ACCURATE HUMAN MOTION CAPTURE AND MODELING USING LOW-COST SENSORS

A Dissertation

by

PEIZHAO ZHANG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Jinxiang Chai
Committee Members,	John Keyser
	Scott Schaefer
	Jianhua Huang
Head of Department,	Dilma Da Silva

May 2017

Major Subject: Computer Science

ABSTRACT

Motion capture technologies, especially those combined with multiple kinds of sensory technologies to capture both kinematic and dynamic information, are widely used in a variety of fields such as biomechanics, robotics, and health. However, many existing systems suffer from limitations of being intrusive, restrictive, and expensive.

This dissertation explores two aspects of motion capture systems that are low-cost, non-intrusive, high-accuracy, and easy to use for common users, including both full-body kinematics and dynamics capture, and user-specific hand modeling.

More specifically, we present a new method for full-body motion capture that uses input data captured by three depth cameras and a pair of pressure-sensing shoes. Our system is appealing because it is fully automatic and can accurately reconstruct both full-body kinematic and dynamic data. We introduce a highly accurate tracking process that automatically reconstructs 3D skeletal poses using depth data, foot pressure data, and detailed full-body geometry. We also develop an efficient physics-based motion reconstruction algorithm for solving internal joint torques and contact forces based on contact pressure information and 3D poses from the kinematic tracking process.

In addition, we present a novel low-dimensional parametric model for 3D hand modeling and synthesis. We construct a low-dimensional parametric model to compactly represent hand shape variations across individuals and enhance it by adding Linear Blend Skinning (LBS) for pose deformation. We also introduce an efficient iterative approach to learn the parametric model from a large unaligned scan database. Our model is compact, expressive, and produces a natural-looking LBS model for pose deformation, which allows for a variety of applications ranging from user-specific hand modeling to skinning weights transfer and model-based hand tracking.

ACKNOWLEDGMENTS

First and foremost, I would like to express my deep gratitude to my advisor, Professor Jinxiang Chai, not only for the great opportunity he gave me to learn and perform research at Texas A&M University, but also for his invaluable guidance and continuous support throughout my Ph.D. study. His advice and detailed comments help me to learn a lot on how to analyze, prioritize and solve problems. Without his assistant, this dissertation would not have been possible. In addition, I would also like to thank my committee members, Professor John Keyser, Professor Scott Schaefer and Professor Jianhua Huang, for their valuable time and precious advice.

I would also like to thank Professor Karen C. Liu and Kristin Siu at Georgia Institute of Technology for the collaboration and discussion on the full-body motion capture project.

Furthermore, I would extend my gratitude to all my friends and colleagues in the lab, Fuhao Shi, Jianjie Zhang, Peihong Guo, Xiaolin Wei, Jianyuan Min, Yen-lin Chen, and Hui Lou for their generous help, support and the laughter they brought into my work and life. It has been a wonderful experience working with them during the past few years.

Finally, I would like to take the opportunity to thank my wife, Chun Wu. She is always devoting her love, taking care of my life and supporting me all the time. I would also like to express my gratitude for my parents and sister. I would never have gone this far without their unconditional love, support, and encouragement.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supervised by a dissertation committee consisting of Professor Jinxiang Chai, Professor John Keyser and Professor Scott Schaefer of the Department of Computer Science and Engineering and Professor Jianhua Huang of the Department of Statistics.

The physics-based motion optimization part (Section II.5) and relevant experiments in Chapter II were conducted by Jianjie Zhang. The pressure-sensing shoes were designed and developed by Peter Presti. The joint torque patterns captured by the Vicon system and force plates for evaluation in Section II.7.5 were provided by Xiaolei Lv.

All other work conducted for the dissertation was completed by the student, under the advisement of Professor Jinxiang Chai of the Department of Computer Science and Engineering.

Funding Sources

This work is partially supported by the National Science Foundation under Grants No. IIS-1055046, IIS-1065384 and IIS-1064983, and NIH R01 NS069655-05.

Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the National Science Foundation.

TABLE OF CONTENTS

Page

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Human motion capture, including full-body movement and hand articulations, are widely used in a variety of fields such as filmmaking (e.g., *Avatar*, *The Avengers*), video game development (e.g., *NBA 2K16*, *FIFA 16*), virtual reality (e.g., *Oculus rift + touch*) and human-computer interaction (e.g., *Kinect*, *LeapMotion*). By combining with multiple kinds of sensory technologies to capture both kinematic and dynamic information, they also play important roles in fields such as biomechanics, kinesiology, robotics, and health. For example, optical motion capture system [3] with ground reaction force plates could be used for clinical gait analysis to help to identify and make treatment decisions for people with posture or walking-related problems.

Although kinematics and dynamics motion capture systems play crucial roles in various fields, many existing systems suffer from several limitations and are not suitable for common users. Firstly, they are intrusive and inconvenient to use. For example, optical motion capture systems require users to wear special suits with markers, and cumbersome devices are needed for magnetic [5] and inertial [6] systems. Secondly, these systems are expensive since the specialized hardware aims at capturing high-fidelity data. For example, *Vicon* system with 12 cameras to cover a room costs more than $\$120,000$. Additionally, the capture volume is restrictive. Capturing dynamic data usually involves using unmovable force platforms, which limits the performance volume to a highly restricted area.

Advancements in hardware technology have permitted sensory devices to become smaller and cheaper. On the one hand, low-cost depth cameras and practical markerless motion capture systems such as *Microsoft Kinect* [7] and *Intel Realsense* [8] have become

available to common users. These systems are easy to set up and non-intrusive because no special suits, markers or other devices are required. However, these systems are still limited to the kinds of motions that could be performed. For example, *Kinect* may fail to produce good results when large occlusions such as turning the body around happens. In addition, these systems only focus on capturing kinematic motion. Deriving contact and location information from these systems are difficult due to noisy, incomplete, or insufficient data. On the other hand, the use of small sensors allows us to collect various types of data about human subjects unobtrusively. For example, pressure sensors embedded in the shoes collect pressure information applied on the shoes. But these sensors alone are not enough to capture dynamic data for the human body due to the lack of kinematic information.

By combining small and affordable sensors with low-cost depth sensors, we have a powerful amount of information that allows us to capture both kinematic and dynamic data in a non-intrusive way. One of our goals in this dissertation would be building such a system that is low-cost, non-intrusive, high-accuracy, and easy to use for common users.

In addition to full-body motion capture, the use of low-cost depth sensors makes capturing human hand movement practical and popular nowadays, especially for human-computer interaction. To get high-accuracy motion capture data, one important step in many capture systems is to build a subject-specific model to represent the hand shape of the user, since shapes vary hugely across subjects. One efficient way for 3D hand modeling is to use a skinned mesh model with Linear Blend Skinning (LBS) for pose deformation. But such a model has a large number of degrees of freedom to represent subjects in different shape, which is highly redundant and easy to get unnatural-looking models.

To make the subject modeling step intuitive and easy to use for common users, another goal of this dissertation is to build a system that allows novel users to generate a user-specific hand model rapidly and efficiently using non-intrusive inputs such as depth images

and semantic measurements.

In this dissertation, we explore two aspects of motion capture systems to achieve the aforementioned goals, including full-body kinematics and dynamics capture, and user-specific hand shape modeling using low-cost sensors. Our full-body motion capture system allows novel users to capture both kinematics information (3D skeletal poses) and dynamics data (contact forces and internal joint torques) without using intrusive motion capture systems and unmovable force platforms, largely extends the potential applications of the system. Our user-specific hand shape modeling system allows common users to build accurate shape models rapidly and easily that could be used for model-based pose tracking as well as other applications.

For full-body motion capture, we present a new method to capture both kinematic and dynamic data that uses input data from three depth cameras and a pair of pressure-sensing shoes. We choose depth sensors and foot pressure sensors because they are non-intrusive. More importantly, they are complementary to each other as they capture fundamentally different aspects of the motion. The pressure-sensing shoes provide high-resolution contact timing and location information that is difficult to derive automatically from computer vision algorithms. On the other hand, depth data from the depth cameras provide kinematic information that can filter out noise in the pressure sensors and provide global position and orientation necessary to estimate dynamic quantities such as the center of pressure. The result is that our system is easy to set up and can be used to acquire motions difficult to capture in restrictive lab settings such as highly dynamic motions that require a large amount of space.

We first introduce a novel tracking process that automatically reconstructs 3D skeletal poses using input data captured by three Kinect cameras and wearable pressure sensors. We formulate the problem in an optimization framework and incrementally update 3D skeletal poses with observed depth data and pressure data via iterative linear solvers. The

Figure I.1: Our system automatically and accurately reconstructs full-body kinematic and dynamic data using input data captured by three depth cameras and a pair of pressure-sensing shoes. (top) reference image data; (bottom) the reconstructed full-body poses and contact forces (red arrows) and torsional torques (yellow arrows) applied at the center of pressure.

system is highly accurate because we integrate depth data from multiple depth cameras, foot pressure data, detailed full-body geometry, and environmental contact constraints into a unified framework. In addition, we develop an efficient physics-based motion reconstruction algorithm for solving internal joint torques and contact forces in the quadratic programming framework. During reconstruction, we leverage Newtonian physics, friction cone constraints, contact pressure information, and 3D kinematic poses obtained from the kinematic tracking process to reconstruct full-body dynamic data.

We demonstrate our system by capturing high-quality kinematic and dynamic data for a wide range of human movements (Figure I.1). We assess the quality of reconstructed motions by comparing them with ground truth data simultaneously captured with a full marker set in a commercial motion capture system [3]. We show the superior performance of our system by comparing against alternative methods. In addition, we evaluate the importance of each key component of our 3D motion capture system by dropping off each component. Finally, we validate the quality of reconstructed dynamic data by comparing

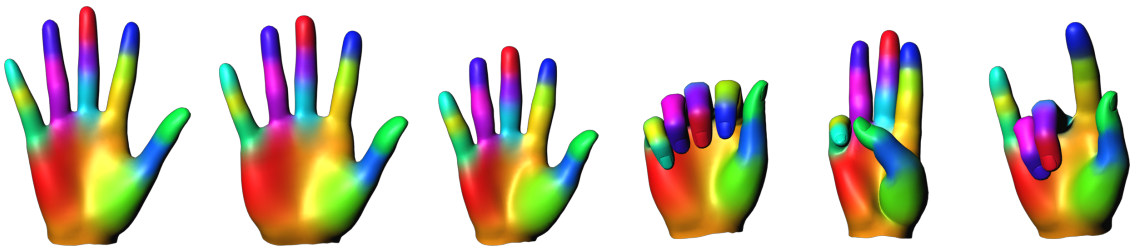


Figure I.2: Random sampled skinned mesh models color coded by skinning weights.

joint torque patterns obtained by our system against those from a *Vicon* system and force plates.

For subject modeling, we present a novel low-dimensional parametric model for 3D hand modeling and synthesis. Our core idea is to construct a low-dimensional parametric model to compactly represent hand shape variations across individuals and enhance it by adding Linear Blend Skinning (LBS) for pose deformation. We also introduce an efficient iterative approach to learn the parametric model from a large unaligned scan database. Our model is compact and expressive since its representation allows shape variations across different subjects, but is specific enough not to allow arbitrary variations that are not similar to those seen in the database. With this parametric model, we could randomly sample the parameters to generate an infinite number of natural-looking hand models in different shapes and under different poses (Figure I.2). Furthermore, we could choose the parameters so that the model would match various forms of user input. We demonstrate the power and effectiveness of our parametric model by exploiting a variety of applications that range from user-specific hand modeling using various forms of input constraints including depth images, partial scans, color images, and semantic measurements, to skinning weights transfer and model-based hand tracking. We assess the accuracy and effectiveness of our model via cross validation. We validate our model by evaluating the key components of our model.

I.1 Contributions

The primary contribution of this dissertation is to present novel algorithms and models for two aspects of motion capture systems that are low-cost, non-intrusive, high-accuracy, and easy to use for common users, including:

- A fully automatic motion capture system for full-body kinematics and dynamics capture using three depth sensors and a pair of pressure-sensing shoes.
- A low-dimensional parametric model for 3D hand modeling and synthesis.

Our full-body motion capture system is made possible by a number of technical contributions, including:

- The first system to use multiple cameras and a pair of pressure-sensing shoes for accurately reconstructing both full-body kinematics and dynamics.
- The use of a signed distance field for full-body kinematic tracking.
- The idea of incorporating depth data, pressure data, full-body geometry and environmental contact constraints into a unified framework for kinematic pose tracking.

Our 3D hand modeling system makes the following contributions:

- A compact parametric hand model that accurately models geometric variations in hand shapes and poses across individuals.
- An iterative optimization approach that efficiently learns the parametric hand model from a large set of unaligned hand scan database.
- A wide range of applications, including 3D hand modeling using depth images, incomplete scans, semantic constraints, and color images, as well as skinning weights transfer, and model-based 3D hand tracking.

In the next chapter, we will describe how to reconstruct both kinematic and dynamic data using low-cost sensors, including kinematic pose tracking, physics-based motion reconstruction, and full-body shape representation and reconstruction. We then describe our 3D hand modeling system, including how to represent the model, how to construct the low-dimensional parametric model from an unaligned scan database, and its applications.

CHAPTER II

LEVERAGING DEPTH CAMERAS AND WEARABLE PRESSURE SENSORS FOR FULL-BODY KINEMATICS AND DYNAMICS CAPTURE *

Motion capture technologies have revolutionized computer animation over the past decade. With detailed motion data and editing algorithms, we can directly transfer the expressive performance of a real person to a virtual character, interpolate existing data to produce new sequences, or compose simple motion clips to create a repertoire of motor skills. With appropriate computational models and machine learning algorithms, we can use motion data to create more accurate and realistic models than those based on physics laws and principles alone. Additionally, kinematic and dynamic information of human motion are extremely valuable to a wide variety of fields such as biomechanics, robotics, and health, where there continues to be a growing need for efficient, high-quality, and affordable motion capture systems.

Yet despite decades of research in computer graphics and a plethora of approaches, many existing motion capture systems still suffer from several limitations. Firstly, many systems require the subject to wear cumbersome devices or limit the subject's motion to a restricted area. Additionally, in order to capture high-fidelity data, the specialized hardware for these systems is often expensive and requires extensive training to operate. Finally, current motion capture technology specializes in capturing only kinematic information of the movement rather than its underlying dynamics. Combining multiple kinds of sensory technologies in order to acquire this dynamic information is common prac-

*Reprinted with permission from "Leveraging Depth Cameras and Wearable Pressure Sensors for Full-body Kinematics and Dynamics Capture" by Peizhao Zhang, Kristin Siu, Jianjie Zhang, C. Karen Liu, and Jinxiang Chai, 2014. ACM Trans. Graph, 33, 221:1–221:14, DOI 10.1145/2661229.2661286, Copyright 2014 by ACM, Inc.

tice in the fields of biomechanics and kinesiology. However, this data acquisition process typically involves expensive and intrusive optical motion capture systems and unmovable force platforms that can only be operated in a highly restricted environment.

Advancements in hardware technology have permitted sensory devices to become smaller and cheaper. With the advent of affordable depth cameras, image-based motion capture systems hold promise but are still limited in the kinds of motions that can be captured. In order to find a solution to these shortcomings, we are inspired by trends in health technology where the ubiquity of small sensors have made it possible to collect various types of data about human subjects unobtrusively. If combining small and affordable sensors has the potential to provide a powerful amount of information, the question then becomes: What is the ideal set of basic sensors required to capture both high-quality kinematic and dynamic data?

Our answer is a system consisting of a pair of low-cost, non-intrusive pressure-sensing shoes and three *Microsoft Kinect* cameras. Our solution leverages the fact that both of these two sensory technologies are inexpensive and non-intrusive. Additionally, they are complementary to each other as they capture fundamentally different aspects of the motions. The pressure-sensing shoes provide high-resolution contact timing and location information that is difficult to derive automatically from computer vision algorithms. On the other hand, depth data from the *Kinect* cameras provide kinematic information that can filter out noise in the pressure sensors and provide global position and orientation necessary to estimate dynamic quantities such as the center of pressure. The result is that our system is easy to set up and can be used to acquire motions difficult to capture in restrictive lab settings such as highly dynamics motions that require a large amount of space.

Our unified system integrates depth data from multiple cameras, foot pressure data, detailed full-body geometry, and environmental contact constraints. We first introduce a novel tracking process that automatically reconstructs 3D skeletal poses using input data

captured by the *Kinect* cameras and pressure sensors. We formulate this problem in an optimization framework and incrementally update 3D skeletal poses with observed input data via iterative system solvers. In addition, we develop an efficient physics-based motion optimization algorithm to reconstruct full-body dynamics data, internal joint torques, and contact forces across the entire sequence. We leverage Newtonian physics, contact pressure information, and 3D kinematic poses obtained from the kinematic pose tracking process in a quadratic programming framework. By accounting for physical constraints and observed depth and pressure data simultaneously, we are ultimately able to compute both kinematic and dynamic variables more accurately.

We demonstrate our system by capturing high-quality kinematics and dynamics data for a wide range of human movements (Figure I.1). We assess the quality of reconstructed motions by comparing them with ground truth data simultaneously captured with a full marker set in a commercial motion capture system [3]. We show the superior performance of our system by comparing against alternative methods including Wei et al. [2], *Kinect* [7] and full-body pose tracking using Iterative Closest Point (ICP) method (*e.g.*, [9, 10]). In addition, we evaluate the importance of each key component of our 3D motion capture system by dropping off each component in evaluation. Finally, we validate the quality of reconstructed dynamics data by comparing joint torque patterns obtained by our system against those from a *Vicon* system and force plates.

In summary, our proposed system makes the following contributions:

- The first system to use multiple cameras and a pair of pressure-sensing shoes for accurately reconstructing both full-body kinematics and dynamics.
- The use of a signed distance field for full-body kinematic tracking.
- The idea of incorporating depth data, pressure data, full-body geometry, and environmental contact constraints into a unified framework for kinematic pose tracking.

II.1 Background

Various technologies have been proposed for acquiring human body movement. We use a combination of low-cost, portable devices to design a new motion capture system that automatically acquires and reconstructs full-body poses, joint torques, and contact forces all at once. To our knowledge, no single existing motion capture technology can achieve this goal. In the following section, we compare our system with existing motion capture systems popular for both research and commercial use.

One appealing solution for full-body motion capture is to use commercially available motion capture systems including marker-based motion capture (*e.g.*, [3]), inertial motion capture (*e.g.*, [6]), and magnetic motion capture (*e.g.*, [5]). These methods can capture full-body kinematic motion data with high accuracy and reliability. However, they are often cumbersome, expensive and intrusive. Our entire system does not require the subject to wear special suits, sensors, or markers except for a pair of normal shoes. This allows us to capture performance or activities such as sports in their most natural states. More importantly, we aim for much cheaper and more accurate motion with both kinematic and dynamic information.

Image-based systems, which track 3D human poses using conventional intensity/color cameras (for more details we refer the reader to [11]), offer an appealing alternative to full-body motion capture because they require no markers, no sensors, no special suits, and thereby do not impede the subject's ability to perform the motion. One notable solution is to perform sequential pose tracking based on 2D image measurements (e.g., [12]), which initializes 3D human poses at the starting frame and sequentially updates 3D poses by minimizing the inconsistency between the hypothesized poses and observed measurements. This approach, however, is often vulnerable to occlusions, cloth deformation, illumination changes, and a lack of discernible features on the human body because 2D

image measurements are often not sufficient to determine high-dimensional 3D human movement.

One way to reduce the reconstruction ambiguity is to use multiple color cameras to capture full-body performances [13, 14]. Another possibility is to learn kinematic motion priors from pre-captured motion data using generative approaches (*e.g.*, [15, 16]) or discriminative models (*e.g.*, [17, 18]). While the use of learned kinematic models clearly reduces ambiguities in pose estimation and tracking, the 3D motions estimated by these methods are often physically implausible, therefore displaying unpleasant visual artifacts such as out-of-plane rotation, foot sliding, ground penetration, and motion jerkiness.

Our work is closely related to a rapidly growing body of recent literature on 3D pose tracking and detection with depth data (*e.g.*, [19, 20, 21, 22, 2]). These approaches are appealing for human motion capture because current commercial depth cameras are low-cost and can record per-pixel 3D depth information at a high frame rate. However, the use of a single depth camera for online motion capture often produces poor results due to sensor noise and inference ambiguity caused by significant occlusions. Among these, our work is most comparable to Wei et al. [2]. Both systems build upon the full-body motion tracking process that sequentially updates 3D skeletal poses using observed depth image data. However, our kinematic motion tracking process produces much more accurate results because we integrate depth data from three *Kinect* cameras, foot pressure information, detailed full-body geometry, and environmental contact constraints to reconstruct the full-body kinematic poses. In addition, our goal differs that we aim to reconstruct both full-body kinematics and dynamics data while their work is focused only on 3D kinematic pose reconstruction.

Our idea of leveraging Newtonian physics, contact pressure information, and depth image data to reconstruct kinematic and dynamic information is motivated by recent efforts in combining physical constraints and image data for human motion tracking (*e.g.*, [23, 24,

25, 26]). The use of physics for human motion tracking has been shown to be effective for tracking 2D low-body walking motion [23] or normal walking and jogging motion [24] in a recursive Bayesian tracking framework. Notably, Vondrak and his colleagues [26] proposed a video-based motion capture framework that optimizes both the control structure and parameters to best match the resulting simulated motion with input observation data. Our system is different because we optimize kinematic poses, internal joint torques, and contact forces based on observed data. In addition, our idea of combining depth images and pressure data significantly reduces the ambiguity of physics-based motion modeling.

In computer animation, pressure and contact information has been used to reconstruct and synthesize human motion using devices such pressure-sensing mats [27] and Wii balance boards [28]. Unfortunately, these systems do not permit the capture of highly dynamic motions, due to the static restrictions of the pressure-sensing hardware. Meanwhile, in biomechanics and health technology, there are a number of systems that have been used to acquire dynamic information, such as the center of pressure [29], which embed sensor technology directly into wireless footwear. However, the novelty of our system is that instead of collecting these dynamic values for separate analysis, we use this information immediately to assist in our kinematic and dynamic reconstruction.

Among all existing systems, our work is most similar to Wei and Chai [25], where a physics-based model was applied for reconstructing physically-realistic motion from monocular video sequences. Both systems aim to reconstruct full-body kinematic poses, internal joint torques, and contact forces across the entire motion sequence. However, their system relies on manual specification of pose keyframes, and intermittent 2D pose tracking in the image plane to define the objective for the optimization. In addition, they rely on manual specification of contacts or foot placement constraints to reduce the ambiguity of physics-based motion modeling. By contrast, our system is fully automatic. We also complement depth image data with pressure sensor data to obtain more accurate kinematic

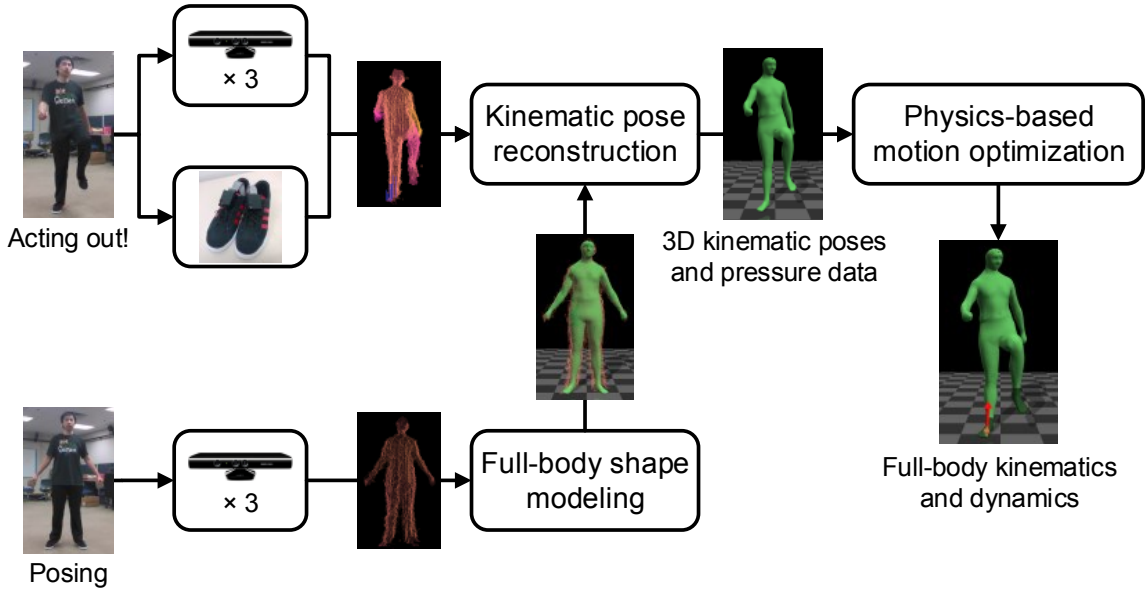


Figure II.1: System overview.

and dynamic information.

II.2 Overview

Our full-body kinematics and dynamics data acquisition framework automatically reconstructs 3D body shape, 3D kinematic poses, internal joint torques, and contact forces as well as contact locations and timings using three *Kinect* cameras and a pair of pressure-sensing shoes. The algorithm consists of three main components summarized as follows (see Figure II.1):

- Kinematic pose tracking. We introduce a novel tracking process that sequentially reconstructs 3D skeletal poses over time using input data captured by the *Kinect* cameras and wearable pressure sensors. We formulate an optimization problem that minimizes the inconsistency between the reconstructed poses and the observed depth and pressure data. We propose a new metric (*signed distance field* term) to evaluate how well the reconstructed poses match the observed depth data. The results are

highly accurate because our system leverages depth data from multiple cameras, foot pressure data, detailed full-body geometry, and environmental contact constraints. Figure II.2 (middle) shows the reconstructed kinematic pose that matches both observed depth data and foot pressure data.

- Physics-based motion optimization. Acquiring full-body dynamics data requires computing both contact forces and joint torques across the entire motion sequence. To achieve this goal, we introduce an efficient physics-based motion reconstruction algorithm that solves contact forces and joint torques as a quadratic programming problem. During reconstruction, we leverage Newtonian physics, friction cone constraints, contact pressure information, and 3D kinematic poses to reconstruct contact forces and joint torques over time. Figure II.2 (right) shows the reconstructed contact forces and torsional torques applied at the center of pressure.

- Full-body shape modeling. Reconstructing the body shape of the subject is important to our task because our kinematic tracking process relies on the full-body geometry to measure how well the reconstructed skeletal poses match the observed depth data. Furthermore, incorporating physical constraints into the reconstruction process requires the shape of the human subject to estimate the moment of inertia of each body segment. To address this challenge, we automatically construct a skinned full-body mesh model from the depth data obtained by three *Kinect* cameras so that the full-body mesh model can be deformed according to pose changes of an underlying articulated skeleton using Linear Blend Skinning (LBS) [30]. Each user needs to perform the shape modeling step only once (Figure II.3).

II.3 Data Acquisition and Preprocessing

Our system captures full-body kinematics and dynamics data using three synchronized depth cameras and a pair of pressure-sensing shoes (Figure II.4). In our experiment, *Kinect*

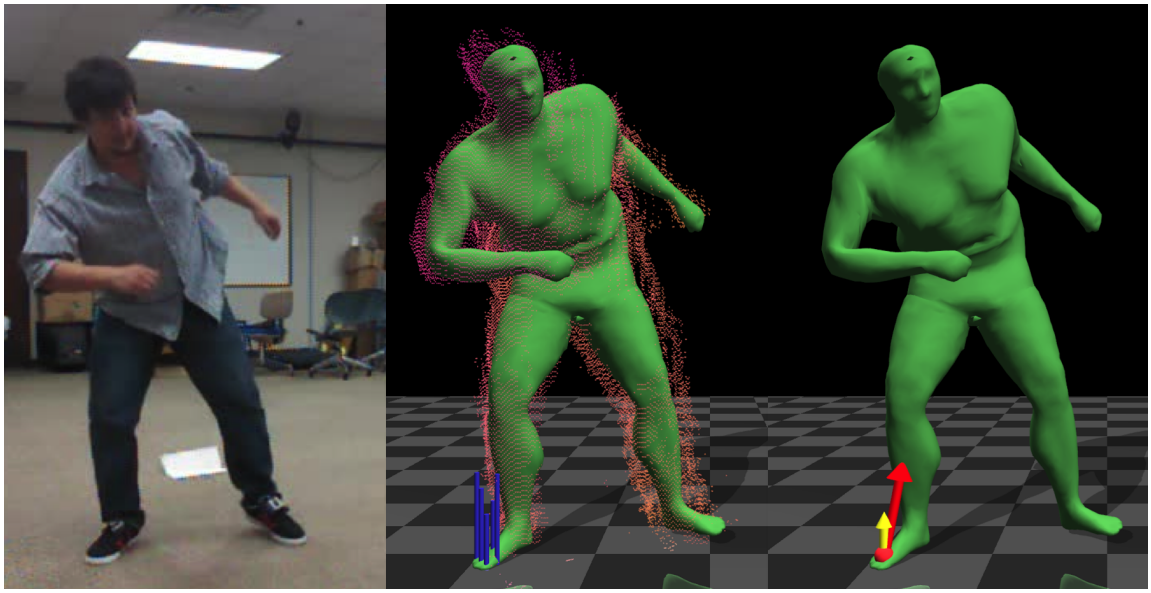


Figure II.2: Full-body kinematic and dynamics data acquisition. (left) reference image; (middle) the reconstructed 3D kinematic pose superimposed on observed depth and pressure data (blue lines); (right) the reconstructed pose, contact force (red arrows) and torsional torque (yellow arrows) applied at the center of pressure (red spheres).

