Interaction Capture and Synthesis of Human Hands

by

Paul G. Kry

B.Math., University of Waterloo, 1997

M.Sc., University of British Columbia, 2000

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE STUDIES

(Computer Science)

The University of British Columbia

December 2005

Abstract

This thesis addresses several issues in modelling interaction with human hands in computer graphics and animation. Modifying motion capture to satisfy the constraints of new animation is difficult when contact is involved because physical interaction involves energy or power transfer between the system of interest and the environment, and is a critical problem for computer animation of hands. Although contact force measurements provide a means of monitoring this transfer, motion capture as currently used for creating animation has largely ignored contact forces. We present a system of capturing synchronized motion and contact forces, called *interaction capture*. We transform interactions such as grasping into joint compliances and a nominal reference trajectory in an approach inspired by the equilibrium point hypothesis of human motor control. New interactions are synthesized through simulation of a quasi-static compliant articulated model in a dynamic environment that includes friction. This uses a novel position-based linear complementarity problem formulation that includes friction, breaking contact, and coupled compliance between contacts at different fingers. We present methods for reliable interaction capture, addressing calibration, force estimation, and synchronization. Additionally, although joint compliances are traditionally estimated with perturbation-based methods, we introduce a technique that instead produces estimates without perturbation. We validate our results with data from previous work and our own perturbation-based estimates. A complementary goal of this work is hand-based interaction in virtual environments. We present techniques for whole-hand interaction using the Tango, a novel sensor that performs interaction capture by measuring pressure images and accelerations. We approximate grasp hand-shapes from previously observed data through rotationally invariant comparison of pressure measurements. We also introduce methods involving heuristics and thresholds that make reliable drift-free navigation possible with the Tango. Lastly, rendering the skin deformations of articulated characters is a fundamental problem for computer animation of hands. We present a deformation model, called EigenSkin, which provides a means of rendering physically- or example-based deformation models at interactive rates on graphics hardware.

Contents

List of Tables

List of Figures

Acknowledgements

I would like to thank Dr. Dinesh Pai for his guidance, kindness, and patience during this work. I feel very fortunate to have been mentored by someone with such keen insight. Without his encouragement and support, this thesis would never have been possible. I would also like to thank Dr. Doug James for getting me started; it was a pleasure to work with you on EigenSkin. To those at Rutgers (Danny, Nick, Shin, Suburna, and Tim) I thank you all for your help during my visit. Likewise, I am very grateful to KangKang and Andrea at UBC for making prints and deliveries for me while I was so far away. Finally, thanks to my family for being there when I was in need, and a special thank you to Anthony Santoro for making my time New Jersey such a pleasant experience.

Paul G. Kry

The University of British Columbia
December 2005

In loving memory of Patricia and Ralph Bowers

Chapter 1

Introduction

Computer animation of hand motion is difficult because of the complexities of human motor control and biomechanics, but also because every human is an expert in human motion; we are very familiar with how our hands move since we use them almost constantly on a daily basis. Because motion capture directly records the subtleties of human motion, it has become popular for computer animation. With ease, one can place motion capture markers on humans, capture their motion, and in turn create compelling animation from the capture data. In comparison, the alternative of using artists and algorithms to produce convincing motion is considerably more time consuming and difficult.

The major challenge in motion capture research is modifying a motion to satisfy the constraints of new animation [Hodgins and Pollard 1997; Gleicher 1998; Popovic and Witkin 1999]. Motion transformation becomes difficult when contact is involved because physical interaction involves energy or power transfer between the system of interest and the environment. Contact is central to grasping and manipulation. As such, hands present an interesting challenge due to their highly articulated kinematics, and the numerous contacts they exhibit during interaction. In a physically coupled system, for instance, a hand and object, the contact forces are key. They provide a means for us to understand the behaviour and stability of the system. This concept forms the base for the main contribution of this thesis.

This thesis addresses several different issues in modelling interaction with human hands in computer graphics and animation. Figure 1.1 shows the spectrum of issues addressed. The primary goal of this thesis, however, is the development of a technique for capturing and resynthesizing interaction, specifically that of hand interactions such as grasping. In our method, we capture both motion and contact forces, and we call this *interaction capture*. Then, rather than using either the motion or force directly, we instead estimate an intermediate representation that describes the active control (a reference joint angle trajectory) and passive behaviour of the captured subject (compliance of the joints). It is from this intermediate representation that we synthesize new interaction, through simulation of a quasi-static compliant kinematic model,

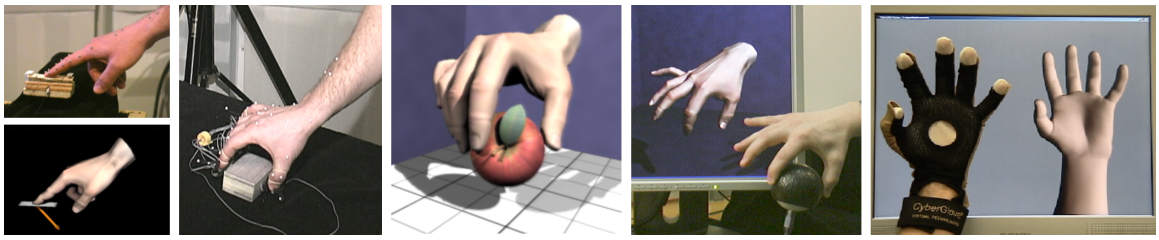


Figure 1.1: A preview of our results. From left to right, (a) interaction capture using an instrumented platform, and a visualization of the capture data, (b) capture of a grasp on a small box using fingertip-mounted force sensors, and (c) captured interaction applied to a new object, (d) grasp approximation in a virtual reality interface using the Tango, (e) demonstration of EigenSkin deformations driven by a CyberGlove.

in a dynamic environment with friction. The term quasi-static does not mean the model is stationary, but rather that the model is in an equilibrium state at every point in time. This is a nice feature of our simulation approach since the trajectory of the hand still captures all the original dynamics and control of a complex biological system, while deviations from the original motion are possible due to the compliance in the joints. These deviations or displacements from the original trajectory accommodate differences in the new environment and its dynamics. For instance, given a captured interaction of a grasp on a small cube (see Figure 1.1b), we can then apply the same interaction with the same joint compliances to an object with a completely different shape of similar size (see Figure 1.1c). The captured interaction also provides a convenient basis to manipulate the interaction with a slippery cube made of ice, or a very heavy cube made of lead, or a cube attached to a lever on a machine.

From our perspective, the problem with motion capture is that it only captures the motion, and not how humans move. That is, humans do not produce motion by just controlling position, but instead appear to control impedance, and an equilibrium point, through co-activation of antagonistic muscles [Feldman 1986; Hogan 1984; Bizzi et al. 1992]. The impedance is highly dependent on task, intent, and object geometry; it is also important for stability [Burdet et al. 2001; Rancourt and Hogan 2001]. With our interaction capture approach, the impedance and equilibrium point trajectory form our intermediate representation. Simulation of new interaction then emulates the production of human movement, preserving the intent of the captured motion, while impedance allows changes in environmental conditions (i.e., changes in surface shape, object mass, and friction).

Capturing the variation of impedance in different interactions is important and has been an active area of research. The traditional approach to estimating impedance involves perturbation [Xu and Hollerbach 1999; Hajian 1997; Hasser and Cutkosky

2002; Milner and Franklin 1998; Gomi and Kawato 1997]. That is, while performing an action, the subject has a small force applied to their hand or finger. System parameters are then estimated by observing the response. This is not ideal since perturbation not only complicates the capture process, but also changes the motion we are trying to measure. Additionally, most previous studies assume that contact is not present when estimating these system parameters even though it is during contact that knowing the behaviour of the system is most useful. In our approach, we instead propose an estimation method based on observation alone of the forces and motion (measured at high temporal rate) at the time of contact. Delays in human sensory feedback make this approach possible. The short period after contact begins can be used for system identification since the motion of the system is altered by newly formed contacts. The changes in motion due to reflexes is typically delayed as long as 100 ms because of neural transmission time combined with the time between the arrival of activation signals at the muscles and muscle force production. We take steps towards validating our approach through perturbation-based estimates.

Detailed anatomical models with biologically-justified muscle deformation and activation are becoming popular in computer graphics for simulating human motion [Pai et al. 2005a; Tsang et al. 2005; Teran et al. 2003]. Our approach, in contrast, uses a simple robotics-inspired model of a kinematic structure with torsional springs at the joints (i.e., compliant joints). Muscle and tendon properties along with the interconnections of these tissues with bones define the passive behaviour of a human hand. We abstract these complexities by modelling the behaviour at the level of kinematic joints. It is important for this simplified model to exhibit biologically based properties since human joint stiffness can be quite low in comparison to those used in many robotics applications. That is, robot control often uses high gain and tight feedback loops (e.g., when position control is important), while for biological systems the compliance tends to be higher, which helps accommodate the significant delays in sensory feedback. This issue is recognized as important by other recent work in computer graphics [Neff and Fiume 2002; Yin et al. 2003; Pollard and Zordan 2005], however, stiffness parameters have typically been selected by hand. Interaction capture can instead provide estimates from capture data.

Interaction capture and synthesis has potential applications in variety of areas, such as robotics, ergonomics, prosthetics, virtual reality, and entertainment. Additionally, even though our focus is on hands and grasping, much of the work we present here applies equally to arms, interaction using tools, and compliant articulated structures in general. We focus on hands because they are our primary tools for interacting with the world around us. Hands have always been important in computer graphics [Catmull 1972; Gourret et al. 1989; Rijpkema and Girard 1991; Koga et al. 1994]. Though they have typically seen little attention in comparison to face modelling or body motion, hands have recently started attracting much more attention, as witnessed by the growing literature [Albrecht et al. 2003; ElKoura and Singh 2003;

Kurihara and Miyata 2004; Pollard and Zordan 2005; Tsang et al. 2005].

Hand based interaction is an important problem complementary to the primary goal of this thesis. With the philosophy that interaction in virtual environments should be as natural as picking up and manipulating real objects, we introduce a novel interface that employs a ball-shaped device called the Tango (see Figure 1.1d). The Tango performs interaction capture by measuring pressures on its surface and linear accelerations using an embedded accelerometer. It also serves as a proxy object, providing the user with valuable passive force feedback. The key idea is to use pressure and acceleration to recognize the shape and movement of the user's hand during grasping. This allows the user to interact with 3D virtual objects using a hand avatar, without the need for a glove or other sensors attached to the hand. We create grasp-shape approximations from examples by comparing pressure measurements with previously collected data. However, rotationally invariant comparisons are critical for both efficiency, and reducing the number of examples required. Following Kazhdan et al. [2003], our comparisons use rotationally invariant features computed from pressures decomposed into a spherical harmonic representation. Using acceleration information to control position is straightforward; however, we introduce methods involving heuristics and thresholds that make reliable drift-free navigation possible with the Tango.

Finally, this thesis also addresses the problem of rendering articulated deformation, which is a fundamental problem for computer animation of hands, and articulated characters in general. The articulated movement of human hands results in complex skin shapes due to the deformation of soft tissues surrounding the bones. Rendering these deformations in real time is very challenging. Currently, most real time character animation is done using a common linear transform blending technique called (among other things) Skeletal-Subspace Deformation (SSD) [Magnenat-Thalmann et al. 1988]. While popular, this technique is also widely known to suffer from several key problems: buckling near joints in extreme poses, poor behaviour for complicated joints (e.g., shoulders and thumbs), and limitations on the range of deformations that can be easily modelled. Our method, called EigenSkin [Kry et al. 2002], extends the common Skeletal-Subspace Deformation (SSD) technique to provide efficient approximations of complex skin deformations exhibited in simulated, measured, or artist-drawn characters. We start with an SSD model and a set of key poses not well approximated by SSD. Vertex displacements between a given pose and the SSD model are mapped back to the neutral character pose. This provides a displacement field pose correction. Instead of storing these displacement fields for each key pose and then interpolating between them at runtime (as in pose space deformation [Lewis et al. 2000] or shape by example [Sloan et al. 2001]), we use Principal Component Analysis (PCA) to construct an error-optimal basis for representing the pose corrections. However, we do not simply use PCA on the displacement fields defined over the entire surface, since this would lead to a large number of basis functions and would

be inefficient for hardware rendering. Instead, we decompose the model into locally supported domains that are learned from the influence of individual joints on the displacement fields. The resulting memory-sensitive set of locally supported correction bases constitutes the EigenSkin approximation, and is suitable for rendering with graphics hardware (see Figure 1.1e). Pose-space interpolation of basis coordinates is performed on the main CPU, while bone transformations and EigenSkin corrections can be compiled in a static display list and computed in a vertex program on the GPU.

1.1 Thesis Organization

Here follows a description of the organization and contributions of each chapter.

First, Chapter 2 provides a general survey of related work in graphics and other fields of research. Note that discussion and citations of the most important related work also appear throughout the thesis.

Chapter 3 describes our approach to the measurement of motion and forces. We start by briefly describing our methods and considerations for motion capture (Section 3.1). This is followed by a description and analysis of different methods for capturing forces (Section 3.2). This includes methods for calibration, force estimation, and synchronization, for fingertip mounted force sensitive resistors (Section 3.2.2) and fingernail sensors [Mascaro and Asada 2001] (Section 3.2.3). Section 3.3 discusses considerations necessary for the use of the Tango as an interaction capture device, and Section 3.4 discusses synchronization methods for all sensors. The final section in this chapter provides a summary of how the techniques in this chapter work with the rest of the thesis.

In Chapter 4 we describe our technique for building our intermediate representation of the captured interaction. Section 4.1 provides the framework for the compliant articulated structure that we employ both in this chapter, and for simulation in Chapter 5. We then introduce our approach for estimating joint compliance (Section 4.2). Here, we also present validation through comparison of values reported in previous work with our own perturbation based estimates (Section 4.2.2). Once compliances are estimated, we build the reference trajectory as described in Section 4.3. We also investigate compliance at the fingerpad in Section 4.4 as a means of justifying Tikhonov regularization during simulation. Finally, we provide a short summary to connect this chapter to the work in the rest of the thesis.

Our synthesis algorithm is described in Chapter 5. Section 5.1 derives a linear complementarity problem (LCP) that deals with a quasi-static compliant kinematic structure in contact with a dynamic environment with friction. Though an LCP framework is commonly used for friction with multiple contacts, our formulation is novel due to the incorporation of position-level compliant coupling between different

finger contacts. Section 5.2 discusses our simulation results.

Chapter 6 presents our work on using the Tango for 3D whole hand interaction in virtual environments. Section 6.1 presents our technique of approximating grasps from pressure using examples and rotationally invariant matching features. Sections 6.2 and 6.3 describe methods for using the Tango for positioning and interaction in a virtual environment, which is followed by a discussion of results (Section 6.4).

Chapter 7 presents our work on EigenSkin [Kry et al. 2002], which provides a means of rendering the complex soft-tissue deformations that occur during hand movement. Section 7.1 introduces skeletal subspace deformation (SSD), and is followed by an explanation of how we compute locally supported displacements (corrections) to the SSD model for our example data (Section 7.2). Displacements are then converted to a reduced basis (Section 7.3), and interpolation functions are computed for the reduced basis coordinates corresponding to the example poses (Section 7.4). Section 7.5 discusses implementation issues using a vertex program in graphics hardware, followed by a discussion of results and examples (Section 7.6). We conclude the chapter with a discussion of issues involved in building an EigenSkin hand model from real world data (Section 7.7).

Finally, Chapter 8 presents our conclusions, a discussion of limitations, and suggestions for future work.

Chapter 2

Related Work

We cite and discuss the most relevant research in the context of our work in subsequent chapters; however, as this work builds on ideas in several different areas of study, we also provide this wider survey of the related work in this chapter. These areas are divided into four groups: graphics, robotics, neuroscience, and anatomy.

2.1 Computer Graphics and Animation

In this section, we start by discussing previous work on computer animation of articulated characters. We classify the different approaches as procedural, motion capture based, or hybrid approaches that combine the two through compliant articulated structures. In addition to the coarse level animation of articulation, rendering realistic skin deformations is an important problem. We finish this section with a discussion of related work on deformation models for rendering articulated characters and their acquisition.

2.1.1 Procedural Animation Synthesis

Most early grasping and manipulation work in computer graphics used procedural animation methods. Animation generated by a simple procedure or algorithm has a degree of flexibility that provides an advantage in generating motion. Changing parameters can generate a wide variety of motions. The disadvantage is that for human motion these methods can often produce results that appear synthetic.

The early work of Rijpkema and Girard [1991] synthesizes grasps using a knowledge-base of grasp-strategies for different shape primitives. Grasp selection is initiated by having the user choose the appropriate primitive strategy. The hand shape for the grasp is then computed using a combination of heuristics and inverse kinematics. A final grasp shape is computed by bending the joints in each finger towards a computed pinch configuration and locking each joint individually when its further motion would cause collision.

Sanso and Thalmann [1994] demonstrate an autonomous approach that combines both heuristics and a taxonomy of grasps. Objects are annotated with primitives at features considered important for grasping. For a user-selected feature-grasping task, a simplified grasp taxonomy is consulted to select a grasp type. Compatible hand postures are then found with a combination of heuristics and a minimization of hand motion. The final finger contact positions are determined via collision detection with an interpolated motion of two hand shapes. The resulting motion is essentially a key frame animation that brings the arm from a rest pose to the computed grasp pose.

Grasping in the Jack system also uses an autonomous approach [Douville et al. 1996]. High-level goals are first broken down into a manipulation plan which is subsequently executed as a procedural animation in a simulation. The motions of individual fingers are controlled independently with carefully constructed parallel finite automata. Each node of the automata contains a user-written control-program designed to achieve a specific action, such as adjusting a joint angle until contact. Transitions occur when an associated user written condition evaluates to true.

Huang et al. [1995] also present an automated system for animating grasps. Heuristics, such as forcing the thumb to be visible to the avatar and preventing the palm from facing the exterior side of the body, help select the approximate hand positions necessary for grasping the object. Fingers close around an object until they make contact, at which point the joints above the contact stop moving while the joints below the contact continue to close on the object. Joint animations result from bell shaped angular velocity spline functions. Rezzonico et al. [1995] extend this approach to work with a data glove in an interactive setting. They address the problem that glove configurations do not match the virtual hand, which occurs since the user is grasping empty space. A state machine monitors whether or not the hand is free, in the process of grasping, or has securely grasped the object. Correct hand configurations result from opening the hand until the fingers are no longer penetrating the object.

Motions that are more complex can be modelled quite effectively by restricting the investigation to a very specific domain of motions, for example, hand motions for playing musical instruments. Kim et al. [2000] present an algorithm for animating a violinists' hands using a neural network trained with examples generated with optimization. For hands that play guitar, ElKoura and Singh [2003] present a procedural algorithm where resulting hand configurations are modified with the k nearest neighbours in a database of captured hand motion. A related technique is the use of inverse kinematics by example [Rose et al. 2001].

2.1.2 Animation via Motion Capture

With respect to hand capture, there has been a considerable amount of research on marker-less vision based methods. Pose estimation is difficult even with marker based

methods, but recent work in vision shows promising results [Lu et al. 2003; Lin et al. 2004]. As our focus on grasping and manipulation is a higher-level problem, we rely instead on commercial solutions to hand tracking such as data gloves and marker based tracking.

Though most work on motion capture in the graphics community does not address hand grasping and manipulation directly, many of the methods are general enough to be useful, to some degree, for hand related problems.

Example Based Motion Synthesis

A recent trend in motion capture reuse has been to assemble small clips of motion from a database to generate new motion [Kovar et al. 2002a; Lee et al. 2002; Arikan and Forsyth 2002]. The small motion clips connect seamlessly because transitions between clips are only made when the configuration and velocities are similar. Joint angles and joint velocities are used with a custom weighted distance metric by Lee et al. [2002] and Arikan and Forsyth [2002], while Kovar et al. [2002a] use marker locations at several consecutive frames in their metric. The later metric compares a larger number of dimensions but avoids skeleton specification and implicitly takes into account velocity and acceleration information. Data driven approaches generate excellent motion that preserve the natural subtleties of the original motion. The drawback, however, is that they only produce similar motions to those that were captured. In addition, for any contact that occurs in a new motion, there must be an example in the database.

Identifying and Respecting Contacts

Contact is a key element to interesting motion, and for motion involving contact, motion capture alone only provides part of the picture. The forces involved during contact tell much about the interaction being performed (e.g., light touch or hard grasp), and provide an explanation for object motion during manipulation.

Measuring forces during motion capture is common in rehabilitation research (e.g., gait analysis). For computer animation, contact forces have been of less interest than contact events such as foot falls. Such events are commonly identified by an automatic analysis, user annotation, or a combination of both.

For example, Bindiganavale and Badler [1998] find contact constraints (foot plants and hand object contacts) by correlating sign changes in acceleration with the proximity of end-effectors to tagged objects. Their work on retargeting motion with hand object contact focuses on the preservation of important spatial relationships in the motion, such as picking up a cup and bringing it to the mouth, despite differences in link lengths between the captured and animated subjects. Inverse kinematics solutions that satisfy the constraints at zero crossings are linearly blended into the existing

motion to create the new motion. Although hand and finger motion is captured with a CyberGlove, hand movement in the retargeted motion is unaltered and iconic.

Kovar et al. [2002b]. identify foot plants with a combination of automatic methods and user annotation. They present a method for cleaning up the foot sliding that often occurs due to motion capture inaccuracy and the kinematic fitting process. Their solution uses inverse kinematics to satisfy the contact constraints, but also introduces small changes in bone length to avoid high joint accelerations that would otherwise be necessary (and noticeable) to satisfy the constraints.

Although the medical community commonly uses combined motion and force capture for gait analysis, it has received little attention in graphics. A recent exception is FootSee [Yin and Pai 2003], which measures ground force contacts during motion capture, and builds a model for synthesizing posture and motion from new ground contact data.

Optimization and Dimensionality Reduction

Optimization is a common approach to solving reuse problems in motion capture. Optimization with space-time constraints, for example, has been used for adapting motion capture to new characters with different limb lengths [Gleicher 1998], for editing motion [Gleicher 1997], and for building transitions that either join motions or create cyclic motions [Rose et al. 1996]. Generally, these approaches produce excellent results. However, they do not incorporate the physical constraints that would ensure a physically valid motion. This is important for full body motion and is addressed by Fang and Pollard [2003] for creating new motion from key poses.

Building on the approach of optimizing to create physically valid motion, Safonova et al. [2004] propose a method to produce natural human motion using a low dimensional representation of the joint configurations that occur for specific behaviours (e.g., jumping). This has the benefit that coordinated motions (arm swings that happen at the same time as leg swings) will appear naturally in the optimized solution. Contact constraints pose a problem since the low dimensional representation does not allow constraints to be satisfied. They solve this using inverse kinematics to compute joint angles at frames that have constraints. This requires an extra term in the objective function, a weak constraint that penalizes joint configurations the farther they are from the low dimensional joint space.

With respect to modelling human hand constraints, Lin et al. [2000] also use a reduced dimension approach. They describe valid hand shapes with principal components in conjunction with joint limits and a fixed two-thirds distal-proximal interphalangeal relationship. This model does not include the space of natural hand shapes that involve contact.

Bowden [2000] incorporates a temporal aspect to the process of learning a statistical model of human motion. First, PCA reduces body shapes for specific tasks to a low

dimensional space. This space is reduced further via K-means clustering and a PCA computation on each cluster. Temporal constraints are modelled as a Markov process.

2.1.3 Hybrid Approaches

Compliant kinematic structures provide a means of mixing motion capture and procedural methods. As such, our work on interaction synthesis is most closely related to hybrid approaches. The behaviour of a hand or limb is dependent on the anatomy, its current configuration, muscle activations, and tendon stiffness. But for animation, it can often be sufficient to capture the system properties in a simplified model where joint properties such as compliance capture the properties of complex anatomy. The related works in this section make use of this simplification.

Knowing the compliance of joints is useful for maintaining physical plausibility in animation. It allows us to predict how a motion might change under different conditions, such as catching a heavy object instead of a light object. Zordan and Hodgins [2002] address this in a system that allows motion capture to hit and react. Here, motion capture provides commands to a proportional derivative (PD) joint controller. The gains of the PD controller lower briefly when new contacts occur, permitting perturbations of the existing motion while maintaining balance. The result is new motion that contains reactions to momentary contacts that were not present in the original motion. Another example of compliance, though not involving contact, is the work of Neff and Fiume [2002]. They adjust gains of an equilibrium point controller (equivalent to PD) to create a motion that appears either more tense or more relaxed. A related earlier approach to producing computer animation of human movement with natural yet constrained styles is that of Lee et al. [1990] on strength guided motion.

Compliance also proves useful for solving under-constrained inverse kinematics. For example, Zhao and Badler [1994] use variable joint compliances to resolve redundant degrees of freedom, while Yamane and Nakamura [2003] combine a singularity robust inverse with a feedback law to find configurations satisfying a set of pin constraints. Another compliance related example which does not involve contact is the work of Yasumuro et al. [1997]. Hand postures are computed by minimizing the energy stored in torsional springs of a compliant hand model. The total energy provides a measure of the hand posture's naturalness with respect to a weighted set of *natural pose* joint-pair affine relationships. They estimate the affine transforms from a set of photographs. Residual variances are normalized and used as gains to balance internal torques, due to deviations form the natural pose relationships, with user specified torques. The motivation is that a joint that only exhibits weak linear relationships with other joints should be easily commanded into configurations that violate the relationships. Consider the similarities of this approach with more formal dimension reduction methods [Safonova et al. 2004; Lin et al. 2000; Bowden 2000]. Omitting

joint coupling equations allows joints to be completely independent (e.g., thumb with respect to the fingers). This is also possible in a PCA setting by explicitly partitioning dimensions, but can also exist indirectly since a poor linear relationship between joints yields independence. Safonova et al. penalize configurations that wander from the average pose with a Mahalanobis distance, implicitly capturing the independence for these weak relationships provided the example data has large enough variances. In contrast, Yasumuro's approach penalizes configurations based on a weighted distance to the subspace.

2.1.4 Rendering Deformations for Articulated Characters

Rendering the animation of articulated models is a hard problem when displaying skin deformations due to soft tissue deformations is important. In graphics, recent work has investigated deforming articulated characters using geometric methods [Magnenat-Thalmann et al. 1988; Singh and Kokkevis 2000; Lewis et al. 2000] and physically-based methods [Wilhelms and van Gelder 1997; Scheepers et al. 1997; Gourret et al. 1989; Teran et al. 2003]. Despite this, most character animation in interactive applications, such as video games, is based on a geometric skeletal deformation technique commonly referred to as linear blending, matrix palette skinning, or Skeletal-Subspace Deformation (SSD), in which vertex locations are weighted averages of points in several coordinate frames (see [Magnenat-Thalmann et al. 1988; Magnenat-Thalmann and Thalmann 1991]).

One alternative is to store a large database of character poses, and interpolate between them [Maestri 1999]. While these approaches give animators great control over character deformation, they have the disadvantage of requiring a potentially very large number of poses for animation, and lack an underlying kinematic model. Nevertheless, such approaches are common, especially for facial animation [Parke and Waters 1996].

A hybrid approach which effectively combines SSD and morphing, is the work of Lewis et al. [2000] who introduced *Pose Space Deformations* (PSD) to overcome the limitations of linear transform blending while retaining a kinematic approach. Starting with a (simple) SSD model, they then store vertex displacement offsets between the SSD surface and various character poses. At run time, the character may be simulated by mapping interpolated displacements onto the underlying SSD character model, thereby providing a kinematic deformation model that also has artist-drawn poses. While this is a big improvement over character morphing, and sufficiently interactive for animators, storing surface displacements for each pose in a large pose space is a memory inefficient approach for hardware applications.

Similar to PSD, Sloan et al. show a more efficient method of interpolating an articulated figure using example shapes scattered in an abstract space [Sloan et al. 2001]. The abstract space consists of dimensions describing global properties of the shape,

such as age and gender, but also includes dimensions used to describe configuration, such as the amount of bend at an elbow. Interpolation occurs in the rest pose before SSD is applied, but involves blending over all example shapes for every vertex. This becomes inefficient and difficult to map to hardware with the large number of examples required for a highly articulated figure since the independence of abstract space dimensions is not taken into account (e.g., left elbow and right elbow).

Additional geometric alternatives include methods based on examples which globally fit a deformation model [Wang and Phillips 2002; Mohr and Gleicher 2003]. The work of Mohr and Gleicher creates better deformations through the use of additional skinning transformations such as half joint rotations. This very effectively fixes the collapsing joint problem. Example poses are used to both find weights, and to adjust the rest pose vertices. This is solved as an alternating least squares problem (fixing weights to solve rest pose and fixing rest pose to solve weights). In contrast, Wang and Phillips find the optimal weights for the example poses without changing the rest pose. However, instead of weighting the transforms with a single weight, they weight all entries of the skinning transforms. A variation of ridge regression with multiple smoothing parameters provides smooth estimation of the weights. Both these approaches, since they fit a deformation model globally, lack the ability to generate pose dependent effects, such as wrinkles that form only at specific configurations.

In Chapter 7 we describe the EigenSkin approach [Kry et al. 2002] for creating a model of human hand deformations. EigenSkin works by correcting an approximate deformation model with a displacement, computed as a superposition of displacements where each is a function of an individual joint angle. This approach has similarities to the reduced representation of deformations used by Blanz and Vetter [1999] in the creation of a morphable model for face synthesis.

In addition to character poses created by 3D artists, samples of deformation behaviour can be computed using physically-based and reality-based deformable models. Such models have been widely used [Terzopoulos and Fleischer 1988; Terzopoulos and Witkin 1988; Metaxas and Terzopoulos 1992; Cani-Gascuel 1998; O'Brien and Hodgins 1999; Pai et al. 2001; Allen et al. 2002], although most approaches are not intended for real time (hardware) rendering. Recently, approaches for fast simulation of physical dynamic volumetric deformations have appeared [Zhuang and Canny 1999; Debunne et al. 2001; Picinbono et al. 2001] for interactive applications, such as surgical simulation. Other recent fast methods for large dynamic deformations include piecewise linear deformations [Muller et al. 2002], interactive skeleton-driven deformations [Capell et al. 2002], and precomputed vibrations from modal analysis [James and Pai 2002a].

Our interest is more closely related to quasi-static deformation, for which fast deformation techniques also exist [Cotin et al. 1999; James and Pai 1999] but are unfortunately restricted to small deformations unlike those associated with articulated characters. An exception to this is [James and Pai 2002b] which uses a set of piecewise

precomputed deformation models on a kinematic chain.

More closely related to character animation is *anatomically based modelling* of physical deformable models [Wilhelms and van Gelder 1997]; examples include musculature [Chen and Zeltzer 1992; Scheepers et al. 1997; Irving et al. 2004] and faces [Lee et al. 1995]. In an inverted approach, Pratscher et al. [2005] generate the musculature from a sculpted surface skin, which permits the synthesis of anatomically inspired animation starting from only the surface geometry.

Articulated Deformation Acquisition

For acquisition of deformable geometry, previous work has used Cyberware scans [Allen et al. 2003; Allen et al. 2002; Blanz and Vetter 1999], stereo vision and optical flow [Borshukov et al. 2003; Bregler et al. 2000; Lang et al. 2003], silhouettes [Sand et al. 2003], and structured light [Wang et al. 2004]. The common approach fits a given deformation model to the data, and hence a common problem is noise, holes (incomplete scans), and unparameterized data. Noise is often assumed insignificant (though, Allen et al. [2003] reduces the influence of unreliable data at the edges of scans), and missing data is often smoothly filled (i.e., by solving Laplace's equation [Pai et al. 2001], minimizing curvature of a discrete thin plate [Allen et al. 2002]).

Lang et al. [2003] use the ACME system to measure the deformation response of an articulated hand phantom and a stuffed toy tiger (extending previous work of Pai et al. [2001]). Here, the problem of segmentation is addressed by clustering parts of the articulated structure that exhibit similar twists. Optical flow tracks deformation of the articulated model. A linear deformation model is computed to describe the local response due to forces applied to the model surface, while the rotational component is factored out allowing for large global deformations.

In contrast, Allen et al. [2002] avoid the problem of finding the segments of an articulated structure. A coarse subdivision mesh with the correct articulation structure is assumed and details from laser scans are incorporated as normal displacements. The final model is constructed by smoothly blending the seams between the different data sets. With a similar result, Sand et al. [2003] instead acquire full body pose based deformations with a combination of motion tracking and vision methods to track silhouettes. In this case, the surface deformations are fit to normal displacements on a set of elongated pill shaped bones. The model is limited by what can be observed in the silhouettes. Although the model allows concavities, they will never result from the fitting process. Additionally, as both the above models are based on normal displacements, texturing is tricky because texture cannot translate on the surface tangent plane.

An alternative to these methods is to acquire models from photographs, which is possible given a strong prior model. This has been done for faces [Blanz and Vetter 1999], and for hands [Albrecht et al. 2003]. Similarly, Allen et al. [2003] take advantage

of the low-dimensional model they build, with significant effort, from a portion of their dataset by fitting the remaining scans using the low-dimensional basis. Further, each new reparameterized scan allows refinement of the basis.

An important factor in the acquisition of deformable models is the collection of sufficient quantities of data. For example, Allen et al. [2002] minimize the scanning required by carefully choosing what to scan. Kurihara and Miyata [2004] realise a hand model from only five medical scans using a locally weighted pose space deformation. In contrast, structured light scanners can capture geometry and texture at video rates. For example, Wang et al. [2004] capture face geometry and texture at 40 Hz with an RMS depth uncertainty of 0.05 mm in a measurement area of approximately 25 cm square (see also, [Zhang et al. 2004]).

2.2 Robotics

In the field of robotics, grasping and manipulation continue to be important problems and active areas of research [Bicchi 2000; Shimoga 1996]. Robot control and path planning have similarities to the procedural methods in Section 2.1.1, but with the difference that the methods here are primarily concerned with direct applications in robotics and only take inspiration from human motion, as opposed to the methods in computer graphics that endeavour to synthesize human-like motion.

2.2.1 Path Planning

Probabilistic road map methods, including rapidly expanding random trees (RRT), provide a fast method for building collision free paths in high dimensional configuration spaces. Koga et al. [1994] handle complicated manipulation tasks with this type of path planning. Here, the grasp configurations are not found autonomously, but instead a set of possible grasps is provided. Given the task of moving an object from one place to another, the planner constructs a collision free multi-arm motion that can involve regrasping the object. Their method also applies results from neurophysiology to create motion with plausible human postures. However, the overall motion still suffers the problem common to most procedural animation systems in that it lacks the subtleties seen in real human motions.

More recently, Yamane et al. [2004] use a path planning approach combined with motion data. The path of the manipulated object is found with an RRT algorithm, modified to ensure inverse kinematics constraints are solvable. Note this approach differs significantly from the traditional RRT methods designed for high dimensional spaces since the object configuration space has small dimension in comparison to the joint space. In their inverse kinematics computations they use marker positions rather than joint angles so as not to restrict themselves to a particular kinematic model.

Choosing marker positions that both satisfy the hard constraints (hand and feet positions), while setting other marker positions as weighted combinations of observed poses, they implicitly bias their inverse kinematics solution to natural poses. The path is smoothed (shortened) by trying to short-cut interior nodes in the tree. Once a final path is chosen, the piecewise linear path is approximated with a radial basis function, thus smoothing the path further. Velocity profiles from the motion capture data are applied to the chosen path to give a final natural motion with bell shaped velocity curves.

Optimization provides an interesting alternative to high dimensional search. Given a collision free path, Quinlan and Khatib [1993] use gradient descent to find a locally-shortest smooth path that avoids obstacles. They draw an analogy between this method and the behaviour of an elastic band that tries to shorten its length but must wrap around any obstacles in its path. More recently, an approach that exploits graphics hardware for solving the path planning problem in real time for reaching actions of animated characters has been shown by Liu and Badler [2003].

2.2.2 Learning from Human Demonstration

For robots there is noticeable interest in learning control and grasp strategies from humans [Bicchi 2000]. For example, Bekey et al. [1993] add heuristics chosen by observing humans to improve robot control. Also, Pollard and Hodgins [2002] show a method that effectively learns a manipulation strategy from human examples of quasi-static manipulation tasks. Here, force closure and ground contact are maintained during the task while the original manipulation wrenches are preserved as closely as possible. Other examples include work of Kang [1994] on robot instruction by human demonstration, and Kang and Ikeuchi [1997] on mapping human grasps to manipulator grasps.

Grasp identification also has importance in learning strategies from human demonstration. Kang and Ikeuchi [1994] describe a grasp identification method based on the *contact web*, a graph structure with finger contact information and positions. Bernardin et al. [2003] look at the problem of grasp identification by combining pressure sensors with a CyberGlove with an accuracy of 92.2%. Finally, Ekvall and Kragic [2004] describe a grasp identification method for their human demonstrated grasping system.

2.2.3 Grasp Quality

Pollard and Wolf [2004] use example grasps to simplify grasp synthesis when there are a large number of contacts. Grasp synthesis algorithms typically require time exponential in the number of contacts [Pollard 2004]. Pollard and Wolf present a polynomial time method for synthesising partial and full force-closure grasps that are

above a quality threshold. Borst et al. [2004] addresses grasp quality with respect to how well a grasp can compensate either arbitrary disturbances or a set of disturbances specific to a task that are known a priori.

Miller and Allen [2000] build a physics based grasping simulator that can be used with arbitrary articulated robot hand models. The system serves as a tool for evaluating robot hand designs through grasp quality measurements, but also has use in grasp planning. Grasp motions are either user-controlled or generated procedurally by closing the fingers around the object at selected velocities.

2.3 Neuroscience and Psychology

Psychologists and neuroscientists have strived to understand human motion for some time, and in their efforts have developed a number of empirical laws and models that attempt to describe various aspects of human motion.

Generally, these laws and models are violated under certain conditions, or are valid only in specific situations. Despite any implication from their naming, these biological laws do not govern all human motion. Nevertheless, for synthesizing motion with human qualities, they can provide a good first order approximation.

Some related work cited here covers partially disjoint topics such as reaching, hand motion (ignoring finger motion), and human motion in general. We look to this work for insights and understanding in addition to any specific ideas directly applicable to the grasping and manipulation synthesis problem.

Good surveys on the computational approaches to human motion exist [Wolpert and Ghahramani 2000; Flash and Sejnowski 2001]. In contrast to computational models, MacKenzie and Iberall [1994] provide a largely descriptive review of grasping from the viewpoint of psychologists, covering such aspects as the different phases of grasping and a discussion of the grasp taxonomy.

2.3.1 Biological Motion Laws and Motion Planning

An example of early work, Fitts' law predicts "movement time" based on the distance to and the size of a target [Fitts 1954]. Though primarily one dimensional, it is commonly used to evaluate 2D point-and-click user interfaces and is applied by the Jack System to select movement times for autonomous reaching [Douville et al. 1996].

Donders' and Listing's laws concern the kinematics of the eye. Donders states there is a one-to-one mapping between eye orientation (torsion) and gaze direction (a neural, rather than mechanical constraint), while Listing states that there exists a plane such that the only ocular orientations possible are those to which the eye can be rotated from a rest position using a rotation axis that lies in this plane. Listing's law does not hold during the vestibular ocular reflex (eye movement that counters

head motion) or vergence between targets at different distances. Donders law has been reinterpreted for head and neck configurations, the outstretched arm pointing, and to a lesser extent, the arm while reaching [Flash and Sejnowski 2001]. In this vein, early work of Soechting and Flanders [1989] provided Koga et al. [1994] a means of generating *human-like* solutions for under-determined inverse kinematics problems. However, Soechting et al. [1995] show that Donders' law as applied to arm motion has questionable validity. Instead, they suggest a minimum work transport model for predicting arm configurations. Their work is similar to the minimum torque change model of Uno et al. [1989], and related to, yet quite different from, the minimum jerk model of Flash and Hogan [1985] since this model is stated in hand coordinates. Previous work appears divided as to whether planning is done in hand or joint coordinates, though some recent evidence exists that planning is done in both coordinate systems [Flash and Sejnowski 2001].

Optimization based models, such as minimum jerk, generally assume that human motions are controlled such that they are as smooth as possible. Ignoring control and the mechanical properties of the muscles and tendons[1], trajectory smoothness is a reasonable assumption since muscle forces are integrated twice to give positions. Optimization based models have the feature that they can account for some of the motion laws proposed in previous work [Todorov and Jordan 2002]. An earlier example, Atkeson and Hollerbach [1985] interpret bell shaped hand velocity curves during unconstrained reaching motion using simplified arm dynamics, and show that the velocity profiles are similar to those created with the minimum jerk model (Koga et al. [1994] also take advantage of bell shaped velocity curves for human motion synthesis). Likewise, it has been shown that minimum-jerk predictions obey the two-thirds power law relating velocity to curvature [Viviani and Flash 1995; Richardson and Flash 2002]. Richardson and Flash further suggest that segmented control does not necessarily exist for complex trajectories because they generate such trajectories with a cost functions that makes no use of segmentation. An alternative view, however, is that motion is planned so as to be robust against inherent noise in the motor system [Harris and Wolpert 1998]. This minimum variance model also agrees with the two-thirds power law and Fitts' law.

The reach motion planning model of Rosenbaum et al. [2001] also proposes a cost function to minimize. Here, the function is related to movement time and obstacle avoidance rather than torque, work, or jerk, and the aim is to satisfy rather than optimize. They show that their model can express effects such as increased opening of the hand for movements at faster speeds. Obstacle avoidance is accomplished with a diffusion based search method, much simpler than but similar in spirit to RRT methods [Koga et al. 1994].

