of the increased number of views which imposes more constraints. The accuracy gain for PSO parameterizations with more than 16 particles and more than 25 generations is even less significant.

In order to assess the behavior of the method with respect to the number of available views of the scene, experiments with varying number of views were conducted. Figures 4.1(e) and (f) show the behavior of the presented method as a function of the cardinality of the input multiframes. For the experiments with less than 8 views, the views were empirically selected from the available ones so as to be as complementary as possible. More specifically, views with large baselines and viewing directions close to vertical were preferred. In these experiments, 128 PSO particles and 35 generations were used, and no segmentation noise was introduced in the rendered skin and edge maps. The obtained results (Figures 4.1(e) and (f)) show that the performance improvement from one view to two views is significant. Adding more views improves the results noticeably but not significantly.

In order to assess the tolerance of the method to different levels of segmentation errors, all the rendered silhouette and edge maps were artificially corrupted with different levels of noise. The used type of noise is similar to [35]. More specifically, positions are randomly selected within a map and the labels of all pixels in a circular neighborhood of a random radius are flipped. The aggregate measure of noise contamination is the percentage of pixels with swapped labels. In the plots of Figures 4.1(g) and (h), the horizontal axis represents the percentage of noise-contaminated pixels in each skin map. Edge maps were contaminated with one third of this percentage. Noise was independently added to each artificial map r_s and r_e. In this experiment, 128 PSO particles and 35 PSO generations were used, and multiframes of eight views were considered.The plots indicate that the method exhibited robustness to moderate amounts of noise and failed for large amounts of noise. The exhibited robustness can be attributed to the large number of employed views. Since the noise of each view was assumed to be independent from all other views, the emerged consensus (over skin detection and edge detection) managed to cancel out low-variance noise.

Figure 4.1 also demonstrates that the design choices regarding the objective function E in Equation 4.1 are correct. This can be verified by the observed monotonic relation between E and the actual 3D hand pose estimation error.

Finally, Table 4.2 provides information on the runtime of these experiments, running on the computer described in the next paragraph. The table shows the number of multiframes per second for various parameterizations of the PSO (number of generations and number of particles per generation) and various number of views. The entry in boldface corresponds to 20 generations, 16 particles per generation and 2 views. According to the quantitative results presented earlier, this setup corresponds to the best trade-off between accuracy of results, computational performance and system complexity. This figure shows that the presented method is capable of accurately and efficiently recovering the 3D pose of a hand observed from a stereo camera configuration at 6.2Hz. If 8 cameras are employed, the method delivers poses at a rate of 1.6Hz.

Real-world image sequences were acquired using a multicamera system which is installed around a $2 \times 1m^2$ bench and consists of 8 *Flea2* PointGrey cameras. Cameras are synchronized by a timestamp-based software that utilizes a dedicated *FireWire 2* interface ($800\,MBits/sec$) which guarantees a maximum of $125\,\mu sec$ temporal discrepancy in images with the same timestamp. Each camera has a maximum framerate of $30\,fps$ at highest available image resolution which is 1280×960 pixels. The workstation where images are gathered has a quad-core Intel i7 920 CPU, 6 GBs RAM and an Nvidia GTX 295 dual GPU with $894\,GFlops$ processing power and 896 MBs memory per GPU core.

Using this experimental setup, several sequences of multiframes have been acquired, capturing various types of hand activities such as isolated motions, hand-environment interactions including object grasping. Figure 4.2 provides indicative snapshots of 3D hand pose estimation superimposed on the original image data. As it can be observed, the hand model estimated by the presented method is in agreement with the image data, despite the complex hand articulation and the relatively distant hand views.

4.2 Single Hand Tracking Using a Depth Sensor

The method presented in [46] treats the same scenario as above, i.e. a hand freely articulating without interacting with the environment. The input modality is however changed, the multi-camera system is replaced by a single RGB-D sensor. Within the model-based approach, we formulate the task as an optimization problem, seeking for the hand model parameters that minimize the discrepancy between the appearance and 3D structure of hypothesized instances of the adopted hand model (see Section 3.2) and actual observations. The selected visual cues are skin color and the depth map (see Sections 3.3.4 and 3.3.5 respectively). The resulting optimization problem is solved using the Particle Swarm Optimization (PSO) variant described in Section 3.5.1. Extensive experiments demonstrate that accurate and robust 3D tracking of hand articulations can be achieved in near real-time (15Hz at the time of publication of [46], currently more than 20Hz due to better exploitation of the existing parallelism and the advancement of hardware).

4.2.1 Method Outline

The raw input from the Kinect sensor is preprocessed to extract the visual cues of skin color and depth in the area of interest. Firstly, a median filter of size 3 is applied to the raw input depth map. The raw skin color map as well as the filtered depth map are both masked using the average depth of the solution for the previous frame. This is achieved by keeping a pixel if it is simultaneously skin colored and has a depth that is within $25cm$ of the previous solution. This test essentially enforces the implicit assumption that the tracked hand does not move more than $25cm$ per frame with respect to depth. The masked skin colored and depth maps are cropped and scaled to 64×64 pixels. The resulting, cropped skin colored and depth maps, o_s and o_d respectively are used in the next steps.

Given a hand pose hypothesis h and camera calibration information C, a synthetic depth map $r_d(h, C)$ is generated by means of rendering. By comparing this map with the respective observed depth map o_d, a "matched depths" binary map $r_m(h, C)$ is produced. More specifically, a pixel of r_m is set to 1 if the respective depths in o_d and r_d differ less than a predetermined value d_m or if the observation is missing (signified by 0 in o_d), and 0 otherwise. This map is compared to the observed skin color map o_s, so that skin colored pixels that have incompatible depth observations do not positively contribute to the total score.

The objective function is a quantification of the distance between a hand pose hypothesis h and the observation maps O. Specifically, the function $E(h, O)$ measures the discrepancy between the observed skin and depth maps O computed for a given frame and the skin and depth maps that are rendered for a given hand pose hypothesis h:

$$E(h, O) = D(O, h, \mathrm{C}) + \lambda_k \cdot kc(h). \tag{4.4}$$

In Equation 4.4, λ_k is a normalization factor that has been determined experimentally to balance the contributions of the two terms. The function D in Equation 4.4 is defined as

$$D(O, h, C) = \frac{\sum min(|o_d - r_d|, d_M)}{\sum(o_s \vee r_m) + \epsilon} + \lambda \left(1 - \frac{2\sum(o_s \wedge r_m)}{\sum(o_s \wedge r_m) + \sum(o_s \vee r_m)}\right). \tag{4.5}$$

The first term of Equation 4.5 models the absolute value of the clamped depth differences between the observation O and the hypothesis h. Clamping to a maximum depth d_M is necessary, because otherwise a few large depth discrepancies disproportionally penalize an otherwise reasonable fit. This fact, in turn, creates large variations of the objective function's value near the optimum, hindering the performance of the adopted optimization strategy. A small value ϵ is added to the denominator of this term to avoid division by zero. The second term of Equation 4.5 models the discrepancies between the skin-colored pixels of the model and the observation. λ is a constant normalization factor that has been experimentally determined so that the contribution of both term is balanced. The sums are computed over entire feature maps.

The term kc in Equation 4.4 adds a penalty to kinematically implausible hand configurations, similarly to Equation 4.3. The elaborate collision scheme described in Section 4.3.1 was also considered here for kc, taking into account all possible pairs of relatively moving hand parts. Experimental results however demonstrated that for the majority of encountered situations, it suffices to penalize only adjacent finger inter-penetration. Thus we define kc as:

$$kc(h) = \sum_{p \in \mathrm{Q}} -\min(\phi(p, h), 0), \tag{4.6}$$

where Q denotes the three pairs of adjacent fingers, excluding the thumb, and ϕ denotes the difference (in radians) between the abduction-adduction angles of those fingers in hypothesis h. In all experiments, λ was set to 20 and of λ_k to 10. The depth thresholds were set to $d_m = 1cm$ and $d_M = 4cm$.

4.2.2 Experimental evaluation

The experimental evaluation of the presented method was based on synthetic data with ground truth information and on real-world sequences obtained by a Kinect sensor. The presented method ran on a computer equipped with a quad-core Intel i7 950 CPU, 6 GBs RAM and an Nvidia GTX 580 GPU with 1581 $GFlops$ processing power and 1.5 GBs memory. On that system, the average frame rate was 15Hz. As described in [15] there was still room for performance improvements, with the current implementation of the presented method running on a modern PC with rates exceeding $20Hz$.

The synthetic sequence that is used for evaluation consists of the same 360 consecutive hand poses used also for [48], described in Section 4.1. Several experiments were carried out to assess the influence of specific factors to the performance of the method. Specifically, the considered factors are: PSO parameters, the placement of the hand with respect to the virtual camera and observation noise.

Figure 4.3(a) illustrates the behavior of the method with respect to the PSO parameters (number of generations and particles per generation). The product of these parameters determines the computational budget, as described in Section 3.5.1. The horizontal axis of the plot denotes the number of PSO generations. Each plot of the graph corresponds to a different number of particles per generation. Each point in each plot is the median $\mathcal{D}$ of the error Δ for 20 repetitions of an experiment run with the specific parameters. A first observation is that $\mathcal{D}$ decreases monotonically as the number of generations increase. Additionally, as the particles per generation increase, the resulting error decreases. Nevertheless, employing more that 25 generations and more than 64

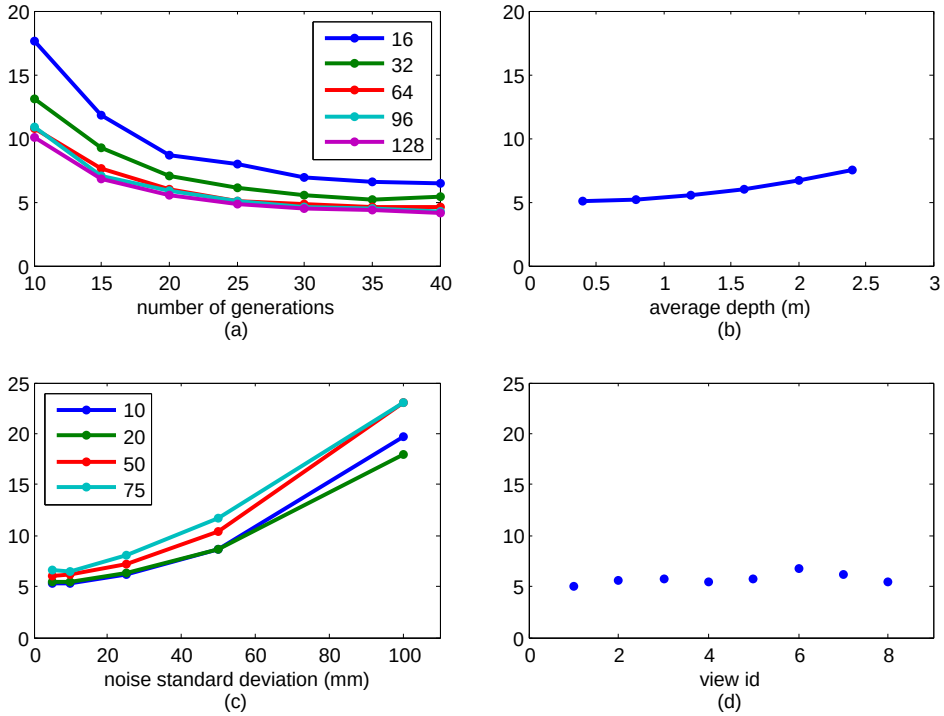


Figure 4.3: Quantitative evaluation of the performance of the method with respect to (a) the PSO parameters (b) the distance from the sensor (c) noise and (d) viewpoint variation.

particles results in insignificant improvement of the method's accuracy. The gains, if any, are at most $0.5mm$. For this reason, the configuration of 64 particles for 25 generations was retained in all further experiments.

Another investigation considered the effect of varying the distance of the hand from the hypothesized sensor. This explores the usefulness of the method in different application scenarios that require observations of a certain scene at different scales (e.g., close-up views of a hand versus distant views of a human and his/her broader environment). To do this, we generated the same synthetic sequences at different average depths. The results of this experiment are presented in Figure 4.3(b). At a distance of half a meter the error is equal to $5mm$. As the distance increases, the error also increases; Interestingly though, it doesn't exceed $7.5mm$ even at an average distance of $2.5m$.

The method was also evaluated with respect to its tolerance to noisy observations. Two types of noise were considered. Errors in depth estimation were modeled as a Gaussian distribution centered around the actual depth value with the variance controlling the amount of noise. Skin-color segmentation errors were treated similarly to [35], by randomly flipping the label (skin/non-skin) of a percentage of pixels in the synthetic skin mask. Figure 4.3(c) plots the method's error in hand pose estimation for different levels of depth and skin segmentation error. As it can be verified, the hand pose recovery error is bounded in the range $[5mm..25mm]$, even in data sets very heavily contaminated with noise.

Finally, we assessed the accuracy in hand pose estimation with respect to viewpoint variations. This was achieved by placing the virtual camera at 8 positions dispersed on

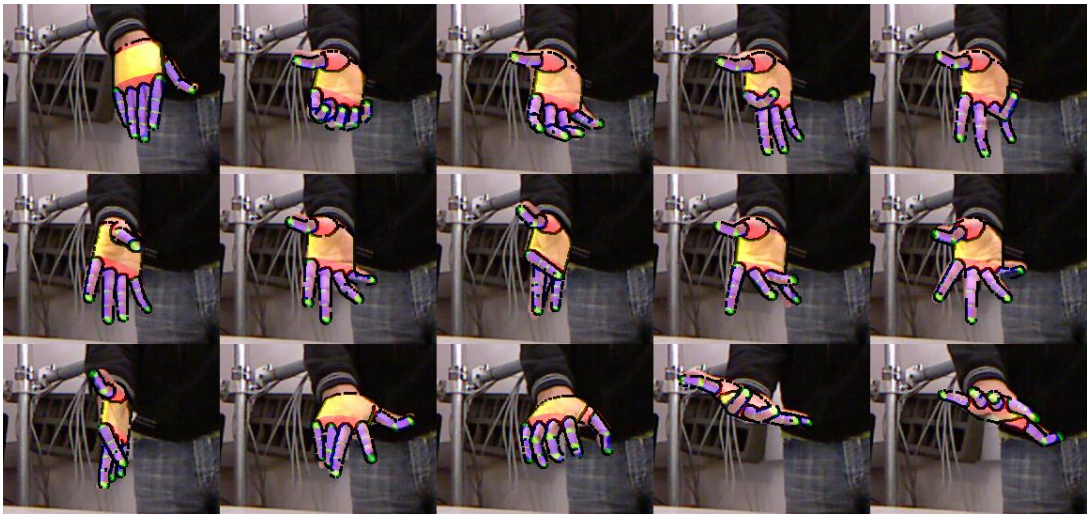


Figure 4.4: Indicative results on real-world data.