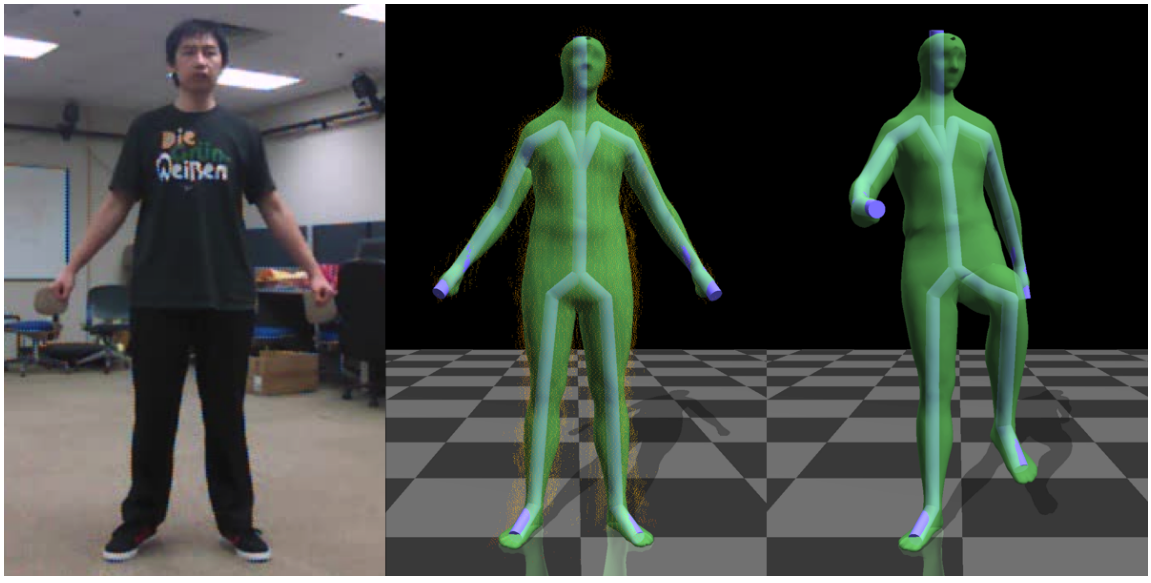


Figure II.3: Full-body shape modeling. (left) "A"-pose of the subject; (middle) the subject's 3D body shape reconstructed from observed depth data; (right) the reconstructed body shape under a new pose obtained from our motion acquisition system.

Figure II.4: Data acquisition. (left) three *Kinect* cameras and a pair of pressure-sensing shoes; (right) input data to our motion capture system includes the point cloud obtained by three cameras and pressure data (blue lines) recorded by pressure sensors.

cameras are used for motion capture but other commercially available depth cameras could be used as well.

II.3.1 Depth Data Acquisition

Current commercial depth cameras are often low-cost and can record 3D depth data at a high frame rate. A number of options exist; our system uses three *Microsoft Kinect* cameras, which cost roughly around five hundred dollars in total. Each camera returns 320 by 240 depth images at 30 frames per second (fps) with a depth resolution of a few centimeters. The three cameras are arranged uniformly in a circle with a radius of about 3 m, pointing to the center of the circle. The camera height is about 1 m. We found that this camera configuration yields the best trade-off between capture volume and depth data accuracy. We also found that in this configuration, interference of structured lights between cameras is not a major issue because each camera receives very little infrared (IR) light from other cameras. Most of the IR light is reflected back by the subject and most of the remaining IR light does not reach other cameras due to the large angle (120 degrees)

and large distance (2.6 *m*) between cameras.

II.3.2 Pressure Data Acquisition

The subject wears a pair of shoes during data acquisition. The insole of each shoe is equipped with eight highly accurate Tekscan [1] Flexiforce® sensors (the accuracy is linear within $\pm 3\%$ of full scale) that correspond to eight points on the feet as shown in Figure II.5. These sensors act as force-sensing resistors that are connected to a small microprocessor board enclosed and attached to the top of the shoe. Data is transmitted via a wireless Bluetooth connection at 120 *fps*.

II.3.3 Data Synchronization

We connect each depth camera to a different computer and connect the pressure shoes to one of them. We synchronize each computer's system time using Network Time Protocol (NTP). Data from different devices is synchronized by aligning timestamps to the timeline of the first camera. The Network Time Protocol provides very high accuracy synchronization in the local network, usually 5 – 10 *ms* in our experiments. This accuracy is sufficient for synchronization between *Kinect* sensors since the time interval between *Kinect* frames is about 33.3 *ms*. The pressure-sensing shoes are running at a much higher frame rate (120 *fps*), hence picking the frame with the closest timestamp for alignment usually gives satisfactory results.

II.3.4 Depth Camera Calibration

Reconstructing 3D body poses using multiple depth cameras requires computing the relative positions and orientations of each depth camera. For depth camera calibration, we use a large calibration box to find the rigid transformations between three cameras by aligning visible faces of the box and their intersection points. The calibration box is color coded so that each face/plane can be easily identified in RGBD images. Briefly, we first

detect each plane of the box by using color detection and RANSAC [31] techniques. We then extract the intersection point of three neighboring faces (or two neighboring faces and the ground plane) that are visible to the same camera. We align the intersection points from different cameras based on the known geometry of the calibration box. We move the box around in the scene to get a sufficient number of constraints to solve for the transformation matrices.

II.3.5 Depth Data Filtering

Given the calibrated camera parameters and the timestamps of each camera, we align the depth data from the three cameras to obtain a point cloud of the subject at each frame using the rigid transformations obtained from the calibration step (see Figure II.4 (right)). We introduce a simple yet effective filtering technique to reduce noise in point cloud data. Specifically, we first build a neighbor graph, each node of which represents a point from the point cloud. We connect two nodes if their distance is smaller than a threshold. We obtain the filtered point cloud by extracting the largest connected components from the neighbor graph. This process usually does not discard noisy points close to the body, but we have found that these points do not affect the accuracy of our full-body tracking process. Combining depth data with pressure data for kinematics and dynamics data capture also requires enforcing ground contact constraints. To this end, we extract the 3D ground plane by applying the RANSAC technique [31] to the observed depth data.

II.3.6 Full-body Pose Representation

We use a skinned mesh model to approximate full-body geometry of human subjects (see Section II.6). This mesh is driven by an articulated skeleton model using Linear Blend Skinning (LBS). The skinned mesh model contains 6449 vertices and 12894 faces; and our skeleton model contains 24 bone segments. We describe a full-body pose using a set of independent joint coordinates $\mathbf{q} \in R^{36}$, including absolute root position and orientation as

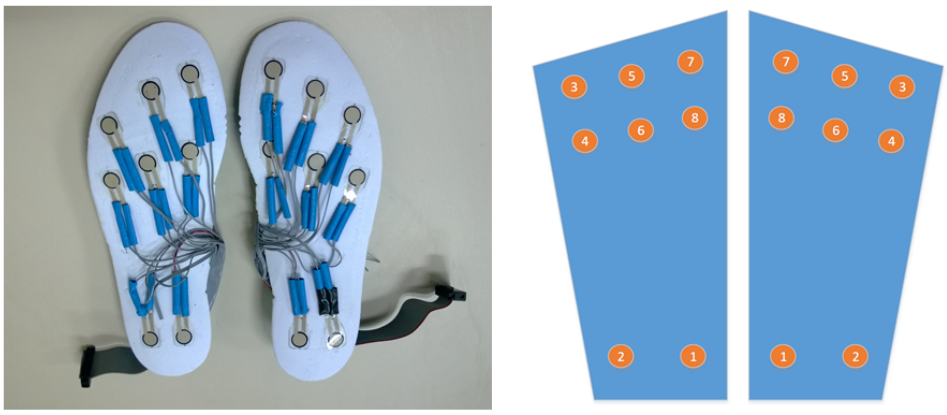


Figure II.5: Insoles of our pressure-sensing shoes. (left) *Tekscan* [1] Flexiforce® pressure sensors on the insole of the shoes; (right) corresponding assignments for the sensors.

well as the relative joint angles of individual joints. These bones are head (1 Dof), neck (2 Dof), lower back (3 Dof), and left and right shoulders (2 Dof), arms (3 Dof), forearms (1 Dof), upper legs (3 Dof), lower legs (1 Dof), and feet (2 Dof).

II.4 Kinematic Pose Tracking

We now describe our kinematic pose tracking algorithm that sequentially reconstructs 3D human poses from the observed point cloud and pressure sensor data. We formulate the sequential tracking problem in an efficient optimization framework and iteratively register a 3D skinned mesh model with observed data via linear system solvers. In the following section, we explain how to incorporate point cloud, pressure data, full-body geometry, contact constraints and pose priors into our tracking framework.

Let O_i be the point cloud obtained from *Kinect* cameras and S_i be the readings from pressure sensors at the current frame i. We want to estimate from O_i and S_i the skeletal poses $\mathbf{q}_i$ for the current frame given previously reconstructed poses $\mathbf{q}_{i-1}, ..., \mathbf{q}_{i-M}$. Dropping the index i for notational brevity, we aim to estimate the optimal skeletal poses $\mathbf{q}^*$ that best match observed data O and S.

We estimate the full-body kinematic poses by minimizing an objective function consisting of five terms:

$$\min_{\mathbf{q}} \ \lambda_1 E_{SDF} + \lambda_2 E_{Boundary} + \lambda_3 E_{PD} + \lambda_4 E_{GP} + \lambda_5 E_{Prior}, \tag{II.1}$$

where E_{SDF}, $E_{Boundary}$, E_{PD}, E_{GP} and E_{Prior} represent the *signed distance field* term, *boundary* term, *pressure data* term, *ground penetration* term, and *prior* term respectively. The weights $\lambda_1, ..., \lambda_5$ control the importance of each term and are experimentally set to 2, 2, 100, 100, and 0.1 respectively. We describe details of each term in the following subsections.

II.4.1 Signed Distance Field Term

We adopt an analysis-by-synthesis strategy to evaluate how well the hypothesized pose $\mathbf{q}$ matches the observed point cloud O. Specifically, given a hypothesized joint angle pose $\mathbf{q}$, we first apply the corresponding transformation $T_{\mathbf{q}}$ obtained by forward kinematics to each vertex of the skinned mesh model to synthesize a 3D geometric model of the human body. Given the calibrated camera parameters, we can further project the posed 3D mesh model onto the image plane and render the hypothesized depth images from each viewpoint. The hypothesized point cloud is formed by aligning the rendered depth images from each viewpoint using the calibrated camera parameters.

So how can we evaluate the distance between the observed and hypothesized point clouds? This often requires identifying the correspondences between the two sets of depth points. Previous approaches (e.g., [9, 10]) often apply Iterative Closest Points (ICP) method to find the correspondences between the two data sets. However, ICP techniques often produce poor results for human pose registration (for details, see our evaluation in Section II.7.2). To address this challenge, we propose to compute signed distance fields from the two point clouds and register the hypothesized and observed signed distance fields

via 3D image registration techniques, thereby avoiding building explicit correspondences between the hypothesized and observed point clouds.

A signed distance field (SDF) [32] is often represented as a grid sampling of the closest distance to the surface of an object described as a polygonal model. SDFs are widely applied in computer graphics and have been used for collision detection in cloth animation [33], multi-body dynamics [34], and deformable objects [35]. In our application, we compute SDFs from the point clouds and apply them to iteratively register the hypothesized joint angle pose $\mathbf{q}$ with the observed point cloud O.

We define the SDF on a $50 \times 50 \times 50$ regular grid in three dimensional space. We define the voxel values of the signed distance field V from a point cloud C as follows:

$$V(\mathbf{p}_i) = f_s(\mathbf{p}_i) \cdot \min_{\mathbf{r} \in C} \|\mathbf{p}_i - \mathbf{r}\|^2, \tag{II.2}$$

where $\mathbf{p}_i$ is the coordinates of the center of the ith voxel V^i, $\mathbf{r}$ is a point in the point cloud C, and

$$f_s(\mathbf{p}) = \begin{cases} -1, & \text{if } \mathbf{p} \text{ inside;} \\ 1, & \text{if } \mathbf{p} \text{ outside.} \end{cases} \tag{II.3}$$

That is, for a volume V, each voxel V^i represents its smallest signed distance to the point cloud C.

We compute the SDF of the observed point cloud in two steps. We first obtain the value of each voxel by searching the closest points in the point cloud. The sign of the voxel value is determined by projecting the voxel V^i onto each of the depth images and comparing the projected depth value d_{proj} with the corresponding depth value d_o in each of the observed depth images. We set the sign to be negative if $d_{proj} > d_o$ for all three images. The sign is set to be positive if d_o does not exist or $d_{proj} < d_o$ for any image. The SDF of the hypothesized point cloud is computed in a similar way.

Once we compute the SDFs for the hypothesized and observed point clouds, we can use them to evaluate the following term in the objective function:

$$E_{SDF}(\mathbf{q}) = \sum_{i \in S_{SDF}} \|V_R^i(\mathbf{q}) - V_O^i\|^2, \tag{II.4}$$

where $V_R^i(\mathbf{q})$ is the value of the ith voxel of the hypothesized SDF and it depends on the hypothesized skeletal pose $\mathbf{q}$. V_O^i is the voxel value of the ith voxel of the observed SDF, and S_{SDF} includes the indices of all the voxels used for evaluating E_{SDF}. Note that not all the voxels are included for evaluation. In our implementation, we exclude voxels with zero gradients because they do not contribute to the pose updates. To speed up the tracking system, we also ignore the voxels that are far away from the surface of the rendered skinned mesh model as they provide little guidance on the tracking process.

A major benefit of the signed distance field term is that it merges all the observation information from depth cameras, including both depth and boundary information. This significantly reduces ambiguity for 3D pose reconstruction. In our experiment, we have found that using a coarse resolution SDF is often sufficient for tracking 3D poses since it provides us a large number of constraints, even more than using the point clouds itself, due to the use of information inside and outside the point clouds. Another benefit of the SDF term is that the function $E_{SDF}(\mathbf{q})$ is continuous, which makes the gradient differentiable everywhere with respect to the hypothesized pose $\mathbf{q}$. This property is particularly appealing to our pose tracking solver because we apply gradient-based optimization to do the pose tracking. As shown in our results, our method produces more accurate results than alternative solutions such as ICP (*e.g.*, [9, 10]) and model-based depth flow [2].

II.4.2 Boundary Term

In practice, even with ground truth poses, the hypothesized point cloud might not precisely match the observed point cloud due to camera noise, cloth deformation, calibration errors, and blurry depth images caused by fast body movements. Therefore, the signed distance field term alone is often not sufficient to produce satisfactory results, particularly when significant occlusions occur. This motivates us to introduce the boundary term to further improve the tracking accuracy.

Intuitively, the boundary term minimizes the size of non-overlapping regions between the hypothesized and observed point clouds. To be specific, we penalize the distances between the hypothesized points $\mathbf{p}(\mathbf{q})$ in the non-overlapping region and their closest points $\mathbf{p}^*$ from the observed point cloud. We have

$$E_{Boundary}(\mathbf{q}) = \sum_{\mathbf{p} \in S_B} \|\mathbf{p}(\mathbf{q}) - \mathbf{p}^*\|^2. \tag{II.5}$$

A critical issue for the boundary term evaluation is to determine which points in the hypothesized point cloud should be included for evaluation (*i.e.*, S_B). Our evaluation considers all the points in non-overlapping regions of the hypothesized and observed depth images from each camera viewpoint. This ensures that the hypothesized point cloud moves towards the observed point cloud to reduce the size of non-overlapping regions as quickly as possible.

In our implementation, we search the closest points based on a bidirectional distance measurement in order to ensure one-to-one correspondences. For observed depth points in the non-overlapping region, we first find the closest points in the hypothesized point cloud. Then for the hypothesized depth points who have multiple correspondences, we pick the one with the largest distance to ensure a one-to-one correspondence. Correspondences for hypothesized depth points are determined similarly.

II.4.3 Pressure Data Term

Depth data alone is often not sufficient to accurately reconstruct the movement of both feet because the observed depth data is often very noisy. The most visible artifact in the reconstructed motion is footskate, which can be corrected by existing methods if the footplants are annotated [36]. However, footplant constraints are extremely hard to derive from noisy depth image data. To address this challenge, we complement depth data with pressure data obtained from a pair of pressure-sensing shoes. When a pressure sensor is "on", we can enforce the corresponding footplant constraints on pose reconstruction.

Under the assumption that the only contact the feet have is with the ground plane, we define the pressure data term as follows:

$$E_{PD}(\mathbf{q}) = \sum_m b_m dist(\mathbf{p}_m(\mathbf{q}), G_F), \tag{II.6}$$

where the function *dist* measures the distance between the global coordinates of the mth pressure sensor $\mathbf{p}_m(\mathbf{q})$ and the 3D ground plane G_F. Here the local coordinates of each pressure sensor are known in advance so that we can apply forward kinematics to map the local coordinates of the mth pressure sensor to its global 3D coordinates $\mathbf{p}_m(\mathbf{q})$ under the current pose $\mathbf{q}$. In our implementation, we use a binary variable b_m to indicate whether the mth pressure sensor is "on". This variable provides a means to exclude erroneous non-zero pressure data that can be received even when airborne. Such readings can occur because the sensors are attached to the insole of the shoe rather than the exterior of the shoe sole.

We adopt a simple yet effective rule to determine if a particular pressure sensor is "on" or "off". At each iteration of kinematic pose optimization, we evaluate whether the pressure sensor is "off" based on the following two criteria: (1) we consider all the pressure sensors from a foot as "off' if the sum of pressure values is smaller than a threshold ε_1 and (2) we consider a particular pressure sensor is "off" if its vertical position in the previous

iteration of kinematic pose optimization is above the ground plane and its distance to the ground plane is larger than a threshold ε_2. We experimentally set ε_1 and ε_2 to 0.008 and 0.05 m respectively.

II.4.4 Ground Penetration Term

The pressure data term alone often cannot avoid foot-ground penetration. This is because we model each foot using a detailed mesh model and therefore a small number of contact points are often not sufficient to avoid ground penetration. We introduce the ground penetration term to address this issue.

We sample a set of points $n = 1, ...N$ on each foot and prevent them from penetrating into the ground. In particular, we penalize the penetration between the foot and the ground G_F, resulting in the following objective term:

$$E_{GP}(\mathbf{q}) = \sum_n \|f_p(\mathbf{p}_n(\mathbf{q}), G_F)\|^2, \tag{II.7}$$

$$f_p(\mathbf{p}_n(\mathbf{q}), G_F) = \begin{cases} 0, & \text{if no penetration} \\ dist(\mathbf{p}_n(\mathbf{q}), G_F), & \text{otherwise} \end{cases}, \tag{II.8}$$

where $\mathbf{p}_n(\mathbf{q})$ is the global coordinates of the nth contact point on the foot. Like the pressure data term, the function $dist$ measures the distance between the global coordinates of the nth contact point $\mathbf{p}_n(\mathbf{q})$ and the 3D ground plane G_F.

II.4.5 Prior Term

We incorporate the prior term into our tracking process for two reasons. First, the depth data is sometimes ambiguous because of significant occlusions, camera noise, cloth deformation, or blurry depth images caused by fast body movements. Second, the reconstructed joint angle poses may violate the joint limits. We utilize subspace pose priors

embedded in a highly varied motion capture database to solve this problem.

We construct separate PCA models for the pose of each body part (arms, shoulders, spines, legs, and feet). The training data we use is from the CMU mocap database, which includes 4.6 hours of highly varied motions. We use the constructed PCA models to constrain the solution space of kinematic tracking. In our implementation, we enforce the subspace constraints as soft constraints, resulting in the following objective term:

$$E_{Prior}(\mathbf{q}) = \|P_k^T(P_k(\mathbf{q}-\mu)) + \mu - \mathbf{q}\|^2, \tag{II.9}$$

where P_k is the first k principal components of the PCA model and μ is the mean vector of the PCA model. The numbers of dimension of the PCA models (k) are automatically determined by keeping 95% of original variations.

We have found that enforcing such weak PCA priors allow us to achieve similar results as the joint limit constraints while still enabling us to optimize the pose using iterative linear solvers.

II.4.6 Kinematic Pose Reconstruction

Solving the objective function described in Equation (II.1) requires minimizing a sum of squares of non-linear functions. We apply a Gauss-Newton optimization algorithm to solve this problem. Given a known, current estimate of $\mathbf{q}$, we iteratively solve for increments to the parameters $\delta\mathbf{q}$ using linear system solvers. Note that our kinematic pose tracking process is fully automatic as we initialize the pose at the first frame using *Microsoft Kinect* for Windows [7].

For each subsequent time step, we initialize the current pose using the previously estimated pose and iteratively perform the following steps until the change of the pose is smaller than a specified threshold:

- Step 1: Given the current pose $\mathbf{q}$ and the full-body skinned mesh model, we ren-

der the depth images $D_R(\mathbf{q})$ from each camera viewpoint. For a point $\mathbf{p} \in R$ in the rendered depth image, we use OpenGL's selection buffer to determine which bone segments the point is associated with as well as the local coordinates of the corresponding surface point. This step is necessary for evaluating the partial derivatives $\partial\mathbf{p}/\partial\mathbf{q}$ because the global coordinates of surface points are dependent on both the local coordinates and associated bone segments.

- Step 2: We compute the hypothesized and observed signed distance fields V_R and V_O based on the point clouds C_R and C_O obtained from the hypothesized and observed depth images $D_R(\mathbf{q})$ and $D_O(\mathbf{q})$ (see Equation (II.2)).

- Step 3: We calculate the gradients of the hypothesized signed distance field and other partial derivatives in Equations (II.4), (II.5), (II.6), (II.7) and (II.9) to form linear equations.

- Step 4: We compute the optimal increment $\delta\mathbf{q}$ using linear system solvers and update the current pose: $\mathbf{q} = \mathbf{q} + \delta\mathbf{q}$.

The algorithm usually converges within 10 iterations as we initialize the solution using previously reconstructed poses. The output of the kinematic tracking process includes kinematic pose $\mathbf{q}$ at current frame as well as contact states (b_m) and global 3D coordinates ($\mathbf{p}_m(\mathbf{q})$) of each pressure sensor.

II.5 Physics-based Motion Optimization

In this section, we describe how to reconstruct full-body dynamics data using both observed pressure data and reconstructed kinematic motion data obtained from Section 5. We formulate a quadratic programming problem to seek optimal values for internal joint torques and contact forces that best match observed pressure data and reconstructed

kinematic poses as well as contact states. Similar to the kinematic tracking process, we solve the full-body dynamics reconstruction process in a sequential manner.

II.5.1 Full-body Dynamics

The Newtonian dynamics equations for full-body movement can be defined as follows:

$$M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q}, \dot{\mathbf{q}}) + h(\mathbf{q}) = \mathbf{u} + \mathbf{J}^T\mathbf{f}, \tag{II.10}$$

where $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ represent the joint angle poses, velocities, and accelerations respectively. The quantities $M(\mathbf{q})$, $C(\mathbf{q}, \dot{\mathbf{q}})$ and $h(\mathbf{q})$ are the joint space inertia matrix, centrifugal/Coriolis, and gravitational forces respectively. The vectors $\mathbf{u}$ and $\mathbf{f}$ are joint torques and contact forces respectively. The contact force Jacobian matrix $\mathbf{J}$ maps joint velocities to world space cartesian velocities at the contact points. Human muscles generate torques about each joint, leaving global position and orientation of the body as unactuated joint coordinates. The movement of global position and orientation is controlled by contact forces $\mathbf{f}$. Modifying those coordinates requires contact forces $\mathbf{f}$ from the environment.

Enforcing Newtonian dynamics constraints requires computing the mass and moment of inertia of each body segment. To achieve this goal, we first reconstruct a full-body skinned mesh model to approximate the whole-body geometry of the subject (see Section 7). We then voxelize the reconstructed skinned mesh model. For each voxel, we compute its geodesic distance to all bone segments and associate it with a particular bone segment that is closest to the voxel. Assuming the weight of the subject is known, we can estimate the density of a subject's body and use it to compute the physical quantities of each bone segment, including mass and moment of inertia.

II.5.2 Friction Cone Constraints

During ground contact, the feet can only push, not pull on the ground, contact forces should not require an unreasonable amount of friction, and the center of pressure must fall within the support polygon of the feet. We use Coulomb's friction model to compute the forces caused by the friction between the character and environment. A friction cone is defined to be the range of possible forces satisfying Coulomb's function model for an object at rest. We ensure the contact forces stay within a basis that approximate the cone with nonnegative basis coefficients. We model the contact between the foot and ground using eight contact points (see Figure II.5), which are consistent with the locations of pressure sensors. This allows us to represent the contact forces $\mathbf{f}$ as a linear function of nonnegative basis coefficients:

$$\mathbf{f}(\mathbf{w}_1, ..., \mathbf{w}_8) = \sum_{m=1}^{8} \mathbf{B}_m \mathbf{w}_m \text{ subject to } \mathbf{w}_m \geq \mathbf{0}, \tag{II.11}$$

where the matrix $\mathbf{B}_m$ is a 3×4 matrix consisting of 4 basis vectors that approximately span the friction cone for the m-th contact force. The 4×1 vector $\mathbf{w}_m$ represents the nonnegative basis weights for the m-th contact force.

II.5.3 Pressure Data

Each pressure sensor records an analog resistance reading proportional to the applied pressure, which is then converted to a digital value. The relationship between the analog resistance reading R_m and the digital pressure force value P_m returned is defined as follows:

$$P_m = k_m / R_m, \tag{II.12}$$

where k_m is a scaling parameter for each sensor and assumed to be unknown.

II.5.4 Full-body Dynamics Reconstruction

We formulate full-body dynamics reconstruction in a quadratic programming framework. Given observed pressure data R_m and reconstructed kinematic poses $\mathbf{q}$ and contact states b_m obtained from the tracking process, the optimization simultaneously computes joint torques $\mathbf{u}$, contact forces $\mathbf{f}(\mathbf{w})$, and pressure sensors coefficients $\mathbf{k} = [k_1, ..., k_8]^T$ that maximize the performance of the following multiobjective function:

$$\begin{aligned} &\arg\min_{\mathbf{u},\mathbf{w},\mathbf{k}} E_{pressure}(\mathbf{w},\mathbf{k}) + \lambda_1 E_{reg}(\mathbf{k}) + \lambda_2 E_{torque}(\mathbf{u}) \\ &\text{subject to } M(\mathbf{q})\ddot{\mathbf{q}} + C(\mathbf{q},\dot{\mathbf{q}}) + h(\mathbf{q}) = \mathbf{u} + \mathbf{J}^T\mathbf{f}(\mathbf{w}), \\ &\mathbf{w} \geq \mathbf{0}. \end{aligned} \tag{II.13}$$

In the above, the first term $E_{pressure}$ evaluates the consistency between the reconstructed contact forces and observed pressure forces. Specifically, the pressure term is defined as follows:

$$E_{pressure} = \sum b_m ||f_{m,\perp} - k_m/R_m||^2, \tag{II.14}$$

where $f_{m,\perp}$ is the vertical component of the reconstructed contact force at the mth sensor. And R_m and k_m are the reading and scale of the mth pressure sensor.

The second term E_{reg} is a regularization term that ensures the scaling parameters of all the pressure sensors are as close as possible. This is achieved by minimizing the variance of the scale parameters for all the "on" pressure sensors:

$$E_{reg} = \frac{1}{(\sum b_m - 1)} \sum b_m (k_m - \frac{\sum b_m k_m}{\sum b_m})^2. \tag{II.15}$$

The third term E_{torque} minimizes the sum of squared torques at the current frame. The optimization is also subject to the discretization of Newtonian dynamics equations determined by a finite difference scheme and friction cone constraints $\mathbf{w} \geq \mathbf{0}$.

In our implementation, we use the backward difference approximation to compute joint velocities and use the central difference approximation to compute joint accelerations with δt set to $1/30\ s$. We solve the optimization problem using quadratic programming.

II.6 Full-body Shape Modeling

This section describes how to reconstruct full-body mesh models of human subjects using a small number of depth images captured by three *Kinect* cameras. We model full-body geometry of human subjects as a skinned mesh model. We introduce an efficient full-body shape modeling technique that automatically reconstructs a detailed skinned mesh model of a subject using the depth data obtained from three *Kinect* cameras. Each user needs to perform this step only once. Note that the user should not wear overly loose clothing like skirt for modeling, as it will mislead the system and produce an inaccurate shape model for estimating physical quantities of human bodies.

II.6.1 Shape Representation

Our human body model is based on statistical analysis of a database of pre-registered 3D full-body scans [37]. In particular, we apply PCA to hundreds of aligned body scans [38] to construct a low-dimensional parametric model for human body representation. We represent human body geometry using a mean mesh model $\mathbf{A}$ and a weighted combination of eigen mesh basis P:

$$\mathbf{M}(\mathbf{X}) = P\mathbf{X} + \mathbf{A}, \tag{II.16}$$

where $\mathbf{M} = [x_0, y_0, z_0, x_1, y_1, z_1, \ldots, x_n, y_n, z_n]$ is a long vector stacking all the vertices of the mesh model and $\mathbf{X}$ is the low-dimensional shape parameter to represent a full-body geometric model.

