2015

HIGH QUALITY HUMAN 3D BODY MODELING, TRACKING AND APPLICATION

Qing Zhang
University of Kentucky, qzhan7@uky.edu

STUDENT AGREEMENT:

I represent that my thesis or dissertation and abstract are my original work. Proper attribution has been given to all outside sources. I understand that I am solely responsible for obtaining any needed copyright permissions. I have obtained needed written permission statement(s) from the owner(s) of each third-party copyrighted matter to be included in my work, allowing electronic distribution (if such use is not permitted by the fair use doctrine) which will be submitted to UKnowledge as Additional File.

I hereby grant to The University of Kentucky and its agents the irrevocable, non-exclusive, and royalty-free license to archive and make accessible my work in whole or in part in all forms of media, now or hereafter known. I agree that the document mentioned above may be made available immediately for worldwide access unless an embargo applies.

I retain all other ownership rights to the copyright of my work. I also retain the right to use in future works (such as articles or books) all or part of my work. I understand that I am free to register the copyright to my work.

REVIEW, APPROVAL AND ACCEPTANCE

The document mentioned above has been reviewed and accepted by the student's advisor, on behalf of the advisory committee, and by the Director of Graduate Studies (DGS), on behalf of the program; we verify that this is the final, approved version of the student's thesis including all changes required by the advisory committee. The undersigned agree to abide by the statements above.

Qing Zhang, Student

Dr. Ruigang Yang, Major Professor

Dr. Miroslaw Truszczynski, Director of Graduate Studies

DISSERTATION

Qing Zhang

The Graduate School
University of Kentucky
2015

HIGH QUALITY HUMAN 3D BODY MODELING, TRACKING AND APPLICATION

DISSERTATION

A dissertation submitted in partial
fulfillment of the requirements for
the degree of Doctor of Philosophy
in the College of Engineering at the
University of Kentucky

By
Qing Zhang
Lexington, Kentucky

Director: Dr. Ruigang Yang, Professor of Computer Science
Lexington, Kentucky 2015

ABSTRACT OF DISSERTATION

HIGH QUALITY HUMAN 3D BODY MODELING, TRACKING AND APPLICATION

Geometric reconstruction of dynamic objects is a fundamental task of computer vision and graphics, and modeling human body of high fidelity is considered to be a core of this problem. Traditional human shape and motion capture techniques require an array of surrounding cameras or subjects wear reflective markers, resulting in a limitation of working space and portability.

In this dissertation, a complete process is designed from geometric modeling detailed 3D human full body and capturing shape dynamics over time using a flexible setup to guiding clothes/person re-targeting with such data-driven models. As the mechanical movement of human body can be considered as an articulate motion, which is easy to guide the skin animation but has difficulties in the reverse process to find parameters from images without manual intervention, we present a novel parametric model, GMM-BlendSCAPE, jointly taking both linear skinning model and the prior art of BlendSCAPE (Blend Shape Completion and Animation for PEople) into consideration and develop a Gaussian Mixture Model (GMM) to infer both body shape and pose from incomplete observations. We show the increased accuracy of joints and skin surface estimation using our model compared to the skeleton based motion tracking.

To model the detailed body, we start with capturing high-quality partial 3D scans by using a single-view commercial depth camera. Based on GMM-BlendSCAPE, we can then reconstruct multiple complete static models of large pose difference via our novel non-rigid registration algorithm. With vertex correspondences established, these models can be further converted into a personalized drivable template and used for robust pose tracking in a similar GMM framework. Moreover, we design a general purpose real-time non-rigid deformation algorithm to accelerate this registration.

Last but not least, we demonstrate a novel virtual clothes try-on application based on our personalized model utilizing both image and depth cues to synthesize and retarget clothes for single-view videos of different people.

KEYWORDS: 3D Human Body Reconstruction, Mesh Deformation, BlendSCAPE, Gaussian Mixture Model, Virtual Try-on

Author's signature: Qing Zhang

Date: 8/6/2015

HIGH QUALITY HUMAN 3D BODY MODELING, TRACKING AND APPLICATION

By
Qing Zhang

Director of Dissertation: Ruigang Yang

Director of Graduate Studies: Miroslaw Truszczynski

Date: 8/6/2015

RULES FOR THE USE OF DISSERTATIONS

Unpublished dissertations submitted for the Doctor's degree and deposited in the University of Kentucky Library are as a rule open for inspection, but are to be used only with due regard to the rights of the authors. Bibliographical references may be noted, but quotations or summaries of parts may be published only with the permission of the author, and with the usual scholarly acknowledgments.

Extensive copying or publication of the dissertation in whole or in part also requires the consent of the Dean of the Graduate School of the University of Kentucky.

A library that borrows this dissertation for use by its patrons is expected to secure the signature of each user.

Name	Date
Qing Zhang	8/6/2015

~ Dedicated to my beloved family ~

ACKNOWLEDGMENTS

First and foremost, I would like to sincerely thank my PhD advisor Dr. Ruigang Yang for his long term generous funding support in my PhD life and studies. He is the first advisor who led me into the research field of computer science, accompanied me to overcome all kinds of difficulties along the way, and always encouraged me to become professional. During the past nine years, I have spent countless nights working for paper deadline rushes, experienced the anxiety and frustration of rejections. Ruigang is the person who asked me to wake up and overcome the obstacle. And then I realize where I am stuck and get a clearer prediction of what I should do with the complexity in research projects. I have no doubt that his optimistic foresight in the most advanced research ideas and his extraordinary skills in paper and proposal writing will guide me to face challenges in my future career. Therefore, my gratitude goes to him and his family.

Next, I am grateful to my intern mentors Zhengyou Zhang and Zicheng Liu at Microsoft Research, Redmond, where I started my first professional career in an industrial research project. Zhengyou taught me the truth of critical thinking and pointed out my potential to be a scientist. Zicheng shared a lot of his research experience with me. His strong mathematics skills that have been successfully applied in his research career motivated me to pursue my concurrent master degree in mathematics. I also need to mention that one of our serious discussions about the state of the art in human body modeling gave birth to this dissertation.

I also would like to thank my mentors at Microsoft Research Asia, Beijing, Bennett Wilburn and Yasuyuki Matsushita who guided me how to productively write papers from any possible subtle problems and how to create experiential setup to support my researches wisely. Also the vision group director, Yi Ma showed me his talent of

how to bring the most advanced theoretic researches into solving practical computer science problems.

I would like to my dissertation committee: Fuhua (Frank) Cheng, Brent Seales, Jun Zhang and Qiang Ye. With special thanks to Frank, who is also my grand-advisor in Tsinghua University and a co-advisor of my PhD study, for his kind and suggestive advice throughout all of my research career. And I would also like to give special thanks to Qiang Ye, who is the advisor of my master study in Department of Mathematics and gave me a lot of support in solving optimization problems. And I would also like to thank Debby Keen, the mentor of my teaching job, for giving me many suggestions on teaching skills and how to present myself clearly.

Last but not least, I would like to thank all of my coauthors and colleagues in Graphics and Vision Technology (Gravity) Lab. They accompanied me to experience the toughness of research life and collaborated together in making plausible paper results and all kinds of interesting stuffs.

TABLE OF CONTENTS

Acknowledgments . . . iii

Table of Contents . . . v

List of Tables . . . vii

List of Figures . . . viii

Chapter 1 Introduction . . . 1
1.1 Dissertation Statement . . . 1
1.2 Background . . . 1
1.3 Trends in Related Work . . . 2
1.4 Dissertation Overview . . . 6

Chapter 2 Gaussian Mixture Based Human Body Shape and Pose Model . . . 9
2.1 Previous Work . . . 9
2.2 GMM-BlendSCAPE . . . 11

Chapter 3 Single View 4D Self Portrait Framework . . . 26
3.1 Previous Work . . . 28
3.2 Pairwise Nonrigid Registration Framework . . . 30
3.3 Global Nonrigid Registration Algorithm . . . 33
3.4 Training of Parametric Kinematics . . . 37
3.5 Results and Evaluation . . . 39

Chapter 4 Real Time General Mesh Embedded Deformation . . . 43
4.1 Previous Work . . . 45
4.2 Linear Embedded Deformation . . . 47
4.3 Real Time Algorithm for General Mesh Embedded Deformation . . . 51
4.4 Applications and Results . . . 55

Chapter 5 Body Swap: Application to Virtual Try-On . . . 61
5.1 Related Work . . . 62
5.2 System Overview and Preliminary . . . 65
5.3 Depth Sequence Tracking . . . 66
5.4 Video Re-targeting . . . 72
5.5 Results . . . 76

Chapter 6 Conclusion . . . 78
6.1 Contributions . . . 78
6.2 Future work . . . 79

Appendix: Rigid Transformation Representation and Solver 80

Bibliography 84

Vita 94

LIST OF TABLES

4.1 The complete model is built from eight partial scans while the user rotates himself about 45° degrees for each shot. The average number of nodes used to align each two scans is 1000. The 3D Self-Portrait [1] takes 2040 seconds while our method takes 12.16 seconds for 5 iterations in total. 56
4.2 A benchmark dataset 57
4.3 Speed test on synthetic data. The time unit is second. 58
4.4 The interactive mesh editing results 59

LIST OF FIGURES

1.1 A pipeline of nonrigid reconstruction framework. Multiple single view scans are combined to build multiple complete 3D models that serves training samples for a final animatable avatar. A fitted pose is showed for a given point cloud. 5
1.2 An example of the body swap application. A pre-recorded video (left) is customized to a user (right) by taking a KinectFusion scan or input body sizes. The application provides user (Person B) a feeling of what it looks like by trying on virtual clothes. 6
1.3 Dissertation Chapter Overview 7

2.1 A transformation of triangle from template to a target pose. 12
2.2 The effect of the regularization weight λ_Q. 14
2.3 The mean error from the template to 70 training data using the ground truth vertex correspondence. The process converges almost within 6 iterations. 18
2.4 The first 6 iterations of computed skinning weight and skeleton. In the above row, only skinning weights for right upper arm, left shoulder and pelvis part are color coded. 18
2.5 A pose driven sequence comparing the LBS system (above row) and our model (below row). The LBS system cannot accurately represent the surface around areas such as the armpit even with optimized skinning weights. 19
2.6 Auto registration results from the template to point cloud of sample poses. The error is computed by ground truth correspondences. The red/blue colors denote the data point cloud with random noise and the fitted model. 23
2.7 A result of fitting a female template to an incomplete point cloud sequence from Kinect sensor. Note that we estimate the shape at the beginning of the video, *e.g.*, T-pose, and then we fix shape parameters and track poses only for the rest of the video. 24
2.8 The mean joint errors of our GMMBS tracking algorithm with the provided groundtruth joint locations using PDT dataset. 25
2.9 Quantitative comparison of mean errors with existing motion tracking methods [62–64] using PDT dataset. 25

3.1 The initial alignment of partial point cloud from eight views using our GMM-BlendSCAPE template fitting. 31
3.2 We search for corresponding points by aligning patches controlled by the same graph nodes using ICP. 33

3.3 Stages in our global registration. All the partial scans are initially aligned to the target using the general template model. Virtual cameras are estimated in the coordinate system of the target pose to determine the loop closure. The fitted template model is reduced to a rough graph to guide the embedded registration. Pairwise accumulated registraton error is distrubuted after each loop adjustment. 35
3.4 The reconstructed mannequin of an almost static pose. Error map compared to the groundtruth is plotted. 40
3.5 The deviation of mannequin data. The left is the rotation angle changes in degrees and the right is the translation in milimeters. 40
3.6 The reconstructed mannequin of some articulated arm movement. 41
3.7 The reconstructed watertight models after our global registration. The bottom row shows the input partial scans and the upper row shows the reconstructed models at each pose. 41
3.8 The final fitting result with our personalized parametric avatar. We compare our avatar with the general SCAPE model to show more realistic details. 42

4.1 A graph hierarchy built for the bar-twist example. Two layers of graphs are plotted. In order to pass the transformation from a graph of coarser level, we simplify the mesh while preserving the node positions. In other words, the nodes of a lower level is the subset of a higher level. 52
4.2 The interactive editing of synthetic data. The first row shows control vertices and parts. The second and third rows are the results from the embedded method and from ours respectively. The last row shows the compared results 60

5.1 The pipeline of our body swap system. 63
5.2 The personalized BlendSCAPE models. Column a) shows the fitted BlendSCAPE model to partial scans of column c); Column b) shows the personalized model after adding the detailed shape layer on the BlendSCAPE model. 67
5.3 The pose and mesh refinement for the first five iterations. 70
5.4 The pipeline of face retargeting process 74
5.5 Illustration of dealing with occlusion cases 75
5.6 The video result of replacing the dancing girl to a taller female user. . . . 76
5.7 The video result of replacing the male dancing to a taller and stronger male user. 77
5.8 The movie reshape result by entering body sizes. The user modified the video by entering a larger bust size. 77

Chapter 1 Introduction

1.1 Dissertation Statement

with sufficient data Computer Vision can 1) capture high quality human body shapes from low-cost sensors; 2) produce realistic character animations; 3) achieve high speed and practical computation; and 4) be applied to augmented reality applications.

1.2 Background

Human body understanding has a long history of studies attracting most of scientists' interests, evidenced by famous human anatomy drawings from Leonardo da Vinci in 15th century. In the modern computer vision community, research of human body has spanned a wide range from human performance capture, action analysis to health care and daily entertainment.

Thanks to the recent emergence of high resolution cameras, such as PointGrey [2] and real-time consumer level 3D sensors, such as SwissRanger [3] and Kinect [4], it becomes possible to create high-quality 3D models using a single hand-held camera at home, *e.g.*, [5, 6]. The trend from the manufacture revolution in 3D printing industry, *e.g.* [7], has also stimulated the desire of ordinary users to create their 3D portraits in flexible and cheap ways.

With a big leap in human body modeling algorithms and software, one could access a realistic virtual avatar to show off in social media websites, advanced video games or teleconferences. By taking scans of the human body or inspecting body information regularly, future medical care could track users' health and fitness more accurately and effectively. Creating virtual human bodies can also change online shopping, *e.g.*, "Virtual Try-on", previewing virtual outfits on customer's own body

models before making decisions on purchases.

The requirement of approaches for lightweight full body acquisition rises from the fact that traditional techniques in computer vision can only collect motionless 3D information from surrounding observations, *e.g.*, [8–10], or a laser scanner [11], which are either sparse or incomplete due to occlusions like the armpit or crotch. While motion capture techniques, such as the commercial system [12], can deliver highly accurate spare point and skeleton measurements, however, it requires markers on tight suits, which may interfere with the nature pattern of locomotions or muscle deformations, and the dedicated infrared light setup is not portable and is impractical for ordinary users.

Simplification of data collection results in inevitable incompleteness and ambiguities. How to build complete models from limited observations is one of the motivations of this dissertation. Fortunately, large human measurement data projects, such as CAESAR [13], provide opportunities for statistically studying human body shapes and motions. Employing training data could be an effective and simple way to overcome such shortcomings. The main task of this dissertation, therefore, is to present automatic and efficient approaches to combine an existing database into a consumer level data capture system to build high quality human 3D body models.

1.3 Trends in Related Work

To start with existing trends in recent related researches, I basically category two main trends: mesh manipulate and data-driven human body modeling.

In the first part, how to merge partial deformable surfaces over time consistently into a complete model is considered as an is an ill-posed problem [14] since the occluded part can be in any shape at any instant. In general, this problem turns out to be a general mesh deformation and registration problem, which has been studied for decades but still remsains challenge. To deal with the free-form deformation,

many assumptions in terms of regularization has been proposed to constrain desired properties.

Existing non-rigid registration methods achieve highly accurate alignments for subtle warps, but most of them are not suitable for large-scale deformations. Chang and Zwicker [15, 16] solve a discrete labeling problem to detect the set of optimal correspondences and apply graph cuts to optimize for a consistent deformation from source to target. Huang and colleagues [17] use a technique that finds an alignment by diffusing consistent closest point correspondences over the target shape while preserving isometries as much as possible, but the correspondence search is sensitive to topological changes and holes. Mitra and colleagues [18] aggregates all scans into a 4D space-time surface and estimates inter-frame motion from kinematic properties of the deforming surface. Shart *et al.* [19] introduced a volumetric space-time reconstruction technique that represents shape motion as an incompressible flow of material through time. Wand *et al.* [20, 21] introduced a statistical framework that performs pairwise alignment and merging over all adjacent scans within a global non-linear optimization process.

Many methods make use of a template model to simplify correspondence estimation and provide a prior for geometry and topology reconstruction. Unsupervised methods are proposed that require no manual intervention [22, 23] but typically lead to higher computational complexity that makes these methods less suitable for long sequences. Park and Hodgins [24, 25] develop a system that uses a dense and large set of markers to capture and synthesize dynamic motions such as muscle bulging and flesh jiggling. Li and coworkers [26, 27] developed a registration framework that solves for point correspondences, surface deformation, and region of overlap within a single global optimization.

One of my basic assumption is, as we observe in the real world, most dynamic objects behave continuously and predictable in a short temporal interval, especially

when capturing videos of a person in a designed scene. This makes the problem trackable when the object deforms. The general deformation framework, however, mostly emphasizes on detail preservation via some simple assumptions, *e.g.*, local rotation or normal changes should be smooth, objects should deform like rubber. While these assumptions prevents implausible artifacts like stretch and shear, unfortunately, they are only simple priors suitable for physical objects but difficult to constraints human body motions in a wide range of poses and shape deformations.

The concern of "embedded deformation" provides a natural way to constrain the manipulation of mesh to the deform space of objects embedded within it. Without a strong shape prior, we still do not know what the deform space is and how the local features rotate, *e.g.*, there are large regions of the body where it is impractical to find useful correspondences. Therefore, we turn our attention from the generic regularization to data-driven methods, which integrate the strong body shape prior to prevent the registration from undergoing implausible deformations, and have the ability to explain poor or missing data and inherently resolve the ambiguities in pairwise alignment.

The data-driven models are powerful as they enable the inference of object from incomplete noisy and ambiguous 2D or 3D data. Specifically, the data-driven template can model a consistent human body of a sufficient level of details in the case that the general completion method has limitations due to insufficient point correspondences. Importantly, the representation of the template-based method allows to model the pose and the body shape deformation in each individual spaces and to be combined properly. Therefore, it greatly improves robustness to missing data and ambiguities and also provides a simple manner to describe pose dependent muscle deformations.

Similar to motion capture and skeleton tracking [28], although data-driven template serves as a strong shape prior, it is still difficult to infer accurate pose and shape without any manual intervention, in particularly, when there are significant limb oc-

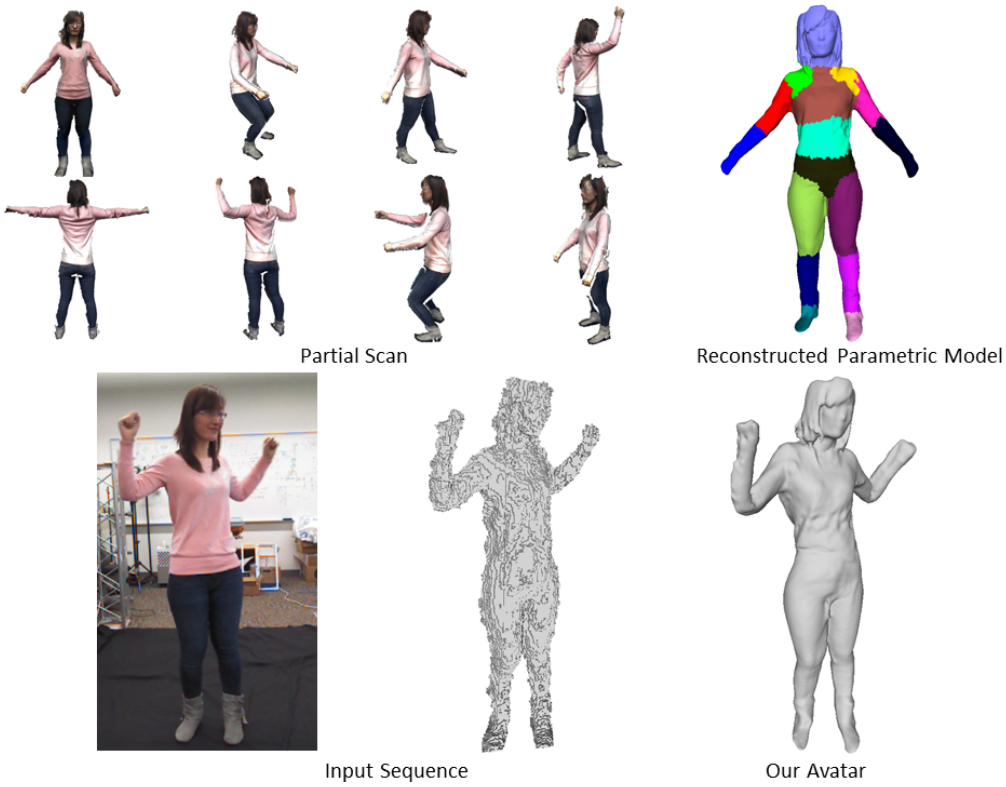


Figure 1.1: A pipeline of nonrigid reconstruction framework. Multiple single view scans are combined to build multiple complete 3D models that serves training samples for a final animatable avatar. A fitted pose is showed for a given point cloud.

clusions involved in partial body scans. Therefore, I claim that the human body model should provide sufficient level of detail, a easy and direct way to manipulate, and can be estimated robustly with little manual intervention.

In the last part, I expand the problem to clothes animation editing, which is a challenge task to recover clothes geometry from 2D images, and becomes even harder to obtain the motion when the garment swinging with rapid body movement. Unlike the traditional approaches based on clothes simulation [29] and trained clothes template [30], our method only focuses on the visual effect of different people trying on the same virtual clothes via image re-targeting technique guided by our estimated body shape.

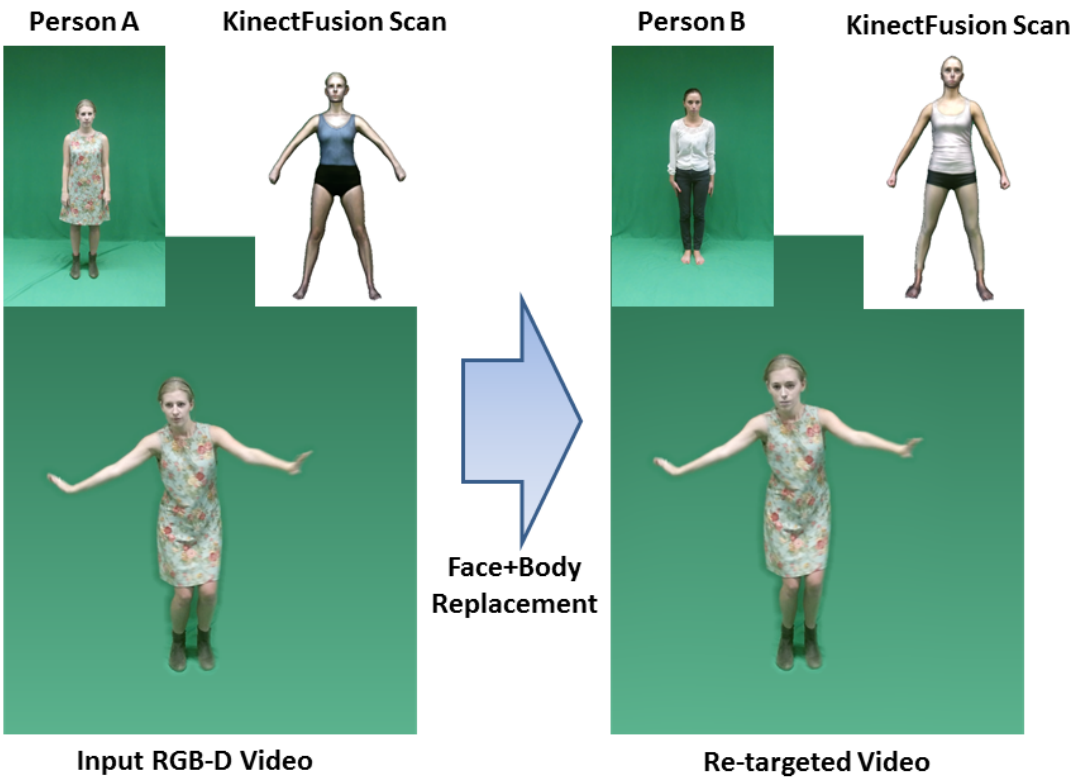


Figure 1.2: An example of the body swap application. A pre-recorded video (left) is customized to a user (right) by taking a KinectFusion scan or input body sizes. The application provides user (Person B) a feeling of what it looks like by trying on virtual clothes.

1.4 Dissertation Overview

The core contribution of this dissertation is the mathematical design of the Gaussian Mixture Model (GMM) based human shape and pose estimation framework: a general solution to estimate human body geometry from highly incomplete and noise data. The following topics unify the contribution of the dissertation in formulation, system setup, performance analysis and also its typical applications in the coming chapters.

• GMM-BlendSCAPE: the statistical human body estimation model I develop to overcome the challenge of automatic template model fitting in a general case which contains data noise, occlusions and large deformations. Different from the original BlendSCAPE model, a skinning weight optimization is designed to make this model

consistent with both BlendSCAPE and skeletal LBS system, making it more accurate and able to be driven by either approach.

• A novel nonrigid reconstruction algorithm which generates 4D complete models using multiple partial scans from a single-view depth camera. Based on the GMM-BlendSCAPE fitting scheme, a good alignment initial guess can be provided markerlessly without any manual intervention enabling a robust nonrigid registration for large pose difference. Figure 1.1 illustrates the system overview.

• A generic acceleration scheme for the embedded mesh deformation that significantly reduces the computation cost and makes the nonrigid registration practical for light-weight applications.

• Body Swap: A novel virtual clothes try-on application based on the personalized template deployed in a GMM framework to guide re-targeting of clothes video from a pre-recorded model to incoming customers. Figure 1.2 illustrates a re-targeting example.

Dissertation Roadmap

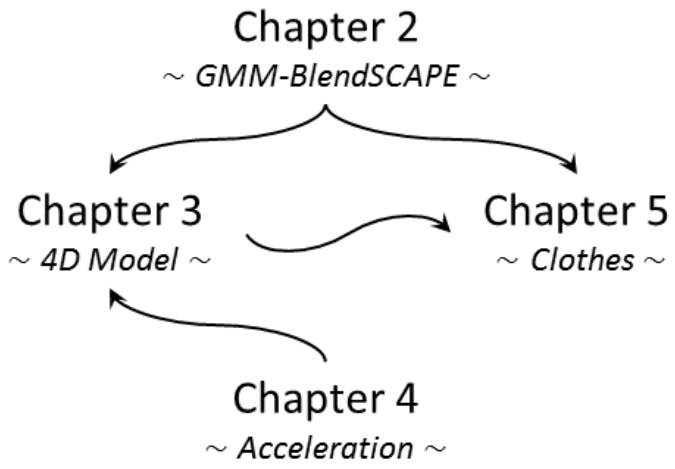


Figure 1.3: Dissertation Chapter Overview

I illustrate the structure of this dissertation in a manner as Figure 1.3.

Chapter 2 introduces the model of GMM-BlendSCAPE including the basic formulation, how to train the model from a database and customized for a certain person, and how to fit to observation efficiently.

Chapter 3 presents the nonrigid registration algorithm to build 4D complete models from low-cost depth scans based on initial alignment technique described in Chapter 2.

Chapter 4 continues the analysis of the nonrigid deformation and presents an efficient algorithm to achieve fast performance.

Chapter 5 describes a "Virtual Try-on" application of the personalized model using the presented GMM framework to reconstruct the body shape sequence and guide the 2D image editing to swap different bodies.

Chapter 6 summarizes techniques and points to limitations and future researches.

Chapter 2 Gaussian Mixture Based Human Body Shape and Pose Model

Estimating the geometry of a moving human body comprises a variety of challenges: the body shape is unknown, the pose varies a lot, the skin surface deforms nonrigidly according to movement, only a limited observation is available and also it may contain noise, *etc.* In motion capture and analysis field, the shape of the object is usually pre-obtained or has a preknowledge of its mechanical properties in general, and the problem is specifically designed for skeletal tracking [31,32]. If neither shape nor pose provided, the estimation problem is usually restrict to a specific kind of objects, *e.g.*, human body animation, and also additional controlled environment is required, such as surrounding cameras and wearing retro-reflective markers [33].

As a central contribution of this dissertation, a probabilistic human body model (GMM-BlendSCAPE) is introduced in this chapter, aiming for the goal of estimating human body robustly and markelessly of large shape and pose variations from a limited number of views, *e.g.*, a single view covering less than 50% of the subject.

2.1 Previous Work

Articulated Motion Estimation Human pose and motion estimation has a vast literature previously summarized in [34, 35]. Based on commercial video cameras, advances in methodology have been made. [36] tracked a hand wearing color coded glove in real-time. [37] proposed a local mixture of Gaussian processes to regress human pose. [38] tracked humans using twists and exponential maps from an initial pose.