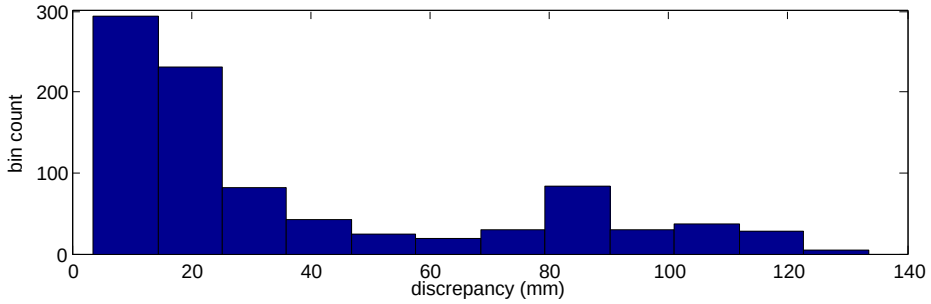


Figure 4.5: Performance of single-frame hand pose estimation.

the surface of a hemisphere placed around the hypothesized scene. These positions were the same as the camera positions in the experiments presented in Section 4.1. The data points of Figure 4.3(d) demonstrate that viewpoint variations do not significantly affect the performance of the method.

Several long real-world image sequences were captured using a Kinect, recording both RGB and depth data. The sequences exhibit hand waving, palm rotations, complex finger articulation as well as grasp-like hand motions [1]. Indicative snapshots are shown in Figure 4.4. As it can be observed, the estimated hand model is in very close agreement with the image data, despite the complex hand articulation and significant self occlusions.

Finally, besides tracking, we tested the capability of the presented method to perform automatic hand model initialization, i.e., single-frame hand pose estimation. Essentially, this boils down to the capability of PSO to optimize the defined objective function even when parameter ranges are very broad. To do so, the presented algorithm ran many times, each initialized at different hand positions and orientations close to the observed hand (the largest skin color blob). In order to localize the hand in 3D, the median depth of this blob was used to estimate the depth of the observation (Z dimension), and the ray defined from the image centroid of this blob was solved for the X and Y world coordinates. The best scoring hypothesis among the multiple runs was kept as the recovered pose. To

[1] Results available online at http://www.youtube.com/watch?v=Fxa43qcm1C4 .

assess the method, a set of 45 frames was selected at regular intervals from a real-world sequence and each hand pose recognition was performed 20 times. For the quantitative assessment of the hand pose recognition accuracy, we used as a reference the hand model parameters that were recovered from an experiment that tracked the hand articulation over the whole sequence. Figure 4.5 shows the histogram of estimation error $\mathcal{D}$ for all the performed (20×45) experiments. As it can be verified, in 74% of them, the estimated pose deviated $4cm$ or less from the tracked pose. The secondary histogram peak around $8cm$ corresponds to some ambiguous poses for which sometimes the mirrored pose was estimated.

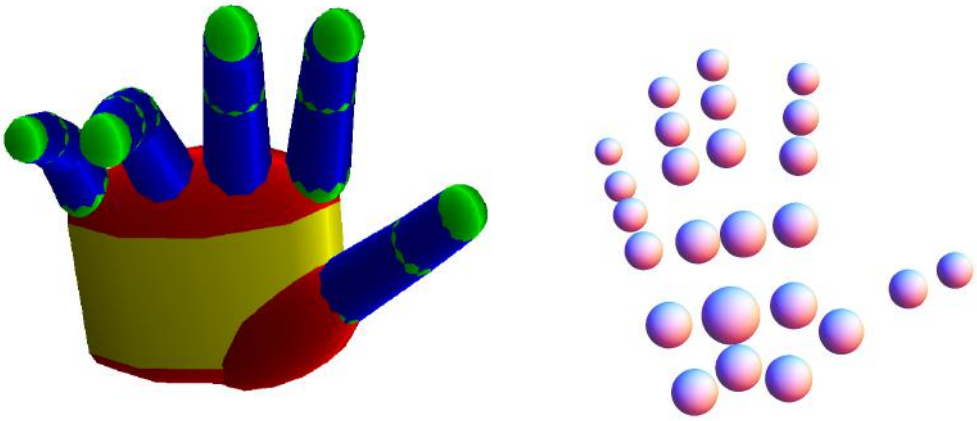
4.3 Tracking a Hand Manipulating an Object using Multi-view Input

Due to occlusions, the estimation of the full pose of a human hand interacting with an object is much more challenging than pose recovery of a hand observed in isolation. In [49] we extend the formulation of the hand pose estimation and tracking as an optimization problem to include a manipulated object of known shape class. The parameter space, and therefore also the solution, are the 26-DOF hand pose together with the pose and model parameters of the manipulated object. The parameterization jointly considers the hand and the manipulated object so that a solution (a) best explains the incompleteness of observations resulting from occlusions due to hand-object interaction and (b) is physically plausible in the sense that the hand does not share the same physical space with the object. The presented method was the first to efficiently solve the continuous, full-DOF, joint hand-object tracking problem based solely on markerless multicamera input. Additionally, it was the first to show how hand-object interaction can be exploited as a context that facilitates hand pose estimation, instead of being considered as a complicating factor, as for example in the case of [42]. The employed optimizer is the Canonical variant of PSO (see Section 3.5.1). Extensive quantitative and qualitative experiments with simulated and real world image sequences as well as a comparative evaluation with a state-of-the-art method for pose estimation of isolated hands are presented.

4.3.1 Method Outline

We parameterize the human hand as described in Section 3.2, illustrated in Figure 4.6. A human hand pose is denoted as h. In this work, an elaborate collision model is used (Equation 4.11). The considered primitives of the hand's collision model are spheres, illustrated in Figure 4.6.

For the representation of an object, in principle any parametric model can be used. The representation of common handheld objects such as cuboids, ellipsoids and cylinders requires 3, 3 and 2 intrinsic shape parameters, respectively. More complex parametric shape models like superquadrics require as many as 6 parameters. Regardless of the intrinsic shape parameterization, 7 additional parameters are required, 3 for 3D position and 4 for a quaternion-based representation of 3D orientation. In this work, we provide experimental results with ellipsoids, cuboids and cylinders, respectively with a total of 10, 10 and 9 parameters. An object pose is denoted as o.



(a) (b)

Figure 4.6: Graphical illustration of the employed 26-DOF 3D hand model, consisting of 37 geometric primitives (a) and the 25 spheres constituting the hand's collision model (b).

There is no inherent limitation that prevents the method from being able to handle more complex object models, provided that this does not increase the dimensionality of the problem prohibitively. Interestingly, a complex, known 3D object represented as a mesh has less DOFs compared to the used parametric object models (6 DOFs, 3D pose).

Given a joint parametric hand-object model, the goal is to estimate the parameters for the model, $m = (h, o)$ which give rise to the hand-object configuration that (a) is most compatible to the image features present in multiframe M and (b) is physically plausible in the sense that two different rigid bodies cannot share the same physical space (inter-penetration constraints). To achieve this, the objective function $E(m, M)$ is defined as:

$$E(m, M) = \sum_{I \in M} D(I, m) + \lambda_k W(m). \tag{4.7}$$

In Equation 4.7, the first term quantifies the discrepancies of a given hand-object model m to the actual camera-based observations, while the second term quantifies the penetration depth between the hand and the object, but also among hand parts (fingers, palm, etc). λ_k is a weighting factor experimentally set to $\lambda_k = 0.1$ for all the presented results. The sum in the first term runs over all available views I in multiframe M: a partial score $D(I, m)$ of the candidate hypothesis m to each available image I in the multiframe M is computed. Consequently, they are all summed to form the first term of E, which constitutes the overall contribution of visual evidence in the objective function.

To compute $D(I, m)$, we first render comparable image features from each hypothesized hand-object model. More specifically, an edge map $r_e(m)$ and a skin color map $r_s(m)$ can be generated by means of rendering. The implicit assumption made at this point is that an object does not contain skin-colored pixels. Thus, the hand component h of m contributes to the skin color map $r_s(m)$ by setting visible hand pixels to 1, while the object component o of m contributes to the skin color map $r_s(m)$ by setting map pixels to 0. Experimental results have verified that the presence of a moderate number of skin-colored pixels on the object's surface does not affect the accuracy of the method.

$D(I,m)$ is then defined as:

$$\begin{aligned} D(I,m) &= 1 - \frac{2\sum o_s(I) \wedge r_s(m)}{(\sum o_s(I) \wedge r_s(m)) + (\sum o_s(I) \vee r_s(m))} \\ &\quad + \lambda \frac{\sum o_d(I) \cdot r_e(m)}{\sum r_e(m) + \epsilon}, \end{aligned} \tag{4.8}$$

where $o_s(I), o_d(I)$ are respectively the observed skin color and edge maps and ϵ is a small quantity to prevent division by zero. The first row of Equation 4.8 models the discrepancies between the skin-colored pixels of the model and the observations. Sums are computed over entire feature maps. In contrast to Equation 4.2, this part of the objective function is now normalized to the interval [0..1]. The second row models the discrepancies between the rendered edge maps and the observed edge maps. This is achieved by summing the values of the distance-transformed observation edge map that concur with the edges of the rendered model. λ is a constant normalization factor that was set to 0.02 in all experiments.

The role of function $W(m)$ in Equation 4.7 is to penalize (a) hand configurations where hand parts intersect each other (self-penetration) and (b) hand-object configurations where the hand h intersects the object o (interpenetration). Let $P(p_i, p_j)$ be the minimum magnitude 3D translation that is required so that the volume of intersection of geometric primitives p_i and p_j becomes equal to 0. This is effectively computed using the *Open Dynamics Engine* (ODE) [70]. Let also S_h be the primitives of the hand's collision model, as shown in Figure4.6(b). The self-penetration P_{hh} of a given hand configuration is defined as

$$P_{hh} = \max_{i \in S_h, j \in S_h, i \neq j} \{P(i,j)\}. \tag{4.9}$$

The interpenetration P_{ho} is similarly defined as

$$P_{ho} = \max_{i \in S_h} \{P(i,o)\}. \tag{4.10}$$

Then, $W(m)$ is defined as

$$W(m) = \max\{P_{hh}, P_{ho}\}. \tag{4.11}$$

With this approach, both self- and inter- penetrations are treated in a uniform manner.

4.3.2 Experimental Evaluation

The presented method has been validated extensively based on both synthetic and real-world sequences of multiframes. First, we demonstrate the accuracy and the computational performance of the presented Hand-Object Pose Estimation method, hereafter abbreviated as *HOPE* on a synthetically rendered dataset where hands perform different grasps on a variety of objects, shaped as spheres, cylinders or boxes. We also compare the performance of *HOPE* to that of the method in Section 4.1, hereafter abbreviated as *PEHI* (Pose Estimation of Hands in Isolation). A final experiment with synthetic data involves the application of *HOPE* to a data set showing hands in isolation. The goal of this experiment is to show that *HOPE* can also estimate the pose of hands in isolation effectively, as a special case. Besides the synthetic data, we also provide qualitative evidence on how *HOPE* and *PEHI* perform on real sequences of multiframes. Although

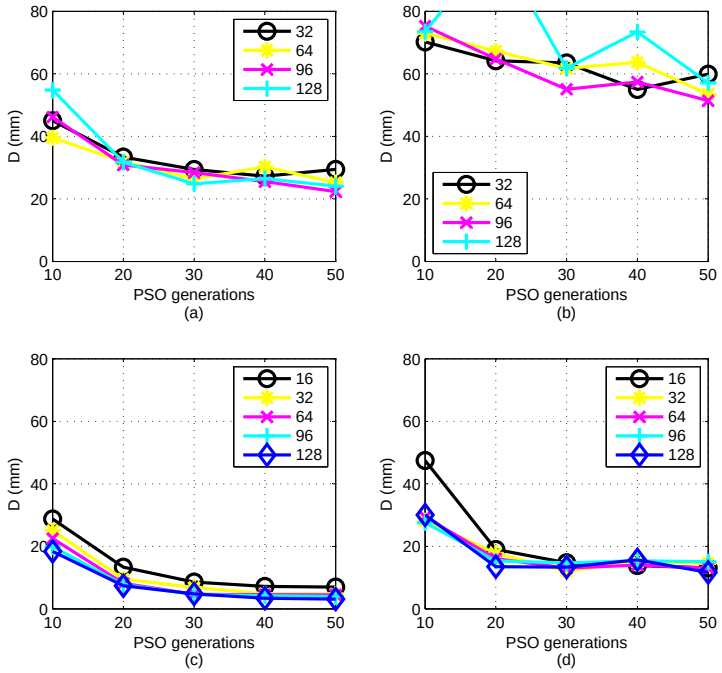


Figure 4.7: Mean error $\mathcal{D}$ for hand pose estimation (in mm) for *HOPE* (left) and *PEHI* (right) for different PSO parameters and number of views. (a),(b): Varying PSO particles and generations for 2 views. (c),(d): Same as (a),(b) for 8 views.

ground truth information is not available, these indicative results confirm the superiority of *HOPE* over *PEHI* which is in accordance with the experimental results over synthetic data.