[1]Smoothness may also result from muscle and tendon properties, combined with the low-pass relationship between motorneuron activation and muscle tension [Zajac 1989]

Note that dynamics based models have less relevance at the finger level. In addition, it appears reasonable to assume that methods of planning hand motion differ from methods used for the fingers when fingers are used together for grasping and manipulation.

2.3.2 Synergistic Control

Jerde et al. [2003] investigate the recognition of hand shapes, specifically for identifying sign language finger spelling. Discriminant analysis based classification suggests that four principal components have lower accuracy than four joint angles (selected via backwards elimination). From this, they conclude finger-spelling hand shapes are controlled by individual finger movements influenced by biomechanical and neuromuscular constraints rather than synergistic control.

Santello et al. [1998] investigate the final imagined hand postures for a large number of familiar objects, and find that hand joint angles, although distinct, show a high amount of correlation. Just two principal components accounted for more than 80% of the variation, and measurements projected onto these two components exhibited a bilinear distribution. Though small, the subsequent components did not represent random variability but instead gave additional information about the object. They suggest hand posture control involves a few postural synergies to regulate the general shape of the hand, coupled with finer control for subtle adjustments. They also notice that postural synergies did not coincide with grip taxonomies, suggesting that hand posture may be regulated independently from the control of contact forces[2].

Haggard and Wing [1998] notice that hand preshaping for grasping mostly occurs after a "via" point has been reached when a curved path is required for hand transport. They note that the preshaping is delayed for curved movements relative to straight movements. In comparison, Santello and Soechting [1998] notice that maximum aperture of the hand is reached approximately midway in the reaching movement. At this point in time, the hand posture is influenced by the shape of the object. In contrast, the shape of the hand at this point does not determine the shape of the object. Instead, they observe that during the approach, the posture of the hand gradually molds to the shape according to the contours of the object. They also notice that the proximal and distal movements of the arm are dependent as subjects had difficulty controlling their hand shape for grasping an object when their arm was fixed.

Santello et al. [2002] look at the influence of sensory input on hand motion during grasping. The experiment involved twenty different objects presented to the subject with a virtual image projected with a parabolic mirror. Hand shapes were recorded while the subject tried to grasp the virtual object with and without visual cues. Hand

[2]They report that this is consistent with experiments on monkeys, and with the different sensory demands for control during contact, specifically the dependence on tactile feedback.

shapes were also recorded for grasps of the real object. Santello et al. found that the absence of vision did not affect the hand shape, and the presence of the object only influenced hand shape when there was contact. Two principal components accounted for 75% of the variance, and the second component only became important for the second half or the reaching motion, an observation which agrees with their earlier work on gradual molding.

The virtual finger hypothesis states that the control of the hand configuration is simplified by having a set of fingers working together as a functional unit [Arbib et al. 1985; MacKenzie and Iberall 1994]. With respect to this hypothesis, Santello and Soechting [1997] investigate the ability of subjects to match the size of an object (cube, parallelepiped, or cylinder, all in multiple sizes) with the aperture of different thumb-finger combinations. Subjects either perceived the object visually without touching, haptically with the opposite hand, or they had no feedback. They find that their experiments did not completely satisfy the virtual finger hypothesis because they expected one principal component per virtual finger, but instead found that all degrees of freedom of the hand were controlled as a unit.

Other work has focused more closely on the forces involved during multi-fingered grasping. Using PCA, Santello and Soechting [2000] find evidence that a small number of force patterns (or synergies) dominate control, while the less significant components are still important for fine control. Rearick and Santello [2002] observed that grasping strategies were affected by the predictability of the center of mass and not by hand dominance. Zatsiorsky et al. [2003] also found that subjects used a common multi-digit synergy in a load and torque balancing task, and they suggest this as evidence of virtual finger control.

Baud-Bovy and Soechting [2001] also investigate the virtual finger hypothesis. They asked subjects to use a tripod grasp to hold an apparatus with orientable contact surfaces, while 6 DOF force sensors in the apparatus measured index and middle forces (static grasps allowed the thumb force to be computed for equilibrium). The thumb force did not depend on surface orientation and instead consistently pointed midway between the opposite fingers. Baud-Bovy and Soechting liken the action of the opposing fingers to that of a single virtual finger because small variations in the force of one finger were observed with similar variations in the other. They also hypothesized that forces would be focused at a point maximizing the safety margin for all three fingers in the grasping task (defined as the tangent of the angle between the force direction and the border of the fiction cone), but interestingly this was observed to be false in several cases.

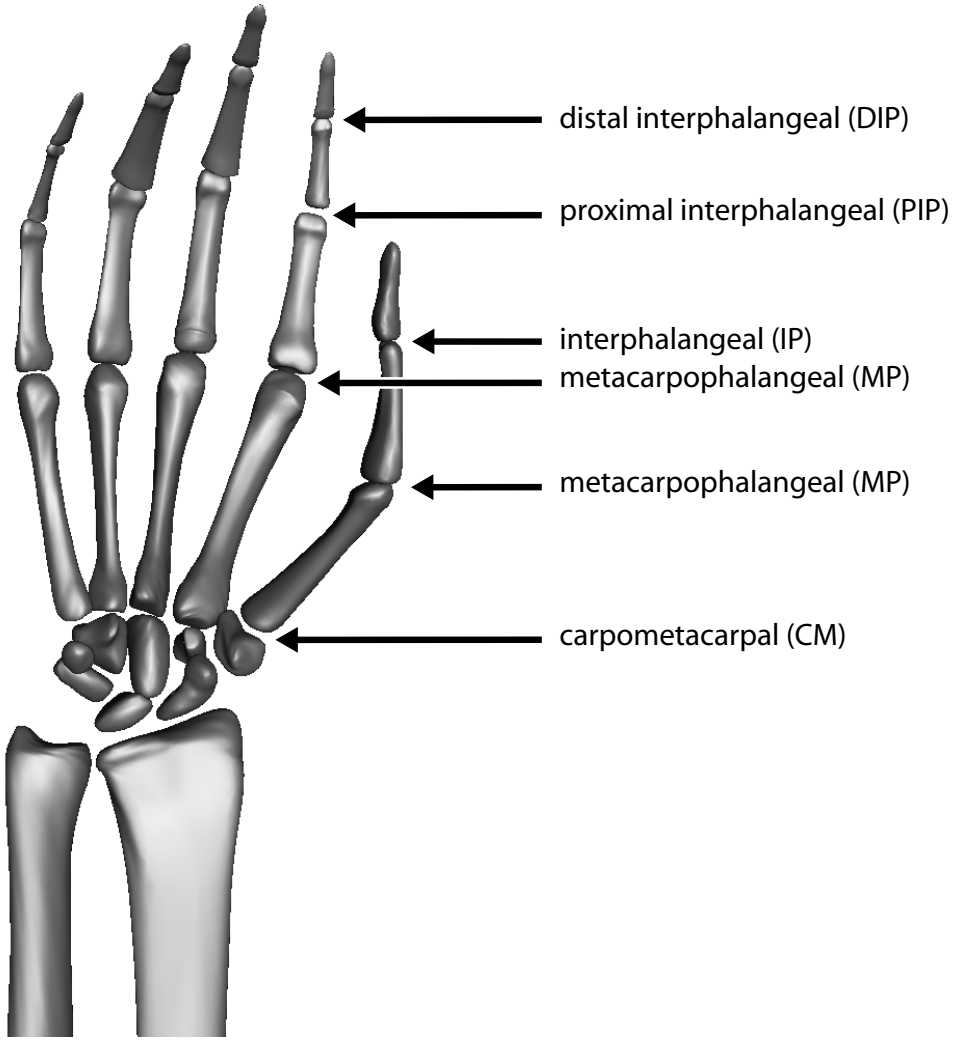


Figure 2.1: Hand bones and joint name abbreviations shown for the index finger and thumb.

2.4 Anatomy

The human hand is a very complex mechanism. It consists of many different components that interact with one another during movement. Detailed illustrations of these components of the hand can be found in any first year anatomy textbook (for example, see [Moore and Dalley 1999]). Figure 2.1 shows an image of the hand bones with several joints labelled with their names as well as the abbreviations used throughout the rest of this thesis.

The bones largely define the mechanics of how the hand can move, however, tendons and muscles also play an important role. For instance, biomechanical constraints on the motion of the hand exist due to connections between the tendons. A good example of this is the coupling of the motion of the two distal joints of a finger, though these

joints can move independently when there is a forces acting at the fingertip. Similarly, there is coupling across fingers that can be easily demonstrated by first making a fist and then attempting to straighten the ring finger at the knuckle.

Constraints on hand movement are not solely due to biomechanics; there also exist neurological constraints. For example, Hager-Ross and Schieber [2000] attempt to quantify the independence of finger movements. They report factors affecting independence beyond the most obvious biomechanical constraints due to the interconnection of tendons in the back of the hand. These factors include the organization of motor units, and likelihood of shared central inputs to the motorneuron pools that innervate the different finger muscles. Here, a motor unit refers to a group of muscle fibres and the single motor nerve that activates the fibres, while a motorneuron pool is all neurons that innervate one muscle. Hager-Ross and Schieber report the interesting observation that the fingers in the dominant hand are no more independent than those of the non-dominant hand.

Tsang [2005] provides a good recent survey of studies concerning hand anatomy, biomechanics, and neurological constraints.

Chapter 3

Interaction Capture

In physics based simulation, the position and orientation of an object can be viewed as driven by torques and forces. The torques and forces cause accelerations, which are integrated twice to evolve the object's position and orientation over time. In the inverse problem, we can try to approximate the forces and torques necessary to explain observed motion via inverse dynamics; however, this is a tricky problem because interaction forces, such as friction, cannot always be estimated from the motion alone.

In observing human grasping and manipulation, we capture both hand motion and contact forces. We call this *interaction capture*. The contact forces during interaction are important, not only because they explain the motion of the object, but because they cause changes in the hand configuration. That is, forces acting on the hand have a passive effect on the motion, creating subtle variations in hand configuration due to biomechanical properties such as compliance in tendons and muscles. Likewise, forces cause tactile and kinaesthetic feedback, which plays an important role in active control.

Our goal in interaction capture is to acquire both the active control and passive behaviour of a motion. Therefore, although we measure both motion and force, ultimately we are interested in neither; from force and motion data we estimate an intermediate representation that encapsulates the active control and passive behaviour of the system. This estimation process is described in Chapter 4, while this chapter describes our approach to problems concerning the direct measurement of interaction. Specifically, we analyze and discuss problems concerning measurement, calibration, estimation, and synchronization of different sensors. Capturing hand motion and contact forces are not independent problems because of sensor synchronization and the compatibility of sensors during simultaneous measurement; this is discussed in the last section of this chapter.

The interactions we investigate include surface interaction tasks such as scratching (for example, to remove a sticker from a surface) and surface exploration (for example, to feel for a bump or small crack on the surface). We also consider grasping to be a type of interaction, and we specifically investigate precision grasps. These grasps only involve the tips of the thumb and the fingers and can consist of only two fingers or as many as all five. For instance, we often employ a two finger precision grasp (thumb and index finger) to pick up small objects such as the as nuts and bolts that hold a computer together, while for larger objects we tend to use more fingers (i.e., when there is room on the surface of the object to establish contact with additional fingers). Note that precision grasps only form a small part of most grasp taxonomies (see MacKenzie and Iberall [1994]), while surface interactions do not appear in grasp taxonomies since they are not grasps. Nevertheless, the techniques in this thesis can also be applied to other grasps, such as power grasps, by instead looking at tool and arm compliance as opposed to compliances in the fingers (this is an interesting direction for future work).

3.1 Capturing Motion

The large amount of occlusion that occurs during grasping as fingers wrap around an object makes capturing grasping motion difficult with vision-based methods. For motion capture, we primarily use a Vicon motion capture system [Vicon Motion Systems], which employs multi-camera stereo vision to reconstruct marker locations, and we take special care in the placement of markers.

3.1.1 Vicon Motion Capture

Our Vicon system uses six M2 cameras that detect infrared light and can be used for capture at rates up to 1000 Hz. Infrared emitter rings located at each camera illuminate retro-reflective markers. The system strobes the emitter rings for easy identification of marker locations through subtraction of background illumination. The Vicon software reconstructs marker locations and estimates the joint angles of a given kinematic model (specified with a template and calibrated by the Vicon software). We do all our reconstruction processing off-line, though this is also possible in real-time.

Marker Placement and Occlusions

We modify the default Vicon hand marker placements by moving markers for the distal finger segments to the tops of stilts. This increases marker visibility when an object is grasped. Figure 3.1 shows the marker placement on stilts attached to different sensors, and mounted directly on the fingernails with double-sided tape.

Figure 3.1: Markers on the distal phalanx are placed on stilts rather than the fingertip, shown on fingernail sensors (left), force sensitive resistors (middle), and attached to the fingertip (right). This improves marker visibility during grasping.

Markers identify joint positions best when placed on the axis of a joint. This fixes the marker position in both the parent and child reference frames, but is not possible for MP joints and impractical for PIP joints due to occlusion. Instead, for MC and PIP joints, we place the marker directly above the joint axis and set constant parameters in the template skeleton with our best estimates of the marker to joint-center distances. The remaining markers on the hand and forearm are placed consistently using photographs and their positions are set as unconstrained for the Vicon calibration process.

Although the wires for fingertip-mounted sensors may lie naturally or comfortably on the back of the hand, this can cause marker occlusions. To avoid this, we either leave the wires loose (i.e., for the force sensitive resistors described in Section 3.2.2) or route them along the palm side of the hand (i.e., for the fingernail sensors described in Section 3.2.3, see Figure 3.2). In the case of the fingernail sensors, routing the wires on the palm side of the hand also helps the wires reach the fingertips during extreme configurations, such as the open hand and closed fist. The wires are somewhat stiff and tend to move more easily under the palm than when stretched over the back of the hand.

Specular Reflections

Since all LEDs in the Vicon emitter rings are strobed simultaneously, it is possible that one camera can see false markers caused by the specular reflection of light from another camera's emitter ring. This can happen quite often for shiny surfaces and causes noticeable problems with the reconstruction. For hand tracking, we use many markers all in close proximity. When specular reflections show up in the same small area, the Vicon reconstruction software can identify many false possibilities for marker locations. This creates new markers that can be either correctly left unlabelled or confused with real markers. Alternatively, specular reflections can combine with good marker data causing a displacement in the real marker's reported position. To

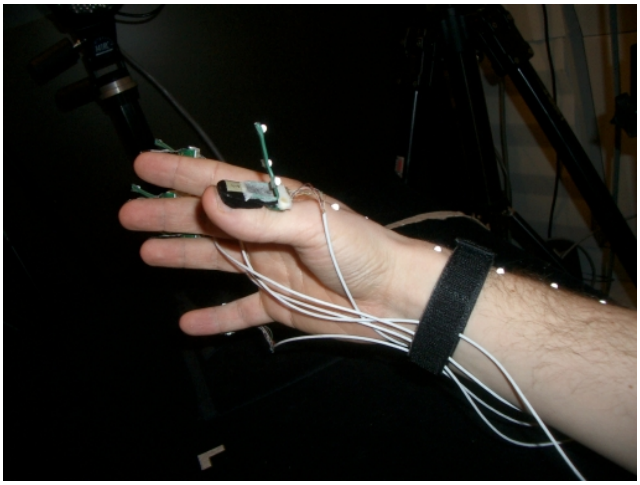


Figure 3.2: Routing wires on the palm side of the hand reduces marker occlusions.

minimize this problem we apply flat black masking to shiny surfaces of all our sensors.

Kinematic Model Approximations

We calibrate our kinematic hand model template to match the subject using Vicon software. This needs to be done once per subject, while a recalibration of marker positions is necessary each time the markers are re-attached. Our template model allows for many of the bone lengths and joint positions to be estimated, however we make a few approximations. For example, the location of the thumb CM joint is difficult to estimate, so we fix its location relative to the wrist with a displacement that we visually approximate for the capture subject. Another example is that our model restricts the MC joints to lie in a plane; however, the palm can curl such that these joints are no longer planar. Although this is not important for the precision grasps we investigate, full hand interactions would be more accurately modelled by including these degrees of freedom into the kinematic model. Figure 3.3 shows the calibrated skeleton for the one subject that was used in our interaction capture trials.

Post Processing

Vicon motion capture can be performed in real time using the Vicon real time engine. However, an unpredictable lag in the arrival of data due to reconstruction processing and network transmission makes synchronization difficult. Instead, recording motion capture for later post processing is preferable because we can reliably use high capture rates, and end up with better capture data; we can spend more computation on reconstruction, and more user time in cleaning and evaluating the capture data off-line.

Using the reconstruction software with parameters tuned for small volume capture,

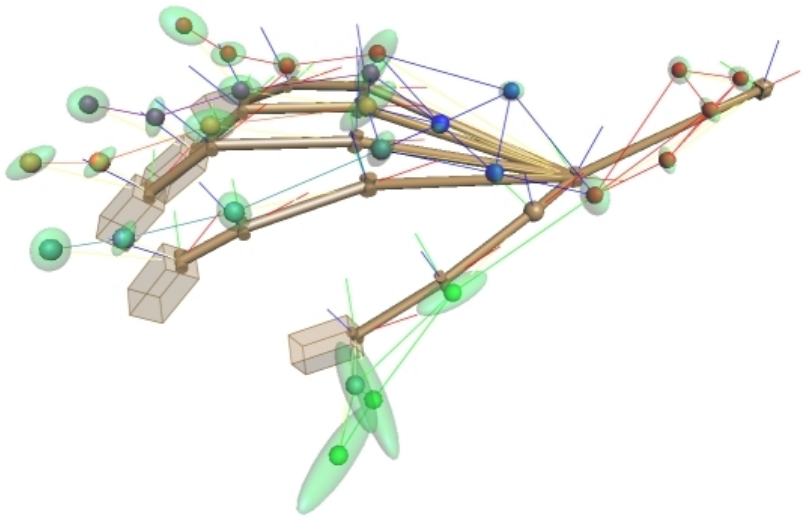


Figure 3.3: The calibrated kinematic model produced by ViconIQ for a typical range of motion trial. The pose shown has joints at their mean positions. Transparent green ellipsoids show marker covariances.

we reconstruct data captured at a rate of 500 Hz with calibration errors of approximately 0.3 mm. Capture at 1000 Hz reduces the resolution of the Vicon cameras to 640×210 as opposed to 640×480 when capturing at 500 Hz. The lower resolution and reduced light sensitivity (due to higher shutter speed) make reliable capture at 1000 Hz more difficult. Because of this, we use 500 Hz as the maximum motion capture rate.

Occlusions and camera noise are unavoidable and prevent perfect reconstructions. We filter noisy trajectories and fill missing data using a variety of tools (though usually a spline interpolation suffices). Typically, in our capture trials, marker data is missing only for a short time (individual frames, or 0.01 to 0.1 seconds), but longer durations on the order of seconds can also occur, in which case we discard the motion.

3.1.2 Other Options

For this work, we had access to other motion capture solutions. The CyberGlove [Immersion Corporation], used previously in our EigenSkin demonstration, estimates hand configuration based on the measurement of strain gauges that are sewn into the glove. The provided software estimates joint angles from the strain gauges.

The glove has the advantage that it measures small relative joint angles changes reasonably well, with a resolution of about 0.5 degrees. Likewise, it has an advantage over vision-based methods since occlusion is not an issue. Unfortunately, even with calibration, the estimated fingertip positions have significant errors, which are visible in situations where the fingers touch (e.g., when the thumb and index finger touch to

make an O.K. gesture). This is due to an assumed kinematic model and errors in the joint angle estimation, the effects of which are compounded by the kinematic chain from palm to fingertip. The problem is significant, and since contact interactions are exactly what we want to measure, we did not use the CyberGlove for interaction capture.

Like the CyberGlove, the Aurora [Northern Digital Inc.] is not affected by occlusions. Using magnetic tracking, it provides absolute positioning with sub millimetre accuracy. The resolution for relative motions is better than 0.01 mm. The sensor coils are extremely small (like a grain of rice), and each coil can be tracked in 5 dimensions (the missing dimension is rotation about the axis of the coil). Thus, pairs of coils combine to give 6 DOF tracking. Up to 8 single coils can be tracked simultaneously at 22 Hz. One disadvantage is the small workspace, which is a cube with sides of length 0.5 m. In addition, the system fails to track coils within the workspace when they are moving quickly. More importantly, ferrous metals distort the Aurora's magnetic field and affect sensor accuracy. This makes our force-torque sensors incompatible for simultaneous capture, and as such, we have not used the Aurora for interaction capture.

3.2 Capturing Forces

Force-torque sensors are ideal for capturing interaction because they provide information on the full wrench at the contact. Frictional slip forces at the contact can be measured, as well as the frictional torque that opposes rotation due to the contact patch that forms during contact with deformable fingerpads. Accurate measurements are important for producing estimates of the passive behaviour of the hand during interaction (discussed in Chapter 4).

Section 3.2.1 provides more detail on our use of force-torque sensors, while Section 3.2.2 and 3.2.3 introduce fingertip-mounted sensors and presents methods that address important calibration and synchronization issues.

3.2.1 Force Capture with Instrumented Objects

Force-torque sensors employ strain gauges and are often used in robotics applications, such as at the end of a robot arm for measuring the load at the gripper. The Zebra is a six-axis force torque sensor made by Zebra Robotics Inc. (though no longer available). We have used this sensor extensively for gathering force data, and for calibrating other sensors (e.g., fingernail touch sensors and force sensitive resistors). However, the Nano17 from ATI Automation [ATI Industrial Automation] is better suited for capturing fingertip forces; it is designed for accurate measurement of small loads and it has a small size.

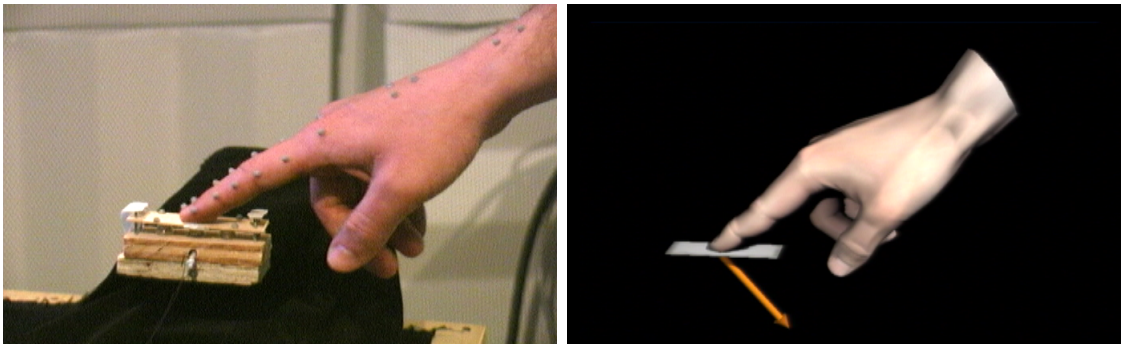


Figure 3.4: Force capture using an instrumented object. Left shows interaction with a wooden-plank torque-arm attached to the Zebra sensor (unseen in background under black cloth). Right shows a visualization of the capture data with contact force rendered as an arrow.

The Nano17 is measured with 16 bits accuracy using its own dedicated National Instruments data acquisition equipment. In contrast, the Zebra's acquisition circuit provides 10 bit measurements of the sensor's strain gauges, and has a sensitivity of approximately 0.1 N per bit. For additional sensitivity, we have attached a torque arm (a wooden plank) to the end of the sensor (though this technique can be applied to any force-torque sensor). The result is a two-axis linear force sensor with improved linear resolution. Torque about the x axis is used to estimate linear forces in the z direction, and torque about the z axis is used to estimate linear forces in the x direction. Measurement accuracy for these two axes increases by a factor of 10, allowing forces as small as 0.01 N to be detected. Because the plank is slightly flexible and can vibrate briefly at the time of contact, we can expect to see the impulse response of the plank at the time of contact recorded in the force measurements. In the case of the plank attached to the Zebra sensor, however, we typically observe little vibration in the measurements at the time of contact. Vibrations at the time of contact can be important when estimating system parameters while using other sensors and this is addressed in more detail in Section 4.2.

Note that we need not interact with any sensor directly (i.e., the force sensor does not need to be located at the point of contact). Instead, we can mount a rigid object on the sensor and transform the measured wrenches to contact frame coordinates estimated from motion capture. Proximity detection between the hand and object models (posed with motion capture data) provides a means of computing the contact frame. This is similar to what we do for the Zebra sensor, as shown in Figure 3.4 with a wooden-plank torque-arm attached, except that we do not improve linear force accuracy when transforming a wrench to different coordinates (see the adjoint definition in Section 4.1).

Although multiple sensors are required to measure the multiple contact forces involved in grasping, we can still capture interesting interaction with a single force

torque sensor. For example, in Chapter 4 we estimate compliances for single finger surface exploration and scratching tasks (surface exploration is shown in Figure 3.4).

Sections 3.2.2 and 3.2.3 describe methods for sensing forces at the fingertips rather than on the object. The sensors used in Section 3.2.2 measure normal forces and could likewise be mounted at pre-selected finger plant locations on the object. Although we did not collect any data in this manner, we did perform early experiments with a different one dimensional pressure sensor called the Tactex MTCExpress [Tactex Controls Inc.]. The sensor is a commercially available rectangular touch-pad, and contains a 12 by 6 grid of light-based pressure sensors. Multiple centers of pressure can be tracked, where locations and pressures are estimated by interpolation. Though useful in our initial tests, we did not find a means of calibrating the touch pad for sufficiently reliable measurement of normal forces. Other commercial systems are more suited for measuring normal contact forces. For example, [Pressure Profile Systems Inc.] makes a pressure sensor system called ConTacts. These sensors have a size similar to force sensing resistors, however, instead of using resistance to measure pressure, they use capacitance (like the Tango, see Section 3.3). The sensors are small and flexible, making them well suited for capturing grasping forces, as demonstrated by Bernardin et al. [2005] for grasp classification.

3.2.2 Force Sensitive Resistors

To capture forces during precision grasps, we use sensors attached to the fingertip. [Interlink Electronics] makes a variety of force sensing resistors (FSRs) that have advantages for estimating fingertip forces: reasonable accuracy, and small size. The smallest sensors are flat plastic discs with 5 mm diameter, shown in Figure 3.5, a size convenient for mounting on a fingerpad.

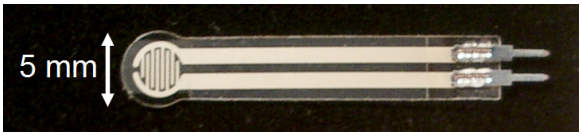


Figure 3.5: Force sensitive resistor.

Pressure applied normal to the surface of an FSR can be measured as a change in resistance. We pair each FSR in series with a 3 kΩ resistor and apply a constant potential of 5 volts. We then measure the variable FSR resistance as a voltage using a 16 bit data acquisition system (a DAQPad from National Instruments).

Measurements vary depending on the distribution of the applied force, as well as any bending of the sensor itself. These situations must be avoided when attaching the sensors at each fingertip (see the description of sensor housing below). Dynamic force measurements are not recommended by the manufacturer since accuracy can vary from ±5% to ±25% with mechanical settling times on the order of seconds.

Nevertheless, our experience is that these approximate measurements are quite good in practice (see the discussion of calibration below).

Sensor Housing

As mentioned previously, measurements vary due to the distribution of the applied force and bending of the sensor itself. To prevent bending we mount the sensor on a small rigid piece of metal that mounts tightly on the finger with elastic bands (see Figure 3.6). Then, for consistent mechanical activation of the sensor, we bend a small strip of the housing's metal back over the mounting surface, squeezing a small flexible ellipsoid and a thin foam pad between the strip and the sensor. In addition to providing consistent mechanical activation, this pre-loads the sensor and avoids the onset pressure threshold, which can be as high as 0.2 N over the surface of the 5 mm diameter sensors. On the outer surface of the metal strip, we place a non-slip rubber pad to provide an artificial finger surface for interaction. Finally, as mentioned in Section 3.1.1 and seen in Figure 3.6, the metal sensor housing is coated in flat black masking tape to avoid interference (due to reflections) during Vicon motion capture.

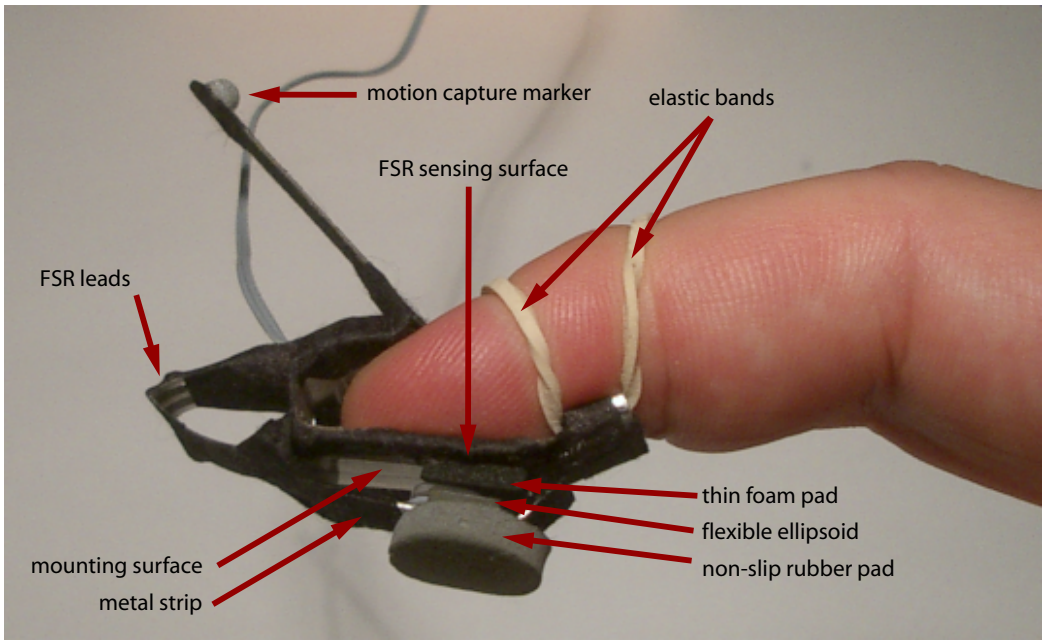


Figure 3.6: Close view of FSR housing on index finger.

Because the sensors only measure forces normal to their surface, we mount the sensors such that their normals point in preselected directions. For the index, middle, and ring fingers, this direction is the normal at the middle of the fingerpad. For the thumb and little finger, however, this varies between the middle of the pad and the side of the finger depending on the grasp. We must orient the sensing direction, in advance of capture, to match the task. When using FSR data for analysis and

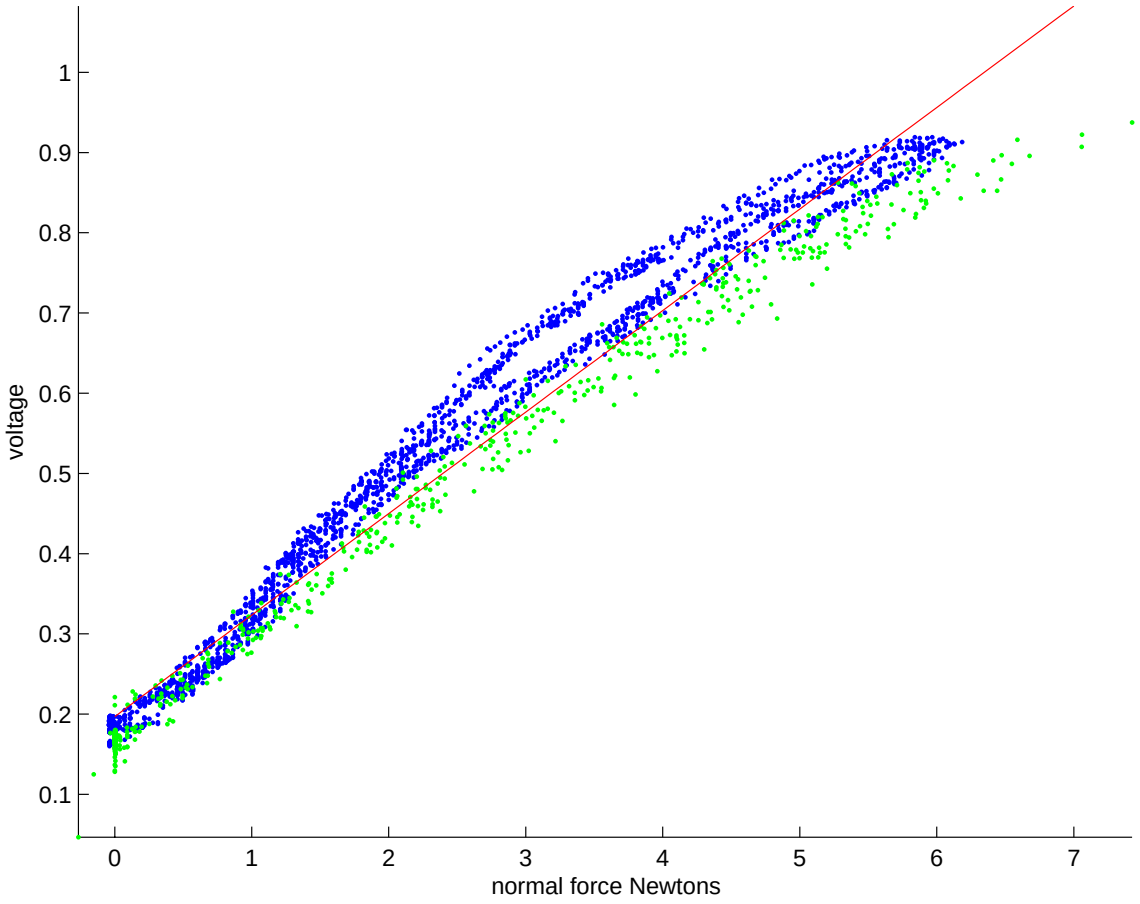


Figure 3.7: Linear fit for approximating normal forces from FSR voltages, for both slow force changes (dark blue), and fast force changes (light green) combined.

synthesis in Chapters 4 and 5, we use force positions and directions (fixed in the distal segment frames) that approximate those of the sensors. These are user selected and stored in an XML configuration file.

Calibration

Because the conductance of the force sensitive resistors is (approximately) linearly related to the applied pressure, we do not expect the force-voltage response to be linear; however, we do find that a linear fit is a reasonable approximation due to sensor pre-loading and the small range of forces that we are measuring. The overall quality of the linear fit can be seen in Figure 3.7.

Likewise, although dynamic measurements are not recommended, the mechanical settling time is fast enough to have only a small effect on our measurements. This can also be seen in Figure 3.7 by comparing the measurements for fast force changes (green) and slow force changes (blue). This suggests that normal force estimates for slowly changing interactions will typically be underestimated, while those for fast

changing interactions will be overestimated. However, the error in measurements introduced by mechanical settling time of the FSRs is insignificant in comparison to the effect of tangential forces.

The calibration data in Figure 3.7 was collected by asking a subject to press a target location (on a force sensor) at different speeds (slow and fast) while applying only forces normal to the target surface. However, nothing in the calibration setup prevented the production of tangential forces. The measurement data shows that small tangential forces were indeed produced, perhaps due to a small posture changes during the different (decreasing and increasing) force production tasks. In fact, the two bands in the blue samples can, in part, be explained by a difference in x direction tangential forces. Note that the finger points in the positive x axis of the frame fixed to the distal joint (to which the sensor is attached). The lower band of slow movement samples (blue) comes from periods in which the normal force was increasing, with positive x tangential forces (approximately 0.5 N at 4 N of normal force). The upper band comes from periods where the normal force was decreasing, with negative x tangential forces (about -0.2 N at 4 N of normal force).

Because the force sensitive resistors only detect pressure applied normal to their surface, the effect of tangential forces on voltage is likely due to mechanical coupling in the sensor housing. Figures 3.8 and 3.9 show the effect in detail. Both figures show calibration data collected where the subject was instructed to produce a variety of forces with tangent forces confined to a specified direction; a different data set was collected for both x and y tangent directions. We model the voltage as a linear function of the two forces (the normal force and either the x or y force). The figures show the contours of the least-squares best-fit plane for different voltage readings, with samples coloured to match their corresponding voltage measurement.

In Figure 3.8 we see that forces in the y direction have little effect on the voltage. Though the contours are not perfectly vertical (i.e., zero correlation between tangential force and measurement), this can be attributed to alignment of the measurement frame and the subject's force production. Likewise, the measurement setup does not prevent the production of forces in the other tangent direction.

Figure 3.9, in contrast to Figure 3.8, shows that forces in the x direction have a significant effect on measured voltage. For example, with a measurement of 0.5 volts we will estimate a normal force of approximately 2.4 N, though the true normal force could be as large as 3.0 N or as small as 1.6 N. The metal strip that pre-loads the FSR likely causes this effect. Recall, the metal strip was bent back over the mounting surface. At the location of the bend, where it joins with the mounting surface, is likely acting like a hinge; depending on the direction of the tangential forces (towards or away from the hinge), the metal strip applies more or less pressure to the FSR.

For our FSR measurements, we estimate only the normal force and assume zero tangential components. We cannot do better from the measurements alone because the measurement is only one dimensional. However, if we were to solve for the tan-

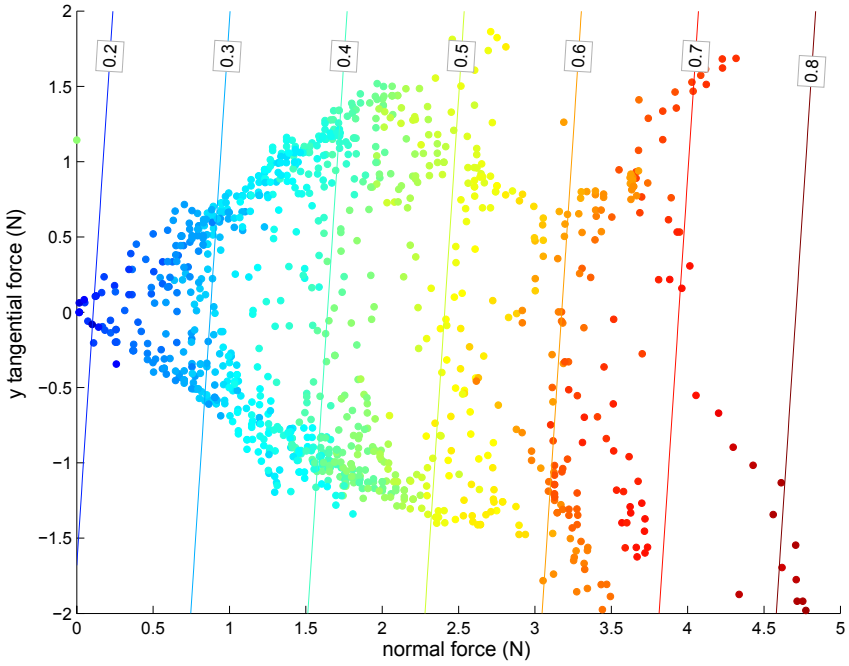


Figure 3.8: Effect of y direction tangential forces on measured voltage. The positive z direction is normal to the fingernail surface, while the positive x direction is the finger pointing direction; the positive y direction points to the left of the finger.

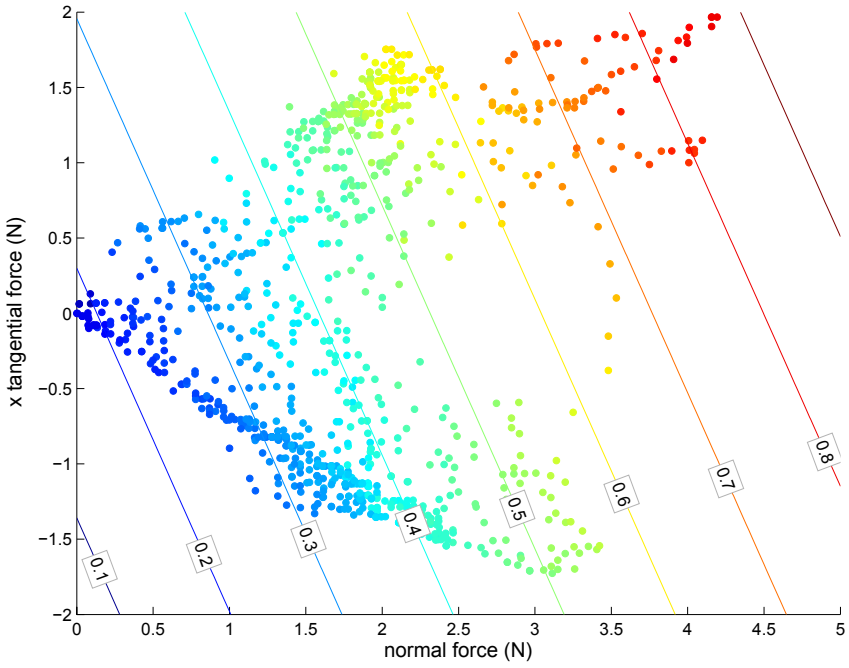


Figure 3.9: Effect of x direction tangential forces on measured voltage. The positive x direction corresponds to the finger pointing direction.

gential forces necessary to explain force closure on a stationary grasped object of known mass, then we would need to include the coupling shown in Figure 3.9 as a constraint.

3.2.3 Fingernail Sensors

The fingernail force sensors developed by Mascaro and Asada [2001a] estimate the pressure on the finger pad based on the distribution of blood underneath the fingernail. Each sensor consists of a small fingernail sized circuit board, on which resides a set of 4 light emitting diodes (LEDs) and 6 photodiodes (see Figure 3.10). The circuit board is embedded in a slightly flexible transparent epoxy that is moulded to fit an individual user's fingernails. Generally, these sensors are not transferable between users because of fingernail shape variations (i.e., the sensors will not make a flush attachment with a different user's fingernails).