The recent availability of depth cameras has spurred further progress. Based on Iterative Closest Point (ICP) approach, [39] tracked a skeleton from a starting position. [40] built heuristic detectors for upper body parts using a linear programming.

Learning based methods are proposed to label parts in depth images. [41] classified head and limbs and provide both location and orientation by finding geodesic extrema interest points. [42] clustered appearances by finding body segments as pairs of parallel lines. [43] presented poselets to detect clusters in both 3D pose and 2D image using SVMs. [44] used an auto-context to obtain a coarse body part labeling. [32] trained deep randomized decision forests to classify parts at 200 frames per second on consumer hardware.

Shape and Pose Representation Earliest animatable body models tracked the human body relying on simple geometric body shape representation [45–48]. SCAPE (Shape Completion and Animation for PEople) [49] firstly modeled a more detailed and realistic body shape using a large training database that spans variation in both subject shape and pose and can fit to incomplete and noise data.

Based on SCAPE model, many variant applications have been developed. BlendSCAPE [50], the model our fitting approach based on, took all the body parts into a blend weighted consideration without explicitly identifying each one and can be easily employed into a global fitting scheme. The Stitched Puppet [51] chopped the 3D mesh model into multiple body parts and fitted them together using a particle-based max-product belief propagation. The TenBo (Tensor-Based Human Body Modeling) [52] decomposes the shape parameters and combines the pose and shape in a tensor way to add shape variations for each body part.

Human Model Fitting The SCAPE model have been employed into many applications: Home 3D body scan [53] applied it to Kinect point cloud data and combined the silhouette information. [54] fitted the body to multi-camera image data. Naked Truth [55] estimated human body shape under clothes. [33] fitted the model to sparse markers from a motion capture system. FAUST [56] provided high resolution 3D

input scans and evaluated fitting methods with ground-truth generated by accurate texture matching.

2.2 GMM-BlendSCAPE

The novelty of the GMM-BlendSCAPE different from the existing BlendSCAPE [50] are in two folds: an adaptive skinning weight for a particular human shape and a robust template fitting deployed in a probabilistic framework.

The BlendSCAPE Model The BlendSCAPE, firstly introduced in [50] utilizing a skinning triangle mesh as the template, is a full body pose and shape deformation model. The template of human is taken at a standard A-pose as the rest pose, consisting of the surface vertex set $\mathcal{V}^0 = \{\boldsymbol{v}_m^0 \mid m = 1, \ldots, M\}$, the triangle face index set $|\mathcal{F}| = F$ and the skinning weight associated with each vertex $\boldsymbol{w} = [w_{m,b}]_{M\times B}$ of body parts or bones, indexed by b, in the kinematic tree. Suppose the rigid transformation of each bone is $[\boldsymbol{R}_b^{\boldsymbol{\theta}}\ \boldsymbol{t}_b^{\boldsymbol{\theta}}]$, the deformed vertex position is represented by a weighted sum as follows:

$$\boldsymbol{v}_m^{\boldsymbol{\theta}} = \sum_{b=1}^{B} w_{m,b}[\boldsymbol{R}_b^{\boldsymbol{\theta}}\ \boldsymbol{t}_b^{\boldsymbol{\theta}}]\boldsymbol{v}_m^0, \tag{2.1}$$

where $\boldsymbol{v}_m^0$ is in homogeneous coordinates. Given the template model of a general shape and pose, the 3×3 linear transformation $\boldsymbol{A}_f$ of a triangle face deforms each edge of the rest pose to the corresponding target edge, *i.e.*, $\boldsymbol{A}_f\boldsymbol{T}_f^0 = \boldsymbol{T}_f$, where $\boldsymbol{T}_f^0 = [\boldsymbol{v}_{f,2}^0 - \boldsymbol{v}_{f,1}^0,\ \boldsymbol{v}_{f,3}^0 - \boldsymbol{v}_{f,1}^0]$, $\boldsymbol{T}_f = [\boldsymbol{v}_{f,2} - \boldsymbol{v}_{f,1},\ \boldsymbol{v}_{f,3} - \boldsymbol{v}_{f,1}]$, and the subscript $1, 2, 3$ denotes the corresponding index of the triangle as illustrated in Figure 2.1.

The linear transformation $\boldsymbol{A}_f$ depends on the pose parameters $\boldsymbol{\theta}$, the stacked Euler vectors of body parts rotations (see the Appendix for computation 6.2), and the shape parameters $\boldsymbol{\beta}$. Specifically, $\boldsymbol{A}_f(\boldsymbol{\theta}, \boldsymbol{\beta}) = \boldsymbol{B}_f(\boldsymbol{\theta})\boldsymbol{D}_f(\boldsymbol{\beta})\boldsymbol{Q}_f(\boldsymbol{\theta})$, the three 3×3 matrix are decomposed as follows:

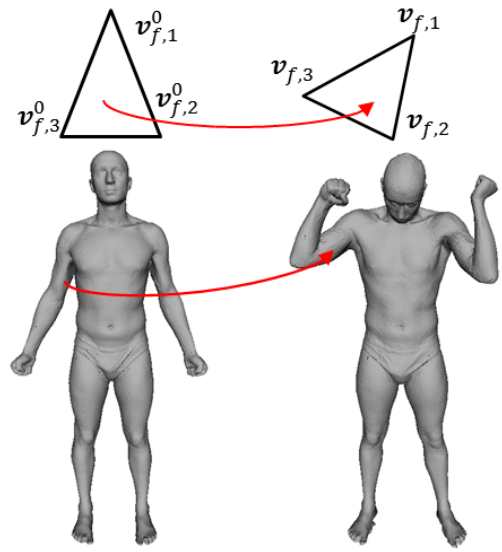


Figure 2.1: A transformation of triangle from template to a target pose.

• $\boldsymbol{B}_f(\boldsymbol{\theta})$ - a weighted "blend" of part's rotations: $\boldsymbol{B}_f(\boldsymbol{\theta}) := \sum_b w_{f,b}\boldsymbol{R}_b$ and the weight $w_{f,b}$ of a triangle is computed as the average weight of its three vertices.

• $\boldsymbol{D}_f(\boldsymbol{\beta})$ - the shape variation of different people, whose stacked $9F \times 1$ vector $\boldsymbol{D}$ can be described from a linear PCA space: $\boldsymbol{D} = \overline{\boldsymbol{U\beta} + \boldsymbol{\mu}}$, where $\boldsymbol{U}, \boldsymbol{\mu}$ are pre-trained PCA parameters, and $\boldsymbol{\beta}$ represents the coefficients.

• $\boldsymbol{Q}_f(\boldsymbol{\theta})$ - the pose related "blend" nonrigid deformation, *s.t.*, $\boldsymbol{Q}_f(\boldsymbol{\theta}) = \boldsymbol{Q}^0_f + \sum_c \theta_c \boldsymbol{Q}^c_f$, where θ_c is the c-th element of the pose vector $\boldsymbol{\theta}$ and $\boldsymbol{Q}^0_f, \boldsymbol{Q}^c_f$ are learned coefficients. Ideally, $\boldsymbol{Q}_f(\boldsymbol{0}) = \boldsymbol{Q}^0_f = \boldsymbol{I}$ and $\boldsymbol{Q}^c_f$ is sparse since the nonrigid skin and muscle deformation is only related to the rotations of a few adjacent body parts.

To build the correspondence between the template at rest pose and an arbitrary configuration, a coupling energy term is defined to stitch all triangle faces together:

$$E_c(\boldsymbol{\theta}, \boldsymbol{\beta}) = \sum_{f=1}^{F} a_f \left\| \boldsymbol{T}_f - \boldsymbol{B}_f(\boldsymbol{\theta})\boldsymbol{D}_f(\boldsymbol{\beta})\boldsymbol{Q}_f(\boldsymbol{\theta})\boldsymbol{T}^0_f \right\|^2_F, \tag{2.2}$$

where a_f is the area of triangle f on the template mesh and $\| \cdot \|_F$ stands for the Frobenius norm.

It is easy to show that if both $\boldsymbol{\theta}, \boldsymbol{\beta}$ are given, vertex positions of the deformed mesh can be determined up to a global translation by solving the linear least square problem $\min_{\boldsymbol{v}} E_c(\boldsymbol{\theta}, \boldsymbol{\beta})$.

To train BlendSCAPE, we utilize the registered CAESAR database containing two body scan corpora: one containing a person in many poses and one containing people of different shapes in roughly a fixed pose.

In the former case, $\boldsymbol{D}_f(\boldsymbol{\beta})$ is first set to identity, if $\boldsymbol{B}_f$ is given, $\boldsymbol{Q}_f$ is solved by minimizing the following function:

$$E_Q(\boldsymbol{Q}_f) = \sum_{f=1}^{F} a_f \left\| \boldsymbol{T}_f - \boldsymbol{B}_f \boldsymbol{D}_f \boldsymbol{Q}_f \boldsymbol{T}_f^0 \right\|_F^2 + w_Q \sum_{f_1, f_2 \text{ adj}} a_{f_1, f_2} \left\| \boldsymbol{Q}_{f_1} - \boldsymbol{Q}_{f_2} \right\|_F^2, \tag{2.3}$$

where $w_Q = 0.001$, $a_{f_1,f_2} = \frac{a_{f_1}+a_{f_2}}{3}$ and f_1, f_2 are adjacent faces. The problem can be solved efficiently by taking each column vector of $\boldsymbol{Q}_f$ as unknown. Once $\boldsymbol{Q}_f$ obtained, the decomposition of $\boldsymbol{Q}_f$ is solved by minimizing the object function:

$$E_{Q^c}(\boldsymbol{Q}_f^0, \boldsymbol{Q}_f^c) = \sum_{\boldsymbol{\theta}} \left\| \boldsymbol{Q}_f^0 + \sum_c \theta_c \boldsymbol{Q}_f^c - \boldsymbol{Q}_f \right\|_F^2 + \lambda_Q \left(\|\boldsymbol{Q}_f^0 - \boldsymbol{I}\|_F^2 + \sum_c \|\boldsymbol{Q}_f^c\|_F^2 \right), \tag{2.4}$$

in which λ_Q controls the relative influence of sparsity experimentally validated by comparing with the training samples. The recall errors of different λ_Q are shown in Figure 2.2. In general, the larger λ_Q is, the sparser the decomposition is. We choose $\lambda_Q = 5$ in our experiment to comprise accuracy (mean fitting error less than 2cm) and overfitting.

In the latter case, suppose $\boldsymbol{\theta}, \boldsymbol{B}_f$ are known and $\boldsymbol{Q}_f^0, \boldsymbol{Q}_f^c$ have been trained from the above for the template reference person, therefore $\boldsymbol{Q}_f$ is known, for different

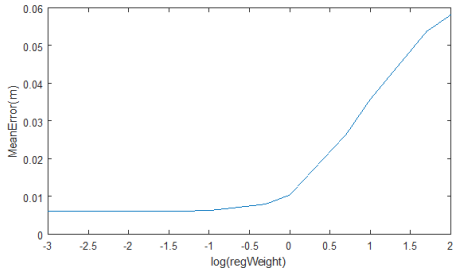


Figure 2.2: The effect of the regularization weight λ_Q.

people, the shape deformation is solved by minimizing the following function:

$$E_D(\boldsymbol{D}_f) = \sum_{f=1}^{F} a_f \left\| \boldsymbol{T}_f - \boldsymbol{B}_f \boldsymbol{D}_f \boldsymbol{Q}_f \boldsymbol{T}_f^0 \right\|_F^2 + w_D \sum_{f_1, f_2 \text{ adj}} a_{f_1,f_2} \left\| \boldsymbol{D}_{f_1} - \boldsymbol{D}_{f_2} \right\|_F^2 , \quad (2.5)$$

where $w_D = 0.001$ and each column of $\boldsymbol{D}_f$ can be solved similarly. Once $\boldsymbol{D}_f$ are solved for all the people in the database, we reshape and stack them into a $9F \times S$ matrix (S is the sample number) and apply the PCA to obtain $\boldsymbol{U}, \boldsymbol{\mu}$, *s.t.*, $\boldsymbol{D} = \overline{\boldsymbol{U}\boldsymbol{\beta} + \boldsymbol{\mu}}$.

The whole training process of BlendSCAPE is summerized in Algorithm 1, in which the step of skinning weight adaptation algorithm will be introduced in the next section.

Algorithm 1 BlendSCAPE Training

Input: registered pose dataset, registered shape dataset, initial skinning weight $\boldsymbol{w}$

Output: BlendSCAPE: $\boldsymbol{w}$, $\boldsymbol{Q}_f^0$, $\boldsymbol{Q}_f^c$, $\boldsymbol{U}$, $\boldsymbol{\mu}$

Set $\boldsymbol{\beta} = \boldsymbol{0}$, use the registered pose dataset to apply adaptation Algorithm 2
Solve $\boldsymbol{D}_f$ for all the registered shape dataset
Stack $\boldsymbol{D}_f$ and apply PCA to train $\boldsymbol{D} = \overline{\boldsymbol{U}\boldsymbol{\beta} + \boldsymbol{\mu}}$

Skinning Weight Adaptation Different from existing approaches, where skinning weights $w_{m,b}$ are either designed manually [57] or solved by diffusion techniques, *e.g.*,

heat equilibrium [58], we emphasize that skinning weights and nonrigid deformation terms $\boldsymbol{Q}_f^0, \boldsymbol{Q}_f^c$ also depend on body shapes as personal parameters and need to be estimated from multiple pose samples for a given shape.

Our skinning weight adaptation is inspired by skeletal rigging approaches [59, 60]. Suppose a skeleton has been embedded in the template mesh, for a certain pose $\boldsymbol{\theta}$, the rigid transformation of b-th bone in the kinematic tree relative to that of the rest pose is denoted by $[\boldsymbol{R}_b^{\boldsymbol{\theta}}\ \boldsymbol{t}_b^{\boldsymbol{\theta}}]$. If multiple pose samples are given as the training data ($|\{\boldsymbol{\theta}\}| =: \Theta = 70$ samples in total), we use them to optimize the skinning weights; otherwise, we synthesize Θ different sample poses for a certain body shape using the same pose parameters $\boldsymbol{\theta}$. The skinning weights are optimized by minimizing the following problem:

$$
\begin{aligned}
& E_w(\boldsymbol{w}) = E_{wd} + E_{w1} + \lambda_s E_{ws}, \\
\text{where} \quad & E_{wd} = \frac{1}{M\Theta} \sum_{m=1}^{M} \sum_{\boldsymbol{\theta}} \left\| \sum_{b=1}^{B} w_{m,b} \boldsymbol{R}_b^{\boldsymbol{\theta}} \boldsymbol{v}_m^0 + \boldsymbol{t}_b^{\boldsymbol{\theta}} - \boldsymbol{v}_m^{\theta} \right\|_2^2, \\
& E_{w1} = \sum_{m=1}^{M} \left| \sum_{b=1}^{B} w_{m,b} - 1 \right|^2, \\
& E_{ws} = \sum_{b=1}^{B} \boldsymbol{w}_b^T \boldsymbol{L} \boldsymbol{w}_b, \\
\text{subject to} \quad & w_{m,b} \geq 0,\ \forall m, b
\end{aligned}
\tag{2.6}
$$

where $\boldsymbol{w}_b = [w_{1,b}, w_{2,b}, \ldots, w_{m,b}]^T$ is the stacked weight vector for the b-th bone, $\lambda_s = 0.001$ is a spatial smooth factor, and $\boldsymbol{L}$ is the $M \times M$ spatial mesh Laplacian matrix which can be pre-computed on the template mesh using the method [61].

Although the above optimization problem is a linear non-negative least square (NNLS) problem, due to the large matrix size $(3M\Theta + M + MB) \times MB$, it is hard for a general NNLS algorithm to deploy. Specially, the active-set algorithm requires a huge mount of memory and takes nearly impossible long time to run, while the interior-point algorithm is hard to converge to a globally reasonable solution. To

solve this least square problem, we utilize a strategy recommended in [59]:

After taking the derivatives of E_w with respect to unknowns $\boldsymbol{w} = [\boldsymbol{w}_1, \boldsymbol{w}_2 \ldots, \boldsymbol{w}_b]^T$, we can get a constrained linear system:

$$\text{solve} \quad \boldsymbol{A}\boldsymbol{w} = \boldsymbol{b} \quad \text{subject to} \quad \boldsymbol{w} \geq 0 \tag{2.7}$$

In the iterative approach, $\boldsymbol{w}$ is first solved from the unconstrained linear system and then the lower bound is found by $\boldsymbol{\delta} = \min(\boldsymbol{w}, 0)$, and then solve the system:

$$\text{solve} \quad \boldsymbol{A}\boldsymbol{w} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{\delta} \quad \text{subject to} \quad \boldsymbol{w} \geq 0 \tag{2.8}$$

The process repeats until $\|\boldsymbol{\delta}\|$ is small enough. In our implementation, it converges fast within less than 3 iterations and in the last step, $\boldsymbol{w}$ is normalized to row sum to 1.

If skinning weights $\boldsymbol{w}$ are fixed and Θ pose samples are given, we can estimate each rigid transformation $[\boldsymbol{R}_b^{\boldsymbol{\theta}}\ \boldsymbol{t}_b^{\boldsymbol{\theta}}]$ from the following least square:

$$\min_{\boldsymbol{R}_b, \boldsymbol{t}_b} \sum_{m=1}^{M} w_{m,b} \|\boldsymbol{R}_b^{\theta} \boldsymbol{v}_m^0 + \boldsymbol{t}_b^{\theta} - \boldsymbol{v}_m^{\boldsymbol{\theta}}\|^2. \tag{2.9}$$

Note that this problem is an approximation of the term E_{wd} in 2.6 when $w_{m,b} \approx 1$, therefore we truncate the equation to only involve rows that $w_{m,b} > 0.8$.

As an optional output, when $[\boldsymbol{R}_b^{\boldsymbol{\theta}}\ \boldsymbol{t}_b^{\boldsymbol{\theta}}]$ are fixed, it is able to compute joint positions in the kinematic skeletal tree by minimizing the following function:

$$E_J(\boldsymbol{c}) = \sum_{b_1, b_2} \sum_{\theta} \left\| (\boldsymbol{R}_{b_1}^{\boldsymbol{\theta}} - \boldsymbol{R}_{b_2}^{\boldsymbol{\theta}}) \boldsymbol{c}_{b_1,b_2} + (\boldsymbol{t}_{b_1}^{\boldsymbol{\theta}} - \boldsymbol{t}_{b_2}^{\boldsymbol{\theta}}) \right\|_2^2, \tag{2.10}$$

where b_1-th bone is the parent of b_2-th bone in the kinematic tree and $\boldsymbol{c}_{b_1,b_2}$ denotes the position of the joint connecting them.

To summarize, the skinning adaption stage is an iterative training process taking

the given shape parameter samples as input and generating skinning weights $\boldsymbol{m}$, pose basis $\boldsymbol{Q}_f^0, \boldsymbol{Q}_f^c$ and also optimized joint positions. The whole pipeline is summarized in Algorithm 2.

Algorithm 2 The Skinning Weight Adaptation Algorithm

Input: shape parameter $\boldsymbol{\beta}$, BlendSCAPE
Output: optimized $\boldsymbol{w}$, $\boldsymbol{Q}_f^0$, $\boldsymbol{Q}_f^c$

Synthesize Θ pose samples using initial BlendSCAPE by (2.2)
while converged $\neq$ true **do**
  Solve rigid transformation $[\boldsymbol{R}_b^{\boldsymbol{\theta}}\ \boldsymbol{t}_b^{\boldsymbol{\theta}}]$ for each bone by (2.9)
  Solve new skinning weights $\boldsymbol{w}_{\text{new}}$ by (2.6)
  Compute joints by (2.10)
  if $\|\boldsymbol{w}_{\text{new}} - \boldsymbol{w}\|_F < \epsilon$ **then**
    converged = true
  else
    $\boldsymbol{w} \leftarrow \boldsymbol{w}_{\text{new}}$
  end if
end while
Compute $\boldsymbol{B}_f(\boldsymbol{\theta}) = \sum_b w_{f,b}\boldsymbol{R}_b^{\boldsymbol{\theta}},\ \forall f, \boldsymbol{\theta}$ using the optimized $\boldsymbol{w}$ and $\boldsymbol{R}_b^{\boldsymbol{\theta}}$
Solve $\boldsymbol{Q}_f$ by (2.3)
Train $\boldsymbol{Q}_f^0, \boldsymbol{Q}_f^c$ by (2.4).

Evaluation and Deform Results We first evaluate our skinning weight adaptation algorithm in Algorithm 2 by computing the data fitting error with the 70 training pose data. Taking the ground truth vertex correspondence, the deforming error is computed by comparing the linear deformed vertex (2.1) using the skinning weight of each iteration. The average deforming error of all 70 samples is plotted in Figure 2.3.

As the adaptation converges in less than 6 iterations, we plot the weight and skeleton optimization for the first 6 iterations as shown in Figure 2.4.

For qualitative comparison with the linear blending system (LBS) [57, 62], we show a sequence driven by LBS and our BlendSCAPE model in Figure 2.5, in which we assign LBS with initial skinning weight and compare with our result after weight adaptation in the row below.

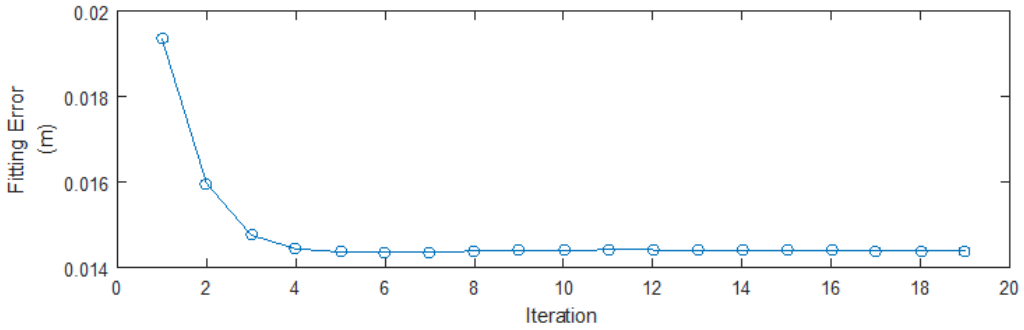


Figure 2.3: The mean error from the template to 70 training data using the ground truth vertex correspondence. The process converges almost within 6 iterations.

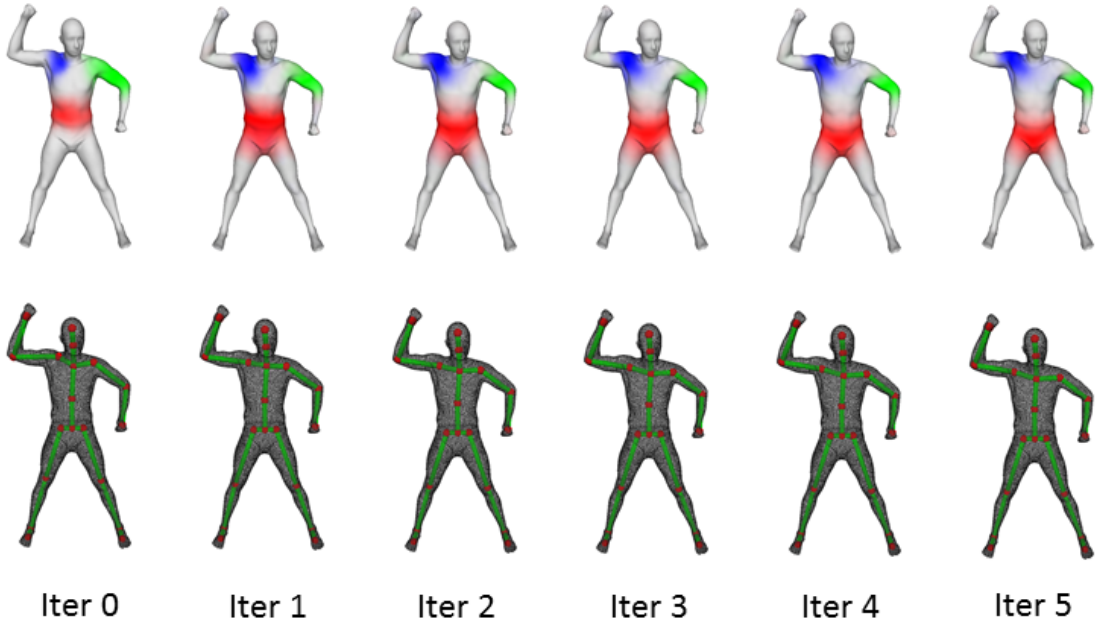


Figure 2.4: The first 6 iterations of computed skinning weight and skeleton. In the above row, only skinning weights for right upper arm, left shoulder and pelvis part are color coded.

Shape and Pose Fitting within Probabilistic Framework In section, we fit the trained BlendSCAPE model to an incomplete observation of arbitrary human body and shape. Different from SCAPE-based approaches [49, 50, 52] that assume a closed enough initial guess allowing to find the closest point correspondences and

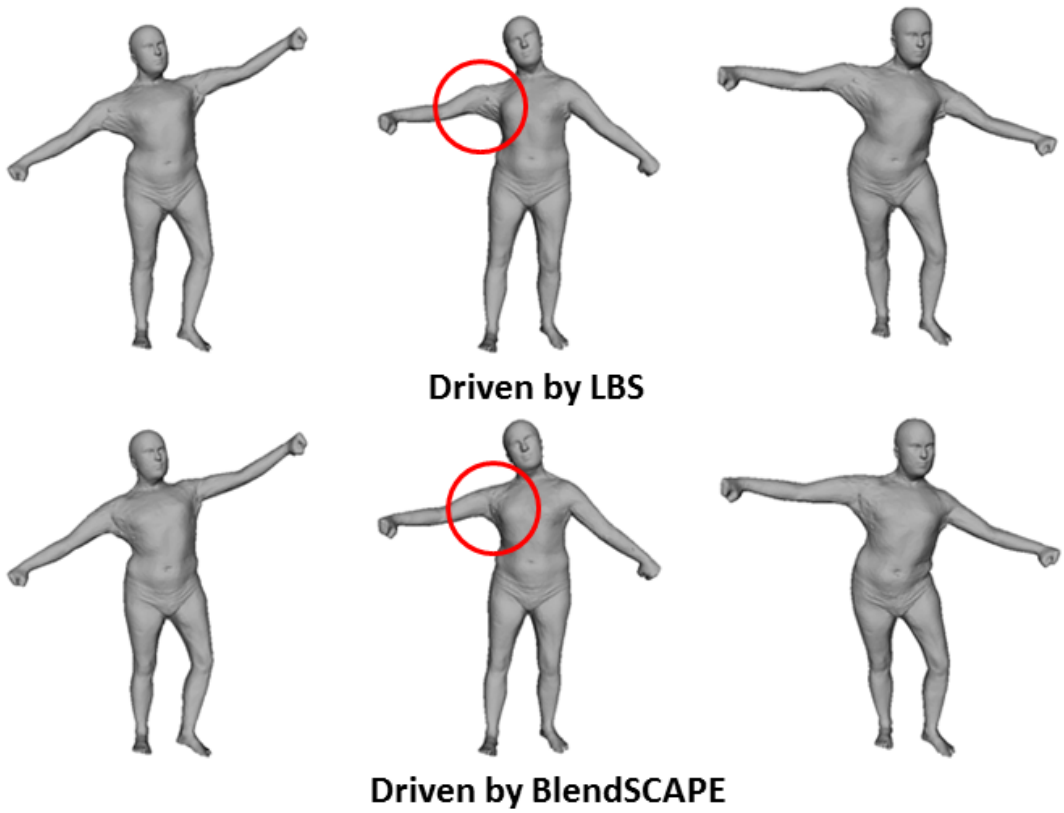


Figure 2.5: A pose driven sequence comparing the LBS system (above row) and our model (below row). The LBS system cannot accurately represent the surface around areas such as the armpit even with optimized skinning weights.

mocap approaches [33] that require sparse tracking markers, we deploy the model fitting within a Gaussian Mixture Model (GMM) framework, which takes all the data points in the observation into account, inherently robust to noise and occlusion, and also enabling fitting from a large distance. The organization of this section is first to formulate the model fitting as a Maximum Likelihood (ML) problem using the GMM assumption and then to solve it using an Expectation Maximization (EM) algorithm.