Experiments with synthetically produced sequences of multiframes were performed to assess the presented method based on ground truth data. Specifically, we simulated different grasps of three different objects (an ellipsoid, a cylinder, and a box) performed by the synthetic hand model. The interaction of the hand with each of these three objects was observed by 8 virtual cameras surrounding the scene. This resulted in three sequences consisting of 116 multiframes of 8 frames, each. The required cue maps (edges, skin color) were synthesized through rendering. The error metric quantifying the discrepancy between two hand poses, a ground-truth and an estimated one (see Section 3.7) was computed as follows. The five fingertips as well as the center of the palm were selected as reference points. For each such reference point, the Euclidean distance between its estimated position and its ground truth position is first calculated. These distances are averaged across all multiframes of each sequence, and all sequences. This results in a single error value $\mathcal{D}$ for the whole dataset.

Figures 4.7(a) and (c) illustrate the estimated error $\mathcal{D}$ of the *HOPE* method as a function of the PSO parameters. In Figure 4.7(a), $\mathcal{D}$ is plotted as a function of the number of PSO generations and particles per generation, for multiframes consisting of 2 views. The cameras providing these two views are placed opposite to each other. $\mathcal{D}$ takes values between $22mm$ and $55mm$. It can be verified that for more than 30 generations and more than 32 particles/generation the error in 3D hand pose recovery for *HOPE* does not vary considerably and it is in the order of $25mm$.

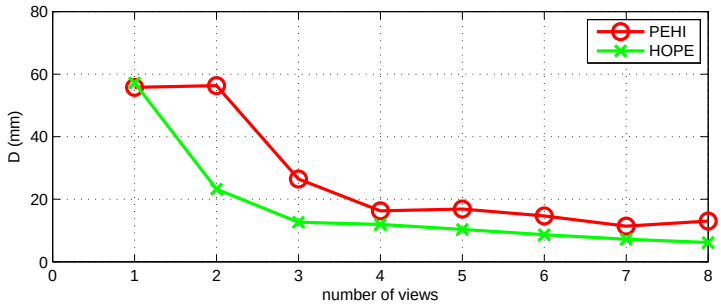


Figure 4.8: Mean error $\mathcal{D}$ for hand pose estimation (in mm) of *HOPE* (green) and *PEHI* (red) for different number of views. The computational budget is fixed to 40 generations and 64 particles/generation.

Figure 4.7(b) is analogous to that of Figure 4.7(a) for the *PEHI* algorithm. In this case, the mean error $\mathcal{D}$ does not decrease monotonically as a function of particles. This is attributed to the incomplete/occluded hand observations that undermine the convergence of *PEHI*. $\mathcal{D}$ now ranges between $51mm$ and $101mm$. It can be verified that for more than 30 generations and more than 32 particles/generation the error in 3D hand pose recovery for *PEHI* is in the order of $55mm$. Thus, the error of *PEHI* is on average more than twice the error of *HOPE*.

Figures 4.7(c) and (d) are analogous to Figures 4.7(a) and (b), except the fact that each multiframe now consists of 8 rather than 2 views. $\mathcal{D}$ takes values between $3mm$ and $29mm$ for *HOPE* and between $12mm$ and $47mm$ for *PEHI*. For more that 30 PSO generations and more than 32 particles per generation the error of *PEHI* is still more than twice the error of *HOPE*. Interestingly, what *HOPE* achieves with 16 particles and 20 generations is equal or better to what *PEHI* achieves with any of the tested particles/generations combinations.

In order to better assess the behavior of the method with respect to the number of available views, additional experiments with a varying number of views were conducted. Figure 4.8(e) shows the behavior of *HOPE* and *PEHI* as a function of the size of a multiframe. For the experiments with less than 8 views, these were selected empirically to be as complementary as possible. The computational budget of PSO was set to 64 particles running for 40 generations. The obtained results demonstrate that modeling the occluder and the physical constraints is more beneficial than adding an extra camera. As an example, exploiting these constraints with two cameras is still better than with three cameras and the hand alone. In fact, what *HOPE* achieves with three views is already better to what *PEHI* achieves with as many as eight.

Overall, the experiments in Figures 4.7 and 4.8 show a consistent and significant superiority of *HOPE* over *PEHI* which is dominant in the case of a limited number of available views. This is important because it enables practical joint hand-pose estimation by a multicamera system with a few cameras that is associated with less costs, complexity of setup and requirements for computational resources.

Besides its superiority in hand pose estimation, *HOPE* also estimates the model parameters of the manipulated object. The average positional error of object detection across all sequences of multiframes in the experiments presented in Figures 4.7 and 4.8 is $3mm$ (Euclidean distance between true and estimated positions) and the average ori-

Table 4.3: Estimated/actual parameters for the object models in the experiments with synthetic data.

Object	Estimated/Actual parameters (in mm)
Cylinder	Radius: 54/55, Height: 127/128
Ellipsoid	X: 54/55, Y: 83/85, Z: 126/128
Box	X: 77/77, Y 128/129, Z: 155/156

Table 4.4: Estimated/actual parameters for the object models in the experiments of Figure 4.11.

Object	Estimated/actual parameters (in mm)
Cylinder	Radius: 51/53, Height: 121/131
Ellipsoid	X: 128/116, Y: 128/116, Z: 122/116
Box	X: 66/67, Y: 158/150, Z: 84/93

entation error is 2°. Table 4.3 shows the actual and estimated object parameters. The later are averaged for all the multiframes of the sequence that depicts the corresponding object. It can be verified that for all types of objects, the estimated model parameters are very close to the ground truth.

The runtime of a GPU-powered implementation of *HOPE* [15] for runs of 40 PSO generations and 64 particles per generation was 0.31sec for a single-view multiframe and 2.19sec for an 8-view multiframe at the time of publication of the method. An online version of the system employing 4 cameras, operated at 2 fps. In multiframes of sizes larger than 2, *PEHI* was approximately 20% faster than *HOPE*. This overhead is attributed to the computation of the $W(m)$ component of the objective function. Since this is a fixed overhead that is independent of the multiframe size, the relative difference in computational performance decreases with the number of views.

Concluding the quantitative assessment, we applied both *HOPE* and *PEHI* to a synthetic image sequence (400 multiframes, 8 frames/multiframe) showing non-rigid motion of hands in isolation. Figure 4.9 plots the mean error $\mathcal{D}$ as a function of the number of the employed views. For both algorithms, 40 PSO generations and 64 particles per generation were used. For *HOPE*, a cylindrical object was hypothesized. The results show that the performance of the two algorithms is comparable, a fact that indicates the capability of *HOPE* to track hands observed in isolation. Expectedly, *HOPE* estimated the presence of very small objects (size in the order of a few mms).

Real-world image sequences were acquired using a multicamera system (Figure 4.10) installed around a $2 \times 1m^2$ bench and consisting of 8 synchronized and calibrated *Flea2* PointGrey cameras. Each camera has a maximum framerate of 30 fps, at 1280×960 image resolution. However, the core processing is performed on 256×256 windows centered around the previous multiframe solution, as described in 3.3.2.

Three sequences of multiframes have been acquired, each showing a hand grasping and manipulating a spherical (301 multiframes), a cylindrical (261 multiframes), and a box (251 multiframes) object. Figure 4.11(a) provides sample results obtained by applying *HOPE* (top row) and *PEHI* (bottom row) to a specific multiframe of the sphere sequence.

Table 4.5: The mean value of the objective function of *HOPE* and its standard deviation when optimization searches for cylinders, ellipsoids and cuboids for a sequence showing an ellipsoid (sphere).

	Cylinder	Ellipsoid	Cuboid
Mean value	3.02	2.65	3.95
Stdev.	0.68	0.57	1.17

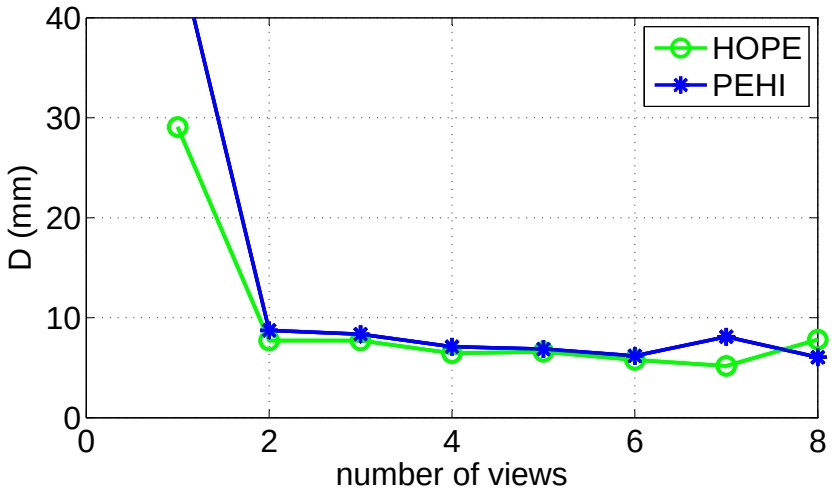


Figure 4.9: Performance of *HOPE* and *PEHI* on a synthetic sequence of multiframes that shows hands in isolation. 64 PSO particles and 40 generations have been used in both cases.

Since the hand is mostly occluded by the sphere in all views, *HOPE* estimates the hand configuration correctly while *PEHI* fails completely. Similar results were obtained in the case of the cylinder sequence which shows a hand grasping and turning a cylindrical object up-side down. Figure 4.11(b) shows four frames acquired from the same camera in different moments in time. *HOPE* tracks the configuration of the hand throughout the whole sequence whereas *PEHI* looses track of the hand as soon as the later becomes severely occluded by the object. Figure 4.11(c) shows a similar result for the box sequence. Additionally, in Table 4.4, we compare the actual, physically measured object shape parameters to the ones estimated by *HOPE*, computed by averaging estimations for all multiframes of a given sequence. The standard deviation of these estimations is in the order of a few millimeters. It can be verified that the error in object shape estimation is satisfactory.

For *HOPE*, we also ran a simple classification experiment. Although shape classification is not the focus of this work, this experiment provides an indirect indication of the accuracy of the optimization process. For the sphere sequence (Figure 4.11(a)), we ran *HOPE* assuming a cuboid, an ellipsoid and a cylinder. Table 4.5 shows the mean value and the standard deviation of the objective function of *HOPE* in all multiframes of the sequence. As it can be verified, the hypothesis of an ellipsoid better explains the observed scene. In fact, 98.67% of the multiframes were better explained by the ellipsoid, 1.33% by the cylinder and none by the cuboid.

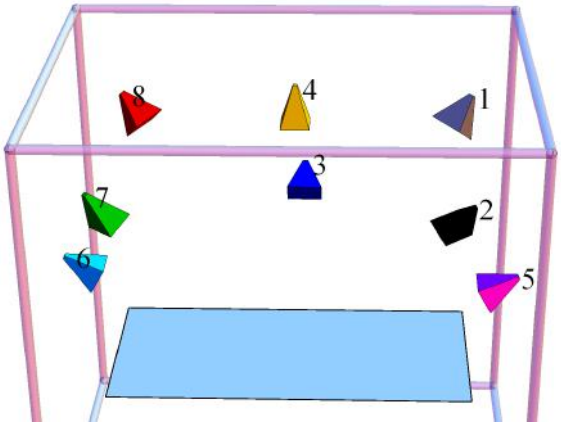


Figure 4.10: Camera setup for the experiments with real data.

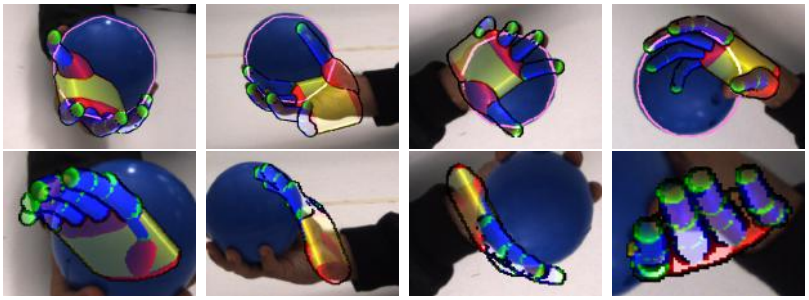
Finally, Figure4.12, shows sample snapshots from the results obtained on a sequence of a hand performing fine manipulation of an elongated cuboid. Visual inspection confirms that the accuracy of *HOPE* is quite satisfactory, despite the complex and challenging hand-object interaction. Sample videos out of these experiments are available online[2].

4.4 Tracking Two Hands in Strong Interaction

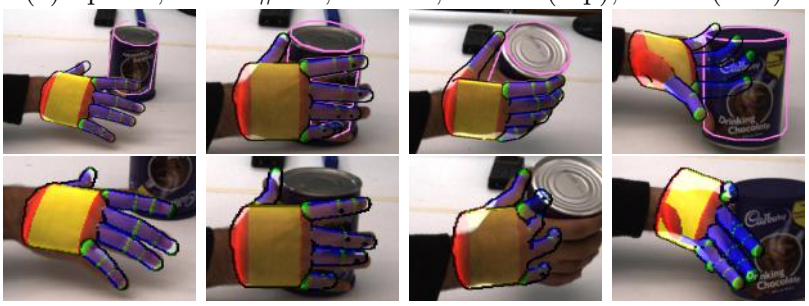
We now examine a third hand tracking scenario, namely tracking two hands using input from an RGB-D sensor. Specifically, we aim to track the full articulation of two hands that interact with each-other in a complex, unconstrained manner. Following the joint modelling approach, the resulting optimization problem has a 54-dimensional parameter space that models all possible configurations of two hands, each represented as a kinematic structure with 26 Degrees of Freedom (DoFs). To the best of our knowledge, the presented method was the first to attempt and achieve the articulated motion tracking of two strongly interacting hands at the time of its publication. Extensive quantitative and qualitative experiments with simulated and real-world image sequences demonstrate that an accurate and efficient solution of this problem is indeed feasible.

The adopted hand model (see Section 3.2) is adapted to handle the articulation ranges of both a right and a left hand. One of each is assumed to be present in the scene. Within the model-based formulation, the complex occlusions faced in this scenario are effortlessly handled. These occlusions include the self-occlusions that occur in the single hand-tracking case, and additionally those that occur while the hands are in close interaction. The input preprocessing is similar to [46], with the exception that the working resolution for the cropped skin color and depth map is 128×128 pixels. The objective function is the same as the one used in [46], described in Section 4.2 (Equations 4.4, 4.5 and 4.6). Finally, the same PSO variant as in [46] is used here as well to solve the resulting optimization problem.