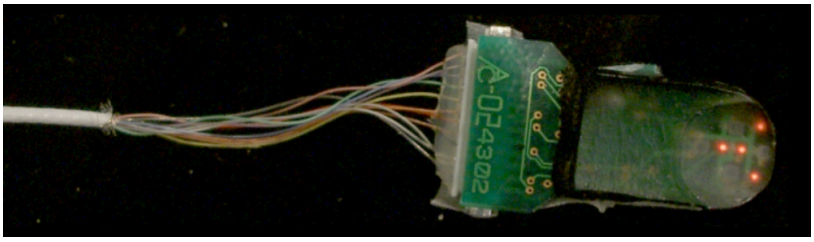


Figure 3.10: Fingernail sensor close-up, nail attachment side.

The LEDs provide uniform illumination of the tissue under the fingernail, while the photodiodes detect the reflected light intensity on a small portion of the nail. Mascaro and Asada place the photodiodes for optimal differentiation of the blood distributions under the fingernail that result from typical pressures and tractions. A small cable containing a bundle of 12 wires carries power, LED control signals, and photodiode signals. Each cable also has shielding to reduce signal noise, as the photodiode signals have small amplitudes.

The 6 signals from 5 fingers require 30 channels of analog to digital conversion. We use a 16 channel (single-ended) DAQPad from National Instruments, combined with the AMUX-64T multiplexor accessory, allowing up to 64 inputs.

The main advantage of these sensors is that the fingerpad is unobstructed [Mascaro and Asada 2001]. Disadvantages include an effectively slow measurement rate that is directly related to the speed of blood movement (typically a few hertz, maximum), and variable accuracy in force estimation (essentially a tricky inverse problem, see [Mascaro and Asada 2004] and estimation discussion below).

Simultaneous use of the fingernail sensors and the Vicon motion capture system requires special considerations, such as marker visibility and specular reflections, as well as careful synchronization due to light interference. Specifically, because the

fingernail sensors are sensitive to the light emitted by the Vicon strobes, measurement of the fingernail sensors must be made between strobes.

Synchronization

The Vicon motion capture system is our primary motion sensor, and as such, we are interested in synchronizing fingernail sensor measurements with Vicon capture. More importantly, careful timing of fingernail sensor measurements is essential because the photodiodes in these sensors are sensitive to the light emitted by the Vicon strobes. Fortunately, there is a port on the back of the Vicon data station (port J2) that provides a signal identifying the exact moment when each motion capture frame is acquired; this frame capture signal is synchronized with the strobing of the infrared emitter rings.

We have constructed a circuit for timing (triggering) measurements of the fingernail sensor photodiodes between Vicon strobes. The circuit, shown in Appendix A, takes two readings for each Vicon frame, allowing for background light subtraction (just like that done by the Vicon system).

Empirical Force Estimation

Mascaro and Asada [2004] describe linear, quadratic, and neural net models for estimating forces from fingernail sensor readings. Their trials suggest the simplest model, the linear model, performs just as well as any of the three. In practice, we observe that the linear model demonstrates an unfortunate amount of drift, resulting in negative force estimations that must be interpreted as zero, or large positive force estimations during periods where there is no contact at the fingertip. Our more demanding usage conditions likely explain the problems we encounter with the linear model. Unlike Mascaro and Asada's well-controlled trials, our tests involve normal hand grasps on non-flat surfaces at different orientations using different parts of the fingerpad.

Instead of linear, quadratic, or neural network models for force estimation, an alternative is to use the training data directly. We have investigated an empirical model that uses the k-nearest neighbours of a sensor reading to estimate a probability distribution function (PDF) for the forces.

Let $D \subset \mathbb{R}^m$ be the measurement domain, $x_i \in D$ be measurements, and $f_i \in \mathbb{R} \geq 0$, for $i = 1..n$, be the corresponding normal force measurements (we only consider normal forces though the method applies equally to 3 dimensional forces). Our measurements have $m = 6$, one dimension corresponding to each of the 6 photodiodes. This could be extended to $m = 24$ to include approximate measurement of subsurface scattering effects, or $m = 8$ to include PIP and DIP joint angle measurements, or both with $m = 26$ (this is discussed further at the end of this section).

Given a new measurement $x \in D$, we find its k nearest neighbours. We use a sphere tree data structure for point to point-cloud distance queries in a space of arbitrary

dimension d. Our implementation is adapted from the sphere tree proximity detection method described by Quinlan [1994]. The k nearest neighbours can be found in $\mathcal{O}(dk \log n)$ time.

We build the tree on centered and whitened data. That is, we perform principal components analysis (PCA) and rescale dimensions so they contribute equally in distance computations. This lets us easily compute Mahalanobis-like distances in truncated spaces by simply summing fewer terms in our ℓ^2 distance computation. The bounding sphere tree is still valid for truncated spaces, though possibly less efficient.

We convert centered data to the whitened space with multiplication by the matrix $B = (\sigma_1^{-0.5} u_1, \cdots, \sigma_m^{-0.5} u_m)$, where u_i is the i^{th} principal direction. That is, $r = B(x - \hat{x})$, with $\hat{x}$ being the average of the training data. A truncated representation of x is simply the first d components of r (with $d < m$). Computing the distance in a truncated space has the advantage of making our sampled data less sparse, but also lets us remove dimensions that only capture noise.

Dimension reduction with PCA requires centering the data, but the photodiode signals exhibit noticeable drift over time as many factors change the amount of blood under the fingernail. Any amount of drift is important as the signals have a small magnitude. We have tried addressing this by centering data with local averages, similar in concept to a high pass filter that removes the DC component of the signal, but we observed little improvement with this approach.

We build a histogram that provides a discretized conditional probability density function,

$$\begin{aligned} p(f|r) &= \frac{\Big|\{f_j \;:\; f_j \in \text{bin}(f),\; j \in N(r,k)\}\Big|}{\Delta f \; k}, \\ \text{bin}(f) &= \Big[\, \lfloor f/\Delta f \rfloor \, \Delta f, \quad \lfloor 1 + f/\Delta f \rfloor \, \Delta f \Big), \end{aligned} \tag{3.1}$$

where Δf is the histogram interval or *bin* size, and $N(r,k)$ denote the set of k nearest neighbours of the measurement $r \in \mathbb{R}^d,\; d < m$ (the truncated PCA space).

A simple force estimate for an observation r may be computed as the average of the distribution, but with one minor modification. If there are no neighbours with a force larger than a specific threshold, then the predicted force is zero. Otherwise, we take the mean of only the neighbours that have a force greater than the threshold. This seems necessary when using the mean as an estimate; in most cases, we observe the number of nearest neighbours with a zero force reading dominate. Figure 3.11 shows results of our estimates using Equation 3.1 with this modification. Here, the threshold is set to 0.18 N, and the probability density function (PDF) in the figure only shows densities of forces above this threshold. We used $d = 3$ as the first three principal components of the FNS measurements explained almost all the variance in the data. Prediction performance seen here for calibration data greatly exceeds

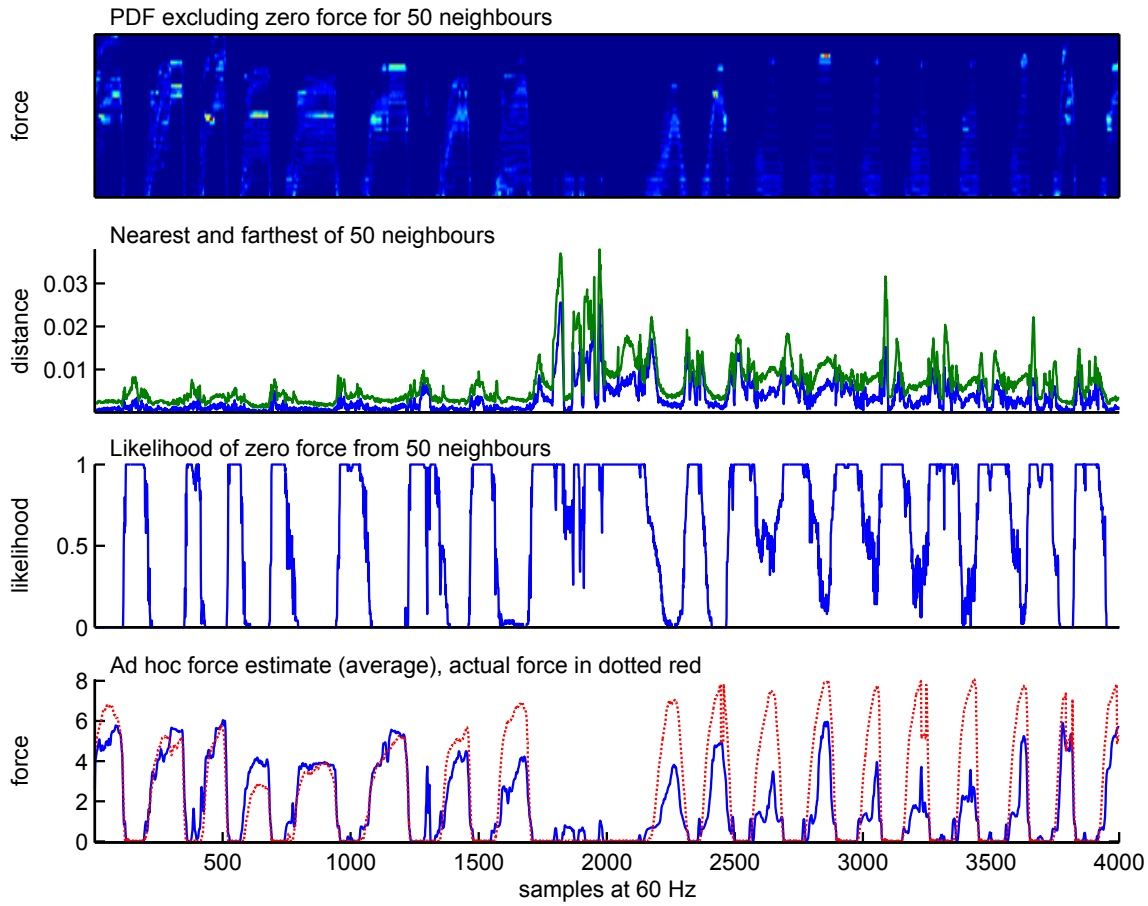


Figure 3.11: Index finger empirical force estimation given 83 seconds of training data. The 66 seconds of test data immediately followed the training data. The PDF histogram has 50 bins (Δf approximately 0.1 N) and is shown with the MatLab JET colour map. Nearest and farthest neighbours give an indication of confidence. Likelihood of zero force gives the proportion of neighbours whose force falls in the zero bin (< 0.1 N).

that of normal capture conditions. In real trials, we observe far less correspondence between positive force predictions and finger-surface contacts seen in recorded video.

Discussion

Generally, these sensors are difficult to use for robust estimation of force magnitude during grasping motions. As a result we have not used them for capturing any of the interactions in Chapter 4. Nevertheless, here we discuss a few ideas that may yet lead to better force estimation with the fingernail sensors.

Firstly, a Parzen window density estimation can be used as an alternative to Equa-

tion 3.1. That is, we might consider a continuous PDF estimate,

$$g(r,f) = \sum_{j \in N(r,k)} w(r,j)\ e^{-(f-f_j)^2/\sigma_f^2}, \tag{3.2}$$

$$w(r,j) = e^{-||r-r_j||^2/\sigma_r^2}, \tag{3.3}$$

$$p(f|r) = \frac{g(r,f)}{\int_{-\infty}^{\infty} g(r,f)\ \partial f}. \tag{3.4}$$

Here $w(r,j)$ weights training sample j with an exponential falloff with standard deviation σ_r; samples when far enough will contribute very little. The standard deviation σ_f spreads force probabilities slightly to neighbouring forces, and this too could be a function of the distance, $||r - r_j||^2$, as the force predicted by farther samples should be incorporated with a larger degree of uncertainty (i.e., larger σ_f). Note that σ_r replaces the need to pre-select a number of nearest neighbours. Tracking the distribution's mode(s) may give better results. For example, a particle filter tracking algorithm (e.g., [Isard and Blake 1998]) could be used to track the modes, though this would require a model for computing $p(r|f)$, such as that described by Mascaro and Asada [2001b].

Secondly, considering Mascaro and Asada also use the fingernail sensors to predict figure posture, it is perhaps obvious that including distal joint angles in the force estimation could improve estimates. In general, since the fingernail sensors try to detect information about the blood distribution underneath the fingernail, readings depend on many factors beyond the application of pressures on the fingertip. One large influence on the readings is tension on the sensor leads, which is unfortunately something that can occur quite easily in moving the hand among configurations. Likewise, contact between the fingernail sensor and adjacent fingers has a similar effect, and can occur quite easily when trying to grasp small objects.

Additionally, training data should be collected under conditions that match (as close as possible) the task to be captured. Insufficient sampling of the measurement space can lead to poor force estimation for any estimation method. Important factors may include, seating posture, arm posture, hand configuration, grasp formation speed, among others. Fast motions during both training and capture should be avoided.

Lastly, because the fingernail sensor LEDs can be controlled independently, we can approximately measure subsurface scattering effects. For each measurement, instead of reading the 6 photodiodes twice, we can take 5 readings: one with all LEDs off for background subtraction, one for each of the four LEDs illuminated on its own (a sixth reading with all LEDs on may also be useful for comparison). This gives us 24 measured voltages for estimating the pressure. In preliminary tests we observed a significant feature in the captured data, suggesting that these measurements contain more information; the capture contained examples where the different signals from one photodiode, when illuminated with different LEDs, had derivatives with opposite

signs. To pursue this method, the synchronization circuit described in Appendix A requires modifications to control the independent illumination of the LEDs.

3.3 The Tango

The Tango (whose name is derived from the word "Tangoreception", meaning pertaining to the sensation of touch) is a hand-size object shaped like a ball. The Tango is a unique sensor as it provides a means of measuring both force and motion. There are 256 non-overlapping capacitance-based pressure sensors on the device's surface, and a 3-axis accelerometer within. Pressure measurements come at 100 Hz as an 8×32 tactual image where the pressure at each *taxel* (tactile sensor element) is computed by dividing two 8 bit signals. Pressure measurements at each taxel are aligned the taxel's surface normal. Figure 3.12 provides a view of the device.

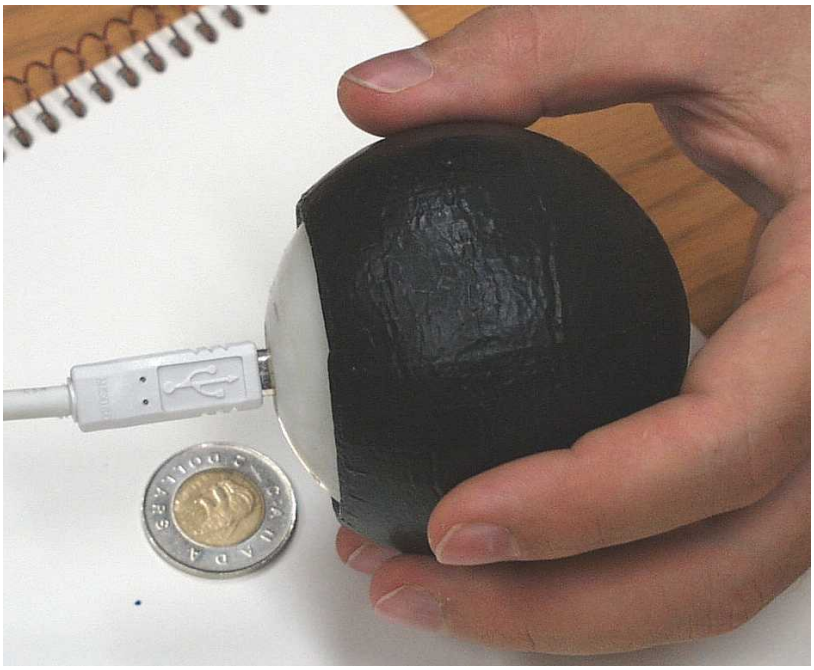


Figure 3.12: The Tango device.

The device is designed to be comfortable to grasp, and large enough to contain all necessary on-board circuitry. Just like a proxy object used in a virtual environment, the surface of the Tango provides passive force feedback, which makes virtual objects more tangible. The Tango uses a high-speed USB interface to send data to the host computer. The device is hot-pluggable. No separate power supply is needed when the device is operating using USB power. There exist three prototypes of the device. The latest version has a wireless Bluetooth communications interface and onboard battery. The battery is rechargeable and is charged when the USB cable is connected. This

version solves an important problem; the USB cable is reasonably stiff and makes it difficult to manipulate the Tango freely. See [Pai et al. 2005b] for more information on the Tango design and hardware.

Tango pressure data is based on 8 bit analog to data conversion, and the raw data has a considerable amount of noise in the time domain. Fortunately, the combined effect lets us improve the resolution of taxel pressure estimates with filtering. We perform smoothing with exponential filters on both taxel and accelerometer data. Figure 3.13 shows calibration data being collected for an individual taxel using the force sensor located in the tip of the WHaT [Pai and Rizun 2003], while Figure 3.14 shows a comparison between raw calibrated pressure data for a taxel, filtered taxel pressures, and force measurements from the WHaT. Here, the filter parameter is 0.9 and the sampling rate is 50 Hz[1].

In addition to smoothing, additional corrections can be applied to taxel data. Due to construction, specifically due to stretching of the surface of the Tango during deformation, taxel measurements along meridians are correlated. Linear effects are correctable with a meridian Green's function calibration [Pai et al. 2005b]. Just the same, we do not correct any of the Tango data that we collect; this has little effect on our results since the corrections are quite subtle.

We use a threshold to remove noise about the zero pressure reading. Due to higher noise levels in taxels near the poles, we also apply a hand tuned scaling to minimize their effect, though, ideally we should set threshold, filter, and scaling parameters based on the measurement variance observed at each taxel for zero pressure. Figure 3.15 shows a typical raw pressure measurement during a grasp.

When Vicon and Tango data are captured together, we can associate taxel clusters (see Section 6.1.1) with fingertips using a proximity computation. When a cluster's total pressure exceeds a user adjustable threshold, we let the cluster contribute to the force at the fingertip to which it is closest, but only if the distance between a fixed point on the fingertip and the cluster center is less than 20 mm. The force direction is computed as the normal at the point on the surface of the Tango closest to the fixed point on the fingerpad. The fixed point on the fingerpad is selected as a point near the center, and is specified in an XML configuration file.

Although the tango does not provide data of high enough quality for the compliance estimation techniques described in Chapter 4, we can still use it as an interaction capture device for a hand with assumed compliance. Chapter 5 contains such an example where we use synchronized Tango-Vicon interaction capture for synthesizing new finger gaiting motion on a light bulb. Focusing instead on whole hand interaction interfaces, chapter 6 discusses a method of using the Tango on its own; that is, as an interaction capture device where both contact forces and hand configurations are

[1]In this case, the 50 Hz sampling rate was selected to match a target graphics rendering frame rate as opposed to the full 100 Hz sampling rate available with the Tango.

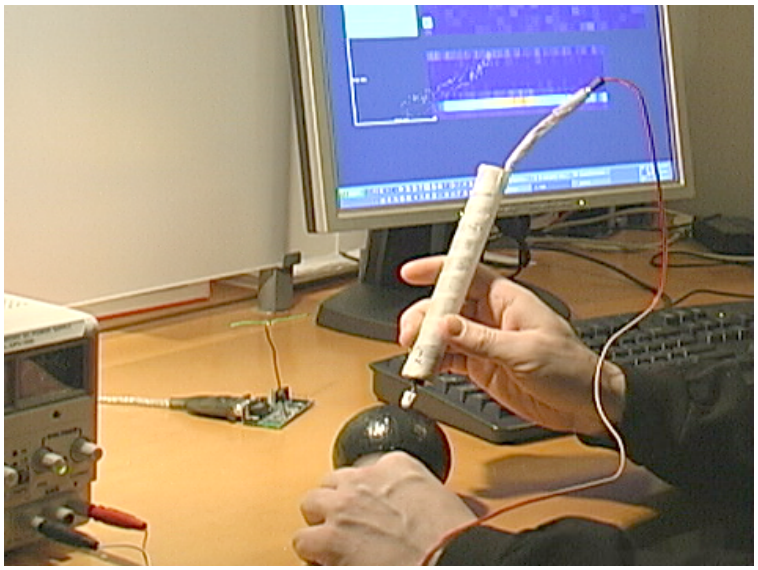


Figure 3.13: A snapshot of Tango calibration data being taken with the WHaT.

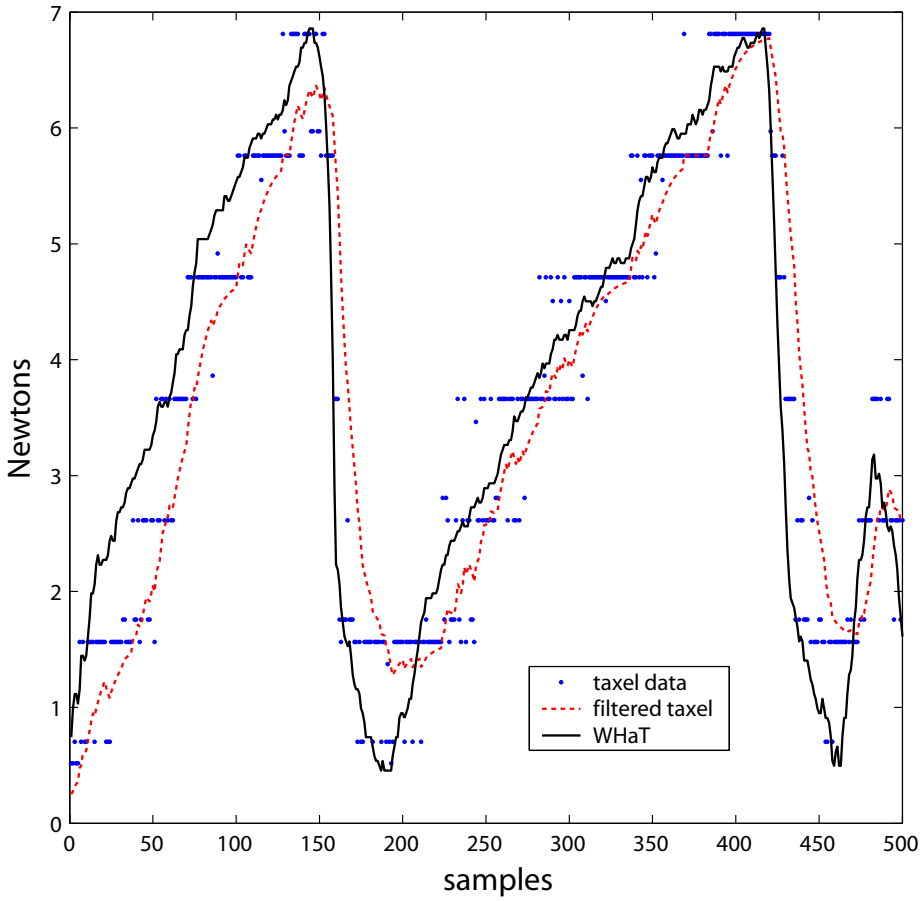


Figure 3.14: Comparison of taxel pressure data, filtered taxel pressure data, and WHaT force data, with samples at 50 Hz.

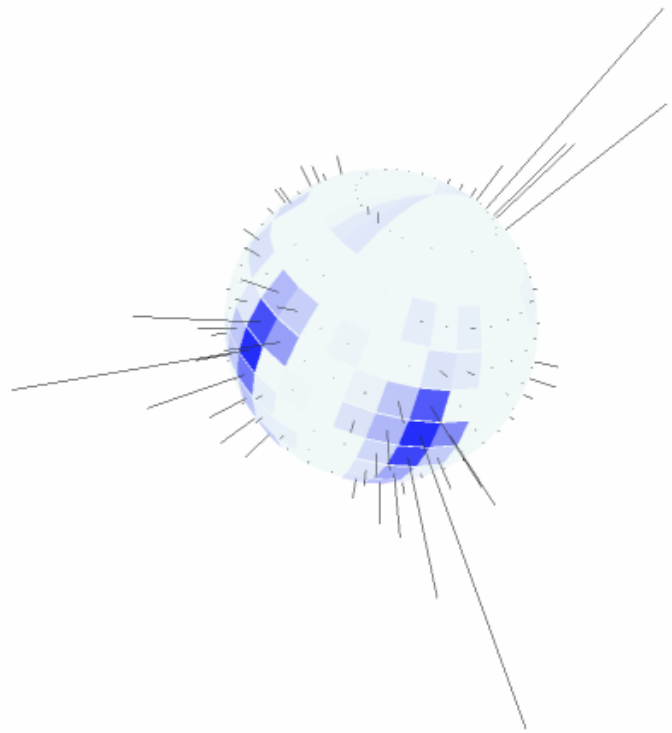


Figure 3.15: Raw pressure distribution measurements on the Tango during a three-finger grasp. Higher pressures are indicated by darker taxels and longer lines emanating from taxel centers.

estimated using only Tango data.

3.4 Synchronization

Tight synchronization is important in Chapter 4 for estimating the passive behaviour of the hand during captured interaction. As mentioned previously in the section concerning fingernail sensor synchronization, the Vicon motion capture system has a port on the back of its data station that provides a signal identifying the exact moment when each motion capture frame is acquired. The port also conveniently provides a means of remotely starting and stopping capture via a toggle switch.

The Vicon system serves as our primary motion sensor, so synchronization of other measurements to the Vicon capture is important. Sensors that use a National Instruments data acquisition (DAQ) device can have measurements triggered on the frame signal by connecting the frame capture signal directly to the `START_SCAN` trigger on the DAQ. We use this synchronization method for the force sensitive resistors.

System	Rate	Absolute Accuracy	Relative Accuracy	Communication	Type
Vicon	60-1000[a] Hz	< 1 mm	< 1 mm	offline or LAN[e]	push
CyberGlove	100 Hz	1.0[d] deg	< 0.5 deg	serial via API	pull
Aurora	22-45[b] Hz	0.82 mm, 0.5 deg	0.04 mm, 0.02 deg	serial	pull
FNS[h]	60[c,g] Hz	poor[j]	poor[j]	DAQ API	push
FSR[i]	60[c] Hz	n/a[j]	n/a[j]	DAQ API	push
Zebra	1 kHz	0.1 N	0.1 N	parallel via API	pull
Nano 17	> 1[f] kHz	< 6[k] mN	< 6[k] mN	DAQ API	pull
Tactex	> 60 Hz	poor[j]	good[j]	serial via API	pull
Tango	100 Hz	n/a[j]	n/a[j]	USB via API	push

[a] Rates up to 1000 Hz available, but with reduced field of view as rate increases.
[b] 22 Hz for 6 to 8 coils, 45 Hz for 5 or less.
[c] Measured via data acquisition device, National Instruments DAQPad, at rates up to 333 kS/s, generally synchronized to Vicon capture rate via `START_SCAN` trigger.
[d] Repeatability between wearings.
[e] Network connection for real time capture with Vicon Real-Time server.
[f] Synchronization via `START_SCAN` trigger unavailable with factory provided cable.
[g] Fingernail sensor force estimates are not accurate past 10 Hz.
[h] Fingernail sensor (FNS).
[i] Force Sensitive Resistor (FSR).
[j] Error unmeasured or difficult to characterize; special calibration required.
[k] Multiple calibrations available, best resolution is 0.8 mN.

Table 3.1: Measurement rate, accuracy, and interface for different sensors.

For other sensors not measured with a DAQ, we connect the frame capture signal to a counter on the DAQ, allowing us to time stamp measurements from these sensors. We use this method, for instance, with the Zebra force torque sensor. Note that although the Nano17 uses a DAQ, the `START_SCAN` trigger signal is not available without modified cabling; the DAQ cable connects directly to the Nano17's interface power supply.

For each type of sensor we have a Java package that communicates with the sensor's Application Programming Interface library (API) through Java Native Interface (JNI) wrapping functions. As summarized in Table 3.1, the sensor API can provide measurements at a fixed rate (pushed by the sensor to the program, or polled), or may have software triggered measurements (pulled by the program from the sensor). When sensors push data at regular rates (e.g., the Tango), the ideal solution for measurement synchronization is resampling, however, we believe this level of accuracy is unnecessary for our purposes. Instead, we use time stamps, and replace missing samples with interpolated data.

3.5 Summary

In this chapter we have shown methods for capturing the movement of hands and contact forces that result during interactions. This sets the stage for constructing an intermediate representation that conceptually explains how the motion was produced. In the next Chapter we use this data to estimate joint compliances and a reference or non-contact trajectory, which ultimately makes possible the synthesis of new interaction through simulation as described in Chapter 5.

Chapter 4

Compliance Estimation

Though we capture both motion and force in the previous chapter, we do not use either directly. Instead, we estimate an intermediate representation that describes the active control and passive behaviour of the captured subject. This is because during contact, force and motion are related through impedance. In modelling a mechanical system we can use a collection of springs, dampers, and masses. The impedance consists of the model parameters that describe the passive behaviour of the system components. For instance, a linear second order system (mass, spring, damper) can have its impedance describe by stiffness, damping, and inertia. The process of finding these parameters for a system is called system identification.

Various models are used in previous studies of fingers and arms. For example, Milner and Franklin [1998] estimate endpoint (i.e., the fingerpad at the end of the finger) stiffness and stiffness ellipsoids, while Hajian [1997] estimate endpoint stiffness, damping, and inertia. Some researchers estimate joint properties (e.g., [Xu and Hollerbach 1999]), as opposed to the properties of an effective endpoint model [Hasser and Cutkosky 2002]. The system identification method most commonly used to identify these parameters it to perturb the motion of the finger or arm with a small force and measure the result. This perturbation method works well, but unfortunately complicates the capture process and also changes the motion or task that was being captured. Additionally, most previous studies assume that contact is not present when estimating these system parameters even though it is during contact that knowing the behaviour of the system is most useful for simulating new interactions. With the measurements obtained from the interaction capture methods in the previous chapter, however, we can use a different approach based on observing the behaviour of the system at the time of contact. This forms the primary focus of this chapter.

Our model of the human hand consists of a kinematic structure with torsional springs at the joints. It is the compliance (inverse of stiffness) of these springs that we estimate from interaction capture. We use this same compliant kinematic structure for simulation in Chapter 5. The compliance describes the passive behaviour

of the system, while the trajectory of rest joint angles (the nominal reference trajectory) embodies the control and dynamics of our captured hand motion. We use joint compliances rather than endpoint compliances because having the joint properties is important for simulation of the whole hand. Endpoint compliance alone, though useful for describing the behaviour at the contact, does not help us determine the new shape of the kinematic structure after a small endpoint displacement (since the kinematic structure is generally underconstrained). In contrast, given joint compliances we can compute just how much each joint will respond to accommodate a small endpoint displacement. Using joint compliances also lets us easily capture non-linear pose dependent variations of the endpoint compliance.

We start this chapter by introducing important notation and equations needed both in this chapter and Chapter 5. Sections 4.2 and 4.3 discuss estimation of joint compliances and reference trajectories, while Section 4.4 provides an analysis of the fingerpad compliance. This is important as justification for matrix regularization that is applied during simulation in Chapter 5.

4.1 Preliminaries

We use a notation for spatial dynamics which is similar to [Kry and Pai 2003]. The homogeneous coordinates of frame i with respect to another frame j is given by the 4×4 matrix ${}^{j}_{i}\mathsf{E}$. We use leading subscripts and superscripts to indicate frames. The homogeneous coordinates of a three dimensional vector x in frame i are denoted ${}^{i}x$. The homogeneous coordinates of this vector in frame j are given by left multiplying by ${}^{j}_{i}\mathsf{E}$.

The spatial velocity ϕ describes the relative motion of a body with respect to the fixed world frame. In coordinates of frame i, it is given by the size 6 column vector ${}^{i}\phi = ({}^{i}\omega^T, {}^{i}v^T)^T$, where ω is the angular velocity and v is the linear velocity of the point at the origin of frame i. Spatial forces, called wrenches, are represented as $w = (\tau^T, f^T)^T$, where τ is the (rotational) torque and f is the (translational) force.

Spatial velocities and wrenches transform according to the adjoint transformation ${}^{j}_{i}\mathsf{Ad}$. The 6×6 adjoint matrix is defined as,

$$ {}^{j}_{i}\mathsf{Ad} = \begin{pmatrix} \Theta & 0 \\ [p]\Theta & \Theta \end{pmatrix}, \quad \text{where} \quad {}^{j}_{i}\mathsf{E} = \begin{pmatrix} \Theta & p \\ 0 & 1 \end{pmatrix}. $$

Here, Θ is a 3×3 rotation matrix, p is a 3×1 displacement, and $[p]$ denotes the skew symmetric 3×3 matrix equivalent to the cross product $p\times$. Spatial velocities, being contravariant quantities, transform by left multiplying, ${}^{j}\phi = {}^{j}_{i}\mathsf{Ad}\ {}^{i}\phi$. Because we write spatial wrenches as column vectors, these covariant quantities are transformed by left multiplying with the inverse transpose, ${}^{j}w = {}^{i}_{j}\mathsf{Ad}^T\ {}^{i}w$.

We also define $\Gamma(r)$, a 3×6 matrix for computing the linear velocity of a point r for a given spatial velocity. That is, $\dot{r} = \Gamma(r)\phi$ where

$$\Gamma(r) = \begin{pmatrix} [-r] & I \end{pmatrix}. \tag{4.1}$$

Lastly, we define the spatial cross product of $\phi = (\omega^T, v^T)^T$ as the linear operator with coordinate matrix

$$[\phi] \quad = \quad \begin{pmatrix} [\omega] & 0 \\ [v] & [\omega] \end{pmatrix}.$$

As such, the Newton-Euler equation for a rigid body can be written in body coordinates as

$$w = M\dot{\phi} - [\phi]^T M\phi, \tag{4.2}$$

where M is the mass inertia matrix, ϕ is the spatial velocity, w is the wrench acting on the body, and all quantities are in body coordinates.

For convenience, Table 4.1 summarizes the variables we use to describe the compliant hand model and our synthesis method.

4.1.1 Jacobian

For a given configuration of an articulated structure, the Jacobian is a linear transformation from joint velocities to the spatial velocity of a given link. For convenience, since our kinematic structure is a tree rather than a chain, we write the Jacobian as a matrix, J, where the columns consist of *all* the twists in *world coordinates* caused by the changes of all degrees of freedom. That is, each twist is the spatial velocity that results from an angular velocity of 1 rad/s at the joint. Given a joint velocity, $\dot{\theta}$, the spatial velocity of body i will be the sum of only the twists on the path between body i and the root. We select these joints affecting the motion of body i using a diagonal matrix, S_i, with ones and zeros on the diagonal. Thus, $JS_i\dot{\theta}$ gives the spatial velocity of body i in world coordinates. Further, the velocity of a point r_i on body i (e.g., a contact point) is given by $\Gamma(r_i)JS_i\dot{\theta}$, or simply $\dot{r}_i = \mathsf{J}_i\dot{\theta}$ where

$$\mathsf{J}_i = \Gamma(r_i)JS_i. \tag{4.3}$$

Likewise, the Jacobian transpose maps wrenches to joint torques. Given a linear force f_i at a contact point r_i, we can compute

$$\boldsymbol{\tau} = \mathsf{J}_i^T f_i. \tag{4.4}$$

4.1.2 Compliant Articulated Structure

Our compliant hand model assumes that each joint of an articulated model has a torsional joint spring that holds a preferred configuration, with an additional 3 linear

Variable	Description
n	degrees of freedom in the kinematic chain including the translation at the root
K	$n \times n$ matrix of joint stiffnesses
C	$n \times n$ matrix of joint compliances
J	$6 \times n$ Jacobian matrix giving world coordinate twists for each DOF
S_i	$n \times n$ diagonal matrix selecting the DOFs on the path from body i to the root
θ	equilibrium configuration
θ_r	reference configuration
$\boldsymbol{\tau}$	joint torques including the linear forces at the root
r_i	contact location in world coordinates on body i
x_i	contact location in world coordinates on the object and corresponding to r_i
ϕ	spatial velocity of the object
q	configuration of the object (position and orientation)
f_i	linear force applied at the contact point on body i
J_i	Jacobian matrix for contact point r_i
N	current number of contacts
$\Delta\mathsf{u}$	size $3N$ block vector of small changes in position (displacements) at current contacts
$\Delta\mathsf{f}$	size $3N$ block vector of small changes in linear forces at current contacts
G	$3N \times 3N$ combined effective compliance matrix

Table 4.1: Description of variables.

springs at the root (one for each linear axis). We can write this as

$$\boldsymbol{\tau} = K(\theta - \theta_r), \tag{4.5}$$

where K is the stiffnesses of the joints (and the linear root springs), θ is the current configuration, and θ_r is the reference or rest configuration. Note that θ includes the root translation and $\boldsymbol{\tau}$ contains linear forces. As some joints may be assigned infinite stiffness, we instead write Equation 4.5 with a compliance matrix C (i.e., the inverse of the stiffness matrix K). Given $\boldsymbol{\tau}$, a torque which includes linear force at the root, the equilibrium configuration, θ, is given by

$$\theta = \theta_r + C\boldsymbol{\tau}. \tag{4.6}$$

4.1.3 Effective Compliance

The compliant hand model acts much like a generalized spring, where internally the torques due to forces at contact points (computed with Equation 4.4) and the angular displacements are always in equilibrium. Furthermore, we can describe the behaviour of a point rigidly fixed to a body in the articulated structure based on the displacement of the point caused by a small force applied at the point. This is called the effective compliance (inverse of effective stiffness) and is given by a 3×3 matrix.

A small force, Δf_i, when applied to the compliant kinematic structure at a contact point, r_i, results in a displacement, Δu_i, at the contact point. The force at point r_i maps to a wrench via $\Gamma(r_i)^T$. This wrench on body i maps to internal torques via J^T, the Jacobian transpose, combined with joint selection matrix, S_i (which zeros torques at non-affected joints, i.e., those not on the path between i and the root). The torques then map to joint displacements via the compliance matrix (Equation 4.6), and joint displacements map to a twist for body i via JS_i. Finally, $\Gamma(r_i)$ converts the twist to a displacement at the contact point. We can write all this as $\Delta u_i = \mathsf{C}_i \Delta f_i$, where C_i is the effective compliance and is given by

$$\mathsf{C}_i = \mathsf{J}_i C \mathsf{J}_i^T. \tag{4.7}$$

Recall, J_i from Equation 4.3 is for the current configuration θ. This 3×3 effective compliance matrix need not be invertible, for example, if the joints are lined up in a singular configuration. However, we ensure the matrix is full rank by giving the skeleton's root non-zero linear compliance in all directions; adding a small amount of compliance at the root is equivalent to regularization. This is important if we want to solve for forces given displacements, and we explore fingerpad compliance at the end of this Chapter as a source of regularization at the end of this chapter.

With contacts on multiple different links in the kinematic structure, the effective compliance becomes coupled because of shared joints on the path between the contacts and the root (see Figure 4.1). This relationship can be written as a matrix,

$$\mathsf{G} = \mathsf{J}\ C\ \mathsf{J}^T, \tag{4.8}$$

where the matrix J^T (similarly J) is a block vector constructed as

$$\mathsf{J}^T = ({\mathsf{J}_0}^T, \cdots, {\mathsf{J}_N}^T). \tag{4.9}$$

Grouping linear forces and displacements into a block vectors, $\Delta\mathsf{f}$ and $\Delta\mathsf{u}$ respectively, the local linear model of the compliance for the current equilibrium configuration is written as

$$\mathsf{G}\Delta\mathsf{f} = \Delta\mathsf{u}. \tag{4.10}$$

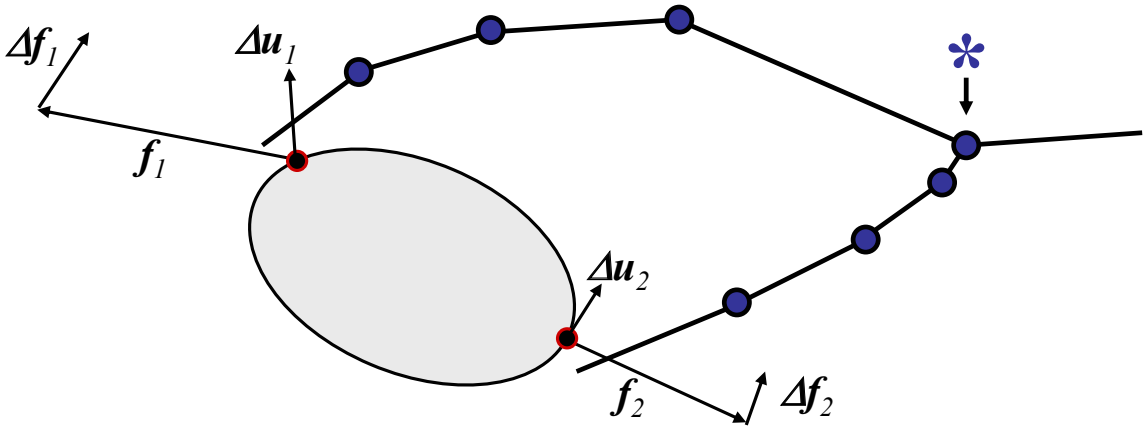


Figure 4.1: The effective end point compliance for multiple contacts is coupled because of shared joints. In this example this coupling is due to the wrist, which is marked with an asterix.