Suppose the input observation is a 3D point cloud $\mathcal{X} = \{\boldsymbol{x}_n \mid n = 1, \ldots, N\}$ and each vertex $\boldsymbol{v}_m \in \mathcal{V}$, ($\mathcal{V} = \{\boldsymbol{v}_m \mid m = 1, \ldots, M\}$) of the human body of pose $\boldsymbol{\theta}$ and shape $\boldsymbol{\beta}$ is considered as the Gaussian centroid of $\mathcal{X}$, the probability of an observed

data point $\boldsymbol{x}_n$ can be expressed as

$$p(\boldsymbol{x}_n) = w_n \frac{1}{N} + (1 - w_n) \frac{1}{M} \sum_{m=1}^{M} p(\boldsymbol{x}_n | \boldsymbol{v}_m), \tag{2.11}$$

$$p(\boldsymbol{x}_n | \boldsymbol{v}_m) = \frac{1}{(2\pi\sigma^2)^{3/2}} \exp\left(\frac{-\|\boldsymbol{x}_n - \boldsymbol{v}_m\|^2}{2\sigma^2} \right), \tag{2.12}$$

where we assume that the noise and outliers are accounted in the mixture model and have a uniform distribution $\frac{1}{N}$ and balanced by a weight $0 < w_n < 1$. And also each Gaussian has an equal isotropic covariance σ^2 and the prior probability of each vertex is $p(\boldsymbol{v}_m) = \frac{1}{M}$.

The estimation of the vertices $\mathcal{V}$ can be modeled as a Maximum Likelihood (ML) problem $\prod_{n=1}^{N} p(\boldsymbol{x}_n)$, which turns out to minimize the negative log-likelihood $E = -\sum_{n=1}^{N} \log p(\boldsymbol{x}_n)$ and usually can be iteratively solved by the Expectation Maximization (EM) algorithm.

In the expectation or E-step of the algorithm, *a posteriori* probability distribution $p^{old}(\boldsymbol{v}_m | \boldsymbol{x}_n)$ of mixture components is calculated by Bayes rule:

$$p_{mn}^{old} := p^{old}(\boldsymbol{v}_m | \boldsymbol{x}_n) = \frac{\exp\left(\frac{-\|\boldsymbol{x}_n - \boldsymbol{v}_m\|^2}{2\sigma_{old}^2} \right)}{\sum_{m=1}^{M} \exp\left(\frac{-\|\boldsymbol{x}_n - \boldsymbol{v}_m\|^2}{2\sigma_{old}^2} \right) + c}, \tag{2.13}$$

where $c = (2\pi\sigma_{old}^2)^{3/2} \frac{w_n}{1-w_n} \frac{M}{N}$ and all the variables are known.

In the maximization or M-step, the new parameters are found by minimizing an upper bound of the negative log-likelihood E as the objective function:

$$\begin{aligned} \min_{\boldsymbol{\theta}, \boldsymbol{\beta}} \quad & -\sum_{n=1}^{N} \sum_{m=1}^{M} p_{mn}^{old} \left(\log \left(\frac{1 - w_n}{M} p^{new}(\boldsymbol{x}_n | \boldsymbol{v}_m) \right) + \log \frac{w_n}{N} \right) \\ \propto \quad & Q := \sum_{n=1}^{N} \sum_{m=1}^{M} \frac{1}{2\sigma^2} \sum_{n,m} p_{mn}^{old} \|\boldsymbol{x}_n - \boldsymbol{v}_m\|^2 + \frac{3P}{2} \log \sigma^2, \\ \text{where} \quad & P = \sum_{n=1}^{N} \sum_{m=1}^{M} p_{mn}^{old} \end{aligned} \tag{2.14}$$

in which unknowns to solve are $\boldsymbol{v}_m(\boldsymbol{\theta}, \boldsymbol{\beta})$ and σ^2. The EM algorithm for body and clothes estimation is summarized in Algorithm 3.

Algorithm 3 EM algorithm for fitting body parameters

Input: initial $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, σ^2, data points $\mathcal{X}$
Output: optimized $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, σ^2

Uniformly downsample (Poisson-Disk Sampling) the point cloud to a comparable number of M
while $\boldsymbol{\theta}$, $\boldsymbol{\beta}$ not converged **do**
  E-step: Compute posteriors $\{p_{m,n}\}$ by Eq. 2.13
  M-step: Run the iterative solver by Algorithm 4 for $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, $\boldsymbol{v}$ and σ^2.
end while

Parameters Optimization and Iterative Solution The M-step to minimize the objective function 2.14 involves the vertex positions, therefore we can combine the BlendSCAPE model 2.2 together and get the following minimization problem:

$$\min_{\boldsymbol{\theta},\boldsymbol{\beta},\sigma^2} E_c + \lambda_{\text{data}} Q \tag{2.15}$$

where the weight factor λ_{data} controls how the strong data points affect the template model, we choose $\lambda_{\text{data}} = 10$ by default.

Taking the partial derivative of 2.15 with respect to σ^2 and let it be zero, we get

$$\sigma^2 = \frac{1}{3P} \sum_{n=1}^{N} \sum_{m=1}^{M} p_{mn} \|\boldsymbol{x}_n - \boldsymbol{v}_m\|^2. \tag{2.16}$$

If σ^2 is fixed, the problem 2.15 is a nonlinear optimization with respect to $\boldsymbol{\theta}$, $\boldsymbol{\beta}$. We design a linear iterative solution shown in 4 despite of a general nonrigid solver. The first step is to solve all the vertices of the BlendSCAPE template from 2.2 as a linear least square problem, next is to fix the shape parameter and solve the pose change $\Delta\boldsymbol{\theta}$, and the last step is to renew the shape parameter $\boldsymbol{\beta}$ with all the others fixed.

For the pose change $\Delta\boldsymbol{\theta}$, which are essentially small rotations and can be dispensed to each rigid part, we approximate it as the twist change to the rotation matrix such that $\boldsymbol{R}_b \leftarrow (\boldsymbol{I} + \Delta\hat{\boldsymbol{\theta}}_b)\boldsymbol{R}_b$, in which $\Delta\hat{\boldsymbol{\theta}}_b$ is the 3×3 skew-symmetric matrix or cross product matrix of the twist vector $\Delta\boldsymbol{\theta}_b$. To solve $\Delta\boldsymbol{\theta}$, resulting in the following linear minimization.

$$E_R(\Delta\boldsymbol{\theta}) = \sum_{f=1}^{F} a_f \left\| \boldsymbol{T}_f - \sum_{b=1}^{B} w_{f,b}(\boldsymbol{I} + \Delta\hat{\boldsymbol{\theta}}_b)\boldsymbol{R}_b\boldsymbol{D}_f\boldsymbol{Q}_f\boldsymbol{T}_f^0 \right\|_F^2 + w_R \sum_{b_1,b_2} \|\Delta\boldsymbol{\theta}_{b_1} - \Delta\boldsymbol{\theta}_{b_2}\|^2 , \tag{2.17}$$

where $w_{f,b}$, $\boldsymbol{D}_f$, $\boldsymbol{Q}_f$ are known and defined as in (2.3) and the last term prevents large joint rotations where b_1 and b_2 are adjacent bone indices and $w_R = 0.1$ is a trade-off parameter.

For the shape update, since $\boldsymbol{\beta}$ is linearly involved when PCA basis $\boldsymbol{U}, \boldsymbol{\mu}$ are fixed, the objective reduces to minimizing a simple quadratic function:

$$\begin{aligned} \min_{\boldsymbol{\beta}} \quad & \sum_{f=1}^{F} a_f \left\| \boldsymbol{T}_f - \boldsymbol{R}_b \left(\overline{\boldsymbol{U}\boldsymbol{\beta} + \boldsymbol{\mu}} \right) \boldsymbol{Q}_f \boldsymbol{T}_f^0 \right\|_F^2 \\ \text{subject to} \quad & -3\boldsymbol{\sigma} \leq \boldsymbol{\beta} \leq 3\boldsymbol{\sigma}, \end{aligned} \tag{2.18}$$

where $\boldsymbol{\sigma}$ is the standard deviation of $\boldsymbol{\beta}$ along each dimension computed during the training stage (2.5). The optimization iterates until converged to a local optimum of 2.15.

Evaluation and Fitting Results To evaluate the accuracy of our GMM-BlendSCAPE fitting, we qualitatively compare the fitting result from the male template to each pose in the training data set, where we set the initial pose and shape as identities and the fitting process converges in 5 iterations on average. Figure 2.6 presents the quantitative results by aligning the fitted results to each point cloud. The error is also computed by the ground truth point correspondence.

Algorithm 4 E-step Optimization For $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, σ^2

Input: initial $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, σ^2, $\{p_{m,n}\}$, data points $\mathcal{X}$
Output: optimized $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, σ^2

while $\boldsymbol{\theta}$, $\boldsymbol{\beta}$, σ^2 not converged **do**
  Solve $\mathcal{V}$ by minimizing Equation (2.15)
  Solve $\Delta\boldsymbol{\theta}$ by Equation (2.17)
  Update $\boldsymbol{R}_b \leftarrow (\boldsymbol{I} + \Delta\hat{\boldsymbol{\theta}}_b)\boldsymbol{R}_b$ and then $\boldsymbol{\theta}$
  Compute $\boldsymbol{Q}_b$ by the updated $\boldsymbol{\theta}$
  Solve $\boldsymbol{\beta}$ by Equation (2.18) (skipped if $\boldsymbol{\beta}$ is fixed for pose tracking purpose)
end while

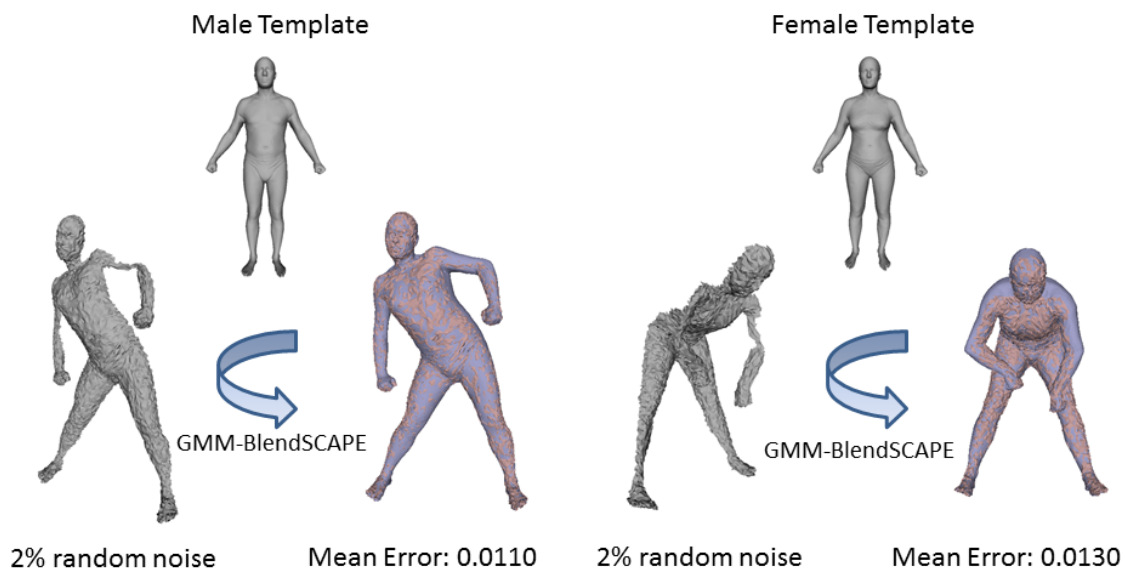


Figure 2.6: Auto registration results from the template to point cloud of sample poses. The error is computed by ground truth correspondences. The red/blue colors denote the data point cloud with random noise and the fitted model.

For incomplete data set, we qualitatively compare our fitting results with the LBS fitting algorithm [62] in Figure 2.7.

GMM-BlendSCAPE Fitting Accuracy Evaluation We evaluate the accuracy of GMM-BlendSCAPE tracking algorithm on a publicly free dataset PDT [63] which contains ground truth joint locations. We fix a known body shape and only com-

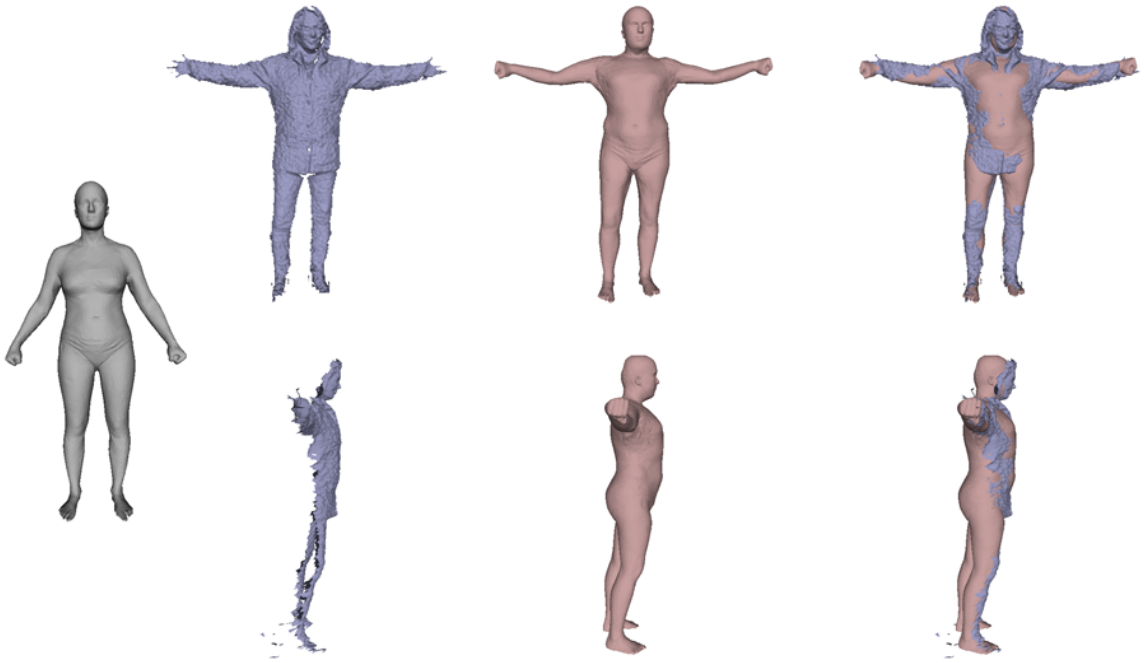


Figure 2.7: A result of fitting a female template to an incomplete point cloud sequence from Kinect sensor. Note that we estimate the shape at the beginning of the video, *e.g.*, T-pose, and then we fix shape parameters and track poses only for the rest of the video.

pare the pose tracking accuracy with the groundtruth joint locations 2.8 and several existing motion tracking approaches 2.9 such as [62–64].

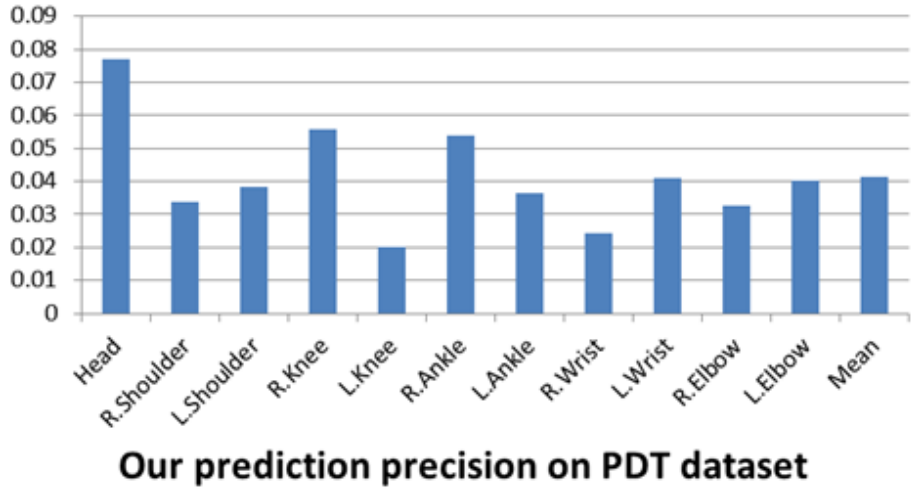


Figure 2.8: The mean joint errors of our GMMBS tracking algorithm with the provided groundtruth joint locations using PDT dataset.

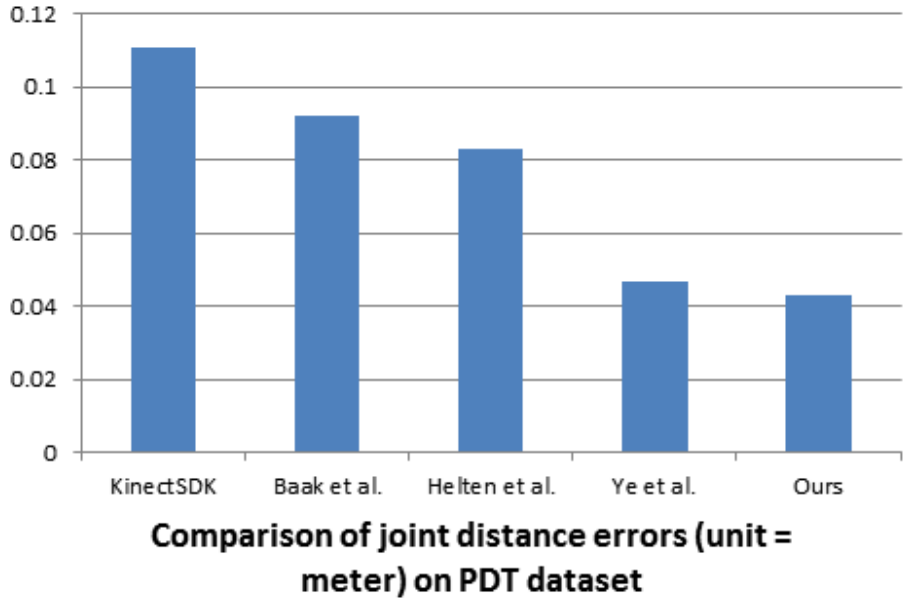


Figure 2.9: Quantitative comparison of mean errors with existing motion tracking methods [62–64] using PDT dataset.

Chapter 3 Single View 4D Self Portrait Framework

Instead of fitting a general shape template from training database, in this chapter, we presents a novel algorithm to build the complete 4D model, a personalized template, from partially scanned data. As an active research topic, a number of approaches have been developed to reconstruct complete models from depths. However, due to the relatively low-quality depths they produce, multiple overlapping depth maps have to be fused together to not only provide more coverage, but also reduce the noise and outliers in the raw depth maps. Therefore these modeling approaches are limited to static objects (e.g., the well-received KinectFusion system [65]), or human in mostly static poses (e.g, the home body scanning system [66] and the 3D self-portrait system [1]). Our main idea is to first create a *drivable and detailed* human model, and then use the *personalized* model to synthesize a full 3D model that best fit the raw input depth map containing dynamic human motion.

The entire modeling pipeline can be separated into three steps. In the first step, an image+depth sequence is captured using a depth camera (*e.g.*, Kinect sensor). Each capture provides a partial surface and a texture of the subject person at each time instant. The system allows to capture a desired local part and update the details to the final complete model. To allow robust body parts registration, the image sequence is used to locate temporal feature correspondences, which help track and warp each articular or rigid part. In the second step, a parametric body template is associated with the pre-defined key poses and shape detail parameters are automatic customized to the subject person. In the third step, the template model is refined and registered with respect to each partial scan and achieves a consistent and realistic complete model animation.

While capturing human bodies has been widely studied using either an array

of surrounding cameras, (*e.g.*, [8, 9]), or a full body scanner (*e.g.*, Cyberware body scanner), we think the setup is expensive and cumbersome while obtained surface data is incomplete due to occlusions. Encouraged by the recent development of handy range sensors, we expect that color+depth videos will be easily captured and widely used in our daily life. Therefore, we start with a single depth camera which is much affordable and practical to carry around for outdoor capturing activities, however, less visible part of the object can be observed at each time instant. The desire of our proposed method is to complete the partial data into a fully animated 3D human body model.

In the simplest case, if the object is rigid or less deformed, this completion task becomes the well-studied Structure-from-Motion (*e.g.*, [67, 68]) problem using 2D image sequence or the Iterative Closest Point (ICP [69]) problem using 3D point cloud. Although non-rigid registration techniques [53,70] have been presented for registering and recovering human body under small deformations, modeling a complete model of a particular person is still a challenging problem.

Our system first capture the human subject under different poses. The subject needs to stand still for a few seconds per pose while a single depth sensor that is mounted on a motorized tilt-unit scans the subject to obtain a relatively high-quality partial 3D model. Unlike previous methods, the subject does not need to rotate around and be scanned in the same pose from multiple angles. From the collection of partial scans of different poses (some from the front, some from the back, and some from the side), a *complete* 3D model is reconstructed using non-rigid point registration and merging algorithms. The model is not only personalized to the subject, but also *rigged* to support animation. Now our system is ready to synthesize high-quality dynamic models using the low-quality depth input directly from the sensors. Note that we are not simply driving the personalized model using standard skeleton-based animation techniques. In each frame, the personalized model is updated to produce a

best fit to the input for the visible part. Figure 1.1 shows a complete example of our system. It should be noted that we achieve all of these using no more than a single depth sensor.

To the best of our knowledge, our system is the first that can automatically reconstruct a human model that is not only detailed but also drivable while using only a single commodity depth camera. Our method does not rely on any training database, requires very little user cooperation (each pose is scanned only once), and can create high-quality dynamic models of human motions. Therefore we believe our system can be used to expand the applications of depth sensors to the dynamic human modeling area.

3.1 Previous Work

We review the related recent works in 3D human model reconstruction, mesh deformation and registration.

The model completion task is closely related to two techniques: the deformation models [71–73] and the performance capture techniques [26, 27, 74]. Pekelny's method [74] aims to build a complete model over time with a single depth camera by assuming the deformation as articulated and piecewise rigid, and merging partial rigid surfaces over time using the Iterative Closest Point (ICP) method. Li's method [26] emphasizes on how to robustly register the pre-defined template model to non-rigid partial scans frame by frame via non-linear optimization and also uses the temporal coherence to fill holes in almost complete input mesh sequence [27].

Structure from Motion (SFM) techniques (*e.g.*, [5, 10]), which was originally limited to static scenes, have been extended to reconstruct dynamic non-rigid scenes by making extra assumptions about shape deformation. The motion of a non-rigid time-varying object can be decomposed into a rigid transformation and non-rigid deformation. Represented by a set of sparse feature points and their motions, shape

deformation has been successfully reconstructed using different models, including a combination of several basic shapes [75, 76], Gaussian distributions [14], or based on Probabilistic Principal Components Analysis [77]. Multiple view methods are widely deployed to capture scenes (*e.g.*, [8, 9]). Surface reconstruction can then be done using either Multi-view stereo algorithms [78, 79], or Shape from Silhouette techniques [80–82].

Hole filling is also known as a common problem in the geometric modeling community. Many methods have been developed to address this issue (*e.g.*, [27, 83–86]). Typically, they are focusing on high-quality static models that are acquired using laser range scanner with relatively small missing parts. The problem we are trying to solve here is significantly more challenging. We allow 3D models acquired by a single depth camera (Time-of-Flight or Kinect depth sensor) as our input, since a laser range scanner can hardly capture dynamic scenes. Compared with range scanners, depth cameras contain more noise, and the input scan is *less* than 50% complete (one depth map for each instant).

The mesh embedded deformation [72] uses a rough guided graph to deform the mesh as rigid as possible. Based on the embedded model, the approach of Li *et al.* [87] uses a pre-scanned object as shape prior and register. Despite of the nonlinear embedded approach, linear mesh deformation methods such as [88, 89] are more likely to deal with small deformation and details transfer.

For handling the loop closure problem, the real time method [66] diffuses the registration error and online updates the model. This method aims to align scans of static objects. The global registration for articulated models [90] can cope with large input deformation, but is less suitable for aligning human body and garment.

The full body multiple Kinect scanning system [70] captures a dense sequence of partial meshes while the subject standing still on a turntable. All the partial scans are registered together based on the error distribution approach [91]. 3D Self-

Protraits [1] presents the first autonomous capture system for self-portraits modeling using a single Kinect. The user stands as still as possible during capture and turn roughly 45 degrees at each scan.

For registering dynamic input scans without large rotation change, the global linear approarch [61] registers all the scans using the linear deformation model which assumes small rotation angle of input scans.

3.2 Pairwise Nonrigid Registration Framework

In this section, we build complete 3D models for all the captured poses using partial scans. First, we introduce our data capture setup and the initial alignment using a general template model. Then we formulate the nonrigid registration problem using the embedded model of a simple yet efficient loop constraints.

System Initial Setup We utilize the Kinect Fusion Explorer [65] tool in Microsoft Kinect SDK to capture partial 3D meshes and colors. The subject person stands in front of the sensor approximately one meter away. The Kinect sensor is tilt from 13 degree to -27 degree during each capture. It takes four seconds per scan and the subject person keeps almost still at each pose. In order to build complete models, we take multiple scans at different angles to ensure most of body can be seen at least once.

Input meshes of Kinect Fusion are extracted from a volume of size 512^3 and 768 voxels per meter. We uniformly sample the input mesh to an average edge length of $4mm$ and erode from its boundary by $2cm$ to cut off sensor outliers. The floor is removed using background subtraction.

Since there is neither a semantic information from the scanned meshes nor natural correspondences, we adopt our GMM-BlendSCAPE fitting algorithm in the previous chapter to align a generic template onto each of the scanned input point cloud. In

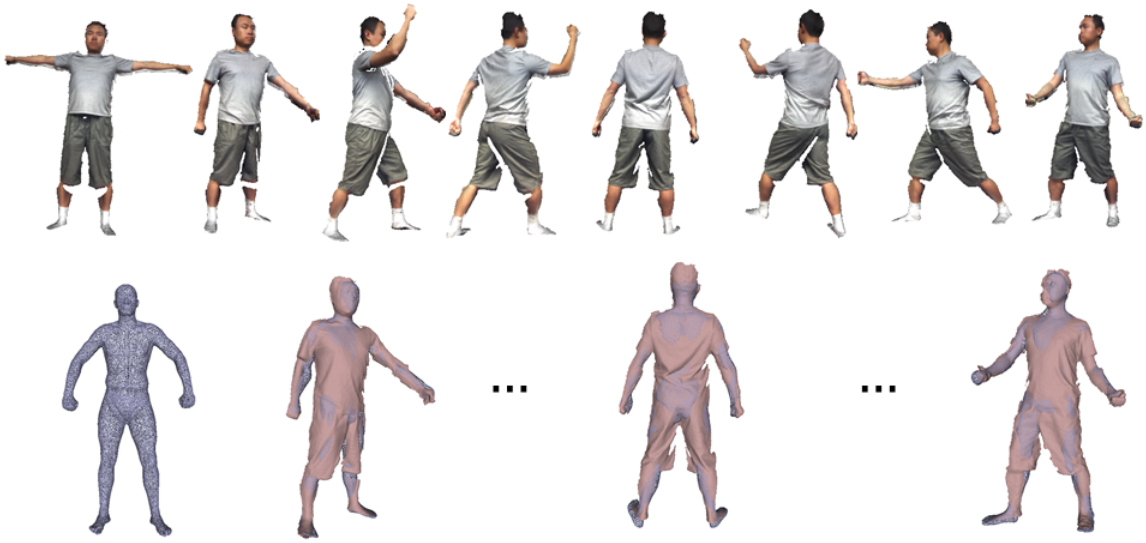


Figure 3.1: The initial alignment of partial point cloud from eight views using our GMM-BlendSCAPE template fitting.

despite of large pose difference, our fitting process generally provides sufficient good initial fitting results as shown in Figure 3.1.

Pairwise Nonrigid ICP For pairwise registration of partial scans, we employ the embedded deformation model [26, 72], which describes plastic deformation and is effective to handle articular human motion [26]. The embedded method defines the deformation of each vertex $\boldsymbol{v}$ on the mesh influenced by K nearest nodes $\boldsymbol{g}$ on a coarse guide graph. In our case, two meshes $\mathcal{M}_i, \mathcal{M}_j$ have already aligned with their graphs $\mathcal{G}_i, \mathcal{G}_j$ after our template fitting step, and also $\mathcal{G}_i, \mathcal{G}_j$ have the same face connectivity. The transformation from $\mathcal{G}_i$ to $\mathcal{G}_j$ is defined on each node $\boldsymbol{g}^m$: a 3×3 local rotation matrix $\boldsymbol{R}_i^m$ and a translation vector $\boldsymbol{t}_i^m$. Given transformations, the node on deformed graph $\tilde{\mathcal{G}}_i$ is simply added the translation: $\tilde{\boldsymbol{g}}^m = \boldsymbol{g}^m + \boldsymbol{t}_i^m$ on the graph and the deformed vertex is computed as $\tilde{\boldsymbol{v}} = \sum_{k=1}^{m} w_k(\boldsymbol{v}_i) \left[\boldsymbol{R}_k(\boldsymbol{v}_i - \boldsymbol{g}_k) + \boldsymbol{g}_k + \boldsymbol{t}_k\right]$ where $w_k(\boldsymbol{v}_i)$ is the influence weight inversely proportional to the distance from $\boldsymbol{v}_i$ to its control nodes $\|\boldsymbol{v}_i - \boldsymbol{g}_k\|$.