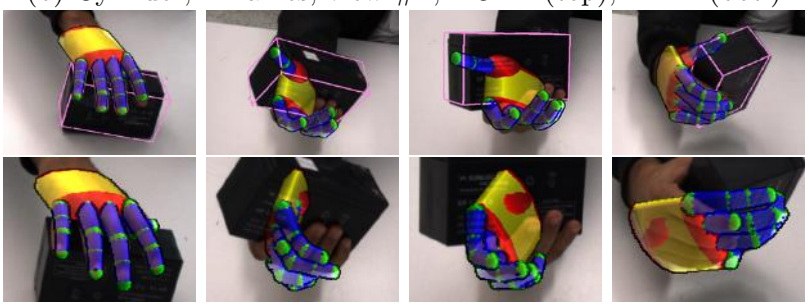
[2]http://www.youtube.com/watch?v=N3ffgj1bBGw



(a) Sphere, frame #103, 4 views, *HOPE* (top), *PEHI* (bot.)



(b) Cylinder, 4 frames, view #2, *HOPE* (top), *PEHI* (bot.)



(c) Box, 4 frames, view #1, *HOPE* (top), *PEHI* (bot.)

Figure 4.11: Sample frames from the results obtained by *HOPE* and *PEHI* in real-world experiments. For *HOPE* the projection of the estimated 3D object model is shown in pink color.

4.4.1 Experimental evaluation

Synthetic data as well as real-world sequences obtained by a Kinect sensor [12] were used to experimentally evaluate the presented method. Experiments were performed on a computer equipped with a quad-core Intel i7 950 CPU, 6 GB RAM and an Nvidia GTX 580 GPU with 1581 $GFlops$ processing power and 1.5 GB of memory.

The quantitative evaluation of the presented method has been performed using synthetic data. The employed synthetic sequence consists of 300 poses that encode typical interactions of two hands. To quantify the accuracy in hand pose estimation, the distance Δ between corresponding phalanx endpoints in the ground truth and in the estimated hand poses is measured. The average of all these distances, for both hands, over all the frames of the sequence constitutes the resulting error estimate $\mathcal{D}$. It is worth noting that these distances include estimations for hand points that, because of occlusions, are not observable in many frames frames of the sequence.

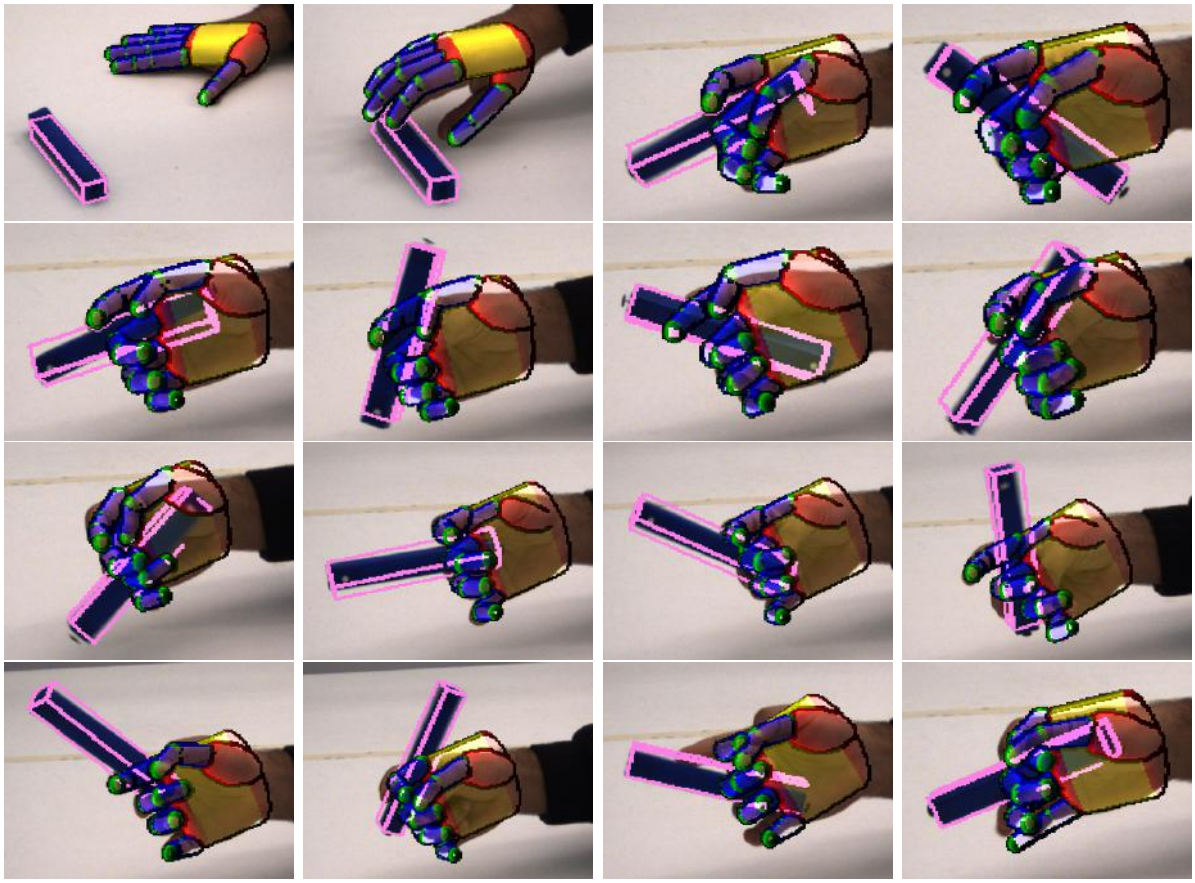


Figure 4.12: Snapshots from an experiment where a hand performs a complex manipulation of an elongated cuboid.

The influence of several factors to the performance of the method was assessed in respective experiments. Figure 4.13 illustrates the behavior of the method with respect to the PSO parameters (number of generations and particles per generation). The product of these parameters determines the computational budget of the presented methodology, i.e. the number of objective function evaluations for each each tracking frame. The horizontal axis of the plot denotes the number of PSO generations. Each plot of the graph corresponds to a different number of particles per generation. Each point in each plot is the median $\mathcal{D}$ of the error Δ for 20 repetitions of an experiment run with the specific parameters. A first observation is that $\mathcal{D}$ decreases monotonically as the number of generations increase. Additionally, as the particles per generation increase, the resulting error decreases. Nevertheless, employing more that 45 generations and more than 64 particles results in disproportionally small improvement of the method's accuracy. The gains are at most $2mm$ or roughly 30%, for a 5-fold increase in computational budget. For this reason, the configuration of 64 particles for 45 generations was retained in all further experiments. In terms of computational performance, tracking is achieved at a framerate of 4Hz on the computational infrastructure described above.

In another experiment we assessed the effect of varying the distance of the hands from the hypothesized sensor. By doing so, we explored the usefulness of the method

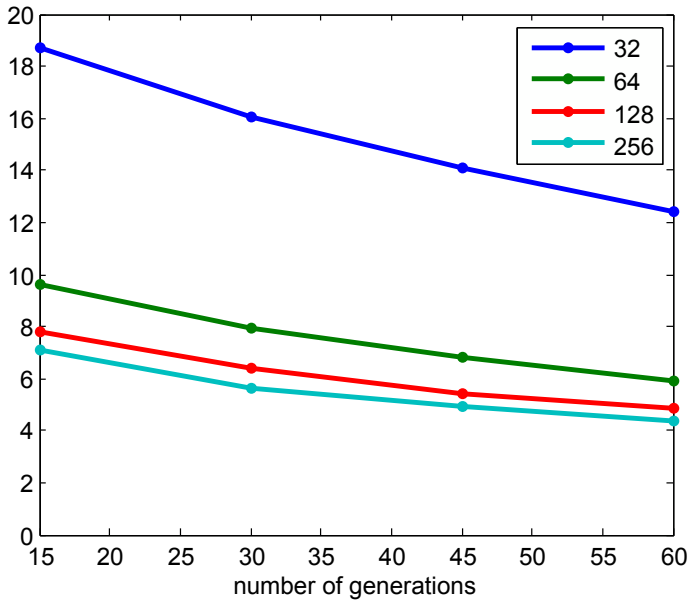


Figure 4.13: Quantitative evaluation of the performance of the method with respect to the PSO parameters. Each line of the graph corresponds to a different number of particles as shown in the legend.

in different application scenarios that require observations of a certain scene at different scales (e.g., close-up views of hands versus distant views of a human and his/her broader environment). To do this, we generated the same synthetic sequences at different average depths. The results of this experiment are presented in Figure 4.14. At a distance of 50cm the error is equal to $6mm$. As the distance increases, the error also increases; Interestingly though, it doesn't exceed $8.5mm$ even at an average distance of $2.5m$. The used synthetic maps do not contain any kind of noise, in contrast to what happens in practice: the amount of noise is related to the distance from the sensor for data acquired with a Kinect.

The tolerance of the method to noisy observations was also evaluated. Two types of noise were considered. Errors in depth estimation were modeled as a Gaussian distribution centered around the actual depth value with the variance controlling the amount of noise. Skin-color segmentation errors were treated similarly to [35], by randomly flipping the label (skin/non-skin) of a percentage of pixels in the synthetic skin mask. Figure 4.15 plots the method's error in hand pose estimation for different levels of depth and skin segmentation error. As it can be verified, the hand pose recovery error is bounded in the range $[6mm..23mm]$, even in data sets very heavily contaminated with noise.

The accuracy in hand pose estimation with respect to viewpoint variations was also assessed. This was achieved by placing the virtual camera at 8 positions dispersed on the surface of a hemisphere around the hypothesized scene. The data points of Figure 4.16 demonstrate that viewpoint variations do not significantly affect the performance of the method.

In a final experiment, we measured the performance of our single hand tracker (Section 4.2) on the synthetic data set of the previous experiments. To do so, the system described

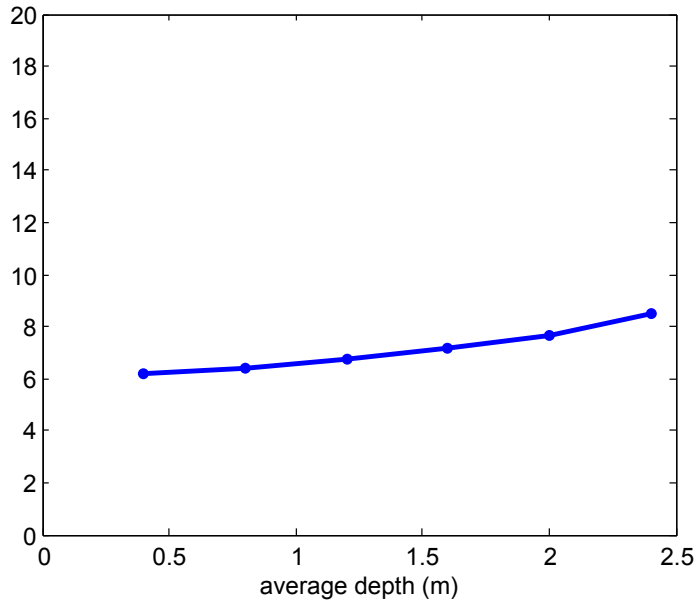


Figure 4.14: Quantitative evaluation of the performance of the method with respect to the average distance from the sensor.

in that work was used to track one of the two visible hands. The resulting error $\mathcal{D}$ for this experiment was 145mm. In practice, one instance of the single hand tracker is able to track accurately each of the two hands while it is not in interaction with the other. However, as soon as occlusions become extended due to hands interaction (for example, when one hand passes in front of the other), the track is often completely lost.

Towards the qualitative evaluation of the presented approach in real data, several long real-world image sequences were captured using a Kinect, recording both RGB and depth data. A video with the results obtained from one such sequence of 1776 frames is available online [3]. Indicative snapshots are shown in Figure 4.17. Evidently, the estimated hand postures are in very close agreement with the image data, despite the complex articulation and strong interactions of the two hands.

4.5 Single Hand Tracking using the Visual Hull

In our work presented in [54], the visual hull [60] is explored as the main visual cue of a model-based hand-tracking method that uses input from a calibrated multi-camera system of arbitrary 3D configuration. The use of the visual hull as the main observation cue manages a balanced contribution of observations from all available views to a single visual score. More specifically, the available observations contribute in the computation of the visual hull according to the available calibration information. This visual hull is consequently used as the main visual cue, directly yielding a single score for each candidate hand pose (see Equation 4.12 below). This is in contrast to the methods

[3] http://youtu.be/e3G9soCdIbc

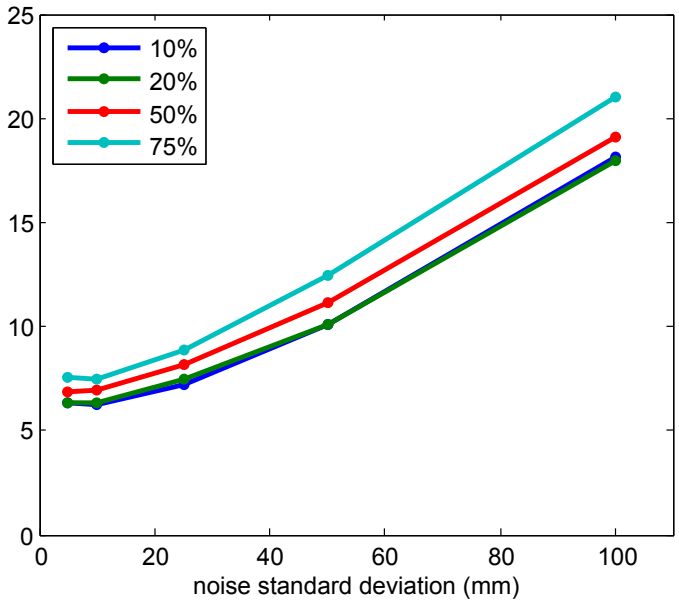


Figure 4.15: Quantitative evaluation of the performance of the method with respect to synthesized depth and skin-color detection noise.

presented in Sections 4.1 and 4.3 that compute a partial score for each available view and then sum them to yield an overall score (see Equations 4.1 and 4.7).