We further build a skinned mesh model for the registered mesh model so that the mesh model can be deformed according to pose changes of an underlying articulated skeleton

using Linear Blend Skinning (LBS).

II.6.2 Shape Reconstruction

To reconstruct a full-body skinned mesh model for the subject, we instruct the user to perform a reference pose ("A" -pose, see Figure II.3) for about one second. As a result, we obtain three sequences of depth images. Our goal herein is to reconstruct both full-body poses and full-body geometry from the recorded depth image sequences. We formulate the problem as an optimization and seek to find the optimal shape parameter $\mathbf{X}$ and skeletal pose $\mathbf{q}$ that best fit the observed point cloud C:

$$\mathbf{X}^*, \mathbf{q}^* = \arg\min_{\mathbf{X},\mathbf{q}} \sum_i \| \mathbf{p}_i(P\mathbf{X} + \mathbf{A}) \bigoplus T_{\mathbf{q}} - \mathbf{p}_i^* \|^2, \tag{II.17}$$

where $\mathbf{p}_i(\mathbf{M})$ is 3D coordinates of the ith vertex of the parametric mesh model $\mathbf{M}$ and $\mathbf{p}_i^*$ is the 3D coordinates of the closest point of $\mathbf{p}_i$ in C. The operator $\bigoplus$ applies the corresponding transformation $T_{\mathbf{q}}$ to each vertex of the surface mesh model $\mathbf{p}_i(\mathbf{M})$ to obtain 3D full-body geometric model under the pose $\mathbf{q}$.

We have found that direct optimization of the cost function is not efficient and the optimization is prone to falling into local minima. To address this issue, we introduce an iterative optimization algorithm to decompose the large optimization problem into two smaller problems that can be solved efficiently. We initialize the pose using the "A" -pose. In each iteration, we keep one group of the unknowns unchanged and search for an optimal update for the other group of unknowns using the following steps:

- Non-rigid shape estimation. In this step, we estimate the shape parameter $\mathbf{X}$ from the observed point cloud while keeping the pose $\mathbf{q}^*$ constant. This requires solving

the following optimization problem:

$$\mathbf{X}^* = \arg\min_{\mathbf{X}} \sum_i \|\mathbf{p}_i(P\mathbf{X}+\mathbf{A}) \bigoplus T_{\mathbf{q}^*} - \mathbf{p}_i^*\|^2. \quad \text{(II.18)}$$

We extend iterative closest points (ICP) techniques to iteratively estimate the shape parameter $\mathbf{X}$. Briefly, we search the closest points for each vertex of the current mesh model $\mathbf{M}(\mathbf{X})$ on the observed point cloud and use them to update the shape parameter $\mathbf{X}$ with least-square fitting techniques.

- Skeletal pose update. We fix the shape parameter $\mathbf{X}$ and use it to update the skeletal pose $\mathbf{q}$ based on the observed point cloud. This problem can be solved efficiently using the kinematic tracking algorithm described in Section II.4.

II.7 Results

We demonstrate the power and effectiveness of our system by capturing a wide range of human movements using our proposed system (Section II.7.1). Our comparison against alternative methods shows the system achieves state-of-the-art accuracy (Section II.7.2 and II.7.3). We assess the performance of our kinematic tracking process by dropping off each term in the cost function (Section II.7.4). We validate the quality of dynamics data obtained from our system by comparing joint torques patterns obtained from our system against those reconstructed from the *Vicon* system and force plates (Section 8.5).

For the current implementation, our kinematic tracking and physics-based optimization process run at 6 *fps*. It takes three seconds to complete the offline full-body shape modeling process.

II.7.1 Test on Real Data

We have tested our system on a wide variety of human actions, including walking, running, jumping, dancing, and sport activities such as basketball, baseball, and boxing.

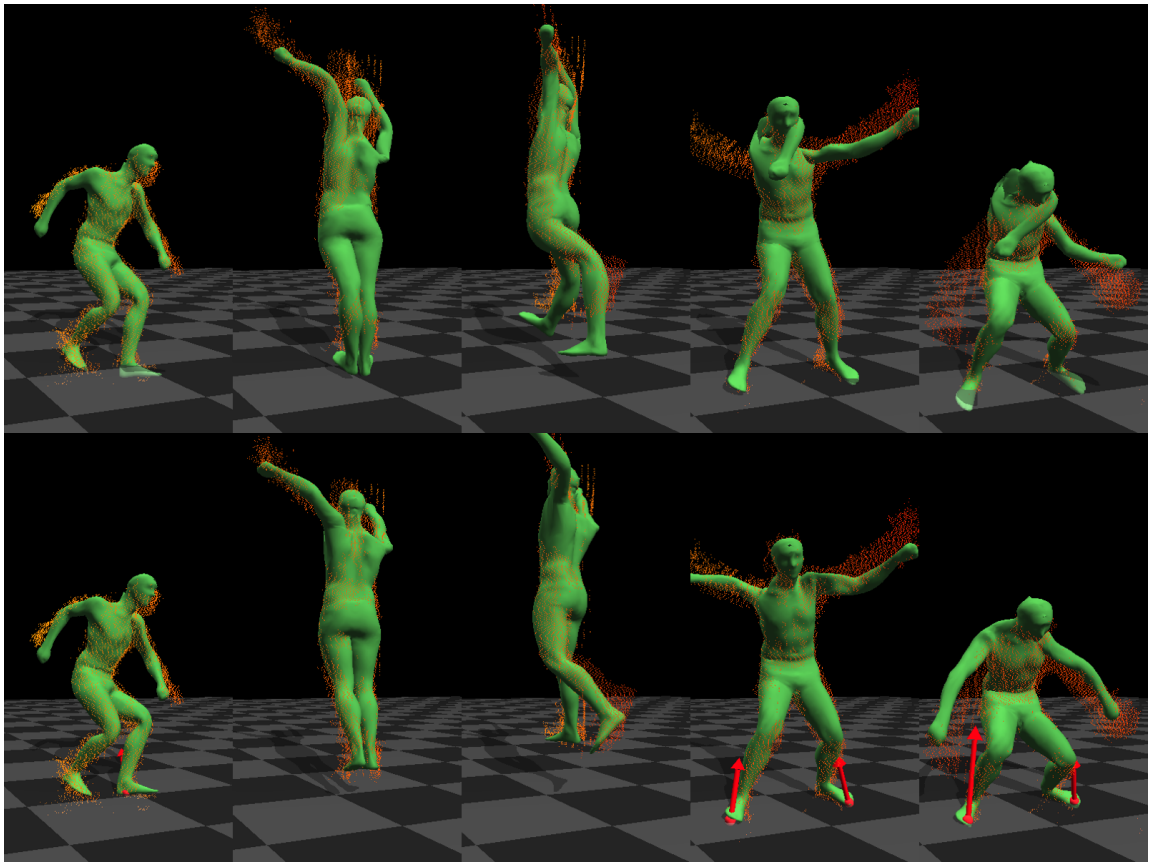


Figure II.6: Comparison against Wei et al. [2]. (top) results obtained from Wei et al. [2]; (bottom) our results.

The results show the performance of our system on a large number of complex and fast motions that a single camera could not capture, such as jumping with a 360 degree rotation and kicking while rotating. We also demonstrate the robustness of our system on several long sequences like boxing, stealing, and dancing.

II.7.2 Comparisons against Alternative Methods

We have evaluated the effectiveness of our kinematic tracking system by comparing against alternative full-body tracking methods. It is worth pointing out that our whole motion capture system can automatically and accurately capture internal joint torques and contact forces, as well as contact locations and timings, across the entire sequence, a capability that has not been demonstrated in alternative tracking systems.

II.7.2.1 Comparison against Wei et al.

We compare our system against the state-of-the-art in full-body motion capture using a single depth camera [2]. For a fair comparison, we first extend their tracking algorithm to multiple *Kinect* cameras by combining all the information obtained from three depth cameras. The accompanying video highlights a side-by-side comparison between the two systems. Figure II.6 shows the advantage of our system.

II.7.2.2 Comparison against ICP techniques

We compare our 3D kinematic tracking process described in Section II.4 against Iterative Closest Point (ICP) techniques [9, 10]. Specifically, we apply ICP to minimize the distances between the observed point cloud obtained from three depth cameras and the hypothesized point cloud rendered from the skinned mesh model by iteratively finding the closest correspondences between them. We start both methods with the same initial pose. The accompanying video clearly shows that our tracking process is much more robust and accurate than the ICP algorithm. In the jumping example shown in Figure II.7, our track-

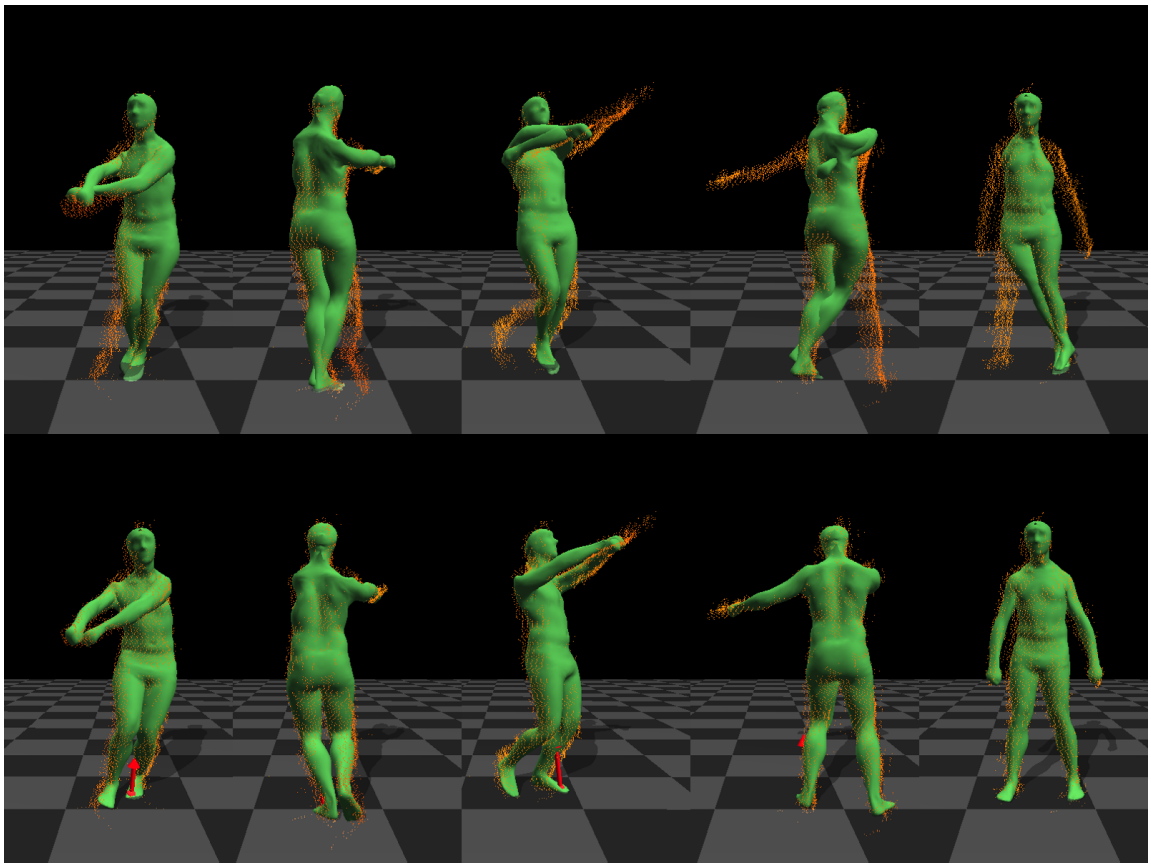


Figure II.7: Comparison against ICP algorithm. (top) results from ICP algorithm; (bottom) our results.

ing process successfully tracks the entire motion sequence while ICP fails to track most of frames. This is because ICP is often very sensitive to initial poses and prone to local minima, particularly when tracking high-dimensional human body poses from noisy depth data.

II.7.2.3 Comparison against *Vicon*

In this experiment, we quantitatively assess the quality of the captured motion by comparing against motion data captured with a full marker set in a twelve-camera *Vicon* system [3]. The average reconstruction error, which is computed as the average 3D joint

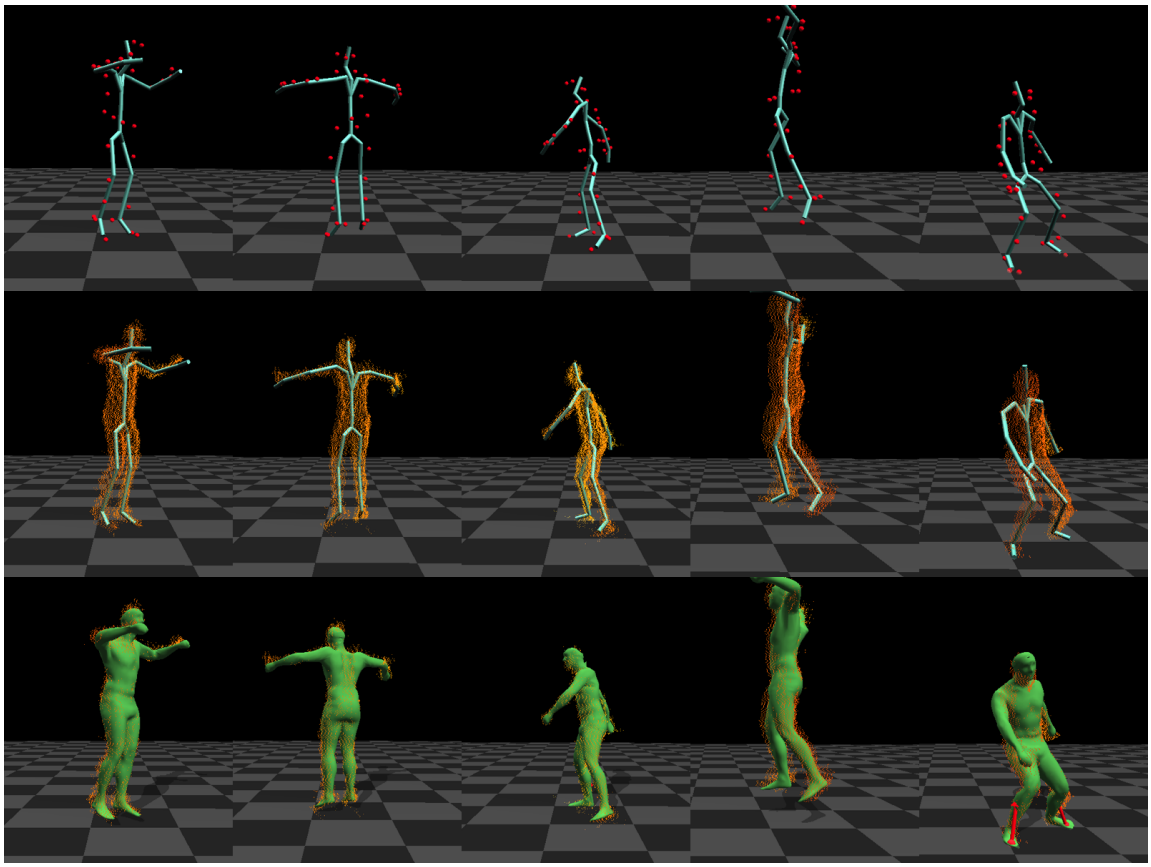


Figure II.8: Comparison against *Vicon* [3]. (top) results from a twelve-cameras *Vicon* system with a full set of markers; (middle) our results with a skeleton model; (bottom) our results with a skinned mesh model.

position discrepancy between the estimated poses and the ground truth mocap poses, is about 3.8 *cm* per joint per frame. Figure II.8 shows a side-by-side comparison between our result and the result obtained by the *Vicon* system.

II.7.3 Quantitative Evaluation

We quantitatively evaluate the reconstruction accuracy and robustness of our system by comparing against four alternative methods, including our algorithm without pressure data, Wei et al. [2], *Kinect* [7], and ICP on six different actions. The ground truth data is obtained by motion data captured with a twelve-camera *Vicon* system in a full marker

set. For a fair comparison, we include the prior term in all alternative methods except *Kinect* [7]. For *Kinect*, we obtain separate poses from three depth cameras at each frame and choose the pose closest to the ground truth data as the output.

II.7.3.1 Reconstruction Accuracy Evaluation

To evaluate the reconstruction accuracy, we compute average joint position errors and variances for each method by comparing against ground truth poses obtained from the *Vicon* system (Figure II.9). The evaluation shows that our system produces a much lower error and variance (3.8 ± 1.3 *cm*) than Wei et al. [2] (5.0 ± 2.2 *cm*) and *Kinect* (7.7 ± 2.5 *cm*). Among all the methods, ICP produces largest errors for all the test data. The evaluation also shows that complementing depth data with pressure data improves the accuracy from 4.1 ± 1.3 *cm* to 3.8 ± 1.3 *cm*. Figure II.10 compares average reconstruction errors of each joint for our method, Wei et al. [2] and *Kinect* system. Our system produces more accurate reconstruction results than two alternative methods for all the joints.

II.7.3.2 Robustness Evaluation

To evaluate the system robustness, we compute the percentage of failure frames for each motion. Here we define a reconstructed frame as "failure" if the average joint position discrepancy is larger than 6 *cm*. Figure II.11 shows that our system produces a much lower failure rate (5.9%) than alternative methods (14.9% for Wei et al. [2] and 68.3% for *Kinect* [7]).

II.7.4 Evaluation of Kinematic Pose Tracking Process

We have evaluated the importance of key components of our kinematic tracking process by dropping off each term in Equation (II.1).

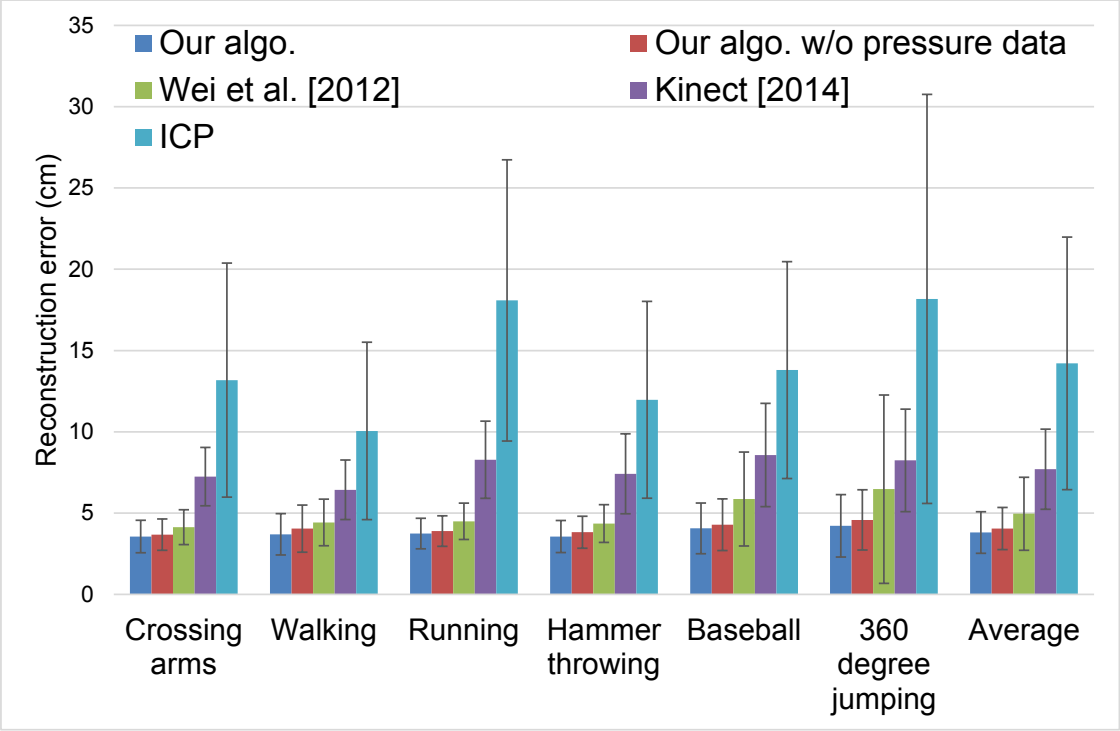


Figure II.9: Evaluation of reconstruction accuracy (average joint position errors and variances) for five methods on six test actions.

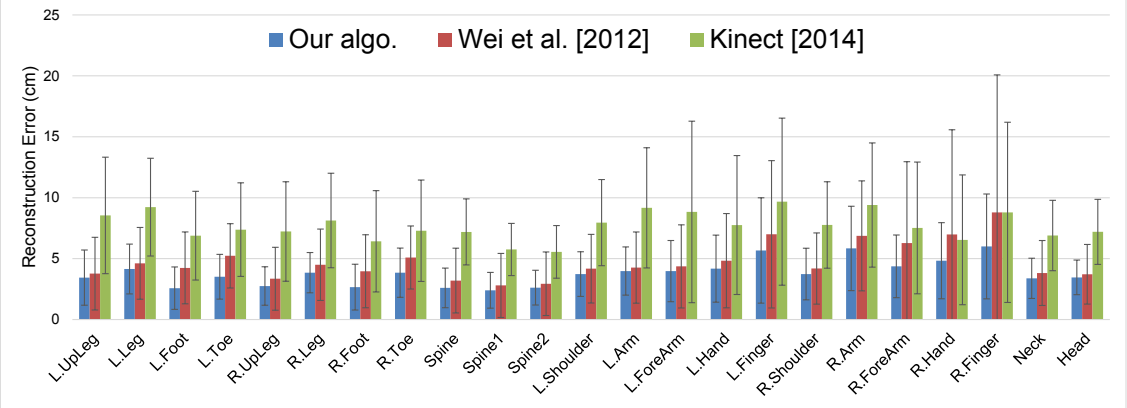


Figure II.10: Average joint reconstruction errors and variances on six action sequences.

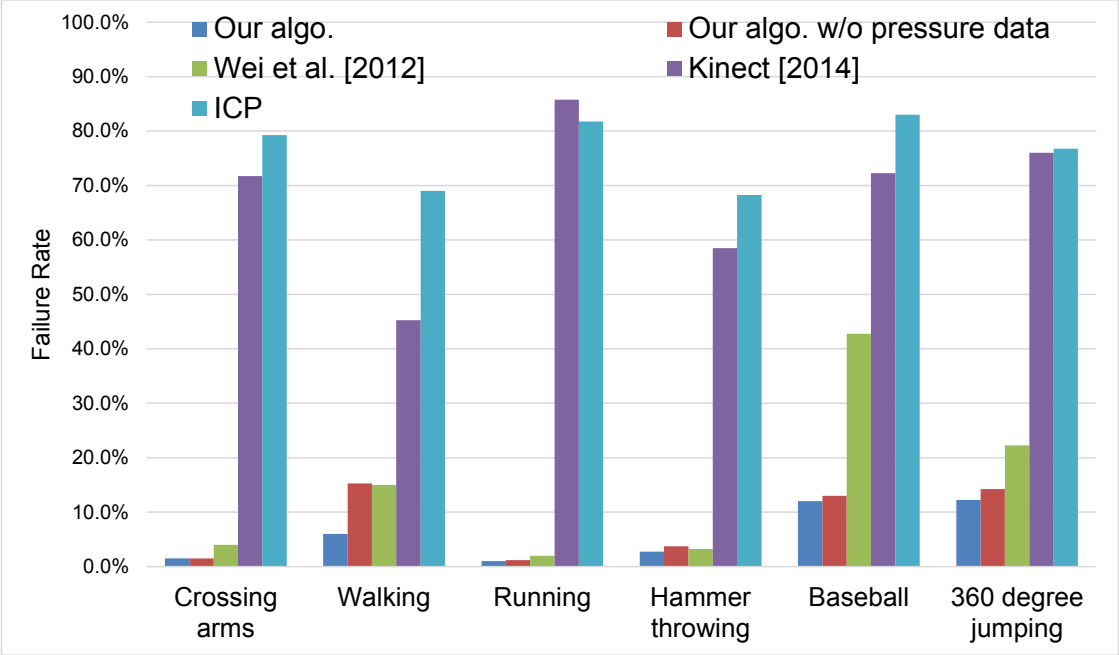


Figure II.11: Evaluation of system robustness (percentage of frames whose average reconstruction error is larger than 6 *cm*) on six test sequences.

II.7.4.1 Importance of the Boundary Term

We evaluate the importance of the boundary term by comparing the results with and without this term. Figure II.12 clearly shows the importance of the boundary term.

II.7.4.2 Importance of the Pressure Data/Ground Penetration Term

Figure II.13 shows a side-by-side comparison with and without the pressure data/ground penetration term. The term is critical to our system for two reasons. First, it enables us to remove foot skating artifacts and avoid the ground penetration issue in the reconstructed kinematic motion. Second, it significantly reduces the reconstruction ambiguity of full-body dynamics.

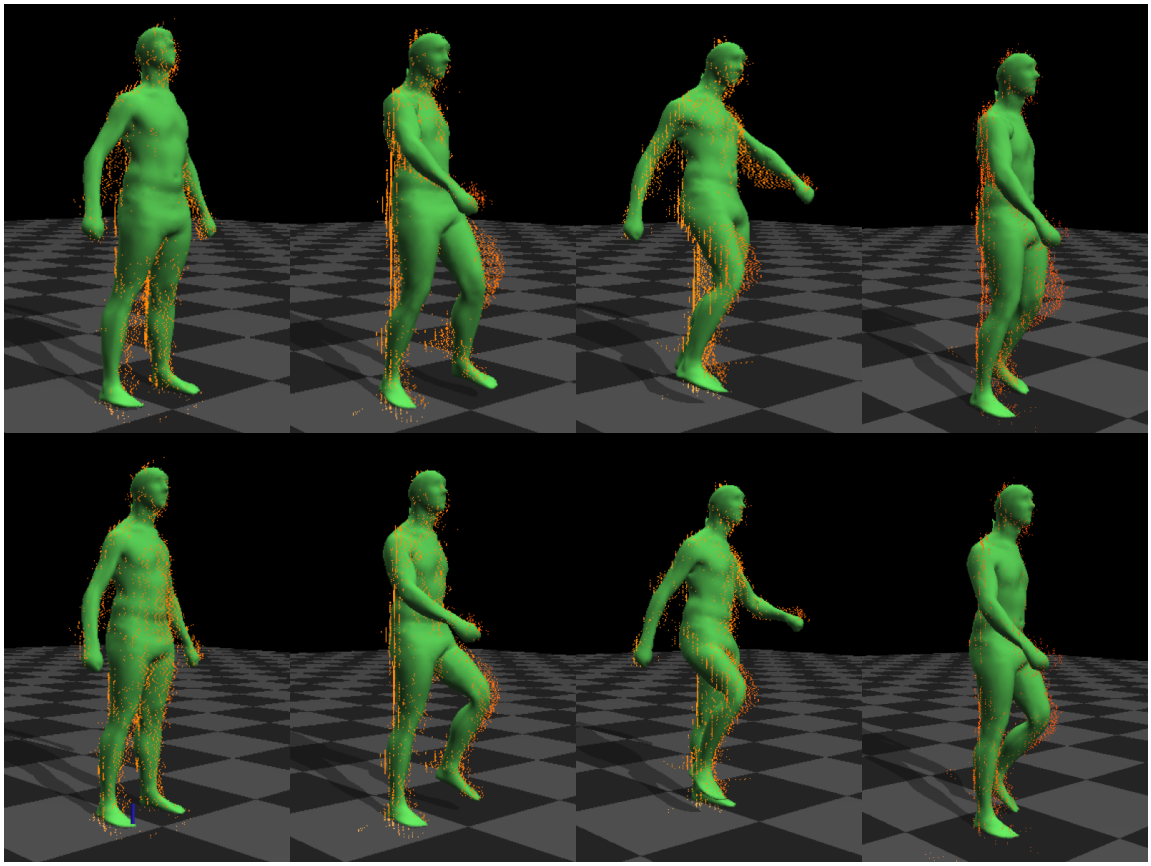


Figure II.12: Importance of the boundary term. (top) result without the boundary term; (bottom) result with the boundary term.