It can be easily verified that if $(\boldsymbol{R}_1^m, \boldsymbol{t}_1^m)$, $(\boldsymbol{R}_2^m, \boldsymbol{t}_2^m)$ are two consecutive deformations of $\mathcal{G}_i$, the total deformation is $(\boldsymbol{R}_2^m \boldsymbol{R}_1^m, \boldsymbol{t}_2^m \boldsymbol{t}_1^m)$. Let $\boldsymbol{R}_2^m = (\boldsymbol{R}_1^m)^{-1}$ and $\boldsymbol{t}_2^m = -\boldsymbol{t}_1^m$, then the mesh deformed by $(\boldsymbol{R}^m, \boldsymbol{t}^m)$ can be restored using $\left((\boldsymbol{R}^m)^{-1}, -\boldsymbol{t}^m\right)$. We assume all the $\{\boldsymbol{R}^m\}$ are almost rigid and this property holds in our case.

For registering $\mathcal{M}_i$ to $\mathcal{M}_j$, transformations $(\boldsymbol{R}_i^m, \boldsymbol{t}_i^m)$ are solved by minimizing the energy function similar to [26]:

$$
\begin{aligned}
&\min\ E_{\text{fit}} + \lambda_{\text{reg}} E_{\text{reg}}, \\
\text{where}\quad & E_{\text{fit}} = \sum_c \alpha_{point} \|\boldsymbol{v}_i^c - \tilde{\boldsymbol{v}}_i^c\|^2 + \alpha_{plane} \left|\tilde{\boldsymbol{n}}_i^T (\boldsymbol{v}_i^c - \tilde{\boldsymbol{v}}_i^c)\right|^2, \\
& E_{\text{reg}} = \sum_m \sum_{l \in N(m)} \left\|\boldsymbol{R}_i^m (\boldsymbol{g}_i^l - \boldsymbol{g}_i^m) - (\boldsymbol{g}_i^l + \boldsymbol{t}_i^l - \boldsymbol{g}_i^m - \boldsymbol{t}_i^m)\right\|^2,
\end{aligned}
\tag{3.1}
$$

in which the fitting term E_{fit} constrains the deformed position of a subset of vertices and the regularization term E_{reg} ensures the smoothness of the deformation and is balanced by a weight λ_{reg}. Specially, $\tilde{\boldsymbol{v}}_i^c$ specifies the destination of $\boldsymbol{v}_i^c$ and $\tilde{\boldsymbol{n}}_i$ is the normal on the surface of $\mathcal{M}_j$ accordingly.

To search for the correspondence, we can benefit from the same embedded graph, that is associated graphs $\mathcal{G}_i$ and $\mathcal{G}_j$ have the same face connectivity, and then we are able to segment each mesh by corresponding graph nodes as shown in Figure 3.2. The same colored region denotes vertices influenced by same graph nodes. When searching correspondences from $\mathcal{M}_i$ to $\mathcal{M}_j$, we perform iterative closest point (ICP) algorithm to align large patches (area > threshold) and search for the closest point after ICP. Faraway or normal inconsistent pairs are excluded. We obtain in roughly 2000 correspondences for a pair of scans.

The cost function equation (3.1) is minimized by Gauss-Newton solver and see [26, 72] for details. After registration, we get all of the transformations $\{(\boldsymbol{R}_i^m, \boldsymbol{t}_i^m)\}$, the deformed graph, the deformed mesh and a corresponding point set. Another tradeoff is to set a relatively larger regularization weight w_{reg} and smaller fitting

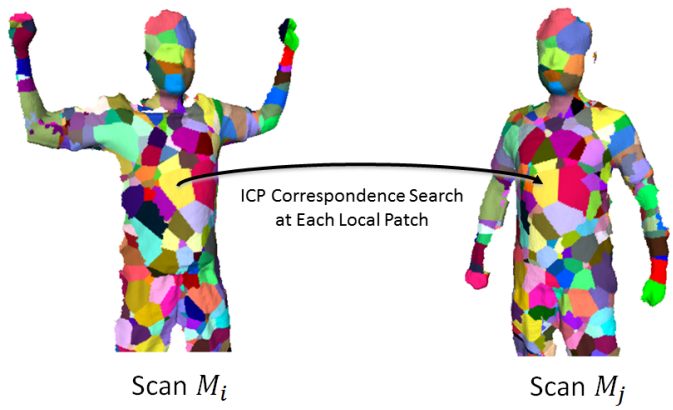


Figure 3.2: We search for corresponding points by aligning patches controlled by the same graph nodes using ICP.

weight w_{fit}. It results in slower convergence to correct destination of the overall algorithm but benefits the avoidance of severe failure deformation of the graph such as self intersection and volume collapse due to error accumulation. In our experiment, it shows that the whole algorithm still converges within 5-10 iterations as Figure 3.5.

3.3 Global Nonrigid Registration Algorithm

Now that we have n partial scans in the capture stage and they are aligned with graphs in 3.1. In this section,we register all scans to each pose while achieving global geometry consistency. Inspired by [70, 91], we develop an iterative optimization scheme to 1) pairwise register scans and 2) adjust them by distributing accumulative error using loop closure constraints. Different from the method in [70], since the deformation of a graph is simply adding the translation $\boldsymbol{t}_i^m$ to each node, $\boldsymbol{R}_i^m$ does not interfere with the graph directly. Therefore, we deal with translational and rotational error distribution separately, and translational error optimization is simpler and more efficient.

Preprocessing Given input scans and graphs, we initially register all the graphs to the target graph and deform all scans accordingly as shown in Figure 3.3. To

suppress outliers occurring near joints, we remove faces of long edge length and clean disconnected small patches from the deformed mesh. To reduce the influence of badly deformed vertices, we compute the affine transformation near each vertex and compare the deviation angle of the corresponding Laplacian coordinates. Each vertex is assigned to a confidence weight W_{lap} inversely proportional to the deviation.

After the rough registration, the covered region on the target graph of each scan is known. By aligning the torso part (chest and abdomen), we can roughly determine each virtual camera pose in the target coordinate system. Sorting angles from the target camera to each virtual camera, we finally get a circle of n scans denoted as $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_n$ and the target scan *w.l.o.g.*, is denoted as $\mathcal{M}_1$ in Figure 3.3.

Bi-directional Loop Constraints Now we have a loop of n scans $\mathcal{M}_i, i = 1, \ldots, n$, the graph $\mathcal{G}_1$ is aligned with $\mathcal{M}_1$ correctly and we use it as the embedded graph to register $\mathcal{M}_1$ to $\mathcal{M}_2$ by using the deformation described in 3.2. After the registration, $\mathcal{M}_1, \mathcal{G}_1$ are deformed as $\mathcal{M}_{1,2}, \mathcal{G}_{1,2}$ and transformations are denoted as $\{(\boldsymbol{R}_1^m, \boldsymbol{t}_1^m)\}$. By using the weight and node indices of $\mathcal{G}_2$ but the node positions of $\mathcal{G}_{1,2}$, we register $\mathcal{M}_2$ to $\mathcal{M}_3$ and get $\mathcal{M}_{2,3}, \mathcal{G}_{2,3}$. The process continues until registering T_n back to $\mathcal{G}_1$ with transformations $\{(\boldsymbol{R}_n^m, \boldsymbol{t}_n^m)\}$. We call this step as the pairwise registration in the context of this section. For a globally correct registration, we have $\mathcal{G}_{n,1} = \mathcal{G}_1$, that is for each node, $\boldsymbol{t}_1^m + \boldsymbol{t}_2^m + \cdots + \boldsymbol{t}_n^m = 0$, and the deformed mesh $\mathcal{M}_{n,1}$ is consistent with $\mathcal{M}_1$. When the deformation is highly rigid, applying the multiplication of consecutive deformations, the product of rotations along the loop will be an identity, that is $\boldsymbol{R}_n^m \boldsymbol{R}_{n-1}^m \cdots \boldsymbol{R}_1^m = \boldsymbol{I}$.

Due to error accumulation, the pairwise registration will drift and violate such constraints. Similar to [70], we distribute the accumulated rotational and translational error and choose a weight $w_i = 1/\mathrm{Dist}(\mathcal{M}_{i,i+1}, \mathcal{M}_{i+1})$ to transformations $\{(\boldsymbol{R}_i^m, \boldsymbol{t}_i^m)\}$, where $\mathrm{Dist}(\mathcal{M}_{i,i+1}, \mathcal{M}_{i+1})$ is the average fitting error of E_{fit} in 3.1, for all $i = 1, \ldots, n$.

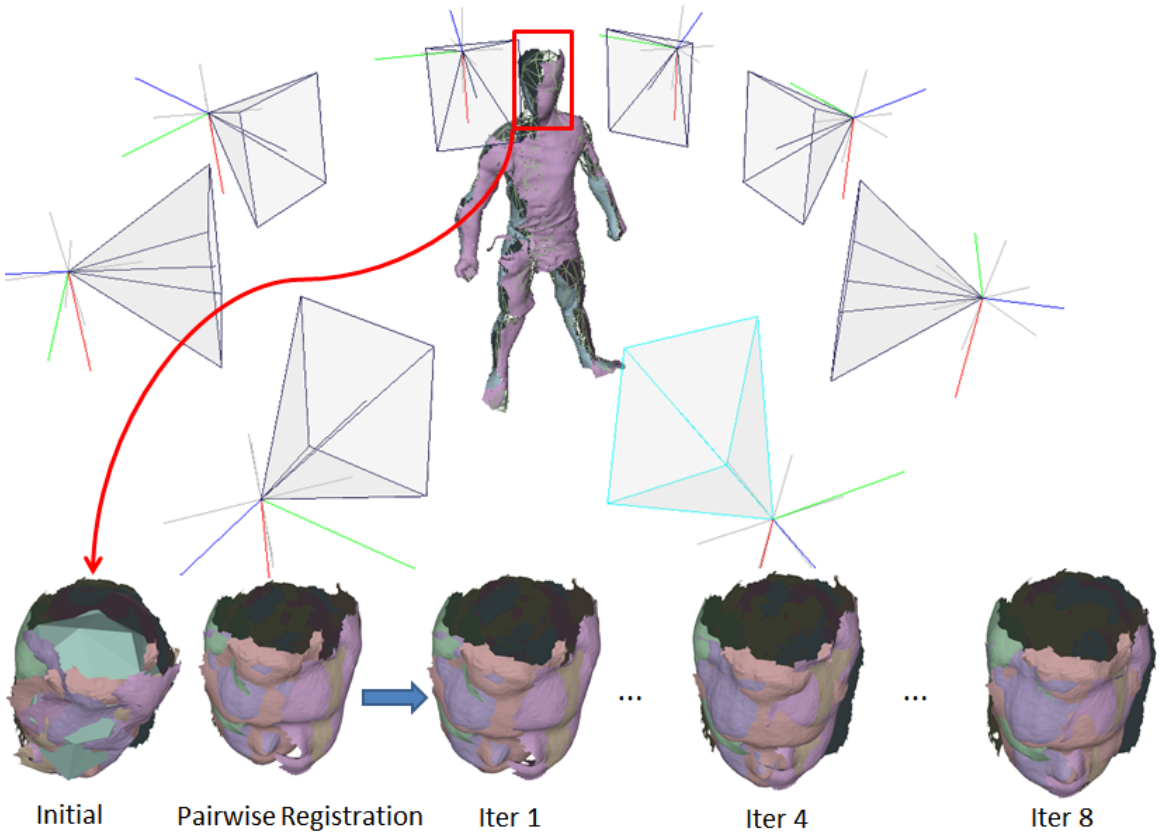


Figure 3.3: Stages in our global registration. All the partial scans are initially aligned to the target using the general template model. Virtual cameras are estimated in the coordinate system of the target pose to determine the loop closure. The fitted template model is reduced to a rough graph to guide the embedded registration. Pairwise accumulated registraton error is distrubuted after each loop adjustment.

($n+1$ we refer to 1.) Since every node will be optimized in the same way, we ignore the superscript k in the following.

The translational error is distributed by solving the following optimization,

$$\min \sum_{i=1}^{n} w_i^2 \left\| \hat{\boldsymbol{t}}_i - \boldsymbol{t}_i \right\|^2, \quad s.t., \sum_{i=1}^{n} \boldsymbol{t}_i = 0, \tag{3.2}$$

and the solution is found using Lagrange multipliers, $\hat{\boldsymbol{t}}_i = \boldsymbol{t}_i - \alpha_i \sum_{j=1}^{n} \boldsymbol{t}_j$, with the scalar

α_i as

$$\alpha_i = \frac{1}{w_i^2} \Big/ \sum_{j=1}^{n} \frac{1}{w_j^2} \tag{3.3}$$

The rotational error distribution is to minimize the total rotational deviation:

$$\min \sum_{i=1}^{n} w_i \angle(\hat{\boldsymbol{R}}_i, \boldsymbol{R}_i), \quad s.t., \boldsymbol{R}_n^m \boldsymbol{R}_{n-1}^m \cdots \boldsymbol{R}_1^m = \boldsymbol{I}, \tag{3.4}$$

where the angle between two rotations is defined as $\angle(\boldsymbol{A}, \boldsymbol{B}) = \cos^{-1}\left(\frac{tr(\boldsymbol{A}^{-1}\boldsymbol{B})-1}{2}\right)$. Analyzed in [91], the optimal $\hat{\boldsymbol{R}}_i$ is computed as

$$\begin{aligned} \hat{\boldsymbol{R}}_i &= \boldsymbol{E}_i^{<\alpha_i>} \boldsymbol{R}_i, \\ \boldsymbol{E}_i &= (\boldsymbol{R}_k \boldsymbol{R}_{k-1} \cdots \boldsymbol{R}_1 \boldsymbol{R}_n \boldsymbol{R}_{n-1} \cdots \boldsymbol{R}_{k+1})^{-1}, \end{aligned} \tag{3.5}$$

where α_i is referred to equation (4.7), and $\boldsymbol{E}_i^{<\alpha_i>}$ is defined to be the rotation matrix that shares the same axis of rotation as $\boldsymbol{E}_i$ but the angle of rotation has been scaled by α_i.

Once all the optimal $\left\{\left(\hat{\boldsymbol{R}}_i^m, \hat{\boldsymbol{t}}_i^m\right)\right\}$ are obtained, we use the total transformation $\left\{\left(\left(\hat{\boldsymbol{R}}_1^m \hat{\boldsymbol{R}}_{i-1}^m \hat{\boldsymbol{R}}_i^m\right)^{-1}, \ldots, -\hat{\boldsymbol{t}}_i^m - \hat{\boldsymbol{t}}_{i-1}^m - \cdots - \hat{\boldsymbol{t}}_1^m\right)\right\}$ to deform the mesh $\mathcal{M}_i$ with $\mathcal{G}_{i-1,i}$ back to $\mathcal{M}_1$.

This can be easily verified by that $\mathcal{M}_1$ can be deformed using the total transformation $\left\{\left(\hat{\boldsymbol{R}}_1^m \cdots \hat{\boldsymbol{R}}_{i-1}^m \hat{\boldsymbol{R}}_i^m, \hat{\boldsymbol{t}}_i^m + \hat{\boldsymbol{t}}_{i-1}^m + \cdots + \hat{\boldsymbol{t}}_1^m\right)\right\}$ to register with $\mathcal{M}_i$ and then we can apply the multiplication property to deform $\mathcal{M}_i$ back to $\mathcal{M}_1$. After all the meshes $\mathcal{M}_i$ updated, we repeat the pairwise registration step from $\mathcal{M}_1$ and $\mathcal{G}_1$. The graphs $\mathcal{G}_1, \mathcal{G}_{1,2}, \ldots, \mathcal{G}_{n,1}$ will finally converge to a constant graph and $\left\{\left(\hat{\boldsymbol{R}}_i^m, \hat{\boldsymbol{t}}_i^m\right)\right\}$ converges to the globally optimal solution as plotted in Figure 3.5.

In the sense that the error distribution step can prevent graph drifting and pull it towards the optimal position, we can perform an interleaved bi-directional way to avoid large accumulative errors. The basic idea is to perform an inverted itera-

tion using the order of $\mathcal{M}_1, \mathcal{M}_n, \mathcal{M}_{n-1}, \ldots, \mathcal{M}_3, \mathcal{M}_2, \mathcal{M}_1$ after a forward directional iteration. The directional scheme is in essential the same to the multiple cycle blending technique described in [91] and the total time complexity to convergence is the same because they traverse in both direction in one iteration and we perform in each direction once but need two iterations.

Postprocessing Once all the partial scans are registered to the target pose, the final water-tight surface is extracted by using Screened Poisson Surface method [92] which takes the point confidence into account. We assign a blending confidence for each point $W = W_{normal} * W_{sensor} * W_{lap}$: W_{normal} is inversely proportional to the angle between the original input normal and the z-axis; W_{sensor} is proportional to the distance from a point to the mesh boundary; W_{lap} is inversely proportional to the deviation Laplacian coordinates, and the final weight W is pruned to $[\epsilon, 1]$, $\epsilon > 0$. The surface color is transferred from the input color and diffused using Poisson blending method [93] to achieve seamless.

3.4 Training of Parametric Kinematics

In this section, we align all the complete 3D models built in the above section to train an animatable parametric model and fit it to the new incoming depth sequence. Different from the generic template based methods [49, 52, 53] that varies at the ability of representing level of details. Our complete models are inherently specified to a certain user and have no shape variations, therefore we only need to train the pose variation for a personalized model.

To train the parametric model similar to our generic BlendSCAPE model, all of the 3D models are required to be mesh topology consistent (*i.e.*, one-to-one vertex correspondence). We pick a neutral pose as the reference pose and deform it to all the other 3D models. Similar to the nonrigid ICP registration in section 3.2, we register

the reference model to each complete model by taking the alignment of their associated graphs as the initial guess. As a result of nonrigid registration, corresponding points are found with normal consistency. We employ the detail synthesize method to make subtle adjustment of the warped reference model:

$$\min_{d_i} \sum_{\boldsymbol{v}_i} \|\boldsymbol{v}_i + d_i \boldsymbol{n}_i - \boldsymbol{v}_i^c\|^2 + \beta \sum_{i,k} |d_i - d_k|^2 , \tag{3.6}$$

in which $\boldsymbol{v}_i$ and $\boldsymbol{v}_i^c$ are corresponding points, d_i is the distance along its normal direction $\boldsymbol{n}_i$. The distance field is diffused among neighboring vertices i and k. $\beta = 0.5$ in the experiments.

After registered to all the other $n-1$ example poses, the model at the reference pose is taken as the template for training. Since the embedded graph is fit by GMM-BlendSCAPE template, on which each vertex is associated with a skinning weight, we first transfer the skinning weight to the reference model. And then all the rest of training stages are the same as training the GMM-BlendSCAPE by fixing shape parameters, *i.e.*, the same steps in Algorithm 1 by always setting the shape related matrix $\boldsymbol{D}_f$ be identity. When fitting the trained personalized model to incoming new point cloud, same EM iterations in Algorithm 3 are performed to estimate the pose of the personalized model.

Note that in our case, we have less (usually $n = 8$) poses than the SCAPE training data. However, since the regression is linear, the ability of its representation depends on the range of joint angles instead of number of samples. In our capture stage, the subject person is required to perform different joint configurations as much as possible. And then the trained model ends up being able to recover the personalized style of movement.

3.5 Results and Evaluation

We validated our system by scanning the mannequin for performance evaluation and accuracy comparison. We scanned male and female subjects at several challenging poses to build 3D model training samples. We captured several video sequences to validate the fitting using our trained model.

Mannequin Validation As an accuracy test of our system pipeline, we acquired a 3D model of an articular mannequin and compared our results to a model captured using a high-performance structured light scanner with a 0.5mm spatial resolution.

In this test, we manually turned the mannequin around by approximately 45 degrees at each time. The mannequin was not totally rigid, and its arms and legs were slightly moved when turned around. In this case, we directly perform the pairwise registration step with loop closure adjustment. We compare it with the groundtruth to achieve an average alignment error of 2.45mm. We also compare the result with the previous paper [1] and the comparable result is shown in Figure 3.4.

In another mannequin set, we test the performance of our system by capturing large pose changes. The mannequin's arms and legs were articulately moved to several poses. The qualitative evaluation results are shown in Figure 3.6. In Figure 3.5, we show the algorithm performance to register all scans to the target pose 3.3. According to the results, the optimization procedure converges in $5 - 10$ iterations for both rotational and transnational error distributions. The final average variation in rotation is less than 0.5 degree and the variation in translation is less than $0.1mm$, which we set as a terminating condition for real person modeling.

Real Person Examples We validate our system to reconstruct both female and male persons in regular clothes. It takes several minutes to capture static scans and then watertight example poses are reconstructed as shown in Figure 3.7. We pick the

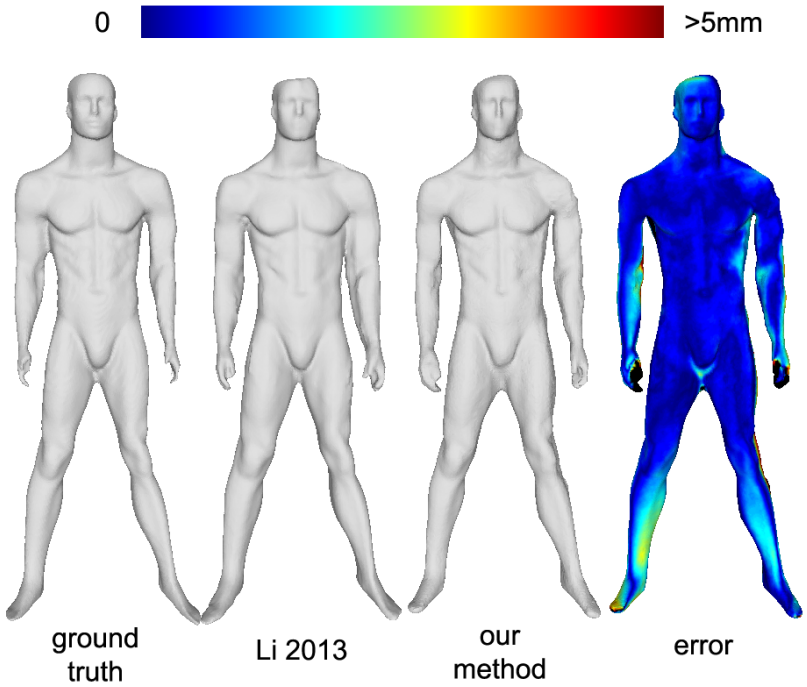


Figure 3.4: The reconstructed mannequin of an almost static pose. Error map compared to the groundtruth is plotted.

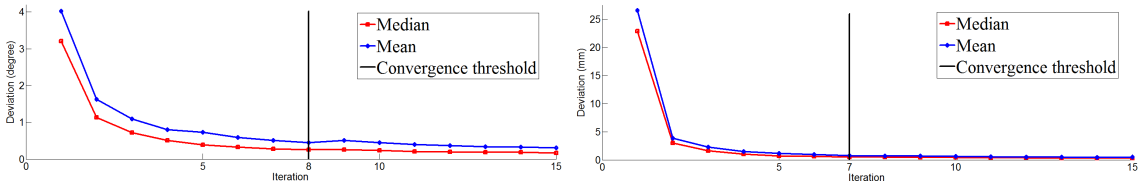


Figure 3.5: The deviation of mannequin data. The left is the rotation angle changes in degrees and the right is the translation in milimeters.

neutral pose as the reference and train parametric model. The final avatar is at the resolution of 100k faces.

Driving and Fitting to Video Sequence After training the parametric model, we test our drivable avatar using the full body video sequence at a distance about $2m$ to the Kinect sensor. Our parametric model is initially driven to the pose estimated by skeleton and then iteratively fitted to the input point cloud. Figure 3.8 shows

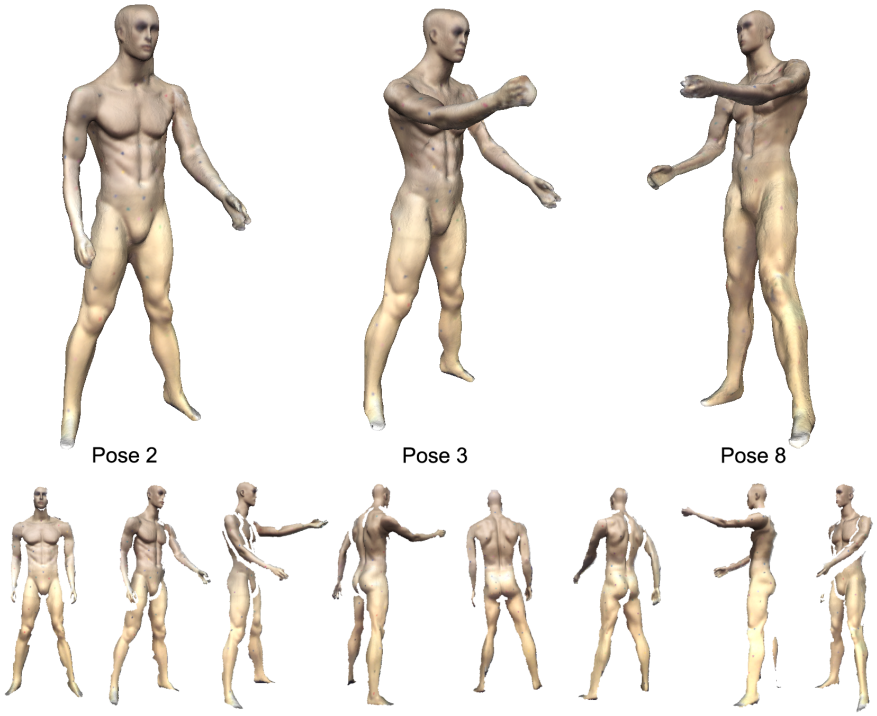


Figure 3.6: The reconstructed mannequin of some articulated arm movement.

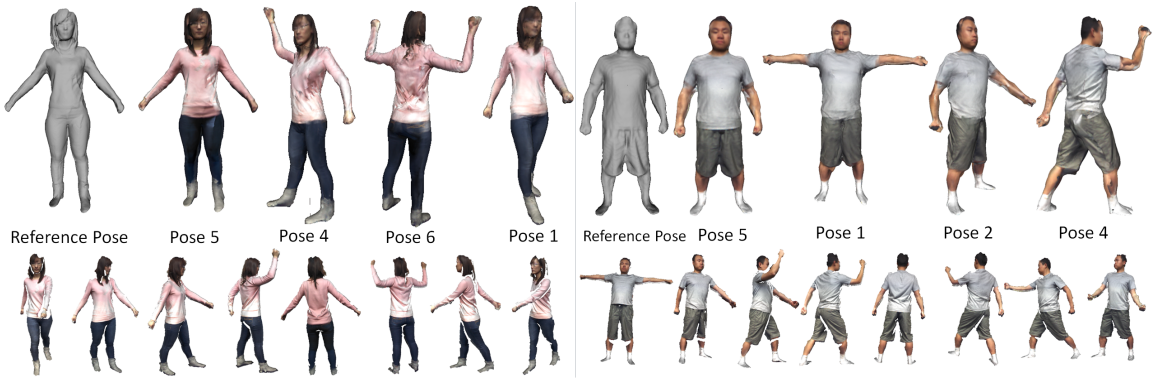


Figure 3.7: The reconstructed watertight models after our global registration. The bottom row shows the input partial scans and the upper row shows the reconstructed models at each pose.

several frames of our final fitting result.