The formulation as an optimization problem makes use of an objective function that quantifies the discrepancy between the visual hull of the hypothesis and observation. A term that quantifies the discrepancy between hypothesized and observed image edges is also used. This objective is minimized using the variant of the Particle Swarm Optimization (PSO) algorithm presented in Section 3.5.1. We investigate the behavior of the resulting system in extensive experimental evaluations, comparing it with our approach in [48] that treats the available observations from each view in an independent way, as described in Section 4.1. The comparisons exhibit the unexpected result that, for synthetic image sequences without noise simulation the two methods perform comparably. The situation changes when artificially introducing progressively increasing levels of noise to the data: there is a point where the method of Section 4.1 breaks down whereas this approach manages to keep track. Finally results in real-world data confirm the applicability of this approach.

4.5.1 Method Outline

The objective function $E(h, O)$ is appropriately designed and formulated so as to quantify the discrepancy between a hand pose h, parameterized as described in Section 3.2, and the observation O. As already introduced, the employed image features are the visual hull and image edges. The visual hull is computed from the foreground masks of the object of interest. For our case, given a multiframe M, we use the skin color maps as the foreground masks, and the method proposed in [61] is employed to compute the observed visual hull $o_v(M)$. The edge maps are also further processed, undergoing distance transform, yielding

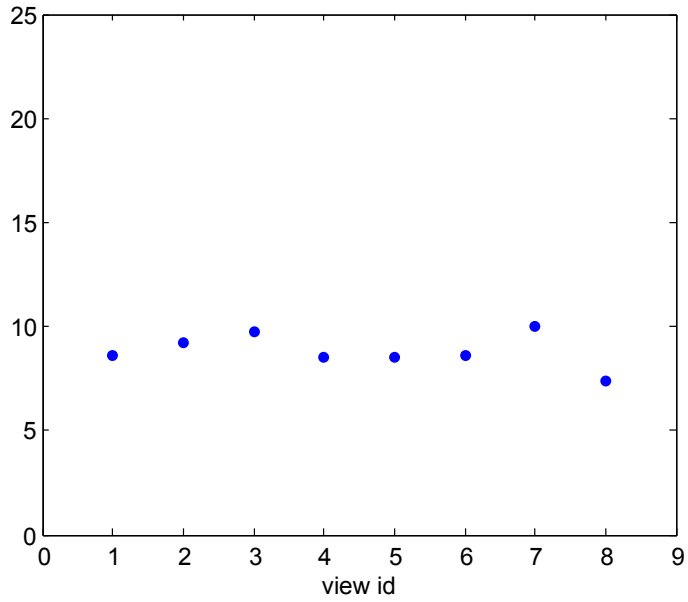


Figure 4.16: Quantitative evaluation of the performance of the method with respect to viewpoint variation.

the observed distance transform maps $o_d(I)$, where I denotes the respective input image of the multiframe $I \in M$.

Given this input, we proceed with the formulation of the objective function E. Specifically, given a hand pose h, a set of observations O and the camera calibration information C we compute

$$E(h, O) = D(O, h, \mathrm{C}) + \lambda_k \cdot kc(h). \tag{4.12}$$

where D quantifies the discrepancy between observed and hypothesized hand volumes and image edges, and kc adds a penalty for kinematically impossible hand configurations. The term λ_k is an experimentally determined normalization factor.

The function D in Equation 4.12 is computed as following. Given a hand pose hypothesis h and camera calibration information $c_i \in \mathrm{C}$, skin occupancy maps $r_s(h, c_i)$ and edge maps $r_e(h, c_i)$ for each view i are generated by means of rendering. The volume reconstruction methodology of [61] is then once again employed with input the rendered maps $r_s(h, c_i)$ to produce an occupancy volume $r_v(h)$ that can be directly compared with the observation $o_v(M)$. The comparison between these occupancy maps quantifies the discrepancy between the observed and the hypothesized hand pose. This is achieved by computing

$$D(M, h) = 1 - \frac{2 \sum o_v \wedge r_v}{(\sum o_v \wedge r_v) + (\sum o_v \vee r_v)} + \lambda \frac{\sum o_d \cdot r_e}{\sum r_e + \epsilon}, \tag{4.13}$$

where for the sake of notational simplicity o_v denotes $o_v(M)$, o_d denotes $o_d(I)$, r_e denotes $r_e(h, c_i)$ and r_v denotes $r_v(h)$. A small term ϵ is added to avoid division by zero. Regarding the operators, $\wedge$ and $\vee$ denote per-voxel operations of the respective maps and the summation is taken over the entire maps.

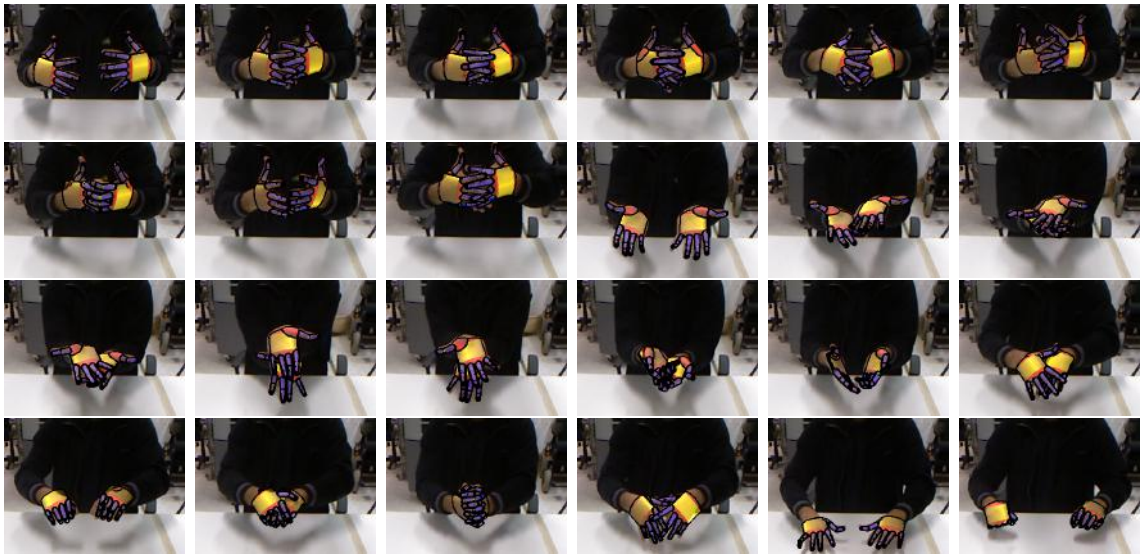


Figure 4.17: Snapshots from an experiment where two hands interact with each other (cropped 320×240 regions from the original 640×480 images).

The function kc in Equation 4.12 adds a penalty to kinematically implausible hand configurations. It is defined similarly to [48] as in Equation 4.3, penalizing only adjacent finger inter-penetration. Specifically:

$$kc(h) = \sum_{p \in \mathrm{Q}} \begin{cases} -\phi(p) & \phi(p) < 0 \\ 0 & \phi(p) \geq 0 \end{cases}, \tag{4.14}$$

where Q denotes the three pairs of adjacent fingers, excluding the thumb, and ϕ denotes the difference (in radians) between the abduction-adduction angles of those fingers. In all experiments, the value of λ_k was set to 0.1 and the value of λ was set to 0.01.

4.5.2 Experimental Evaluation

A number of quantitative experiments was conducted, designed to compare the behavior of the presented method to that of Section 4.1. These experiments analyzed the behavior of the objective functions of the methods, investigated the parameterization of PSO, assessed the effect of segmentation noise and also explored the behavior for different numbers of available views of the scene. Qualitative results in real-world data are also presented.

In order to perform a fair comparison of the presented method to that of Section 4.1, we slightly modify the implementation of that method by including the processing step described in Section 3.3.2. The computed visual cues of skin color and edges are appropriately cropped and resized to a fixed working resolution of 128×128. This modification improves the performance of both methods, and therefore we compare them on this basis.

In all the experiments with visual hulls we used a reconstruction space of 128^3 voxels, centered around the previously estimated position of the observed hand. The physical dimensions of this space were $240mm$ along each edge, resulting in a voxel size slightly larger than $2mm$.

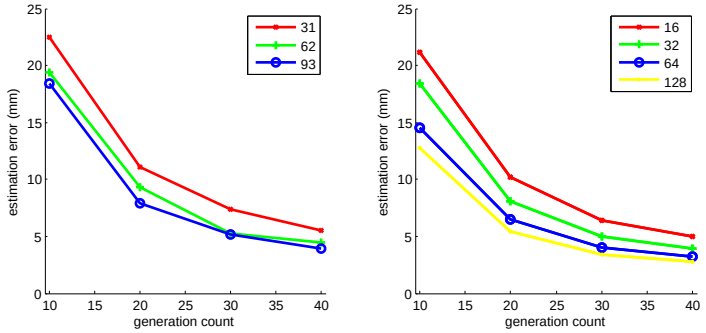


Figure 4.18: Investigation of the PSO parameterization for the presented method (left) and that of Section 4.1 (right).

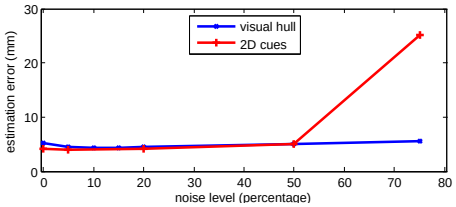


Figure 4.19: Investigation of the effect of noise in the two compared methods. The horizontal axis denotes percentage of corrupted pixels in the synthetic input and the vertical denotes average distance from the ground truth.

We investigated the accuracy of the method for different parameters of PSO. The computational budget of PSO is determined by the number of particles and generations, the product of which yields the total number of objective function computations, as described in Section 3.5.1. We selected a set of values for these two parameters and computed the accuracy of the presented method, as well as that of Section 4.1. In order to quantitatively evaluate the pose estimation accuracy we resort to synthetic datasets with available ground truth. A quantification of the deviation of the estimation from the ground truth is then possible. More specifically, for all the experiments we used a sequence of 245 multiframes depicting a hand that performs simple everyday motions such as palm flipping and finger bending. In order to compare the estimated pose, we adopt a metric along the lines of the description in Section 3.7: 21 landmark points are placed on the virtual hand, 3 on each finger and the remaining 6 on the palm. The average distance of all these landmarks between the estimated and the known pose is measured, and the average over all the frames of the sequence is computed. We perform this experiment multiple times and compute the median of these values as the final error estimation for a given configuration of parameters.

The accuracy of the presented method and of that in Section 4.1 are shown in Figure 4.18. The left plot shows the performance of the system using the visual hull as the main observation cue whereas the right shows the results we got with the method of Section 4.1, with the slight modification described previously. The horizontal axis corresponds to the number of generations used for the experiment whereas the vertical axis denotes the measured pose estimation error as described above. Different graphs

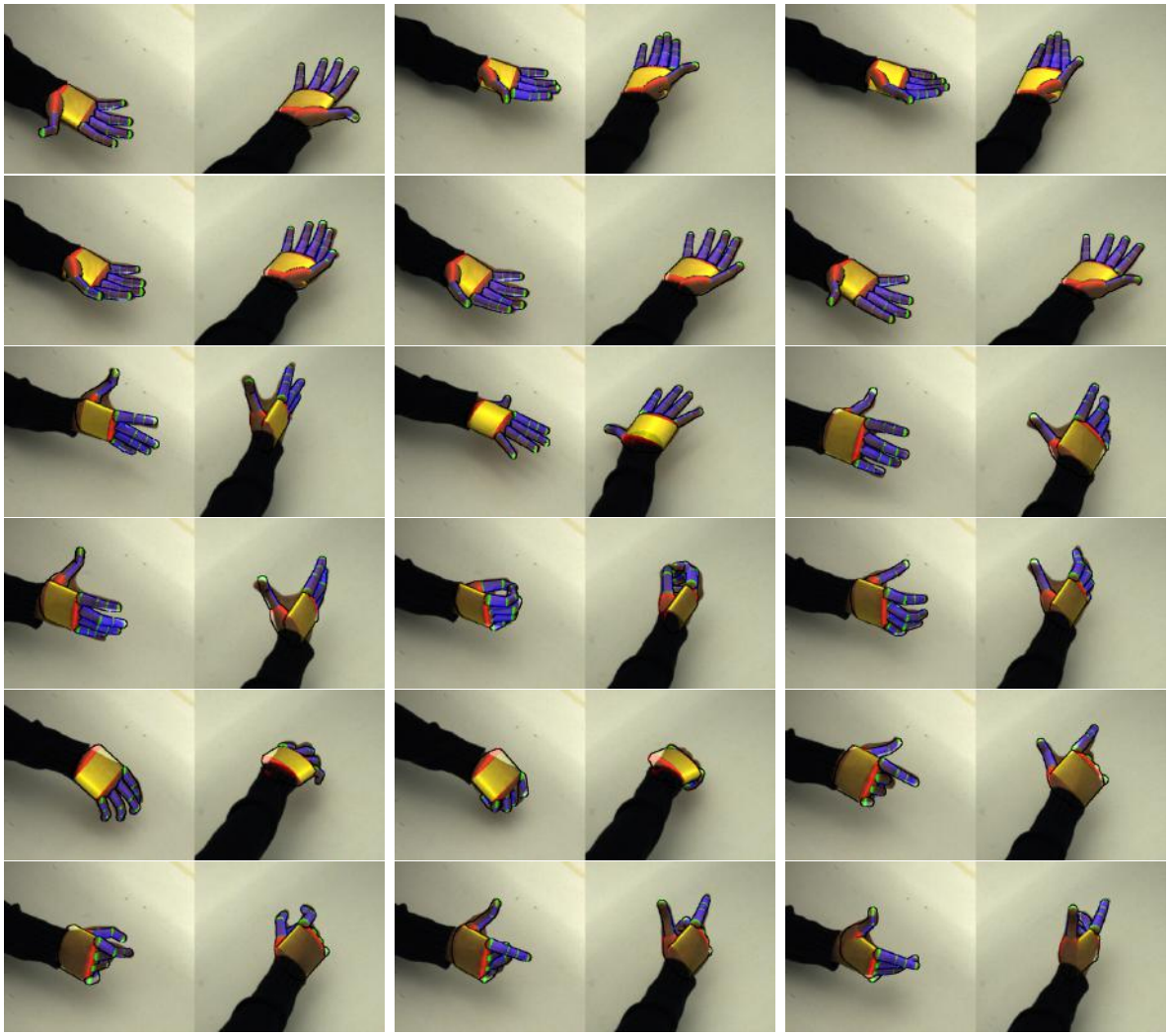


Figure 4.20: Results of the presented method in real-world data. Each pair of images illustrates the same pose from different views.

in the plots correspond to different particle count configurations. Both methods benefit when using more generations or particles, with the number of generations playing a more important role in the performance. Evidently the performance of both methods is comparable. Indicatively, the average error of the presented method for 62 particles and 30 generations is around $5.3mm$ whereas that of Section 4.1 achieves $4.1mm$. In every case the differences are small, getting even smaller as the available budget increases. The performance improvement of both methods for more particles or generations is small. The additional computational budget for this improvement is disproportionate, and so for the remaining experiments we fixed these parameters to $(62, 30)$ for the presented method and to $(64, 30)$ for that of Section 4.1.