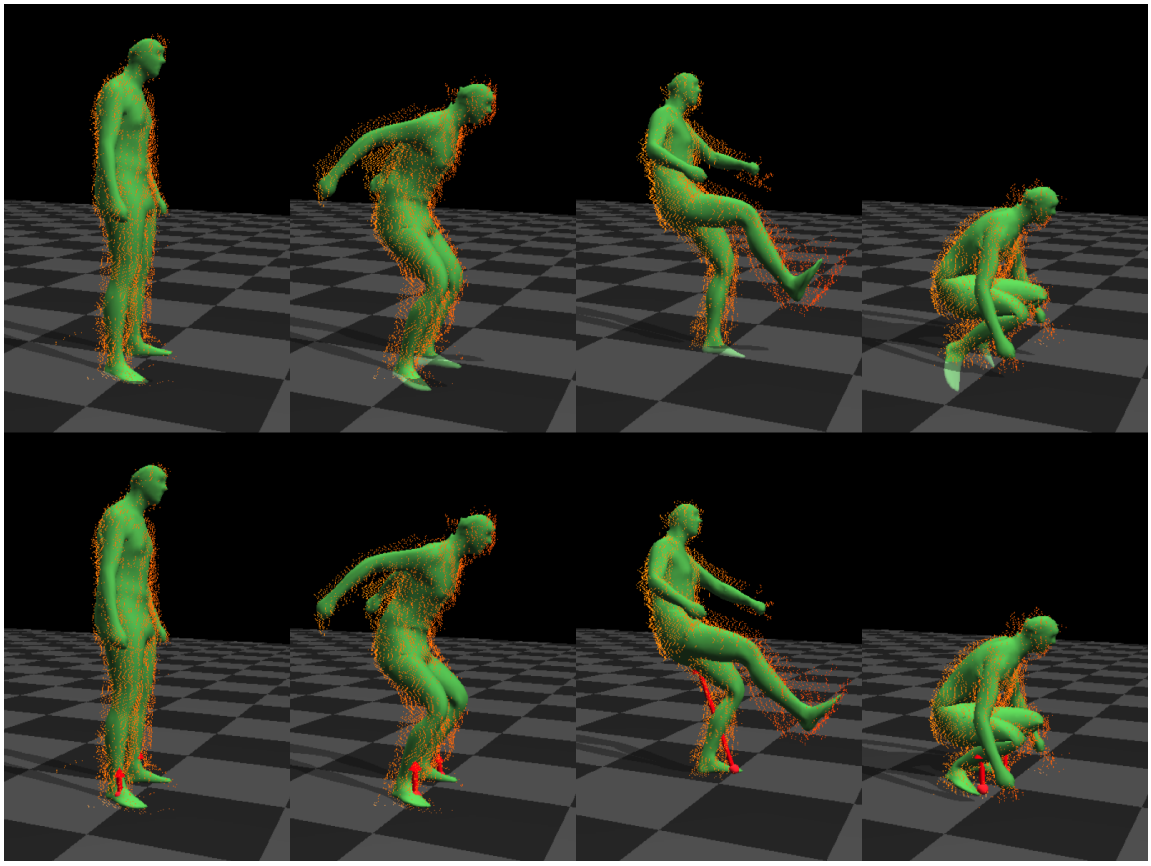


Figure II.13: Importance of the pressure data/ground penetration term. (top) result without the pressure data/ground penetration term; (bottom) result with the pressure data/ground penetration term.

II.7.4.3 Importance of the Prior Term

Figure II.14 shows a side-by-side comparison with and without the prior term. The use of the prior term improves the reconstruction accuracy of full-body poses, particularly the torso part in this example.

II.7.5 Comparison against Vicon and Force Plates

We have validated the effectiveness of our dynamic data capture process by comparing the reconstructed internal torques with those obtained from a twelve-camera *Vicon* system in a full marker set and force plates. We capture 120 walking sequences using the *Vicon* system and force plates and reconstruct the internal joint torques based on the recorded force data from force plates and the full-body kinematic motion data obtained from the *Vicon* system via inverse dynamics technique. Figure II.15 (a) plots internal joint torques of the left knee from 120 walking sequences (blue curve). We repeat the captured motion five times and extract the joint torque patterns of the left knee by temporally aligning and averaging 120 sequences (red curve in Figure II.15 (a)).

We capture a walking sequence of a different subject using our full-body kinematics and dynamics capture system. Figure II.15 (b) shows a plot of internal joint torque of the left knee for a single walking cycle of the reconstructed dynamic data (blue curve). The figure shows that our reconstruction data (blue curve) has very similar patterns as those (red curve) obtained from the *Vicon* system and force plates.

II.8 Discussion

In this chapter, we have developed an end-to-end full-body motion capture system using input data captured by three depth cameras and a pair of pressure-sensing shoes. Our system is appealing because it is low-cost and fully automatic, and can accurately reconstruct full-body kinematics and dynamics data. The system is also non-intrusive and

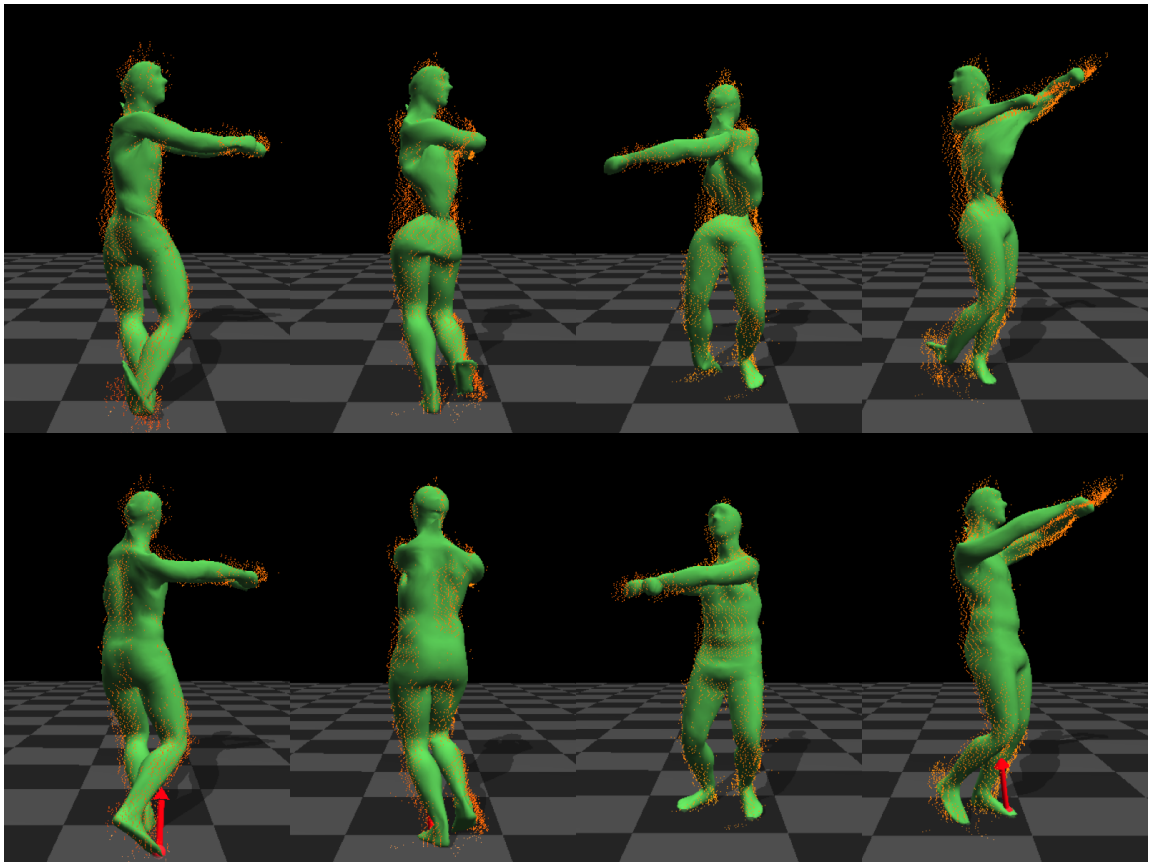
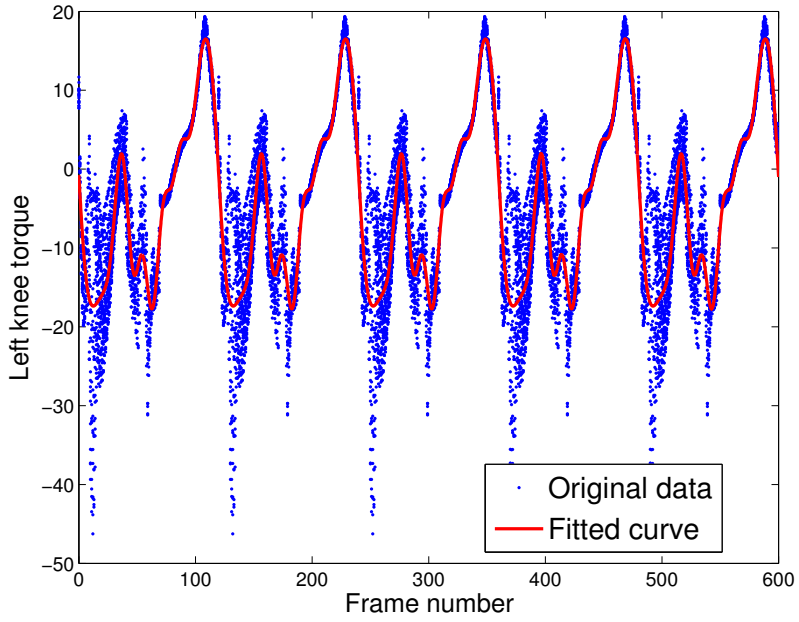
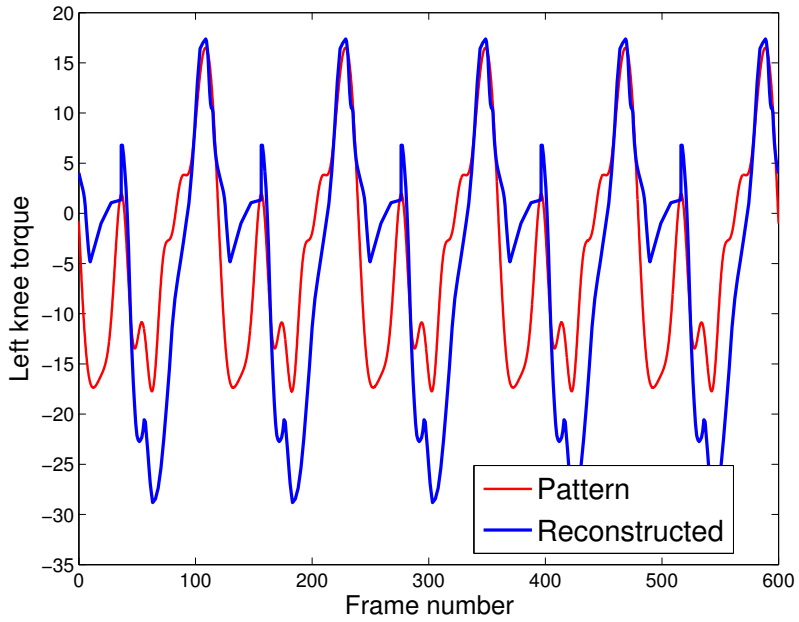


Figure II.14: Importance of the prior term. (top) result without the prior term; (bottom) result with the prior term.



(a)



(b)

Figure II.15: Validation of reconstructed dynamic data. (a) internal torque patterns (red curve) obtained by temporally aligning and averaging 120 walking sequences captured by the *Vicon* system and force plates; (b) internal joint torques patterns (blue curve) from our result superimposed on internal torque patterns (red curve) obtained from the *Vicon* system and force plates.

easy to set up because it requires no markers and no special suits. We have demonstrated the power of our approach by capturing a wide range of complex human movements. The system achieves state-of-the-art accuracy in our comparison against alternative methods.

Complementing depth data with pressure data not only improves the accuracy and robustness of the kinematic tracking process but also enables us to automatically capture and reconstruct full-body poses, joint torques, and contact forces all at once. The current system is based on three depth cameras and our own version of prototype pressure sensors. Our framework, however, is flexible and is not limited to particular types of sensors. For example, any pressure sensor commercially available (*e.g.*, Tekscan F-Scan [1]) could be plugged into our system. We could also replace three depth cameras with a single consumer-level video camera to acquire motions difficult to capture in the lab, such as a run on the beach or a boxing match.

We choose to reconstruct human body kinematic data and dynamic data in a sequential manner because we are focused on online applications. An alternative solution is to use batch-based optimization [25] to reconstruct kinematics and dynamics data for a certain period of time. For our application, however, batch-based optimization is very time consuming and memory-intensive because it requires solving a complex non-linear optimization with a huge number of constraints. We have also chosen to sequentially reconstruct kinematics and dynamics data because kinematic motion data obtained from the tracking process are often highly accurate and often sufficient to reconstruct the dynamics data. If the kinematic motion data are not reliable, a better solution is to use all the observed data, along with physical constraints, to simultaneously optimize kinematic and dynamic variables. This inevitably requires solving a more challenging optimization problem and certainly will slow down the entire reconstruction process.

Our full-body shape modeling process enables our system to work for human subjects of different body sizes and proportions. In the future, we would like to include more body

scans into the training data sets to improve the generalization ability of our parametric model, as the current training data sets are still not sufficient to model shape variations across all the human subjects. Another way to improve the accuracy and robustness of the system is to combine depth data with color image data. We are particularly interested in incorporating color and texture information obtained from a video camera into the current tracking framework.

Our system often fails to produce good results when a large portion of depth data is missing (*e.g.*, when a large part of the body is out of the camera range) or when significant occlusions occur (*e.g.*, when the hands are extremely close to the torso that it cannot be distinguished from the subject's torso). Another limitation of the current system is that it can only capture contact phenomena between feet and the ground. The current system is not suitable to capture motion with complex contact phenomena such as falling down to the ground and rolling on one's back. In the future, we wish to explore how to capture full-body kinematics and dynamics data for these kinds of motions. We are also interested in extending the current system to capture interactions between multiple subjects.

We believe the new type of data captured by our system will provide insights into designing controllers for simulated virtual humans and biped robots, as well as extending our current biomechanics knowledge in motor control. In particular, the captured kinematics and dynamics data could be leveraged for many applications in human motion processing, analysis and synthesis, such as motion filtering, motion editing, motion registration, and physics-based motion control and optimization. For example, the motion can be cleaned to remove noise at the level of the driving signal (joint torques), it can be more accurately edited to meet new constraints, it would allow us to register the motion more accurately using both kinematics and dynamics data, it can serve as a basis for development of control algorithms for human movement, and it can be used to build much more precise models to predict how human takes a compensatory step to maintain the balance. One of the imme-

diate directions for future work is, therefore, to investigate the applications of the captured data to human motion analysis, synthesis and control.

CHAPTER III

A DATA-DRIVEN GENERATIVE SKINNED MESH MODEL FOR ACCURATE AND ROBUST 3D HAND MODELING

This chapter aims to construct a 3D parametric model for human hands. Such a model has many important applications in computer graphics and animation. For example, it can be applied to tracking hand motion from videos or depth images, modeling natural-looking hands from various types of user constraints such as depth data, and completing partial 3D hand scans. One appealing solution to this problem is to construct data-driven parametric hand models from 3D hand scans. Data-driven parametric hand models are advantageous for hand modeling and synthesis because they are very compact and they can be used to generate an infinite number of natural-looking hand models that are not in scan database.

Our core idea is to construct a low-dimensional parametric model that compactly represent hand shape variations across individuals and enhance it by adding Linear Blend Skinning (LBS) for pose deformation. Mathematically, we model a 3D hand mesh model by $H(\alpha, \beta, \mathbf{q}; \mathbf{W})$, where the shape parameters α, β provides a low-dimensional representation of hand shape variations across individuals, the pose parameter $\mathbf{q}$ specifies the joint angle values of the 3D hand pose, and $\mathbf{W}$ represents the skinning weights required for skinning deformation. Our parametric model provides a continuous and compact representation for allowable shape variations across different human subjects, but is specific enough not to allow arbitrary variations that are not similar to those seen in the database. With this parametric model, we could randomly sample the parameters α, β and $\mathbf{q}$ to generate an infinite number of natural-looking hand models in different shapes and under different poses. Furthermore, we could choose the parameters so that the model would match various forms of user input.

We propose to learn the parametric hand model directly from a large set of hand scans. To learn the parametric model, we formulate an optimization problem and introduce an iterative method to find the optimal solution. We estimate the shape parameters and poses for each subject using an initial skinning weights, and then update the skinning weights based on all the estimated subject-specific hand parameters. This process is repeated and finally we extract the low-dimensional hand model.

We have demonstrated the power and flexibility of our parametric model in a variety of applications, ranging from 3D hand modeling based on various forms of input constraints, including depth images obtained by depth sensors, incomplete scan data, a small set of color images, and semantic constraints defined by the user, to transferring skinning weights to a pre-existing hand model and model-based hand tracking using a single depth camera. We assess the accuracy and effectiveness of our model via cross validation. We validate our model by evaluating the key components of our model.

The contributions of this chapter are summarized as follows:

- A compact parametric hand model that accurately models geometric variations in hand shapes and poses across individuals.
- An iterative optimization approach that efficiently learns the parametric hand model from a large unaligned hand scan database.
- A wide range of applications, including 3D hand modeling using depth images, incomplete scans, semantic constraints, and color images, skinning weights transfer, and model-based 3D hand tracking.

III.1 Background

Our work focuses on learning a generative hand model that allows for rapid construction of a subject-specific skinning hand model from various inputs. To achieve this goal,

we learn a low-dimensional parametric skinning hand model for both shape and pose deformation using a large database of high-quality scanning mesh models of human hands. We summarize the related works as follows.

We first discuss hand model representation. To represent a human hand, one possibility is to use simple geometric primitives to approximate hand geometry (e.g., [39, 40]). A more efficient way for 3D hand modeling is to use a skinned mesh model with Linear Blend Skinning (LBS) for pose deformation as done in many hand tracking algorithms (e.g., [41, 42, 43, 44]). More advanced models (e.g., [45, 46, 47]) are also proposed to generate a more plausible or realistic hand model for pose deformation. However, they only account for shape variations induced by pose deformation for a specific subject.

Next we discuss pose deformation modeling. A common method for pose deformation is LBS [30], which is widely supported in industry. A large number of LBS-based techniques (e.g., [48, 49, 50]) are proposed to reduce artifacts caused by LBS and achieve better deformation results. One possible way to correct pose deformation artifacts is to use vertex offsets extracted from multiple example meshes (e.g.,[48, 49]). The offsets are applied in local coordinates of the skeleton to the template mesh, and the amount of offsets to apply is computed based on the pose distance to the examples using interpolation. Another possibility to reduce artifacts is to use scaling. Mohr and his colleagues [50] extend LBS by adding additional rotation and scaling joints in the skeleton to get more realistic deformation based on example meshes. Jacobson and Sorkine [51] further improves the deformation quality by using endpoint weight functions on the skeleton instead of applying scaling directly. This also allows the algorithm to model some shape variations such as arm lengths across individuals in a more visually appealing way. However, the shape variations this method could model is highly limited. Our method is different that, unlike these works that either use vertex offsets or scaling for pose corrections for a single subject, we combine them together to model shape variations across subjects, and our model

becomes a standard LBS model when the shape parameters are fixed.

In the following we discuss about statistical shape modeling. One way to build a subject-specific hand model is by deforming a reference model to fit the input constraints [52, 53]. But this is usually time-consuming and may not work well due to missing or noisy input.

A more appealing solution is to use data-driven methods to build statistical or generative models from database to represent shape and pose variations across subjects, an area that has been extensively explored for human body shape modeling. Allen et al. [54] uses vertex offsets to model shape variations across subjects in a same pose by applying Principal Component Analysis to the aligned meshes. Pose variations are not modeled so extra steps are needed for pose deformation. SCAPE [55] and its successors (e.g., [56, 38, 57]) use transformation matrices on triangles to model shape variations across subjects, pose-dependent shape variations, as well as rigid-body pose deformation. These models are redundant as they use transformation matrices to model variations. In addition, they do not use an explicit skeleton for rigid-body deformation hence more complicated translational or rotational-invariant encodings of triangles are needed and thus not suitable for real-time applications such as model-based tracking. In contrast, we use a more compact but still powerful representation to model variations, including skeleton scales and vertex offsets for shape deformation, as well as a skeleton for rigid pose deformation, which excludes a large number of unnatrual shape and poses and will make the optimization easier.

Methods based on LBS for shape modeling are also proposed (e.g., [58, 59]) whose goals are to generate model fast and to be compatible with existing software. SMPL [59] uses a model based on blend skinning (Linear blend skinning or Dual-Quaternion blend skinning) to represent identity-dependent and pose-dependent shape variations, as well as rigid pose deformation. Similar to our method, they apply the shape deformations to the template mesh, which makes the model compatible with LBS. However, we model

the identity-dependent shape variations in a more compact and expressive way that we incorporate skeleton scales on top of vertex offsets for shape deformation, while they only use vertex offsets similar to Allen et al. [54] for this purpose.

More specific to hands, Khamis et al. [60] and Tan et al. [61] propose a linear model to represent hand shape variations using vertex offsets with a global scale and use LBS for pose deformation, and learn the model from low-resolution depth images. Our model is different and more expressive because we decouple shape variations into skeleton scale variations and vertex offset variations and model each of them using a low-dimensional model. In addition, we construct the parametric model from high-quality scanning mesh models annotated with a large set of point correspondences rather than low-resolution unlabeled depth images, thereby significantly improving the resolution and accuracy of our parametric mesh model.

We also discuss about database registration and model learning. In order to learn statistical models from a scan database, a common first step is to register each scan with a template mesh to bring them in correspondences. This could be done by non-rigid template fitting (e.g., [54, 38]) based on manually labeled landmarks for each of the scan. After registration, the aligned meshes are used for model learning. However, registering scans individually may lead to inconsistent correspondences between subjects and inconsistent shapes for the same subject in different poses due to missing data, scan noise or inaccurate landmark labeling, which is undesired for shape modeling.

Similar to Hirshberg et al. [57] that learns the shape model and does the registration simultaneously, we estimate the shape and poses for each subject using multiple scans directly at the same time, and learn the parametric model from all these parameters. Comparing to their method that optimizes for transformation matrices directly, our model optimizes for the underlying shape and pose parameters that are more compact and explicit, has less parameters, but still expressive, and thus easier to optimize and harder to get

undesired deformations.

Finally we discuss about automatic skinning. Various algorithms (e.g., [62, 4, 63, 64, 65]) have been proposed to automatically learn skinning weights for LBS models. Dionne et al. [4] uses a single mesh and skeleton to learn the skinning weights based geodesic distance. Methods using a set of mesh examples of the same shape to learn a skeleton and skinning weights are also proposed (e.g., [63, 64]). Recent methods [66, 67] impose additional constraints such as sparseness, orthogonal, convex or soft joint constraints during skinning weights optimization, and learn them from different poses of a same subject. In contrast, we learn the skinning weights from all subjects under different poses and joint constraints are automatically enforced since the skeleton is incorporated in our hand model to represent both shape and pose variations.

III.2 Overview

Our goal is to develop an efficient system for human hand modeling. For this purpose, we construct a low-dimensional parametric hand model $H(\alpha, \beta, \mathbf{q}; \mathbf{W}, U)$ from a large set of scan meshes and use it to create natural-looking hand mesh models with skeleton and skinning weights embedded.

Our system consists of the following major components:

- Model representation. We model 3D hands by constructing a low-dimensional parametric model that compactly represents hand shape variations across individuals, and enhance it by adding Linear Blend Skinning (LBS) for pose deformation. The shape variations are modeled using skeleton scales and vertex offsets in low-dimensional spaces that is compact but more expressive. Our parametric model provides a continuous and compact representation for allowable shape variations across different human subjects, but is specific enough not to allow arbitrary variations that are not similar to those seen in the database.

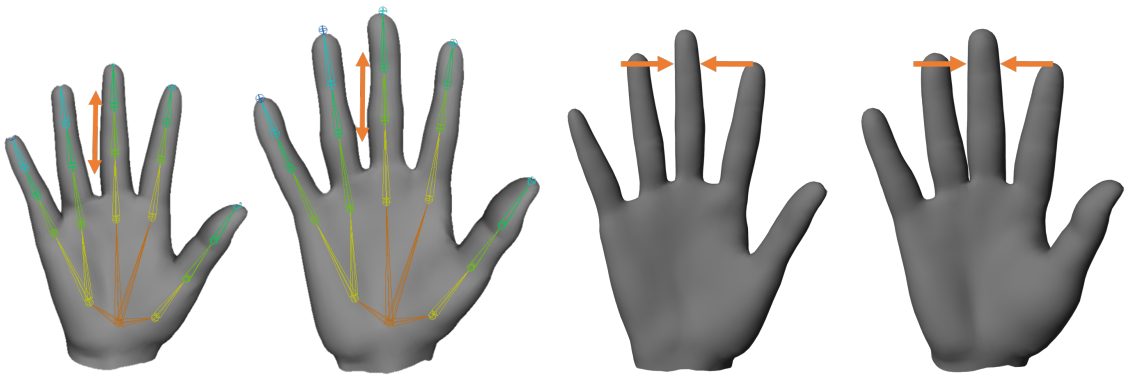
- Model learning. We formulate the model learning problem as an optimization and propose an iterative method to find the solution. Based on an initial skinning weights, we estimate the shape and poses for each subject, and then update the skinning weights based on the estimated shape and poses. We repeat this process and extract the low-dimensional shape model from the estimated shapes. Unlike previous work, we learn the model from scan database directly and estimate the shape parameters using meshes from multiple poses simultaneously.

- Applications. Based on our parametric hand model, we build various applications. We reconstruct user-specific hand models from different inputs, including depth data, incomplete scans, color images and semantic measurements. The reconstructed user-specific hand model could then be used for other applications such as model-based hand tracking. We could also make a static hand model deformable by transferring the skeleton and skinning weights.

III.3 Model Representation

We model 3D hands by constructing a low-dimensional parametric model using a hand shape model and a hand pose model. The hand shape model compactly represents hand shape variations across individuals, and it is enhanced by Linear Blend Skinning (LBS) for pose deformation using the pose model. Mathematically, we model a 3D hand mesh model by $H(\alpha, \beta, \mathbf{q}; \mathbf{W}, U)$, where the shape parameters α, β provides a low-dimensional representation of hand shape variations across individuals, the pose parameter $\mathbf{q}$ specifies the joint angle values of the 3D hand pose, $\mathbf{W}$ represents the skinning weights required for skinning deformation, and U is the bases for shape parameters.

Our parametric model provides a continuous and compact representation for allowable shape variations across different human subjects, but is specific enough not to allow arbitrary variations that are not similar to those seen in the database.



(a) (b)

Figure III.1: Hand shape variations. (a) skeleton size variation; (b) More subtle shape variation.

III.3.1 Hand Shape Model

Hand shape model captures shape variations across subjects under identical pose, which could be modeled using skeleton scales and vertex offsets. Skeleton scales describe the overall shape of the subject, and vertex offsets models more subtle variations. They are modeled based on the template mesh $\hat{M}$ and skeleton $\hat{K}$ that the mesh after shape deformation could be used for pose deformation.

III.3.1.1 Skeleton Scales

It is obvious to see that skeleton size varies among subjects, and the shape of a subject could be largely determined by its skeleton size, as shown in Figure III.1 (a). For a given template skeleton $\hat{K}$ with bone sizes $\hat{\mathbf{B}} = [\hat{b}_0, \hat{b}_1, \ldots, \hat{b}_{n-1}]$, we represent the skeleton size of a new subject $\mathbf{B} = [b_0, b_1, \ldots, b_{n-1}]$ using scales $\mathbf{S} = [s_g, s_0, s_1, \ldots, s_{n-1}]$ by

$$b_i = s_g \cdot s_i \cdot \hat{b}_i, \tag{III.1}$$

where s_g is the overall scale and s_i is the scale for bone i.

To apply the scale to the vertex, we use a method similar to Linear Blend Skinning except that we scale the vertices in the local coordinate before transforming it back to the world coordinate. That is, for a given mesh, we first transform each vertex to the local coordinates of corresponding bones, scale them by the overall scale s_g uniformly, and then scale again only in the direction of the bone based on bone scales s_i. After that we transform the scaled vertices to world coordinate and compute the final vertex position using linear combination based on the skinning weights. Note that we treat the global scale s_g separately since we apply s_g on all axises uniformly but apply the bone scales only in the direction of the bones. To summarize, for a given vertex $\mathbf{v}_i$ in the template, the scaled vertex in local coordinate of the jth bone $\mathbf{v}'_{ij}$ could be expressed as:

$$\mathbf{v}'_{ij} = \mathbf{S} \otimes (\hat{T}_j^{-1} \mathbf{v}_i) = \mathbf{S} \otimes \mathbf{v}_{ij}, \tag{III.2}$$

where $\hat{T}_j$ is the transformation matrix to the world coordinate of bone j under the rest pose $\mathbf{q}_{rest}$ of the template skeleton $\hat{K}$, $\mathbf{v}_{ij} = \hat{T}_j^{-1} \mathbf{v}_i$ gives the local coordinate of vertex $\mathbf{v}_i$ in bone j, and $\mathbf{S} \otimes \mathbf{v}_{ij}$ applies the scales $\mathbf{S}$ to $\mathbf{v}_{ij}$ as follows:

$$\mathbf{S} \otimes \mathbf{v}_{ij} = s_g s_j (\mathbf{v}_{ij} \cdot \mathbf{d}_j) \mathbf{d}_j + s_g (\mathbf{v}_{ij} - (\mathbf{v}_{ij} \cdot \mathbf{d}_j) \mathbf{d}_j), \tag{III.3}$$

where $\mathbf{d}_j$ is the direction of bone j, and $s_j (\mathbf{v}_{ij} \cdot \mathbf{d}_j) \mathbf{d}_j$ and $\mathbf{v}_{ij} - (\mathbf{v}_{ij} \cdot \mathbf{d}_j) \mathbf{d}_j$ are the components of $\mathbf{v}_{ij}$ parallel to and perpendicular to jth bone's direction.

Although the skeleton scales $\mathbf{S}$ models the structure accurately, it is a redundant representation since the scale between bones are highly correlated. For example, if the proximal phalanx of a finger is longer than the template, it is very likely that the intermediate and distal phalanges of that finger will also be longer. To remove redundant, we apply Principal Component Analysis (PCA) to the scales $\mathbf{S}$ to get a compact representation:

$$\mathbf{S}(\alpha) = \mathbf{S}_0 + \mathbf{C}_\mathbf{S}\alpha, \tag{III.4}$$

where $\mathbf{S}_0$ is the mean skeleton scales, $\mathbf{C}_\mathbf{S}$ is the coefficient matrix of principal components, and α is the low-dimensional shape parameter.