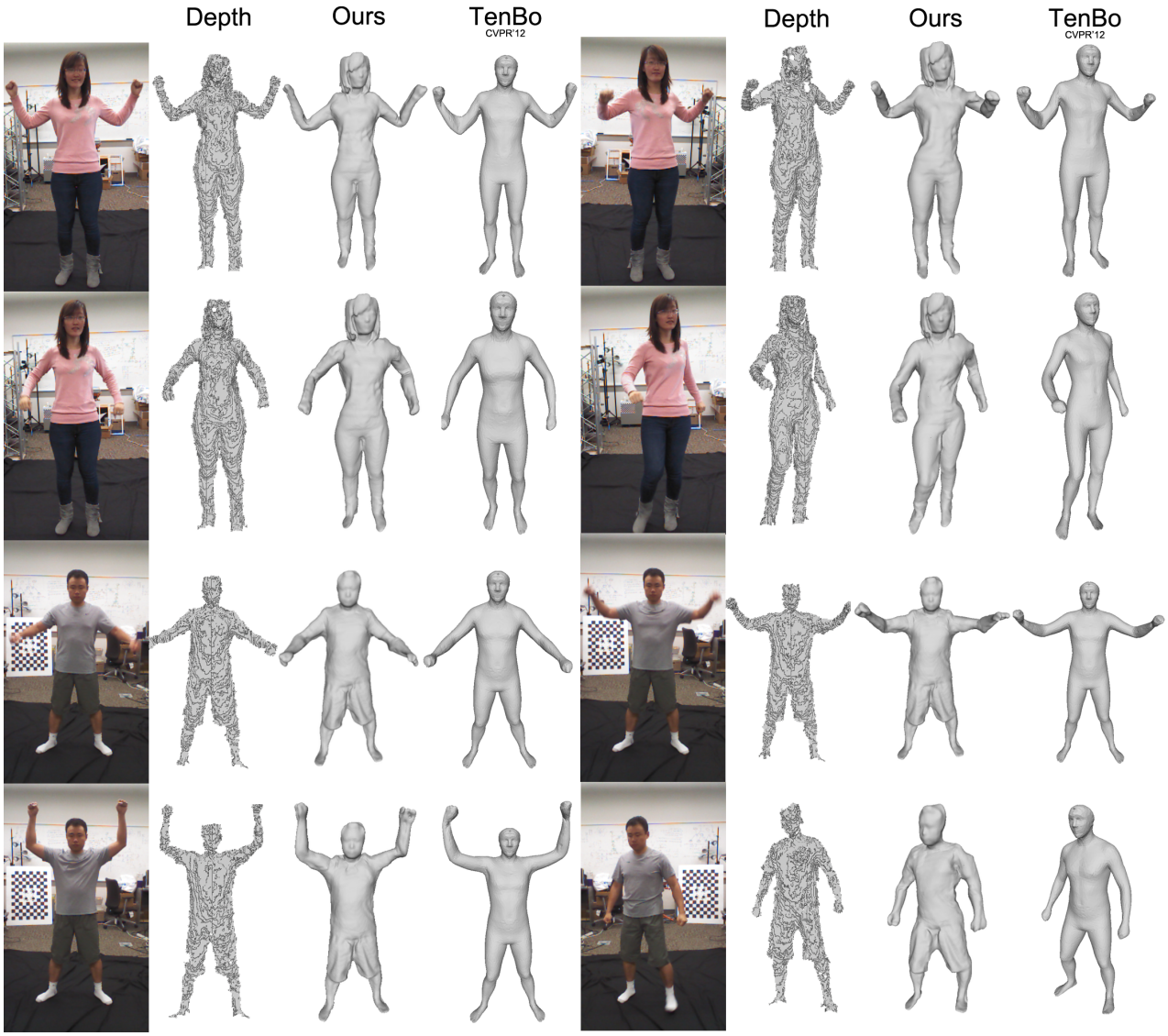


Figure 3.8: The final fitting result with our personalized parametric avatar. We compare our avatar with the general SCAPE model to show more realistic details.

Chapter 4 Real Time General Mesh Embedded Deformation

As analyzed in the previous chapter, the globally nonrigid registration is time consuming mostly due to the involved embedded deformation, which is called in every pairwise registration, and in general, the nonlinear problem is solved in a Gauss-Newton iterative manner that prevents the overall solving speed. Motivated by accelerating the computation of registration framework, we investigate the performance of embedded deformation and provide an efficient and fast solution for generic object deformation in this chapter.

For a general purpose deformation system, a challenge task is to quickly generate convincing results that preserve geometry details and also easy to manipulate for non-expert users. Most of existing real-time deformation approaches relay on skeleton or cage like handles and require sophisticated artistic skills to paint weights. Other mesh deformation techniques that do not depend on such pre-defined handles have less abilities to preserve details and a large computation complexity for high resolution meshes. The embedded deformation and its related extensions [26, 72] offer a amount of powerful and convenient tools to design 3D shapes. The main advantage of embedded deformation is that the computation is independent of both shape's representation and geometry complexity while the manipulation is still direct on the mesh. It also preserves the shape details as-much-as-possible after deforming, which means the local features do not stretch or shear. Due to the flexibility, the embedded deformation is used in non-rigid mesh registration [1], reconstructing complete models from multiple partial meshes. The common issue that arise in current embedded-based approaches is its lack of efficient linear solutions preventing from further real-time interactive manipulation.

In the embedded deformation framework, a reduced model called the deformation

graph is created and the object deforms as a linear combination of transformations of the graph nodes. Manipulating the shape is to solve both deformations of the object and graph while satisfying global consistency. Despite of its simple structure, the optimization involves local affine transformations and globally ends up solving a large nonlinear system by using computational costly Gauss-Newton iterations.

Real-time performance is crucial for interactive 3D modeling. Users always want the deformation to be responsive in real-time and easy to control. It is a main reason that skeleton-based methods still dominate the practical usage, even though they require cumbersome pre-definition of control primitives in order to preserve geometry details properly.

Our goal is to provide similar plausible deformation to the embedded deformation by using efficient linear solvers. We aim for real-time interaction for manipulating high-resolution meshes but don't require any information about the object's skeleton or topology.

In our novel deformation scheme that achieves interactive rate, we divide the nonlinear problem of solving embedded deformation into linear sub-problems and combining them in an iterative way similar to the as-rigid-as-possible surface modeling [94]. Local rotations are solved parallelly by using GPU and then propagated over the embedded graph also by an efficient linear system. In order to accelerate the convergence, we develop a hierarchical structure of graphs and significantly improve the overall speed especially when manipulating high-resolution meshes.

The contributions of our acceleration algorithm are mainly twofold:

• a linear and efficient approach to achieve the comparable quality to the nonlinear embedded deformation.

• a hierarchical strategy to accelerate the rate of convergence and make high-resolution mesh editing interactivable.

4.1 Previous Work

A vast amount of literature deals with geometry deformation and mesh registration. We discuss the following three aspects most related to our approach.

Linear Blend Skinning This sort of approaches such as [95,96] dominates current practical use as the fastest method, because it defines skeletons or cages and utilizes them as handles to manipulate the shape. The object deforms as a linear blending of the predefined structures. Computing the deformed shape is so-called *skinning* [95], the step of computing skinning weights is called *binding* or *rigging*. In general, the LBS approach can deform the object fast (small *pose time*) but require a relatively large *bind time* in the preprocess step.

Cage-based methods such as [97–99] compute weights via mean-value interpolation or solving harmonic constraints and then deform the mesh by translating the cage vertices. Since the cage greatly reduces the object complexity, it is convenient and fast to deform a local region of object by operating on cages. In order to deform detailed shapes, however, refining cage positions may require tedious manual work. [96] unifies multiple types of control handles and relieves users from the burden of manually paining blending weights. Although it allows users to freely choose handles, the bind time is still in the order of seconds per handle for 3D meshes.

Extended to the traditional LBS, [100] enables the deformation to operate within the context of given examples - so-called characteristic shapes. [101] generalizes the skinning concept to multiple deformers like proxy curves and polygons. [102] defines the transformation on tetrabones and adds more physical constraints such as length and balance constraints to make the deformation look more natural. Another trend is extended from mesh-based inverse kinematics (MeshIK) [103], which stacks deformations of mesh triangles into a single vector and applys PCA to extract a feature space from training examples. [104] takes into account of articulated constraints and applies MeshIK to skeletal animation with skinning.

Mesh Deformation We refer this kind of deformation to direct manipulation on mesh by specifying a set of positions or gradients instead of controlling pre-defined structures. Although all the above LBS methods can be modified to implicitly solve transformations of handles, we emphasize this category of approaches does not require to bind handles or simply relies on self-adaptive structures.

The survey paper [105] compares a number of related linear methods such as thin-shell method and Laplacian mesh editing [88]. As-rigid-as-possible surface modeling [94] is considered as one of the best linear deformations by taking local rotations into account. The prior art of nonlinear approach based on embedded deformation technique [72] has been successfully used in many recent applications dealing with flexibly deformation and achieving plausible mesh registration. The primary drawback of these approaches is that they rely on optimization at pose time, preventing an interactive manipulation.

Mesh Registration One application of mesh deformation is registration among multiple objects. [26] introduces the embedded deformation [72] to performance capture by registering a pre-scanned detailed object to input data. [70] utilizes the embedded method [26] to reconstruct human body shape by capturing a dense sequence in a turntable setup. Global registration errors are minimized by using a loop closure distribution approach. 3D Self-portraits [1] register eight surrounding scans of the object and automatically merge them into a complete watertight 3D model, only requiring the subject keeping still during each scan. However, all of these approaches require a large amount of pose time until one can obtain a decent registration result.

In addition to geometry registration, one widely-used free-form deformation is based on Gaussion mixture models (GMM). A general probabilistic framework called coherent point drift (CPD) is developed by [106], in which they fit the GMM centroids to the target data by maximizing the likelihood and impose the coherence constraint by regularizing the displacement field in the maximization step of the EM algorithm.

[90] develops a two-phase global registration approach for articulated objects similar to the EM algorithm. In the first step, they estimate joint locations and solve rigid transformations by searching the closest points; in the second step, they optimize weights for input samples and re-solve discrete labeling of rigid parts. [107] applies the method [90] to make a global consistent avatar by nonrigid registration, however, the EM algorithm costs a large amount of time. Another drawback of this kind of approaches is its lack of local property and it is hard to control a certain portion of the mesh intuitively or edit shape details incrementally.

Different from the above, our method deals with editing on the mesh directly without knowing the topology of the object or cumbersome manual rigging, has the comparable quality to the nonlinear embedded approach, but runs much faster than existing pose-time computing approaches especially when the target mesh is in high-resolution.

In the following sections, we will first give a brief overview of the original embedded deformation method and our linear system to solve translations (4.2). This is then followed by a GPU-based parallel rotation solver (4.2) and a linear metric to regularize all the rotations (4.2). These three steps lead to our basic solution (4.2) and then we present a hierarchy structure of the embedded graph to accelerate the rate of convergence (4.3); the computation complexity will be briefly analyzed (4.3). In Section 4.4, we will discuss possible applications of our algorithm and demonstrate the real-time performance.

4.2 Linear Embedded Deformation

The embedded mesh deformation computes a smooth warping field from a given 3D mesh $\mathcal{M}$ (vertex set $\mathcal{V}$ and face set $\mathcal{F}$) to a target mesh $\widetilde{\mathcal{M}}$ achieving position and normal consistency as described in the previous chapter 3.1. By coupling the nodes

$\boldsymbol{g} \in \mathcal{G}$, the regularization term can be represented by the following:

$$E_{\text{reg}} := \sum_{i=1}^{m} \sum_{\boldsymbol{g}_k \in \mathcal{N}(\boldsymbol{g}_i)} \alpha_{ik}^2 \left\| (\boldsymbol{I} - \boldsymbol{R}_i)(\boldsymbol{g}_i - \boldsymbol{g}_k) + \boldsymbol{t}_i - \boldsymbol{t}_k \right\|^2, \tag{4.1}$$

$$= \sum_{i=1}^{m} \sum_{\boldsymbol{g}_k \in \mathcal{N}(\boldsymbol{g}_i)} \alpha_{ik}^2 \left\| (\tilde{\boldsymbol{g}}_i - \tilde{\boldsymbol{g}}_k) - \boldsymbol{R}_i(\boldsymbol{g}_i - \boldsymbol{g}_k) \right\|^2 \tag{4.2}$$

where the weight α_{ik} is proportional to the degree to which the influence of nodes $\boldsymbol{g}_i$ and $\boldsymbol{g}_k$ overlap.

Existing approaches such as [72] implement Gauss-Newton iterations to optimize the above in an unconstrained nonlinear least-square problem by minimizing the objective function 3.1. In despite of the fixed non-zero pattern of Jacobian matrix in each iteration, which can be precomputed to speed up the solver, the overall computation cost is still too high to achieve interactive performance. The nonlinear problem arises from solving both $\boldsymbol{R}_i$ and $\boldsymbol{t}_i$ simultaneously. However, a direct observation is that if rotations are fixed, solving translations $\boldsymbol{t}_i$ becomes a linear least-square problem. The key insight of our approach is to solve for translation first and then solve for rotation in another efficient step.

Parallel Computation of Rotations When it comes to solving rotations $\boldsymbol{R}_i$ with $\boldsymbol{t}_i$ fixed, we notice that node positions of the deformed graph have been already fixed, and then rotations are mainly determined by the deformed graph to achieve the local orientation consistency. Specifically, every rotation $\boldsymbol{R}_i$ is determined by minimizing the energy function Eq. 4.2.

By representing each $\boldsymbol{R}_i$ using quaternion as shown in Appendix, computing each rotation is essentially equivalent to an eigenvalue decomposition of a 4×4 symmetric matrix, which can be efficiently solved by Jacobi eigenvalue algorithm and more importantly, can be explicitly implemented in GPU programming language such as CUDA. Our result shows that solving 100k rotations only costs several milliseconds

in a middle-end graphics card.

Rotational Regularization When rotation of each node has been solved in parallel, we observe that rotations over the graph are not necessary spatial consistent and make the deformed mesh unsmooth, we therefore provide a rotational regularization to constrain neighboring rotations over the graph. According to the regularization term 4.1, we notice that neighboring nodes $\boldsymbol{g}_i$ and $\boldsymbol{g}_k$ always appear in pairs and $\alpha_{ik} = \alpha_{ki}$ in symmetry. Therefore, we can combine both terms and apply a triangle inequality,

$$\begin{aligned}
&\alpha_{ik}^2 \left\|(\boldsymbol{I} - \boldsymbol{R}_i)(\boldsymbol{g}_i - \boldsymbol{g}_k) + \boldsymbol{t}_i - \boldsymbol{t}_k\right\|^2 \\
+&\alpha_{ki}^2 \left\|(\boldsymbol{I} - \boldsymbol{R}_k)(\boldsymbol{g}_i - \boldsymbol{g}_k) + \boldsymbol{t}_i - \boldsymbol{t}_k\right\|^2 \\
\geq&\alpha_{ik}^2 \left\|\boldsymbol{R}_i(\boldsymbol{g}_i - \boldsymbol{g}_k) - \boldsymbol{R}_k(\boldsymbol{g}_i - \boldsymbol{g}_k)\right\|^2 /2 \\
=&\alpha_{ik}^2 \left\|(\boldsymbol{I} - \boldsymbol{R}_i\boldsymbol{R}_k^T)(\boldsymbol{g}_i - \boldsymbol{g}_k)\right\|^2 /2.
\end{aligned} \tag{4.3}$$

Since $\boldsymbol{g}_i - \boldsymbol{g}_k$ can be in arbitrary configuration, it ends up to minimize the metric $\left\|\boldsymbol{I} - \boldsymbol{R}_i\boldsymbol{R}_k^T\right\|^2$, which is functional and bounded equivalent to the geodesic distance on the unit sphere, *i.e.*, $\Phi_g = \|\log(\boldsymbol{R}_i\boldsymbol{R}_k^T)\|^2$ as shown in [**?**], where the log map gives the skew symmetric matrix that embodies both the unit rotation axis and angle of the matrix and $\|\log(\cdot)\|$ therefore gives the magnitude of the rotation angle in the range $[0, \pi)$.

All the 3D rotations can be represented in the form of $\{\exp([\theta\boldsymbol{u}]_\times) : \theta \in (-\pi, \pi]\}$. Since the embedded warping field is smooth and the deformation is required to be as rigid as possible, we also assume that the rotation of an angle π, an extreme case most of deformation methods can not deal with, will never happen. And then the representation $\boldsymbol{R} = \exp([\theta\boldsymbol{u}]_\times)$ is unique and the exponential mapping $\boldsymbol{R} \mapsto \theta\boldsymbol{u}$ is injective where $\theta \in [0, \pi)$ is the rotation angle and $\boldsymbol{u}$ is a unit axis.

As the representation is unique, we can define a simple metric $\Phi : SO(3) \times SO(3) \to \mathbb{R}^+$ except the rotation angle π case,

$$\Phi(\boldsymbol{R}_1, \boldsymbol{R}_2) = \|\theta_1 \boldsymbol{u}_1 - \theta_2 \boldsymbol{u}_2\|. \tag{4.4}$$

It is easy to verify that the metric satisfies that 1) $\Phi(\boldsymbol{R}_1, \boldsymbol{R}_2) = 0 \Leftrightarrow \theta_1 = \theta_2$ and $\boldsymbol{u}_1 = \boldsymbol{u}_2$; 2) $\Phi(\boldsymbol{R}_1, \boldsymbol{R}_2) = \Phi(\boldsymbol{R}_2, \boldsymbol{R}_1)$; 3) $\Phi(\boldsymbol{R}_1, \boldsymbol{R}_3) \leq \Phi(\boldsymbol{R}_1, \boldsymbol{R}_2) + \Phi(\boldsymbol{R}_2, \boldsymbol{R}_3)$. Also we will show that the metric is bounded equivalent to the geodesic distance and hence the right hand side in Equation 4.3:

$$\begin{aligned}
&\lim_{\boldsymbol{u}_1 \cdot \boldsymbol{u}_2 \to 1} \frac{\Phi(\boldsymbol{R}_1, \boldsymbol{R}_2)}{\Phi_g(\boldsymbol{R}_1, \boldsymbol{R}_2)} = \\
&\lim_{\boldsymbol{u}_1 \cdot \boldsymbol{u}_2 \to 1} \frac{\sqrt{\theta_1^2 + \theta_2^2 - 2\theta_1\theta_2(\boldsymbol{u}_1 \cdot \boldsymbol{u}_2)}}{\arccos\left(2(\cos\frac{\theta_1}{2}\cos\frac{\theta_2}{2} + \sin\frac{\theta_1}{2}\sin\frac{\theta_2}{2}(\boldsymbol{u}_1 \cdot \boldsymbol{u}_2))^2 - 1\right)} \\
&= \frac{|\theta_1 - \theta_2|}{|\theta_1 - \theta_2|} = 1.
\end{aligned} \tag{4.5}$$

Therefore, when neighboring rotation axes are sufficiently close, we can replace regularization terms in Equation 4.1 by the following energy,

$$E_{\text{rot}}(\boldsymbol{\theta u}) = \sum_{i,k} \alpha_{ik}^2 \|\theta_i \boldsymbol{u}_i - \theta_k \boldsymbol{u}_k\|^2, \tag{4.6}$$

which is a straightforward least square problem and can be easily converted into a linear system. When some of $\theta_i \boldsymbol{u}_i$ are known according to the above section 4.2, we can move corresponding columns to the right hand side and solve the rest unknowns efficiently.

To define weights α_{ik}, we first initialize them as $\alpha_{ik}^0 = \max(0, (1 - \|\boldsymbol{g}_i - \boldsymbol{g}_k\|^2/r^2)^3)$ for all neighbors $\boldsymbol{g}_k \in \mathcal{N}(\boldsymbol{g}_i)$ and normalize to sum to one. And then α_{ik} assigned to

all the edges of the graph are solved by the following linear optimization,

$$\operatorname{argmin} \sum_{i,k} \left(\alpha_{ik} - \alpha_{ik}^{0}\right)^2 + \left(\alpha_{ik} - \alpha_{ki}\right)^2 . \tag{4.7}$$

Iterative Solution To summarize, we solve the embedded deformation in an iterative scheme involving three steps:

• SolveT: solving all the transformations by minimizing energy E_1 and fixing all the rotations (initials are identities);

• SolveR: computing all the rotations of the constrained nodes parallelly from eigenvalue decomposition;

• DiffuseR: interpolating all the other rotations in a linear system using Eq. 4.6.

The process iterates by feeding the resulting rotations back into step SolveT until all the transformations converge within a pre-defined threshold.

Comparing to the nonlinear optimization method by computing Jacobians [72], our linear approach involves much less computations and converges fast usually in 3∼5 iterations. Also eigenvalue decompositions can be easily implemented in GPU.

4.3 Real Time Algorithm for General Mesh Embedded Deformation

Hierarchical Algorithm The above iterative method has already performed better than a Gauss-Newton non-linear solver. In practice, however, we found that linear solvers of steps SolveT and DiffuseR prevent the overall performance to achieve real-time when the number of nodes are large. Since the embedded graph is leveraged to be as simple as possible and yet have sufficient number of nodes to represent complex shapes, thousands of nodes might be necessary to provide enough accuracy when the mesh to deform is highly detailed.

One observation is that the graph always deforms consistently with the mesh, that is the warped graph can be thought as a reduced version of the deformed mesh

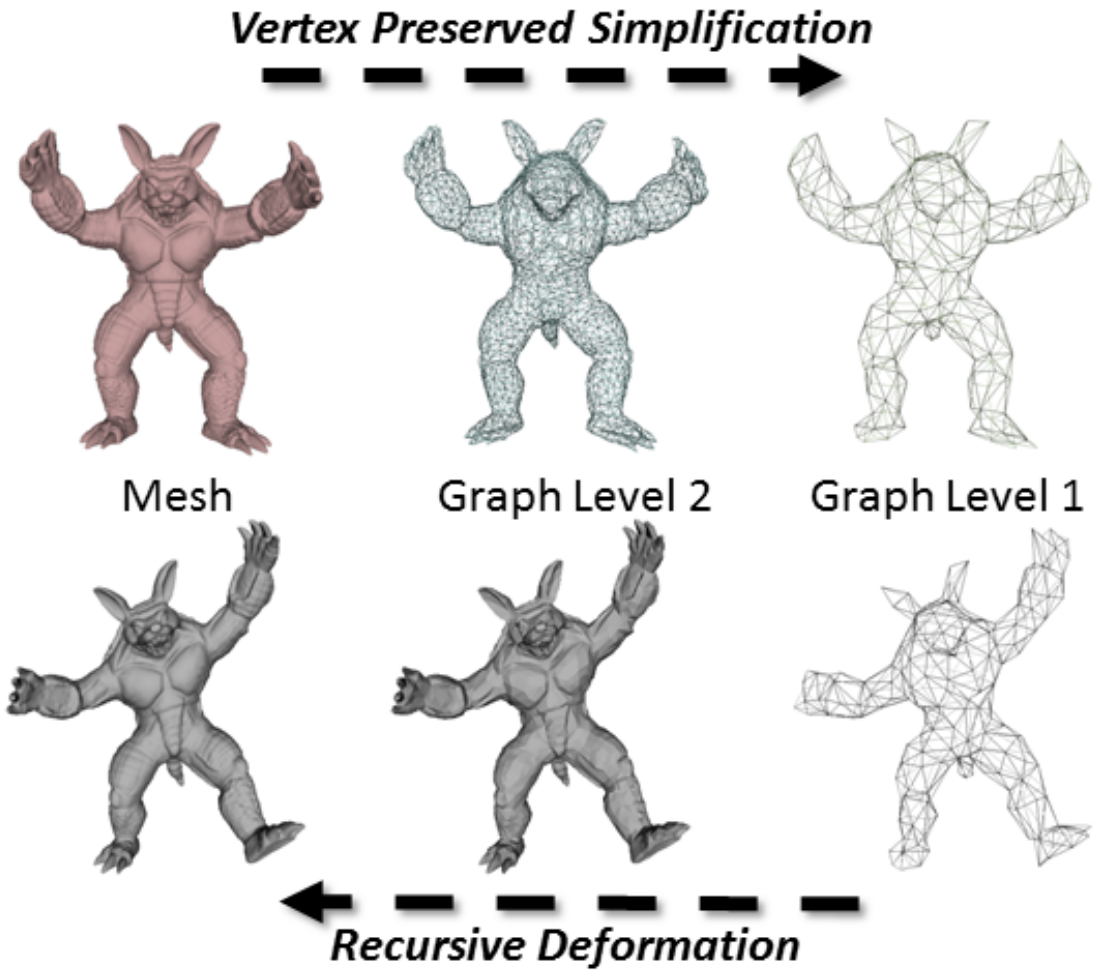


Figure 4.1: A graph hierarchy built for the bar-twist example. Two layers of graphs are plotted. In order to pass the transformation from a graph of coarser level, we simplify the mesh while preserving the node positions. In other words, the nodes of a lower level is the subset of a higher level.

(shown in Figure 4.1). Moreover, if positions of the warped graph are given, the whole deformation process is almost done only except that one more iteration is needed to solve the remaining unknown rotations. Motivated by these facts, we build a graph hierarchy instead of one single embedded graph.

Suppose $\mathcal{G}_l$ is the graph for the mesh defined in Equation 3.1, a new graph $\mathcal{G}_{l-1}$ can be built for the previous graph in the same manner by considering $\mathcal{G}_l$ as the source mesh but with a larger influence radius $r_{l-1} > r_l$. Node positions of a higher

level graph serves as the point terms in Equation 3.1 and we simply ignore the plane fitting term because the normal can not be meaningful when the graph is coarse and not smooth.

By building the hierarchical structure, we provide a recursive algorithm for the approach in 4.2. After finishing the SolveT step, a subset of nodes, which involves in the data term 3.1, is chosen to be a new set of constrained points for a higher level recursive call and the shape of the graph is then deformed according to its higher level graph. Denoting the subset index hierarchy as $\{\text{ind}_l\}$ and node hierarchy as $\{\boldsymbol{g}_l\}$, $l = 1, \ldots, n$, we summarize the entire procedure in Algorithm 5 and in our experiments, we choose $n \leq 3$.

Algorithm 5 The Recursive Linear Embedded Deformation: RLEDeform

Input: $\boldsymbol{x}, \{\text{ind}_l\}, \boldsymbol{y}, \{\boldsymbol{g}_l\}, n$
Output: $\tilde{\boldsymbol{x}}$

$\boldsymbol{R} \leftarrow \boldsymbol{I}$, $\boldsymbol{g} \leftarrow \boldsymbol{g}_n$, converged = false
$\boldsymbol{g}_s \leftarrow \text{ChooseSubsetByIndex}(\boldsymbol{g}_n, \text{ind}_n)$
while converged $\neq$ true **do**
  $\boldsymbol{t} \leftarrow \text{SolveT}(\boldsymbol{x}, \text{ind}_n, \boldsymbol{y}, \boldsymbol{g}_n, \boldsymbol{R})$
  $\tilde{\boldsymbol{g}} \leftarrow \boldsymbol{g}_n + \boldsymbol{t}$
  $\tilde{\boldsymbol{g}}_s \leftarrow \text{ChooseSubsetByIndex}(\tilde{\boldsymbol{g}}, \text{ind}_n)$
  if $n > 1$ **then**
    $\tilde{\boldsymbol{g}} \leftarrow \text{RLEDeform}(\boldsymbol{g}_n, \{\text{ind}_l\}, \tilde{\boldsymbol{g}}_s, \{\boldsymbol{g}_l\}, n-1)$
    $\tilde{\boldsymbol{g}}_s \leftarrow \text{ChooseSubsetByIndex}(\tilde{\boldsymbol{g}}, \text{ind}_n)$
  end if
  $\boldsymbol{R}_s \leftarrow \text{SolveR}(\boldsymbol{g}_s, \tilde{\boldsymbol{g}}_s)$
  $\boldsymbol{R} \leftarrow \text{DiffuseR}(\boldsymbol{R}_s, \boldsymbol{g}_n, \text{ind}_n)$
  if $\|\tilde{\boldsymbol{g}} - \boldsymbol{g}\| < \epsilon$ **then**
    converged = true
  else
    $\boldsymbol{g} \leftarrow \tilde{\boldsymbol{g}}$
  end if
end while
$\tilde{\boldsymbol{x}} \leftarrow \text{Deform}(\boldsymbol{x}, \boldsymbol{g}_n, \boldsymbol{R}, \boldsymbol{t})$

Performance Analysis First of all, we validate that our algorithm converges to the correct solution that minimizes the cost functions 3.1. Suppose $\{\boldsymbol{R}^i\}, \{\boldsymbol{t}^i\}$ are our result sequences at iterations $i = 1, 2, \ldots$. Substituting them into the cost function E_1, since $\boldsymbol{t}^2$ minimizes it by fixing $\boldsymbol{R}^1$, then $E_1(\boldsymbol{R}^1, \boldsymbol{t}^2) < E_1(\boldsymbol{R}^1, \boldsymbol{t}^1)$. On the other hand, $\boldsymbol{R}^2$ minimizes E_1 by fixing $\boldsymbol{t}^2$, and then $E_1(\boldsymbol{R}^2, \boldsymbol{t}^2) < E_1(\boldsymbol{R}^1, \boldsymbol{t}^2)$. Hence $E_1(\boldsymbol{R}^2, \boldsymbol{t}^2) < E_1(\boldsymbol{R}^1, \boldsymbol{t}^1)$. In the same manner, we can get $E_1(\boldsymbol{R}^{i+1}, \boldsymbol{t}^{i+1}) < E_1(\boldsymbol{R}^i, \boldsymbol{t}^i)$, $i = 1, 2, \ldots$. Therefore the sequence $\{E_1(\boldsymbol{R}^i, \boldsymbol{t}^i)\}$ is monotonic decreasing and bounded below by 0. It will converge to a small error $\epsilon > 0$.