4.2 Joint Compliance Estimation

Most methods of system identification in the previous work use perturbation. That is, an external force is momentarily applied to the subject during the captured motion to help in the estimation of parameters [Xu and Hollerbach 1999; Hajian 1997; Hasser and Cutkosky 2002; Milner and Franklin 1998; Gomi and Kawato 1997]. These methods can be used to estimate various properties of the system, such as joint stiffness, the effective endpoint compliance, or system parameters for a second order model (i.e., inertia, damping, and stiffness).

We measure the system without perturbation. This approach is desirable because perturbing the motion not only complicates the capture process, but also unfortunately changes the motion that we are trying to measure. Subjects may change their motion to account for the perturbations, making it difficult to capture natural motions. Instead, our approach makes estimates of the compliance of the joints based on observing the captured motion and forces at the time of contact. An overview of this estimation process is as follows. Forces measured at the contact points provide the wrenches acting on the different links of the kinematic structure. From these wrenches, the internal torques at the joints can be measured. Smooth trajectory approximations are comuted for the joint angles and torques before and after contact. Finally, from these smooth trajectories we estimate the stiffness of the joint at the time of contact. The rest of this section provides additional detail on these steps.

Identification of the exact time of contact is straightforward given the force measurements. Traditional motion capture can use both automatic and manual methods to identify contact events (see [Bindiganavale and Badler 1998; Kovar et al. 2002b]), but these methods are unnecessary given measured contact forces. We first automatically identify contacts with a hysteresis function. Then we manually note the first

onset of force, and the time at which the force starts to increase smoothly after the initial vibrations. The vibrations are likely due to the sensor housing, and typically occur only during the first few hundredths of a second after contact (see Figure 4.3 for example).

We estimate internal torques for the entire trajectory based on the captured configurations and forces. For a wrench measured on a link in the kinematic chain, we use the Jacobian at the current configuration to compute the torques experienced at the joints. For some sensors we only have measurements of one component of the force, as opposed to a full wrench. In this case we must compute an assumed force measurement direction and location for each contact. We do this based on sensor placement on the finger in the case of the force sensitive resistors (i.e., on an angle for the thumb, aligned with the normal of the center of the fingerpad for the index finger, see Section 3.2.2), or based on fingertip positions in relation to the sensor surface in the case of the Tango (see Section 3.3). Multiplying the Jacobian transpose of the current configuration by these forces gives us the internal torques (Equation 4.4). Figure 4.2 shows an example of raw captured angles and forces, along with computed torques, for a sequence of grasps with a small box (similar to the trial shown in Figure 5.5). The data was collected at 500 Hz using the fingertip-mounted force sensitive resistors.

At each contact event, we fit a low degree polynomial to the joint angle trajectory in a small window, just before the time of contact, and just after the contact vibrations. That is, using the Matlab function `polyfit`, we find

$$\theta^-(t) = \sum_{i=0}^{d} a_i^- t^i, \quad \theta^+(t) = \sum_{i=0}^{d} a_i^+ t^i, \quad \tau^+(t) = \sum_{i=0}^{d} b_i^+ t^i, \tag{4.11}$$

where $t = 0$ at the time of contact, d is equal to 1 or 2, and pre- and post-contact are denoted by the $-$ and $+$ superscripts. These low degree polynomial fits serve as a smoothed version of the data from which we can compute the compliance. Quadratic approximations fit reasonably well when the trajectory data has low noise; when the noise dominates we use a linear fit instead. Both before and after the time of contact we use a window size of 100 ms for data fitting (50 samples at 500 Hz). This time interval is short enough to avoid the inclusion of any reaction due to sensory feedback of the contact event.

The compliance is computed by the joint angle difference divided by the torque, but since there are many samples in the post contact trajectory we have many estimates. Note that computing the average of estimates is undesirable as it gives equal weight to estimates close to the time of contact. Near the time of contact there is little information for a good estimate due to the initially small torques and small joint displacements. Instead, we estimate the compliance by dividing the difference of the joint angle slopes by the slope of the torque at the time of contact, $t = 0$. For

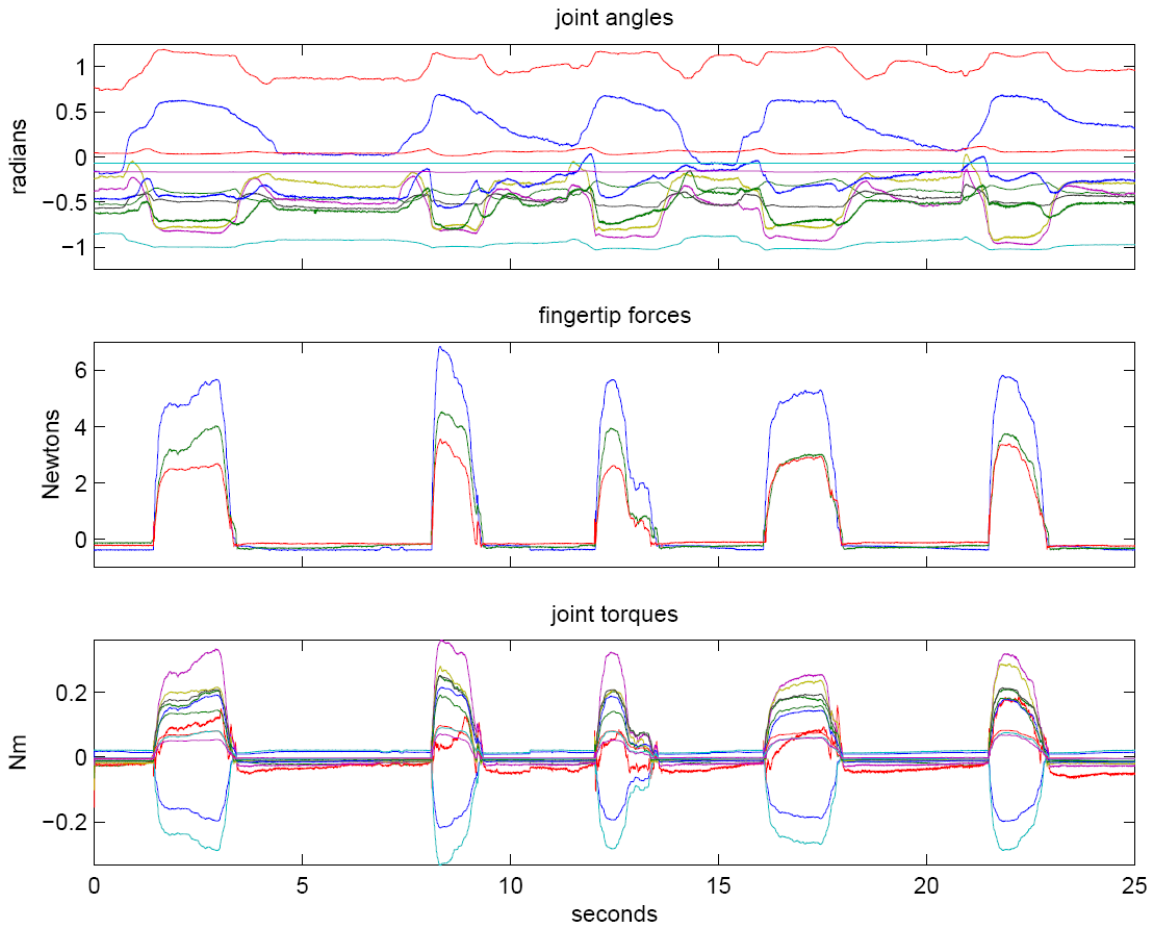


Figure 4.2: Hand joint angles and captured force for a sequence of grasps. Note the large blue force is due to the thumb.

polynomials of degree 1 and greater, the compliance equals

$$c = \frac{a_1^+ - a_1^-}{b_1^+}. \tag{4.12}$$

Figure 4.3 shows a close up of the index MP joint angle and torque at the time of contact for the second grasp. The torques shown here have a constant offset subtracted to zero the pre-contact torques; however, estimation with Equation 4.12 does not require this (it only involves slopes). Likewise, using slopes permits the best-fit joint angle curves to be estimated independently of each other (i.e., they need not be constrained to have the same value at the time of contact, $\theta^-(0) = \theta^+(0)$).

Notice the clear distinction between pre- and post-contact trajectories, and note that they are almost linear. In Figure 4.3, the compliance is approximately equivalent to a 10 degree rotation at the joint for each Newton of force at the fingertip.

This compliance estimation method does not always produce reasonable values for all joints. For example, we get negative compliances for the thumb CM joint, likely

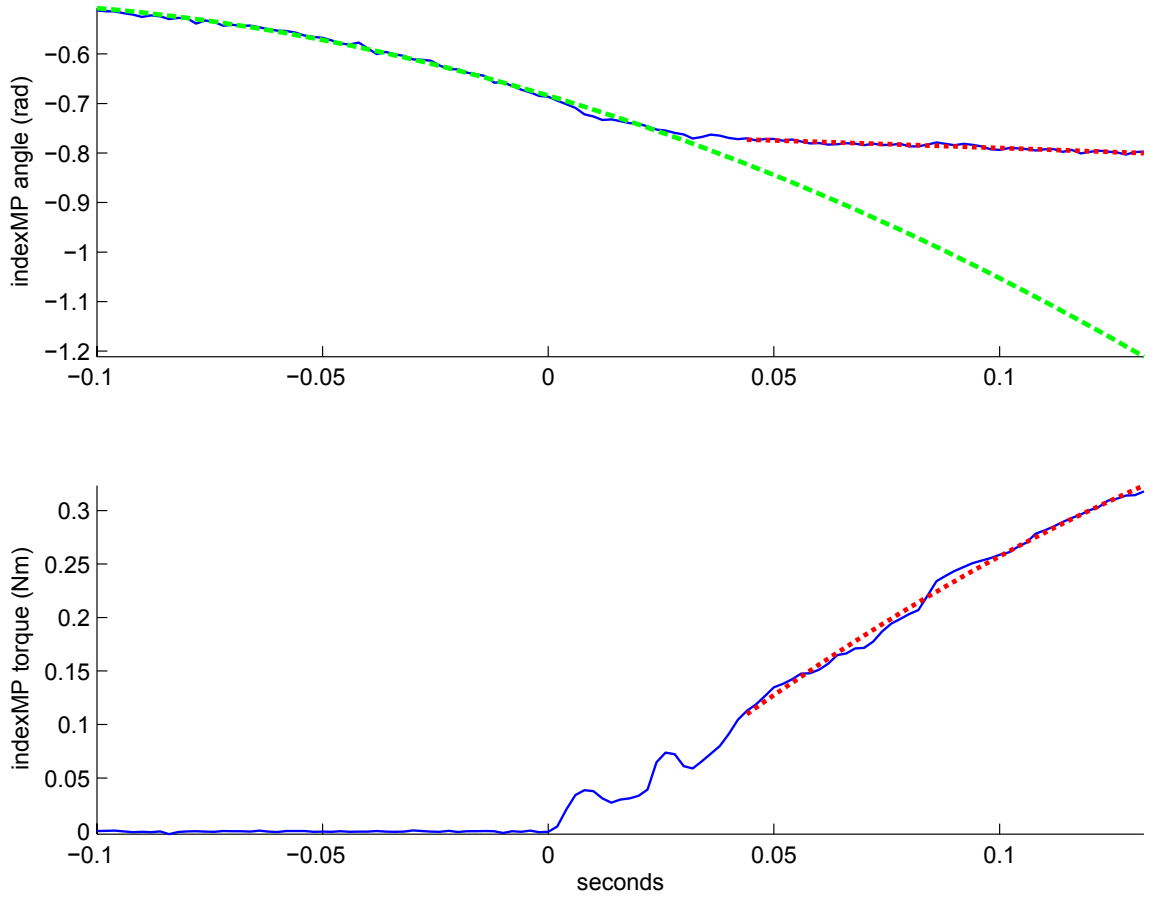


Figure 4.3: Angle and computed torque for index MP flexion at the time of contact with polynomial fit. The pre contact curve and extrapolation are drawn green, while the post contact curves are drawn red. The compliance estimate for this contact is 1.1106 rad/Nm.

because of an incorrect kinematic model (this may also be explained by missing tangential forces as we discuss later). Also, joints such as the wrist have an additional problem; multiple finger plants contribute to the torque, and not all fingers establish contact simultaneously. Polynomials must be fit before contact and after the last finger plant, and substantial gaps between these two times can make for a difficult comparison due to errors in the assumed (extrapolated) trajectory. Lastly, this estimation technique assumes that we are measuring a massless quasi-static compliant kinematic structure. The assumption is reasonable for fingers. However, this may not be the case for larger segments (larger inertia) that are farther along the kinematic chain (and farther from the contact forces).

Table 4.2 provides estimates for 5 different grasps captured with fingertip-mounted FSRs. Note that compliance estimates for the distal joints of the index and middle finger are much lower than we might expect. This can be explained by the very small joint motions that the Vicon software produced when estimating joint angles from

Joint	Grasp 1	Grasp 2	Grasp 3	Grasp 4	Grasp 5	Mean	STD
Thumb MP	0.9480	1.0995	1.1908	0.8046	1.1739	1.0434	0.1643
Thumb IP	0.5178	0.7279	0.2857	0.1063	0.3941	0.4064	0.2348
Index MP	2.0951	1.1106	3.7122	6.1375	0.8587	2.7828	2.1846
Index PIP	1.8806	1.3145	1.4319	1.7980	4.0472	2.0944	1.1174
Index DIP	0.0035	0.0007	0.0013	0.0068	0.0048	0.0034	0.0025
Middle MP	2.4722	2.0180	7.2087	1.7168	2.7850	3.2401	2.2561
Middle PIP	2.1510	2.7479	1.5604	0.2690	5.6582	2.4773	2.0009
Middle DIP	0.1190	0.0753	0.2238	0.0691	0.2927	0.1560	0.0984

Table 4.2: Compliance estimates in rad/Nm for 5 grasps measured with fingertip mounted FSRs, along with their means and standard deviations. The compliance of MP joints is for flexion-extension.

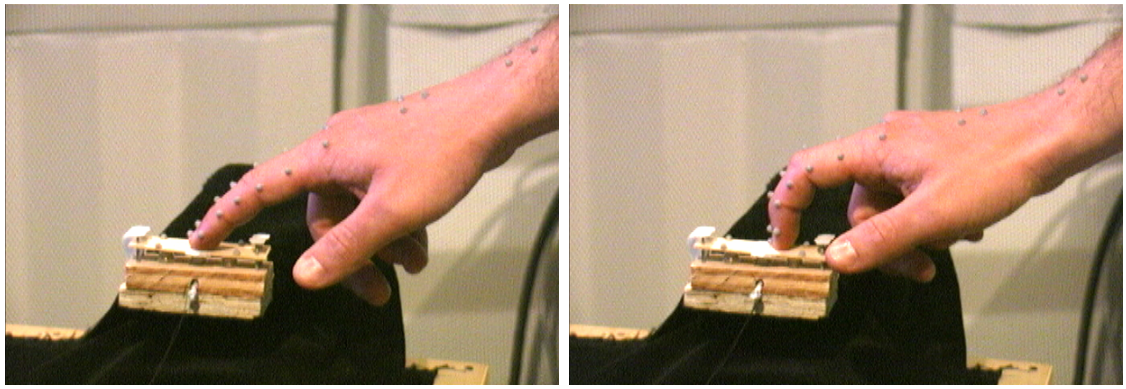


Figure 4.4: Shown left is a frame from the surface exploration trial, while right shows a frame from the surface scratching trial.

marker data; notice the two joint angle trajectories in Figure 4.2 that are almost completely flat.

4.2.1 Single Finger Experiments

Measurements of single finger interactions help validate our estimation procedures. Figure 4.4 shows snapshots of surface exploration and a surface scratching trials. Using a surface instrumented with a force sensor, we compute joint torques using contact force measurements which include both normal and friction forces.

Table 4.3 shows the compliances estimated for surface exploration in comparison to those estimated for surface scratching. The compliances are clearly dependent on the task, with joints tending to be much stiffer during scratching (by about a factor of two). Because of joint angle noise in these trials, the estimates shown here were computed with best-fit linear models of the trajectories (instead of quadratic).

Joint	Exploring	Scratching
Wrist	0.3488	0.1531
MP	0.9882	0.4788
PIP	0.6715	< 0.8426
DIP	2.0052	< 0.8432

Table 4.3: Compliance estimates for exploring and scratching. Units are in rad/Nm and the MP compliance is for flexion-extension.

Erroneous trajectory extrapolation and the high stiffness of PIP and DIP joints during scratching provided few trials (about 1 in 10) where compliances could be estimated; the actual compliance value is likely much lower than shown in the table.

As mentioned previously, compliance estimation with this technique may not always produce reasonable estimates. This is because we are essentially trying to find the equilibrium trajectory through extrapolation, and in some cases, the joint trajectories can be somewhat misleading. Figure 4.5 shows a comparison of two joints for two contacts events, along with approximate equilibrium trajectories (using an assumed compliance of 1 rad/Nm). For the index MP joint, the post-contact equilibrium trajectory can be well approximated by a linear extrapolation of the pre-contact trajectory. In contrast, the PIP joint equilibrium trajectory follows a less obvious path.

Generally, we expect the effective endpoint stiffness to be linearly related to the force [Hajian 1997]. This makes sense from the point of view of the muscles generating the force. In the Hill muscle model, the passive elastic components also have similar relationships, generating higher stiffness at higher tensions, i.e., when producing more force (see [McMahon 1984]). We can look for a relationship between our stiffness estimates and the forces used during an interaction.

At each contact we see the force at the fingertip increase from zero to an approximately steady state in about the first half second. We can consider this steady state force to be the planned contact force. For surface exploration it is typically just under 1 Newton, though it does vary slightly both during the interaction, and across different example contacts of the same task. We take the maximum force from the initial 0.6 seconds of contact as an estimate of the planned force. For the surface exploration trial, the average planned interaction force is 0.8223 N. For the surface scratching task, with stiffer joints, we have an average planned force of 1.3980 N.

We can also compute correlation coefficients to test the relationship between the planned force and stiffness estimate at each contact event. In the surface exploration trial, the index MP compliance estimates and planned forces have a correlation coefficient of 0.5373. For the wrist, the correlation coefficient is 0.3796. This suggests there

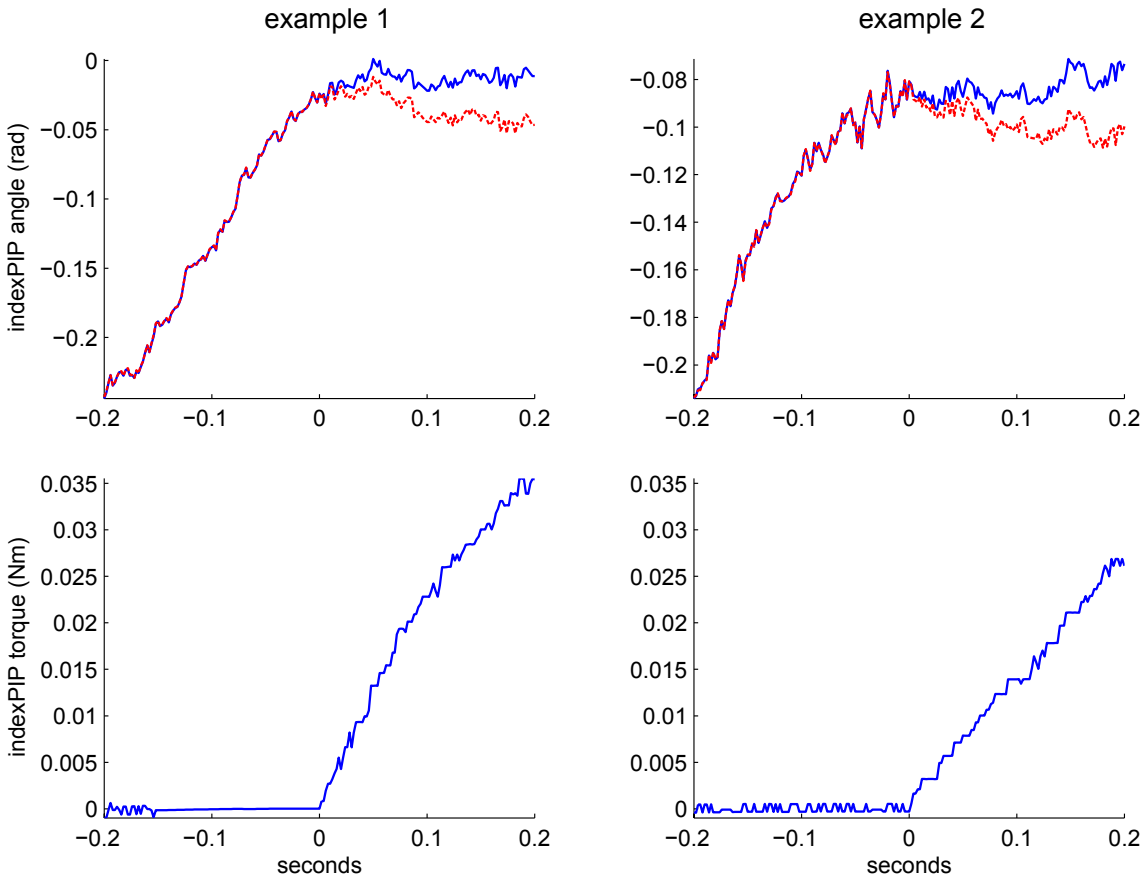


Figure 4.5: Two examples of angle and computed torque for the index PIP joint. Shown in red (dashed) is an equilibrium trajectory for an assumed compliance of 1 rad/Nm.

may be a connection between our estimates produced at the moment of contact and the compliance of joints during the interaction. While we find a negative correlation between the planned force and the DIP and PIP compliances, we also have less faith in our estimates for these joints. Half the contacts in the trial consist of problem cases similar to those shown in Figure 4.5 (i.e., cases for which our method produces negative compliance values), in contrast to those shown in Figure 4.6. Note that in the case of Figure 4.5, the straightening of the joint may play a part; the joint may be quite stiff because it is near a joint limit at zero radians. Additionally, the joint could be near an artificial reconstruction joint-limit (the Vicon reconstruction of a joint angle is constrained by the mean and variance observed in a range of motion trial).

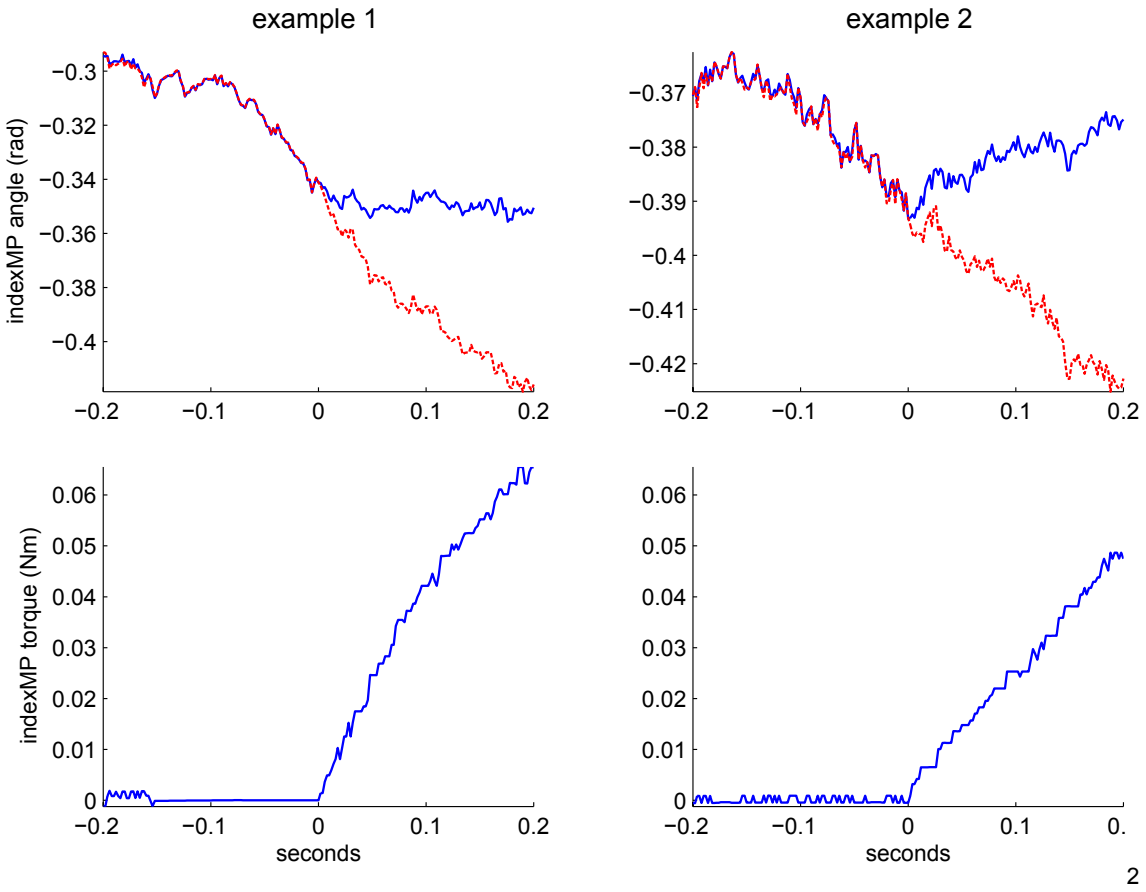


Figure 4.6: Two examples of angle and computed torque for the index MP joint. Shown in red (dashed) is an equilibrium trajectory for an assumed compliance of 1 rad/Nm.

4.2.2 Validation

As mentioned previously, perturbation methods are commonly used to estimate system parameters, such as stiffness. Here, we validate our compliance estimates for the surface scratching and exploring tasks with perturbation data collected during these trials.

We use a small spring loaded platform that is unlatched by pulling a small pin via a piece of thread. Figure 4.7 shows frames of the platform mechanism in action. Note that even though the capture subject is pulling on the string to remove the pin, this is done by slowly applying tension to the string rather than a sharp pull. Because of friction on the pin, the release of the platform is unpredictable. The resulting sudden impulse of force is surprising to the subject, and he subsequently removes his finger from the platform. Figure 4.8 shows the fingertip elevation and vertical force during the perturbation trials, from which we can see the reaction time is just less than 100

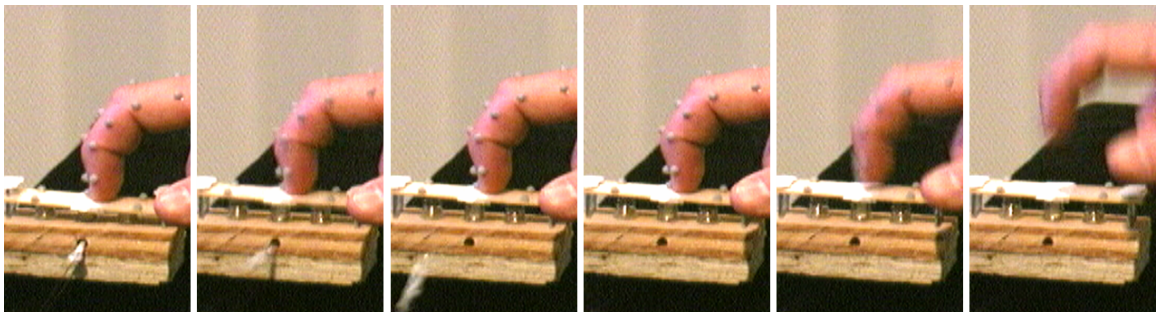


Figure 4.7: Video frames at 30 Hz showing perturbation. The pin has unlatched the platform in the second frame, and the perturbation is complete by the third frame. In the fifth frame, the subject starts to remove his finger quickly. Note that the thumb (foreground) never makes contact with the platform.

ms.

In our world coordinate system, the platform generates a perturbation force in the vertical, or positive z, direction. We average the z coordinate of the 5 motion capture markers on the distal phalanx to produce a position trajectory; numerical differentiation provides velocities and accelerations (seen in Figure 4.9). With this data, we fit parameters (as described below) for a second order system,

$$f(t) = m\ddot{z}(t) + b\dot{z}(t) + kz(t) + f_0, \tag{4.13}$$

where f_0 is the rest force at the time of perturbation. Note that we clamp negative force measurements to zero for model fitting because the platform cannot pull down on the fingertip. The negative forces measured by the sensor (seen in Figure 4.8) are likely due to vibration in the wooden plank (to which the spring-loaded platform is attached).

Similar to [Asada and Asari 1988; Hasser and Cutkosky 2002], we use least squares to find best-fit parameters. We want parameters such that Equation 4.13 is satisfied at every point in time. Four samples provides four equations, which is enough to solve for the four parameters, however, better parameter estimates are obtained by solving the least squares solution for a larger number of samples. We use a small window of samples starting at the time of perturbation, and we take care not to use too few or too many samples. If we use too little data then the estimates will be poor due to measurement noise (and potentially nonsensical, such as negative mass). In contrast, for larger windows the assumption of a constant set of parameters introduces error. For example, the rest force f_0 is not constant since the finger is in the process of performing a task at the time of perturbation. Because of this, we select windows that focus solely on the excitation of the system, and we observe this window is larger in the less stiff surface exploration data (it takes longer to come to rest). The estimates for different window sizes can be seen in Figure 4.10, while Table

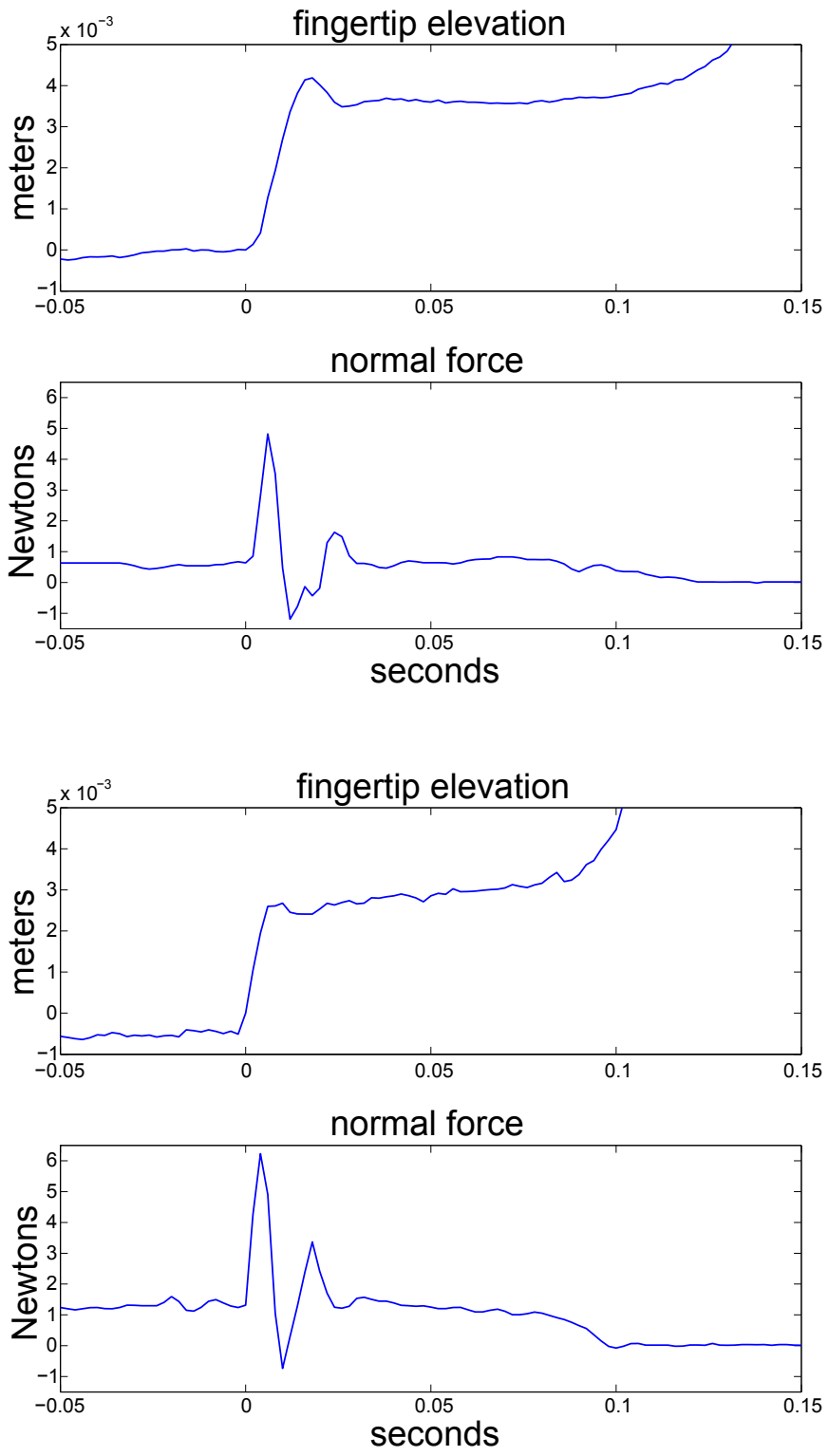


Figure 4.8: Fingertip perturbation data showing reaction time of about 100 ms. Fingertip elevation and force are shown for two different interactions; surface exploration is shown in the top two plots while scratching is shown in the bottom two plots.

Parameter	Exploring	Scratching
f_0	0.6771 N	1.3450 N
m	0.0169 kg	0.0181 kg
b	2.0398 Ns/m	3.8573 Ns/m
k	89.1642 N/m	257.0353 N/m
k_{G}	81.8599 N/m	285.9594 N/m

Table 4.4: Estimated second order system parameters, and the effective endpoint stiffness in the vertical direction computed from joint compliance estimates, k_{G}.

4.4 provides the estimates for selected window sizes (20 ms or 13 samples for touching, 24 ms or 11 samples for scratching). Figure 4.11 shows a comparison of measured and reconstructed forces for the selected parameters. Table 4.4 also includes the effective endpoint stiffness, computed from the joint compliances in Table 4.3 using the posture at the time of perturbation; the stiffness estimated via perturbation matches surprisingly well with our joint compliance estimates.

We can compare our estimated second order system parameters to those of Hajian [1997]. We notice that the mass estimates in Table 4.4 are somewhat higher than the approximately 6 grams reported by Hajian [1997]. However, our mass estimate should be larger as we are not measuring the fingertip alone. Hajian used straps to fix the wrist and forearm to a rigid table, which effectively makes the palm, wrist, and arm a mechanical ground.

Both Hasser and Cutkosky [2002] and Hajian [1997] find the damping coefficient to be proportional to interaction force. Hajian reports fingertip damping estimates of approximately 2 Ns/m at 2 N. Our damping coefficient estimates are in the same range, and agree with the observation that the coefficient should increase with larger interaction forces. Lastly, for different subjects, Hajian [1997] reports stiffness measurements between 41.6 N/m and 237 N/m at 2 N, with mean of 137 N/m. Taking into account the difference in interaction force (their 2 N versus our 0.7 N), this appears to agree fairly well with estimates from our surface exploration trial, for which the finger posture is similar.

Joint compliance estimates may also be compared with values reported in previous work. Looking specifically at the MP joint, Milner and Franklin [1998] report a stiffness of 7.25 Nm/rad ($-x$ direction in their Table IV) for the extended finger. The posture in our surface exploration task is similar, however, our estimates are somewhat lower at 1 Nm/rad. Their estimate corresponds to a force of 10 N at the fingertip, and a torque of 0.8 Nm at the joint. This is approximately 10 times larger than our joint torque of 0.08 Nm, corresponding to the planned fingertip contact force of 0.8 N (during surface exploration). Given the larger fingertip forces in their study,

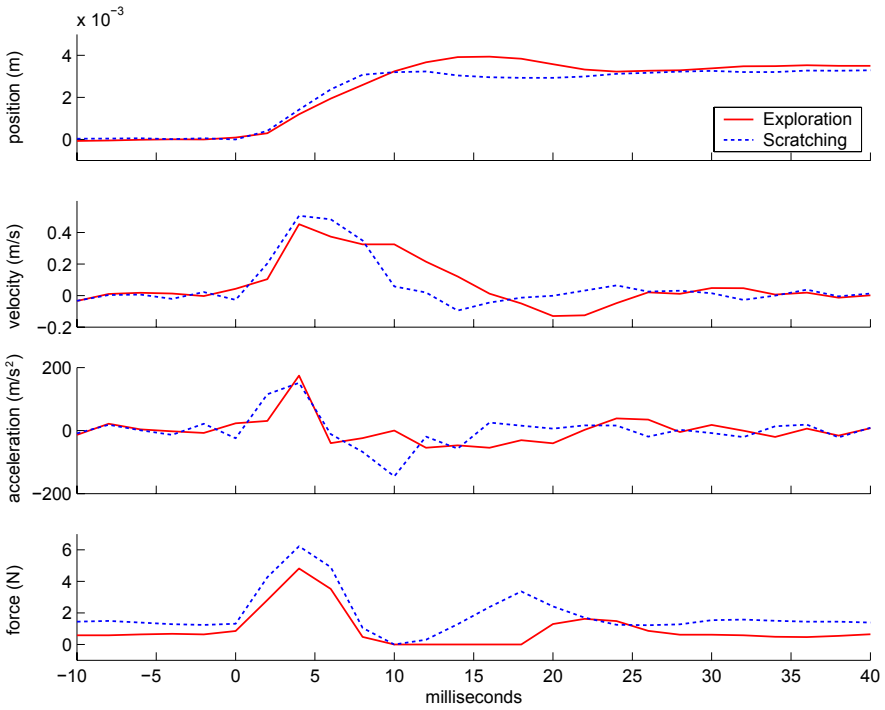


Figure 4.9: Perturbation data for estimation. Derivatives of position provide velocity and acceleration, while forces are measured (negative forces clamped to zero).

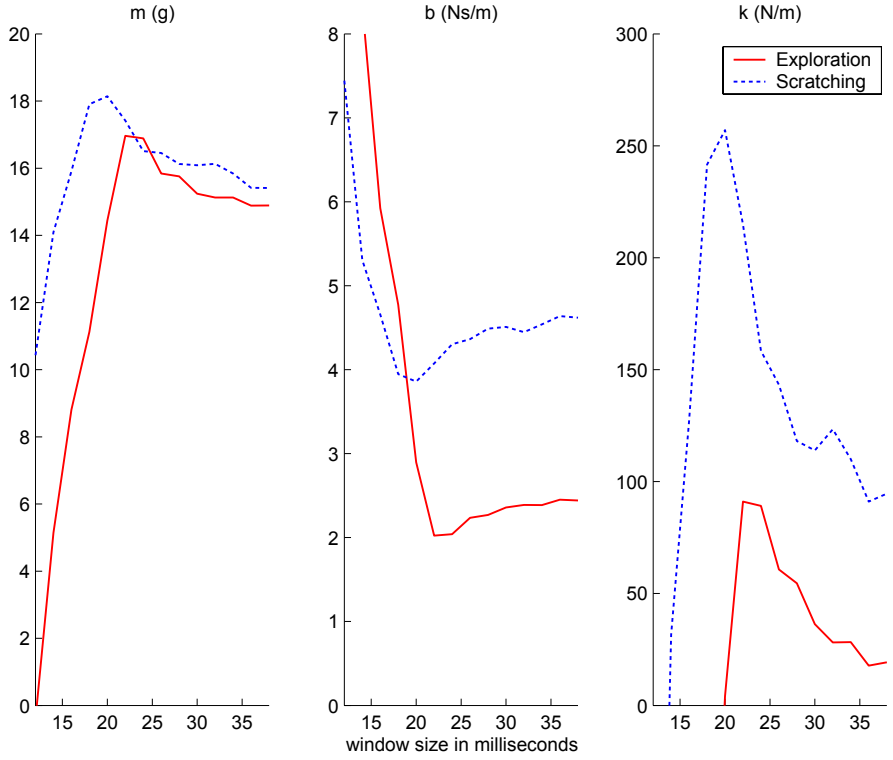


Figure 4.10: Parameters estimated with different window sizes.