III.3.1.2 Vertex Offsets

To model subtle details such as palm and finger thicknesses of the subject, we use vertex offsets $\mathbf{O} = [O_0^x, O_0^y, O_0^z, O_1^x, \ldots, O_{|\hat{M}|-1}^z]$, i.e., the displacements between the subject and the template mesh for this purpose, as shown in Figure III.1 (b). To make the offsets invariant, we model them in the coordinate of the template $\hat{M}$ under rest pose, which eliminates the skeleton sizes and pose variations for different subjects. That is, we apply the offsets to the template before scaling the vertices using Equation (III.2), which could be expressed as:

$$\mathbf{v}_i = \hat{\mathbf{v}}_i + \mathbf{O}_i, \tag{III.5}$$

where $\mathbf{v}_i$ is the ith vertex with offset, $\hat{\mathbf{v}}_i$ is the vertex i on the template mesh $\hat{M}$, and $\mathbf{O}_i = [O_i^x, O_i^y, O_i^z]$ is the offset vertex.

Similar to skeleton scales, the vertex offsets are highly redundant as neighboring vertices in the mesh usually have similar displacements. Again, we apply PCA to the vertex offsets $\mathbf{O}$ from all subjects to get a compact representation as:

$$\mathbf{O}(\beta) = \mathbf{O}_0 + \mathbf{C}_\mathbf{O}\beta, \tag{III.6}$$

where β is the low-dimensional shape parameters, $\mathbf{O}_0$ is the mean offsets and $\mathbf{C}_\mathbf{O}$ is the shape basis.

III.3.1.3 Shape Model

To summarize, we represent the ith vertex in local coordinates of the corresponding bone j for a given shape α and β as :

$$\mathbf{v}'_{ij}(\alpha,\beta) = \mathbf{S}(\alpha) \otimes (\hat{\mathbf{T}}_j^{-1}(\mathbf{O}_i(\beta) + \hat{\mathbf{v}}_i)) \tag{III.7}$$

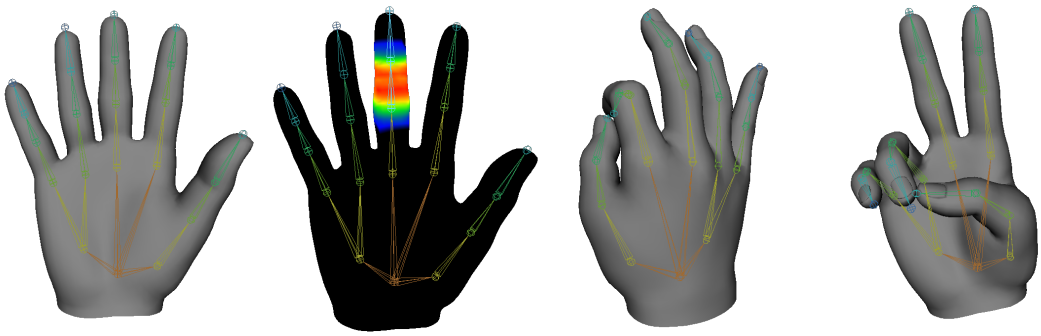
where $\hat{\mathbf{v}}_i$ is the ith vertex of the template mesh $\hat{M}$, $\mathbf{Q}_i(\beta)$ is the ith offset vertex computed from Equation (III.6), $\hat{\mathbf{T}}_j$ is the transformation matrix of bone j of the template skeleton $\hat{K}$, $\mathbf{S}(\alpha)$ is the skeleton scales defined in Equation (III.4) and $\otimes$ is the scaling operator in Equation (III.3).

III.3.2 Hand Pose Model

For a specific hand mesh model with a skeleton and skinning weights attached, we use LBS for pose deformation, as shown in Figure III.2. LBS defines how each vertex of the template mesh $\mathbf{M}_s$ deforms according to pose $\mathbf{q}$ based on the underlying skeleton $\mathbf{K}_s$ and skinning weights W for subject s. The vertex $\mathbf{v}_i$ after pose deformation is described as:

$$\begin{aligned} \mathbf{v}_i(\mathbf{q}) &= \sum_{j=0}^{n-1} w_{ij} \mathbf{T}_j(\mathbf{q}) \hat{\mathbf{T}}_j^{-1} \hat{\mathbf{v}}_i \\ &= \sum_{j=0}^{n-1} w_{ij} \mathbf{T}_j(\mathbf{q}) \hat{\mathbf{v}}_{ij}, \end{aligned} \tag{III.8}$$

where $\hat{\mathbf{v}}_i$ is the ith vertex of the template mesh M_s, $\hat{\mathbf{T}}_j$ is the transformation of bone j under rest pose for the template skeleton, $\hat{\mathbf{v}}_{ij} = \hat{\mathbf{T}}_j^{-1} \hat{\mathbf{v}}_i$ gives the local coordinate of vertex i in bone j, $\mathbf{q}$ is the pose for deformation, $\mathbf{T}(\mathbf{q})_j$ is the transformation of the jth bone for pose $\mathbf{q}$, and $\mathbf{W} = [w_{ij}]$ is the skinning weights.



(a) (b) (c) (d)

Figure III.2: Pose modeling. (a) template mesh and skeleton in rest pose; (b) skinning weights for one of the bones (hotter colors for larger weights); (c)-(d) new meshes generated by different joint angle pose $\mathbf{q}$.

III.3.3 Parametric Hand Model

By combining the shape and pose model in Equations (III.7) and (III.8), we have a low-dimensional parametric hand model $H(\alpha,\beta,\mathbf{q};\mathbf{W},U)$ representing by shape parameters α and β, and pose $\mathbf{q}$ as:

$$\begin{aligned}\mathbf{H}_i(\alpha,\beta,\mathbf{q};\mathbf{W},U) &= \sum_{j=0}^{n-1} w_{ij}\mathbf{T}_j(\mathbf{S}(\alpha),\mathbf{q})\mathbf{v}'_{ij}(\alpha,\beta),\\ \mathbf{v}'_{ij}(\alpha,\beta) &= \mathbf{S}(\alpha)\otimes(\hat{\mathbf{T}}_j^{-1}(\mathbf{O}_i(\beta)+\hat{\mathbf{v}}_i)),\end{aligned} \tag{III.9}$$

where $\mathbf{H}_i(\cdot)$ is the ith vertex of the model, $U=\{\mathbf{S}_0,\mathbf{C}_\mathbf{S},\mathbf{O}_0,\mathbf{C}_\mathbf{O}\}$ is the shape bases defined in Equations (III.4) and (III.6). $\hat{\mathbf{v}}_i$ is the ith vertex of the template mesh $\hat{M}$, $\mathbf{O}_i(\beta)$ is the ith offset vertex computed from Equation (III.6). $\mathbf{S}(\alpha)$ is the skeleton scales defined in Equation (III.4) and $\otimes$ is the scaling operator in Equation (III.3). $\hat{\mathbf{T}}_j$ is the transformation matrix of bone j of the template skeleton $\hat{K}$, $\mathbf{T}_j(\mathbf{S},\mathbf{q})$ is the transformation of bone j to world coordinate based on the skeleton scaled by $\mathbf{S}$.

The parameters of our low-dimensional parametric model, including statistical rep-

resentations of shape variations U and skinning weights $\mathbf{W}$, are all learned from a scan database (Section III.4). These parameters are fixed after learning so the model could also be written as $H(\alpha,\beta,\mathbf{q})$ or $H(\alpha,\beta,\mathbf{q};\mathbf{W})$ for brevity.

Using this model, we could generate natural-looking skinned hand meshes under different poses by randomly sampling from the low-dimensional shape and pose parameters α, β and $\mathbf{q}$. Our parametric model provides a continuous and compact representation for allowable shape variations across different human subjects, but is specific enough not to allow arbitrary variations that are not similar to those seen in the database.

If the skeleton scales $\mathbf{S}$ and vertex offsets $\mathbf{O}$ are known, we could also represent this high-dimensional hand model as follows:

$$\begin{aligned}\tilde{\mathbf{H}}_i(\mathbf{S},\mathbf{O},\mathbf{q};\mathbf{W}) &= \sum_{j=0}^{n-1} w_{ij}\mathbf{T}_j(\mathbf{S},\mathbf{q})\tilde{\mathbf{v}}'_{ij}(\mathbf{S},\mathbf{O}),\\ \tilde{\mathbf{v}}'_{ij}(\mathbf{S},\mathbf{O}) &= \mathbf{S}\otimes(\hat{\mathbf{T}}_j^{-1}(\mathbf{O}_i+\hat{\mathbf{v}}_i)).\end{aligned} \tag{III.10}$$

III.4 Model Learning

In this section, we propose a new algorithm to learn the shape bases $U=\{\mathbf{S}_0,\mathbf{C}_\mathbf{S},\mathbf{O}_0,\mathbf{C}_\mathbf{O}\}$ and the skinning weights $\mathbf{W}$ from a large, unaligned scan database.

Our goal is to learn the shape bases U and the skinning weights $\mathbf{W}$ so that we could reconstruct the scan database with minimal error. Since U are extracted from the skeleton scales $\{\mathbf{S}^s\}$ and vertex offsets $\{\mathbf{O}^s\}$ of all subjects, we first estimate them for each subject and then extract the bases afterward. We formulate the following problem to estimate these parameters:

$$\min_{\{SP^s\},\mathbf{W}} \sum_s \sum_t \sum_{(i,j)\in C^{st}} \left\|\tilde{\mathbf{H}}_i(\mathbf{S}^s,\mathbf{Q}^s,\mathbf{q}_t^s;\mathbf{W}) - Y_j^{st}\right\|^2, \tag{III.11}$$

where the subject-specific hand parameters $SP^s = \{\mathbf{S}^s, \mathbf{O}^s, \{\mathbf{q}_t^s\}\}$ are the skeleton scales $\mathbf{S}^s$, vertex offsets $\mathbf{O}^s$ and poses $\{\mathbf{q}_t^s\}$ for subject s and $\mathbf{W}$ is the skinning weights. $\tilde{\mathbf{H}}_i(\cdot)$ is the ith vertex of the generated mesh defined in Equation (III.10), Y_i^{st} is the ith vertex of the tth scan for subject s, and C^{st} is the set of corresponding vertex index pairs between the generated mesh and the scan.

This problem is a nonlinear optimization that is not easy to solve directly. Firstly, it contains a large number of parameters since we need to estimate all shape and poses parameters for all subjects together due to the skinning weights, which is infeasible for a large high-resolution database. Secondly, the correspondences C to the scan meshes are unknown so step to estimate the correspondences is needed.

To address the aforementioned issues, we propose an iterative method to estimate the subject-specific parameters and the skinning weights individually based on each other. During each iteration, we also update the correspondences based on previous results to get better matches. This iterative method reduces the computational cost significantly since we could learn the subject-specific parameters for each subject independently.

III.4.1 Learning Process Overview

The learning process is summarized in Figure III.3. We break the learning process into four components, and then estimate the parameters iteratively. We first initialize the skinning weights $\mathbf{W}$ and the correspondences C^{st} for each scan. Then we estimate the subject-specific parameters SP^s independently for all subjects. The skinning weights and the correspondences are then updated based on the estimated parameters. We repeat this process and finally extract the low-dimensional parametric model.

The main components of our algorithm are summarized as follows:

- Correspondence estimation. We estimate the correspondences between template and scan by fitting the scan using previous estimated mesh using deformation transfer,

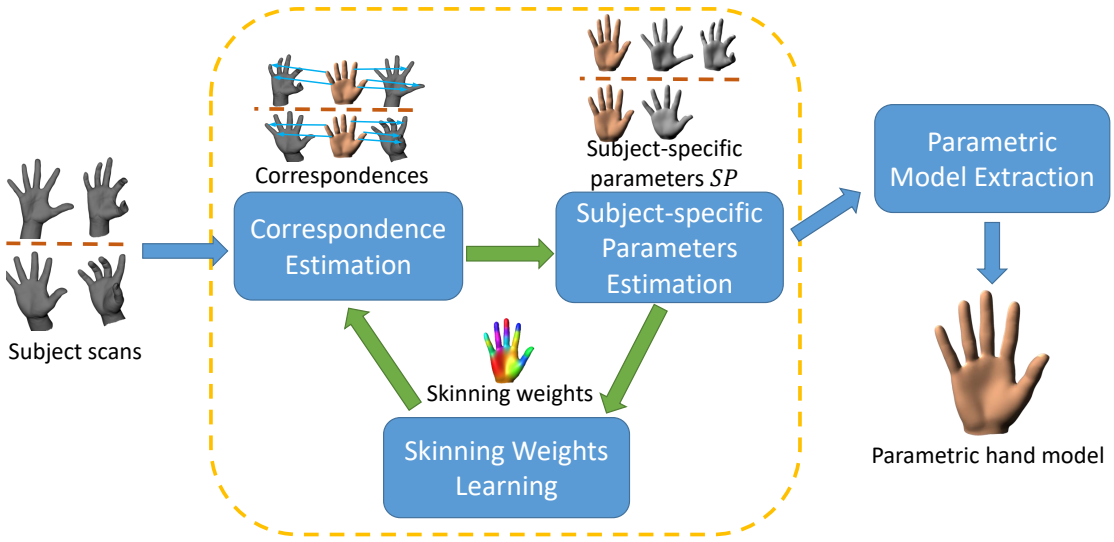


Figure III.3: Learning process overview. We break the learning process into four components and learn the model iteratively.

and then extract the correspondences based on the fitting result. Note that the final correspondences will still between the template and the scan. We update the correspondences iteratively during the learning process. This results in better matched and consistent correspondences that will be helpful for optimization in other steps since we utilize information from previous result, and the mismatched correspondences caused by fitting error could be corrected gradually.

- Subject-specific parameters estimation. We propose an algorithm to estimate the subject-specific hand parameters, including shape and poses for each subject in the scan database using our high-dimensional generative hand model. We formulate an optimization problem to minimize the reconstruction error while keeping the estimated shape smooth. We explicitly optimize for a subject-specific template mesh

and skeleton, and deforms it to fit multiple scans simultaneously. Our algorithm provides more accurate results because we utilize information from multiple scans to get consistent shapes and reduces the influences of missing or noisy data.

- Skinning weights learning. We propose an algorithm to learn the skinning weights from the scan database. We formulate an optimization problem to minimize the reconstruction error caused by skinning weights. We use all meshes from different subjects under different poses to estimate the skinning weights, which provides better skinning weights for all shapes and poses.

- Parametric model extraction. Based on subject-specific hand parameters, we could extract the shape bases U using PCA to build the low-dimensional parametric hand model.

Based on the above components, we could iteratively learn the model as follows:

- Initialization. We initialize the template mesh $\hat{M}$, skeleton $\hat{K}$ and skinning weights $\mathbf{W}$ as described in Section III.4.2.

- Model parameters estimation. We iteratively update subject-specific hand parameters SP and skinning weights $\mathbf{W}$ for several iterations:

  - Correspondence estimation. We update correspondences CI^{st} for each scan based on previous result as described in Section III.4.3.

  - Subject-specific parameters estimation. We estimate the skeleton scales $\mathbf{S}^s$, vertex offsets $\mathbf{O}^s$ and poses $\{\mathbf{q}^{st}\}$ for each subject s as discussed in Section III.4.4.

  - Skinning weights learning. We update the skinning weights as in Section III.4.5.

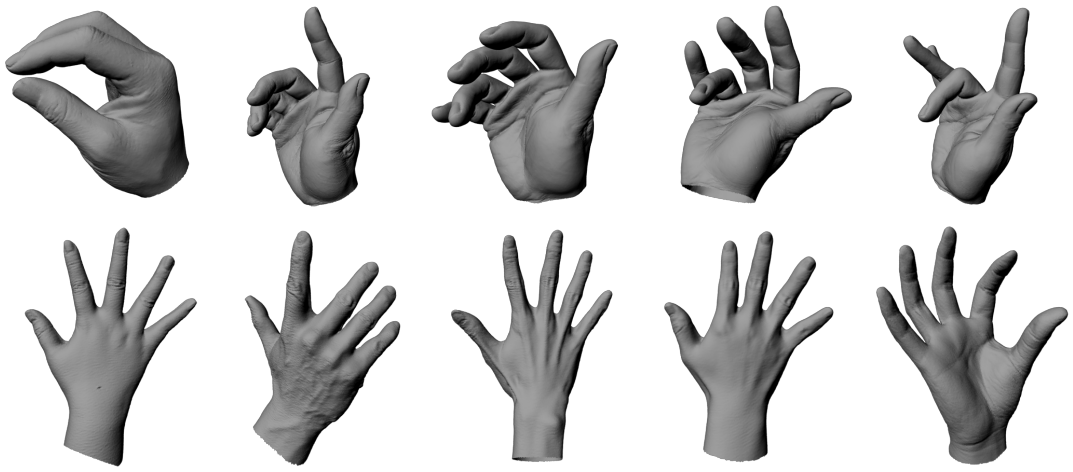


Figure III.4: Hand scan database, which includes meshes from different subjects and under different poses.

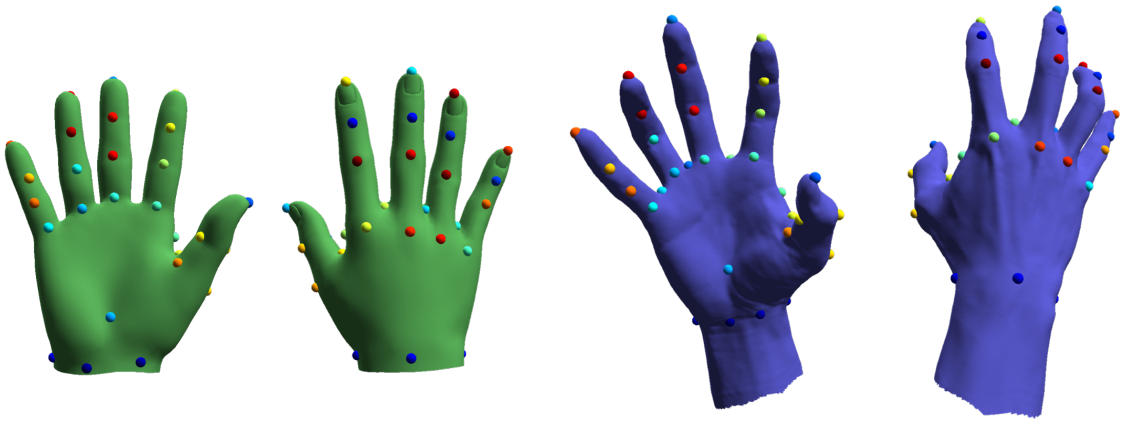
- Parametric model extraction.We extract the parametric hand model as described in Section III.4.6.

III.4.2 Data Acquisition and Preprocessing

In the following we discuss how to build the scan database as well as preprocessing steps.

III.4.2.1 Scan Database Acquisition

To build a database for model learning, we scan a large number of human right hands using an Artec Eva 3D scanner. The scanner provides dense and detailed 3D triangular meshes for each scan, containing about 70k vertices and 140k faces. We captured data from 166 subjects (103 males and 63 females). Each subject performed $1 \sim 5$ poses selected from a set of predefined poses $\{\hat{\mathbf{q}}_i\}$, resulting in 466 mesh models in total. Figure III.4 shows some examples of the database.



(a) (b)

Figure III.5: Landmark correspondences. Landmarks with identical colors are correspondences. (a) landmarks on the template mesh model; (b) corresponding landmarks on the scan mesh model.

III.4.2.2 Landmark Correspondences

For each scan mesh model Y^{st}, we manually place a set of 38 markers on the surface of the mesh to establish vertex correspondences CM^{st} to the template. Figure III.5 shows an example of landmark correspondences.

III.4.2.3 Hand Mesh Representation

We use a skinned mesh model to represent 3D hands. Our template mesh $\hat{M}$ contains 4138 vertices and 8227 faces, and the template skeleton $\hat{K}$ contains 22 bone segments. We describe a hand pose using a set of independent joint coordinates $\mathbf{q} \in \mathbf{R}^{33}$, including absolute root position and orientation as well as the relative joint angles of individual joints. For Distal Interphalangeal (DIP) and Proximal Interphalangeal (PIP) joint, we use 1 degree of freedom (Dof) to describe their movement. We choose to model Metacarpophalangeal (MCP) joints using a ball and socket joint, thus each finger has 5 Dofs except the thumb finger. For the thumb finger, we use 1 Dof for Interphalangeal (IP) joint, and 3 Dofs for

MCP and Trapeziometacarpal (TM) joint.

III.4.2.4 Skinning Weights Initialization

We initialize the skinning weights $\mathbf{W}$ using Geodesic Voxel Binding [4] using the template mesh and skeleton. It is a fully automatic method that provides smooth skinning weights that are sparse and have strong locality, and the user could adjust the level of sparseness and locality.

III.4.3 Correspondence Estimation

This section will discuss how to estimate the correspondences $CI^{st} = \{ (m_i, y_i) \}$ between a template mesh M and a scan mesh Y^{st} for each scan in the database, where m_i is the vertex index of M and y_i is the corresponding index in Y.

Since we use scan meshes to estimate the subject-specific parameters directly, it is important to find good correspondences for optimization. Our idea is that, we first get an intermediate representation $\bar{Y}^{st}$ for each scan Y^{st} by fitting Y^{st} using the template mesh, or the mesh generated from the previous step if exists, and then extract the correspondences based on the fitted mesh. Note that the final correspondences will still between the template and the scan. We repeat this step iteratively to update the correspondences during the learning process.

The advantage is that, comparing to finding closest correspondences directly, this method results in better matched and consistent correspondences, which will be helpful for optimization in other steps to avoid local minimum. Compared to other algorithms that learn the model from aligned meshes, our method could correct mismatched correspondences caused by alignment error gradually, since we optimize against the scan directly, and the correspondences will be updated iteratively based on the previous estimated result.

To get the intermediate representation $\bar{Y}^{st}$, we apply deformation transfer [68] to fit the scan Y^{st} based on manually labeled markers using the template mesh or previous estimated

mesh $\tilde{H}(\mathbf{S}^s, \mathbf{O}^s, \mathbf{q}_t^s)$ defined in Equation (III.10) if exists, where $\mathbf{S}^s$, $\mathbf{O}^s$ and $\mathbf{q}_t^s$ are the subject-specific parameters estimated in the previous step. Using previous estimated mesh for fitting is better since it is already close to the scan and it also uses the shape information from other scans that could compensate for missing or noisy data.

After having the intermediate representation $\bar{Y}^{st}$, we find the closest correspondence between $\bar{\mathbf{Y}}_{m_i}^{st}$ and the scan $\mathbf{Y}_{y_i}^{st}$ for each vertex index m_i of $\bar{Y}$, and get the final correspondences between the template and the scan $CI^{st} = \{ (m_i, y_i) \}$.

To find the closest correspondence, we use an aggressive strategy that we find the correspondences in both directions and keep the one with longest distance from all closest points. That is, for each vertex on the scan, we find its closest vertex on the fitted template. And for vertices having identical closest vertex on the template, we keep the one with longest distance as the correspondences. The correspondences from template mesh to the scan are established in a similar way, and then the results from both directions are merged together to get the final correspondences. This strategy could fit the scan better since it prefers to reduce the maximum distances instead of the minimum ones.

Part of the arms are usually visible in the scan meshes, but our template mesh contains the hand only. This may cause a problem for the above strategy because it will deform the triangles under the palm severely to fit the arm part of the scan, which is undesired. To address this issue, for vertices below the palm on the template mesh, we find their correspondences only in the direction from template to the scans. These vertices will then stay to their closest vertices instead of being stretched to fit the arms.

III.4.4 Subject-specific Parameters Estimation

In this section, we propose an algorithm to estimate subject-specific hand parameters for each subject in the scan database, including high-dimensional shape parameters, i.e., skeleton scales $\mathbf{S}^s$ and vertex offsets $\mathbf{O}^s$, as well as poses $\{ \mathbf{q}_t^s \}$ for scans $Y^s = \{ Y^{st} \}$ of the

subject s, using our high-dimensional hand model Equation (III.10).

We formulate an optimization problem to minimize the reconstruction error while keeping the estimated shape smooth. We explicitly optimize for a subject-specific template mesh $\tilde{\mathbf{H}}(\mathbf{S},\mathbf{O},\mathbf{q}_{rest})$ under rest pose $\mathbf{q}_{rest}$, keeps it smooth and deforms it to fit multiple scans simultaneously.

Comparing to previous work that fits the template to each scan individually, our algorithm produces more accurate results for the following reasons. Firstly, we have a consistent shape representation across poses which explains scans better and have better vertex correspondences. Secondly, the influences of missing or noisy data are reduced since information from multiple scans are used together. For example, if a missing finger part could be seen in a different scan even if they are not the same pose, these data will still be used to constrain the missing part.

The problem could be formulated as follows:

$$\min_{\mathbf{S},\mathbf{O},\{\mathbf{q}_i\}} E_{Rest}(\mathbf{S},\mathbf{O}) + \sum_i E_i(\mathbf{S},\mathbf{O},\mathbf{q}_i), \tag{III.12}$$

where $\mathbf{S}$, $\mathbf{O}$ and $\{\mathbf{q}_i\}$ are the skeleton scales, vertex offsets and poses for the scans of a given subject. $E_{Rest}(\mathbf{S},\mathbf{O})$ is objective function for the subject-specific template mesh and skeleton in rest pose, and $E_i(\mathbf{S},\mathbf{O},\mathbf{q}_i)$ is the objective function for each scan mesh under pose i.

The term E_{Rest} is mainly used to regularize the shape of the subject in rest pose $\mathbf{q}_{rest}$, and is defined as:

$$E_{Rest}(\mathbf{S},\mathbf{O}) = w_S E_S(\mathbf{S},\mathbf{O},\mathbf{q}_{rest}) + w_{Jo} E_{Jo}(\mathbf{S},\mathbf{O}) + E_{Reg}(\mathbf{S},\mathbf{O}), \tag{III.13}$$

where E_S is a *smoothness* term used to ensure the smoothness of the subject-specific template mesh. E_{Jo} is a *skeleton joint regularization* term, which indicates that after applying

the shape parameters, the relative positions between the skeleton joints and their neighbor vertices should keep unchanged. This term also prevents the skeleton from moving outside of the template mesh. E_{Reg} is a *regularization* term used to avoid large changes of the shape parameters. w_S, w_{Jo} are weights for the terms.

The data term E_i for each scan in the subject is mainly used make sure the similarity of the estimated model to the scans, which could be defined as:

$$E_i(\mathbf{S},\mathbf{O},\mathbf{q}_i) = w_M E_M(\mathbf{S},\mathbf{O},\mathbf{q}_i) + w_{Data} E_{Data}(\mathbf{S},\mathbf{O},\mathbf{q}_i) + E_{PReg}(\mathbf{q}_i), \tag{III.14}$$

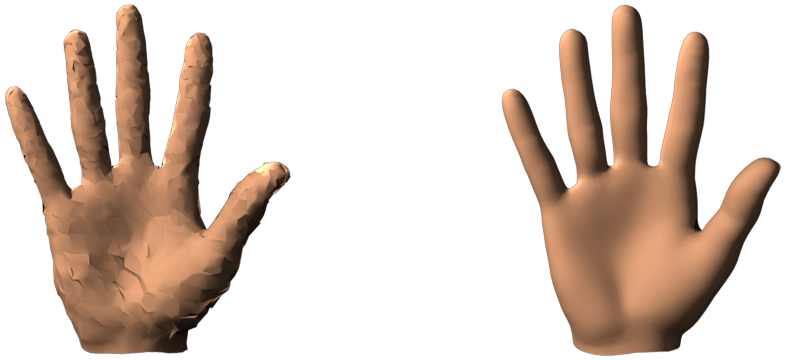
where E_M and E_{Data} are the *landmark* term and the *data* term, which ensures that the template mesh after deformation should fit all the scans. E_{PReg} is the *regularization* term for the pose $\mathbf{q}_i$, w_M and w_{Data} are the weights for the corresponding terms.

III.4.4.1 Smoothness Term

The mesh generated by Equation (III.10) may results in choppy and not smooth surfaces (see Figure III.6 (a)) due to the high-dimensional vertex offsets. To ensure the smoothness of the mesh $\tilde{\mathbf{H}}$ (see Figure III.6 (b)), we use a smoothness term similar to [68] as follows:

$$E_S(\mathbf{S},\mathbf{O},\mathbf{q}) = \sum_{i=1}^{|T|} \sum_{j \in \text{adj}(i)} \|\mathbf{Q}_i(\mathbf{S},\mathbf{O},\mathbf{q}) - \mathbf{Q}_j(\mathbf{S},\mathbf{O},\mathbf{q})\|_F^2, \tag{III.15}$$

where $\mathbf{Q}_i$ is the non-translational part of the affine transformation for the ith triangle on the template mesh $\hat{M}$ before and after deformation. The template mesh after deformation is computed by Equation (III.10) so that $\mathbf{Q}_i$ depends on the scales, vertex offsets and the pose. $\text{adj}(i)$ gives the neighboring triangle indices for triangle i. This term ensure the smoothness by enforcing neighboring triangles to have similar transformations in the non-



(a) (b)

Figure III.6: Importance of the smoothness term. (a) result without smoothness term; (b) result with smoothness term.

translational part.