It is also easy to check $\|\boldsymbol{t}^{i+1} - \boldsymbol{t}^i\| \leq E_{\text{fit}}(\boldsymbol{R}^i, \boldsymbol{t}^{i+1}) + E_{\text{fit}}(\boldsymbol{R}^i, \boldsymbol{t}^i) < 2E_{\text{fit}}(\boldsymbol{R}^i, \boldsymbol{t}^i) < 2E_1(\boldsymbol{R}^i, \boldsymbol{t}^i)$. Therefore $\{\boldsymbol{t}^i\}$ converges as E_1 goes to small enough. $\boldsymbol{R}^i$ can be considered as a function of $\boldsymbol{t}^i$ according to our representation and hence the sequence $\{\boldsymbol{R}^i\}$ also converges. Note that our solution does not mean a global minimum, where the error ϵ comes from the inconsistency in the energy E_{fit}. According to our results, we can still achieve a comparable solution to the original Gauss-Newton solver.

To compare with existing linear deformation approaches, as-rigid-as-possible surface modeling [94] can be viewed as an iterative version of Laplacian mesh editing [88] and each iteration solves a Laplacian deformation with local rotations. If we build a single graph to be exact the mesh itself and apply the SolveR step to all the nodes, our approach is then equivalent to the as-rigid-as-possible deformation. Since nonzero entries of the Laplacian matrix mainly locate near its main diagonal, each iteration takes time of approximately $\mathcal{O}(m_v^3)$ and the total time complexity is $\mathcal{O}(km_v^3)$, where m_v is the number of mesh vertices and $k \leq 5$ is the number of iterations.

Our method has a lower time complexity in two folds: firstly, each embedded graph layer has much less nodes than the mesh in the below layer $m_1 \ll m_2 \ll \cdots \ll m_n = m \ll m_v$ (in which we choose $\frac{m_l}{m_{l-1}} = \gamma \approx 10$ in our experiments); secondly, the convergence is faster thanks to the upper layer deformation. According to our results in section 4.4, k is usually no more than 2 except for the top layer. In summary, our

embedded hierarchy approach can be considered as an acceleration scheme for the as-rigid-as-possible deformation, reducing the time complexity to $\mathcal{O}(\sum_{l=2}^{n} m_l^3 + km_1^3) \approx \mathcal{O}(\frac{r^3}{r^3-1}m^3)$.

To compare with the nonlinear method [72], the linear system $\boldsymbol{J}^T\boldsymbol{J}$ it solves is much larger due to that the number of unkowns is the same as the number of constraints in Equation 3.1, which is larger than the number of nodes m in our method, and also a Gauss-Newton solver takes more iterations to converge.

4.4 Applications and Results

We present both synthetic and real world experimental results to demonstrate the qualitative and speed performance of our method and also compare with existing approaches such as the as-rigid-as-possible modeling [94] and the nonlinear embedded manipulation [72]. As an interesting application, we provide an implementation of the 3D Self-Portraits [1] using our deformation model and compare the overall performance. Our performance measurement is taken on a 3.4GHz quad-core CPU and a Nvidia GTX 560Ti graphic card using unoptimized Matlab/C++ hybrid code.

User Interactive Editing First of all, we evaluate our method with a benchmark dataset provided from the survey paper [105]. Time costs are measured for Laplacian mesh editing, as-rigid-as-possible modeling, embedded deformation and our method. Specially, we time for our single layer and hierarchical versions separately. In Table 4.2, the single layer version has the same bind time as the embedded manipulation but has a much faster pose time. According to our experiments, when the number of nodes does not exceed 1500, the deforming time of the single layer method is always less than 0.5 second. We choose an additional graph layer of 120 $\sim$ 250 nodes for testing our hierarchy method and achieve an average of 10 $\sim$ 16 frames per second even in Matlab code.

Table 4.1: The complete model is built from eight partial scans while the user rotates himself about 45° degrees for each shot. The average number of nodes used to align each two scans is 1000. The 3D Self-Portrait [1] takes 2040 seconds while our method takes 12.16 seconds for 5 iterations in total.

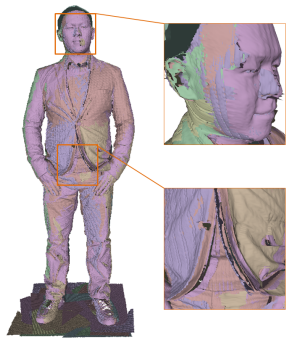
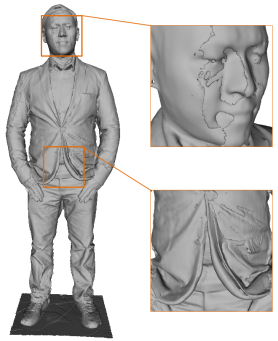
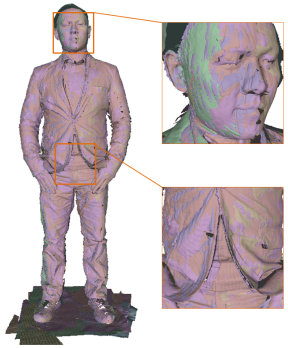
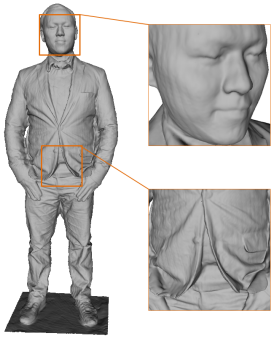
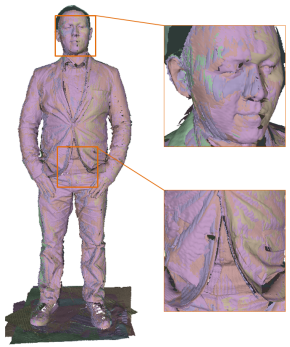
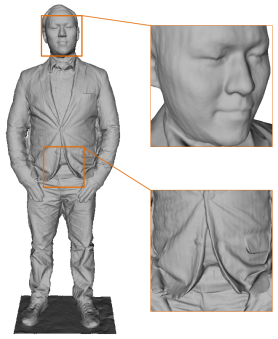
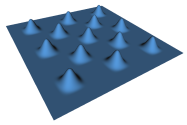
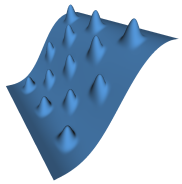
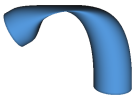
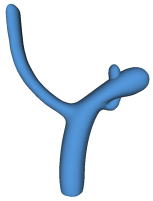
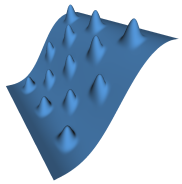
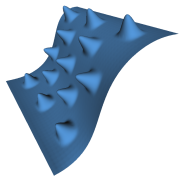
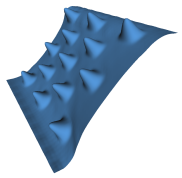
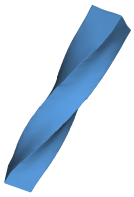
Approach	**Alignment**	**Result: Frontal View**	**Result: Left View**	**Result: Right View**
Rigid Alignment				
3D Self Portaits [1], 2040 seconds				
3D Self Portaits with Our Deformation Method, 12.16 seconds				

Table 4.2: A benchmark dataset

Approach	**Pure Translation**	120° **Bend**	135° **Twist**	70° **Bend**
Original Model	 40401 vertices	 4802 vertices	 6084 vertices	 5261 vertices
Laplacian Surface Editing [88]	 2.16 sec	 0.23 sec	 0.31 sec	 0.26 sec
As-Rigid-As-Possible Modeling [94]	 7.63 sec	 1.28 sec	 2.09 sec	 1.01 sec
Emedded Manipulation [72]	 1681 nodes bind 0.27 sec pose 35.89 sec	 530 nodes bind 0.029 sec pose 8.39 sec	 1001 nodes bind 0.043 sec pose 9.83 sec	 1053 nodes bind 0.046 sec pose 24.47 sec
Our Method (Two Layers)	 121 nodes bind 0.31 sec pose 0.094 sec	 138 nodes bind 0.061 sec pose 0.073 sec	 249 nodes bind 0.090 sec 0.068 sec	 252 nodes bind 0.10 sec pose 0.062 sec

For more synthetic object editing, we present an interactive UI allowing the user to click a part on the model, and drag or rotate that part. The selected part serves both point and normal terms for the deformation energy Eq. 3.1. In Table 4.3, the selected parts are shown in the first column. When operating on one part, we assume that the others are fixed. To compare with the embedded method [72], we record every destination points with their according transformations and execute the nonlinear embedded approach offline. Results and performance comparisons are shown in Table 4.4 and Table 4.3. In the experiment, we choose two graph layers for Alligator and Hand examples and three graph layers for the others. Our supplemental video also provides the complete UI operations using a simple Matlab interface.

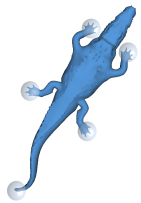
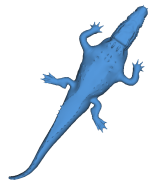
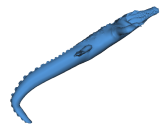
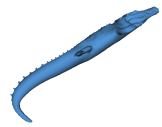
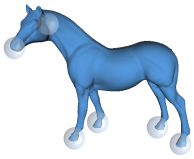
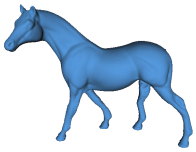
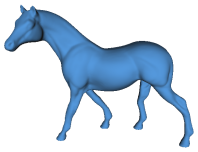
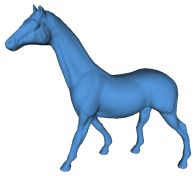
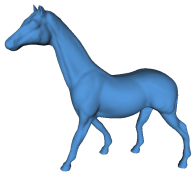
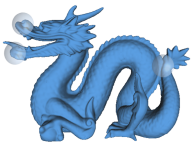
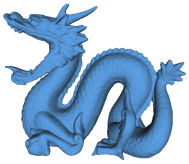
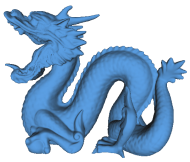
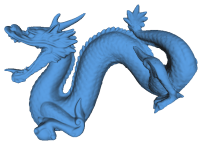
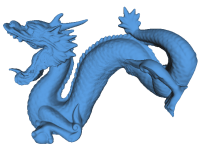
Table 4.3: Speed test on synthetic data. The time unit is second.

	Alligator	Hand	Horse	Armadillo	Dragon
♯Vert	17k	36k	48k	173k	437k
♯Nodes	915	1748	2204	2881	4979
♯CtrlPts	3946	4219	7090	17319	49081
BindTime	0.19s	0.35s	0.44s	1.78s	3.74s
Embedded	13.94s	161.1s	327.27s	92.74s	407.70s
Ours	0.079s	0.047s	0.106s	0.114s	0.134s

Non-rigid Mesh Registration Another interesting application of nonrigid registration is so-called 3D Self-Portaits [1], which reconstructs a person's full body model from multiple scans, and can potentially be used for 3D printing, social network and entertainment. The actor is required to rotate himself in front of a scanner but keep still and the same pose every time. Once all the partial scans are registered, a complete model can be extracted by using volumetric methods.

The whole process is essentially to perform a global non-rigid registration in an iterative closest point (ICP) strategy and apply loop closure constraints [1]. In our setup, we use a structured light device to capture eight scans and the actor rotates about 45 degrees each time. We first estimate the rotation axis as an initial guess and

Table 4.4: The interactive mesh editing results

Original Model	**Embedded [72]**	**Our Method**	**Embedded [72]**	**Our Method**
				
				
				
				

align all eight scans rigidly by using ICP method. Since subtle movements can easily result in errors by only using rigid registration (fist row in Table 4.1), a nonrigid refinement is necessary to make subtle surfaces consistent.

In this case, we build three graph layers for each partial mesh once in the preprocess (250k vertices and 3k nodes on average). With a similar correspondence searching step [1], we deform each partial mesh to its next neighbor. Loop closure constraints are applied after all the meshes are registered. To avoid local minima, we also employ a relaxation framework: if the global registration is done, we reduce the regularization weight in half $w_{\text{reg}} \leftarrow w_{\text{reg}}/2$ and repeat the process.

We obtain a visually plausible result after 5 iterations in a total of 12.16 seconds

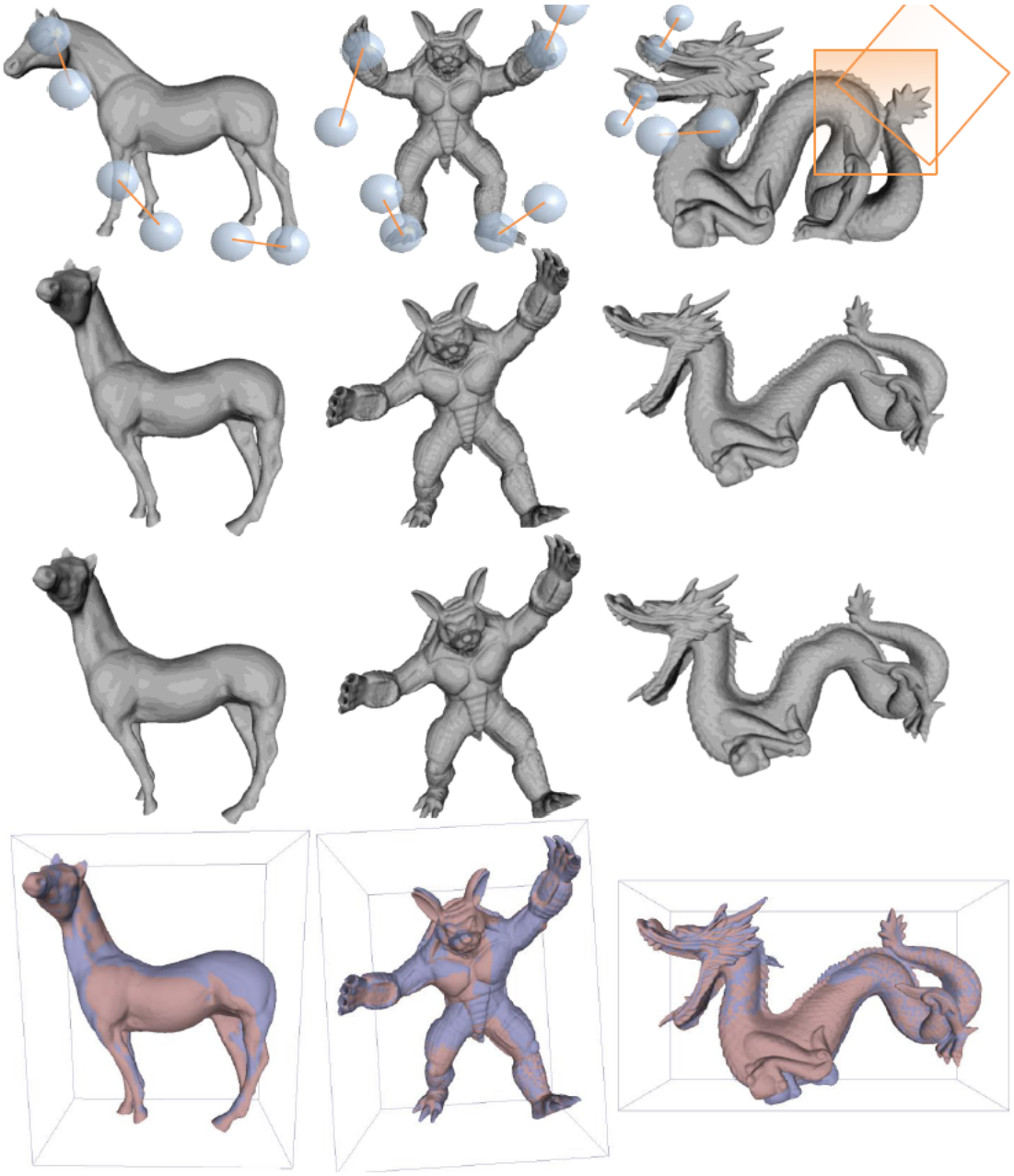


Figure 4.2: The interactive editing of synthetic data. The first row shows control vertices and parts. The second and third rows are the results from the embedded method and from ours respectively. The last row shows the compared results .

while the original approach [1] costs about 2040 seconds in 5 iterations. The final output results are comparable as shown in Table 4.1.

Chapter 5 Body Swap: Application to Virtual Try-On

In this chapter, we focus on an application of our GMM-BlendSCAPE framework to swap dressed people in 2D videos. Instead of explicitly modeling clothes geometry such as [30] or clothes physical simulation such as [108], we only estimate 3D body shape that leads to clothes geometry change and finally use the shape prior to guide video re-shaping or re-targeting.

Clothing animation plays an important role in all kinds of applications involving dressed virtual characters. Early clothing representation relies on texture mapping on the body geometry or coarse clothes meshes. To make better quality animations, physics based simulation (PBS) [108] are employed into clothes modeling, which has the advantage of producing realistic visual effect, however with a relatively high computational cost. Furthermore, the results of clothes simulation are specific to a particular body model. Each character requires a new simulation with typically manual initialization. These limitations make PBS suitable to animated movies that have an abundant time budget and a limited number of characters, but not for applications such as internet-scale virtual fashion or retail clothing try-on.

Virtual try-on, due to its large commercial potential, has been explored before. The general idea is to track the user's motion, in either 2D or 3D [108–111], and synthesize clothes that can be overlayed on the user's image. However, realistic cloth simulation is a long-standing open problem in computer graphics. Instead we choose an image-based approach to replace/reshape the human who is under the clothes with a different one. In order to generate photo-realistic results, we present this body swap application as a RGB-D video synthesis system that can replace the full body of a human subject, including face, with a different one. The different one can come from another person, or computer synthesized. The driving application for our system is

virtual try-on. With the caveat of capturing with a RGB-D camera, we provide a way to let anyone to virtually appear on fashion shows, with the proper body shape and face appearance.

Our system requires a standard RGB-D camera. It first acquires a body model of Actor A. The body model is based on a personalized GMM-BlendScape model, which is deformable and animatable as presented in previous chapters. Then RGB-D video footage of Actor A wearing different clothes are captured. The body pose of Actor A is tracked with a novel formulation that combines a probabilistic tracking using Gaussian Mixtures with a personal BlendScape model. For a new actor B, his body shape is also acquired and fitted with another BlendShape model. Now given the correspondence between the two models of Actor A and Actor B, we apply image warping to the RGB frames, which contains A, to match the body-shape of B. In addition, the face of Actor A is also tracked and replaced with B's face. The end result is a new video sequence that looks as if Actor B were in it.

The overall pipeline is illustrated in Figure 5.1. From a technical standpoint, mostly related to our work is the MovieReshape by [112], in which the body shape of a human subject can be changed as a post-processing step. While very realistic results are demonstrated, it was acknowledged that loose clothes (such as long skirts) would problematic. We are able to overcome this difficulty by using (1) 3D data, and (2) a probabilistic formulation on in our tracking method, instead of explicit correspondences. In addition, our system can replace a subject's face as well.

5.1 Related Work

Cloth Modeling and Simulation Modeling real fabric material has a long list of literature but still remains an open problem to achieve realistic effect. Elastic models have been devoted to find numerical solutions to a variety of specific structures [113, 114]. To widen the range of materials, data driven approaches are developed to make

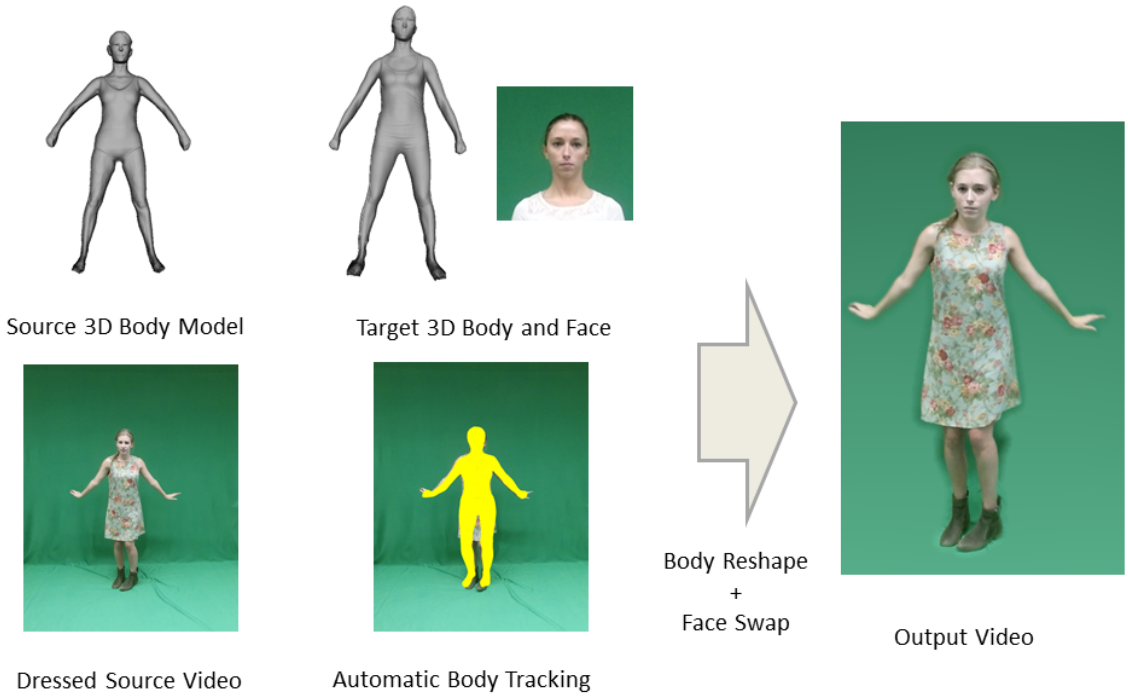


Figure 5.1: The pipeline of our body swap system.

the elastic bending effect more accurate [115, 116].

The active topic on cloth simulation focuses on modeling the physical properties of cloth and developing stable methods that can deal with cloth collisions, friction, and wrinkle buckling [29]. To overcome the high computational cost of traditional simulators, efficient approaches are developed including the Verlet integration scheme [117] and GPU acceleration [118] and widely applied in game development.

Morphable Clothes Model To avoid employing complex simulator, many variant applications have been developed based on the simple SCAPE-like model [49]. The Naked Truth [55] estimates human body shape under clothing. DRAPE (DRessing Any PErson) [30] uses the naked body under clothing and learn the clothing shape and pose model. All these methods rely on a large training database. These result models lack facial details, hairs, and clothing effect.

Virtual Try-On Compared with the previous two problems, virtual try-on and personalized 3D garment design is a much less studied problem. Most existing systems treat virtual clothing as static texture patches and use image-based rendering techniques to virtually drive the cloth [119]. Many methods rely on a pre-captured database with subjects in a large variety of poses to find a best match and perform local refinement [109, 120–122]. While these methods, to a large extend, ignore the interaction between users and the clothes, some pioneered this area by combining real-world data with physically based cloth simulation. There are two main strategies to animate virtual clothing in a virtual try-on system. A straightforward and robust way is to create an avatar that has the same body shape as the user, and then simulate virtual clothing on it. The body size can either be specified by the user input [110, 123, 124], or using depth sensors [125]. While these techniques can accurately model virtual clothing on a static body shape, they cannot easily handle body motions. The triMirror system [110] simulated virtual clothing on a moving avatar, whose motion was controlled by the user's skeleton pose. However, as its result showed, their system seemed to use a pre-defined avatar which did not exactly match the user's body shape. Alternatively, it is preferable to obtain body shape from depth data. One such example is the Fitnect [111] system. While it successfully animated part of the clothing by body motion, the rest still needed to be static. In addition, it only treated clothes as a piece of cloth in front of the user, and it had difficulty in forcing the clothes to follow body motion exactly. Compared to the existing methods, our system can effectively capture the pose and shape of the user, and provide realistic cloth simulation.

Face Replacement A lot of previous works have been devoted in this area. The commercial software *FaceSwap* can realistically swap inner face of two subjects. In [126], they successfully replace the face in a video sequence with different ex-

pression, however, they didn't handle poses so that their result is more like the morphed version of source and target person. [127] comes up with an efficient pipeline to transfer one specific photographic style to a static head portrait. [128] renders a target movie sequence to a tone and style of given exampled movie sequence smoothly in temporary space.

5.2 System Overview and Preliminary

Our system consists of three stages (summarized in Figure 5.1): building a personalized morphable body model for either an actor or a new customer (Sec 5.2), automatically tracking the animated body in a video sequence (Sec 5.3), and reshaping the video sequence to the new customer's body size with face replacement (Sec 5.4). Specifically, for the video of the actor or person A, we record a single-view RGB-D video sequence dressed in the target clothes; on the customer side or person B, we only capture static KinectFusion [65] scans or allow users to enter body sizes. We currently use a Kinect V2 depth sensor, which inherently provides a background segmentation and a initial skeleton tracking.

As an application of our GMM-BlendSCAPE framework (Chapter 2), we apply the markerless fitting algorithm to the RGB-D video, also considering both the sensor's skeleton tracking and image silhouette cues to enhance the body tracking accuracy.

Build A Personalized Morphable Model The first step of our system is to build a personalized morphable model which is later used for tracking or animating in a video. To capture details of body shape, we take multiple KinectFusion scans of a person who dresses tightly and keeps a static pose during a scan as Figure 5.2. The scans are similar to the input of the 3D Self-Portaits system [1]: the person rotates 45 degrees at each scan and body poses are not required to be exactly the same.

Different from 3D Self-Portaits, which builds a static detailed human model, our

goal is to use the model to animate a video sequence of the actor wearing outer clothes. Therefore, we do not require a precise detailed personal model and only need the body shape of the subject, who is required to wear tight-fitting clothes.

Similar to the process of GMM-BlendSCAPE parameter estimation in Chapter 2, we apply the shape adaption for all the static scans this time. First we fit the BlendSCAPE model to the frontal partial scan using the algorithm GMM-BlendSCAPE and provides a rough initial guess of the body shape $\boldsymbol{\beta}$. And then we fix the shape and apply GMM-BlendSCAPE to every static scans and get multiple pose parameters. Next, each pose parameter is fixed and and a common $\boldsymbol{\beta}$ is solved by taking all scans into account. The process iterates and converges to a local optimum of a unique shape parameter $\boldsymbol{\beta}$ and multiple pose parameters in Figure 5.2.

The personalized model is based on BlendSCAPE model but added a detailed layer $\boldsymbol{P}$ to the pose-independent shape matrix, $\boldsymbol{D}^{\text{new}} \leftarrow \boldsymbol{P}\boldsymbol{D}(\boldsymbol{\beta})$. To establish the detail transformation $\boldsymbol{P}$, we simply project the aligned BlendSCAPE model to the frontal and back scans along each vertex normal direction and compute the new transformation for each triangle as $\boldsymbol{D}^{\text{new}}$. This personalized BlendSCAPE model is ready for pose stracking, which will be discussed in the next section.

5.3 Depth Sequence Tracking

The core contribution of our system is a robust automatic body tracking framework with the personalized BlendSCAPE model built in the above paragraph. The inputs are the human body mesh sequence extracted from depths and masks and associated with skeleton provided by Kinect depth sensor.

Pose Initialization Before fitting personalized BlendSCAPE to the mesh sequence, we compute an initial pose configuration with joints read from Kinect data. We first compute the global transformation $[\boldsymbol{R}_g \ \boldsymbol{t}_g]$ by aligning the root joint and its chil-

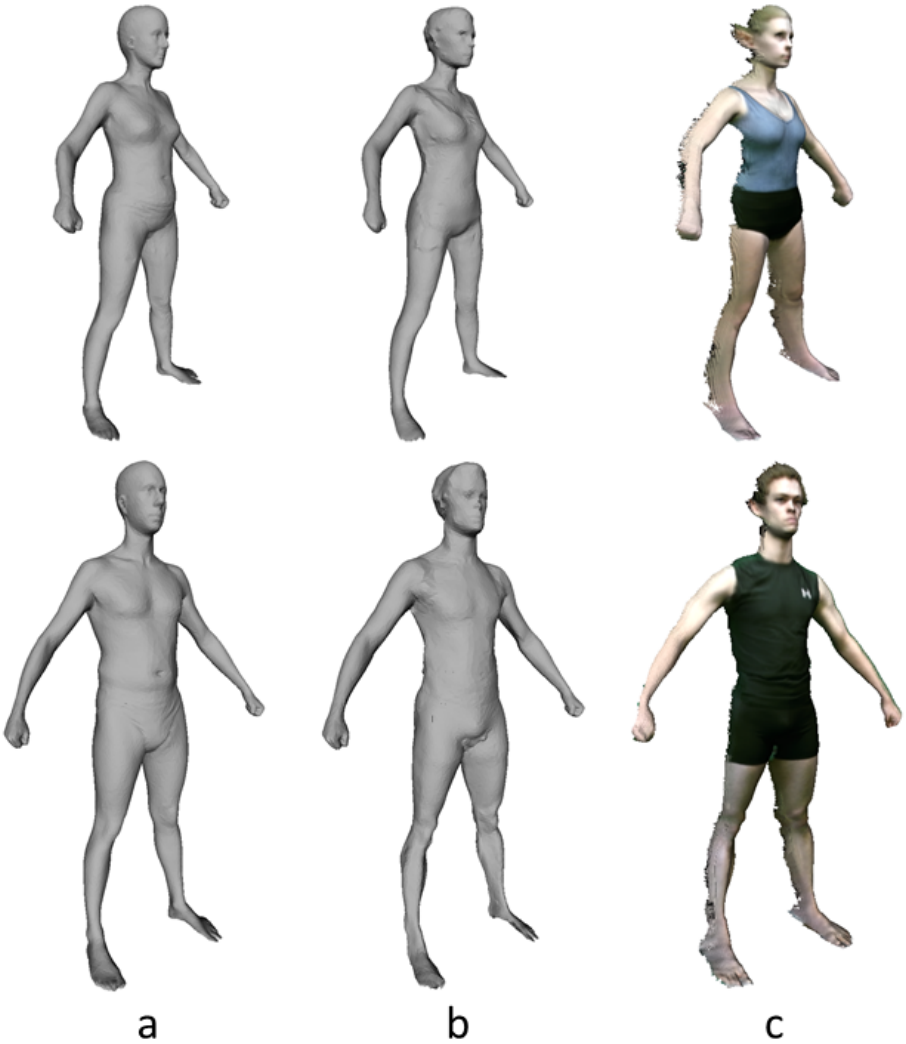


Figure 5.2: The personalized BlendSCAPE models. Column a) shows the fitted BlendSCAPE model to partial scans of column c); Column b) shows the personalized model after adding the detailed shape layer on the BlendSCAPE model.

dren around the body pelvis area to the input skeleton. And then we employing an optimization strategy similar to [60] to solve $[\boldsymbol{R}_b\ \boldsymbol{t}_b]$ of each bone, where bones are updated one by one while keeping the remaining bones fixed and traversed in the skeleton kinematic tree.