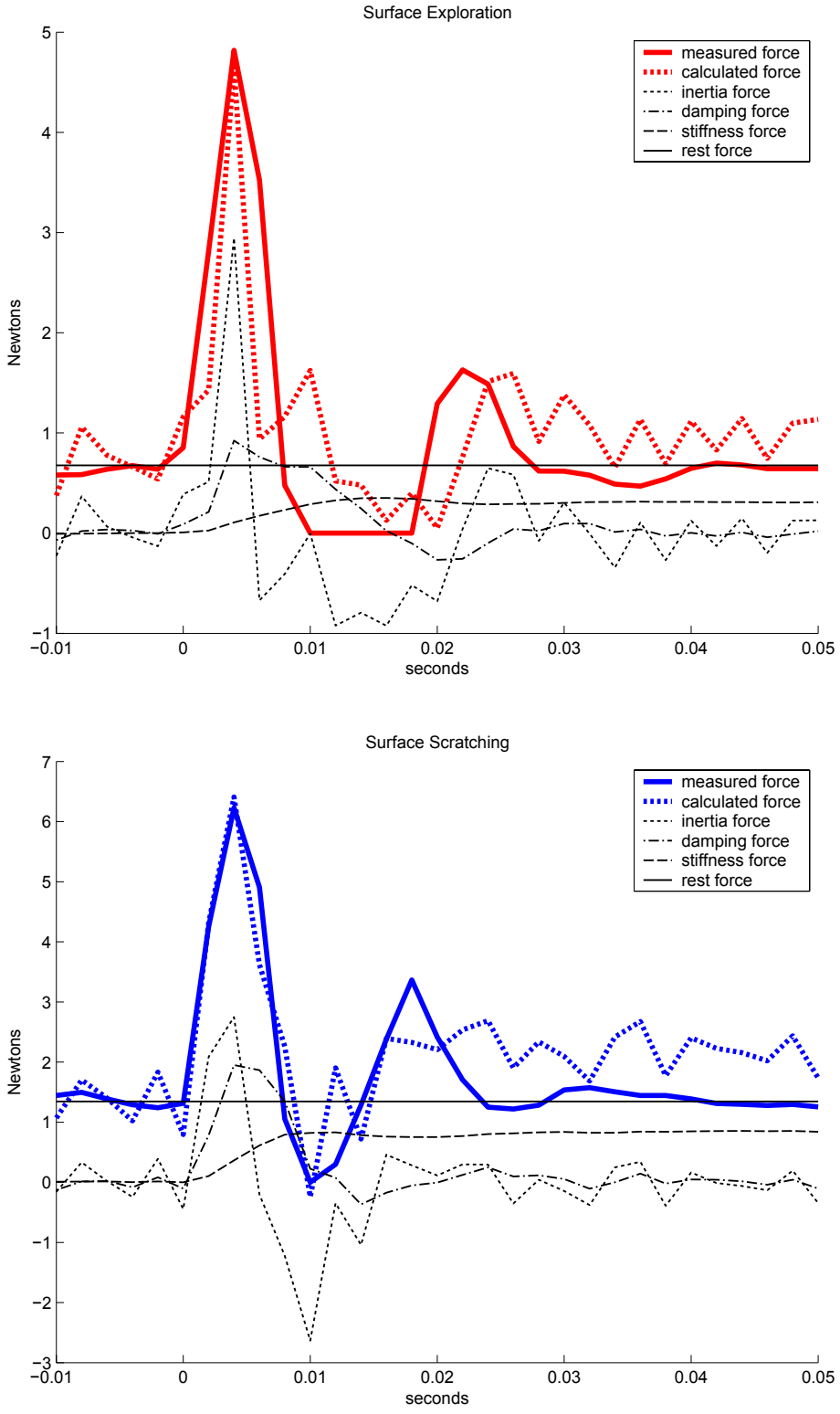


Figure 4.11: Comparison of measured force and the force calculated from the estimated system parameters.

the factor of 7.25 is not inconsistent with expectations. Milner and Franklin [1998] also report an effective endpoint stiffness of 1000 N/m for the extended finger under 10 N of load, which they note is approximately double the value estimated by Hajian [1997] at this load. For a relaxed finger, Milner and Franklin [1998] report an effective endpoint stiffness of 283 N/m.

4.2.3 Discussion

Joint compliances computed with our approach capture the task dependence that is important for realistic animation. We also observe some correlation between stiffness and force production, which is predicted in the previous work. We find the joint compliance estimates we compute at the time of contact to be consistent with the effective endpoint stiffnesses that we compute with perturbation-based system identification. Furthermore, the higher order (damping and inertia) parameters that we estimate by perturbation agree well with those in the literature.

There also exist some limitations to our approach. Unmeasured tangential (friction) forces can have a significant impact on our estimates of the torques at each joint. For instance, varying the friction force can change the sign of an estimated torque, as the sign depends on whether the total force at the fingertip points above or below the joint's axis of rotation. Measurements from force torque sensors are preferable, but estimating joint torques from the contact wrench is still susceptible to error for very much the same reason. That is, mapping contact forces to joint torques is sensitive to errors in the shape and position of the kinematic structure. The kinematic structure and joint estimates are computed by the Vicon software from marker positions, which can often be incomplete (i.e., needed to be filled using motion capture editing tools). More importantly, because the markers move with the surface of the skin, their movement only approximately reflects the position of the underlying bones. Fortunately, there is room for error in our compliance estimates; compliance allows the hand and fingers to find their own equilibrium during contact, in both the real world and simulated interaction.

Though fingertip-mounted FSRs only provide estimates of the normal force, we may still be able to estimate tangential forces under some situations. For example, we could solve for the tangential forces necessary to explain force closure on a stationary grasped object of known mass. Similarly, we might try to estimate tangential forces at the time of contact with an object resting on a force-torque sensor. In this work, however, when tangent forces are unavailable, we build our compliance estimates from the normal forces alone, and note the potentially large source of error.

Lastly, though we have only looked at estimating joint compliances at the time of contact, we may likewise estimate values at the time of breaking contact.

4.3 Reference Trajectory Computation

Assuming the hand acts like a spring, the current configuration, compliances, and joint torques (estimated from contact forces) give us a means of computing a reference configuration. For a measured equilibrium configuration, θ, and contact force f_i at point r_i on body i, we compute the reference configuration as

$$\theta_r = \theta - \sum_{i \in \text{contacts}} C \mathsf{J}_i^T f_i. \tag{4.14}$$

Since we are only estimating the compliance at the instant of contact, we assume the compliance to be fixed throughout the interaction.

A smooth reference trajectory to match the smoothness of our pre-contact trajectory is desirable, but since the initial collision contact force can contain vibration, we allow an adjustable filtering of the force before computing reference configurations (a mild low pass exponential filter with $\alpha < 0.5$ serves well for this purpose). Alternatively, we could forgo filtering and smoothly blend the extrapolated joint trajectory (computed for compliance estimation) with the trajectory from Equation 4.14 over a small post-contact time period. Then, similarly, we would blend back to the motion capture data in a small period of time prior to breaking contact.

4.4 Fingerpad Compliance Estimation

Compliance at the fingerpad has an important role in interaction. When we grasp an object, our soft fingerpads make conforming contact patches that improve our grasp. The deformation behaviour of the fingerpad is quite complex, and has been studied and approximated with several simplified models in previous work [Gulati and Srinivasan 1997; Pawluk 1997; Birch and Srinivasan 1999]. We estimate fingerpad compliances using a highly simplified model in the direction normal to the fingerpad. Ultimately, our desire for these estimates is because we let the fingerpad stiffness under a small load justify our choice of the small linear compliance that we use to regularize the effective endpoint compliance matrix (see Section 4.1.3 and the end of this section).

We measure fingerpad compression with a straightened finger, for a sequence of touches against a flat rigid surface instrumented with a force sensor. We restrict our compliance estimate to the direction normal to the fingerpad, following Pawluk [1997]. Figure 4.12 shows the measurement setup, which uses the Vicon system to track the position of the fingertip and the rigid platform. The two fingernail markers are averaged to get an estimate of the height of the finger, while the four markers on the platform are averaged to compute the height of the surface. Measuring the position of the platform is important; the wooden plank attached to the force sensor flexes slightly during contact forces. Subtracting these two values gives us

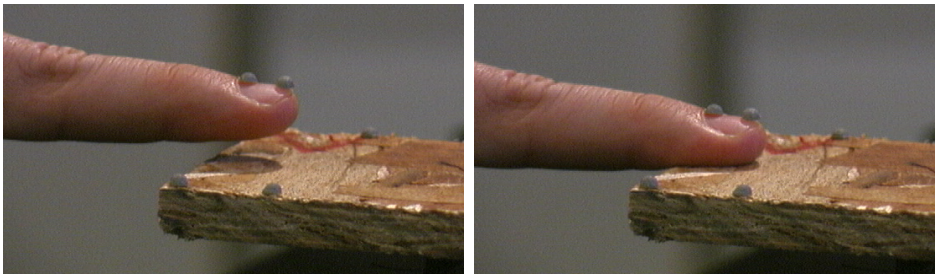


Figure 4.12: Fingerpad compression measured with Vicon motion capture markers and a wooden plank instrumented with a force sensor.

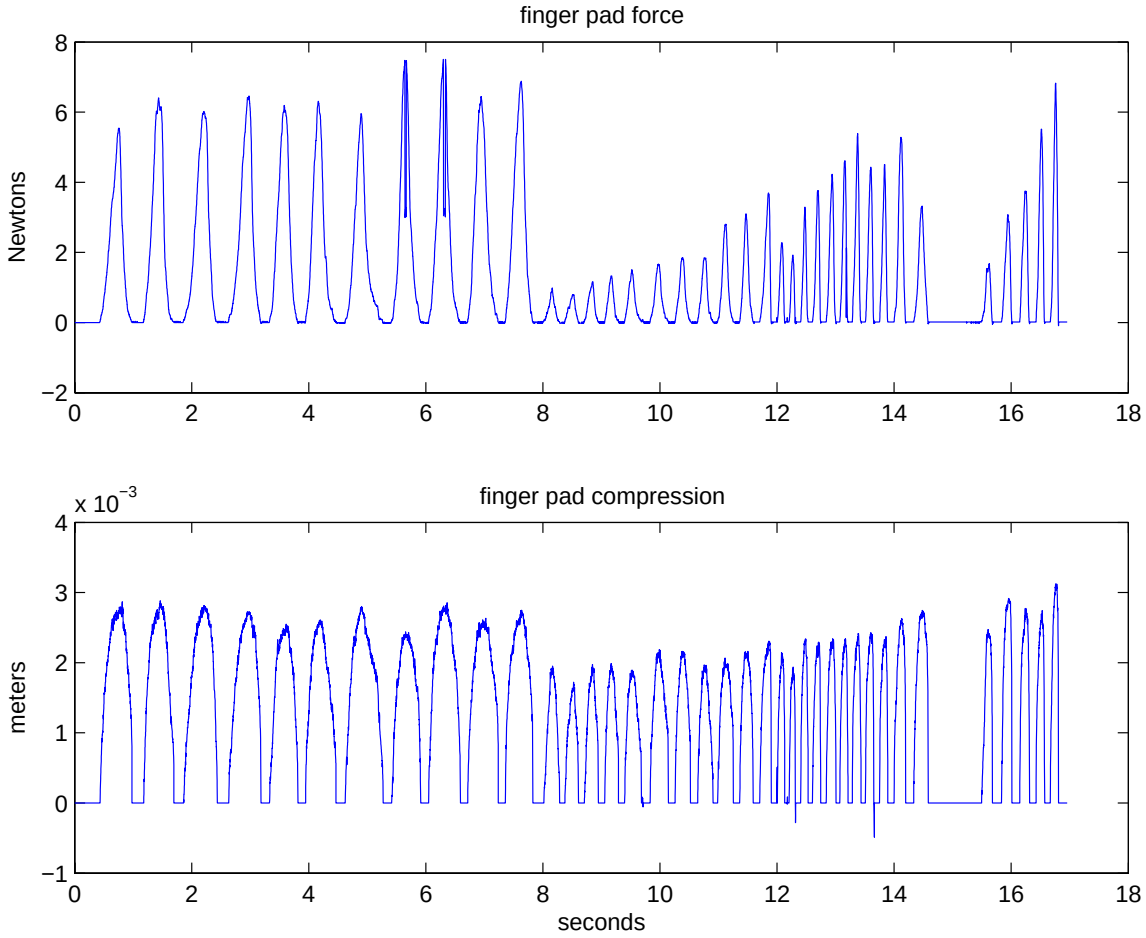


Figure 4.13: Force and displacement data for fingerpad compliance estimation.

the elevation of the fingernail above the platform. During contact, we compute the fingerpad compression (i.e., the deformation displacement) as the elevation minus the initial elevation at the time of contact. When contact forces are zero, we let the compression equal zero. Figure 4.13 shows the force and compression data collected for 38 contacts using the index finger. Because the subject controls the finger position

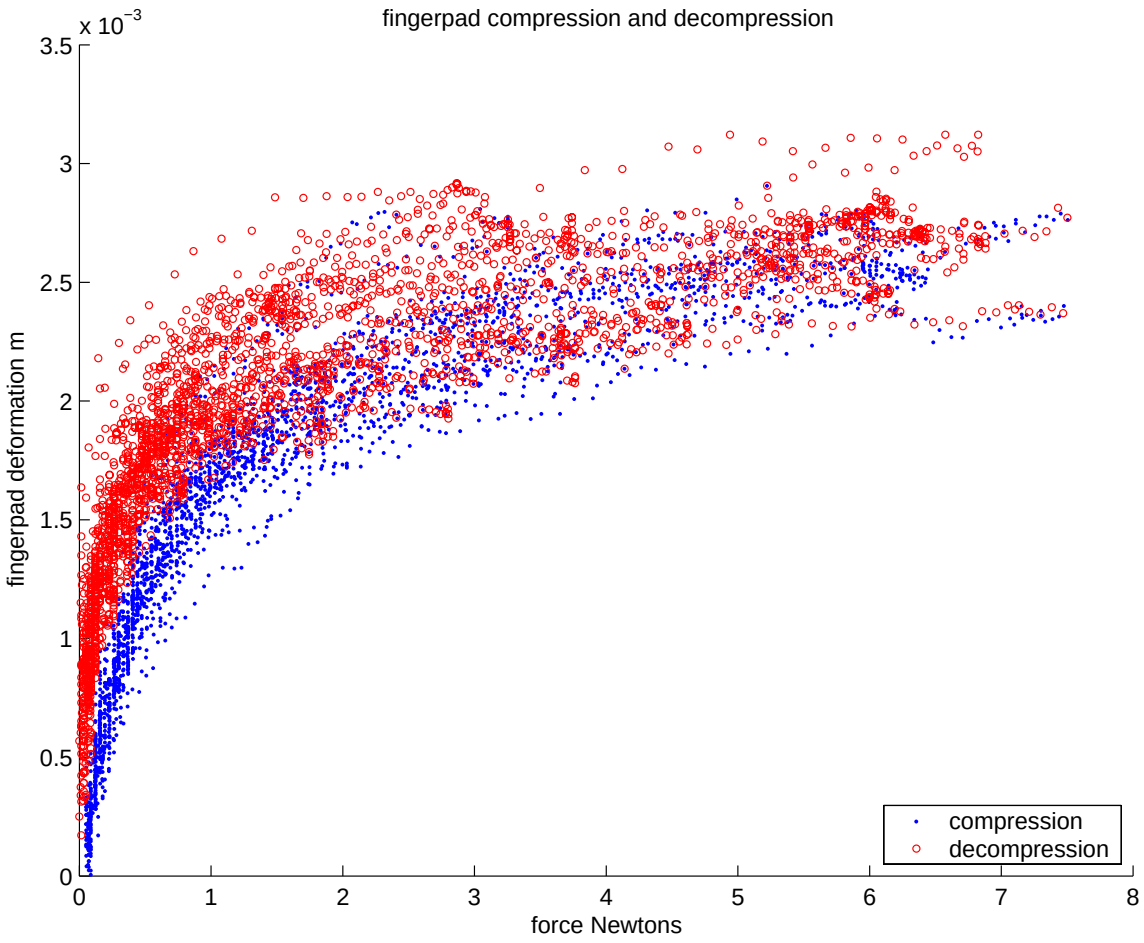


Figure 4.14: Force displacement plot for samples taken during both compression and decompression phases of each contact.

and orientation, there is some variation in each of the contacts.

Figure 4.14 shows contact forces plotted against fingerpad deformation, where blue samples come from the compression phase and red samples come from the decompression phase. Note the difference between the two sets of samples, which is likely due to blood being squeezed out of the fingerpad (and returning before the next compression).

If we approximate the fingerpad as a sphere and assume a simple Hertzian contact model, we can compute the Young's modulus for comparison with previous work. Note, however, that previous work generally puts more faith into more realistic viscoelastic models. Figure 4.15 shows the data for one press and release cycle along with the best fit curve $d = kf^{2/3} + b$, where d is the displacement and f is the force. The Hertz model motivates the form of this curve. Note that our best-fit parameters includes the $f = 0$ intercept, b, as this lets us ignore any offset that might exist in the fingerpad compression data. We find the best-fit parameters for

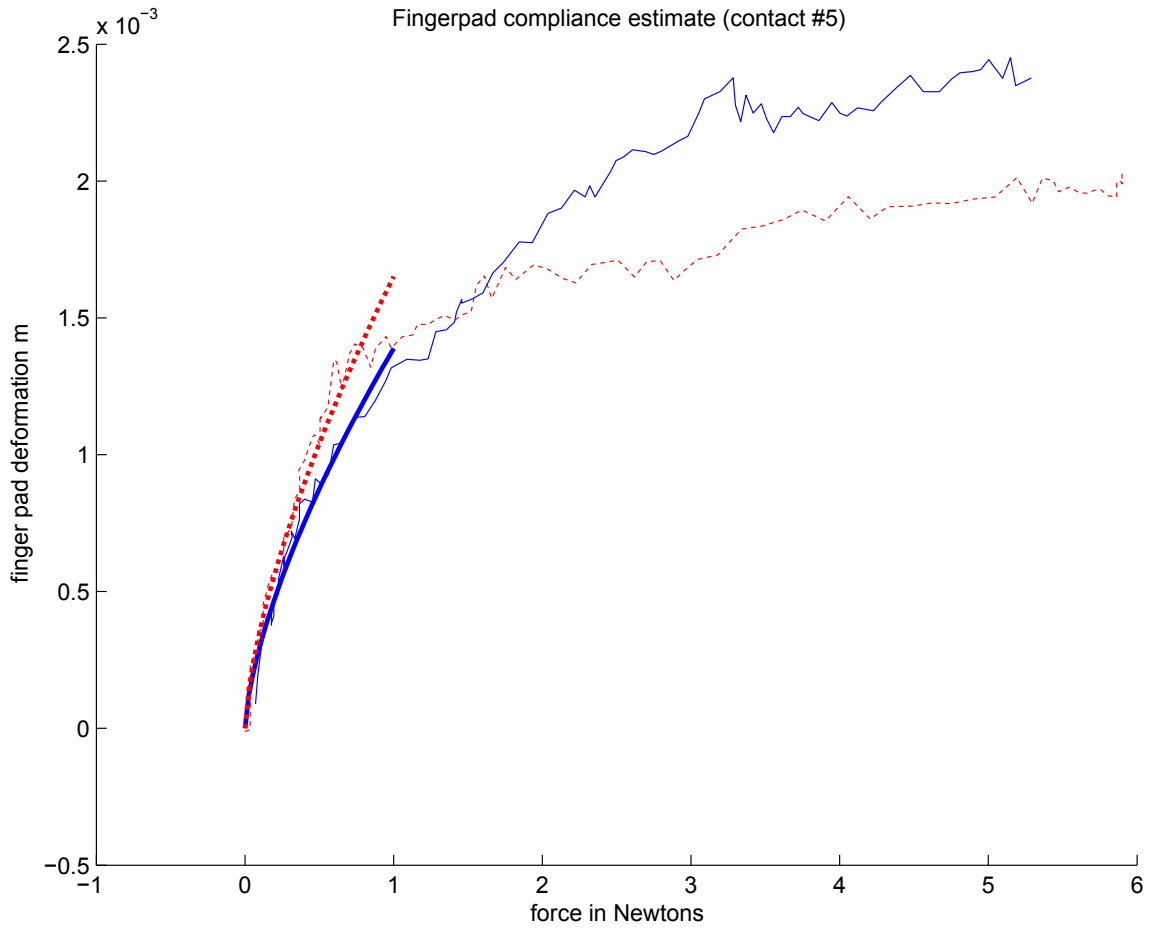


Figure 4.15: Fingerpad force displacement data for a single contact, compression (solid blue) and decompression (dashed red). Hertzian contact model curve shown as thick curves for forces less than 1 N.

samples with a force less than 1 N; the best fits are shown as dotted curves in Figure 4.15 (note that data and curves are corrected, for zero displacement and force, by subtracting b). We only use the displacements for small forces because the Hertzian contact model is only valid for small displacements and forces. From k we compute Young's modulus as $3/(4k^{3/2}\sqrt{R})$, with $R = 7.5$ mm as the radius of the fingerpad (see [Johnson 1985]). This assumes that Poisson's ratio is 0.5 (i.e., incompressibility) following usual practice, though this may not be true [Birch and Srinivasan 1999].

During compression, the mean value of Young's modulus is 120.18 kPa with standard deviation 21.883 kPa, while for decompression the mean is 183.86 kPa with standard deviation 68.256 kPa. A Young's modulus estimate of 250 kPa is reported by Park et al. [2003], while Birch and Srinivasan [1999] report a value of approximately 200 kPa at 2 Hz.[1]

[1]Birch and Srinivasan [1999] provides impulse response shear and bulk modulus estimates at

Alternatively, we can estimate the linear stiffness of the fingerpad while under load. At 2 N we estimate a fingerpad stiffness of 3.96 kN/m with a standard deviation of 2.05 kN/m. Pawluk [1997] cites approximately 4 kN/m at 2 N, Hajian [1997] cites greater than 3 kN/m at 2 N and 300 N/m at contact, and Gulati and Srinivasan [1995] reports a steady state fingerpad impedance of 1.25 kN/m under a small load (less than 0.5 N).

Though we do not use a model of fingerpad deformation in our interaction simulations, the compliance of the fingerpad can effectively serve as the source of, or a replacement for, the small linear compliance we use in our effective endpoint compliance matrix. Note that the estimated stiffness is only in the normal direction. Though we assume an isotropic stiffness for regularization, with additional measurements we could likewise estimate the stiffness in the tangential directions.

4.5 Summary

This chapter has shown a new method for estimating the compliance of hand joints during a captured interaction by examining the capture data at the time of contact. The compliance estimates let us create a non-contact reference trajectory. We then use the estimates and reference trajectory to resynthesize captured interactions in new dynamic environments. The resynthesis method, which is described in the next chapter, indirectly makes use of our fingerpad compliance estimates by adding a small linear compliance at the wrist of our compliant kinematic structure. This is equivalent to Tikhonov regularization of the effective endpoint compliance matrix as described in Section 4.1.3.

various frequencies, from which Young's modulus and Poisson's ratio can be computed.

Chapter 5

Interaction Synthesis

This chapter describes our method for playing back the reference trajectory against a new simulated environment. This resynthesis of the captured interaction, via simulation, produces new movement in a manner inspired by human motor control. The main difference in our approach is that we do not model how neural feedback alters the sensorimotor program. For an example of this, consider an interaction simulation that produces a grasp motion but with an object slipping through the fingers. This can happen if the object is too heavy for the interaction forces, or if the finger plants are not in locations appropriate for the formation of a stable grasp. The reaction required to regain control of the object may be quite simple or quite complex; reactions in humans may involve simple grip tightening responses, finger gaiting, or even a change in grasping strategy via a new sensorimotor program (possibly selected in a conscious decision). Because of this, we chiefly examine the case of motions where tactile feedback does not substantially alter the planned motion. This assumption can be reasonable in the case of well-learned or practised motions. It is possible for a motor program to operate with effectively no changes due to feedback provided the feedback exactly matches predictions (see [Johansson 1996] concerning discrete event sensor-driven control). We demonstrate, however, one example where our method can accommodate a simple adjustment, specifically the case of a grip tightening response due to slip (see Section 5.2 and Figure 5.8).

Our resynthesized interactions use the compliance and equilibrium point trajectories estimated in the previous chapter. Changes to the environment will result in changes in the interaction. For example, we can adjust the surface shape, the friction, or the mass of the interaction object. The resulting motion will effectively preserve the intent of the captured interaction.

In the synthesis of interactions, we use the same compliant hand model that we presented in Chapter 4 for estimating compliance and the reference trajectory. During simulation, this compliant hand model acts like an elasto-static generalized spring while interacting with a new dynamic object, as we move it through its reference trajectory angles.

The contact between the fingertips and the environment that occurs during precision grasps results in an area contact (or contact patch) because of deformation of the fingertips. We simplify interaction simulation by assuming the contacts are between rigid bodies. That is, the geometry associated with each link of the kinematic structure is rigid. This lets us assume a single point of interaction for each contact between a convex object and a convex fingertip. Nevertheless, our synthesis method can be easily extended to include fingerpad deformation as is mentioned briefly in Section 5.1.5.

Compliance values can be altered, from those that we estimated, to change the behaviour of the simulated hand. Likewise, if estimates were not available for a trial, we can use compliances estimated from other sensors in different trials. This might happen if a sensor does not provide clean enough data for computing compliance estimates (for example, the Tango).

The following section describes a method for stepping the state of a compliant articulated system forward when it is in contact with a dynamic object. We maintain an approximation of the equilibrium configuration at each time step, while using computed forces to advance the dynamics of the object.

5.1 Algorithm

The problem of advancing the system while maintaining equilibrium can be addressed as a problem of computing the interaction forces. Our solution of the new forces accounts for friction and breaking contact by solving a linear complementarity problem. This is a hard problem because of the pose-dependent compliant coupling between different fingers. To address this we use a linear approximation of this coupling, and incorporate it into a commonly used framework for friction with multiple contacts. We also take inspiration from the approach to quasi-static deformable contact presented by Pauly et al. [2004]. The result is a novel formulation that handles multiple contacts, breaking contact, friction, and a quasi-static compliant articulated structure. Note that quasi-static compliance is an important feature of our method as it lets our captured interaction dictate the dynamics of the articulated structure. The interaction forces we compute maintain equilibrium of the internal torques and joint displacements, preserving the nuances of the original motion. Although the compliant structure is quasi-static, we can still synthesize interesting new dynamic interactions; interaction forces are applied to objects in the environment, and produce motion through integration of the Newton-Euler rigid body dynamics equations.

The initial state of the system at time k is given by the following variables.

θ^k	current configuration of the compliant structure
θ_r^k	reference configuration (from the non-contact trajectory)
f_i^k	linear force on finger at contact i
r_i^k	position of contact i on the compliant structure
x_i^k	corresponding position of contact i on the rigid body
C^k	current joint compliances
ϕ^k	spatial velocity of the rigid object
q^k	current configuration of the rigid object

Note that the quasi-static compliant structure is *approximately* maintained in equilibrium. As such, the equality

$$\theta^k - \theta_r^k + \theta_{err} = \sum_i C^k \mathsf{J}_i^{k^T} f_i^k \tag{5.1}$$

will hold for a small error in joint angles θ_{err}, where J_i is the Jacobian for contact r_i and configuration θ^k (see Equation 4.3). This small error exists because of updates in the contact location and changes in the configuration (computed via locally linear approximation).

Ultimately, we want to find the forces, f_i^{k+1}, that will both evolve the state of the rigid body forward (giving us ϕ^{k+1} and q^{k+1} upon integration in time), and let us find the θ^{k+1} necessary to keep the compliant structure in equilibrium given new reference angles θ_r^{k+1}.

5.1.1 Friction and Breaking Contact

We first define vectors at each contact point that let us describe the friction cone. This construction is similar to those used in complementarity formulations of rigid body friction (see, for example, [Lotstedt 1981; Baraff 1994; Stewart and Trinkle 1996; Anitescu and Potra 2002; Lloyd 2005; Cline and Pai 2003; Miller and Christensen 2003]). We define d_{i0} as the outward unit normal at contact point i and a set of paired unit tangent vectors d_{ij}, $j = 1..2m$, such that $||d_i|| = 1$, $d_{i(2j-1)} = -d_{i(2j)}$, $j = 1..m$ (see Figure 5.1). We assemble these vectors into a matrix $D_i = (d_{i0}, \cdots, d_{i(2m)})$, as well as a matrix consisting of just the tangent vectors, $D_{i\star} = (d_{i1}, \cdots, d_{i(2m)})$. Note that we always use the $\star$ subscript to denote "all but the first element".

The Coulomb friction cone is defined as the set of possible forces that can be supported by the frictional surface. We build a polyhedral approximation to the cone using the vectors in D_i. Let β_{i0} be the magnitude of the normal force at contact i, i.e., $\beta_{i0} = d_{i0} f_i$. Given coefficient of friction μ_i, the tangential friction force allowed by Coulomb's Law lies inside a circle in the tangent plane of radius $\mu_i \beta_{i0}$ called the Limit Surface [Goyal et al. 1991]. We approximate this set of allowable friction forces

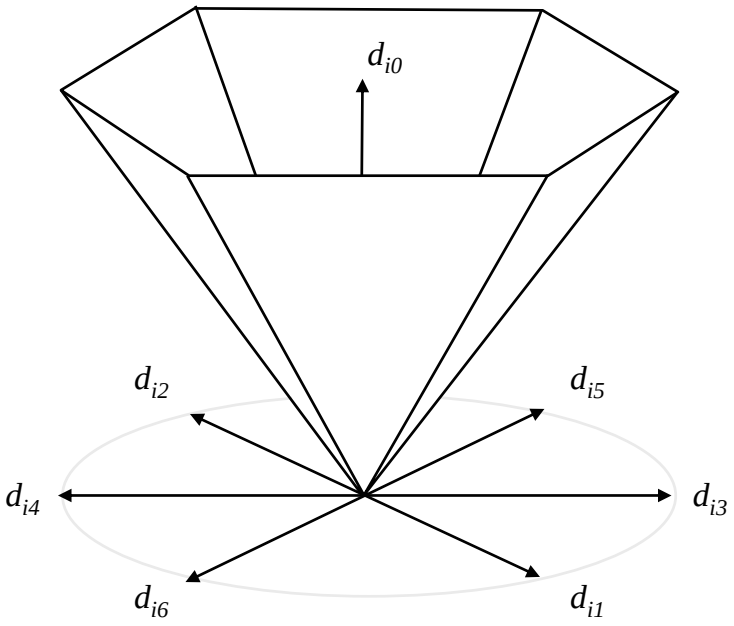


Figure 5.1: An example discretized friction cone with 6 sides ($m = 3$).

by the convex hull of the unit length tangent vectors, $d_{ij} \quad j = 1..2m$, scaled by the normal force times the coefficient of friction,

$$\left\{ \sum_{j=1}^{2m} d_{ij}\beta_{ij} \quad \middle| \quad \beta_{ij} \geq 0, \ \sum_{j=1}^{2m} \beta_{ij} \leq \mu_i \beta_{i0} \right\}. \tag{5.2}$$

The approximation improves as m increases, however, we get reasonable results even with $m = 2$. So, our approximation of Coulomb's law at contact point i can be written as

$$\begin{aligned} f_i &= D_i\beta_i, \\ \beta_{i\star} &\geq 0, \\ F_i\beta_i &\geq 0, \end{aligned} \tag{5.3}$$

where $\beta_i = (\beta_{i0}, \cdots, \beta_{i(2m)})^T$, $\beta_{i\star} = (\beta_{i1}, \cdots, \beta_{i(2m)})^T$, and $F_i = (\mu_i, -1, \cdots, -1)$. Note that the force at the contact i, written $D_i\beta_i$, is actually f_i^{k+1} (i.e., it is the force at the next step for which we are solving). Equation 5.3 gives us a constraint on the force, but we must also constrain the motion to satisfy the principle of maximum dissipation. That is, among allowable friction forces, the actual force maximizes the instantaneous energy dissipation due to frictional work. This minimization of the *negative* frictional work can be written as

$$\begin{aligned} &\min_{\beta_{i\star}} \Delta {u_i}^T D_{i\star}\beta_{i\star}, \\ &\beta_{i\star} \geq 0, \\ &F_i\beta_i \geq 0, \end{aligned} \tag{5.4}$$

where $D_{i\star}\beta_{i\star}$ is the friction force, and Δu_i is the motion at the contact point (a fixed quantity for the purpose of minimization). We can write the Lagrangian for this system as

$$\mathcal{L} = \Delta {u_i}^T D_{i\star}\beta_{i\star} - \nu_{i\star}^T \beta_{i\star} - \lambda_i F_i \beta_i, \tag{5.5}$$

with the KKT optimality conditions (see [Boyd and Vandenberghe 2004]),

$$\begin{aligned}
\beta_{i\star} &\geq 0, \quad \text{(feasibility)} \\
F_i\beta_i &\geq 0, \quad \text{(feasibility)} \\
\lambda_i &\geq 0, \quad \text{(non-negativity)} \\
\nu_{i\star} &\geq 0, \quad \text{(non-negativity)} \\
\lambda_i F_i \beta_i &= 0, \quad \text{(complementary slackness)} \\
\nu_{i\star}^T \beta_{i\star} &= 0, \quad \text{(complementary slackness)} \\
\Delta {u_i}^T D_{i\star} - \nu_{i\star}^T - \lambda_i E_i \star &= 0. \quad \text{(optimality)}
\end{aligned} \tag{5.6}$$

Here, $E = (0, -1, \cdots, -1)$ (i.e., the same as F but with the first entry set to zero). Notice that the optimality condition comes from differentiation with respect to $\beta_{i\star}$, and we can rewrite it as $\nu_{i\star} = D_{i\star}^T \Delta u_i - E_i \star^T \lambda_i$. Defining $\sigma_i = F_i \beta_i$, we can then write the KKT conditions with standard complementarity notation ($a \perp b$ meaning $a \geq 0$, $b \geq 0$, $a^T b = 0$) as

$$\begin{aligned}
\beta_{i\star} \perp \nu_{i\star} &= D_{i\star}^T \Delta u_i - E_i \star^T \lambda_i, \\
\lambda_i \perp \sigma_i &= F_i \beta_i.
\end{aligned} \tag{5.7}$$

To address non-interpenetration, we define s_{i0} as the current linear distance between closest points (or contact points) along the contact normal. This distance is provided by proximity detection, and can equivalently be computed as

$$s_{i0} = d_{i0}^T (r_i^k - x_i^k). \tag{5.8}$$

Recall, d_{i0} is the outward pointing normal at the contact point on the compliant structure. When there is interpenetration this value will be negative, in which case the contact points are selected as the *extremal* points (i.e., they locally identify the maximum interpenetration).

We define the separation, ν_{i0}, of contact point i as

$$\nu_{i0} = {d_{i0}}^T \Delta u_i + s_{i0}. \tag{5.9}$$

For non-penetration, ν_{i0} must be positive, but if we want "non-velcro" forces then we also require the magnitude of the normal force to be non-negative, $\beta_{i0} = {d_{i0}}^T f_i \geq 0$. These values are complementary to each other as we can only have contact forces when the separation is zero, thus,

$$\beta_{i0} \perp \nu_{i0} = {d_{i0}}^T \Delta u_i + s_{i0}. \tag{5.10}$$

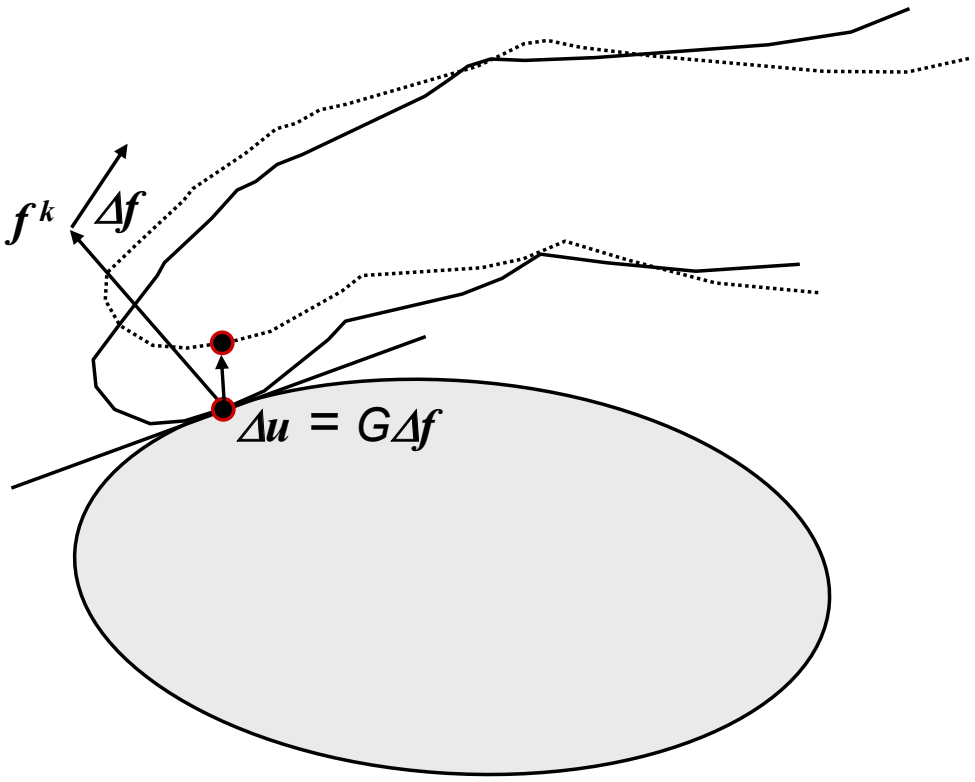


Figure 5.2: Contact motion due to changing forces at the contacts as specified by the effective endpoint compliance G.

We can combine this with Equation 5.7 by defining $\nu_i = (\nu_{i0},\ {\nu_{i\star}}^T)^T$, and $s_i = (s_{i0}, 0, \cdots, 0)^T$, which lets us write the complementarity conditions at point i as

$$\begin{aligned} \beta_i \ \perp \ \nu_i &= {D_i}^T \Delta u_i - {E_i}^T \lambda_i + s_i, \\ \lambda_i \ \perp \ \sigma_i &= F_i \beta_i. \end{aligned} \tag{5.11}$$

5.1.2 Contact Point Motion

Now, let us define Δu_i in an implicit fashion, combining the various sources of motion at the contact point. The problem is complicated due to sliding between the finger and the object, but we can break it down into three sources of movement: that required by the compliant structure for changing forces, that required by the compliant structure for changing reference angles, and that of the rigid body due to rigid body dynamics. We write the block vector $\Delta\mathsf{u}$ as the sum of three terms,

$$\Delta\mathsf{u} = \mathsf{G}\Delta\mathsf{f} + \mathsf{J}\Delta\theta_r - \mathcal{U}_\phi(\mathsf{f}). \tag{5.12}$$

The first term accounts for movement of the contacts due to the change in contact forces, $\Delta\mathsf{f} = \mathsf{f}^{k+1} - \mathsf{f}^k$. This is the motion necessary to maintain balance in the

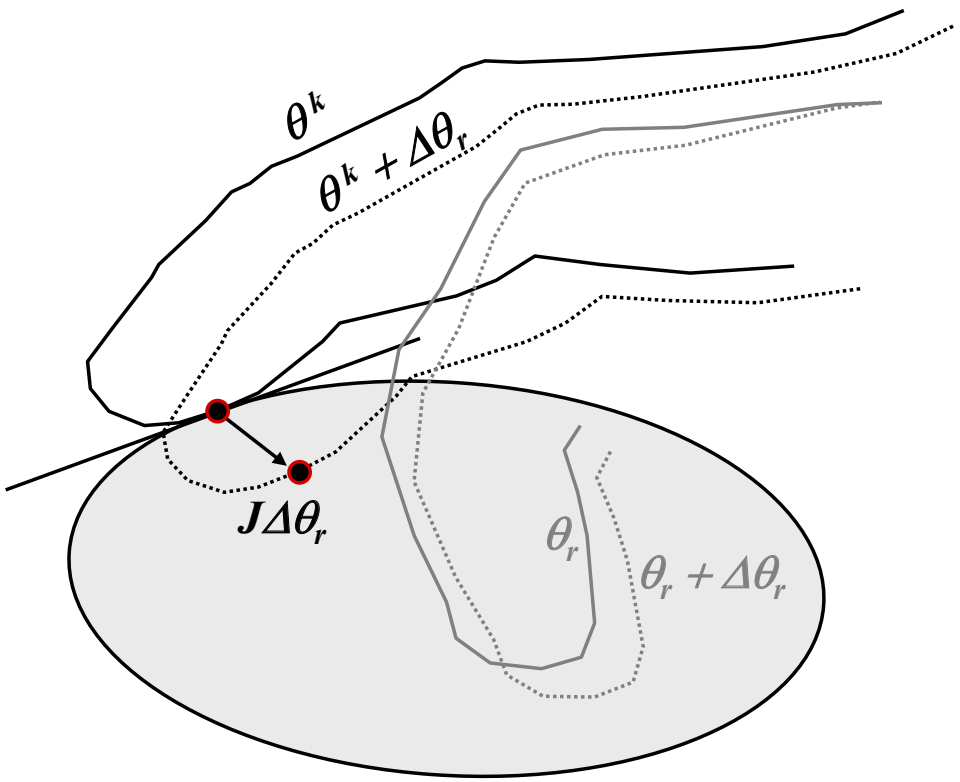


Figure 5.3: Contact motion due to a small change in the reference trajectory $\Delta\theta_r$.

equilibrium equation for the new contact forces; the matrix G gives us the locally linear approximation of this motion for small changes in force (see Figure 5.2).

The second term of Equation 5.12 is also related to maintaining equilibrium, as it gives the motion of the contact points necessary to preserve equilibrium given the current forces and the change in the reference angles, $\Delta\theta_r = \theta_r^{k+1} - \theta_r^k$. An example of this is shown in Figure 5.3.