Transformation matrix $\mathbf{Q}_i(\mathbf{S},\mathbf{O},\mathbf{q})$ could be computed using the method in [68] as follows. For the ith triangle $(\hat{\mathbf{v}}_{i_0},\hat{\mathbf{v}}_{i_1},\hat{\mathbf{v}}_{i_2})$ of the template mesh $\hat{M}$, we have a corresponding triangle $(\mathbf{v}_{i_0},\mathbf{v}_{i_1},\mathbf{v}_{i_2})$ after deformation, where $\mathbf{v}_i = \tilde{\mathbf{H}}_i(\mathbf{S},\mathbf{O},\mathbf{q})$ is the ith vertex position computed by Equation (III.10). We compute $\mathbf{Q}_i$ as:

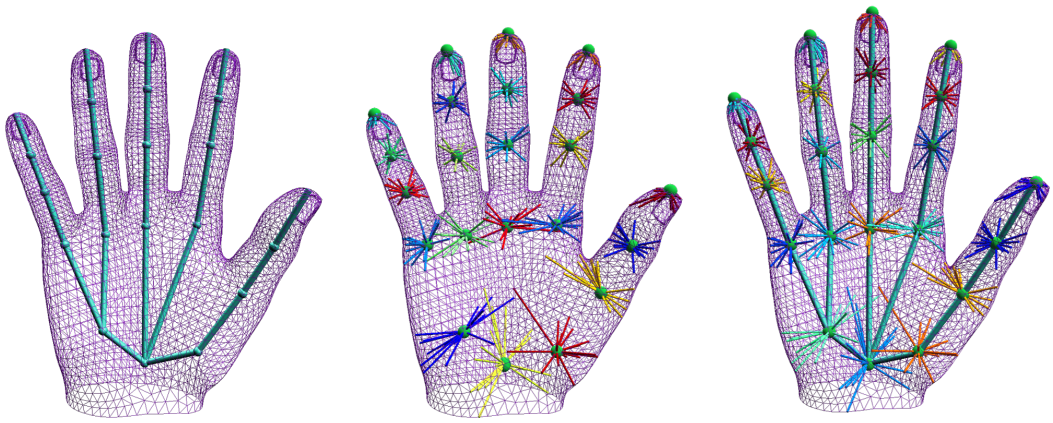
$$\mathbf{Q}_i(\mathbf{S},\mathbf{O},\mathbf{q}) = \mathbf{V}_i(\mathbf{S},\mathbf{O},\mathbf{q})\hat{\mathbf{V}}_i^{-1}, \tag{III.16}$$

where $\hat{\mathbf{V}}_i$ and $\mathbf{V}_i(\mathbf{S},\mathbf{O},\mathbf{q})$ are defined as:

$$\begin{aligned}\hat{\mathbf{V}}_i &= [\hat{\mathbf{v}}_{i_1}-\hat{\mathbf{v}}_{i_0} \quad \hat{\mathbf{v}}_{i_2}-\hat{\mathbf{v}}_{i_0} \quad \mathbf{V}_3(\hat{\mathbf{v}}_{i_0},\hat{\mathbf{v}}_{i_1},\hat{\mathbf{v}}_{i_2})-\hat{\mathbf{v}}_{i_0}],\\ \mathbf{V}_i(\mathbf{S},\mathbf{O},\mathbf{q}) &= [\mathbf{v}_{i_1}-\mathbf{v}_{i_0} \quad \mathbf{v}_{i_2}-\mathbf{v}_{i_0} \quad \mathbf{V}_3(\mathbf{v}_{i_0},\mathbf{v}_{i_1},\mathbf{v}_{i_2})-\mathbf{v}_{i_0}],\end{aligned} \tag{III.17}$$

and $V_3(\mathbf{v}_0,\mathbf{v}_1,\mathbf{v}_2)$ is the vertex perpendicular to the triangle and could be defined as:

$$\mathbf{V}_3(\mathbf{v}_0,\mathbf{v}_1,\mathbf{v}_2) = \mathbf{v}_0 + (\mathbf{v}_1-\mathbf{v}_0)\times(\mathbf{v}_2-\mathbf{v}_0)/\sqrt{|(\mathbf{v}_1-\mathbf{v}_0)\times(\mathbf{v}_2-\mathbf{v}_0)|}. \tag{III.18}$$



(a) (b) (c)

Figure III.7: Skeleton joint term. The skeleton joint term maintains the relative positions between the skeleton and its neighboring vertices. (a) template mesh and skeleton; (b) skeleton joints are represented as linear combinations of neighboring vertices; (c) deformed mesh and skeleton. Skeleton joints (green dots) are estimated using linear combination weights from (b).

III.4.4.2 Skeleton Joint Term

The relative position between the skeleton and the mesh is important since linear blend skinning relies on it for pose deformation, hence we would like to maintain this relationship after shape deformation. To do this, we first build the relationship by representing the skeleton joints as linear combinations of their neighboring vertices, and maintain this relationship during optimization.

To find the relative positions between template mesh and skeleton (see Figure III.7 (a)(b)), we represent each skeleton joint $\hat{\mathbf{J}}_i$ as a linear combination of neighboring vertices on the mesh as:

$$\hat{\mathbf{J}}_i = \sum_{j \in \text{nei}(i)} a_j \hat{\mathbf{v}}_j, \tag{III.19}$$

where $\text{nei}(i)$ is the set of neighboring vertex indices for joint i, $\hat{\mathbf{v}}_j$ is the jth vertex on the

template mesh, $\mathbf{A}_i = [a_{ij}]$ is the coefficients for neighboring vertices. The coefficients $\mathbf{A}_i$ could be estimated in closed form using linear least squares with an additional coefficient regularization term.

Based on the estimated coefficients $\{\mathbf{A}_i\}$, we could compute the joint position $\mathbf{J}_i$ for the deformed mesh in rest pose $\mathbf{q}_{rest}$ (see Figure III.7 (c)) as:

$$\mathbf{J}_i(\mathbf{S},\mathbf{O}) = \sum_{j \in \text{nei}(i)} a_j \tilde{\mathbf{H}}_j(\mathbf{S},\mathbf{O},\mathbf{q}_{rest}). \tag{III.20}$$

Based on this representation, we could use the following to maintain this relationship:

$$E_{Jo}(\mathbf{S},\mathbf{O}) = \sum_j \|\mathbf{J}_j(\mathbf{S},\mathbf{O}) - \mathbf{K}_j(\mathbf{S},\mathbf{q}_{rest})\|^2, \tag{III.21}$$

where $K_i(\mathbf{S},\mathbf{q}) = \mathbf{T}_i(\mathbf{S},\mathbf{q})\hat{\mathbf{T}}_i^{-1}\hat{\mathbf{J}}_i$ is the ith joint position computed from the template skeleton, $\mathbf{T}_i(\mathbf{S},\mathbf{q})$ and $\hat{\mathbf{T}}_i$ are the transformation matrices defined in Equation (III.10), and $\hat{\mathbf{J}}_i$ is the ith joint position of the template skeleton.

This term is important for the subject-specific parameters estimation. Figure III.8 shows a comparison with and without this term. It is clear to see that this term not only reduces the ambiguity of the estimated result (Figure III.8 (c)), but also produces more natural-looking meshes (Figure III.8 (a) (b)).

III.4.4.3 Data Term

The data term is used to make sure that the template mesh after deformation will fit the scan, which is defined as:

$$E_{Data}(\mathbf{S}^s,\mathbf{O}^s,\mathbf{q}_t^s) = \sum_{(j,k)\in CI^{st}} \|\tilde{\mathbf{H}}_j(\mathbf{S}^s,\mathbf{O}^s,\mathbf{q}_t^s) - \mathbf{Y}_k^{st}\|^2, \tag{III.22}$$

where $\tilde{\mathbf{H}}_j(\cdot)$ is the jth vertex after deformation defined in Equation (III.10), $\mathbf{Y}_j^{st}$ is the jth

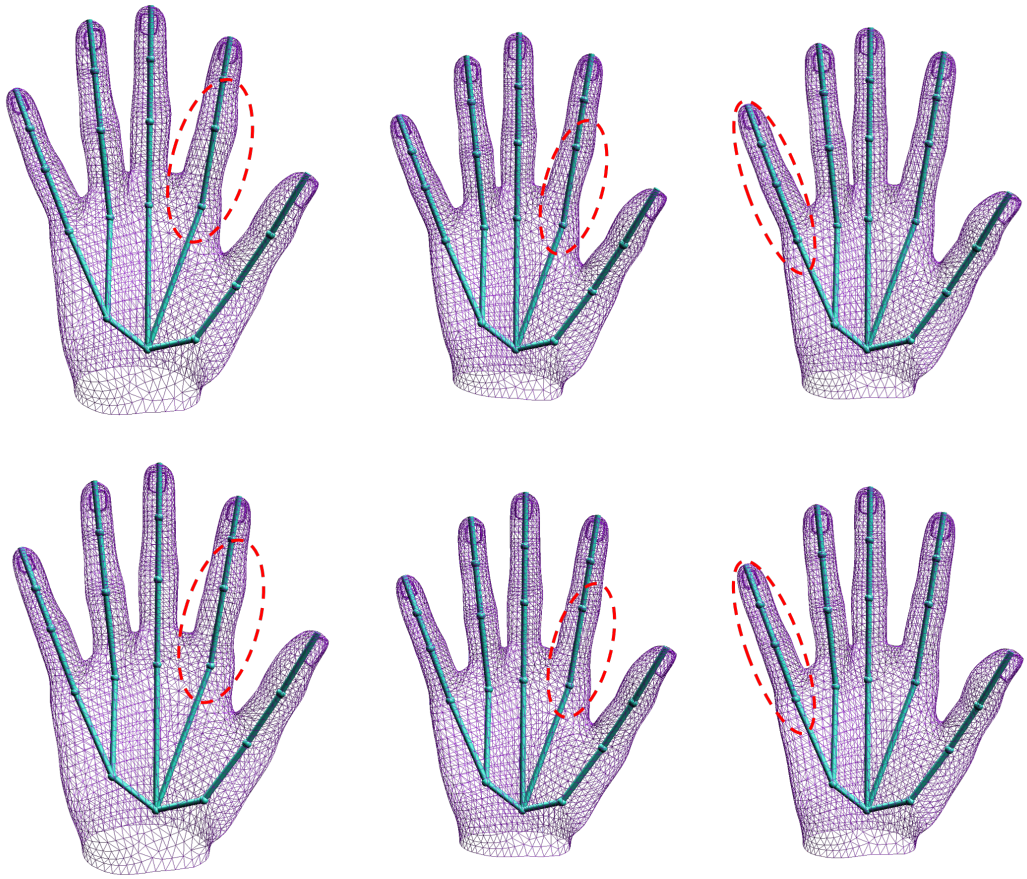


Figure III.8: Importance of the skeleton joint term. (top) estimation result without the skeleton joint term. Note the artifacts caused by inaccurate joint positions shown in the circles; (bottom) estimation result with the skeleton joint term.

vertex of the tth scan for subject s. CI^{st} is the correspondences estimated in previous step and each pair (j,k) in CI^{st} gives the corresponding vertex indices between the jth template vertex and kth vertex on the scan mesh.

III.4.4.4 Landmark Term

For each scan Y^{st}, a set of landmark correspondences CM^{st} to the template (see Figure III.5) are labeled. This term measures the distances for corresponding vertices, which provides strong guidance for the optimization to avoid local minimum, especially when the template and the scan are not close enough. The term is defined as:

$$E_M(\mathbf{S}^s, \mathbf{O}^s, \mathbf{q}_t^s) = \sum_{(j,k)\in CM^{st}} \|\tilde{\mathbf{H}}_j(\mathbf{S}^s, \mathbf{O}^s, \mathbf{q}_t^s) - \mathbf{Y}_k^{st}\|^2. \tag{III.23}$$

III.4.4.5 Regularization Terms

We use the following term to regularize the shape parameters to avoid large changes:

$$E_{Reg}(\mathbf{S}, \mathbf{O}) = w_{ro} \sum_i \|\mathbf{O}_i\|^2 + w_{rs} \sum_i \|\mathbf{S}_i - 1\|^2, \tag{III.24}$$

where w_{ro} and w_{rs} are the weights. This term indicates that the shape parameters should stay close to their default values.

We also define the regularization term for pose $\mathbf{q}$ as:

$$E_{PReg}(\mathbf{q}_i) = w_{rq} \|\mathrm{JA}(\mathbf{q}_i) - \mathrm{JA}(\hat{\mathbf{q}}_i)\|^2, \tag{III.25}$$

where $\hat{\mathbf{q}}_i$ is the predefined pose for the ith scan, $\mathrm{JA}(\mathbf{q})$ gives the joint angles of pose $\mathbf{q}$ by discarding the global positions.

III.4.4.6 Optimization

This is a nonlinear minimization problem, containing a relative large number of variables, including skeleton scales $\mathbf{S}$, vertex offsets $\mathbf{O}$ and poses $\{\mathbf{q}_i\}$. To find an optimal solution, we optimize the variables in multiple stages using Levenberg-Marquardt (LM) algorithm.

We first initialize the skeleton scales and vertex offsets to 1s and 0s respectively. We then optimize for the poses and the skeleton scales jointly using Equation (III.12) without using the data term defined in Equation (III.22). This results in a reasonable initialization of scales $\mathbf{S}$ and poses $\{\mathbf{q}_i\}$.

During each stage of optimization, we iteratively optimize the vertex offsets, scales and poses one by one based on results from previous stage using correspondences *CI* and *CM*.

Figure III.9 and Figure III.10 show some estimation results from the scan database, including two males, a female and a child subjects. We could see that the estimated meshes fit the scans well.

III.4.5 Skinning Weights Learning

In this section, we propose an algorithm to learn the skinning weights $\mathbf{W}$ from the scan database using the previously estimated subject-specific hand parameters.

Our parametric model relies on skinning weights $\mathbf{W}$ for both shape and pose deformation. We initialize the skinning weights only based on the template mesh and skeleton, which may result in undesired deformation for different shape and pose.

To address this issue, we formulate an optimization problem to minimize the reconstruction error caused by inaccurate skinning weights. We use all meshes from different subjects under different poses altogether for optimization. Our result fits all shape and poses better because all information from scans is used.

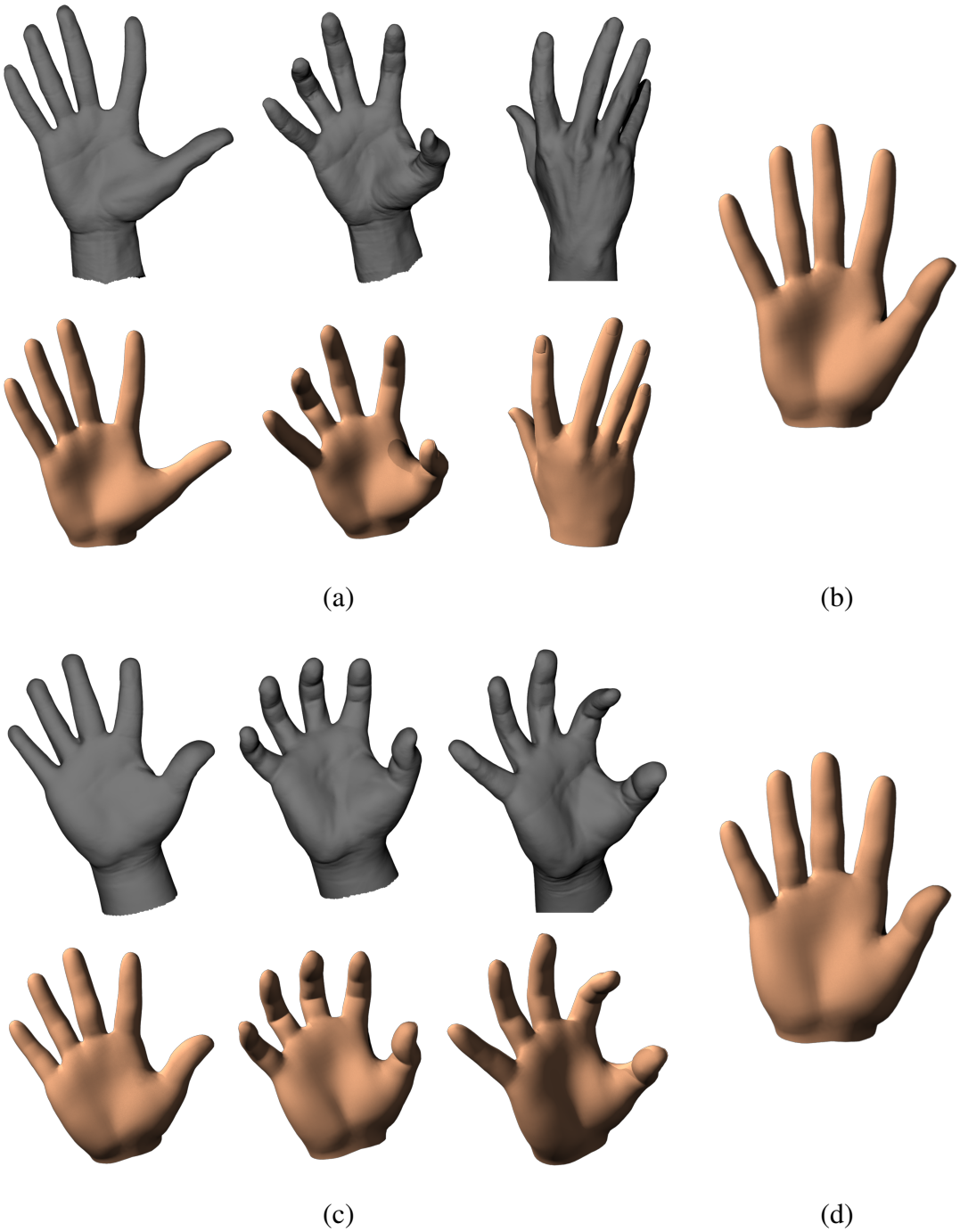


Figure III.9: Subject-specific hand parameters estimation results for two male subjects. (a) (c) scan meshes (gray) and estimated meshes (orange); (b) (d) estimated subject-specific template mesh.

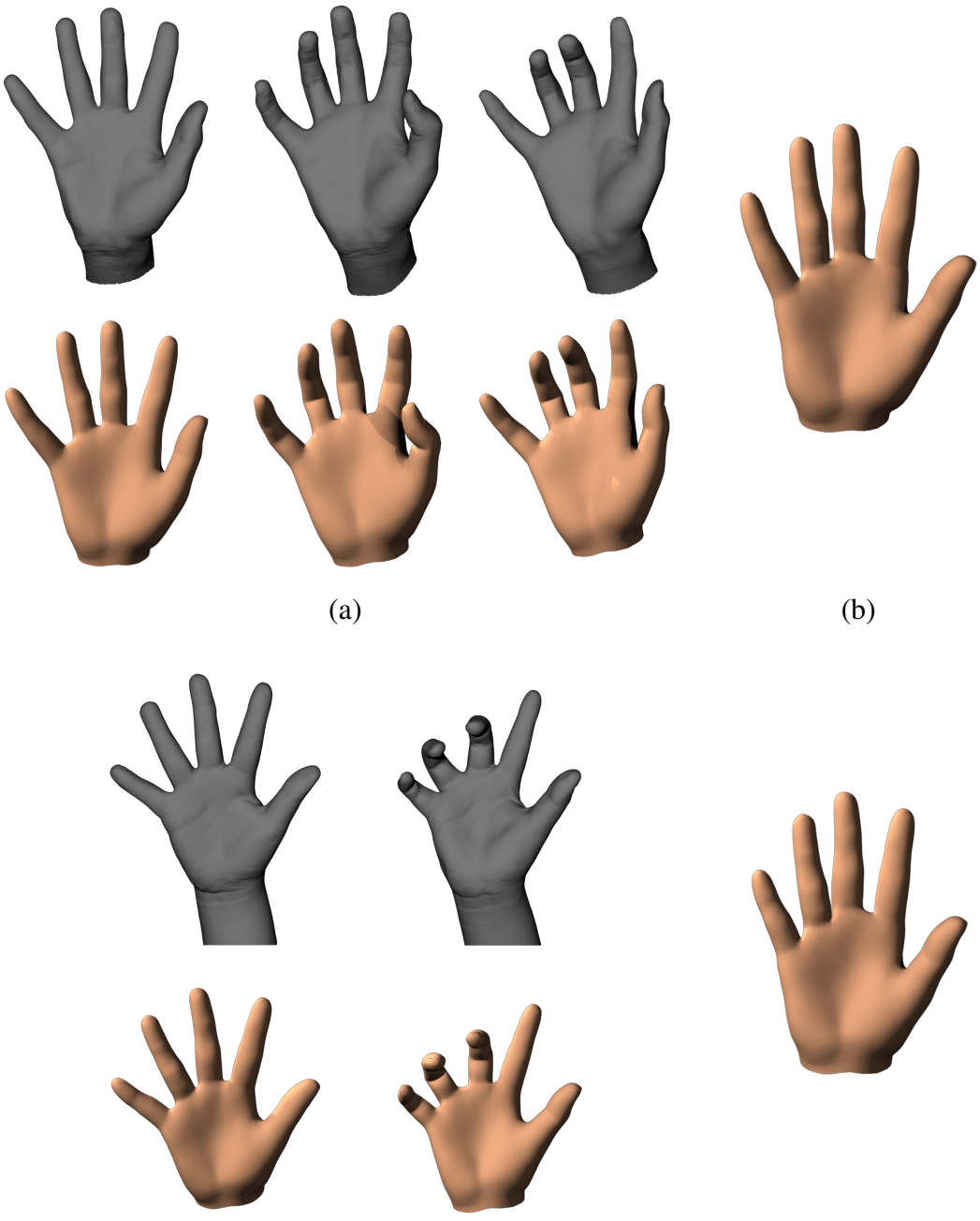


Figure III.10: Subject-specific hand parameters estimation results for a female and child subjects. (a)(c) scan meshes (gray) and estimated meshes (orange); (b)(d) estimated subject-specific template mesh.

For each subject s, we have corresponding subject-specific hand parameters $SP_s = \{\mathbf{S}^s, \mathbf{O}^s, \{\mathbf{q}_i^s\}\}$. We formulate to learn the skinning weights $\mathbf{W} = [w_{ij}]$ as follows:

$$\min_{\mathbf{W}} \sum_s \sum_t \sum_i \|\mathbf{Y}_{C_i^{st}}^{st} - \tilde{\mathbf{H}}_i(\mathbf{S}^s, \mathbf{O}^s, \mathbf{q}_t^s; \mathbf{W})\|^2 \tag{III.26}$$

$$\begin{aligned} \text{subject to:} \quad & w_{ij} \geqslant 0, \forall i, j \\ & \sum_j w_{ij} = 1, \forall i, \end{aligned} \tag{III.27}$$

where $\tilde{\mathbf{H}}_i(\mathbf{S}^s, \mathbf{O}^s, \mathbf{q}_t^s; \mathbf{W})$ is the ith deformed vertex for the tth scan of the subject s defined in Equation (III.10), $\mathbf{Y}_i^{st}$ is the ith vertex of the corresponding scan mesh, and C_i^{st} is the corresponding vertex index on the scan for deformed vertex i. Equation (III.26) evaluates the reconstruction errors for scans from all meshes and Equation (III.27) enforces the non-negative and affinity constraints on the skinning weights.

III.4.5.1 Optimization

Optimize for Equation (III.26) directly may not work well. Firstly, it requires to optimize for all meshes together, which would become infeasible when the number of scans is large. In addition, the estimated skinning weights may result in non-smooth meshes after deformation because the scan meshes are noisy and may have missing data. Adding smoothness terms similar to Equation (III.4.4.1) will be helpful, but it will make the computational cost higher and even infeasible.

To address these issues, we find the optimal skinning weights for each vertex individually since they are independent of each other. This could largely reduce the computational cost. Additionally, in order to get smooth results, instead of finding corresponding C on the scan Y directly, we estimate smooth per-vertex correspondences from an intermediate representation of Y.

That is, for each subject s, we have a subject-specific model $\tilde{\mathbf{H}}(\mathbf{S}^s, \mathbf{O}^s, \mathbf{q}_t^s; \mathbf{W})$ for pose t computed from Equation (III.10). We first find the intermediate representation $\bar{Y}^{st}$ for scan Y^{st} by applying deformation transfer [68] to $\tilde{\mathbf{H}}(\cdot)$ to fit Y. This results in a smooth mesh $\bar{Y}^{st}$ that fits the scan Y^{st} better, compensating for the fitting errors in $\tilde{\mathbf{H}}(\cdot)$ caused by problematic skinning weights and deformation limitation of LBS. Then we could use corresponding vertices between $\tilde{\mathbf{H}}(\cdot)$ and $\bar{Y}^{st}$ as correspondences, and optimize the skinning weights for each vertex $\mathbf{W}_i$ on all meshes as follows:

$$\begin{aligned} \min_{\mathbf{W}_i} & \sum_s \sum_t \|\bar{\mathbf{Y}}_i^{st} - \tilde{\mathbf{H}}(\cdot; \mathbf{W})\|^2 \\ \text{subject to:} \quad & w_{ij} \geq, \forall j \\ & \sum_j w_{ij} = 1, \end{aligned} \tag{III.28}$$

where $\mathbf{W}_i$ is the ith row of the skinning weight for vertex i, $\tilde{\mathbf{H}}_i(\cdot; \mathbf{W})$ is the ith vertex of the subject-specific model, which depends only on the ith row of the skinning weights, and $\bar{\mathbf{Y}}_i^{st}$ is the ith vertex of the intermediate representation.

Note that $\tilde{\mathbf{H}}_i(\cdot; \mathbf{W})$ could also be written as follows based on Equation (III.10):

$$\tilde{\mathbf{H}}_i(\cdot; \mathbf{W}) = \sum_{j=0}^{n-1} w_{ij} \tilde{\mathbf{V}}_j^{st}, \tag{III.29}$$

where $\tilde{\mathbf{V}}_j^{st} = \mathbf{T}_j(\mathbf{S}^s, \mathbf{q}_i^s)\tilde{\mathbf{v}}'_{ij}(\mathbf{S}^s, \mathbf{O}^s)$ is the vertex in world coordinate that is independent of the skinning weights, hence Equation (III.28) is a constrained linear least squares problem that could be solved efficiently. During optimization for each vertex, we only optimize for the bones with non-zero weights while keeping others zero. This is important for good generalization since it preserves the sparseness and locality of the skinning weights.

III.4.6 Parametric Model Extraction

After having subject-specific hand parameters $SP^s = \{ \mathbf{S}^s, \mathbf{O}^s, \{ \mathbf{q}_i^s \} \}$ estimated in Section III.4.4, we learn low-dimensional shape models (Equation (III.4) and Equation (III.6)) by applying Principal Component Analysis (PCA) on the skeleton scales $\{ \mathbf{S}^s \}$ and vertex offsets $\{ \mathbf{O}^s \}$ respectively. By keeping a small number of principal components, we get the shape bases $U = \{ \mathbf{S}_0, \mathbf{C}_\mathbf{S}, \mathbf{O}_0, \mathbf{C}_\mathbf{O} \}$ defined in Equation (III.9).

Combining with the skinning weights $\mathbf{W}$ estimated in Section III.4.5, we have the low-dimensional parametric hand $\mathbf{H}(\alpha, \beta, \mathbf{q}; \mathbf{W}, U)$ defined in Equation (III.9).

III.5 Applications

With our low-dimensional parametric hand model, we build a variety of applications for modeling and synthesis, ranging from 3D hand modeling based on various forms of input constraints, including depth images obtained by depth sensors, incomplete scan data, a small set of color images, and semantic constraints defined by the user, to transferring skinning weights to a pre-existing hand model and model-based hand tracking using a single depth camera.

III.5.1 User-specific Hand Modeling

Based on our compact parametric hand model $H(\cdot)$, we could synthesize a user-specific skinned mesh model for the subject from different sources, including depth images, incomplete scan data, color images, and semantic constraints.

We formulate the user-specific hand modeling as an optimization problem to optimize for the low-dimensional parameters $\{ \alpha, \beta, \mathbf{q} \}$ to fit the observation O using the analysis-by-synthesis strategy as follows:

$$\min_{\alpha, \beta, \mathbf{q}} E(H(\alpha, \beta, \mathbf{q}), O), \tag{III.30}$$

where $E(H(\cdot), O)$ measures the discrepancy between the synthesized model $H(\alpha, \beta, \mathbf{q})$ and the observed data O.

III.5.1.1 Automatic Modeling from a Depth Sensor

Many applications, such as model-based hand tracking, require an accurate hand model for better performance, which is not easy to acquire for a novel user. Our low-dimensional parametric model provides an easy way for a novel user to build an accurate hand model with skinning.

We ask the user to perform several predefined poses in front of a depth camera for a few seconds. To reconstruct both the shape and poses of the subject, we formulate an optimization problem and seek to find the optimal shape parameter α, β and poses $\{\mathbf{q}_t\}$ for each depth image $\mathbf{D}_t$:

$$\min_{\alpha,\beta,\mathbf{q}_t} \sum_{i \in C} \|H_i(\alpha, \beta, \mathbf{q}_t) - \mathbf{p}_i^*\|^2, \tag{III.31}$$

where C is the set of correspondences, and $\mathbf{p}_i^*$ is the corresponding point for H_i on the point cloud generated from D_t.