To estimate $[\boldsymbol{R}_j\ \boldsymbol{t}_j]$ of bone j, we also solve the energy function in eq. **??** in an EM iterative process combining with joint constraints to optimize in the following object function.

$$\sum_{m,n} p_{mn} \left\| w_{m,j}(\boldsymbol{R}_j \boldsymbol{v}_m^0 + \boldsymbol{t}_j) + \sum_{b \neq j}^{B} w_{m,b}(\boldsymbol{R}_b \boldsymbol{v}_m^0 + \boldsymbol{t}_b) - \boldsymbol{x}_n \right\|^2 \tag{5.1}$$
$$+\lambda_J \sum_k \|\boldsymbol{R}_j \boldsymbol{C}_k^0 + \boldsymbol{t}_j - \boldsymbol{C}_k\|^2,$$

where $\lambda_J = MN$ is assigned to a large weight to match the all the possible input joints $\boldsymbol{C}_k$ on the bone. Finding the optimal solution is a standard weighted Absolute Orientation problem and can be solved by dual quaternion representation or SVD decomposition [69]. In practice, we prune the energy by only solving the terms of weight $w_{ij} > 0.6$. Note that since the number of joint terms are always less than 2, when ambiguous multiple rotations are found with small errors, we choose the rotation that has a smaller joint angle to its parent bone in the skeletal tree.

Once all the bone transformations are solved, we can compute the initial pose parameter of the BlendSCAPE model and drive the model for each frame. Although we have observed in our system that the initial process provides rough and unstable pose tracking result, it can reduce pose tracking ambiguities when the human body moves within loose clothes.

GMM-BlendSCAPE Tracking Refinement Since the initialization step considers only one body part at each optimization, it may generate unsmooth pose configuration. We perform a refinement process to perform the following two steps iteratively:

1. GMM-BlendSCAPE Fitting the depth input as in Algorithm 3 .
2. Refining vertices on the model and recompute pose $\boldsymbol{\theta}$.

In the second step, we refine the pose parameter Θ by minimizing the following

energy:

$$E_f(\Delta\Theta) = \lambda_s E_s(\Delta\Theta) + \lambda_p E_p(\Delta\Theta) + \lambda_l E_l(\Delta\Theta), \tag{5.2}$$

$$E_l(\Delta\Theta) := \sum_{m=1}^{M} \|(\boldsymbol{v}_m - \boldsymbol{v}_m^{old}) - \frac{1}{|\boldsymbol{N}_m|} \sum_{j \in \boldsymbol{N}_m} (\boldsymbol{v}_j - \boldsymbol{v}_j^{old})\|^2, \tag{5.3}$$

where E_l is a Laplacian regularizer to ensure the smoothness of body deformation. The solution is similar to the inverse kinematics problem and ends up solving a linear system for parameters $\Delta\Theta$ in each iteration.

The purpose of the whole pose refinement is both to constrain the estimated body within the silhouette of captured depthmap and to push the body inside the clothes surface. The whole refinement algorithm is summarized in Alg. 6. Figure 5.3 gives an example frame after five iterative refinement.

Algorithm 6 Pose Θ Refinement

```
Given the initial Θ
while Θ not converged do
  Update vertices v_m^Θ
  Transfer part labels to the depth mesh
  Check silhouette constraints and compute new positions
  Remove outliers
  Check interpenetration vertices
  Solve inverse kinematics problem by Eq. 5.2
end while
```

Silhouette Penalties The first term E_{sil} in Eq. 5.2 stands for the silhouette penalty. The silhouette constraints are checked for each body part individually:

- Arms within masks of arms and shoulders (left and right respectively);
- Legs within masks of legs and trunk (left and right respectively);
- The head within the mask of head;
- Shoulders and trunk within the whole mask.

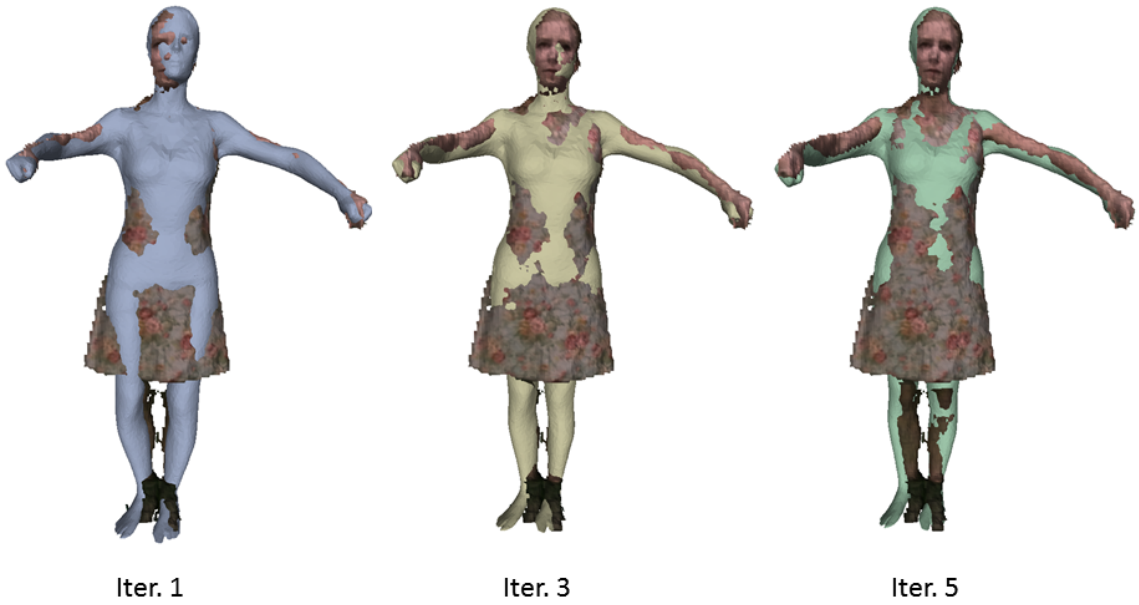


Figure 5.3: The pose and mesh refinement for the first five iterations.

Silhouette penalties have two-fold meanings: first, if the projection of a vertex lies outside of the corresponding silhouette, we find the closest 2D point on the silhouette boundary, unproject it as a line passing through the camera center and image pixel, and then choose the projection on the line as the new position; second, we render the body onto the depth image to get the body silhouette. If a depth point locates outside of its body silhouette, we unproject it as a line and find the closest point on the body silhouette and take its new position as its projection on the line.

Different from existing 2D image based silhouette refinement approaches such as [112, 129], given Kinect meshes as input, we dedicate a more precise scheme to penalize the body shape in 3D as well as considering the body parts semantics.

We label out seven body parts (arms, legs, shoulders, trunk and head) according to the skinning weight beforehand. When a body shape $\{\boldsymbol{v}_m^{\Theta}\}$ is reconstructed via an estimated Θ, we find point correspondences $(\boldsymbol{u}, \boldsymbol{v})$ between points on Kinect mesh and vertices of body by using the following criteria:

(1) The corresponding point $\boldsymbol{u}$ on Kinect mesh locates near (a small geodesic

distance) the intersection of the line passing through the body vertex along its normal and the mesh.

(2) The angle between normals of $\boldsymbol{u}$ and $\boldsymbol{v}$ is small.

(3) The intersection $\boldsymbol{u}'$ of the line through $\boldsymbol{u}$ along its normal and the body should be close to $\boldsymbol{v}$ in the body's geodesic distance.

(4) The angle between the normals of $\boldsymbol{u}'$ and $\boldsymbol{v}$ is small.

Once seed points on the Kinect meshes are found, we assign part labels from the corresponding vertices and propagate the label through the mesh by breadth-first search.

After searching correspondences for every pair of parts, we employ an effective strategy to suppress outliers:

(1) The moving distance will not exceed the standard deviation of all moving distances.

(2) The new position will not be occluded by the new body.

(3) The face normal of the new body will still point towards the camera.

Denoting the final subset of moved points as $\boldsymbol{S}$, the silhouette penalty is simply defined as:

$$E_s(\boldsymbol{\theta}) = \sum_{i \in \boldsymbol{S}} \|\boldsymbol{v}_i^{\text{new}} - \boldsymbol{v}_i\|^2. \tag{5.4}$$

Clothes-body Interpenetration The second term E_p in Eq. 5.2 prevents the body-mesh interpenetration, *i.e.*, vertices "*in front of*" the input mesh. We detect the clothes-body interpenetration along the normal direction $\boldsymbol{n}$ of a body vertex $\boldsymbol{v}$. Suppose the intersection $\boldsymbol{u}$ is sought on the depth mesh, the interpenetration happens when $\boldsymbol{n}^T(\boldsymbol{u}-\boldsymbol{v}) < 0$. Inspired by the refinement method in [30], we define the penalty for the set of penetrated vertices $\boldsymbol{P}$:

$$E_p(\boldsymbol{\theta}) = \sum_{i \in \boldsymbol{P}} \|\epsilon + \hat{\boldsymbol{n}}_i^T(\boldsymbol{u}_i - \boldsymbol{v}_i)\|^2, \tag{5.5}$$

where the clothes thickness term $\epsilon = -1$mm pushes the body sufficiently inside the clothes.

5.4 Video Re-targeting

The re-targeting stage consists of three steps: first, we build the customized BlendSCAPE model for the new person (Sec 5.2); second, we refine the head pose tracking and replace face for the new person in the video; in the last step, the video after face replacement is reshaped by image warping techniques.

Head Pose Refinement Since the depth sequence tracking (Sec 5.3) is a global body pose estimation approach, which is insufficient for accurate head mesh alignment, we perform a head refinement process before the face replacement by adding landmark detection and ICP registration.

In the automatic head tracking procedure,we assume that human head is a relatively rigid object and motion between two successive frames are small. Given a input of the RGB-D sequence and the pre-fitted model of the person, the goal of head refinement is to register the head part to each depth mesh frame. The first refinement is to employ a robust ICP algorithm: for each frame, we initialize the head pose with the previous frame, then perform ICP to register the head part from the model to depth mesh. The motion between successive frames can be simply regarded as rigid transformation, therefore we can incrementally multiply these rigid transformations along the video sequence. To prevent the interruption from sensor noise and occlusions, instead of applying ICP directly, we first extract a loose bounding box of the head from the previous frame, and perform ICP to the points only in the box.

One drawback of ICP refinement is that the tracking will drift by accumulation error and large motion between frames may lead to a tracking failure. To address the failure case, we leverage 2D landmarks to initialize our 3D alignment. First, we

utilize the standard approach [130] to detect 68 2D landmarks in an image, and then we shoot rays from landmarks to get intersection on the mesh, which are considered as the visible 3D landmarks. Given the same set of 3D landmarks in the previous frame, we can solve the transformation between landmarks as the initialization. Once a failure of tracking detected, in which the rotation angle between current frame and previous frame exceed a threshold or the projected landmarks has a big error with the corresponding landmarks detect in color image, we perform the landmark registration in the first failed frame. According to our experiment, our head pose refinement can handle big pose jump even in the turning around case, where no 2D landmarks can be observed.

Face Re-targeting After head pose refinement, we obtain a sequence of the source head aligned to each depth frame. The next is to build correspondence between the target head model and each source head. We adopt a hybrid of Laplacian mesh deformation [88] and embedded mesh deformation [72] to align the detailed full scan of target head model to every source head. Since the source head has already been aligned to each depth frame, we get the alignment of the target head to each depth frame.

We then project visible correspondent vertices in both source and target head model to the image and use such 2D correspondences to guide the image warping. After image warping, we have two face images of the same target shape: the warped source person and the rendered target person. By subtracting a pre-defined face mask on the target head model, we then perform seamless image replacement for all the frames. Figure 5.4 illustrates the whole pipeline of our face re-targeting stage.

Our goal of face replacement is to replace human face in a video sequence with extreme poses naturally and seamlessly with transformed appearance and shapes. To achieve this, we first adjust the target image color to fit the source. We employ a

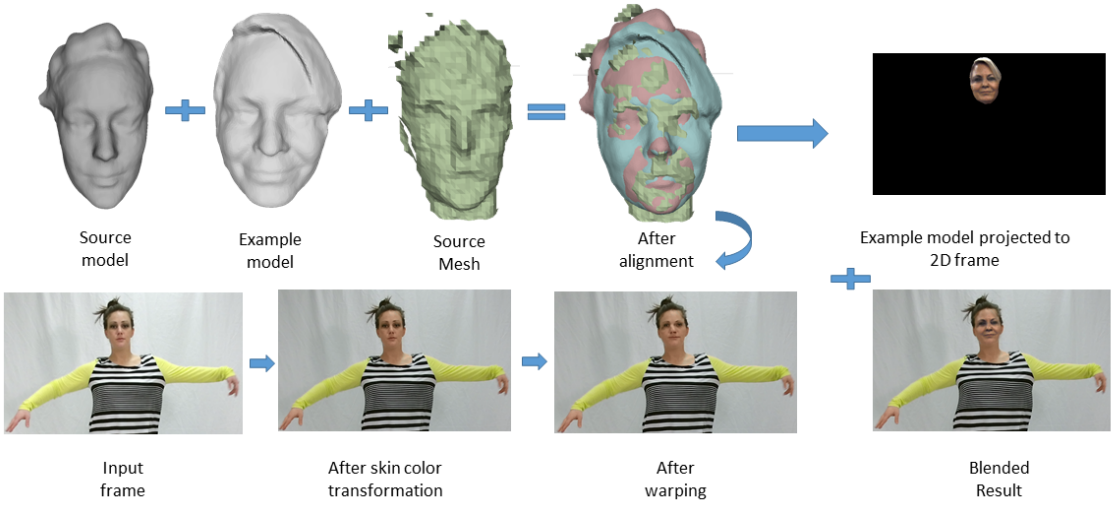


Figure 5.4: The pipeline of face retargeting process

method similar to [131] and [127] to first decompose the source and target images into multiple layers in CIE_LAB color space (7 layers in our case). And then a per pixel gain map is computed in each layer by minimizing the following energy function:

$$\|t_{L(p)} \cdot G_{L(p)} - S_{L(p)}\| + \|G_{L(p)} - G_{L(q)}\| \tag{5.6}$$

in which t_L is the target image in level L, S_L is L-th layer of source image, and p denotes any pixel in the face mask. The unknown to solve is the per-pixel gain map $G_{L(p)}$. The first term ensures that the tuned pixel in target matches the intensity of the one in source, and the second term ensures the spatial smoothness by considering adjacent pixels p and q. After applying the gain map in each layer, we composite the color harmonized target image from all all the color corrected layers. The target image is then matted with source to generate the final result.

It is needed to mention that we deal with the occlusion case independently as shown in the Figure 5.5. In this case, we warp the depth map in the same way as the corresponding color image, and then we apply K means method to segment foreground and background and generate a mask (c in Figure 5.5). The mask is

automatic combined in the matted process as:

$$\alpha \cdot I_{target} + (1 - \alpha) \cdot I_{source} \tag{5.7}$$

in which I_{target} is the warped background (b in Figure 5.5) occluded in some regions that will be finally filled with foreground I_{source} (d in Figure 5.5).

Figure 5.5: Illustration of dealing with occlusion cases

Image Warping We deal with image warping separately in the case of face replacement and body reshaping, because different from body reshaping, face warping requires to keep as rigid as possible and the warping is only performed in the face area. Specifically, we warp the face by using a similar method to [132], in which we add a background fix term to prevent body from moving.

$$E_d + \alpha * E_s + E_b \tag{5.8}$$

After the face re-target, we warp the whole body by using [133] with the projected vertices of source and target model as control points.

5.5 Results

We perform the video reshape and re-targeting from two dancing Kinect color+depth sequences: a slim girl dressing on a skirt and rotating 360 degrees in front of the camera; a male dancer with large arm movement. The result figures 5.6 and 5.7, and the supplemental video show that the quality re-targeting result by modifying videos of loose clothes and with large movement to target persons: the slim girl is replaced by a taller woman and the male dancer is changed to a tall and strong man.

Figure 5.6: The video result of replacing the dancing girl to a taller female user.

Besides taking the target 3D body model, we demonstrate a movie reshape result by entering body parameters with a simple user input interface. In Figure 5.8, the user modified the original video sequence with a larger breast girth.

Figure 5.7: The video result of replacing the male dancing to a taller and stronger male user.

Figure 5.8: The movie reshape result by entering body sizes. The user modified the video by entering a larger bust size.

Chapter 6 Conclusion

In this dissertation, I have explored a potential of automatically recovering human body shape using a single commodity depth sensor from a single view. Three main algorithms have been developed to achieve state-of-the-art results. The first algorithm provides an automatic estimation framework for recovering body size, tracking pose via a generic trained template with levels of details. On top of the pose hallucinating framework, the second algorithm dedicates reconstructing personalized avatars from highly incomplete scans of multiple poses, achieving the first algorithm building personalized detailed human bodies and enabling robust pose tracking. In the last chapter, a novel application is developed based on a robust body and face tracking algorithm in RGB-D video for body swap and clothes re-targeting, namely Virtual Try-On.

6.1 Contributions

• We have successfully introduced Gaussian Mixture Model (GMM) into the body shape and pose inference framework, which advances future techniques in markerless tracking scenarios where finding accurate point correspondences is considered to be challenge and even harder if the human body moves quickly and the subject wears obtrusive clothes. knowledge

• We have investigated the potential to establish soft correspondences with GMM in a global optimal fashion between a generic or personal specific template and high incomplete point cloud. Our algorithm better accommodates fast and complex motion and also adapts significant body size and height change.

• We have established a fast way of scanning human full body models and measuring human body sizes by automatically registering BlendSCAPE model to high quality

single-view scans. It enables users to create and access their 3D virtual body model at home without specific prior knowledge.

• We have designed an acceleration scheme for the embedded mesh deformation of general objects, making the nonrigid deformation and registration process more interactive and responsive.

• We have deployed the GMM-BlendSCAPE model to the body swap and clothes virtual try-on applications. The novel application is able to provide customers a more natural way for future online shopping.

6.2 Future work

In the scope of highly detailed human modeling, looking into the future, we plan to increase the model resolution and speed up the inference fitting process, which ends up incorporating the detailed scanned model from our 4D portrait system to the GMM pose and shape technique. In addition, we plan to employ calibration techniques such as the bundle adjustment to increase the robustness and accuracy of the modeling process. Moreover, we intend to add more machine learning features to deal with motion ambiguities and allow the subjects to perform more free movements.

Appendix: Rigid Transformation Representation and Solver

The rigid group notation

- SO(3) - The special orthogonal group of all 3×3 rotation matrices:

  $\left\{\boldsymbol{R} \in \mathrm{SO}(3) : \boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{I}, det(\boldsymbol{R}) = 1\right\}$.
- so(3) - The Lie algebra of SO(3), consisting of all skew-symmetric 3×3 matrices:

  $\left\{\hat{\boldsymbol{\omega}} \in \mathrm{so}(3) : \hat{\boldsymbol{\omega}}^T = -\hat{\boldsymbol{\omega}}, \quad \boldsymbol{\omega} \times \boldsymbol{x} = \hat{\boldsymbol{\omega}}\boldsymbol{x}, \quad \forall \boldsymbol{\omega}, \boldsymbol{x} \in \mathbb{R}^3\right\}$.
- SE(3) - The special Euclidean group of all rigid transformations of the form:

  $\{[\boldsymbol{R} \quad \boldsymbol{t}] \in \mathrm{SE}(3) : \boldsymbol{R} \in \mathrm{SO}(3), \boldsymbol{t} \in \mathbb{R}^3\}$.
- se(3) - The Lie algebra of SE(3), consisting of all 4×4 twists:

$$\hat{\boldsymbol{\xi}} = \begin{pmatrix} \hat{\boldsymbol{\omega}} & \boldsymbol{v} \\ 0 & 0 \end{pmatrix},$$

where $\boldsymbol{\xi} = [\boldsymbol{v} \quad \boldsymbol{\omega}]^T, \boldsymbol{\omega} \in \mathbb{R}^3, \boldsymbol{v} \in \mathbb{R}^3$ are called twist coordinates.

Convert SE(3) to Lie algebra

Suppose the point $\boldsymbol{p}(t)$ rotates about an axis $\boldsymbol{\omega}$, the projection on the rotation axis is $\boldsymbol{q}(t)$, by computing the velocity, we have

$$\dot{\boldsymbol{p}}(t) = \boldsymbol{\omega} \times (\boldsymbol{p}(t) - \boldsymbol{q}(t)). \tag{1}$$

Define $\boldsymbol{v} := -\boldsymbol{\omega} \times \boldsymbol{q}$ then,

$$\begin{pmatrix} \dot{\boldsymbol{p}} \\ 0 \end{pmatrix} = \begin{pmatrix} \hat{\boldsymbol{\omega}} & \boldsymbol{v} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{p} \\ 1 \end{pmatrix} \tag{2}$$

or $\dot{\bar{\boldsymbol{p}}} = \hat{\boldsymbol{\xi}}\bar{\boldsymbol{p}}$ in homogeneous coordinates. Therefore, $\bar{\boldsymbol{p}}(t) = e^{\hat{\boldsymbol{\xi}}t}\bar{\boldsymbol{p}}(0)$ by solving the differential equation.

Suppose the rotation angle in radian is θ, $\boldsymbol{\omega}$ is a unit vector, the above analysis implies a rigid motion $[\boldsymbol{R} \quad \boldsymbol{t}] \in \mathrm{SE}(3)$ can be presented by a twist $(\boldsymbol{v} \quad \boldsymbol{\omega}\theta) \in \mathrm{se}(3)$ as,

$$\begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} e^{\hat{\boldsymbol{\omega}}\theta} & (\boldsymbol{I} - e^{\hat{\boldsymbol{\omega}}\theta})(\boldsymbol{\omega} \times \boldsymbol{v}) + \boldsymbol{\omega}\boldsymbol{\omega}^T\boldsymbol{v}\theta \\ 0 & 1 \end{pmatrix}. \tag{3}$$

Note that the exponetial map from the Lie algebra se(3) to the group SE(3) is surjective.

Dual quaternion representation

While twists representation are mostly used in mechanics, robots and motion analysis, dual quaternion representation are more widely applied in computer graphics. In essential, a unit quaternion is equivalent to an Euler axis/angle in so(3) and a dual quaternion is equivalent to a twist in se(3).

A rotation $e^{\hat{\boldsymbol{\omega}}\theta}$ around a unit axis $\boldsymbol{\omega}$ of a counterclockwise radian θ can be represented by a quaternion:

$$\boldsymbol{q} = \left[\boldsymbol{\omega}\sin\frac{\theta}{2},\ \cos\frac{\theta}{2}\right]^T =: [\boldsymbol{q_v}\ q_s]^T. \tag{4}$$

Define two matrix functions of the quaternion as:

$$\boldsymbol{P}(\boldsymbol{q}) := \begin{bmatrix} q_s\boldsymbol{I} + \hat{\boldsymbol{q}}_{\boldsymbol{v}} & \boldsymbol{q_v} \\ -\boldsymbol{q}_{\boldsymbol{v}}^T & q_s \end{bmatrix}; \qquad \boldsymbol{Q}(\boldsymbol{q}) := \begin{bmatrix} q_s\boldsymbol{I} - \hat{\boldsymbol{q}}_{\boldsymbol{v}} & \boldsymbol{q_v} \\ -\boldsymbol{q}_{\boldsymbol{v}}^T & q_s \end{bmatrix}. \tag{5}$$

It is easy to show that $\boldsymbol{P}(\boldsymbol{a})\boldsymbol{b} = \boldsymbol{Q}(\boldsymbol{b})\boldsymbol{a}, \forall \boldsymbol{a}, \boldsymbol{b}$ quaternions and

$$\boldsymbol{Q}(\boldsymbol{q})^T \boldsymbol{P}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \quad \text{if } \|\boldsymbol{q}\|^2 = 1, \tag{6}$$

or explicitly,

$$\boldsymbol{R} = (q_s^2 - \boldsymbol{q}_{\boldsymbol{v}}^T \boldsymbol{q}_{\boldsymbol{v}})\boldsymbol{I} + 2\boldsymbol{q}_{\boldsymbol{v}}\boldsymbol{q}_{\boldsymbol{v}}^T + 2q_s\hat{\boldsymbol{q}}_{\boldsymbol{v}}. \tag{7}$$

Analogous to the twist representation, by using dual quaternions $\boldsymbol{p}, \boldsymbol{q}$, a translation vector $\boldsymbol{t} \in \mathbb{R}^3$ can be defined by

$$\begin{bmatrix} \boldsymbol{t} \\ 0 \end{bmatrix} := \boldsymbol{Q}(\boldsymbol{q})^T \boldsymbol{p} = \begin{bmatrix} q_s\boldsymbol{I} + \hat{\boldsymbol{q}}_{\boldsymbol{v}} & -\boldsymbol{q}_{\boldsymbol{v}} \\ \boldsymbol{q}_{\boldsymbol{v}}^T & q_s \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_{\boldsymbol{v}} \\ p_s \end{bmatrix} = \begin{bmatrix} q_s\boldsymbol{p}_{\boldsymbol{v}} - p_s\boldsymbol{q}_{\boldsymbol{v}} + \boldsymbol{q}_{\boldsymbol{v}} \times \boldsymbol{p}_{\boldsymbol{v}} \\ \boldsymbol{q}^T\boldsymbol{p} \end{bmatrix}. \tag{8}$$

Therefore an arbitrary pair of quaternions $(\boldsymbol{q}, \boldsymbol{p})$ can represent a rigid transformation $[\boldsymbol{R}, \quad \boldsymbol{t}]$, which a six degrees of freedom, if and only if they satisfy two constraints: $\boldsymbol{q}^T\boldsymbol{q} = 1$ and $\boldsymbol{q}^T\boldsymbol{p} = 0$.