The third term of Equation 5.12 is the motion of the contact point on the rigid body due to rigid body motion. Figure 5.4 shows one possible motion of the rigid body, with the fixed body contact point, x_i, denoted for both positions of the the body. This motion is due to the body's current spatial velocity, external forces (gravity), and contact forces. For the contact forces, we let $w(\mathsf{f})$ be the wrench on the body (in body coordinates) due to the sum of forces f_i^{k+1} at contact points r_i^k. Notice that we are using forces at the next time step. Because stable dynamics of the rigid body is a concern, we use a discretization of the rigid body dynamics equation that is implicit in the contact forces (the contact forces can change quickly in comparison to the other variables). In addition, we apply forces at r_i^k, rather than x_i^k, to match the forces applied at the fingers (the difference is small). This wrench is written as

$$w(\mathsf{f}) = -{}^w_b\mathsf{Ad}^T\mathsf{\Gamma}^T\mathsf{f}. \tag{5.13}$$

Here, $\mathsf{\Gamma}$, is a block column matrix of size $3N \times 6$ (for N contacts) with block i equal

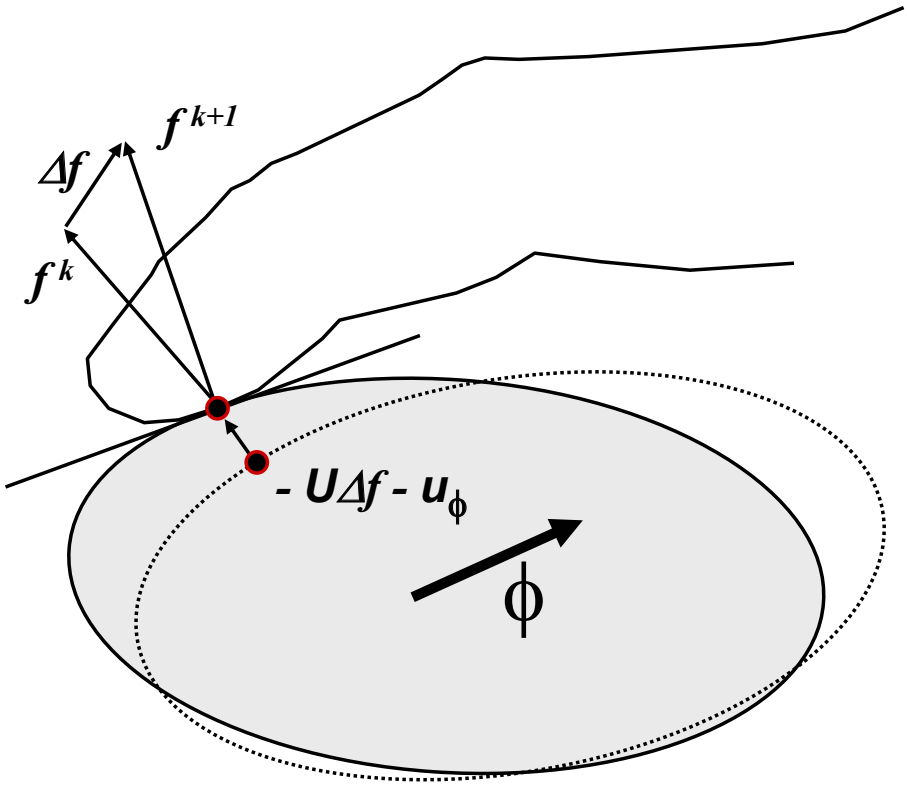


Figure 5.4: Contact motion due to rigid body motion.

to $\Gamma(r_i^k)$. This sums the forces in the block column vector f as wrenches in world coordinates. Notice we use the inverse transpose of the adjoint to map the summed wrench from world coordinates to body coordinates (recall the Newton-Euler equation is in body coordinates). We first rearrange Equation 4.2 to solve for accelerations,

$$\dot{\phi} = M^{-1}(w(\mathsf{f}) + [\phi^k]^T M\phi^k + w_{ext}), \tag{5.14}$$

where w_{ext} is the external wrench and includes forces such as gravity. Taking one Euler step of size h we can write the new velocity as

$$\phi^{k+1} = \phi^k + hM^{-1}(w(\mathsf{f}) + [\phi^k]^T M\phi^k + w_{ext}). \tag{5.15}$$

Taking an additional Euler step, we can write the linear approximation of the motion of the contact points due to rigid body motion as

$$\mathcal{U}_\phi(\mathsf{f}) \;\approx\; h\,\mathsf{\Gamma}\,{}^{w}_{b}\mathsf{Ad}\big(\phi^k + hM^{-1}(w(\mathsf{f}) + [\phi^k]^T M\phi^k + w_{ext})\big). \tag{5.16}$$

Here, the adjoint ${}^{w}_{b}\mathsf{Ad}$ converts the body spatial velocity back to the world coordinates used in our previous equations (most notably Equation 5.12), and the contact point velocities are computed via multiplication by $\mathsf{\Gamma}$. We write the right hand side of Equation 5.16 as,

$$\mathsf{U}_\phi\mathsf{f} + \mathsf{u}_\phi, \tag{5.17}$$

where we collect constant terms, and terms that depend on f,

$$\mathsf{U}_\phi = -h^2\ \Gamma\ {}^w_b\mathsf{Ad}\ M^{-1}\ {}^w_b\mathsf{Ad}^T\ \Gamma^T, \tag{5.18}$$

$$\mathsf{u}_\phi = h\ \Gamma\ {}^w_b\mathsf{Ad}\big(\phi^k + hM^{-1}([\phi^k]^T M\phi^k + w_{ext})\big). \tag{5.19}$$

5.1.3 Linear Complementarity Problem

Finally, we can use the pieces we have derived above to build a Linear Complementarity Problem (LCP). The complementarity conditions at each point (Equation 5.11) can be combined into a single system, where the forces and displacements are coupled using Equation 5.12. Letting

$$\begin{aligned}
D &= diag(D_1, \cdots, D_N),\\
E &= diag(E_1, \cdots, E_N),\\
F &= diag(F_1, \cdots, F_N),\\
s &= ({s_1}^T, \cdots, {s_N}^T)^T,\\
\beta &= ({\beta_1}^T, \cdots, {\beta_N}^T)^T,\\
\nu &= ({\nu_1}^T, \cdots, {\nu_N}^T)^T,\\
\gamma &= (\gamma_1, \cdots, \gamma_N)^T,\\
\sigma &= (\sigma_1, \cdots, \sigma_N)^T,
\end{aligned} \tag{5.20}$$

we now write

$$\begin{pmatrix} D^T(\mathsf{G} - \mathsf{U}_\phi)D & -E^T \\ F & 0 \end{pmatrix}\begin{pmatrix}\beta\\ \lambda\end{pmatrix} + \begin{pmatrix} s - \mathsf{G}\mathsf{f}^k - \mathsf{u}_\phi + \mathsf{J}\Delta\theta_r \\ 0 \end{pmatrix} = \begin{pmatrix}\nu\\ \sigma\end{pmatrix} \perp \begin{pmatrix}\beta\\ \lambda\end{pmatrix}. \tag{5.21}$$

This LCP can be solved with Lemke's method (see [Murty 1988]). Though an LCP may not have a solution, or may not have a unique solution, we observe that for the cases we considered our implementation always finds a solution.

Solving this LCP lets us compute the contact forces, $\mathsf{f}^{k+1} = D\beta$. The new forces let us advance the rigid body system (i.e., compute q^{k+1} and ϕ^{k+1} via integration). We then compute the new joint angles from the equilibrium equation using the current linear approximation,

$$\theta^{k+1} = \theta_r^{k+1} + C^{k+1}\ \mathsf{J}\ \mathsf{f}^{k+1}. \tag{5.22}$$

This is effectively a single linear approximation step towards the equilibrium, and is one of the sources of error in Equation 5.1 mentioned previously. The last step in advancing the system is to identify new points of contact, with the new configuration θ^{k+1} and the new body configuration q^{k+1}. That is, r_i^{k+1} and x_i^{k+1}, the new closest- or contact-points in the new configuration. These new points introduce error into the equilibrium equation because of a shift in the point at which the force is acting.

5.1.4 Emulating impedance control damping

Assuming that human fingertip control is similar to impedance control, we believe it is possible to introduce fingertip damping forces through forces that oppose the rigid body motion. Consider adding an additional wrench to Equations 5.16 and 5.19, specifically,

$$w_c = -\sum_i c_i \Gamma({}^b x_i^k)^T \Gamma({}^b x_i^k)\phi, \tag{5.23}$$

where ${}^b x_i^k$ is the location of contact i in body coordinates, and c_i is the damping coefficient for contact i. Note that if the damping coefficient and stiffness coefficient both increase linearly with the contact force, as suggested by Hajian [1997], then the damping ratio also increases. This can improve stability for small values of c.

5.1.5 Emulating soft fingerpads and contact patches

Even for the smallest forces, the fingertip deforms when in contact giving an area of contact rather than a point. With estimates computed in Section 4.4, and some extra computational expense, we could simulate quasi-static deformation for soft fingertips using the method described by Pauly et al. [2004]. Alternatively, and more simply, we can achieve some of the effects of soft fingerpads by including rotational torques in the friction computation. To accomplish this, D_i would instead consist of 6 dimensional (wrench like) values. Including additional vectors in D_i, for positive and negative rotation about the contact normal, permits rotational friction forces to oppose spinning at the contact. Corresponding to this change, the matrix G must then be redefined to relate full wrenches and full twists, rather than contact forces and contact displacements. This can easily be achieved by removing the Γ operators contained in the Jacobian matrices used in Equation 4.8 (see Equation 4.3).

5.2 Results and Discussion

We have implemented our retargeting simulator in Java using PQP for collision detection [Larsen et al. 1999]. Complex models and multi-finger contact make the simulation run slightly slower than real time. Using simplified geometry to build proxy collision detection models helps in part, though further speed improvements require code optimizations.

Figure 5.5 shows a typical interaction capture trial with a small box. The box measures $3 \times 5 \times 7$ cm and has a mass of approximately 12 g. As seen in the figure, the box is covered in non-reflective masking tape to avoid problems with specular reflections during Vicon motion capture.

Figure 5.6 shows synthesized grasps for two virtual objects, using the same captured grasp of the small box in Figure 5.5. In these examples, both the light bulb and

Joint	Compliance
Index MP flex/ex	5.00e-1
Index PIP	4.45e-1
Index DIP	2.13e0
Middle MP flex/ex	5.00e-1
Middle PIP	3.20e-3
Middle DIP	3.27e0
Thumb MP flex/ex	4.80e-1
Thumb IP	2.70e-1
Index MP ab/ad	3.20e-1
Middle MP ab/ad	3.20e-1
Thumb CM	7.00e-1
Root x	1.00e-3
Root y	1.00e-3
Root z	1.00e-3

Table 5.1: Compliance values used for simulation. Joint compliances are in rad/Nm and linear root compliances are in m/N.

the apple were placed in approximately the same location as where the box was placed during interaction capture. In the simulations, we used a fixed coefficient of friction equalling 0.5; this value is within the range of typical values reported by Han et al. [1996], however, the coefficient is known to vary with the contact force due to the varying area of the contact patch [Barbagli et al. 2004; Ciocarlie et al. 2005]. Incorporating variable friction into our technique through a simple soft finger contact model would not be difficult.

Table 5.1 shows the compliance values that were used during simulations in Figure 5.6. Compliance at the root is important as it regularizes the effective coupled endpoint compliance matrix. Note that for simulation, we assign compliance values to joints that did not have a compliance estimated. Generally, values in the table were either modified, or previously unestimated. For those values that were not estimated (for example, abduction and adduction at the MP joints) the table shows user selected values.

A demonstration of applying the same interaction on the same object but with different masses is shown in Figure 5.7. Because our technique preserves the idea, or intent, of the original interaction, we can produce motion where the virtual hand does not succeed in picking up an object because it is heavier than expected. Likewise, we can monitor slip between the finger and the object and modulate the stiffness (and, in turn, the equilibrium point) of the interaction. Figure 5.8 shows an example where we produce a simple grip tightening response due to slip, as might happen in a sensorimotor program when feedback does not match the prediction.

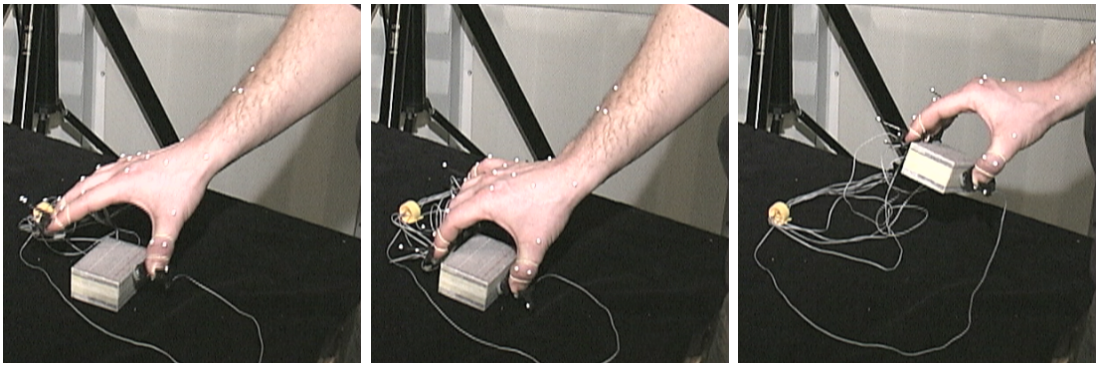


Figure 5.5: Selected frames from video showing interaction capture with a small box.

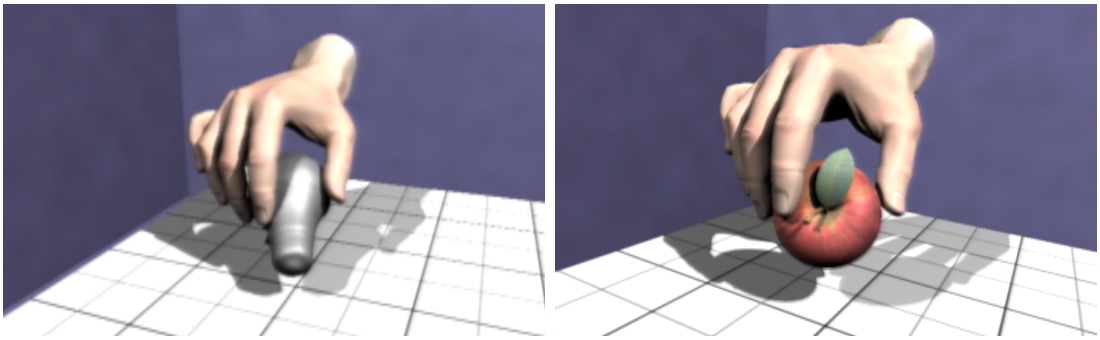


Figure 5.6: Grasp synthesis examples.

Figure 5.9 shows interaction captured with the Tango and the Vicon system, applied to a light bulb. Note that we did not estimate compliances from the Tango interaction capture session because the force capture rate and resolution of the tango are too low. However, we did use the data to estimate an equilibrium point trajectory using assumed compliance values. In the captured interaction, the subject used finger gaiting to manipulate the Tango. Similar finger gaiting manipulation can be seen in the figure for two synthesized interactions with a light bulb. The two simulations involved slightly different initial conditions. Because our technique is completely open loop and involves no planning, the simulation produces very different object trajectories for the same interaction. As a result, it may be difficult to use this captured interaction to screw a virtual light bulb into a virtual socket.

In general, our method tends to produce plausible human motion. Using interaction captured in the single-finger scratching and exploring contact trials, we synthesized new motion under a number of varying conditions (low friction, high friction, a flat surface, and a bumpy surface); we observed plausible simulation results for all variations. In the more complex multi-finger interactions with dynamic objects, however, we occasionally observe vibration artefacts. Specifically, the small box grasp applied to the apple exhibits some oscillation when the object first breaks contact with the ground. This is likely because the ground friction acts like a latch that lets go once

ground contact is broken (but is also a problem for larger virtual masses in general). The system of the apple in the grasp of a quasi-static compliant structure then acts much like a mass on a spring. Vibrations become more noticeable when virtual objects with larger masses are used. This is because the only damping experienced by the system is due to the implicit computation of interaction forces.

Note that our implementation currently allows only a single contact between the grasped object and any individual link in the kinematic structure. That is, we model each fingertip as a rigid body, and only handle a single contact between each of these rigid bodies and the simulated environment. Although it is straightforward to modify our technique for multiple body-link contacts, additional contacts will cause performance to suffer due to an increase in the LCP problem size (Lloyd [2005] describes a fast implementation of Lemke's algorithm that may help here).

An important limitation of our current implementation is that the equilibrium equation does not take into account joint limits. This does not pose a problem in most of our simulations since we are synthesizing interactions under conditions that are quite similar to the conditions under which they were captured. Additionally, since we model each joint with varying degrees of freedom, we do not need to worry, for instance, about abduction/adduction at IP joints. Just the same, we could model normal joint limits through the addition of stiff torsional springs that activate only when limits are exceeded (see [Pollard and Zordan 2005]).

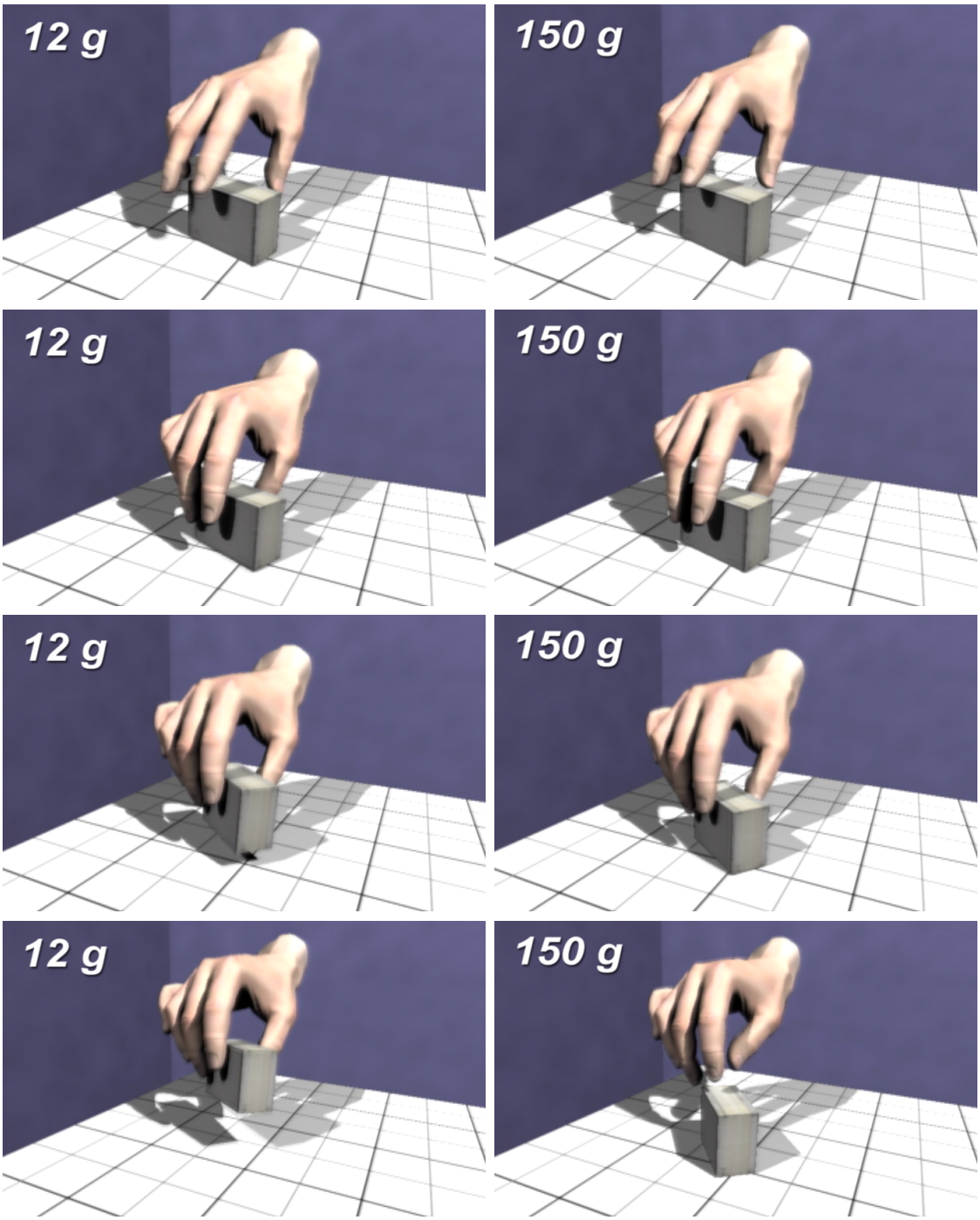


Figure 5.7: Interaction synthesis with objects of different masses. A light grasp fails to pick up the heavy object.

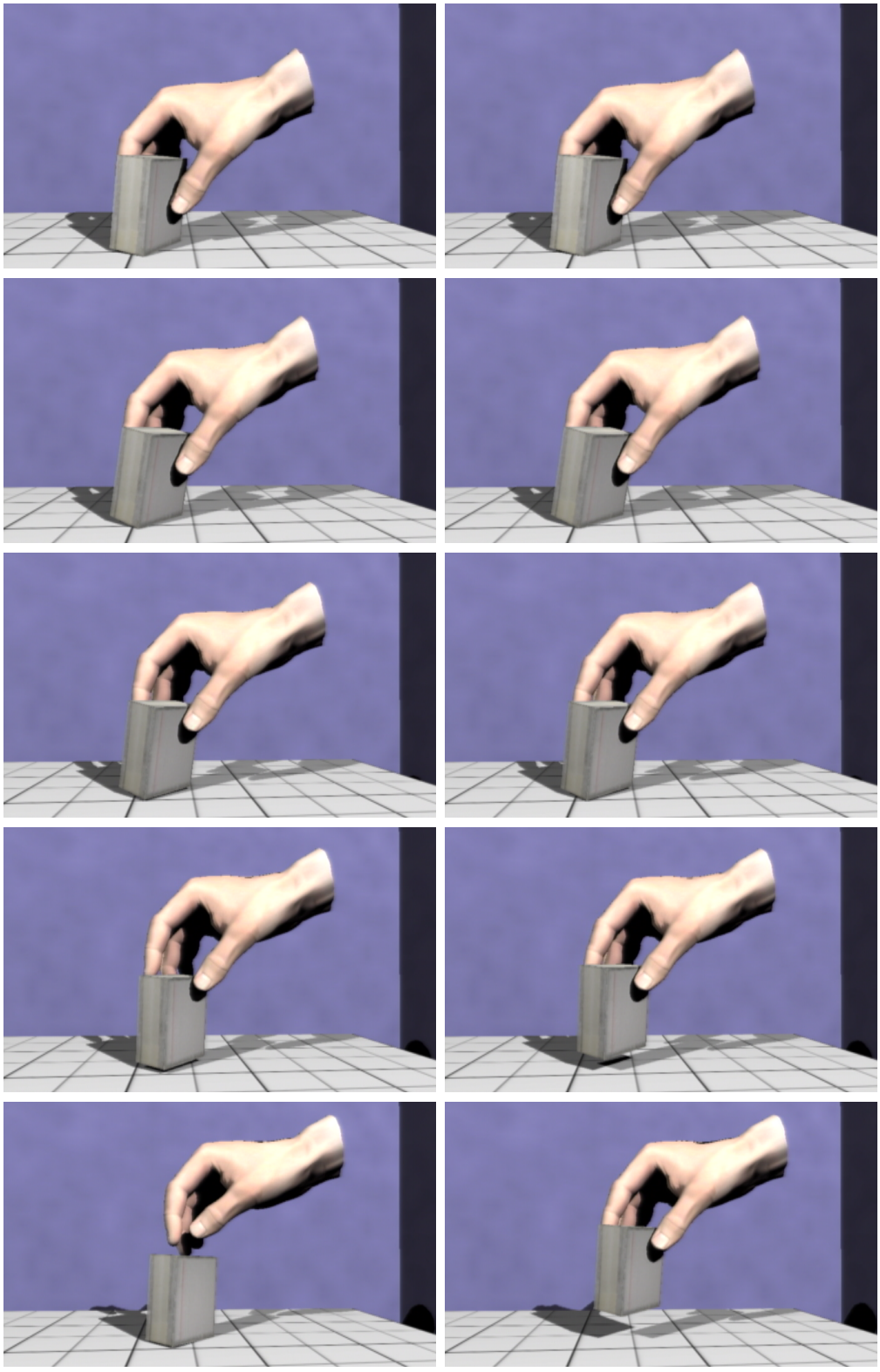


Figure 5.8: Grip tightening response due to slip. The sequence in the right hand column shows the same interaction, however, with a grip tightening that starts at the third frame.

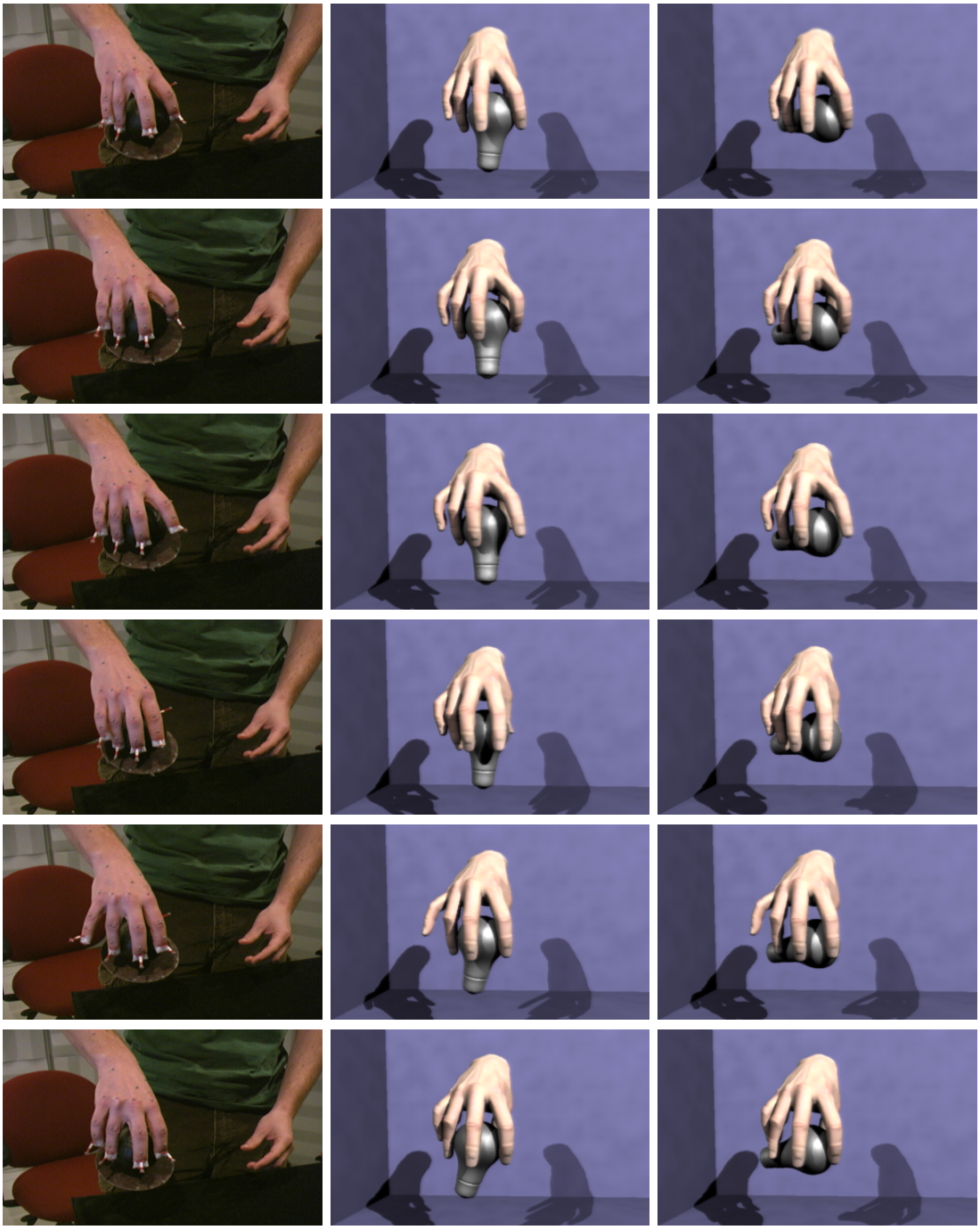


Figure 5.9: Light bulb manipulation from Tango data

Chapter 6

Interaction with the Tango

The last three chapters presented methods for capturing interaction, estimating compliances, and resynthesizing new interaction. As we have presented these concepts, the compliance estimation and resynthesis of interaction is an offline process. Captured interaction, however, does afford possible applications in real time interaction. Likewise, the resynthesis method in the previous chapter could even be employed in online real time interactive simulations using assumed compliances (this is an interesting direction for future work). However, in this chapter we instead examine a collection of simpler problems, specifically focusing on the use of the Tango as a self-contained interaction capture device.

Interaction in virtual environments should be as natural as picking up a real physical object in your hand and manipulating it with your fingers. One way to achieve this type of natural interaction is to use the Tango as an interaction capture device. Recall, the Tango is a ball that can be picked up and manipulated in the hand, but it measures the distribution of pressure applied on its surface (e.g., by fingers), as well as the motion and orientation of the device as a whole. We use this information to reconstruct the approximate shape of the user's hand, which is used in turn to synthesize a grasping hand in a virtual environment.

The grasp-shape and fingertip pressure can be used in a variety of ways for natural user interaction. For example, one could create *free form buttons*, that is, buttons assigned to fingers and not locations on device. Variable pressure measurements provide a useful continuous parameter that can influence the current interaction mode. For instance, one can indicate acquisition and release of an object by squeezing harder or softer. A grasp-type for interaction can be selected by merely shaping the hand appropriately, or grasping the object with two or three fingers. In addition, the motion sensors are useful for positioning and orienting in a 3D virtual world. Finally, as with glove-based interaction, seeing a realistic virtual hand in the environment, whose behaviour is correlated with one's own hand, makes it easier to understand the state of the interaction.

Glove-based interfaces are currently the most common whole-hand user interfaces,

but a lack of force feedback is an important limitation. Recent work shows quantifiable evidence that force feedback improves user performance; for instance, in surgery [Wagner et al. 2002], and in steering [Dennerlein et al. 2000; Forsyth 2004]. Several devices address this limitation by providing active force feedback on individual fingers (e.g., the Rutgers Hand Master [Bouzit et al. 2002], the CyberGrasp [Immersion Corporation], and the multi-finger haptic interface of Springer and Ferrier [1999]). While active force feedback has value for many tasks, whole hand force feedback is expensive and complex. Alternatively, the Tango provides the user with a passive form of force feedback, that is, it serves as a proxy object. This approach is a significant improvement over devices with no force feedback at all, and comes with greatly reduced complexity. This has been demonstrated with other devices, such as the Fingerball developed at the University of Toronto [Zhai et al. 1996], a device which provides a similar form factor for grasping and passive force feedback, but does not measure the pressure distribution. Passive haptics was also found to significantly enhance immersive virtual environments by Insko et al. [2001].

Section 3.3 describes the Tango device in more detail. Note that previous work on the Tango [Pai et al. 2005b] presents a simple method for grasp tracking with a simplified hand model. In this work, we focus on recognizing realistic hand shapes, and on using this recognition and other input from the Tango for 3D interaction.

6.1 Grasp Hand-Shape Approximation

This section describes our technique for approximating the user's hand shape based on pressures observed on the Tango. This involves clustering pressure measurements of adjacent taxels, and computing hand shapes from previously collected examples through the use of rotationally invariant comparisons of pressures measurements.

6.1.1 Clustering and Identifying Fingers

Clustering taxels into different contacts is useful for determining the number of fingers that are involved in a grasp. We perform this clustering by joining neighbouring activated taxels, while maintaining a pressure centroid and a total pressure for the cluster. At each activated taxel, we search its four directly connected neighbours (east, west, north, and south) and perform a merge if any of the neighbours are activated. In addition, we also check two additional taxels along the same meridian (the second taxel to the north and the second taxel to the south). This allows for clusters with a vertical gap, which is important since we occasionally observe pressures at taxels on the equator that do not activate during light grasps. An example of this can be seen in the bottom left corner of Figure 6.1.

In the case of three-finger grasps, we have also explored using heuristics to identify

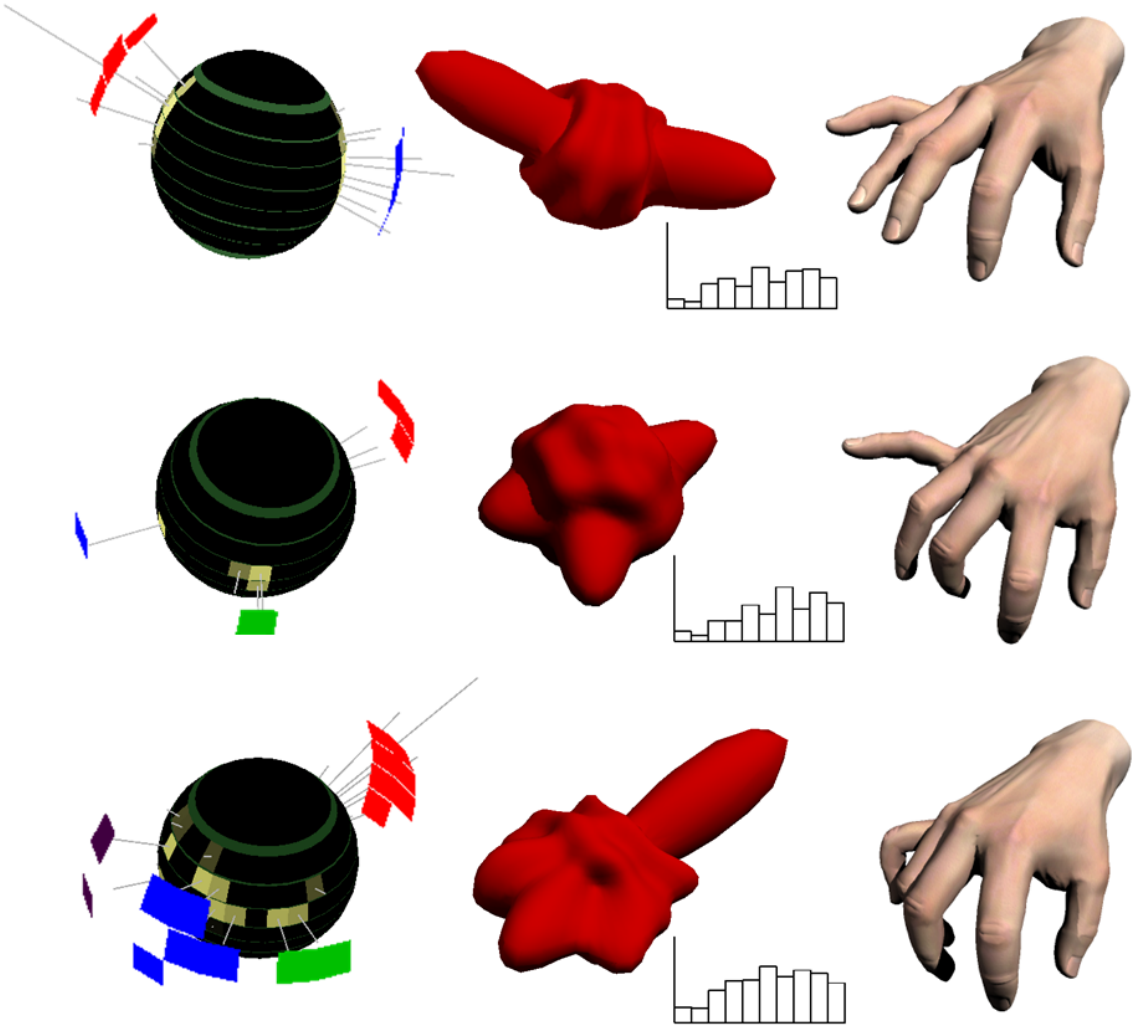


Figure 6.1: Three example data points from the two, three, and four finger trials. Left shows thresholded raw taxel data (pressure magnitudes shown by lines emanating from activated taxels, shaded yellow) with clusters shown in unique colours displaced from the surface. Center shows the spherical harmonic representations for the pressure data and its 10 frequency energy histogram. Right shows the corresponding synchronously-captured hand pose.

exactly which finger is responsible for each cluster. Namely, the thumb cluster is almost always identifiable as the cluster with the greatest total pressure. Assuming the Tango is grasped from above with the right hand, starting at the thumb cluster and travelling westward along the surface of the Tango, we identify the next cluster with the index finger, and the next following with the middle finger. Furthermore, we restrict the search for the middle finger to meridians that are within 45 degrees of the meridian opposite the thumb cluster. This helps to avoid identifying spurious single taxel clusters (caused by noise) as finger plants, as opposed to arbitrarily removing

single taxel clusters from consideration. Results of these finger heuristics can also be seen in the left hand side of Figure 6.1, where thumb, index, and middle finger clusters are coloured red, green, and blue, respectively.

6.1.2 Grasp Hashing with Spherical Harmonics

We would like to approximate the shape of a hand-grasp based solely on the pressures observed on the surface of the Tango. One approach is to approximate the hand configuration with a previously observed configuration that produced the same (or a very similar) pressure distribution. The difficulty with this approach is that we would still like to successfully match two identical pressure distributions applied at different orientations, as this would drastically reduce the number of examples we need to cover the space of possible pressure distributions generated by hand grasps. In other words, we do not want to compute distances between raw 256-dimensional taxel data vectors, but instead we would like to perform rotationally invariant comparisons.

Following the work of Kazhdan et al. [2003] on rotational invariance in shape matching, we convert our spherical pressure functions into rotationally invariant features for matching.

The spherical harmonics are written $Y_l^m(\theta, \phi)$, where θ is the angle from vertical direction (colatitude or polar angle), and ϕ is the angle in the plane (azimuth or longitude). The parameter $l \geq 0$ can be thought of as an integer frequency, while m selects the different harmonics at a given frequency ($-l \leq m \leq l$). The functions are real-valued when $m = 0$, but are otherwise complex. A set of real basis functions may be produced by taking the real part for $m > 0$ and the imaginary part for $m < 0$, i.e.,

$$y_l^m(\theta, \phi) = \begin{cases} \mathrm{Re}(Y_l^m(\theta, \phi)), & m > 0 \\ Y_l^m(\theta, \phi), & m = 0 \\ \mathrm{Im}(Y_l^m(\theta, \phi)), & m < 0 \end{cases} \tag{6.1}$$

These functions, shown in Figure 6.2, form an ortho-normal basis; they are orthogonal, and the integral of any function multiplied with itself over the sphere is equal to one. Because of this, a real spherical function may easily be converted, through projection, to a spherical harmonic basis, with coefficients

$$a_l^m = \int y_l^m(\theta, \phi)\ f(\theta, \phi)\ \partial\theta\ \partial\phi. \tag{6.2}$$

This integral is the continuous equivalent of a dot product. If the function is continuous and frequency-limited, the coefficients a_l^m provide a complete description of the function and will satisfy

$$f(\theta, \phi) = \sum_l \sum_{m=-l}^{l} a_l^m\ y_l^m(\theta, \phi). \tag{6.3}$$

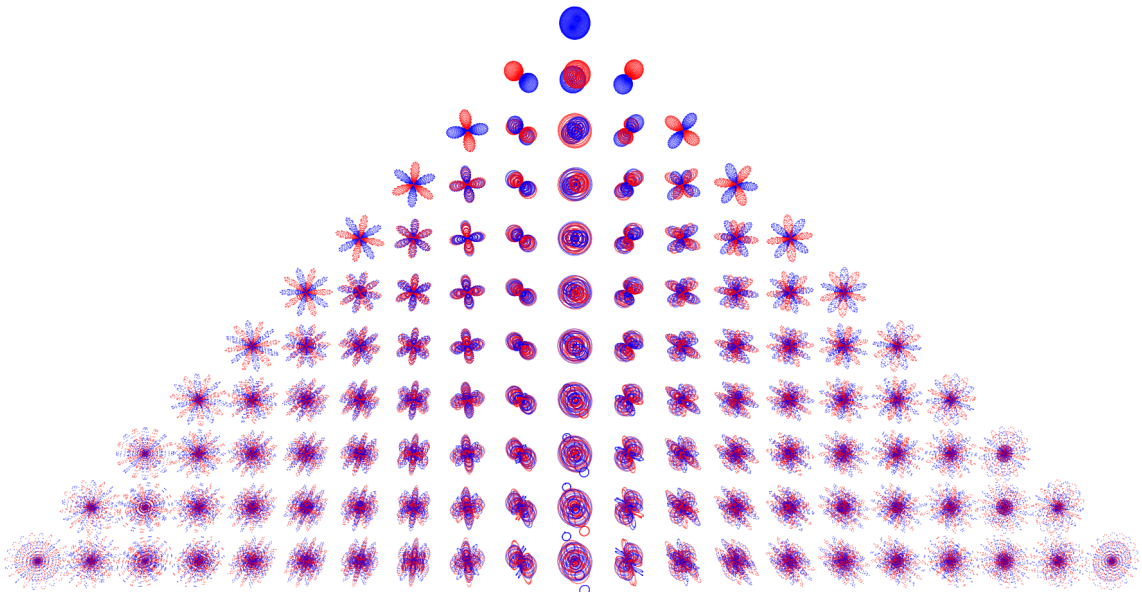


Figure 6.2: Real spherical harmonic basis functions y_l^m for $l = 0 \cdots 10$. Function magnitudes are show with positive areas in blue and negative areas in red.

In our case, we assume a point-sampled pressure function, with measured values centered at each taxel and zero everywhere else. Thus, coefficients are computed as

$$a_l^m = \sum_{i,j} y_l^m(\theta_j, \phi_i)\ p_{ij}, \tag{6.4}$$

where θ_j and ϕ_i provide the polar and azimuth angles that run through the taxel centers, and p_{ij} is the pressure of the taxel located on meridian i and parallel j. We precompute $y_l^m(\theta_j, \phi_i)$ since the taxel locations are fixed.

The pressure function in the spherical harmonic basis is a frequency-limited smoothly varying representation. We use the first 10 frequencies in our spherical harmonic basis, which corresponds to a total of 100 basis functions (due to the $2l - 1$ functions at each integer frequency l). Note that 10 is a user-selected parameter, and l would need to equal 15 to make this change of basis invertible (i.e., 256 spherical harmonic basis functions and corresponding coefficients). With small values of l, the coefficients provide a best fit in a least squares sense, and the change of basis acts much like a low pass spatial filter. Given the noise in Tango measurements (and the size of fingerpads), the benefit of higher spatial frequencies is questionable.