To find the corresponding point $\mathbf{p}_i^*$ for each hypothesized vertex H_i, we first render a depth image R_t of the hypothesized model $H(\alpha, \beta, \mathbf{q}_t)$ using camera parameters identical to the input. Then we find the correspondence using a bidirectional strategy to minimize the non-overlapping region between the hypothesized R_t and observed depth image D_t. That is, for non-overlapping pixels in D_t, we find their correspondences in R_t, and then for pixels in R_t who have multiple closest points, we keep the one with the longest distance as the correspondence. We find the correspondence from R_t to D_t and then merge the correspondences in the same way. Note that forearm is not modeled in our model, but it is visible in D_t. This strategy will force our model to expand in undesired directions. To solve this problem, we keep correspondences only from one direction for hypothesized vertices

near the wrist. That is, for correspondences from D_t to R_t, we discard those correspond to the vertices near the wrist. Vertices near the wrist could be easily identified by manually labeled on the template mesh.

We optimize the shape and poses as follows. We first optimize the pose and shape of the first frame, which gives a rough shape for the subject. We then track the pose for each frame sequentially while keeping the shape parameters fixed. After that we refine the shape in a batch manner using frames whose poses are close to our predefined ones.

For the first frame, we optimize the shape and pose iteratively. We first find the pose of the subject using the default shape parameters, then we optimize for the shape and pose iteratively by keeping one of them fixed.

When optimizing the shape in batch, we use the following objective function:

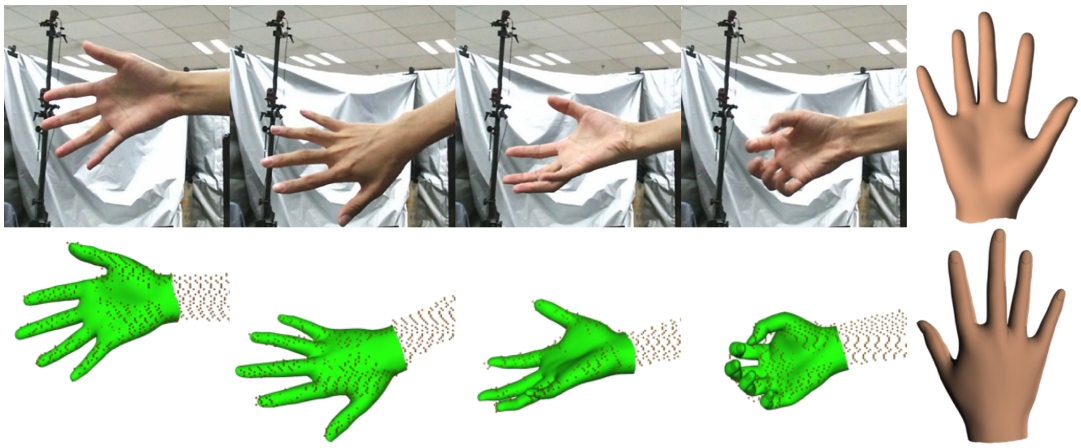
$$\min_{\alpha,\beta,\{\mathbf{q}_t\}} \sum_{t\in F}\sum_{i\in C} \|H_i(\alpha,\beta,\mathbf{q}_t) - \mathbf{p}^*_{ti}\|^2, \tag{III.32}$$

where F is the selected frame indices and $\mathbf{p}_{ti}i^*$ is the corresponding vertex for the ith vertex on the synthesized mesh for the tth frame. We repeat this process for a few iterations and update the poses in F after the batch shape optimization.

Figure III.11 shows some sample images of our modeling result from a depth sensor. Figure III.11 (a) shows the reference images and the reconstructed model with depth data, and Figure III.11 (b) shows the reconstructed hand model in rest pose. We could see that the reconstructed hand shape and poses fitted the depth data well.

III.5.1.2 Reconstruction from Incomplete/Noisy Scan

One way to build a subject-specific hand model is by using a 3D scanner. However, it has two limitations. Firstly, the scan may be incomplete or noisy. The scan from a laser scanner may contain only one side of the hand, or the scan may have holes due to occlusion. Secondly, skeleton or skinning weights are not available for the scan mesh,



(a) (b)

Figure III.11: Automatic hand modeling from a depth sensor. (a) reference images from Kinect sensor (top) and reconstructed model with depth data (bottom); (b) reconstructed model in default pose.

which limits its usage.

Based on our parametric hand model, we could reconstruct a complete mesh model that fits the partial scan best, with skeleton and skinning weights embedded for pose deformation.

For a given partial mesh O, we define the cost function E as follows:

$$E(H(\alpha,\beta,\mathbf{q}),O) = \sum_{(i,j)\in M} \left\| H_i(\alpha,\beta,\mathbf{q}) - \mathbf{v}_j^O \right\|^2 + \lambda \sum_j \left\| H_j(\alpha,\beta,\mathbf{q}) - \mathbf{v}_{j^*}^O \right\|^2 , \tag{III.33}$$

where the first part is the marker term, and the second part is the data term. M is the set of marker correspondences, which are labeled manually. $\mathbf{v}_j^O$ is the jth vertex coordinate on the observed mesh. j^* is the corresponding vertex index on the observed mesh for jth

vertex on synthesized mesh, which could be estimated by finding the closest points as in Section III.4.4.3.

To solve this problem, we employ an iterative approach using Levenberg-Marquardt optimization. We initialize the α and β to 0. We first solve the pose $\mathbf{q}$, and then the scales α using only the markers while keeping others fixed. After that we iteratively solve for β, $\mathbf{q}$ and α one by one for several iterations.

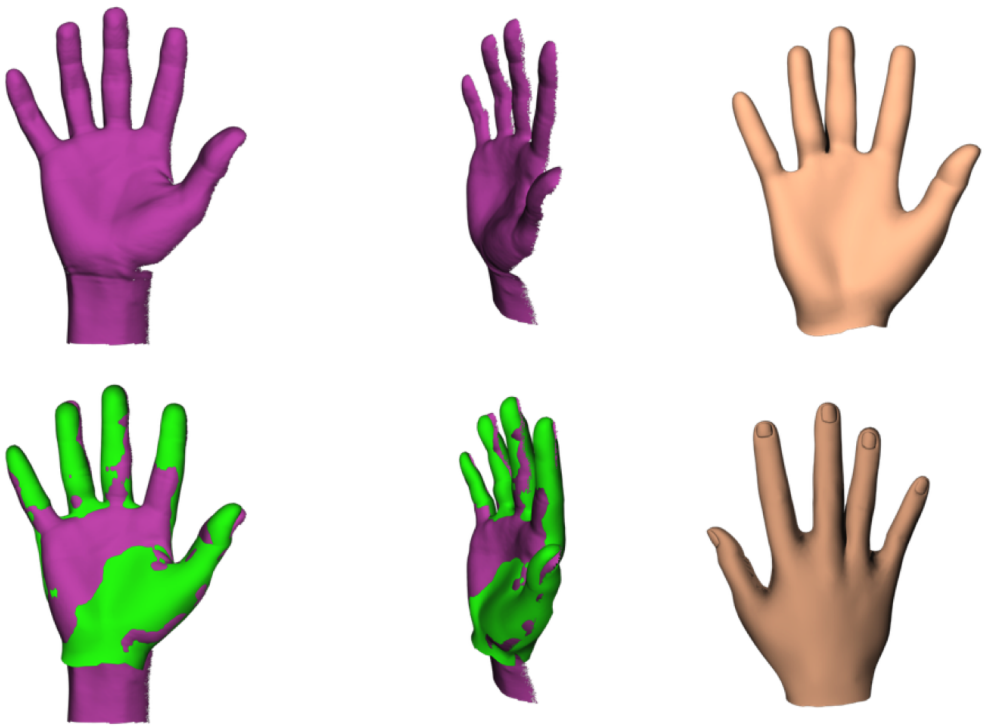
Figure III.12 and Figure III.13 show some sample images of our reconstruction results from incomplete scans. Figure III.12 (a) and Figure III.13 (a) show the incomplete scans (purple) and the reconstructed mesh models (green). Figure III.12 (b) and Figure III.13 (b) show the reconstructed meshes in rest pose. We could see that the reconstructed results not only fit the incomplete scan well, but also provide complete and smooth models.

III.5.1.3 Reconstruction from Color Images

In this section, we propose a method to reconstruct a user-specific hand model from color images, which capture the subject's hand in several predefined poses. We formulate the problem as in Equation (III.30) and adopt an analysis-by-synthesis strategy for optimization. That is, we use silhouette images rendered from our parametric model for cost evaluation. In addition, since no depth information is available, we use measurement information such as finger lengths to determine the absolute scale of the reconstructed model.

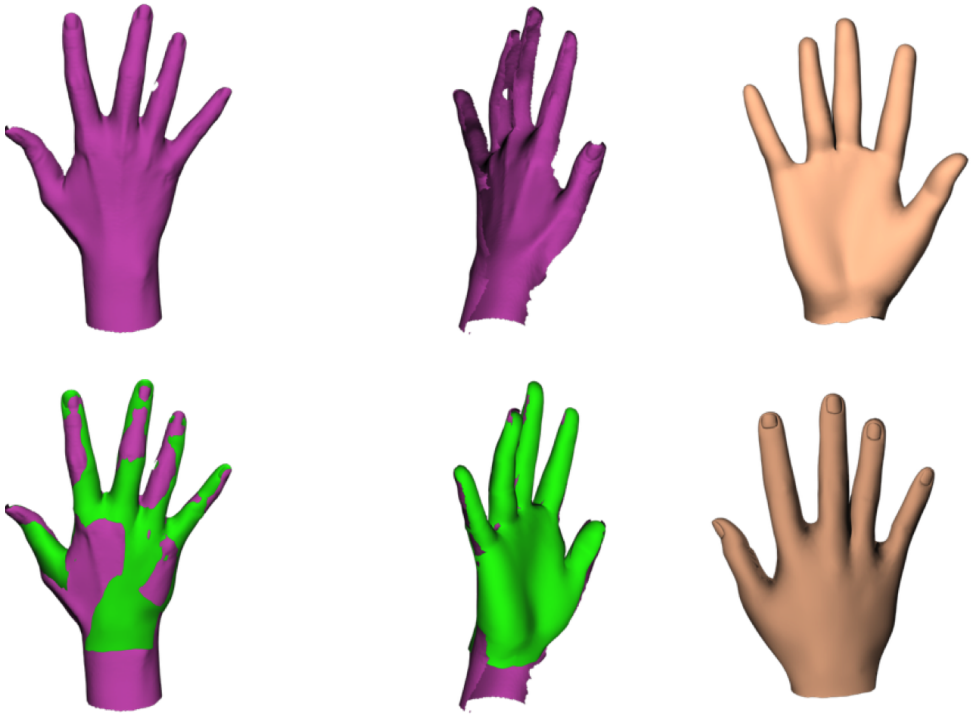
Suppose we have N color images capturing the subject's hand in several predefined poses using a camera with focal length f, we extract the hand silhouettes $\{o_i\}$ from the color images and use them as input. For a given hand model $H(\alpha,\beta,\mathbf{q})$, we could render a synthesized silhouette image using the focal length f as:

$$r(\alpha,\beta,\mathbf{q}) = \{\frac{f}{Z_i}[X_i,Y_i]^T\}, \tag{III.34}$$



(a) (b)

Figure III.12: Reconstruction from incomplete scan data for a male subject. (a) the incomplete scan (purple) and reconstructed model (green) in front and side views; (b) reconstructed model in rest pose.



(a) (b)

Figure III.13: Reconstruction from incomplete scan data for a female subject. (a) the incomplete scan (purple) and reconstructed model (green) in front and side views; (b) reconstructed model in rest pose.

where X_i, Y_i, and Z_i are the ith vertex coordinates synthesized from model $H(\alpha,\beta,\mathbf{q})$, and $r(\cdot)$ represents the set of 2D coordinates for silhouette pixels.

Then we could use the following objective function to estimate the hand parameters α, β, and $\mathbf{q}$:

$$\underset{\alpha,\beta,\{\mathbf{q}_i\}}{\arg\min} \sum_{i=1}^{N} \left(1 - \frac{2\sum(o_i \wedge r_i)}{\sum(o_i \wedge r_i) + \sum(o_i \vee r_i)}\right) + \lambda \sum_{j=1}^{M} \left\| l_j - \tilde{l}_j \right\|, \tag{III.35}$$

where o_i is the ith observed silhouette image extracted from ith color image, r_i is the ith silhouette image rendered from synthesized model $H(\alpha,\beta,\mathbf{q})$, and l_i is the length of the ith finger.

The first term measures the differences between the observed and rendered silhouette images, which tries to maximize the overlapping region between them. To reduce the ambiguity for reconstruction due to lacking of depth information, we use the second term to reduce the differences between the measured finger lengths $\tilde{l}_j$ and the finger lengths l_j computed on the synthesized model. The lengths of the fingers could be measured easily by the user, and we will discuss in more details how to measure the length of the model in Section III.5.1.4.

To solve Equation (III.35), we update $\mathbf{q}_i$, α and β iteratively using Particle Swarm Optimization (PSO) [69] algorithm. Note that all images are used when updating α and β, while individual frame is used to update the pose $\mathbf{q}_i$. We initialize the α and β as 0s, and set $\mathbf{q}_i$ to the pre-defined poses.

Figure III.14 shows some sample images of our synthesis result reconstructed from color images. Figure III.14 (a) shows the reconstructed results (green) on top of the input images. Figure III.14 (b) shows the reconstructed hand model in rest pose. We could see that the reconstructed shape and poses fit the input images well.

(a) (b)

Figure III.14: Reconstruction from color images. (a) reconstructed model (green) on top of input color images; (b) reconstructed model in rest pose.

III.5.1.4 Synthesis Using Semantic Constraints

A simple way to build a user-specific hand model is by specifying semantic measurement constraints, such as length or girth of fingers, or width of the palm. We use these constraints as observation O in Equation (III.30) and find the best shape parameters α and β that satisfies these constraints.

We define semantic constraints using vertex indices on the template mesh, which could also be used for our parametric hand model after shape deformation since they share the same topology and the vertex indices have identical semantic meanings.

The length constraint is defined by the geodesic distance between two vertices, which could be computed as a linear combination of vertices along the path on the template mesh. The vertex indices and the combination weights are then be applied to the deformed mesh created from our parametric model to compute the measurement. The girth measurement is defined as a set of geodesic paths connected one by one.

Based on the measurements, we use the following objective function for optimization:

$$E(\alpha,\beta,O) = \sum_i \|m_i(H(\alpha,\beta,\mathbf{q}_{rest})) - \tilde{m}_i\|^2 + \lambda_1 \sum_i \|\alpha_i - 1.0\|^2 + \lambda_2 \sum_i \|\beta_i\|^2, \tag{III.36}$$

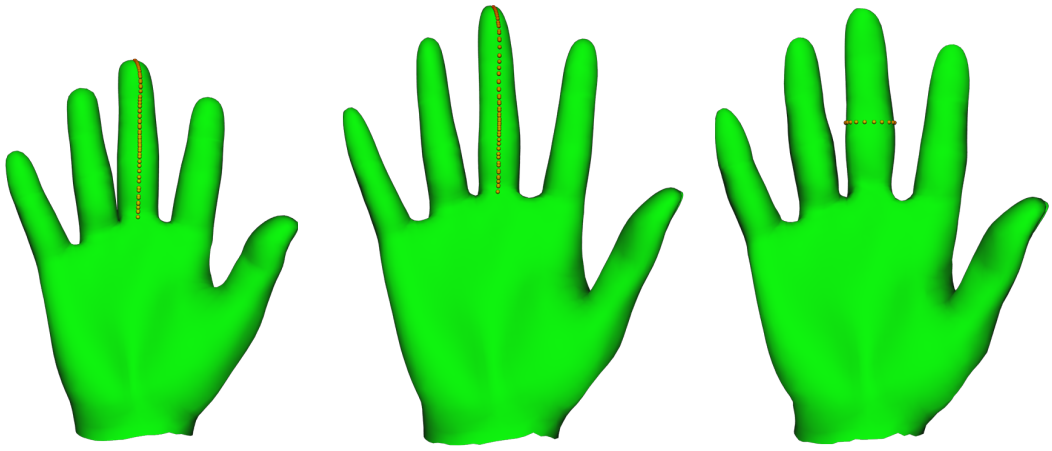
where $m_i(\cdot)$ is the ith measurement result for the model estimated from α, β and rest pose $\mathbf{q}_{rest}$, $\tilde{m}_i$ is the ith input constraint. The first term indicates that the estimated measurements should be close to the input constraints and the last two terms are the regularization terms to reduce reconstruction ambiguity. Since we are optimizing in the low dimensional shape space, the parametric model ensures the natural appearance of the result model, which is useful for interactive hand modeling.

Figure III.15 shows an example of synthesizing a human hand model using semantic constraints. Figure III.15 (a) is the template mesh with middle finger length of 76.6*mm*, (b) shows the synthesis result with middle finger length constraint of 90*mm*, and (c) shows the result with two constraints, middle finger length of 90*mm*, and middle finger girth of 70*mm*. The dots on the meshes show the geodesic paths used to compute the measurements. The synthesized results fit the constraints accurately and the models are still natural-looking.

III.5.2 Skinning Weights Transfer

LBS provides a simple way to make a static hand mesh model deformable, but it requires a time-consuming skinning weights painting process to build connections between mesh vertices and skeleton bones. One application of our parametric model is that we could easily transfer the skeleton and skinning weights from our model to the target static mesh, producing a deformable skinned mesh model.

For a pre-existing static model Y, we first fit our parametric model to Y using methods



(a) (b) (c)

Figure III.15: Reconstruction results using semantic measurements. (a) template mesh; (b) reconstructed model with middle finger length constraint; (c) reconstructed model with middle finger length and girth constraints.

in Section III.5.1, resulting in a subject-specific mesh model M with skeleton K and skinning weights W that matches Y well. We then transfer the skinning weights as follows. We first find the correspondences between Y and M, which could be done by finding the closest neighbor in M for each vertex Y_i on the static mesh. Then we assign the skinning weights for Y using the weights on M based on the correspondences. The estimated skeleton K could be used directly as the skeleton of Y. After these steps, we deform the pre-existing model using K as well as the transferred skinning weights using LBS.

Figure III.16 shows an example of skinning weights transfer result. Figure III.16 (a) is a pre-existing static human mesh model. We then transfer the skeleton and skinning weights to it to get a skinned mesh model, as shown in Figure III.16 (b). Skinning weights are coded by colors. Figure III.16 (c) shows a new pose of the pre-existing model, using the transferred skeleton and skinning weights.

(a) (b) (c)

Figure III.16: Skinning weights transfer result. (a) part of the pre-existing static model; (b) pre-existing model with skeleton and skinning weights transferred, color coded by skinning weights; (c) pre-existing model in a different pose.

III.5.3 Model-based Hand Tracking

Many model-based hand tracking systems (e.g., [41, 42, 43, 44]) use skinned mesh models to represent subject hands. Tracking accuracy may be affected if the model does not fit the subject well. Our parametric model provides an easy way to rapidly build user-specific hand models for tracking.

Based on the subject-specific hand model built using the methods in Section III.5.1, we formulate the hand tracking from a depth sensor as a frame by frame optimization problem. For a user-specific hand model M with skeleton K, we adopt an analysis-by-synthesis method and use the following objective function for tracking:

$$\min_{\mathbf{q}_t} \sum_{i \in C} \|\mathbf{M}_i(\mathbf{q}_t) - \mathbf{p}_i^*\|^2 + \lambda \left\|\mathbf{q}_t - \mathbf{q}_{t-1}\right\|^2, \tag{III.37}$$

where $\mathbf{q}_t$ is the tth frame to track, $\mathbf{q}_{t-1}$ is the tracking result from the previous frame, $\mathbf{M}_i(\cdot)$ is the ith vertex coordinates on the subject-specific hand model, and $\mathbf{p}_i^*$ is the corresponding vertex coordinates for i on the point cloud of the observed depth image, which

Figure III.17: Model-based hand pose tracking result. (top) tracking results (green) on top of reference input images; (bottom) tracked poses in a different view.

could be found using method similar to Section III.5.1.1. The former term measures the correspondence distances between the synthesized mesh and the point cloud of the depth image, and the latter is a regularization term to reduce ambiguity.

Figure III.17 shows some sample frames of the hand pose tracking result using a depth sensor. We first build the user-specific hand model as described in Section III.5.1, and then track the hand poses based on the depth data frame by frame. The figure indicates that the tracked poses fit the reference images well.

III.5.4 Model Synthesis and Interpolation

Based on our low-dimensional parametric hand model, we sample on the parametric space randomly to synthesize natural-looking hand models in different shapes. We make smooth transition between models by linear interpolation in the parametric space. Figure I.2 shows some randomly sampled results, where shape parameters are sampled randomly and poses are picked from a database. Skinning weights are represented by colors.

III.6 Evaluation

To evaluate the representation power and effectiveness of our parametric model, we estimate the reconstruction accuracy through cross validation. In addition, we demonstrate the importance of the components in our model by comparing against alternative methods.

III.6.1 Numerical Evaluation

To evaluate the representation ability of our parametric hand model, we evaluate the average reconstruction errors on all mesh models using leave-one-out cross-validation. That is, for each subject s in the database, we use all the shape parameters except s $\{\mathbf{S}_i, \mathbf{O}_i, i \neq s\}$ to learn our parametric models $\{H_s^d\}$ by keeping d dimensions, and use them to fit all meshes in s to compute the reconstruction error.

To establish semantic correspondences between our model and the scans for fitting, we perform deformation transfer on the subject-specific models learned in Section III.4.4 to fit the scan meshes, and use vertices with identical indices as the correspondences. And then we evaluate the reconstruction errors based on these correspondences. This makes sure that the reconstruction error will not be affected by different correspondence estimation methods and thus easier to evaluate the representation power.

To compute the reconstruction error between two meshes, we use Root Mean Squared Error (RMSE) on the vertex distances, which is more sensitive to larger errors. We show the average RMSE with standard deviations in Figure (III.18). The horizontal axis represents the degrees of freedom (DOF) for the scales and the vertex offsets models, the vertical axis on the left represents the average RMSE, and the vertical axis on the right represents the percentage of energies preserved by both scales and vertex offsets models. We determine the DOFs by keeping different energies for each model, including 10%, 30%, 50%, 60%, 70%, 80%, 90%, 95%, and 98% as shown in the orange curve, resulting in the DOFs of 2, 3, 5, 8, 14, 23, 43, 65, 95, respectively. The average reconstruction

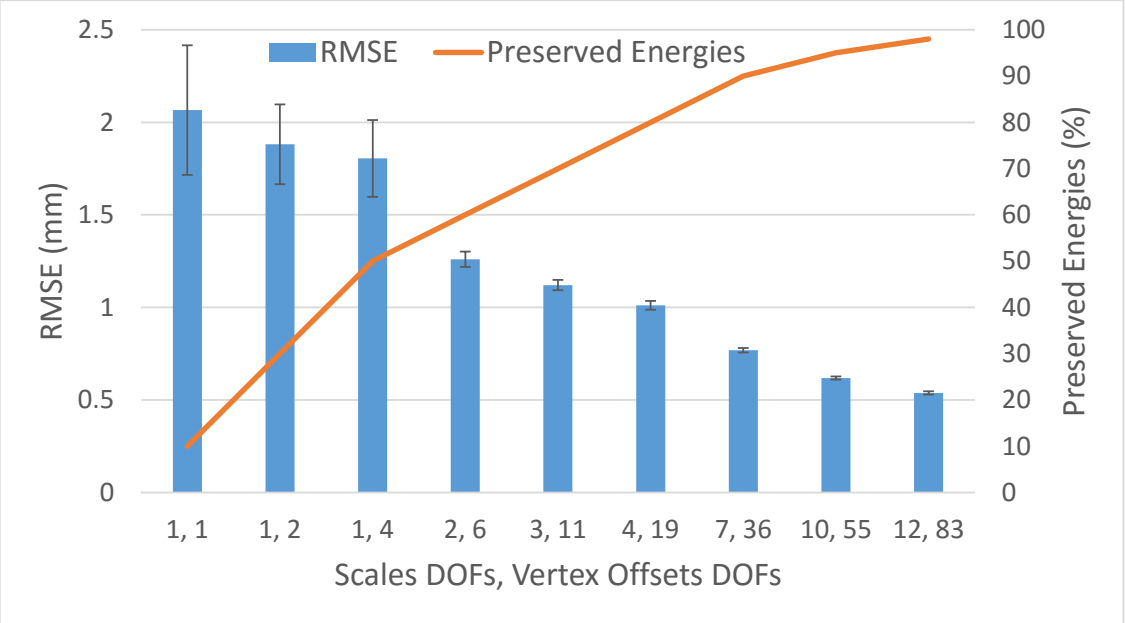


Figure III.18: Evaluation of reconstruction accuracy using cross validation (average root mean squared errors for vertex distances and variances) tested on 466 mesh models. The blue columns represent the average RMSE for models with different DOFs. The orange curve represents the percentage of energies preserved for both scales and vertex offsets models.

RMSEs and standard deviation are shown in the blue columns.

Our parametric model performs well when given a new subject not in the database, with reconstruction errors ranging from 0.54*mm* to 2.1*mm* for different DOFs. When more energies are preserved, we could get significant lower reconstruction error at the cost of higher DOFs. We can clearly see that the skeleton scale model plays an important role in reducing the error since comparing to the DOFs of the vertex offset model, the error tends to reduce quicker when the DOF for the scale model increases.

III.6.2 Component Evaluation

In this section we evaluate the key components of our parametric hand model.

III.6.2.1 Importance of the Hand Shape Model

To evaluate the effectiveness of our hand shape model, we compare the reconstruction error against the PCA model, which models only the vertex offset variations. The model is built based on the subject-specific template mesh estimated in Section III.4.4 using PCA. Similar to Section III.6.1, we perform a leave-one-out cross validation on the PCA model to reconstruct one subject while training using all other subjects. To make the comparison fair, we use the same template shape meshes for each subject as in III.6.1, and keep the same DOFs for the PCA models.

The comparison result is shown in Figure III.19, from which we can see that our model with skeleton scales clearly outperforms the PCA model for all DOFs. The reconstruction error of the PCA model reduces approximately linear, but the error of our model reduces more quickly when the DOF of the scale model increases. This demonstrates the representation power of our shape model.

III.6.2.2 Importance of Skinning Weights Learning

To evaluate our skinning weight learning algorithm that uses all meshes for learning, we compare the reconstruction error against Dionne et al. [4], that uses only a single mesh/skeleton for learning. We use cross validation to evaluate the deformation error. That is, for each subject, we learn the skinning weights using all other meshes not from this subject, and then we get the deformed meshes for s to compute the distance to the ground truth. For Dionne et al. [4], we compute the vertex distance to all meshes using the same skinning weights since it is learned only from the template. Similar to Section III.6.1, we use the same set of shape, pose parameters, and correspondences for each mesh to learn and evaluate the error, and we compute the reconstruction using RMSE.

The result shows that our skin weight learning algorithm achieves an average RMSE of $0.6mm$ with a standard deviation of $0.077mm$, which is better than Dionne et al. [4] with

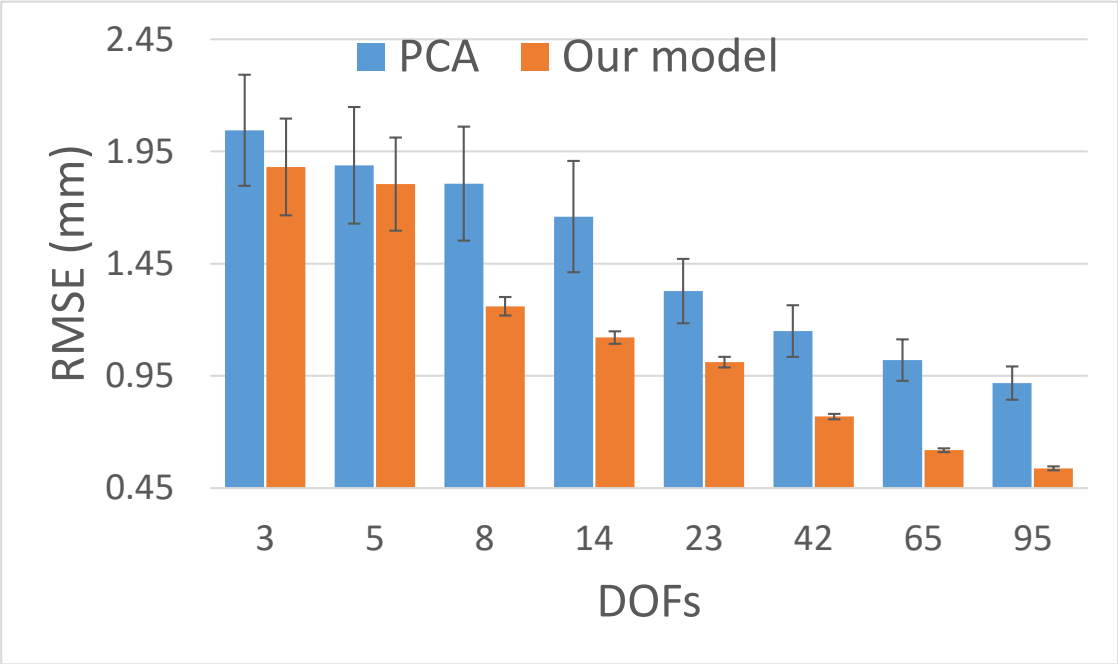


Figure III.19: Comparison of reconstruction accuracy for two algorithms using cross validation (average root mean squared errors for vertex distances) tested on 466 mesh models, we could see a significant reduction in error when more scale DOFs are used .