We conducted another experiment, investigating the effect of noise on both methods. We employed a noise type similarly to [35]: small disks of randomly selected positions and radii were chosen in the synthetic input images and the pixels in them were flipped. Figure 4.19 illustrates the results of this experiment. Both methods behave comparably

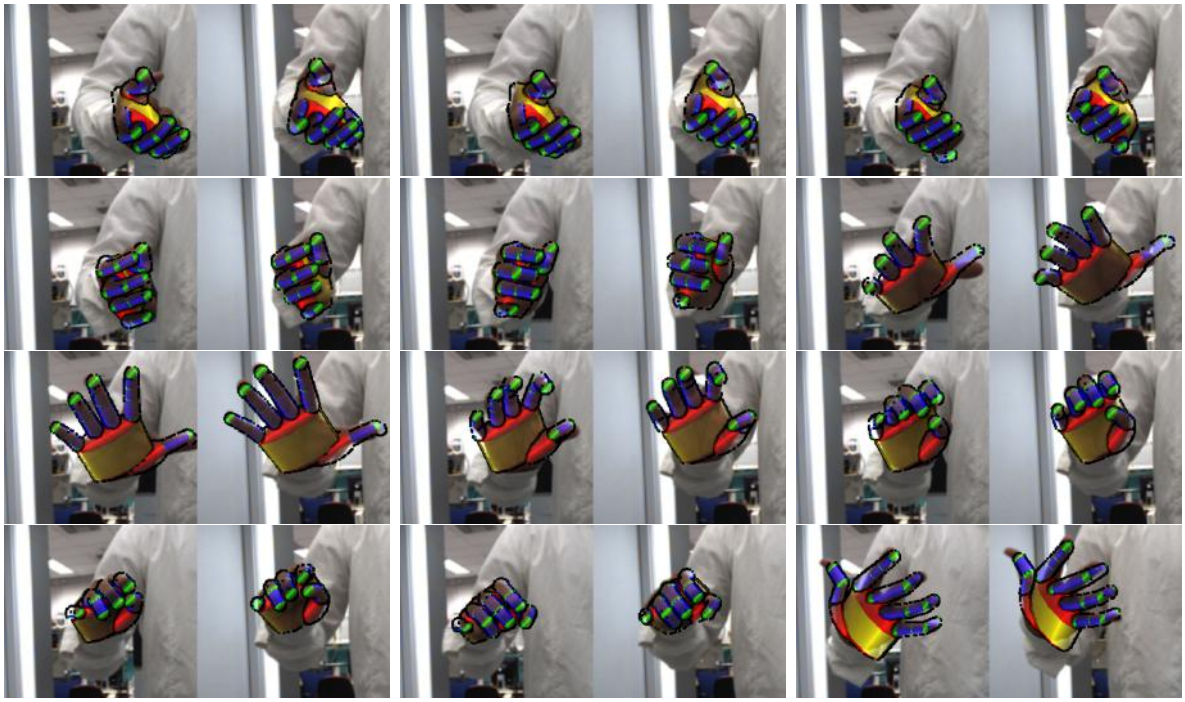


Figure 4.21: Indicative results on data acquired from a stereo baseline system. Each pair illustrates the same pose.

for low and moderate amounts of noise, however the presented method manages to keep track for the large noise level of 75% whereas Section 4.1 fails.

Apart from the quantitative evaluation on synthetic data, we applied the presented method to real-world images. The same multiframe sequence used in Section 4.1 was also used here. The sequence depicts a human hand that performed simple hand motions such as palm flipping, pinching and grasping. The sequence contains in total 390 frames. The hand model was manually initialized for the first frame, and the method successfully tracked all the sequence. Sample results of this track are shown in Figure 4.20. Evidently the fitted hand model closely matches the observations.

We conducted another experiment in data acquired from a narrow baseline stereo camera system. Despite the fact that the recovered visual hulls were elongated, the presented method managed to keep track for several hundred frames while that of Section 4.1 failed. Sample results from this sequence are shown in Figure 4.21.

4.6 Evolutionary Optimization using Quasi-Random Sampling

In the work presented in [62] we shift our focus from the various scenarios of hand tracking to the optimization algorithm used to solve the resulting optimization problems. In all the works presented in the previous Sections, variants of PSO have been used for this task. In this Section we present experimental result we obtained in two different hand tracking scenarios using as optimizer the algorithm described in section 3.5.2. We first

outline the method we used to determine appropriate values for the free parameters of the algorithm. These values were used throughout the experimental evaluation of the presented algorithm, which is compared against PSO. The two different scenarios are the case of single hand tracking that is tackled in Section 4.2 and that of two hands tracking, encountered in Section 4.4.

4.6.1 Meta-optimization

The algorithm outlined in Section 3.5.2 has a number of free parameters, namely the scaling vector s, the contraction coefficients c, the weight parameter a and the number N_T of top scoring positions that contribute to the calculation of h_C. In order to determine appropriate values for these parameters in a systematic way, we resorted to meta-optimization, i.e., the use of an optimization algorithm in order to tune the parameters of another.

To do so, for the case of tracking a single hand, we selected 370 consecutive frames of a hand waving and performing object grasping motions from one of the sequences used in Section 4.2. We tracked this sequence with the method of Section 4.2 and with a very high computational budget, to ensure the highest possible tracking accuracy. We then synthesized the same sequence using our hand model. Having a sequence of synthesized images along with the corresponding hand poses as ground truth, we were able to quantify the performance of a given parametrization of the evolutionary Sobol search algorithm. The quantification was performed with an error metric along the lines of the error metric described in Section 3.7. The meta-optimization for the problem of tracking two interacting hands we followed a similar approach on a sequence showing two hands in strong interaction.

The parametrization of the meta-optimization problem itself, is as follows. We partitioned the scaling vector s in three types of parameters, namely the positional scale, the rotational scale and the scale associated with finger joint angles. Thus, we reduced the 27 or 54 parameters (single/two hands tracking) to just 3. The intention behind not keeping all the different parameters is to avoid over fitting for the specific sequences we used for meta-optimization.

The contraction coefficients c were partitioned with respect to their distance from the root of the kinematic chain, a choice that reflects the way the scoring function E is affected by each of the problem parameters. The intuition is that the parameters describing the position and orientation of the palm must be fixed in order to measure meaningful values when varying the position of, e.g., a fingertip. We thus identified four different levels, starting with position and rotation in the root(palm), the DoFs of the metacarpophalangeal joints at the next level, the proximal interphalangeal (IP) joints at the third level and the distal IP at the last level (bottom to top in Figure 3.1). We did not optimize for N_T, in all experiments we used $N = N_T = 16$ and $G = 25$. The three different scale parameters, the four different contraction coefficients and the weighting parameter a amount to a total of 8 parameters. In order to find optimal values for those parameters we employed PSO.

The separate meta-optimization of the single hand case and the case of two hands, resulted in two different sets for these 8 parameters and in some rather interesting results. For both the single hand and the two hands case, in close agreement with intuition,

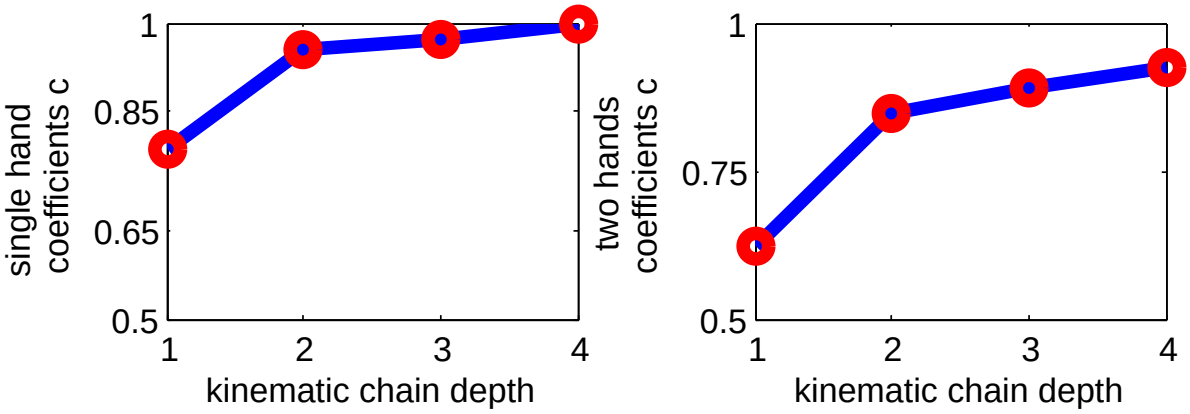


Figure 4.22: The results of meta-optimization yielded an exponential relation between the kinematic chain depth and the contraction coefficients c for both cases, single hand tracking and two hands tracking.

the optimum contraction coefficients decrease exponentially with the distance from the kinematic root (see Figure 4.22). Furthermore, for the single hand case it turned out that the optimal value of a is very close to 0. Thus, the top N_T atoms contribute to the definition of the center h_C with equal weight. This however was not the case for the case of two hands tracking. For that case the optimum value of a was close to 1.6.

4.6.2 Experimental Evaluation

We present experiments that assess the effectiveness of the evolutionary Sobol search approach. The presented method was employed to track the articulation of (a) a single hand and (b) of two strongly interacting hands. The dimensionality of these two problems is 27 DoFs and 54 DoFs, respectively. The results obtained by the presented method are compared quantitatively and qualitatively to those obtained by the methods of Section 4.2 for (a) and of Section 4.4 for (b).

For quasi-random sampling we used the default Matlab implementation of the Sobol sequence of appropriate dimensions (either 27 or 54).

We conducted several experiments to quantitatively assess the performance of the presented method in comparison with the approach presented in Section 4.2. In order to do so, we created synthetic data (i.e., annotated with ground truth) following the description of Section 3.7.

Specifically, a real-world sequence was tracked with good accuracy using the method of Section 4.2 and a large computational budget. This sequence consists of 200 frames depicting a hand performing a variety of motions. It should be stressed that this sequence is different to the one used for the meta-optimization. The resulting track is a sequence of hand poses, closely resembling the observed hand. We used this track to generate synthetic data, i.e. a sequence of synthetic RGB-D images. Since this sequence is produced from a known hand track, we can use that track as the ground truth for that sequence.

Having a sequence with associated ground truth, we compute the distance of a track

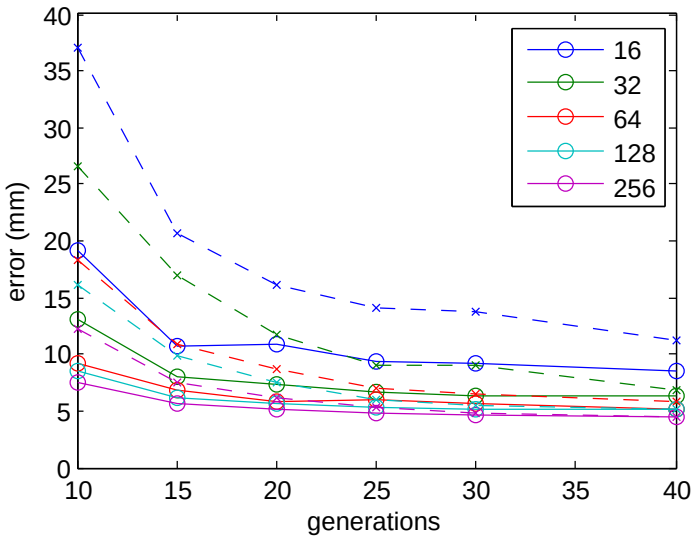


Figure 4.23: The performance of the evolutionary quasi-random search (solid lines) in comparison to that of PSO (dashed) for the problem of single hand tracking and for different particle and generation counts (best viewed in color).

from this ground truth. To do so, we select 21 key points on the hand model we use. The first point is at the root of the kinematic chain inside the palm. Each finger has 4 of the remaining 20 points, starting with one at the base, having one at each of the intermediate joints, and with the last placed at the fingertip. This placement of points can be computed for any given hand pose. Given two hand poses, a ground truth and an estimated one, we can compute the Euclidean distances between such corresponding points. The mean value of all these distances is the error measure we use for a pose estimation of a given frame. For a pose sequence we compute the mean value of such mean distances, resulting in a single error estimate for the whole sequence. Due to its stochastic nature, our algorithm does not perform identically in different runs. Thus, for any given configuration we repeat the experiment 11 times, yielding 11 different mean errors. The median of these 11 values constitutes the error value $\mathcal{D}$ we report in all the experiments below.