With the pressure measurement converted to the real spherical harmonic basis, we sum the energies (coefficient magnitudes) at each frequency to produce a 10 dimensional coordinate, $x = (x_0, \cdots, x_9)^T$,

$$x_l = \sum_{m=-l}^{l} |a_l^m|. \tag{6.5}$$

This coordinate is called an energy histogram. It can be thought of as a feature vector, fingerprint, or hashing of the pressure function, with built-in rotational invariance. Kazhdan et al. [2003] provide a nice discussion of the properties and limitations of this representation. Figure 6.1 shows the $l = 10$ spherical function for different examples of Tango pressure data, along with their corresponding energy histograms.

Data points are very sparse in ten-dimensional space. By first projecting data points into an appropriate lower dimensional space we can generate more meaningful *dimension-reduced* distance comparisons. Principal component analysis (PCA), related to least squares, is a common method of computing nested subspaces that optimally describe the variance of points in a data set [Jolliffe 1986]. Previous work has shown that final grasp postures are well approximated by only a few principal components [Santello et al. 1998]. In our case, we expect variations in hand shape (similarly the pressure distribution and corresponding energy histogram) to be well approximated by a lower dimensional subspace, especially considering that the hand shape is constrained to be grasping an object of fixed shape (i.e., the Tango).

From here on, we call the energy histogram projected into truncated PCA space a pressure hash, or just a hash. Similar to the collisions produced by a hashing function, a collision can be generated at any step of computing a pressure hash: different grasps can produce the same pressure distribution on the Tango, different pressure distributions can result in the same energy histogram, and different energy histograms can project to the same truncated PCA coordinates. A key feature of our hash function, however, is that it is locality-preserving, which allows for meaningful comparison of hash values. Specifically, a set of similar grasps will result in similar hashes, while a set of similar hashes will contain subsets of similar grasps. This property is attributable to all steps in the process being either linear (change of basis and projection) or piecewise linear (energies of basis functions are equal to the absolute values of their coefficients).

6.1.3 Data Collection for Grasp Identification by Example

Pressure hashes give us a means of comparing grasps in a rotationally invariant way. Given a set of example grasps and their resulting pressure hashes, we can compare these hashes with the hash of a newly observed pressure distribution. The proximity of the example hashes to the new hash effectively measures the plausibility with which example grasps can explain the observed pressure. More importantly, the process of collecting examples is greatly simplified due to this rotational invariance, as we do not need to sample the same grasp at all orientations of the ball relative to the hand.

To build our example data set, we acquired synchronized motion capture of the Tango position and orientation, hand configuration, taxel pressures, and Tango accelerations. We used a Vicon motion capture system. Interactions with different numbers of fingers were considered separate "conditions". In total, approximately 10

minutes of capture data was acquired at 60 Hz. For each condition, we compute a separate PCA space of the energy histograms of surface pressure samples. Figure 6.1 shows example data from different trials.

6.1.4 Tracking Grasp Shape

Recall, a pressure reading is first interpreted as a function on the sphere and converted to a bounded frequency spherical harmonic representation. The energies of the basis functions at each frequency are then summed to produce an energy histogram. The histogram is projected into the previously computed truncated PCA space to get a d-dimensional hash representing the current grasp shape (currently, we use $d = 6$). With this hash, we find the k-nearest (Euclidian distance) neighbours in the previously computed data. For this we use a bounding hyper-sphere tree constructed with the method described by Quinlan [1994] but extended to arbitrary dimension. This data structure was also used in Section 3.2.3. Recall that building a tree of data in PCA coordinates lets us easily compute Mahalanobis-like distances in different truncated spaces by simply summing fewer terms in our ℓ^2 distance computation. The bounding sphere tree is still valid for truncated spaces, though possibly less efficient.

Our nearest neighbour searches are very fast because of the simple bounding volume test, combined with small tree depths (our deepest tree has 19 levels for about 9500 data points). Each neighbour has a corresponding hand configuration that we compare with our current hand configuration using a weighted Euclidean distance. The weighted distance metric allows us to ignore the position and orientation of both the forearm and wrist. The closest hand configuration among the k-nearest pressure-hash neighbours becomes the proposal configuration. Our new estimated configuration is then computed as a weighted combination of the current estimate and the proposal configuration (we use a blending ratio equal to 0.8 current, 0.2 proposal). This prevents our estimated poses from moving too quickly, and minimizes the effect of spurious distant proposal configurations. Overall, our method works much like a simplified particle filter tracker. Note that if there is no pressure observed on the Tango, then we can infer nothing about the hand shape. In this case, we use a previously selected rest pose configuration for the proposal.

To improve performance, we use the finger count from clustering to restrict our search for proposals to only the example data containing grasps with the same number of finger plants. This assumes that two-finger grasps involve thumb and index fingers, three-finger grasps involve thumb, index, and middle fingers, and four-finger grasps use all fingers but the little finger. Figure 6.3 shows our approximation result for a two-finger and three-finger grasp.

Remember, this process estimates the hand configuration independent of rotation. The same configuration is produced for the same pressure function, independent of its orientation with respect to the Tango body frame. To account for this, we could

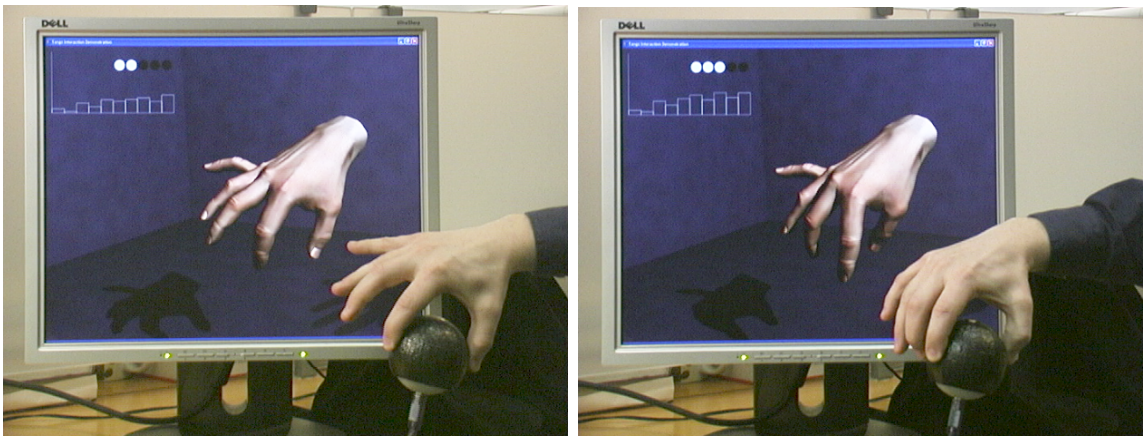


Figure 6.3: Grasp approximation results.

perform a final alignment of the hand with respect to the Tango by placing the fingertips at the actual pressure readings. Although this is not implemented in our demonstration, this could easily be done by aligning the fingers in the estimated pose with finger locations as identified with clustering and heuristics.

6.2 Grasp Based Interaction Modes

Possibly the most natural mechanism for choosing a mode of interaction is to map different grasp types to different modes of interaction. In the real world, we use precision grasps involving only the fingertips for fine manipulation, while for tasks that involve more power we use palmar grasps. In our work, the number of fingers used in a grasp, as identified by clustering, provides a reliable method of mode selection that is robust to errors in the identity of fingers used to create each pressure cluster.

Grasp based selection of the interaction mode is one possibility for handling changes between the positioning methods that we discuss in the next section. Specifically, we have experimented with assigning two- and three-finger grasps to positioning tasks in the horizontal plane, while four- or five-finger grasps are assigned to position control in the vertical plane.

Virtual or free form buttons are an alternative to grasp based mode selection; they can provide a means of switching between different interaction modes. In addition, they can be used to perform a selection (i.e., to command the grasp of an object at the location of the hand). Similar to mouse buttons, we first tried assigning different fingers to different virtual buttons. Here, the location of the finger need not be important, while the actual finger used determines the button pressed; however, this relies on accurate finger identification. More importantly, we observed that it is difficult to control the pressure of one finger independently of the others, and we could not find a set of button triggering pressure thresholds that would provide satisfactory

results. This difficulty can be explained in considering that the user's fingers must satisfy a force closure property on the Tango if a stable grasp is to be maintained.

Thus, as an alternative, the number of fingers used in a grasp can determine the button number. The total pressure triggers a button press with a hysteresis function. The total pressure causes a button down event when it rises above a high threshold, and a button up when it goes below a low threshold. Although the thresholds can be independently tuned for the number of fingers used, we find one set of high and low thresholds sufficient. As the Tango provides only passive haptic feedback, we also provide an auditory cue, signalling button down and up events. We use a two-finger virtual button to toggle between horizontal and vertical position control, and a four-finger virtual button to toggle orientation-only control.

6.3 Position and Orientation Control

The use of accelerometers for positioning, targeting, and gesture based interfaces is becoming more common, for example, in TiltType [Wigdor and Balakrishnan 2003] and Rock 'n' Scroll [Bartlett 2000]. Crossan and Murray-Smith [2004] perform a study on important design considerations. In this section, we describe methods for orienting and positioning with the accelerometers in the Tango.

The accelerometers in the Tango give a reliable estimation of the down direction. We call the down direction, or tilt from vertical, the *attitude*. Note that in some papers, the word attitude is used to mean the full three dimensional orientation of a rigid body. The attitude is always measurable because the accelerometers measure forces acting on a suspended mass, and thus they report the force due to gravity when stationary. They also indicate the direction of movement when abrupt position changes occur. Thus, the accelerometers permit several methods for controlling position.

Integrating accelerations twice to get position is unreliable even with accelerometers with low noise and moderate accuracy. We experimented with several methods for computing an approximation of the relative changes in position while preventing undesirable drift, but found our experiments to be unusable for full 3D positioning. Likewise, for proper acceleration estimation at all orientations, a full orientation measurement is required and not available in the current prototypes. Nevertheless, we find that we can produce adequate approximations of relative changes in elevation (described later in this section).

6.3.1 Orientation and Grasp Frame

The Tango cannot directly measure rotation about the vertical axis, because it has only accelerometers and no gyroscopes, magnetometers or other sensors that could

be used to estimate this orientation (though this could be added to future versions). With this limitation, it is still possible to provide control of the orientation.

Acceleration measurements are in a local reference frame attached to the Tango body, where the negative z axis points along the USB cable. Although the x and y axes of the measurement frame point in the direction of specific meridians in the taxel data, we could instead map accelerations to a hand aligned Tango reference frame. For example, the negative y axis could be chosen to point to the meridian where clustering and finger identification heuristics have identified a thumb contact. Although this change of frame is straightforward to implement, we currently require the user to grasp the Tango in a manner such that the grasp frame and Tango body frame are aligned.

The down direction lets us specify two of the three degrees of freedom of the orientation of the Tango (and thus infer an orientation for the hand). As there is a small amount of noise in accelerometer readings, we first smooth the data with an exponential falloff filter ($\alpha = 0.5$). From the filtered acceleration, $a = (a_x, a_y, a_z)$, we compute the angle from vertical, $\theta = \cos^{-1}(a \cdot -z)$. The hand orientation we compute is simply a rotation by θ about the axis $-z \times a$. We display this orientation during the positioning interactions described below. This provides users with useful feedback because they are presented an image of a virtual hand that mimics their own natural hand movements.

6.3.2 Attitude for Velocity Control

We use the filtered acceleration, a, and current angle from vertical, θ (computed above), combined with minimum and maximum tilt angle parameters to compute a displacement parameter between 0 and 1 as,

$$u = \begin{cases} 1 & \theta_{\max} < \theta, \\ (\theta - \theta_{\min})/(\theta_{\max} - \theta_{\min}) & \theta_{\min} \leq \theta \leq \theta_{\max}, \\ 0 & \theta < \theta_{\min}. \end{cases}$$

When u is greater than zero, we go on to compute a normalized direction

$$\hat{d} = \frac{(a_x, a_y)}{|(a_x, a_y)|}.$$

Given a maximum speed, $s_{\max}$, we compute a two dimensional relative change in position as,

$$u^{\beta}\ \hat{d}\ s_{\max}\ c\ h,$$

where h is the time between measurements, and the exponent β (when greater than one) allows for finer control over small velocities. We use $\beta = 2$, combined with $\theta_{min} = 4°$, $\theta_{max} = 90°$. The scalar value, c, acts like a clutch, but also modulates

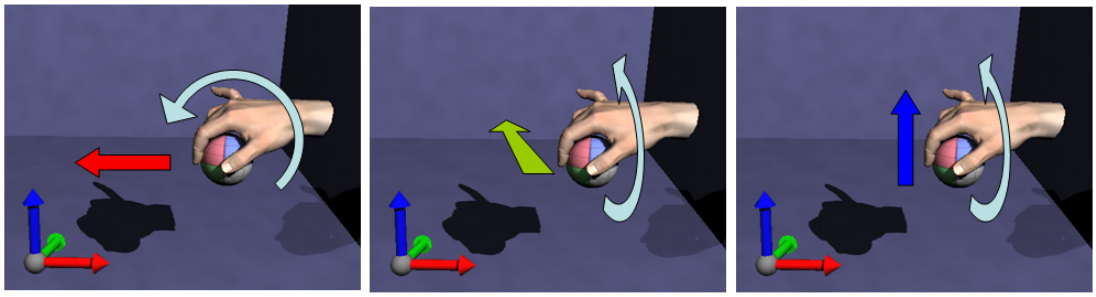


Figure 6.4: Mapping of rotations to positions is analogous to rolling a ball on a flat surface. The x, y, and z axes are shown in red, green, and blue, respectively. Left shows that rotating about the y axis results in a translation in x position. Middle shows that rotating about the x axis results in a translation in y. Right shows the alternative interpretation of rotations about x as translations in z.

the maximum speed, and is computed by first clamping measured total pressures to a given range, followed by a mapping to $[0, 1]$ (just as u was computed from θ).

The overall effect is that if we tilt the Tango to the right (while applying pressure), then we see our virtual hand move to the right. Likewise, if we tilt the Tango towards the screen we see our virtual hand move farther away, while tilting away from the screen brings our virtual hand closer. As shown in Figure 6.4, this mapping of rotations to linear directions is analogous to the direction of movement when rolling a ball on a horizontal surface.

Note that the small but tuneable dead zone around the upright position allows a stationary placement to be easily maintained (even when pressure is applied, which is important for virtual buttons). Another important detail is that we bound the absolute position to a fixed volume, much as a mouse pointer stops at the edge of the screen. This effectively prevents the user from getting lost when commanding velocities that generate positions too far from the volume of interest.

One method of extending position control to three dimensions is to switch between two different modes, either based on the current grasp or selected by pressing virtual buttons (as described earlier). In the second mode, we map tilting towards and away from the screen to vertical motions as shown in Figure 6.4. Here, the mapping of rotations to vertical motion is more arbitrary and we allow the user to invert the mapping if desired.

6.3.3 Attitude for Position Control

In a manner similar to a GyroMouse [Gyration], we considered using changes in attitude as changes in position, while using total pressure as a clutch. Here, changes in position are directly linked to changes in orientation of the Tango, as opposed to the method described above where a static tilted posture results in a constant velocity.

Beyond a simple on/off clutch, as used in the GyroMouse, we also tried using the total pressure to modulate the relative changes in orientation, such that rotations with a light grasp could be used for fine control, while the same small rotations with a firmer grasp could cause larger motions. Unfortunately, even with aggressive acceleration filtering, we could not find parameters for this mode yielding satisfactory results.

6.3.4 Accelerations for Altitude Control

As mentioned previously, navigation by tilting only allows us to modify two of three linear DOFs at once. Mode switching, as discussed above, provides a means of controlling the third; however, an alternative is to monitor vertical accelerations, and from this approximate relative changes in altitude. A useful feature of vertical accelerations is that in the absence of horizontal motions they can be measured simply as the lengthening or shortening of the acceleration vector. More importantly, the measurement can be made without knowledge of the orientation of the device.

We match upward movements to positive spikes in the accelerations and downward movement to negative spikes, but we also discard the spike of opposite sign that follows the initial spike in a given window. More precisely, when the acceleration goes above a threshold, the amount in excess of the threshold is scaled by a user tuneable parameter, and integrated once (by multiplying by the time step) to produce a velocity update, but only if there has not been a recent acceleration measurement exceeding the opposite threshold. Velocities are integrated to produce changes in position, and a viscous damping term provides an exponential decay in the speed so that ultimately the vertical position will come to rest.

Figure 6.5 shows example data collected from an interaction session. The accelerations are computed as the length of the measured acceleration vector minus the acceleration at rest; as a result, the largest negative acceleration possible is -9.8 m/s^2 and corresponds to free fall. Here, the up and down acceleration spike thresholds are 0.15 and -0.2 m/s^2 respectively. Accelerations in excess of the threshold are scaled before they are integrated. We use a scale factor of 120. Although we find this value makes for responsive control within a one meter cubic virtual workspace, lower values improve the user's ability to control positions precisely. Notice that velocity changes occur only for accelerations that exceed the dashed (red) threshold lines, but only when no accelerations exceeded the opposite threshold within the previous 0.2 seconds. The negative acceleration spikes at 1 second and approximately 1.5 seconds are ignored. The velocity damping coefficient used in this example equals 0.004 s^{-1} (i.e., velocity is scaled by a factor of 0.75 at a rate of 60 Hz).

Overall, this method provides a surprisingly compelling means of controlling the vertical position. With practice, the user can control the position very accurately.

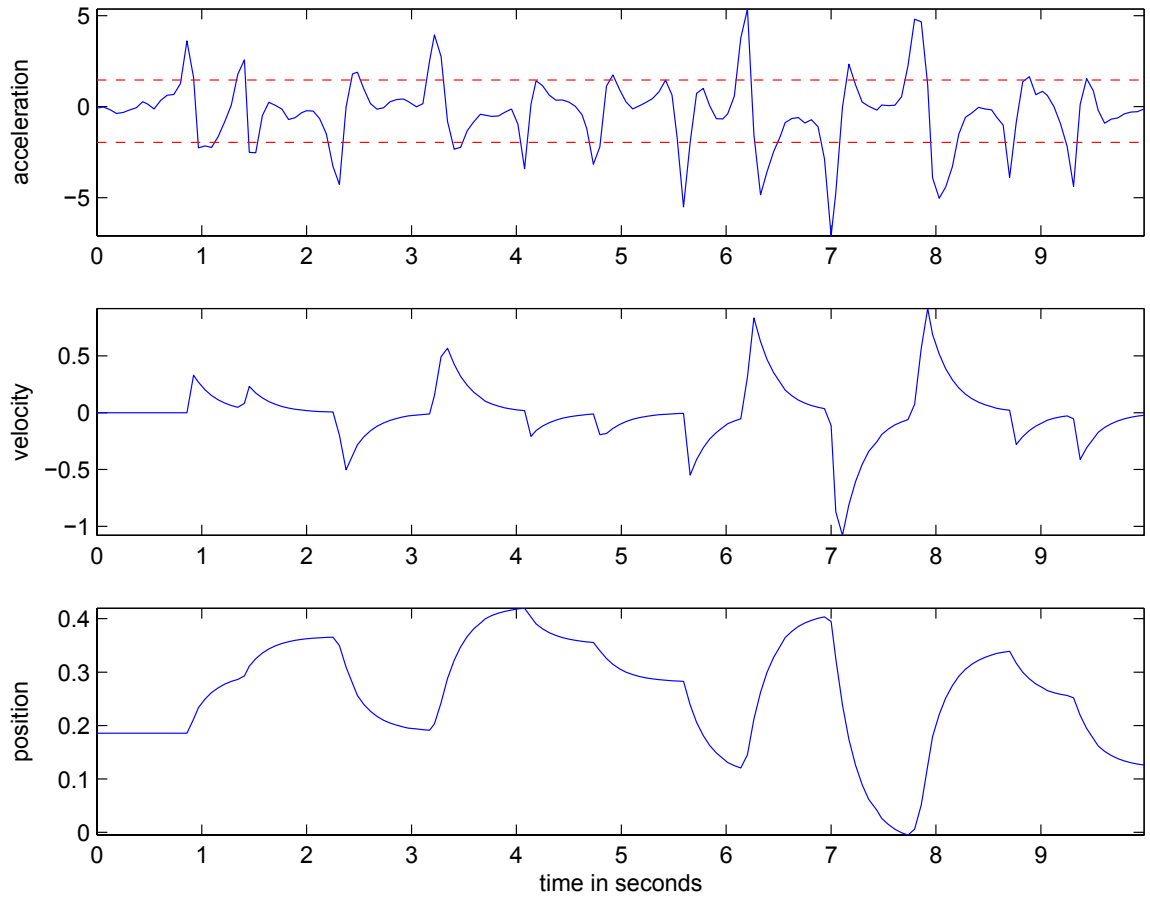


Figure 6.5: A graph showing Tango position control using acceleration. Units are m, m/s, and m/s^2. Velocities are computed as a special function of accelerations, while positions are computed by integrating velocity.

6.4 Discussion

We have developed a small virtual world in which we can explore the performance of positioning, orienting, and object grasping tasks. Figure 6.6 shows a snapshot of the user's view of the world. Grasping in this demonstration environment is performed through a virtual button that toggles grasp and release. If there is no object in close proximity to the fingers then the grasp fails. Otherwise, the object becomes attached to the hand and we keep the most recent hand configuration fixed. We also smoothly blend a small translation into the hand position to better align the hand and grasped object. Note that grasping in our demonstration is iconic, though we could use interaction simulation described in Chapter 5 to bring the hand into proper contact with the object.

We assume all grasps on the Tango are precision grasps (for proper identification of number of fingers from the number of clusters). However, our rotationally invariant

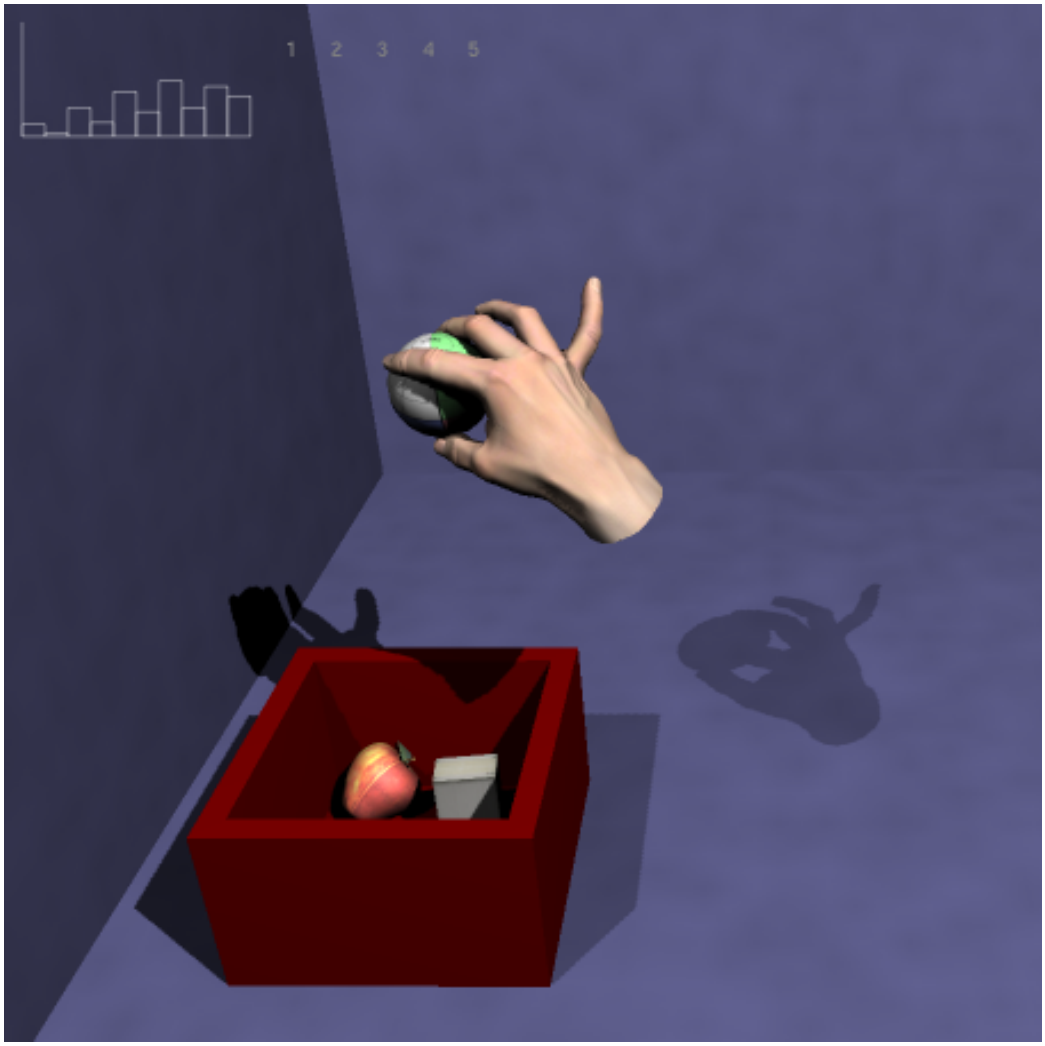


Figure 6.6: A screen shot from the Tango interaction demonstration.

pressure hashes show promise for correctly identifying the number of fingers when all of our example data trials are combined into one. Furthermore, we expect that our method extends to other grasp types, such as conforming and palmar grasps. Reconstruction of hand posture from pressure data is a similar problem to that of reconstructing a full body posture from foot pressure data, addressed by Yin and Pai [2003b], but the latency requirements are more severe for manual interaction than for animation. Other work on grasp tracking includes that of Bernardin et al. [2005], which uses both forces and the hand shape to classify grasps for robotic programming by demonstration.

The methods of position control we present here are somewhat arbitrary, and represent the preliminary steps in exploring the use of the Tango for positioning. These methods require a formal study to determine, ultimately, their usefulness. Similarly, the success rate of our grasp-shape recognition should be quantified. Variable pressure

measurements can also allow for interactions that are more complex. For instance, a possible avenue for future work would investigate using the Tango for tasks that involve deformation. Lastly, we have only considered right-handed use. Although extending our methods to left-handed use is straightforward, harder problems include switching hands or using two hands on the same Tango.

Chapter 7

EigenSkin

The articulated movement of human hands results in complex skin shapes due to the deformation of soft tissues surrounding the bones. Rendering these deformations in real time is very challenging. While the previous chapters concern capture, synthesis, and interaction with human hands from a purely kinematic point of view, in this chapter we address the important problem of rendering these soft tissue deformations.

Currently, most interactive applications use a simple deformation technique called Skeletal Subspace Deformation (SSD). In SSD, each vertex of a rest pose mesh is transformed by an affine combination of bone transforms. Usually only a small number of bone transforms affect any given vertex, making SSD attractive for its small computational expense. However, there is limited flexibility in creating deformations with linear blends of bone transforms. For example, the lateral bulging in a bicep is not possible without the addition of another bone transformation to provide the lateral translation (i.e., in this case, a bone transform that does not correspond to the motion of a physical bone, see, for example, [Mohr and Gleicher 2003]). Extra bones have disadvantages as they add complexity to the control of an animated character and require additional matrix multiplications. This requires longer vertex programs and ultimately longer computation time.

Because most of the muscles that control hand movement are found in the forearm, the soft tissue deformations that hands exhibit tend to be static and only dependent on pose. Because of this, a pose based deformation approach [Lewis et al. 2000; Sloan et al. 2001] is ideal for capturing hand shapes during articulated animations. In this chapter, we consider only skin deformations without contact. The expanded problem of skin deformations during contact and grasping, however, is an interesting avenue for future work.

Our EigenSkin deformation model is created by augmenting an existing approximate deformation model, Skeletal Subspace Deformation (SSD) as presented here, with pose dependent corrections. Instead of storing corrections for each key pose and then interpolating between them at runtime (as in pose space deformation [Lewis et al. 2000] or shape by example [Sloan et al. 2001]), we instead use principal compo-

nent analysis to select a small basis for describing corrections. Using reduced bases for the corrections attributed to individual joints, combined with the assumption that the corrections can be linearly superimposed, yields a representation that is powerful yet inexpensive with respect to both memory and computation.

In this chapter, we first describe SSD along with our method for building the EigenSkin of a character given a set of training data computed with a finite element model. Although we describe this process for position corrections (which we call displacements), it applies similarly to the construction of linear normal corrections, allowing EigenSkin to correct SSD for both shape and shading. This is followed by a discussion of our results, as well as a description of our preliminary work on acquiring EigenSkin from real world examples.

7.1 Notation: SSD and Bone Weights

Let $\mathcal{B}$ be the set of all bone indices, and denote the bones affecting vertex i by the subset of indices $B_i \subset \mathcal{B}$. For a given skeletal configuration, with bone transforms $\{T_b\}_{b\in\mathcal{B}}$, the position of the i^{th} vertex after SSD is

$$\tilde{v}_i = \left(\sum\nolimits_{b\in B_i} w_{ib} T_b\right) v_i, \tag{7.1}$$

where v_i is the position of vertex i in the neutral pose, and w_{ib} give the affine combination of bone transforms for this vertex. In the character's neutral pose we assume that[1] $T_b = I, \forall b \in \mathcal{B}$. When not in the neutral pose, the transforms are determined by bending at the various joints connecting the bones in the articulated skeletal hierarchy.

We compute our SSD bone weights as a function of vertex bone distances in the neutral pose. For each vertex, we find the closest distances to a small set of bones where we use the bone geometry shown in Figure 7.1. The set of bones we use in these distance computations depends on the group to which the vertex belongs. This typically consists of the adjacent bones in the immediate area of the skin (i.e., for the index fingertip we use bones in the index finger, but not those in the thumb or middle finger). Figure 7.2 shows the vertex groups, which roughly correspond to different bones. Bone weights are first computed as the distances raised to small negative powers. These values are subsequently normalized to produce an affine combination by dividing by their sum. The negative exponent causes larger weights to be assigned when vertices are close to the bone, while varying the exponent trades off between weighting all bones equally (small negative exponents approaching 0) or weighting strongly the closest bone (larger negative exponents). We choose this exponent interactively in order to find a set of weights that provide deformations that

[1] This can also be thought of as $T_b = {}^{w}_{b}E_{\text{current}}\,{}^{w}_{b}E^{-1}_{\text{rest}}$ where frame w is the world frame, and frame b is the given bone frames.

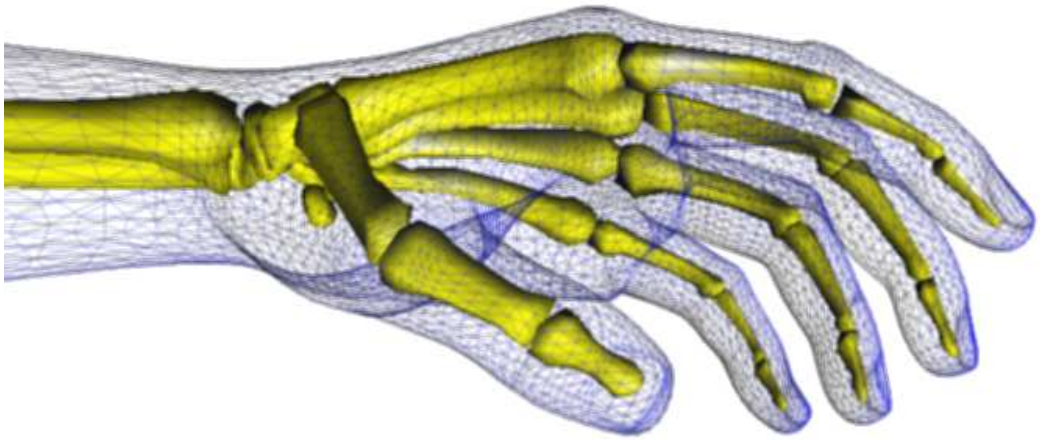


Figure 7.1: Skeleton used to compute vertex bone weights.

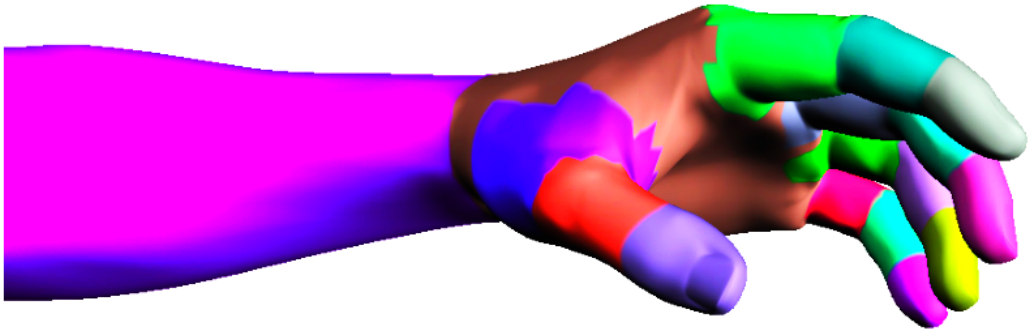


Figure 7.2: Skinning groups used for SSD.

look neither too rubbery nor too robotic. For our hand model, which is measured in millimeters and approximately the size of an average adult male hand, we find that an exponent of negative 3 produces reasonable weights. Alternatively, weights can also be created by an artist in an application such as Maya.

7.2 Locally Supported Joint Displacements

We start with an SSD model and a set of key poses that provide examples of the desired deformation shapes. We refer to these poses as *observed* poses. Let $\mathcal{P}$ be the set of indices of observed poses with $0 \in \mathcal{P}$ representing the rest pose and let the observed vertex positions and bone transforms for pose $p \in \mathcal{P}$ be denoted as v^p and T^p, respectively. The differences between the SSD vertex positions and the observed

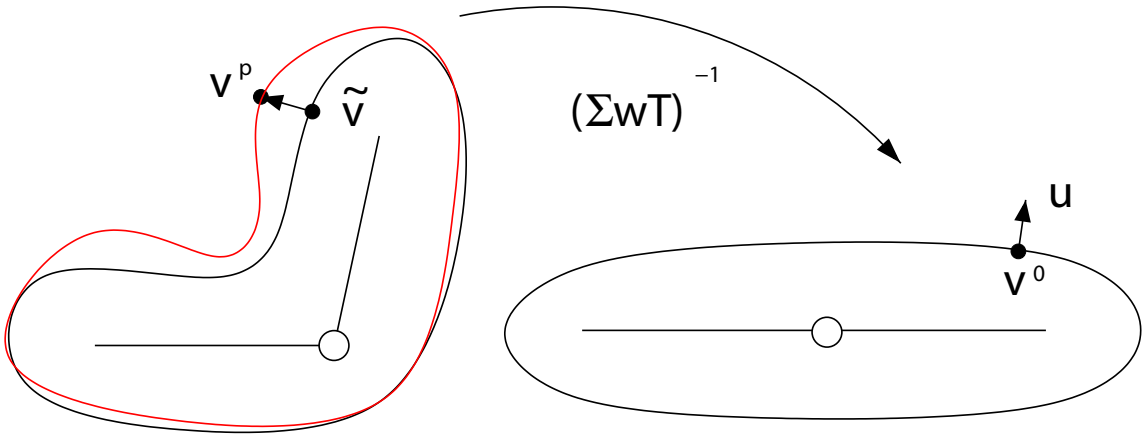


Figure 7.3: The displacement for vertex i and pose p, denoted u_i^p, is the difference between its observed position, v_i^p, and its position under SSD, $\tilde{v}_i^p$, mapped back into the rest pose.

pose positions mapped back into the rest pose yield displacements (see Figure 7.3),

$$u_i^p = \left(\sum\nolimits_{b \in B_i} w_{ib} T_b^p\right)^{-1} v_i^p \quad - \quad v_i^0.$$

The observed mesh shapes result when these displacements are added to the rest pose before applying the bone weighted transformation. If the deformations vary smoothly over pose space, then interpolated displacements provide a good approximation of deformations at configurations between observations.

To make our hardware implementation possible, we exploit the observation that localized changes to the configuration of an articulated character often result in local deformations. This independence occurs in most articulated characters, and certainly exists in realistic human hands. Bending a single joint, though often difficult to demonstrate in human hands without bending any other joints, does not cause noticeable deformations in the other fingers. Likewise, as seen Figure 7.4, bending one joint in our finite element hand model does not cause noticeable deformations in the others Although the finite element model deformations resulting from a change to a single joint are global, the displacement magnitudes are imperceptible at vertices that are far from the joint. Section 7.6 provides additional detail on the finite element model used. Note, however, that the source of deformation examples (for individual joint motions) is not important.

We refer to the set of vertices significantly affected by a joint motion as the *joint support*. Note that the joint supports depend on the SSD weights and in general they do not correspond to the sets of vertices influenced by bone transforms (compare Figure 7.5 and Figure 7.2).

To find the support of a joint we compute the deformations that result from moving the joint to different positions in its full range of motion while keeping all other joints

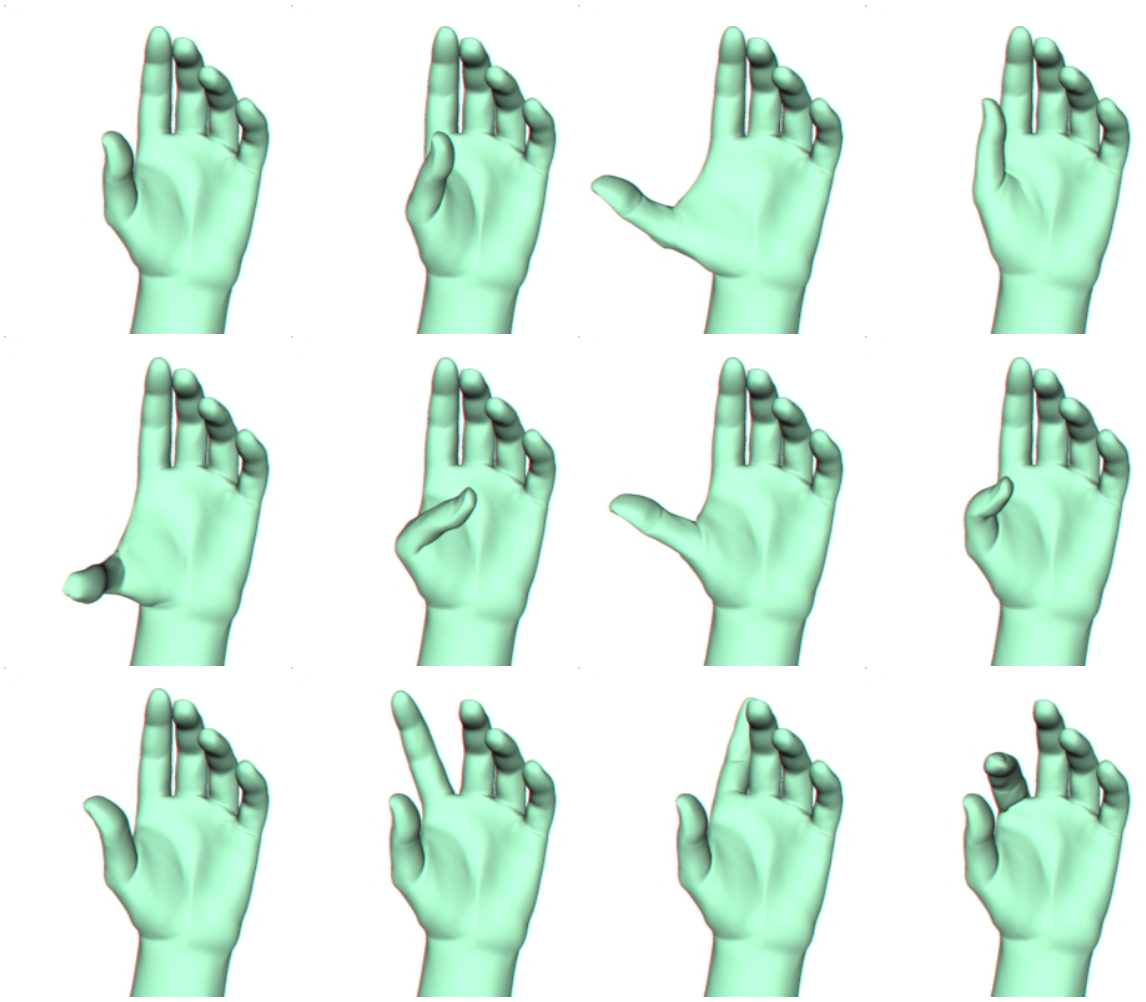
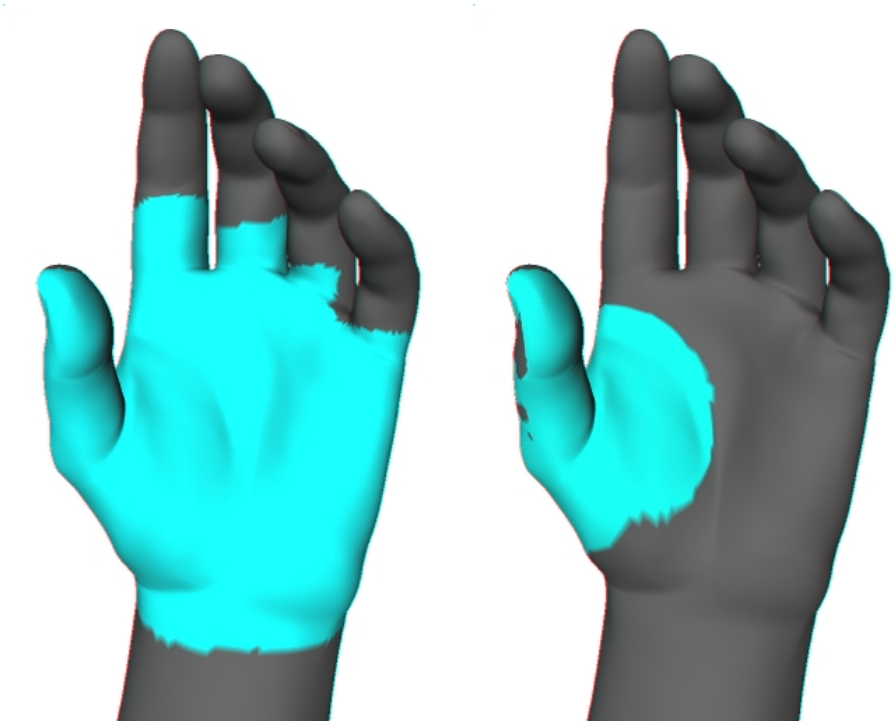


Figure 7.4: A subset of the training data showing some of the thumb and index finger poses computed with our finite element hand model.