Solve rigid transformation by quaternions

Suppose there are two 3D point sets to be aligned: $\boldsymbol{X} := \{\boldsymbol{x}_i \in \mathbb{R}^3\}$ and $\boldsymbol{Y} := \{\boldsymbol{y}_i \in \mathbb{R}^3\}$, associating with a point-wise positive weighting factor w_i, $i = 1, 2, \ldots, N$. To estimate the rigid transformation $[\boldsymbol{R} \quad \boldsymbol{t}] \in \mathrm{SE}(3)$ from $\boldsymbol{X}$ to $\boldsymbol{Y}$ is to minimize the following function

$$\min_{\boldsymbol{R},\boldsymbol{t}} \sum_{i=1}^{N} w_i \|\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} - \boldsymbol{y}_i\|^2. \tag{9}$$

In homogeneous coordinates and quaternion representation, we have

$$\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} = \boldsymbol{Q}(\boldsymbol{q})^T \boldsymbol{P}(\boldsymbol{q})\boldsymbol{x}_i + \boldsymbol{Q}(\boldsymbol{q})^T \boldsymbol{p}. \tag{10}$$

Therefore, minimizing the objective function (9) can be converted into a constrained quadratic minimization of $\boldsymbol{q}$ and $\boldsymbol{p}$

$$\begin{aligned} \min_{\boldsymbol{q},\boldsymbol{p}} \quad & \boldsymbol{q}^T\boldsymbol{C}_1\boldsymbol{q} + w\boldsymbol{p}^T\boldsymbol{p} + \boldsymbol{p}^T\boldsymbol{C}_2\boldsymbol{q} + const. \\ \text{subject to} \quad & \boldsymbol{q}^T\boldsymbol{q} = 1, \quad \boldsymbol{q}^T\boldsymbol{p} = 0, \end{aligned} \tag{11}$$

where

$$\begin{aligned} \boldsymbol{C}_1 &= -2\sum_{i=1}^{N} w_i \boldsymbol{P}(\boldsymbol{y}_i)^T \boldsymbol{Q}(\boldsymbol{x}_i), \\ \boldsymbol{C}_2 &= -2\sum_{i=1}^{N} w_i \left[\boldsymbol{Q}(\boldsymbol{x}_i) - \boldsymbol{P}(\boldsymbol{y}_i)\right], \\ w &= \sum_{i=1}^{N} w_i, \\ const. &= \sum_{i=1}^{N} w_i(\boldsymbol{x}_i^T\boldsymbol{x}_i + \boldsymbol{y}_i^T\boldsymbol{y}_i). \end{aligned} \tag{12}$$

The constrained least square problem can be solved by Lagrange multipliers,

$$\min_{\boldsymbol{q},\boldsymbol{p}} \ \boldsymbol{q}^T\boldsymbol{C}_1\boldsymbol{q} + w\boldsymbol{p}^T\boldsymbol{p} + \boldsymbol{p}^T\boldsymbol{C}_2\boldsymbol{q} + const. + \lambda_1(\boldsymbol{q}^T\boldsymbol{q} - 1) + \lambda_2\boldsymbol{q}^T\boldsymbol{p} \tag{13}$$

Taking the partial derivatives and let them be zeros gives,

$$(\boldsymbol{C}_1 + \boldsymbol{C}_1^T)\boldsymbol{q} + \boldsymbol{C}_2^T\boldsymbol{p} + 2\lambda_1\boldsymbol{q} + \lambda_2\boldsymbol{p} = 0, \tag{14}$$

$$2w\boldsymbol{p} + \boldsymbol{C}_2\boldsymbol{q} + \lambda_2\boldsymbol{q} = 0. \tag{15}$$

To combine above equations we can easily get that: $\boldsymbol{q}$ is the eigenvector corresponding to the largest eigenvalue of the matrix $\frac{1}{2}\left[\frac{1}{2w}\boldsymbol{C}_2^T\boldsymbol{C}_2 - \boldsymbol{C}_1 - \boldsymbol{C}_1^T\right]$, and then $\boldsymbol{p}$ is computed by $\boldsymbol{p} = -\frac{1}{2w}\boldsymbol{C}_2\boldsymbol{q}$.

Bibliography

[1] Hao Li, Etienne Vouga, Anton Gudym, Linjie Luo, Jonathan T. Barron, and Gleb Gusev. 3d self-portraits. *SIGGRAPH Asia*, 32(6), November 2013.

[2] Point Grey Inc. http://www.ptgrey.com.

[3] MESA Imaging Inc. http://www.mesa-imaging.ch.

[4] Microsoft Inc. https://www.microsoft.com/en-us/kinectforwindows.

[5] M. Pollefeys, R. Koch, and L. Van Gool. Self-Calibration and Metric Reconstruction in spite of Varying and Unknown Internal Camera Parameters. In *ICCV*, 1998.

[6] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. *ACM Symposium on User Interface Software and Technology (UIST)*, pages 559–568, 2011.

[7] i.materialise Inc., 2015. http://i.materialise.com.

[8] T. Kanade, P. Rander, S. Vedula, and H. Saito. Virtualized reality: Digitizing a 3d time-varying event as is and in real time. In *Mixed Reality, Merging Real and Virtual Worlds*, pages 41–57. 1999.

[9] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Transactions on Graphics,*, 23(3):600–608, 2004.

[10] Michael Goesele, Noah Snavely, Brian Curless, Hugues Hoppe, and Steven M. Seitz. Multi-View Stereo for Community Photo Collections. In *ICCV*, 2007.

[11] Cyberware Inc., 2014. http://cyberware.com.

[12] Vicon Inc. http://www.vicon.com.

[13] Civilian American and European Surface Anthropometry Resource Project-CAESAR. http://store.sae.org/caesar.

[14] Lorenzo Torresani, Aaron Hertzmann, and Chris Bregler. Learning non-rigid 3d shape from 2d motion. In *In proceedings of NIPS*, 2003.

[15] Will Chang and Matthias Zwicker. Automatic registration for articulated shapes. *Comput. Graph. Forum*, 27(5):1459–1468, 2008.

[16] Will Chang and Matthias Zwicker. Range scan registration using reduced deformable models. *Comput. Graph. Forum*, 28(2):447–456, 2009.

[17] Qi-Xing Huang, Bart Adams, Martin Wicke, and Leonidas J. Guibas. Non-rigid registration under isometric deformations. *Comput. Graph. Forum*, 27(5):1449–1457, 2008.

[18] Niloy J. Mitra, Simon Flory, Maks Ovsjanikov, Natasha Gelfand, Leonidas Guibas, and Helmut Pottmann. Dynamic geometry registration. In *Eurographics Symposium on Geometry Processing*, 2007.

[19] Andrei Sharf, Dan A. Alcantara, Thomas Lewiner, Chen Greif, and Alla Sheffer. Space-time surface reconstruction using incompressible flow. In *Siggraph Asia*. ACM, 2008.

[20] Michael Wand, Philipp Jenke, Qixing Huang, Martin Bokeloh, Leonidas Guibas, and Andreas Schilling. Reconstruction of deforming geometry from time-varying point clouds. In *Eurographics Symposium on Geometry Processing*, 2007.

[21] Michael Wand, Bart Adams, Maksim Ovsjanikov, Alexander Berner, Martin Bokeloh, Philipp Jenke, Leonidas Guibas, Hans-Peter Seidel, and Andreas Schilling. Efficient reconstruction of nonrigid shape and motion from real-time 3d scanner data. *ACM Transactions on Graphics*, 28(2), 2009.

[22] D. Anguelov, D. Koller, H. Pang, P. Srinivasan, and S. Thrun. Recovering articulated object models from 3d range data. In *Proceedings of UAI*, 2004.

[23] Alexander M. Bronstein, Michael M. Bronstein, Alfred M. Bruckstein, and Ron Kimmel. Matching two-dimensional articulated shapes using generalized multidimensional scaling. In *AMDO*, pages 48–57, 2006.

[24] Sang Il Park and Jessica K. Hodgins. Capturing and animating skin deformation in human motion. *ACM Trans. Graph.*, 25(3):881–889, 2006.

[25] Sang Il Park and Jessica K. Hodgins. Data-driven modeling of skin and muscle deformation. *ACM Trans. Graph.*, 27(3), 2008.

[26] Hao Li, Bart Adams, Leonidas J. Guibas, and Mark Pauly. Robust single-view geometry and motion reconstruction. *ACM Trans. Graph.*, 28(5), 2009.

[27] Hao Li, Linjie Luo, Daniel Vlasic, Pieter Peers, Jovan Popovic, Mark Pauly, and Szymon Rusinkiewicz. Temporally coherent completion of dynamic shapes. *ACM Trans. Graph.*, 31(1):2, 2012.

[28] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, pages 127–136, 2011.

[29] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *SIGGRAPH '98: Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 43–54, New York, NY, USA, 1998. ACM.

[30] Peng Guan, Loretta Reiss, David A. Hirshberg, Alexander Weiss, and Michael J. Black. DRAPE: dressing any person. *ACM Trans. Graph.*, 31(4):35, 2012.

[31] Yasutaka Furukawa and Jean Ponce. Dense 3d motion capture from synchronized video streams. In *Computer Vision and Pattern Recognition*. IEEE, 2008.

[32] Jamie Shotton, Toby Sharp, Alex Kipman, Andrew W. Fitzgibbon, Mark Finocchio, Andrew Blake, Mat Cook, and Richard Moore. Real-time human pose recognition in parts from single depth images. *Commun. ACM*, 56(1):116–124, 2013.

[33] Matthew Loper, Naureen Mahmood, and Michael J. Black. Mosh: motion and shape capture from sparse markers. *ACM Trans. Graph.*, 33(6):220, 2014.

[34] Thomas B. Moeslund, Adrian Hilton, and Volker Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer Vision and Image Understanding*, 104(2-3):90–126, 2006.

[35] Ronald Poppe. Vision-based human motion analysis: An overview. *Computer Vision and Image Understanding*, 108(1-2):4–18, 2007.

[36] Robert Y. Wang and Jovan Popovic. Real-time hand-tracking with a color glove. *ACM Trans. Graph.*, 28(3), 2009.

[37] Raquel Urtasun and Trevor Darrell. Sparse probabilistic regression for activity-independent human pose inference. In *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2008), 24-26 June 2008, Anchorage, Alaska, USA*, 2008.

[38] Christoph Bregler and Jitendra Malik. Tracking people with twists and exponential maps. In *1998 Conference on Computer Vision and Pattern Recognition (CVPR '98), June 23-25, 1998, Santa Barbara, CA, USA*, pages 8–15, 1998.

[39] Daniel Grest, Jan Woetzel, and Reinhard Koch. Nonlinear body pose estimation from depth images. In *Pattern Recognition, 27th DAGM Symposium, Vienna, Austria, August 31 - September 2, 2005, Proceedings*, pages 285–292, 2005.

[40] Youding Zhu and Kikuo Fujimura. Constrained optimization for human pose estimation from depth sequences. In *Computer Vision - ACCV 2007, 8th Asian Conference on Computer Vision, Tokyo, Japan, November 18-22, 2007, Proceedings, Part I*, pages 408–418, 2007.

[41] Christian Plagemann, Varun Ganapathi, Daphne Koller, and Sebastian Thrun. Real-time identification and localization of body parts from depth images. In *IEEE International Conference on Robotics and Automation, ICRA 2010, Anchorage, Alaska, USA, 3-7 May 2010*, pages 3108–3113, 2010.

[42] Deva Ramanan and David A. Forsyth. Finding and tracking people from the bottom up. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2003), 16-22 June 2003, Madison, WI, USA*, pages 467–474, 2003.

[43] Lubomir D. Bourdev and Jitendra Malik. Poselets: Body part detectors trained using 3d human pose annotations. In *IEEE 12th International Conference on Computer Vision, ICCV 2009, Kyoto, Japan, September 27 - October 4, 2009*, pages 1365–1372, 2009.

[44] Zhuowen Tu and Xiang Bai. Auto-context and its application to high-level vision tasks and 3d brain image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(10):1744–1757, 2010.

[45] Jonathan Deutscher and Ian D. Reid. Articulated body motion capture by stochastic search. *International Journal of Computer Vision*, 61(2):185–205, 2005.

[46] Hedvig Sidenbladh, Michael J. Black, and David J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *Computer Vision - ECCV 2000, 6th European Conference on Computer Vision, Dublin, Ireland, June 26 - July 1, 2000, Proceedings, Part II*, pages 702–718, 2000.

[47] Ralf Plänkers and Pascal Fua. Articulated soft objects for multiview shape and motion capture. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(9):1182–1187, 2003.

[48] Carsten Stoll, Nils Hasler, Juergen Gall, Hans-Peter Seidel, and Christian Theobalt. Fast articulated motion tracking using a sums of gaussians body model. In *IEEE International Conference on Computer Vision, ICCV 2011, Barcelona, Spain, November 6-13, 2011*, pages 951–958, 2011.

[49] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. Scape: shape completion and animation of people. In *SIGGRAPH*, pages 408–416, 2005.

[50] David A. Hirshberg, Matthew Loper, Eric Rachlin, and Michael J. Black. Coregistration: Simultaneous alignment and modeling of articulated 3d shape. In *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part VI*, pages 242–255, 2012.

[51] Silvia Zuffi and Michael J. Black. The stitched puppet: A graphical model of 3D human shape and pose. In *CVPR*, June 2015.

[52] Yinpeng Chen, Zicheng Liu, and Zhengyou Zhang. Tensor-based human body modeling. In *CVPR*, pages 105–112, 2013.

[53] A. Weiss, D. Hirshberg, and M.J. Black. Home 3d body scans from noisy image and range data. In *ICCV*, pages 1951–1958, 2011.

[54] Alexandru O. Balan, Leonid Sigal, Michael J. Black, James E. Davis, and Horst W. Haussecker. Detailed human shape and pose from images. In *CVPR*, 2007.

[55] AlexandruO. Blan and MichaelJ. Black. The naked truth: Estimating body shape under clothing. In David Forsyth, Philip Torr, and Andrew Zisserman, editors, *ECCV*, pages 15–29. 2008.

[56] Federica Bogo, Javier Romero, Matthew Loper, and Michael J. Black. FAUST: dataset and evaluation for 3d mesh registration. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 3794–3801, 2014.

[57] Matthias Straka, Stefan Hauswiesner, Matthias Rüther, and Horst Bischof. Simultaneous shape and pose adaption of articulated models using linear optimization. In *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part I*, pages 724–737, 2012.

[58] Ilya Baran and Jovan Popovic. Automatic rigging and animation of 3d characters. *ACM Trans. Graph.*, 26(3):72, 2007.

[59] Doug L. James and Christopher D. Twigg. Skinning mesh animations. *ACM Trans. Graph.*, 24(3):399–407, 2005.

[60] Binh Huy Le and Zhigang Deng. Robust and accurate skeletal rigging from mesh sequences. *ACM Trans. Graph.*, 33(4):84, 2014.

[61] Miao Liao, Qing Zhang, Huamin Wang, Ruigang Yang, and Minglun Gong. Modeling deformable objects from a single depth camera. In *ICCV*, pages 167–174, 2009.

[62] Mao Ye and Ruigang Yang. Real-time simultaneous pose and shape estimation for articulated objects using a single depth camera. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 2353–2360, 2014.

[63] Thomas Helten, Andreas Baak, Gaurav Bharaj, Meinard Müller, Hans-Peter Seidel, and Christian Theobalt. Personalization and evaluation of a real-time depth-based full body tracker. In *2013 International Conference on 3D Vision, 3DV 2013, Seattle, Washington, USA, June 29 - July 1, 2013*, pages 279–286, 2013.

[64] Andreas Baak, Meinard Müller, Gaurav Bharaj, Hans-Peter Seidel, and Christian Theobalt. A data-driven approach for real-time full body pose reconstruction from a depth camera. In *Consumer Depth Cameras for Computer Vision, Research Topics and Applications*, pages 71–98. 2013.

[65] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 559–568. ACM, 2011.

[66] Thibaut Weise, Thomas Wismer, Bastian Leibe, and Luc J. Van Gool. Online loop closure for real-time interactive 3d scanning. *Computer Vision and Image Understanding*, 115(5):635–648, 2011.

[67] C. Tomasi and T. Kanade. Shape and Motion from Image Streams under Orthography: A Factorization Approach. *IJCV*, 9(2):137–154, 1992.

[68] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000.

[69] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.

[70] Jing Tong, Jin Zhou, Ligang Liu, Zhigeng Pan, and Hao Yan. Scanning 3d full human bodies using kinects. *TVCG*, 18(4):643–650, 2012.

[71] Brett Allen, Brian Curless, and Zoran Popovic. The space of human body shapes: reconstruction and parameterization from range scans. *ACM Trans. Graph.*, 22(3):587–594, 2003.

[72] Robert W. Sumner, Johannes Schmid, and Mark Pauly. Embedded deformation for shape manipulation. *ACM Trans. Graph.*, 26(3), July 2007.

[73] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. In *Transcation on visualization and Computer Graphics*. IEEE, 2008.

[74] Yuri Pekelny and Craig Gotsman. Articulated object reconstruction and markerless motion capture from depth video. In *Eurographics*, 2008.

[75] V. Blanz and T. Vetter. A morphable model for the synthesis of 3d faces. In *Proceedings of SIGGRAPH*, 1999.

[76] Christoph Bregler, Aaron Hertzmann, and Henning Biermann. Recovering non-rigid 3d shape from image streams. In *CVPR*, 2000.

[77] Lorenzo Torresani, Aaron Hertzmann, and Christoph Bregler. Non-rigid structure-from-motion: Estimating shape and motion with hierarchical priors. *IEEE PAMI*, To appear.

[78] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *Proceedings of CVPR*, pages 519–526, 2006.

[79] M. Goesele, B. Curless, and S. Seitz. Multi-view stereo revisited. In *Proceedings of CVPR*, 2006.

[80] A. Laurentini. The Visual Hull Concept for Silhouette Based Image Understanding. *IEEE PAMI*, 16(2):150–162, February 1994.

[81] W. Matusik, C. Buehler, R. Raskar, S. Gortler, and L. McMillan. Image-Based Visual Hulls. In *Proceedings of SIGGRAPH 2000*, 2000.

[82] G.K.M. Cheung, T. Kanade, J.-Y. Bouguet, and M. Holler. A real time system for robust 3d voxel reconstruction of human motions. In *CVPR*, 2000.

[83] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, 1996.

[84] A. Sharf, M. Alexa, and D. Cohen-Or. Context-based surface completion. *ACM Transactions on Graphics*, 23(2):878–887, 2004.

[85] Tao Ju. Robust repair of polygonal models. *ACM Transactions on Graphics*, 23(3):888–895, 2004.

[86] S. Park, X. Guo, H. Shin, and H. Qin. Shape and appearance repair for incomplete point surfaces. In *Proceedings of ICCV*, pages 1260–1267, 2005.

[87] Hao Li, Bart Adams, Leonidas J. Guibas, and Mark Pauly. Robust single-view geometry and motion reconstruction. In *SIGGRAPH Asia*, pages 175:1–175:10, 2009.

[88] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rossl, and Han-Peter Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004.

[89] Li Zhang, Noah Snavely, Brian Curless, and Steven M. Seitz. Spacetime faces: High-resolution capture for modeling and animation. In *ACM Annual Conference on Computer Graphics*, pages 548–558, August 2004.

[90] Will Chang and Matthias Zwicker. Global registration of dynamic range scans for articulated model reconstruction. *ACM Trans. Graph*, 30(3), 2011.

[91] G.C. Sharp, S.-W. Lee, and D.K. Wehe. Multiview registration of 3d scenes by minimizing error between coordinate frames. *PAMI*, 26(8):1037–1050, 2004.

[92] Michael Kazhdan and Hugues Hoppe. Screened poisson surface reconstruction. *ACM Trans. Graph.*, 32(3):29:1–29:13, July 2013.

[93] Ming Chuang, Linjie Luo, Benedict J. Brown, Szymon Rusinkiewicz, and Michael Kazhdan. Estimating the Laplace-Beltrami operator by restricting 3d functions. *Symposium on Geometry Processing*, July 2009.

[94] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing, Barcelona, Spain, July 4-6, 2007*, pages 109–116, 2007.

[95] Nadia Magnenat-Thalmann, Richard Laperrire, Daniel Thalmann, and Universit De Montral. Joint-dependent local deformations for hand animation and object grasping. pages 26–33, 1988.

[96] Alec Jacobson, Ilya Baran, Jovan Popovic, and Olga Sorkine. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.*, 30(4):78, 2011.

[97] Tao Ju, Scott Schaefer, and Joe D. Warren. Mean value coordinates for closed triangular meshes. *ACM Trans. Graph.*, 24(3):561–566, 2005.

[98] Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic coordinates for character articulation. *ACM Trans. Graph.*, 26(3):71, 2007.

[99] Yaron Lipman, David Levin, and Daniel Cohen-Or. Green coordinates. *ACM Trans. Graph.*, 27(3), 2008.

[100] Ofir Weber, Olga Sorkine, Yaron Lipman, and Craig Gotsman. Context-aware skeletal shape deformation. *Comput. Graph. Forum*, 26(3):265–274, 2007.

[101] Karan Singh and Evangelos Kokkevis. Skinning characters using surface oriented free-form deformations. In *Proceedings of the Graphics Interface 2000 Conference, May 15-17, 2000, Montréal, Québec, Canada*, pages 35–42, 2000.

[102] Xiaohan Shi, Kun Zhou, Yiying Tong, Mathieu Desbrun, Hujun Bao, and Baining Guo. Mesh puppetry: cascading optimization of mesh deformation with inverse kinematics. *ACM Trans. Graph.*, 26(3):81, 2007.

[103] Robert W. Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popovic. Mesh-based inverse kinematics. *ACM Trans. Graph.*, 24(3):488–495, 2005.

[104] Kevin G. Der, Robert W. Sumner, and Jovan Popovic. Inverse kinematics for reduced deformable models. *ACM Trans. Graph.*, 25(3):1174–1179, 2006.

[105] Mario Botsch and Olga Sorkine. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):213–230, 2008.

[106] Andriy Myronenko and Xubo B. Song. Point set registration: Coherent point drift. *TPAMI*, 32(12):2262–2275, 2010.

[107] Yan Cui, Will Chang, Tobias Nöll, and Didier Stricker. Kinectavatar: Fully automatic body capture using a single kinect. In *ACCV Workshops (2)*, pages 133–147, 2012.

[108] Nianchen Deng Xubo Yang Mao Ye, Huamin Wang and Ruigang Yang. Real-time human pose and shape estimation for virtual try-on using a single commodity depth camera. In *n IEEE Transactions on Visualization and Computer Graphics (IEEE Virtual Reality) 2014 Apr;20(4):550-9*.

[109] Zhenglong Zhou, Bo Shu, Shaojie Zhuo, Xiaoming Deng, Ping Tan, and Stephen Lin. Image-based Clothes Animation for Virtual Fitting. In *ACM SIGGRAPH Asia, Technical Briefs*, 2012.

[110] http://www.trimirror.com. trimirror, 2015.

[111] http://www.fitnect.hu. Fitnect, 2015.

[112] Arjun Jain, Thorsten Thormählen, Hans-Peter Seidel, and Christian Theobalt. Moviereshape: tracking and reshaping of humans in videos. *ACM Trans. Graph.*, 29(6):148, 2010.

[113] S. Socrate M.J. King, P. Jearanaisilawong. A continuum constitutive model for the mechanical behavior of woven fabrics. *International Journal of Solids and Structures*, (42):3867, 2005.

[114] Reddyb D. Bowlesc H. C. Bezuidenhoutd D. Zillad P. Yeomana, M. S. and T Franz. A constitutive model for the warp-weft coupled non-linear behavior of knitted biomedical textiles. *Biomaterials*, 31:8484C8493, November 2010.

[115] N. Zhou and T. Ghosh. On-line measurement of fabric bending behavior. *Textile Research Journal*, 7:533C542, July 1998.

[116] Huamin Wang, James F. O'Brien, and Ravi Ramamoorthi. Data-driven elastic models for cloth: modeling and measurement. *ACM Trans. Graph.*, 30(4):71, 2011.

[117] T JAKOBSEN. Advanced character physics. *Game Developers Conference*, pages 383–401, 2001.

[118] MAHER M. BORDES, J. and M. SECHREST. Nvidia apex: High definition physics with clothing and vegetation. In *Game Developers Conference*, 2009.

[119] Stefan Hauswiesner, Matthias Straka, and Gerhard Reitmayr. Temporal coherence in image-based visual hull rendering. *TVCG*, 19(10):1758–1767, 2013.

[120] Jun Ehara and Hideo Saito. Texture overlay for virtual clothing based on PCA of silhouettes. In *Fifth IEEE/ACM International Symposium on Mixed and Augmented Reality, ISMAR 2006, October 22-25, 2006, Santa Barbara, CA, USA*, pages 139–142, 2006.

[121] Hiroshi Tanaka and Hideo Saito. Texture overlay onto flexible object with PCA of silhouettes and k-means method for search into database. In *Proceedings of the IAPR Conference on Machine Vision Applications (IAPR MVA 2009), Keio University, Yokohama, Japan, May 20-22, 2009*, pages 5–8, 2009.

[122] Stefan Hauswiesner, Matthias Straka, and Gerhard Reitmayr. Virtual try-on through image-based rendering. *TVCG*, 19(9):1552–1565, 2013.

[123] Model my diet. http://www.modelmydiet.com, 2015.

[124] http://fits.me/. Fits.me, 2012.

[125] http://www.styku.com/. Styku, 2015.

[126] Kevin Dale, Kalyan Sunkavalli, Micah K Johnson, Daniel Vlasic, Wojciech Matusik, and Hanspeter Pfister. Video face replacement. *ACM Transactions on Graphics (TOG)*, 30(6):130, 2011.

[127] YiChang Shih, Sylvain Paris, Connelly Barnes, William T Freeman, and Frédo Durand. Style transfer for headshot portraits. *ACM Transactions on Graphics (TOG)*, 33(4):148, 2014.

[128] Nicolas Bonneel, Kalyan Sunkavalli, Sylvain Paris, and Hanspeter Pfister. Example-based video color grading. *ACM Trans. Graph.*, 32(4):39, 2013.

[129] J. Gall, A. Fossati, and L. Van Gool. Functional categorization of objects using real-time markerless motion capture. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1969–1976, 2011.

[130] Shaoqing Ren, Xudong Cao, Yichen Wei, and Jian Sun. Face alignment at 3000 fps via regressing local binary features. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1685–1692. IEEE, 2014.

[131] Kalyan Sunkavalli, Micah K Johnson, Wojciech Matusik, and Hanspeter Pfister. Multi-scale image harmonization. In *ACM Transactions on Graphics (TOG)*, volume 29, page 125. ACM, 2010.

[132] Feng Liu, Michael Gleicher, Hailin Jin, and Aseem Agarwala. Content-preserving warps for 3d video stabilization. *ACM Trans. Graph.*, 28(3), 2009.

[133] Scott Schaefer, Travis McPhail, and Joe Warren. Image deformation using moving least squares. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 533–540. ACM, 2006.

Vita

EDUCATION

B.S. in Computer Software, Tsinghua University, 2006
M.S. in Mathematics, University of Kentucky, 2010

INTERNSHIP

Communication and Collaboration Systems, Microsoft Research, 2008
Visual Computing, Microsoft Research Asia, 2010

PUBLICATION

(1) "(Re)Constructing Antiquity: 3D Modeling and Cypriot Votive Sculpture from Athienou-Malloura", Erin Averett, Derek Counts, Kevin Garstki, Adam Whidden, **Qing Zhang**, Bo Fu, Brent Seales, Ruigang Yang, Caitlyn Ewers, Michael Toumazou, Archaeological Institute of America 116th Annual Meeting, 2015

(2) "Quality Dynamic Human Body Modeling Using a Single Low-cost Depth Camera", ***Qing Zhang****, Bo Fu, Mao Ye and Ruigang Yang,* IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014

(3) "A Survey on Human Motion Analysis from Depth Data", Mao Ye, ***Qing Zhang****, Liang Wang, Jiejie Zhu, Ruigang Yang and Juergen Gall,* In Time-of-Flight and Depth Imaging. Sensors, Algorithms, and Applications, Lecture Notes in Computer Science, Volume 8200, 2013, pp 149-187

(4) "Simulation Guided Hair Dynamics Modeling from Video", ***Qing Zhang****, Jing Tong, Huamin Wang, Ruigang Yang and Zhigeng Pan,* IEEE Pacific Graphics, 2012

(5) "Edge-Preserving Photometric Stereo via Depth Fusion", ***Qing Zhang****, Mao Ye, Ruigang Yang, Yasuyuki Matsushita, Bennett Wilburn and Huimin Yu,* IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2012

(6) "Real Time Head Pose Tracking from Multple Cameras with a Generic Model", *Qin Cai, Aswin Sankaranarayanan,* ***Qing Zhang****, Zhengyou Zhang and Zicheng Liu,* IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR) Workshop on Analysis and Modeling of Faces and Gestures (best paper award), 2010

(7) "A Volumetric Approach for Merging Range Image of Semi-Rigid Objects Captured at Different Time Instances", *Miao Liao,* ***Qing Zhang****, Ruigang Yang and Minglun Gong,* International Symposium on 3D Data Processing, Visualization and Transmission (3DPVT), 2010

(8) "Modeling Deformable Objects from a Single Depth Camera", *Miao Liao,* ***Qing Zhang****, Huaming Wang, Ruigang Yang and Minglun Gong,* IEEE International Conference on Computer Vision (ICCV) oral, 2009

(9) "Physically Guided Liquid Surface Modeling from Videos", *Huamin Wang, Miao Liao,* ***Qing Zhang****, Ruigang Yang and Greg Turk,* In Proceedings of ACM SIGGRAPH (ACM Transactions on Graphics), 2009

(10) "Endoscopic Video Texture Mapping on Pre-Built 3D anatomical Objects Without Camera Tracking", *Xianwang Wang,* ***Qing Zhang****, Qiong Han, Ruigang Yang, Melody Carswell, Brent Seales and Erica Sutton,* IEEE Transaction on Medical Imaging (TMI), 2009

(11) "Feature-based Texture Mapping from Video Sequence", *Xianwang Wang,* ***Qing Zhang*** *and Ruigang Yang,* ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games (I3D), 2007