As stated in Section 3.5.2, the two parameters determining the computational budget of the algorithm are the number of atoms N and the number of generations G because their product yields the number of objective function evaluations. Particle Swarm Optimization (PSO) is parametrized similarly by the number of its atoms called particles and generations. We assessed the performance of our algorithm in comparison to PSO as a function of these two parameters. The results of this experiment are visualized in Figure 4.23.

It can be verified that the presented evolutionary Sobol search method performs better or equal to PSO for the problem at hand. This is amenable to dual interpretation: we either get more accuracy with the same computational budget or we get the same accuracy with less computational resources. The differences in favor of the presented approach become far more striking in small atom and generation counts. This is quite important

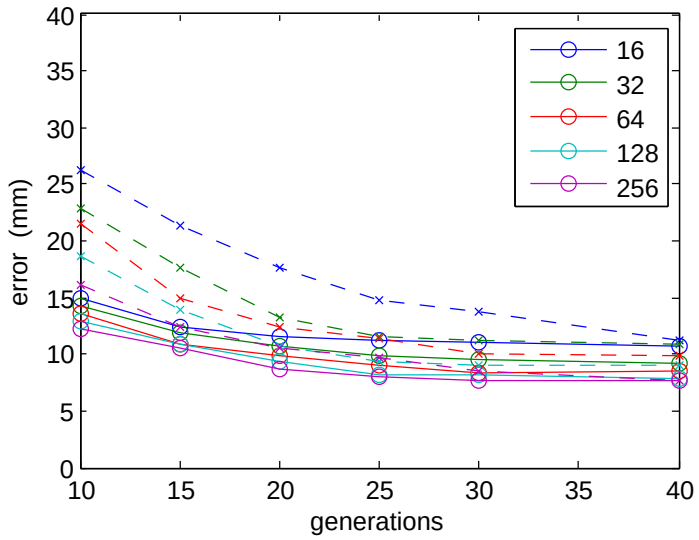


Figure 4.24: The performance of the evolutionary quasi-random search (solid lines) in comparison to that of PSO (dashed) for the problem of tracking two strongly interacting hands and for different particle and generation counts (best viewed in color).

because it means that higher accuracy can be achieved for small computational budgets. As an example, the accuracy obtained by 16 atoms of the presented approach running for 10 generations is equal to the accuracy obtained by 64 particles of PSO for the same number of generations. Given that the objective function E is common for both methods, the evaluation of an atom in our approach is identical to the evaluation of a particle in the method presented in Section 4.2. This means that the presented algorithm achieves a $4\times$ speed-up over the state of the art. As an alternative view of the same results, for the same low budget (16 atoms/particles, 10 generations), the presented approach is almost two times more accurate.

Figure 4.25 shows sample results from the application of the presented algorithm on the synthetic sequence used in this quantitative evaluation.

Similarly to the case of a single hand, we recorded a sequence showing two hands in strong interaction. We employed the evolutionary Sobol search algorithm in a parametric space of 54 DoFs and we experimented with different numbers of atoms and generations. The obtained results are shown in Figure 4.24 in comparison with those obtained by the method of Section 4.4.

The complex interaction between the hands generates even more occlusions, so, it is impossible to resolve ambiguity regarding some poses of the sequence. This, in turn, implies that the lowest achievable error for the case of two hands is higher than that of the single hand case. Nevertheless, the advantages of the presented method are even more prominent in the case of tracking two strongly interacting hands. The lowest budget configuration we tested, $N = 16$ and $G = 10$ was able to achieve an average error of $15mm$ whereas the method of Section 4.4 achieved for the same budget an average error of $26mm$. The presented method achieved for the configuration of $N = 64$ and $G = 25$ an error of 8.9, within $1.5mm$ from the largest budget we tested, namely $(N, G) = (256, 40)$

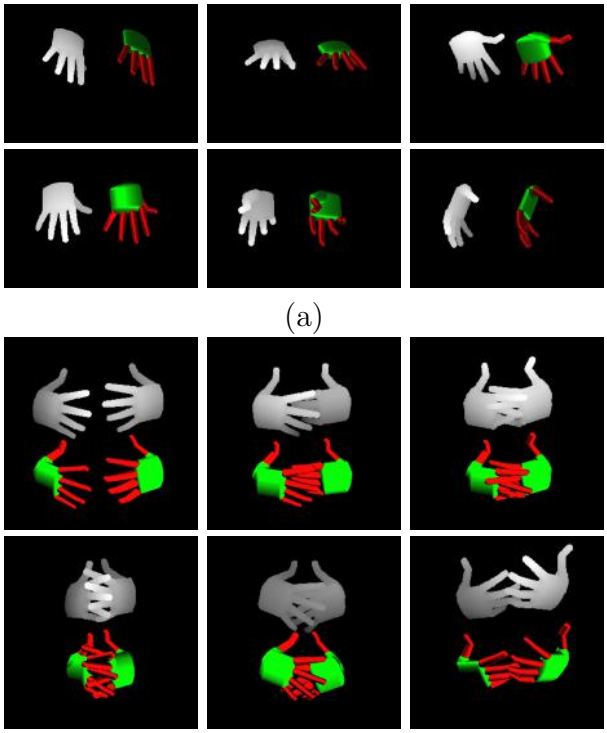


Figure 4.25: Sample results from the application of the presented evolutionary Sobol search method to (a) the single hand tracking and (b) two hands tracking synthetic data sets. For each frame, the rendered depth map together the estimated hand model is shown.

which yielded $7.6mm$ of average error. Thus, for the case of two hands tracking, the presented solution can achieve a speed-up of almost $8\times$, making the tracking of two interacting hands possible in real time.

Figure 4.25 shows sample results from the application of the presented algorithm on the synthetic sequence used in this quantitative evaluation. Finally, Figure 4.26 shows sample results from the application of the presented method on the real world sequences reported in [46, 59] [4].

[4]Results available online at https://www.youtube.com/watch?v=3yvaFuX09xY

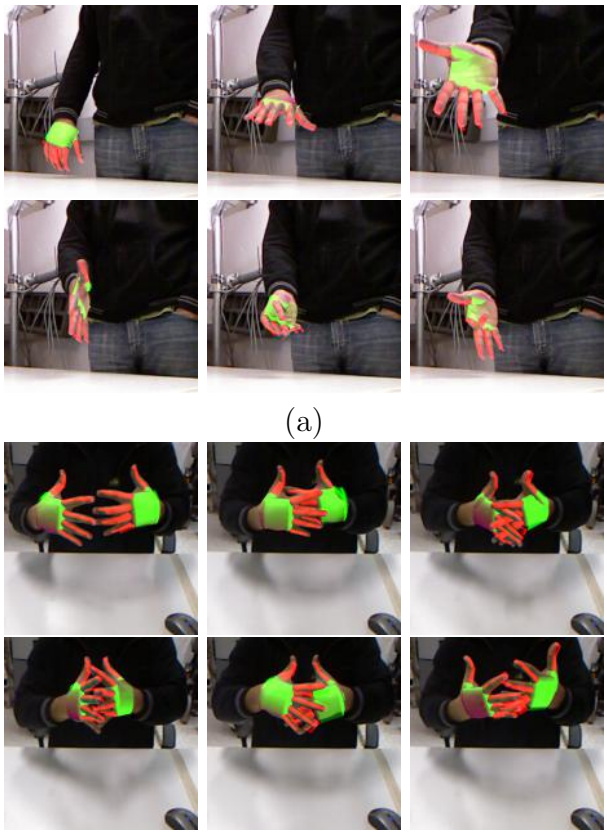


Figure 4.26: Sample results from the application of the presented evolutionary Sobol search method to (a) the single hand tracking and (b) two hands tracking real world sequences reported in Section 4.2 and Section 4.4, respectively.

Chapter 5

Discussion

This thesis presented methods for markerless, model-based recovery and tracking of human hand pose. Examined scenarios include tracking of a single hand in isolation, a hand manipulating an object, as well as tracking two hands in strong interaction. The visual input comes either from a multi-camera network or from an RGB-D sensor.

The most significant contributions of this work are the following:

- Showing that model-based hand tracking is feasible in interactive frame-rates. This was made possible by carefully choosing and designing all the involved computational steps. This in turn allowed for an implementation using GPU acceleration that exploited the resulting computational parallelism.

- Proposing the use of optimization algorithms that robustly tackle the problem. This includes PSO which has been previously employed for the related problem of body pose estimation, and also a novel evolutionary optimization algorithm, specifically tailored for the tackled problem.

- Presenting one of the first model-based methods to tackle the hard problem of hand-object interaction.

- Presenting the first model-based method to tackle strong hand-hand interaction.

5.1 Impact

As discussed in the introductory Section 1, a system with the ability to track the full articulation of a human hand has multiple applications. In this spirit we participated to the ChaLearn Gesture Challenge 2012 [71] using the system described in Section 4.2. We provided its output to a simple classifier that differentiates poses based on the 20-dimensional configuration of the observed fingers. For this submission we were awarded with the First Prize.

Another recognition of the present body of work was the "Maria Michail Manasaki" Bequest Fellowship that was awarded to me for the academic year 2011 – 2012. This fellowship is awarded by the Committee of Graduate Studies of the Computer Science Department of the University of Crete.

Finally, I interned at Microsoft Research in Cambridge, joining the Interactive 3D Technologies group under the supervision of Professor Shahram Izadi. The internship took place during the Spring of 2012, and resulted in a publication [72] and a U.S. patent [73]. The work in [72] describes a wrist-worn device that can accurately and efficiently track the finger articulation in real time. Several applications of this device are also presented.

- Publications
  - As described in detail in Chapter 4, the publications that comprise this thesis are:
    * Oikonomidis, I., Lourakis, M.I., Argyros, A.A.: Evolutionary quasi-random search for hand articulations tracking. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. (2014) [62].
    * Oikonomidis, I., Kyriazis, N., Tzevanidis, K., Argyros, A.A.: Tracking hand articulations: Relying on 3d visual hulls versus relying on multiple 2d cues. In: Ubiquitous Virtual Reality (ISUVR), 2013 International Symposium on, IEEE (2013) 7–10 [54].
    * Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Tracking the articulated motion of two strongly interacting hands. 2012 IEEE Conference on Computer Vision and Pattern Recognition (2012) 1862–1869 [59].
    * Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In: ICCV, IEEE (2011) 2088–2095 [49].
    * Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Efficient model-based 3d tracking of hand articulations using kinect. In: BMVC, Dundee, UK (2011) [46].
    * Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Markerless and efficient 26-dof hand pose recovery. In: ACCV, Springer (2010) 744–757 [48].
  - Additional publications documenting research tightly connected to the work of this thesis:
    * Kyriazis, N., Oikonomidis, I., Argyros, A.: A gpu-powered computational framework for efficient 3d model-based vision. Technical Report 420, FORTH (2011) [74].
    * Kyriazis, N., Argyros, A.: Scalable 3d tracking of multiple interacting objects. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014) 3430–3437 [75].
    * Kyriazis, N., Argyros, A.: Physically plausible 3d scene tracking: the single actor hypothesis. In: Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, IEEE (2013) 9–16 [76].
    * Paliouras, K.: Automatic definition of the objective function for model-based hand tracking. Master's thesis, University of Crete (2014) [77].
    * Douvantzis, P., Oikonomidis, I., Kyriazis, N., Argyros, A.: Dimensionality reduction for efficient single frame hand pose estimation. In: Computer Vision Systems. Springer (2013) 143–152 [78].

    - Song, D., Kyriazis, N., Oikonomidis, I., Papazov, C., Argyros, A., Burschka, D., Kragic, D.: Predicting human intention in visual observations of hand-object interactions. In: ICRA. (2013) [79].
    - Patel, M., Ek, C.H., Kyriazis, N., Argyros, A., Valls Miro, J., Kragic, D.: Language for learning complex human-object interactions. In: ICRA. (2013) [80].

- Awards
  - "Maria Michail Manasaki" Bequest Fellowship [81] of the University of Crete 2011 – 2012.
  - First Prize in ChaLearn Gesture Challenge 2012, sponsored by Microsoft, Redmond, USA, in conjunction with ICPR 2012, Tsukuba, Japan, see [71].

- Highlights[1]
  - Invitation for internship at Microsoft Research in Cambridge. The internship resulted in a publication [72] which was also awarded a U.S. patent [73].
  - More than 510 citations, since 2010, h-index: 7[2].
  - More than 35.000 downloads[3] of 3D single hand tracking software[4].
  - More than 80.000 views[5] of the videos which supplement the thesis-related publications.
  - Contributions to several European projects:
    - GRASP (FP7-215821)
    - RoboHow.cog (FP7-288533)
    - WEARHAP (FP7-ICT-2011-9).

5.2 Future Work

Regarding the continuation of this work, several aspects can benefit from theoretical breakthroughs, as well as further experimentation. Indicatively, promising directions of research involve the visual cues, along with the way they are combined to formulate objective functions, as well as the utilized optimization algorithms.

The employed visual cues are carefully selected so that their computation, synthesis and comparison are all amenable to parallelization. The investigation of other visual cues should follow this direction, enabling real-time or at least interactive frame-rates. One such particular direction is the cue of optical flow, induced by motion due to either actual movement of the observed hand (in consecutive frames) or by viewpoint variation (simultaneous capture from different cameras). Towards this end, within the current

[1] The figures presented were captured in September 2014.

[2] Source: Google Scholar©.

[3] Sources are Google Analytics© and the Apache© server which hosts the web-page of the software.

[4] This is the part of our software implementation which has been made public. This software can be found at `http://cvrlcode.ics.forth.gr/handtracking`.

[5] Source is YouTube©, channels `Antonis Argyros` and `forth3DTracking`.

computational framework it is possible to efficiently generate and evaluate hypothesized motion fields. The evaluation would essentially perform a consistency test on pairs of input images. This approach would rely almost directly on the actual input, avoiding image processing steps that require early decisions such as skin color detection or edge detection.

The optimization algorithms used throughout this work are shown to yield efficient, robust behavior. However further research is worth investing in the selection of optimization algorithms from the relevant literature, or even the design of novel ones, tailored to specific scenarios. Regarding the formulation of the objective function, an idea worth investigating as an alternative to compare occupancy maps such as the skin color maps, is the Jaccard distance [82]. Furthermore, apart from designing the objective function by hand, it is possible to automate the process, yielding potentially more accurate results [77].