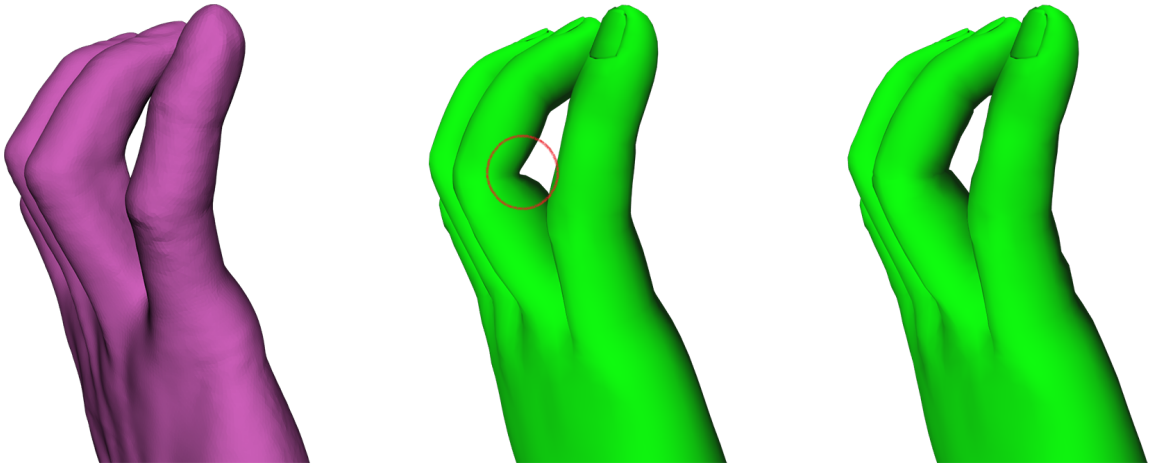


Figure III.20: Skinning weights learning comparison. (left) scan mesh; (middle) estimated mesh using skinning weights from Dionne et al. [4]; (right) estimated mesh after skinning weights learning.

average RMSE 0.65*mm* with a standard deviation of 0.083*mm*. The difference may not seem significant because skinning weight learning only applies to a small part of vertices where two or more bones connected. Most vertices that are only controlled by one bone are not affected.

Despite the numerical differences are not significant, we could still perceptually observe some improvements as shown in Figure III.20. Figure III.20 (left) is the ground truth scan mesh, and the result from Geodesic Voxel Binding [4] and our learning result are shown in the middle and right columns. We could see that our result fits better at the joint of the middle finger, as shown in the red circle.

III.7 Discussion

In this chapter, we introduce a low-dimensional parametric model for human hand modeling and develop a system for user-specific model synthesis. The main contribution of our work is to propose a new parametric model that incorporates global scale, skeleton

bone scales, vertex offsets and skeletal poses that produces a standard LBS model with skinning weights. In addition, we propose an efficient iterative approach to learn the model from a large set of unaligned scan database. Our model is appealing for hand modeling because it is compact, expressive, has strong generalization ability and produces a natural-looking LBS model for pose deformation. We demonstrate the power and effectiveness of our parametric model by exploiting a variety of applications, ranging from user-specific hand modeling from input like depth sensor, partial scans, color images, and semantic measurements, to skeleton and skinning weights transfer and model-based hand tracking.

One limitation of our system is that our shape/pose estimation process requires a set of labels to avoid falling into local minimum and thus is not fully automatic. This labeling process could be time-consuming for a large database. One way to mitigate this problem is to use a "bootstrap" approach. That is, we first label a small set of representative scans and use them to build the parametric model. Then we fit the parametric model to a new set of scans to determine the labels. We correct the labels if they do not match accurately. We then update the parametric model with the newly labeled scans and use it to fit and label another set of scans. We repeat this process and build the model incrementally until all the new set of scans could be labeled accurately.

We show that our parametric model could be applied to various applications like reconstruction from semantic measurements, we believe that our model could also be leveraged in many other applications, such as interactive hand editing, hand details transfer, and hand model compression. One of the immediate directions for future work is, therefore, to investigate more applications of our parametric model.

CHAPTER IV

CONCLUSIONS AND FUTURE WORK

Human motion capture using low-cost sensors is a challenging but important problem that may bring potential impacts to a variety of fields. This dissertation presents two aspects of systems for motion capture, which allow novel users to construct user-specific hand shape models as well as capture full-body kinematic and dynamic data accurately using non-intrusive and low-cost sensors.

In Chapter II, we propose an end-to-end full-body motion capture system using input data captured by three depth cameras and a pair of pressure-sensing shoes. Our system is appealing because it is low-cost, fully automatic, and can accurately reconstruct full-body kinematic and dynamic data. The system is also non-intrusive and easy to set up because it requires no markers and no special suits. We have demonstrated the power of our approach by capturing a wide range of complex human movements. The system achieves state-of-the-art accuracy when compared against alternative methods.

In Chapter III, we introduce a low-dimensional parametric model for human hand modeling and develop a system for user-specific model synthesis. The main contribution of our work is to propose a new parametric model that incorporates global scale, skeleton bone scales, vertex offsets, and skeletal poses and produces a standard LBS model with skinning weights. In addition, we propose an efficient iterative approach to learn the model from a large unaligned scan database. Our model is appealing for hand modeling because it is compact, expressive, has strong generalization ability, and produces a natural-looking LBS model for pose deformation. We demonstrate the power and effectiveness of our parametric model by exploiting a variety of applications, ranging from user-specific hand modeling from input like depth sensor, partial scans, color images, and semantic measure-

ments, to skeleton/skin weights transfer and model-based hand tracking. We also evaluate our model by comparing against alternative methods.

Our full-body motion capture system uses a sequential tracking framework to reconstruct 3D skeletal poses, which plays an important role for accurate tracking since temporal information is utilized. However, the system may fail to track the pose and propagate the failure to later frames due to bad pose initialization, noisy or significant loss of depth data. One important future direction to address this limitation is to integrate our tracking framework with a pose regression process that estimates the 3D pose from depth data without using temporal information. Various methods (*e.g.*, [20, 22, 70, 71, 72]) have been proposed to estimate the 3D pose from a single depth image. Recent methods [73, 74] further take the hierarchy skeleton into account to regress the pose incrementally in a cascaded way. They use features defined in the local coordinates of the skeleton for regression, which are invariant to the pose. However, these features are still not invariant to skeleton size and body shape, so they may not generalize well for different subjects.

To make the features invariant to skeleton size and body shape, we could take our parametric shape model into consideration in defining the feature positions. However, the parametric model we used for our full-body shape modeling process in Chapter II may not work, since it depends only on the vertex offsets and the information of the skeleton are ignored. In contrast, our parametric hand model proposed in Chapter III is a good match for this purpose. Although we present this model for 3D hand modeling, it also works for human body modeling since they are only different in the structure of the skeleton. Our model applies shape deformation on the template mesh based on local coordinates of the template skeleton, meaning that points defined in these coordinates are invariant to the shape. Hence, we could define each feature position by its closest vertex and a local offset in the local coordinate of the corresponding bone. When computing the feature positions for a different shape, both vertex position and the local offset are deformed based on

the shape parameters to achieve both shape and pose invariants. Another possible way to define invariant feature is to compute the feature position as a linear combination of nearby vertices.

Based on this shape invariant feature position, we could define the feature based on the signed distance field, since we have depth data from multiple depth cameras, and train the regressors using standard learning algorithms. We could further improve the regression performance by estimating the pose by parts along the skeleton hierarchy using separately trained cascaded regressors as suggested by Sun et al. [74]. That is, we first estimate the pose of the torso based on the features defined in the local coordinate of the root joint. And then we estimate the poses for the four limbs separately using regressors defined in the coordinates of shoulder and hip joints, on top of the estimated torso pose. Finally, we combine these results together to get the estimated pose.

Another important problem for the regression process is how to combine the 3D pose regressor into our kinematic tracking framework to achieve more robust results. A simple combination that initializes the pose with the regressor and then refines the pose with the tracker may not work well since the pose regressor may produce inaccurate results due to noisy or incomplete data caused by large occlusion, and temporal information is totally discarded. According to our experiments, we found that the pose tracking process works pretty well in most cases and will only fail in a small number of frames. Hence, we will track the pose first and then if it fails, we will reinitialize the pose using the pose regressor. To determine if a pose is a failure, we first synthesize the depth and silhouette images of the pose using each camera viewpoint, and then check if the depth differences and the overlapping regions are small enough. If either of them is larger than the threshold, we mark the pose as a failure and use the pose regressor to reinitialize it.

The main topic of our dissertation is focused on constructing user-specific shape model as well as capturing full-body kinematic and dynamic data using low-cost sensors. How-

ever, it is also important to capture other types of motions such as facial performance and hand articulations, which will allow our systems for wider applications such as in virtual reality and human-computer interaction. For example, if facial performance and hand gestures are also captured, it would allow people to communicate in the virtual world in a more expressive way. Capturing hand gestures would also be helpful for the applications like automatic sign language translation, making the users communicate with others easier. Hence, in the future, we are interested in extending our systems to capture hand motion and facial performance using low-cost sensors such as depth, IR, or video cameras.

REFERENCES

[1] Tekscan, "http://www.tekscan.com/," 2014.

[2] X. Wei, P. Zhang, and J. Chai, "Accurate realtime full-body motion capture using a single depth camera," *ACM Trans. Graph.*, vol. 31, pp. 188:1–188:12, Nov. 2012.

[3] Vicon Systems, "http://www.vicon.com," 2014.

[4] O. Dionne and M. de Lasa, "Geodesic voxel binding for production character meshes," in *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, (New York, NY, USA), pp. 173–180, ACM, 2013.

[5] Ascension, "http://www.ascension-tech.com/," 2014.

[6] Xsens, "http://www.xsens.com," 2014.

[7] Microsoft Kinect API for Windows, "http://www.microsoft.com/en-us/kinectforwindows/," 2014.

[8] Intel RealSense SDK, "https://software.intel.com/en-us/intel-realsense-sdk," 2016.

[9] S. Knoop, S. Vacek, and R. Dillmann, "Sensor fusion for 3d human body tracking with an articulated 3d body model," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 1686–1691, May 2006.

[10] D. Grest, V. Krüger, and R. Koch, "Single view motion tracking by depth and silhouette information," in *Proceedings of the 15th Scandinavian Conference on Image Analysis*, SCIA'07, (Berlin, Heidelberg), pp. 719–729, Springer-Verlag, 2007.

[11] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Comput. Vis. Image Underst.*, vol. 104, pp. 90–126, Nov. 2006.

[12] C. Bregler, J. Malik, and K. Pullen, "Twist based acquisition and tracking of animal and human kinematics," *Int. J. Comput. Vision*, vol. 56, pp. 179–194, Feb. 2004.

[13] D. Vlasic, I. Baran, W. Matusik, and J. Popović, "Articulated mesh animation from multi-view silhouettes," *ACM Trans. Graph.*, vol. 27, pp. 97:1–97:9, Aug. 2008.

[14] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun, "Performance capture from sparse multi-view video," *ACM Trans. Graph.*, vol. 27, pp. 98:1–98:10, Aug. 2008.

[15] V. Pavlovic, J. M. Rehg, and J. MacCormick, "Learning switching linear models of human motion," in *Proceedings of the 13th International Conference on Neural Information Processing Systems*, NIPS'00, (Cambridge, MA, USA), pp. 942–948, MIT Press, 2000.

[16] R. Urtasun, D. J. Fleet, A. Hertzmann, and P. Fua, "Priors for people tracking from small training sets," in *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01*, ICCV '05, (Washington, DC, USA), pp. 403–410, IEEE Computer Society, 2005.

[17] R. Rosales and S. Sclaroff, "Specialized mappings and the estimation of human body pose from a single image," in *Proceedings of the Workshop on Human Motion (HUMO'00)*, HUMO '00, (Washington, DC, USA), pp. 19–, IEEE Computer Society, 2000.

[18] A. Elgammal and C.-S. Lee, "Inferring 3d body pose from silhouettes using activity manifold learning," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR'04, (Washington, DC, USA), pp. 681–688, IEEE Computer Society, 2004.

[19] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun, "Real-time identification and localization of body parts from depth images," in *2010 IEEE International Conference on Robotics and Automation*, pp. 3108–3113, May 2010.

[20] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '11, (Washington, DC, USA), pp. 1297–1304, IEEE Computer Society, 2011.

[21] A. Baak, M. Muller, G. Bharaj, H.-P. Seidel, and C. Theobalt, "A data-driven approach for real-time full body pose reconstruction from a depth camera," in *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, (Washington, DC, USA), pp. 1092–1099, IEEE Computer Society, 2011.

[22] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys, "Accurate 3d pose estimation from a single depth image," in *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, (Washington, DC, USA), pp. 731–738, IEEE Computer Society, 2011.

[23] M. A. Brubaker and D. J. Fleet, "The kneed walker for human pose tracking," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2008.

[24] M. Vondrak, L. Sigal, and O. C. Jenkins, "Physical simulation for probabilistic motion tracking," in *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2008.

[25] X. Wei and J. Chai, "Videomocap: Modeling physically realistic human motion from monocular video sequences," *ACM Trans. Graph.*, vol. 29, pp. 42:1–42:10, July 2010.

[26] M. Vondrak, L. Sigal, J. Hodgins, and O. Jenkins, "Video-based 3d motion capture through biped control," *ACM Trans. Graph.*, vol. 31, pp. 27:1–27:12, July 2012.

[27] K. Yin and D. K. Pai, "Footsee: An interactive animation system," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, (Aire-la-Ville, Switzerland, Switzerland), pp. 329–338, Eurographics Association, 2003.

[28] S. Ha, Y. Bai, and C. K. Liu, "Human motion reconstruction from force sensors," in *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, (New York, NY, USA), pp. 129–138, ACM, 2011.

[29] R. Adelsberger and G. TrÃűster, "Pimu: A wireless pressure-sensing imu," in *2013 IEEE Eighth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pp. 271–276, April 2013.

[30] N. Magnenat-Thalmann, R. Laperrière, and D. Thalmann, "Joint-dependent local deformations for hand animation and object grasping," in *Proceedings on Graphics Interface '88*, (Toronto, Ont., Canada, Canada), pp. 26–33, Canadian Information Processing Society, 1988.

[31] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381–395, June 1981.

[32] B. Curless and M. Levoy, "A volumetric method for building complex models from range images," in *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, (New York, NY, USA), pp. 303–312, ACM, 1996.

[33] R. Bridson, S. Marino, and R. Fedkiw, "Simulation of clothing with folds and wrinkles," in *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '03, (Aire-la-Ville, Switzerland, Switzerland), pp. 28–36, Eurographics Association, 2003.

[34] E. Guendelman, R. Bridson, and R. Fedkiw, "Nonconvex rigid bodies with stacking," *ACM Trans. Graph.*, vol. 22, pp. 871–878, July 2003.

[35] S. Fisher and M. C. Lin, "Deformed distance fields for simulation of non-penetrating flexible bodies," in *Proceedings of the Eurographic Workshop on Computer Animation and Simulation*, (New York, NY, USA), pp. 99–111, Springer-Verlag New York, Inc., 2001.

[36] L. Kovar, J. Schreiner, and M. Gleicher, "Footskate cleanup for motion capture editing," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, (New York, NY, USA), pp. 97–104, ACM, 2002.

[37] B. Allen, B. Curless, and Z. Popović, "The space of human body shapes: Reconstruction and parameterization from range scans," *ACM Trans. Graph.*, vol. 22, pp. 587–594, July 2003.

[38] N. Hasler, C. Stoll, M. Sunkel, B. Rosenhahn, and H.-P. Seidel, "A statistical model of human pose and body shape," *Computer Graphics Forum*, vol. 28, no. 2, pp. 337–346, 2009.

[39] A. Tagliasacchi, M. Schröder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly, "Robust articulated-icp for real-time hand tracking," in *Proceedings of the Eurographics Symposium on Geometry Processing*, SGP '15, (Aire-la-Ville, Switzerland, Switzerland), pp. 101–114, Eurographics Association, 2015.

[40] A. Makris, N. Kyriazis, and A. A. Argyros, "Hierarchical particle filtering for 3d hand tracking," in *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 8–17, June 2015.

[41] W. Zhao, J. Chai, and Y.-Q. Xu, "Combining marker-based mocap and rgb-d camera for acquiring high-fidelity hand motion data," in *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '12, (Aire-la-Ville, Switzerland, Switzerland), pp. 33–42, Eurographics Association, 2012.

[42] S. Melax, L. Keselman, and S. Orsten, "Dynamics based 3d skeletal hand tracking," in *Proceedings of Graphics Interface 2013*, GI '13, (Toronto, Ont., Canada, Canada), pp. 63–70, Canadian Information Processing Society, 2013.

[43] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM Trans. Graph.*, vol. 33, pp. 169:1–169:10, Sept. 2014.

[44] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, and S. Izadi, "Accurate, robust, and flexible real-time hand tracking," in *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, (New York, NY, USA), pp. 3633–3642, ACM, 2015.

[45] H. Huang, L. Zhao, K. Yin, Y. Qi, Y. Yu, and X. Tong, "Controllable hand deformation from sparse examples with rich details," in *Proceedings of the 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '11, (New York, NY, USA), pp. 73–82, ACM, 2011.

[46] I. Albrecht, J. Haber, and H.-P. Seidel, "Construction and animation of anatomically based human hand models," in *Proceedings of the 2003 ACM SIG-*

GRAPH/Eurographics Symposium on Computer Animation, SCA '03, (Aire-la-Ville, Switzerland, Switzerland), pp. 98–109, Eurographics Association, 2003.

[47] T. Kurihara and N. Miyata, "Modeling deformable human hands from medical images," in *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '04, (Aire-la-Ville, Switzerland, Switzerland), pp. 355–363, Eurographics Association, 2004.

[48] J. P. Lewis, M. Cordner, and N. Fong, "Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation," in *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, (New York, NY, USA), pp. 165–172, ACM Press/Addison-Wesley Publishing Co., 2000.

[49] P. G. Kry, D. L. James, and D. K. Pai, "Eigenskin: Real time large deformation character skinning in hardware," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '02, (New York, NY, USA), pp. 153–159, ACM, 2002.

[50] A. Mohr and M. Gleicher, "Building efficient, accurate character skins from examples," *ACM Trans. Graph.*, vol. 22, pp. 562–568, July 2003.

[51] A. Jacobson and O. Sorkine, "Stretchable and twistable bones for skeletal shape deformation," *ACM Trans. Graph.*, vol. 30, pp. 165:1–165:8, Dec. 2011.

[52] T. Rhee, U. Neumann, and J. P. Lewis, "Human hand modeling from surface anatomy," in *Proceedings of the 2006 Symposium on Interactive 3D Graphics and Games*, I3D '06, (New York, NY, USA), pp. 27–34, ACM, 2006.

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

quences," in *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pp. 644–651, June 2014.

[54] B. Allen, B. Curless, and Z. Popović, "The space of human body shapes: Reconstruction and parameterization from range scans," *ACM Trans. Graph.*, vol. 22, pp. 587–594, July 2003.

[55] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, "Scape: Shape completion and animation of people," *ACM Trans. Graph.*, vol. 24, pp. 408–416, July 2005.

[56] Y. Chen, Z. Liu, and Z. Zhang, "Tensor-based human body modeling," in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '13, (Washington, DC, USA), pp. 105–112, IEEE Computer Society, 2013.

[57] D. A. Hirshberg, M. Loper, E. Rachlin, and M. J. Black, "Coregistration: Simultaneous alignment and modeling of articulated 3d shape," in *Proceedings of the 12th European Conference on Computer Vision - Volume Part VI*, ECCV'12, (Berlin, Heidelberg), pp. 242–255, Springer-Verlag, 2012.

[58] B. Allen, B. Curless, Z. Popović, and A. Hertzmann, "Learning a correlated model of identity and pose-dependent body shape variation for real-time synthesis," in *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '06, (Aire-la-Ville, Switzerland, Switzerland), pp. 147–156, Eurographics Association, 2006.

[59] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, "Smpl: A skinned multi-person linear model," *ACM Trans. Graph.*, vol. 34, pp. 248:1–248:16, Oct. 2015.

[60] S. Khamis, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon, "Learning an efficient model of hand shape variation from depth images," in *CVPR*, IEEE âĂŞ Institute of Electrical and Electronics Engineers, June 2015.

[61] D. Joseph Tan, T. Cashman, J. Taylor, A. Fitzgibbon, D. Tarlow, S. Khamis, S. Izadi, and J. Shotton, "Fits like a glove: Rapid and reliable hand shape personalization," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[62] I. Baran and J. Popović, "Automatic rigging and animation of 3d characters," *ACM Trans. Graph.*, vol. 26, July 2007.

[63] S. Schaefer and C. Yuksel, "Example-based skeleton extraction," in *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, (Aire-la-Ville, Switzerland, Switzerland), pp. 153–162, Eurographics Association, 2007.

[64] N. Hasler, T. Thormählen, B. Rosenhahn, and H.-P. Seidel, "Learning skeletons for shape and pose," in *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '10, (New York, NY, USA), pp. 23–30, ACM, 2010.

[65] E. De Aguiar, C. Theobalt, S. Thrun, and H.-P. Seidel, "Automatic conversion of mesh animations into skeleton-based animations," *Computer Graphics Forum*, vol. 27, no. 2, pp. 389–397, 2008.

[66] B. H. Le and Z. Deng, "Smooth skinning decomposition with rigid bones," *ACM Trans. Graph.*, vol. 31, pp. 199:1–199:10, Nov. 2012.

[67] B. H. Le and Z. Deng, "Robust and accurate skeletal rigging from mesh sequences," *ACM Trans. Graph.*, vol. 33, pp. 84:1–84:10, July 2014.

[68] R. W. Sumner and J. Popović, "Deformation transfer for triangle meshes," *ACM Trans. Graph.*, vol. 23, pp. 399–405, Aug. 2004.

[69] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *Trans. Evol. Comp*, vol. 6, pp. 58–73, Feb. 2002.

[70] R. Girshick, J. Shotton, P. Kohli, A. Criminisi, and A. Fitzgibbon, "Efficient regression of general-activity human poses from depth images," in *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, (Washington, DC, USA), pp. 415–422, IEEE Computer Society, 2011.

[71] T. Sharp, "The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, (Washington, DC, USA), pp. 103–110, IEEE Computer Society, 2012.

[72] J. Shotton, "Conditional regression forests for human pose estimation," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, CVPR '12, (Washington, DC, USA), pp. 3394–3401, IEEE Computer Society, 2012.

[73] P. DollÃąr, P. Welinder, and P. Perona, "Cascaded pose regression," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 1078–1085, June 2010.

[74] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, "Cascaded hand pose regression," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 824–832, June 2015.

APPENDIX A

OBJECTIVE FUNCTION LINEARIZATION FOR KINEMATIC POSE TRACKING

In this section, we show how to linearize the non-linear expressions in Equation (II.4), (II.5), (II.6), (II.7) and (II.9) defined in Chapter II so that the non-linear least-square problem can be iteratively solved via linear system solvers.

We first discuss how to linearize the *signed distance field term*. This term can be linearized by using first-order Taylor expansion. Suppose we have the pose for previous frame $\mathbf{q}_{i-1}$, we can get the pose for current frame $\mathbf{q}_i$ by computing a $\delta\mathbf{q}$ using an optical flow-like algorithm.

By assuming the constancy of the signed distance value for the voxels in the volume, we get

$$V(\mathbf{p},t) = V(\mathbf{p}+\delta\mathbf{q}, t+\delta t), \tag{A.1}$$

where $V(\mathbf{p},t)$ represents the signed distance value for the voxel at position $\mathbf{p}$ at time t.

For a pose $\mathbf{q}$, we can compute the world coordinate of any point $\mathbf{p}(\mathbf{q})$ on the mesh model by forward kinematics and skeleton subspace deformation. Therefore, for any point on the model, we have

$$V(\mathbf{p}(\mathbf{q}),t) = V(\mathbf{p}(\mathbf{q}+\delta\mathbf{q}), t+\delta t). \tag{A.2}$$

By expanding $P(q+\Delta q)$ and $V(P(q+\Delta q), t+\Delta t)$ using Taylor expansion, we get

$$\mathbf{p}(\mathbf{q}+\delta\mathbf{q}) = \mathbf{p}(\mathbf{q}) + \frac{\partial\mathbf{p}}{\partial\mathbf{q}}\delta\mathbf{q}, \tag{A.3}$$

$$
\begin{aligned}
V(\mathbf{p}(\mathbf{q}),t) &= V(\mathbf{p}(\mathbf{q}) + \frac{\partial \mathbf{p}}{\partial \mathbf{q}}\delta\mathbf{q}, t+\delta t) \\
&= V(\mathbf{p}(\mathbf{q}),t) + \frac{\partial V}{\partial \mathbf{p}}\frac{\partial \mathbf{p}}{\partial \mathbf{q}}\delta\mathbf{q} + \frac{\partial V}{\partial t}. \qquad \text{(A.4)}
\end{aligned}
$$

Hence, we have

$$
\frac{\partial V}{\partial \mathbf{p}}\frac{\partial \mathbf{p}}{\partial \mathbf{q}}\delta\mathbf{q} = -(V_{t+1} - V_t), \qquad \text{(A.5)}
$$

$$
A_{SDF}\delta\mathbf{q} = B_{SDF}, \qquad \text{(A.6)}
$$

where $A_{SDF} = \frac{\partial V}{\partial \mathbf{p}}\frac{\partial \mathbf{p}}{\partial \mathbf{q}}$, $B_{SDF} = -(V_{t+1} - V_t)$, $\frac{\partial V}{\partial \mathbf{p}} = [\frac{\partial V}{\partial x}, \frac{\partial V}{\partial y}, \frac{\partial V}{\partial z}]$ is the gradient of the signed distance field, and $\frac{\partial \mathbf{p}}{\partial \mathbf{q}}$ is the Jacobian matrix for the point $\mathbf{p}$ with respect to $\mathbf{q}$.

Next we discuss the linearization of the *boundary, pressure data and ground penetration terms.* These terms can be linearized in a similar way. For the corresponding point pairs $\mathbf{p}_i(\mathbf{q})$ and $\mathbf{p}_i^*$, where $\mathbf{p}_i(\mathbf{q})$ is the ith point on the model for pose $\mathbf{q}$, $\mathbf{p}_i^*$ is the target position of $\mathbf{p}_i(\mathbf{q})$, we have

$$
\mathbf{p}_i(\mathbf{q} + \delta\mathbf{q}) = \mathbf{p}_i^*. \qquad \text{(A.7)}
$$

We can linearize the left part of the equation and get

$$
\mathbf{p}_i(\mathbf{q} + \delta\mathbf{q}) = \mathbf{p}_i(\mathbf{q}) + \frac{\partial \mathbf{p}_i}{\partial \mathbf{q}}\delta\mathbf{q} = \mathbf{p}_i^*. \qquad \text{(A.8)}
$$

Hence we have

$$
\frac{\partial \mathbf{p}_i}{\partial \mathbf{q}}\delta\mathbf{q} = \mathbf{p}_i^* - \mathbf{p}_i(\mathbf{q}), \qquad \text{(A.9)}
$$

$$
A_{IK}\delta\mathbf{q} = B_{IK}, \qquad \text{(A.10)}
$$

where $A_{IK} = \frac{\partial \mathbf{p}_i}{\partial \mathbf{q}}$ and $B_{IK} = \mathbf{p}_i^* - \mathbf{p}_i(\mathbf{q})$.

Finally, we discuss the *prior term*. For the prior term $E_{Prior}(\mathbf{q})$, we have

$$\begin{aligned} E(q) &= P_k^T\left(P_k(\mathbf{q}-\mu)\right)+\mu-\mathbf{q} \\ &= (P_k^T P_k - I)(\mathbf{q}-\mu) \end{aligned} \tag{A.11}$$

and

$$\begin{aligned} E(\mathbf{q}+\delta\mathbf{q}) &= (P_k^T P_k - I)(\mathbf{q}+\delta\mathbf{q}-\mu) \\ &= A_{Prior}\delta\mathbf{q} - A_{Prior}(\mu-\mathbf{q}), \end{aligned} \tag{A.12}$$

where $A_{Prior} = P_k^T P_k - I$, I is an identity matrix.

$E(\mathbf{q}+\delta\mathbf{q})$ can be solved by $\frac{\partial E(\mathbf{q}+\delta\mathbf{q})}{\partial\delta\mathbf{q}} = 0$, thus we have

$$\frac{\partial E(\mathbf{q}+\delta\mathbf{q})}{\partial\delta\mathbf{q}} = 2E(\mathbf{q}+\delta\mathbf{q})^T\frac{\partial E(\mathbf{q}+\delta\mathbf{q})}{\partial\delta\mathbf{q}} = 0, \tag{A.13}$$

$$(A_{Prior}\delta\mathbf{q} - A_{Prior}(\mu-\mathbf{q}))^T A_{Prior} = 0, \tag{A.14}$$

$$A_{Prior}^T A_{Prior}\delta\mathbf{q} = A_{Prior}^T B_{Prior}, \tag{A.15}$$

where $A_{Prior} = P_k^T P_k - I$ is the Jacobian matrix for the term and $B_{Prior} = (P_k^T P_k - I)(\mu-\mathbf{q})$.