Finally, another specific direction worth investigating is that of modeling human hand motion. Section 3.6 describes the way we exploit the temporal continuity assumption: we create a set of candidate poses by adding Gaussian noise to the solution for the previous frame. These poses populate the employed evolutionary optimization algorithms, initializing the search process. An idea that is worth investigating is to form this initial set using more inform types of noise. Specifically, knowledge regarding the human hand motion could be exploited, yielding initial poses that conform to or even predict human hand motion. A scenario that can take advantage of such an approach is that of known types of hand motion: if it is a priori known that the observed hand will perform only a specific set of motions, such a set of gestures, then the discussed approach would be a good candidate to exploit this information. Towards this end, Douvantzis in [78] applies PCA to sets of hand poses, aiming to identify the directions of largest variability within the analyzed data. The resulting linear models capture the natural motion of human hands.

References

[1] Wikipedia: Hand — wikipedia, the free encyclopedia (2012) http://en.wikipedia.org/w/index.php?title=Hand&oldid=369608236 [Online; accessed 3-August-2012].

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

[3] Rehg, J.M., Kanade, T.: Visual tracking of high dof articulated structures: An application to human hand tracking. In: ECCV, Springer-Verlag (1994)

[4] Stenger, B., Mendonça, P., Cipolla, R.: Model-based 3d tracking of an articulated hand. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001 (2001) II–310–II–315

[5] de La Gorce, M., Paragios, N., Fleet, D.J.: Model-based hand tracking with texture, shading and self-occlusions. 2008 IEEE Conference on Computer Vision and Pattern Recognition (2008) 1–8

[6] Bray, M., Koller-Meier, E., Van Gool, L.: Smart particle filtering for 3d hand tracking. Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings. (2004) 675–680

[7] Moeslund, T.B., Hilton, A., Krüger, V.: A survey of advances in vision-based human motion capture and analysis. CVIU **104** (2006) 90–126

[8] Jaeggli, T., Koninckx, T., Van Gool, L.: Model-based sparse 3d reconstruction for online body tracking. Proceedings of SPIE (2005) 226–234

[9] Wang, R.Y., Popović, J.: Real-time hand-tracking with a color glove. ACM Transactions on Graphics **28** (2009) 1

[10] Foxlin, E.: Inertial head-tracker sensor fusion by a complementary separate-bias kalman filter. In: Virtual Reality Annual International Symposium, 1996., Proceedings of the IEEE 1996, IEEE (1996) 185–194

[11] Bachmann, E.R.: Inertial and magnetic tracking of limb segment orientation for inserting humans into synthetic environments. PhD thesis, Monterey, California. Naval Postgraduate School (2000)

[12] Microsoft Corp. Redmond WA: Kinect for xbox 360 (2010)

[13] Sturman, D.: Whole hand input (1992) Ph.D. thesis, MIT.

[14] Hollister, A., Buford, W.L., Myers, L.M., Giurintano, D.J., Novick, A.: The axes of rotation of the thumb carpometacarpal joint. Journal of orthopaedic research : official publication of the Orthopaedic Research Society **10** (1992) 454–60

[15] Kyriazis, N.: A computational framework for observing and understanding the interaction of humans with objects of their environment. PhD thesis, University of Crete (2014)

[16] Sudderth, E., Mandel, M., Freeman, W., Willsky, A.: Visual hand tracking using nonparametric belief propagation. In: Computer Vision and Pattern Recognition Workshop, 2004 Conference on. (2004) 189–189

[17] Albrecht, I., Haber, J., Seidel, H.: Construction and animation of anatomically based human hand models. In: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation, Eurographics Association (2003) 109

[18] Cheng, Y.: Mean shift, mode seeking, and clustering. Pattern Analysis and Machine Intelligence, IEEE Transactions on **17** (1995) 790–799

[19] Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM **24** (1981) 381–395

[20] Zhang, Z.: Iterative point matching for registration of free-form curves and surfaces. International journal of computer vision **13** (1994) 119–152

[21] Ristic, B., Arulampalam, S., Gordon, N.: Beyond the Kalman filter: Particle filters for tracking applications. Artech house (2004)

[22] Fletcher, R.: Practical Methods of Optimization. John Wiley\ & Sons, Inc. (1987)

[23] Rios, L.M., Sahinidis, N.V.: Derivative-free optimization: A review of algorithms and comparison of software implementations. Journal of Global Optimization **56** (2013) 1247–1293

[24] Hansen, N.: Black-box optimization benchmarking (bbob) 2013 (2013)

[25] Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments - a survey. IEEE Trans. on Evolutionary Computation **9** (2005) 303

[26] De Jong, K.A.: Evolutionary computation: a unified approach. Volume 262041944. MIT press Cambridge (2006)

[27] Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: International Conference on Neural Networks. Volume 4., IEEE (1995) 1942–1948

[28] Wu, Y., Huang, T.: View-independent recognition of hand postures. In: IEEE Conference on Computer Vision and Pattern Recognition, 2000. Proceedings. Volume 2., IEEE Comput. Soc (2000) 88–94

[29] Rosales, R., Athitsos, V., Sigal, L., Sclaroff, S.: 3d hand pose reconstruction using specialized mappings. Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001 (2001) 378–385

[30] Shimada, N., Kimura, K., Shirai, Y.: Real-time 3d hand posture estimation based on 2d appearance retrieval using monocular camera. In: ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems. (2001) 23–30

[31] Athitsos, V., Sclaroff, S.: Estimating 3d hand pose from a cluttered image. In: CVPR, IEEE (2003) II–432–9

[32] Stenger, B.: Template-based hand pose recognition using multiple cues. Computer Vision—ACCV 2006 (2006) 551–560

[33] Ike, T., Kishikawa, N., Stenger, B.: A real-time hand gesture interface implemented on a multi-core processor. Machine Vision and Applications (2008)

[34] Kjellstrom, H., Romero, J., Martinez, D., Kragic, D.: Simultaneous visual recognition of manipulation actions and manipulated objects. In: ECCV. (2008)

[35] Romero, J., Kjellstrom, H., Kragic, D.: Monocular real-time 3d articulated hand pose estimation. 2009 9th IEEE-RAS International Conference on Humanoid Robots (2009) 87–92

[36] Romero, J., Kjellström, H., Kragic, D.: Hands in action: Real-time 3d reconstruction of hands in interaction with objects. In: IEEE International Conference on Robotics and Automation. (2010) 3–8

[37] Wang, R., Paris, S., Popović, J.: 6d hands: markerless hand-tracking for computer aided design. In: Proceedings of the 24th annual ACM symposium on User interface software and technology, ACM (2011) 549–558

[38] Keskin, C., Kirac, F., Kara, Y.E., Akarun, L.: Real time hand pose estimation using depth sensors. Consumer Depth Cameras for Computer Vision, ICCV Workshop (2011) 1228–1234

[39] Xu, C., Cheng, L.: Efficient hand pose estimation from a single depth image. In: Computer Vision (ICCV), 2013 IEEE International Conference on, IEEE (2013) 3456–3462

[40] Tang, D., Chang, H.J., Tejani, A., Kim, T.K.: Latent regression forest: Structured estimation of 3d articulated hand posture. In: Computer Vision and Pattern Recognition, 2014. CVPR 2014. IEEE Computer Society Conference on, IEEE (2014)

[41] Tompson, J., Stein, M., LeCun, Y., Perlin, K.: Real-time continuous pose recovery of human hands using convolutional networks. In: SIGGRAPH '14. (2014)

[42] Hamer, H., Schindler, K., Koller-Meier, E., Van Gool, L.: Tracking a hand manipulating an object. In: IEEE International Conference on Computer Vision. (2009)

[43] de La Gorce, M., Fleet, D., Paragios, N.: Model-based 3d hand pose estimation from monocular video. IEEE Transactions on Pattern Analysis and Machine Intelligence (2011) 1–15

[44] Ballan, L., Taneja, A., Gall, J., Van Gool, L., Pollefeys, M.: Motion capture of hands in action using discriminative salient points. In: Computer Vision–ECCV 2012. Springer (2012) 640–653

[45] Qian, C., Sun, X., Wei, Y., Tang, X., Sun, J.: Realtime and robust hand tracking from depth. In: Computer Vision and Pattern Recognition, 2014. CVPR 2014. IEEE Computer Society Conference on, IEEE (2014)

[46] Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Efficient model-based 3d tracking of hand articulations using kinect. In: BMVC, Dundee, UK (2011)

[47] Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: CVPR, IEEE (2011)

[48] Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Markerless and efficient 26-dof hand pose recovery. In: ACCV, Springer (2010) 744–757

[49] Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In: ICCV, IEEE (2011) 2088–2095

[50] Diebel, J.: Representing attitude: Euler angles, unit quaternions, and rotation vectors. Matrix **58** (2006) 15–16

[51] Bradski, G.: The opencv library. Dr. Dobb's Journal of Software Tools (2000)

[52] Freeman, W.T., Roth, M.: Orientation histograms for hand gesture recognition. In: International Workshop on Automatic Face and Gesture Recognition. Volume 12. (1995) 296–301

[53] Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Volume 1., IEEE (2005) 886–893

[54] Oikonomidis, I., Kyriazis, N., Tzevanidis, K., Argyros, A.A.: Tracking hand articulations: Relying on 3d visual hulls versus relying on multiple 2d cues. In: Ubiquitous Virtual Reality (ISUVR), 2013 International Symposium on, IEEE (2013) 7–10

[55] Canny, J.: A computational approach to edge detection. Pattern Analysis and Machine Intelligence, IEEE Transactions on **8** (1986) 679–698

[56] Solomon, C., Breckon, T.: Fundamentals of digital image processing: A practical approach with examples in Matlab. John Wiley & Sons (2011)

[57] Argyros, A., Lourakis, M.: Real-time tracking of multiple skin-colored objects with a possibly moving camera. In: ECCV. (2004)

[58] van Rijsbergen, C.J.: Information Retrieval. Butterworth (1979)

[59] Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Tracking the articulated motion of two strongly interacting hands. 2012 IEEE Conference on Computer Vision and Pattern Recognition (2012) 1862–1869

[60] Laurentini, A.: The visual hull concept for silhouette-based image understanding. IEEE Transactions on Pattern Analysis and Machine Intelligence (1994) 150–162

[61] Tzevanidis, K., Zabulis, X., Sarmis, T., Koutlemanis, P., Kyriazis, N., Argyros, A.A.: From multiple views to textured 3d meshes: a gpu-powered approach. In: ECCV Workshops. (2010) 5–11

[62] Oikonomidis, I., Lourakis, M.I., Argyros, A.A.: Evolutionary quasi-random search for hand articulations tracking. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition. (2014)

[63] Kennedy, J., Eberhart, R., Shi, Y.: Swarm Intelligence. Morgan Kaufmann Publishers (2001)

[64] Angeline, P.: Evolutionary optimization versus particle swarm optimization: Philosophy and performance differences. Evolutionary Programming VII, LNCS **1447** (1998) 601–610

[65] White, B., Shaw, M.: Automatically tuning background subtraction parameters using particle swarm optimization. In: IEEE ICME. (2007)

[66] Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Evolutionary Computation **6** (2002) 58–73

[67] Yasuda, T., Ohkura, K., Matsumura, Y.: Extended pso with partial randomization for large scale multimodal problems. In: World Automation Congress (WAC), 2010, IEEE (2010) 1–6

[68] Sobol, I.M.: On the distribution of points in a cube and the approximate evaluation of integrals. USSR Computational Mathematics and Mathematical Physics **7** (1967) 86–112

[69] Niederreiter, H.: Constructions of (t, m, s)-nets and (t, s)-sequences. Finite Fields and Their Applications **11** (2005) 578–600

[70] Smith, R.: Open dynamics engine, http://www.ode.org/ (2006)

[71] IEEE: Demonstration competition at ICPR 2012 (2012) http://gesture.chalearn.org/dissemination/icpr2012/demonstration-competition [Online; accessed June-2014].

[72] Kim, D., Hilliges, O., Izadi, S., Butler, A.D., Chen, J., Oikonomidis, I., Olivier, P.: Digits: freehand 3d interactions anywhere using a wrist-worn gloveless sensor. In: Proceedings of the 25th annual ACM symposium on User interface software and technology, ACM (2012) 167–176

[73] Kim, D., Izadi, S., Hilliges, O., Butler, D., Hodges, S., Olivier, P., Chen, J., Oikonomidis, I.: Wearable sensor for tracking articulated body-parts (2014) US Patent App. 13/644,701.

[74] Kyriazis, N., Oikonomidis, I., Argyros, A.: A gpu-powered computational framework for efficient 3d model-based vision. Technical Report 420, FORTH (2011)

[75] Kyriazis, N., Argyros, A.: Scalable 3d tracking of multiple interacting objects. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2014) 3430–3437

[76] Kyriazis, N., Argyros, A.: Physically plausible 3d scene tracking: the single actor hypothesis. In: Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on, IEEE (2013) 9–16

[77] Paliouras, K.: Automatic definition of the objective function for model-based hand tracking. Master's thesis, University of Crete (2014)

[78] Douvantzis, P., Oikonomidis, I., Kyriazis, N., Argyros, A.: Dimensionality reduction for efficient single frame hand pose estimation. In: Computer Vision Systems. Springer (2013) 143–152

[79] Song, D., Kyriazis, N., Oikonomidis, I., Papazov, C., Argyros, A., Burschka, D., Kragic, D.: Predicting human intention in visual observations of hand-object interactions. In: ICRA. (2013)

[80] Patel, M., Ek, C.H., Kyriazis, N., Argyros, A., Valls Miro, J., Kragic, D.: Language for learning complex human-object interactions. In: ICRA. (2013)

[81] University of Crete: The "Maria Michail Manasaki" Bequest Fellowships (2012) http://www.csd.uoc.gr/en/studies/Scholarships/manasakis.html [Online; accessed June-2014].

[82] Tan, P.N., Steinbach, M., Kumar, V.: Introduction to data mining. (2005)