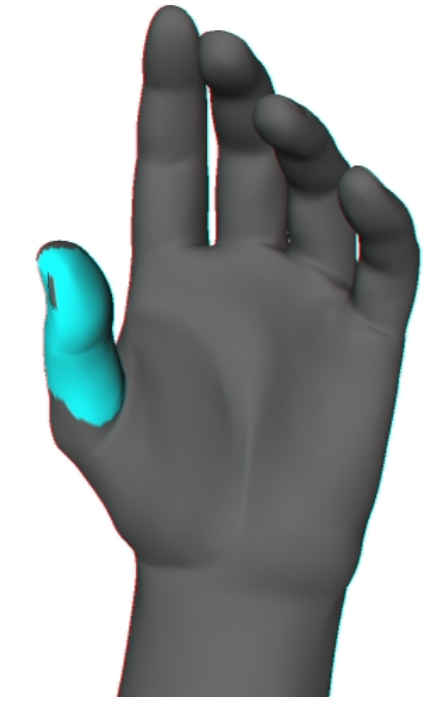
fixed to the rest pose position. The set of vertices having a displacement larger than a given threshold in any of these computed poses then becomes the support of this joint. For example, in our case we used four percent of the maximum observed displacement (we will see that memory constraints also play a large part). Several joint supports of our finite element hand model are shown in Figure 7.5. Note that we consider only single joint perturbations due to the high dimensionality of our hand model's configuration space. Nevertheless, we can still approximate linear coupling effects since we let the joint supports overlap.

For notational convenience, suppose the articulated figure has a tree structure, i.e., does not have loops, such as for humanoids, and joints are denoted by the index of the adjacent bone furthest from the root of the hierarchy. Denoting $0 \in \mathcal{B}$ as the root, joints have nonzero index. Let $P_j \subset \mathcal{P}$ be the set of pose indices used to compute



thumb CM

thumb MP



thumb IP

index MP

Figure 7.5: *Joint supports* for thumb carpal-metacarpal, metacarpo-phalangeal, inter-phalangeal joints and index metacarpo-phalangeal joint.

the support for joint j and let S_j be the set of vertex indices in the joint support. Furthermore, let J_i be the set of joints whose supports contain vertex i. That is, $J_i = \{j | i \in S_j\} \subset \mathcal{B} \backslash \{0\}$.

7.3 Eigendisplacements

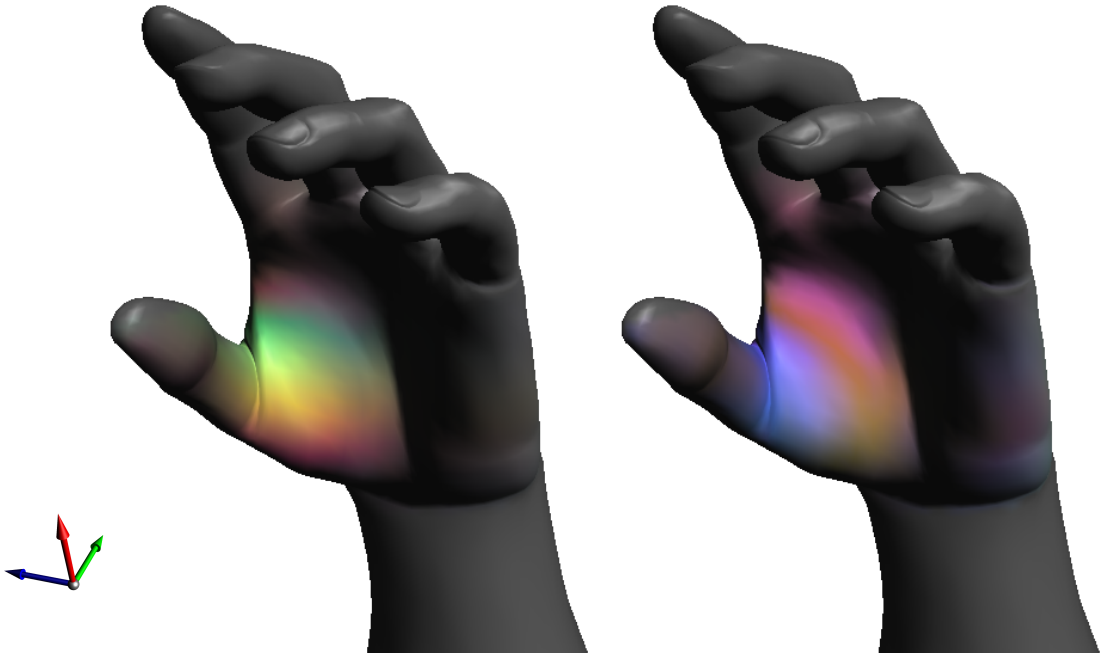
The pose displacements computed for independently perturbed joints may be used as a basis for describing corrections for new configurations. However, significant redundancy exists in the pose displacements, for example, skin bulging in similar directions. Principal Component Analysis (PCA) of joint-support vertex-displacements yields an orthogonal displacement basis, which we term *eigendisplacements*. As guaranteed by PCA, adding successive corrections with the eigendisplacement basis provides approximations which are better in a formal, least squares, sense [Golub and van Loan 1996].

Computing principal components with the Euclidean norm is equivalent to computing the singular value decomposition (in the case of a square symmetric matrix it is equivalent to eigenanalysis). For each joint j we construct a rectangular matrix, $\mathbf{A}_j$, of size $3|S_j| \times |P_j|$. Each column corresponds to one pose and contains the x, y, and z components of the vertex displacements on the joint support. In the singular value decomposition, $\mathbf{A}_j = \mathbf{U}_j \mathbf{D}_j \mathbf{V}_j^T$, the matrix $\mathbf{U}_j$ has the same size as $\mathbf{A}_j$ and consists of columns of eigendisplacements for support j in the same block column format that was used to build $\mathbf{A}_j$. The singular values, in the diagonal matrix $\mathbf{D}_j$, identify the importance that each eigendisplacement has in reproducing the observed poses (they relate to the proportion of variation explained by each principal component). Note that the matrix $\mathbf{V}_j$ and the singular values combine to gives the coordinates of our observed displacements in the eigendisplacement basis. We denote $\hat{u}_i^{jk}$ the eigendisplacement of vertex i in the basis of support j with importance k where k goes from 1 (the principal component) up to $|P_j|$. Figure 7.6 shows the first four eigendisplacements of the thumb carpal-metacarpal joint support in our hand example.

At this point we can truncate each eigendisplacement basis expansion knowing that the error will be minimized in the least squares sense. The hardware limits the size of each truncated basis set as there is a limited amount of per vertex data memory in which we can send the eigendisplacements to the EigenSkin vertex program (see Section 7.5). Letting $n_j < |P_j|$ be the size of the truncated basis set of joint support j, this constraint can be written as

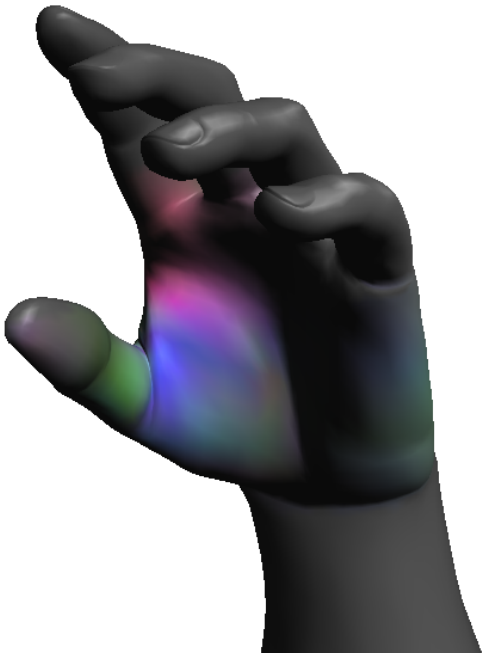
$$\max_i \; n_j |J_i| \; \leq \; \text{maximum possible displacements.}$$

Instead of choosing each n_j individually, we take an equal number of eigendisplacements from each support.

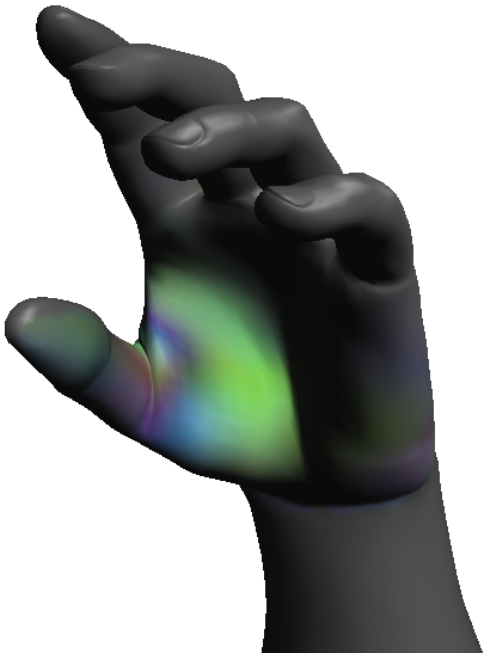


#0 ($\sigma_0 \approx 0.093$)

#1 ($\sigma_1 \approx 0.065$)



#2 ($\sigma_2 \approx 0.023$)



#3 ($\sigma_3 \approx 0.018$)

Figure 7.6: *Eigendisplacements* and singular values, σ, for thumb carpal-metacarpal joint in left-right order of importance. XYZ components of displacement are represented using an RGB *colour* correspondence.

The equation for computing the deformed mesh shape for an arbitrary configuration with bone transforms $\{T_b\}_{b\in\mathcal{B}}$ can then be written as

$$\tilde{v}_i = \sum_{b\in B_i} w_{ib} T_b \left(v_i^0 + \sum_{j\in J_i} \sum_{k=1}^{n_j} \tilde{\alpha}_{jk} \hat{u}_i^{jk} \right) \tag{7.2}$$

where $\tilde{\alpha}_{jk}$ gives the *coordinates* of the displacement correction in terms of the reduced eigendisplacement basis. These coordinates are computed to interpolate between observed displacements, as shown below. Note that Equation 7.2 provides a powerful model for shape deformation (see, in particular, [James and Pai 2002a]).

7.4 Interpolating Eigendisplacement Coordinates

As an articulated character moves between observed configurations, its shape should interpolate the corresponding observed poses. To do this we interpolate the eigendisplacement coordinates of the observed configurations. For the truncated set of eigendisplacements at each support, we need the coordinates in the truncated basis that give displacements closest to the observed displacements. That is, we want to solve for α^p in

$$u_i^p = \sum_{j\in J_i} \sum_{k=1}^{n_j} \alpha_{jk}^p \hat{u}_i^{jk}.$$

This is an over constrained linear system which we can solve using least squares to get the best fit to our observed displacements. Conveniently, the least squares solution for any number of eigendisplacements, n_j, is available from the singular value decomposition computed in Section 7.3. For joint support j, column p of $\mathbf{D}_j\mathbf{V}_j^T$ contains α_{jk}^p for $k = 1..|P_j|$.

This leads us to the problem of computing the eigendisplacement coordinates for arbitrary configurations. Radial basis functions [Powell 1987] (RBF) are a common choice for interpolating scattered data, and have been used by Lewis et al. [2000] for pose space deformation and by Sloan et al. [2001] for shape interpolation with articulated figures. Our interpolation is mostly one dimensional since all our observations involved perturbations of individual joints and most joints had only one DOF. Although we could use a simpler interpolant in these single-DOF cases, we also choose RBFs because they extend easily to the higher dimensional domains needed to let EigenSkin capture non-linear multi-joint coupling effects (a subject of future work).

We use Gaussian interpolation shape functions, $\phi(r) = \exp(-r/r_0)$. In our one dimensional case, the α_{jk} only depend on the distance of joint j from its settings in poses P_j. For revolute joints (i.e., hinge joints), we can easily compute the distance, r, by comparing the joint angles directly. For joints with more than one rotational degree of freedom, we compute distance as the angle in the axis-angle representation of the joint's rotation matrix.

Ideally, with a large number of observed joint perturbations per support we would interpolate using fewer interpolation basis functions (ϕ) than observations. In the case of our hand model, however, we only have approximately half a dozen pose perturbations for each joint degree of freedom (for a total of approximately 120 poses). As such, we include all observations in the interpolation and we can justify this choice due to the negligible total cost of constructing and evaluating the RBF interpolant for only half a dozen poses. The interpolated eigendisplacement coordinates for a new pose are computed as

$$\tilde{\alpha}_{jk} = \sum_{q \in P_j} \lambda_q^{jk} \phi(r_{jq}),$$

where r_{jq} is the distance of joint j in the new pose from its setting in pose q, and the λ_q^{jk} for $q \in P_j$ are given by the solution to the linear system,

$$\alpha_{jk}^p = \sum_{q \in P_j} \lambda_q^{jk} \phi(r_{jq}^p), \quad \text{for} \quad p \in P_j.$$

Here r_{jq}^p is the distance between joint j's position in pose p and its position in pose q (and thus $r_{jp}^p = 0$). The system of equations is square, and invertible provided P_j does not contain two observations with identical joint settings.

Figure 7.7 shows example coordinate interpolation functions for the thumb IP joint eigendisplacements computed with a kernel width $r_0 = 0.13$. In this case, the kernel width was selected as half the average angular distance between adjacent example poses. The significance of the first few eigendisplacements is evident in comparing their coordinate interpolation functions. The first function obtains an absolute magnitude larger than 0.03 between -1 and -0.5 radians, while the last three components do not exceed 0.002.

Notice that we omit the linear (or low degree polynomial) regression term that is often included in RBF interpolants. Omitting this term causes the correction to approach zero when far from the example poses, leaving us with the original SSD approximation. This may be preferable in cases where the regression term would compute a large correction, but joint limits often prevent such a situation from occurring. Future implementations should consider adding a linear regression term to improve the interpolation.

Finally, note that these interpolation functions are used for correcting the shape. However the methods described in Sections 7.2 through Section 7.4 apply similarly to the construction of linear normal corrections. Correcting the normals is necessary when rendering the surface with lighting enabled.

7.5 EigenSkin Vertex Programming

Modern vertex programming hardware (e.g., [Lindholm et al. 2001]) is ideally suited to performing the per-vertex weighted linear superposition of eigendisplacements (con-

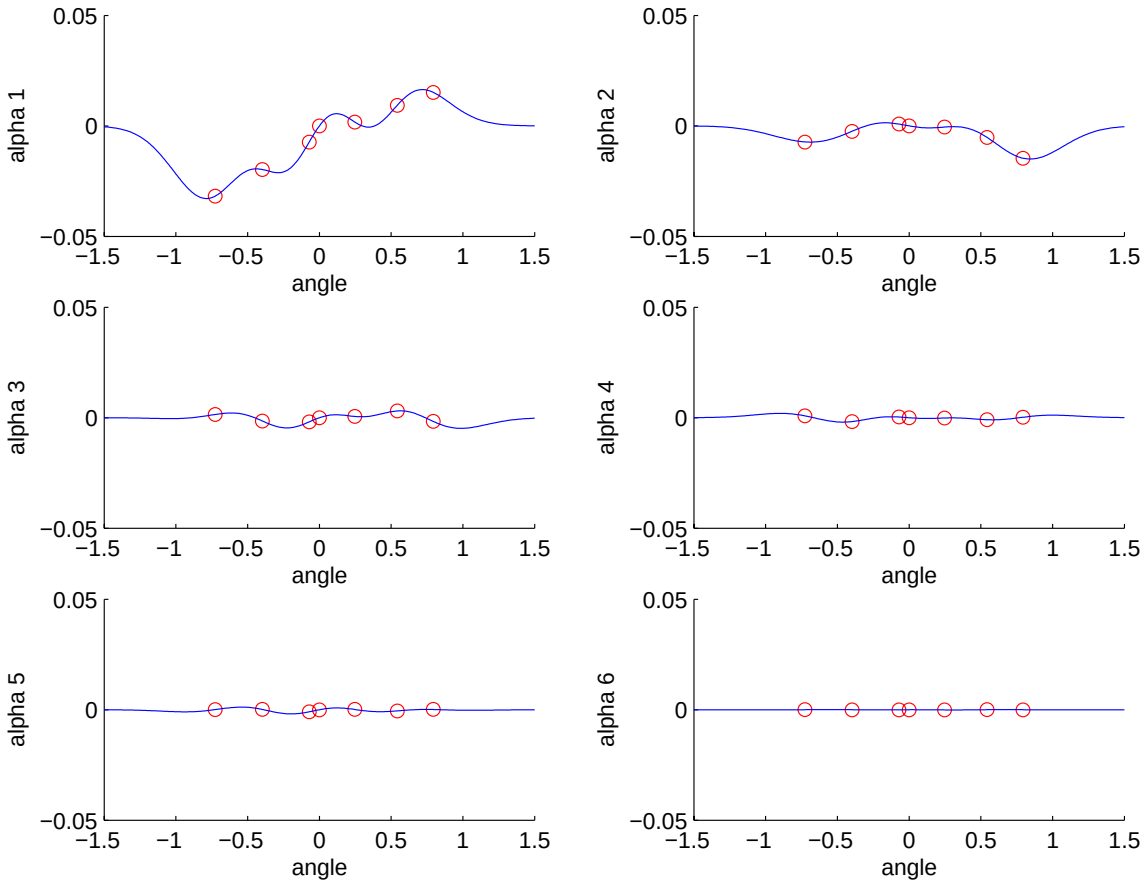


Figure 7.7: Coordinate interpolation functions for the six eigendisplacements computed for the Thumb IP joint. Circles mark the interpolated values.

tained in the large brackets of Equation 7.2) performed prior to the SSD weighted transformation. Depending on the number of eigendisplacements used, the weighted eigendisplacement vector accumulations are about as costly as the weighted transform matrix-vector multiply-accumulate operations.

Current graphics hardware limit per vertex data to 16 4-tuples of floats. In our implementation we impose a limit of 10 eigendisplacements per vertex (or 5 eigendisplacements and 5 normal corrections), which still leaves room for texture coordinates after specifying the vertex position, normal, colour, and bone weights. Notice that this limit is not hard since careful choices and packing of per vertex data permit more than 10 of the 16 available tuples to be allocated for EigenSkin data.

If a vertex is in many supports then the number of eigendisplacements renderable by current hardware may be too severely restricted. In this case it is useful to smoothly mask[2] the support groups to smaller regions, otherwise, fewer eigendisplace-

[2]One possible method for smoothly masking a support would be to reduce the length of the corrections on the support. The size of the support would be reduced due to an increase in the

ments must be used. However, new hardware now allows vertex programs to access texture memory, opening up new alternatives for efficient storage of larger numbers of eigendisplacements.

7.6 Results and Discussion

To illustrate our EigenSkin method, we have constructed a finite element model of the human hand that exhibits subtle nonlinear skin deformations. Figure 7.1 shows a skeleton similar to that used to drive finite element hand model, however, simplfied bones were used in the simulation to avoid bone-bone interpenetration during articulation. The surface skin model and matching skeleton are based on Loop subdivision [Loop 1987] of a hand mesh exported from Curious Labs Poser [Curious Labs Inc.]. A finite element mesh containing 11,171 high-order 10-node tetrahedral elements was generated using NETGEN [Schoberl 1997] (and subsequent simplification). The hand was moved into various poses by applying position constraints to vertices adjacent to the rigid bones, and computing the resulting tissue deformation using geometrically nonlinear static finite element analyses [Zienkiewicz 1977] with (a modified version of) the CalculiX program [Dhondt and Wittig]. The tetrahedral mesh and finite element analyses were generated by Doug James in a manner similar to [James and Pai 2002a].

Approximately half a dozen poses were computed for each joint DOF to estimate the locally supported joint eigendisplacements, and 25 additional poses were used for validation. Finite element analyses were performed on a cluster of modern workstations and consumed several hundred CPU hours. The model was not intended to reproduce detailed skin wrinkling effects, and lacks anatomical details such as tendons, blood vessels, and skin layers. Despite these limitations, the model reasonably describes bulk tissue deformations and it was sufficient to illustrate our method.

As shown in Figure 7.8, the eigendisplacement approximations of the hand model produce a clear improvement over the traditional SSD algorithm. Even with only a few leading eigendisplacements, the EigenSkin approximation is almost indistinguishable from the original FEM model. Even for poses which do not involve poses in the training data, we observe good results as shown in Figure 7.9.

Our interactive simulation uses a CyberGlove [Immersion Corporation] input device to interactively drive our EigenSkin hand model (see Figure 7.10), while graphical feedback is rendered using OpenGL and a vertex program capable graphics card. Radial basis function interpolation of the pose-space data is performed on the main CPU, with eigendisplacement amplitudes and bone transforms set as input parameters to the EigenSkin vertex programs, which are compiled as static display lists. Currently, our unoptimized implementation renders the EigenSkinned hand model

number of vertex displacmenents falling below the threshold defining the support.

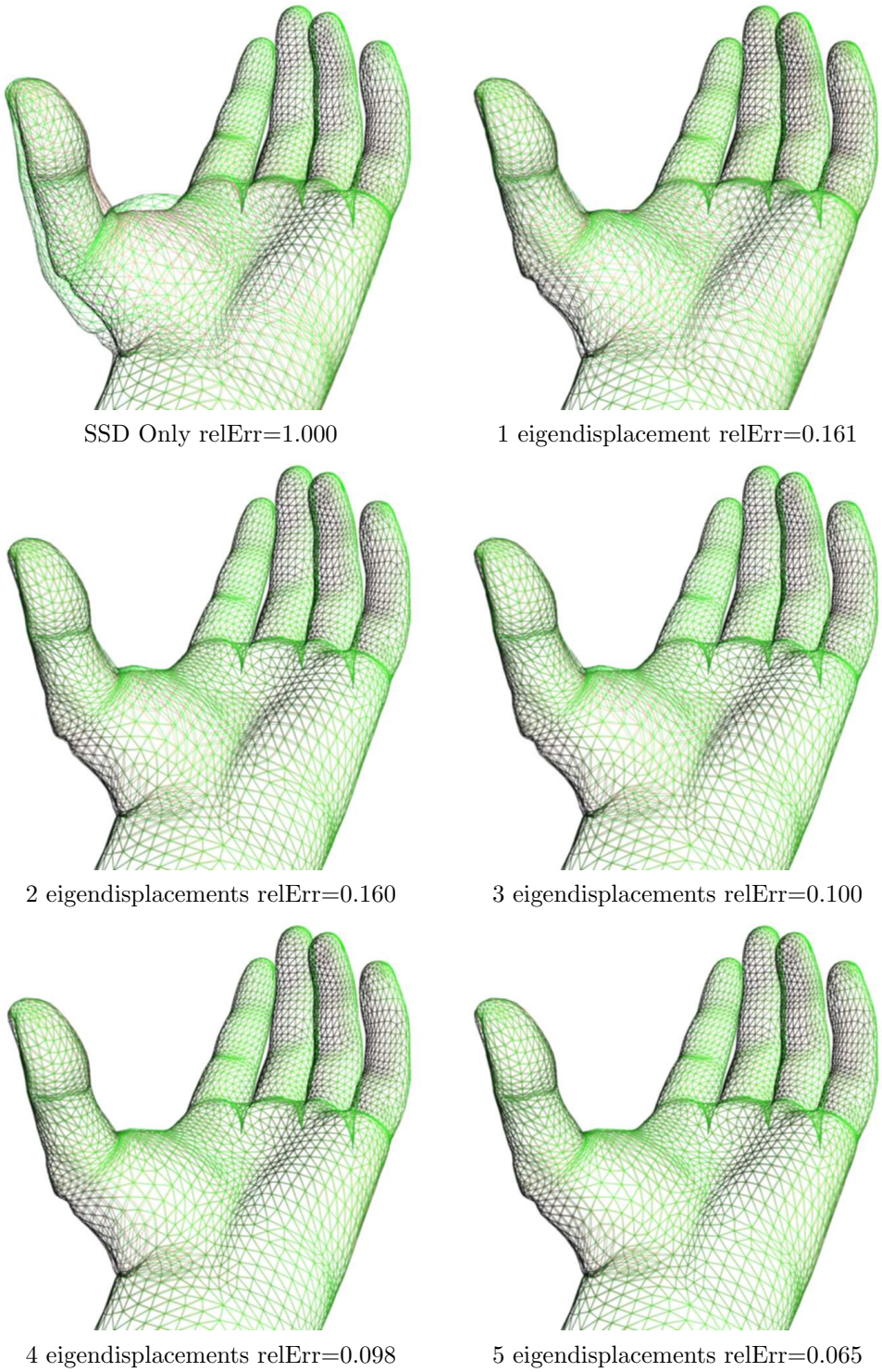
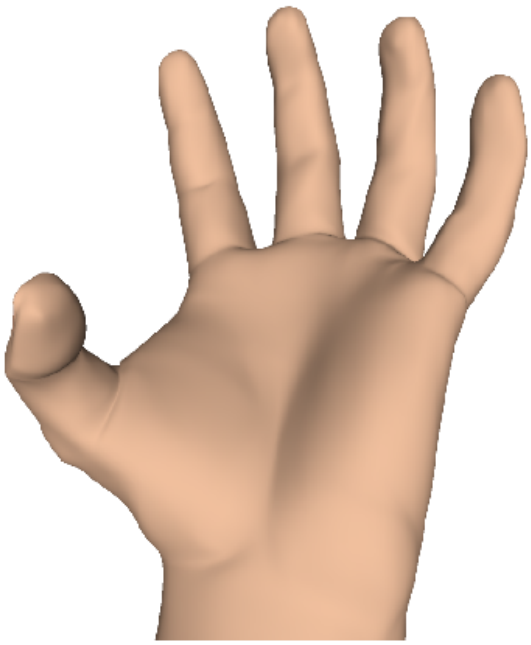
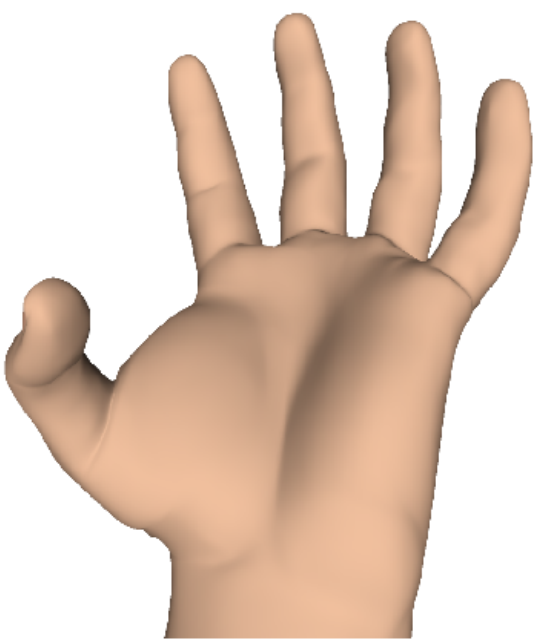


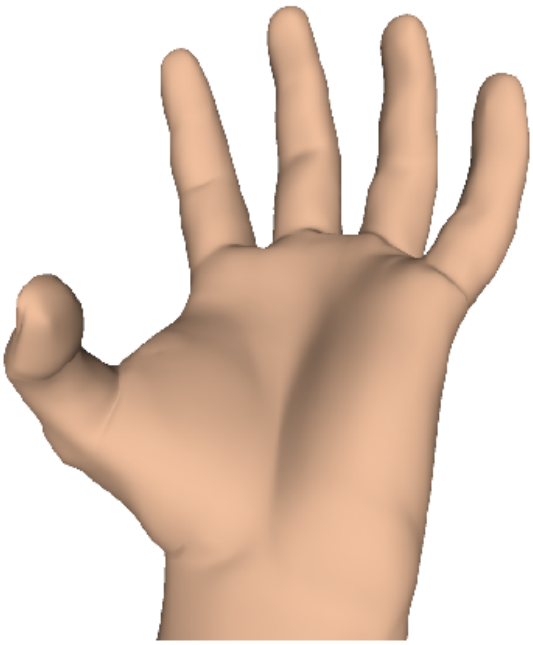
Figure 7.8: *EigenSkin approximation* for 0–5 eigendisplacements of the thumb's CM joint shown with the 13,976 triangle hand model. The pose geometry (skin colour) is approximated by the EigenSkin model (green). The l_2 relative displacement error (relErr) is also printed below each image.



a) FEM simulated pose



b) SSD only



c) EigenSkin

Figure 7.9: Comparison of EigenSkin and SSD for an extreme pose not in the training data. Note significant differences in the thumb between a) the new pose computed from our finite element hand model, b) skeletal-subspace deformation only, and c) EigenSkin with one eigendisplacement and one normal correction per support.

Figure 7.10: An EigenSkin hand example being animated using a CyberGlove. The hand model shown here consists of 55,904 triangles and is drawn using display lists with a vertex program.

only slightly slower than the traditional SSD model. Using a GeForce3 graphics card, a large 55,904 triangle hand model renders at 47 frames per second (FPS), while a coarser 13,976 triangle model achieves 181 FPS. Figure 7.11 shows shapshots from our EigenSkin shadow puppet demonstration.

Our EigenSkin approach works best when SSD corrections are localized, providing independence between different parts of the mesh, and are stable (i.e., corrections vary slowly over pose-space), allowing accurate and efficient interpolation. Under these conditions, very practical results can be obtained in which only one or two eigendisplacements per joint produce a visually dramatic improvement over commonplace Skeletal-Subspace Deformation.

Starting with a reasonable set of bone weights is important because the added displacements only correct the SSD predicted mesh shape near observed configurations. We compute our SSD bone weights as a function of vertex bone distances in the neutral pose. This yields reasonable bone weights that change smoothly over the mesh. Filtering may be required to force each bone's weights to zero at the edges of its influence to prevent discontinuities. In principle, the weights can be computed to optimize the quality of the EigenSkin correction, and this is a possible avenue for future research.

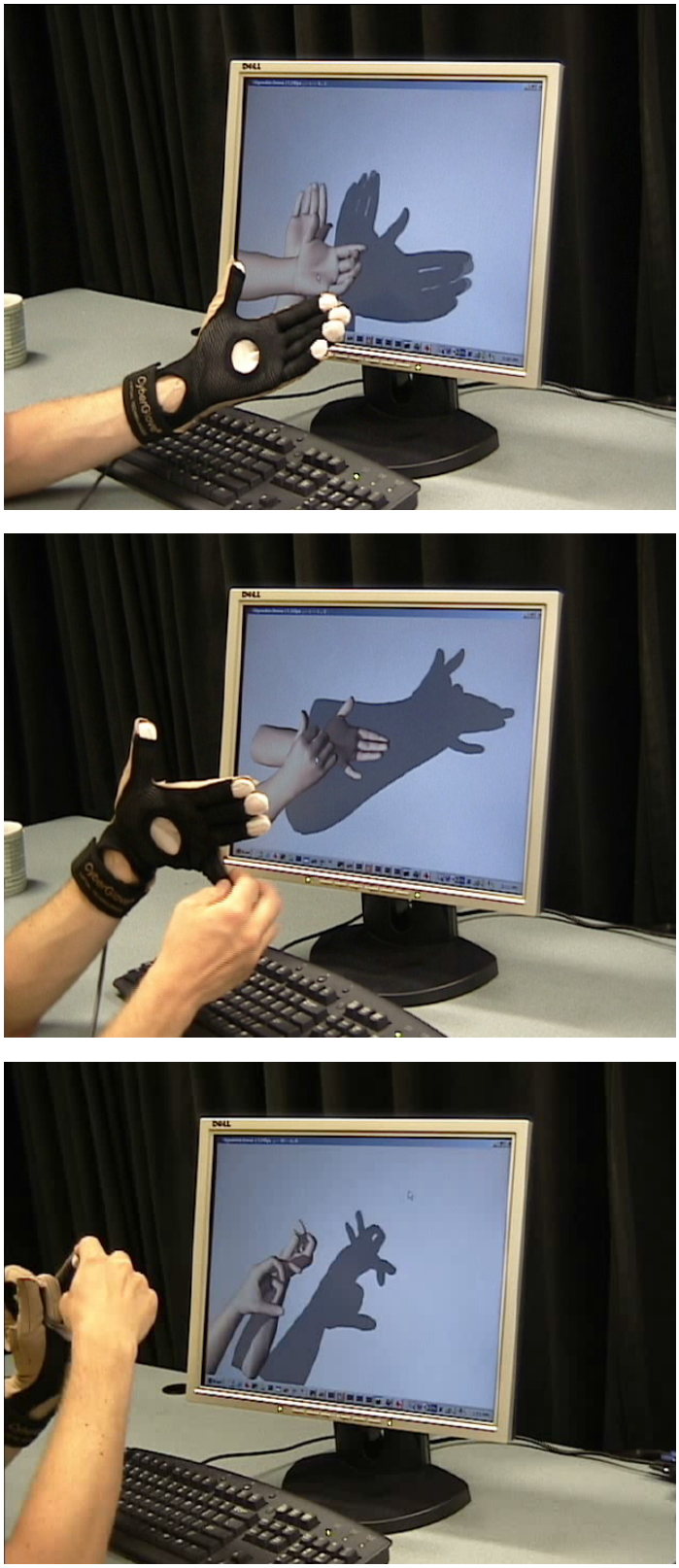


Figure 7.11: Snapshots from the eigenskin shadow puppet demonstration

7.7 EigenSkin Acquisition

In this section we describe our preliminary results towards reality based modelling of EigenSkin. The EigenSkin model uses principal component analysis to select a small basis for corrections to the existing skin. This is possible because a complete set of vectors for each individual joint's deformation response is available for analysis. A significant problem, however, is that in many cases deformation examples can only be observed for general poses. An important feature of the EigenSkin model is that the basis vector coefficients, α in Equation 7.2, are a function of only one joint. Thus, the problem of acquiring an EigenSkin model becomes a problem of determining the joint supports and deformation responses given arbitrary poses.

Our preliminary results are broken into three sections. We first describe creation of the initial SSD model and the data collection process in Section 7.7.1. Then, Section 7.7.2 describes the important step of consistent parameterization. Finally, Section 7.7.3 discusses promising ideas for data fitting.

7.7.1 Initial Model and Data Collection

EigenSkin requires an initial SSD approximation for the deformations. We build this model by exporting geometry from Poser [Curious Labs Inc.], along with the corresponding skeleton using a python script (weights are computed as described in Section 7.1). However, since this skeleton does not match the Vicon calibrated skeleton of the capture subject, we construct a transformation to map one onto the other. We use optimization to find a uniform scale and a joint configuration that minimizes the distance between corresponding joint centers. The resulting transformation is used to warp the skin (via SSD and a uniform scale) to fit the Vicon skeleton.

Surface scans are collected using a Polhemus FastScan. Sweeping the laser across the hand takes several seconds. The scanning software allows for merging of sweeps, but we use individual sweeps because it is typically difficult for the subject to hold still long enough for multiple sweeps of the same pose to merge successfully. Figure 7.12 shows an image of our setup, while Figure 7.13 shows two different sweeps. Note that we capture the configuration of the hand at the same time with the Vicon system. The method described in Section 3.4 is used to place time-stamps on the different sweeps (in fact, we time-stamp every scan line of each sweep). Because the Vicon and FastScan use different coordinate frames, we also scan the Vicon calibration target and build a transformation matrix to map scan data into the Vicon reference frame.

7.7.2 Consistent Parameterization

A consistent parameterization across all scans is a prerequisite for analysis. We have implemented an optimization-based method largely based on the template fit-

Figure 7.12: Scanning with synchronized motion capture

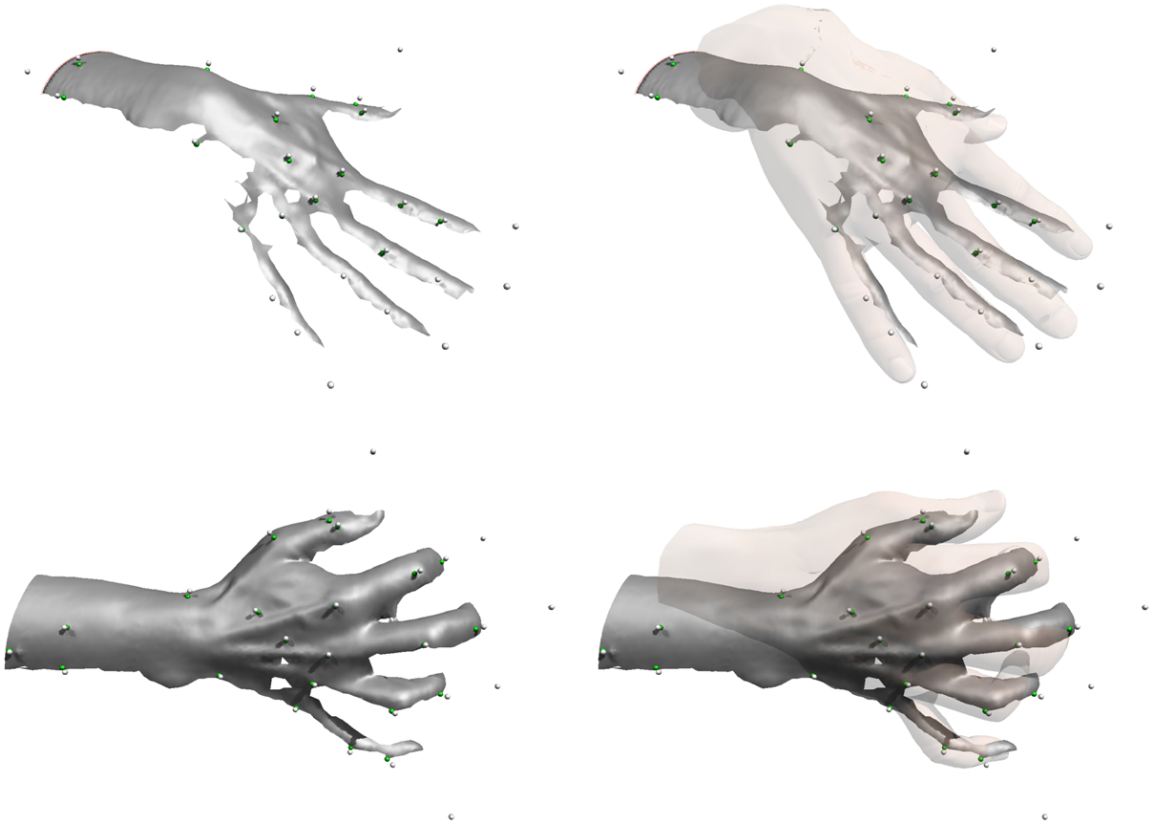


Figure 7.13: Sample hand scans (individual sweeps) taken with the Polhemus FastScan. Right column shows the Vicon posed template overlayed.

ting method presented by Allen et al. [2003]. Having Vicon motion capture data synchronized with our laser scans, we have the benefit of automatic and consistent identification of markers on the scans.

Marker locations on the scan are chosen as the closest points on the scan to the Vicon marker positions measured at the middle of the scan. Note that we could do better since we know the time of each scan line in the sweep. When Vicon markers are farther than 5 mm from the scan, we do not mark any corresponding point on the scan; instead, we use the Vicon measured marker location as a scan vertex. This effectively adds extra constraints to our reparameterization problem (where we would have had none if we were using the scan alone). Scan marker placement is seen in Figure 7.13, where the white spheres show the motion capture marker positions as measured by the Vicon and the green spheres show the corresponding positions on the surface of the scan. Markers, on occasion, are sufficiently visible in the scans for a user to place markers manually and more accurately than our automated method (see bottom left forearm marker in Figure 7.13). However, marker placement is very time consuming and typically not worth the extra effort considering that the marker data term in the optimization is removed in the last step of reparameterization (see [Allen et al. 2003]).

Rather than using a fixed template, we construct posed templates using the Vicon estimated joint angles and our initial SSD approximation. These initial posed templates do not match the scan data very well as seen in the right hand side of Figure 7.13. Once deformed to match the scan using the method described by Allen et al. [2003], they correspond fairly well (see Figure 7.14, and note that we still observe problems towards the fingertips). We could, however, use an iterative approach, and refine a pose based deformation model with each new scan. With this approach, we could expect our initial posed template mesh to correspond more closely with the scan, and in turn, we could expect a better reparameterization result after deformation.

We need not rely on the use of the Polhemus FastScan. Scanners that generate scans at faster rates may be of greater use in acquiring enough data for useful analysis. One alternative is stereo vision cameras such as the PGR Triclops or Bumblebee. Likewise, recent improvements to structured light scanners make it possible to acquire large quantities of low error data quickly, for example, 40 Hz capture of face deformations [Wang et al. 2004].

7.7.3 EigenSkin Fitting

Once we have a SSD model and a set of consistently parameterized meshes, we can then proceed with data analysis. Recall that the important feature of the EigenSkin model is that the basis vector coefficients, α in Equation 7.2, are a function of only one joint. The difficulty of constructing an EigenSkin model from arbitrary poses

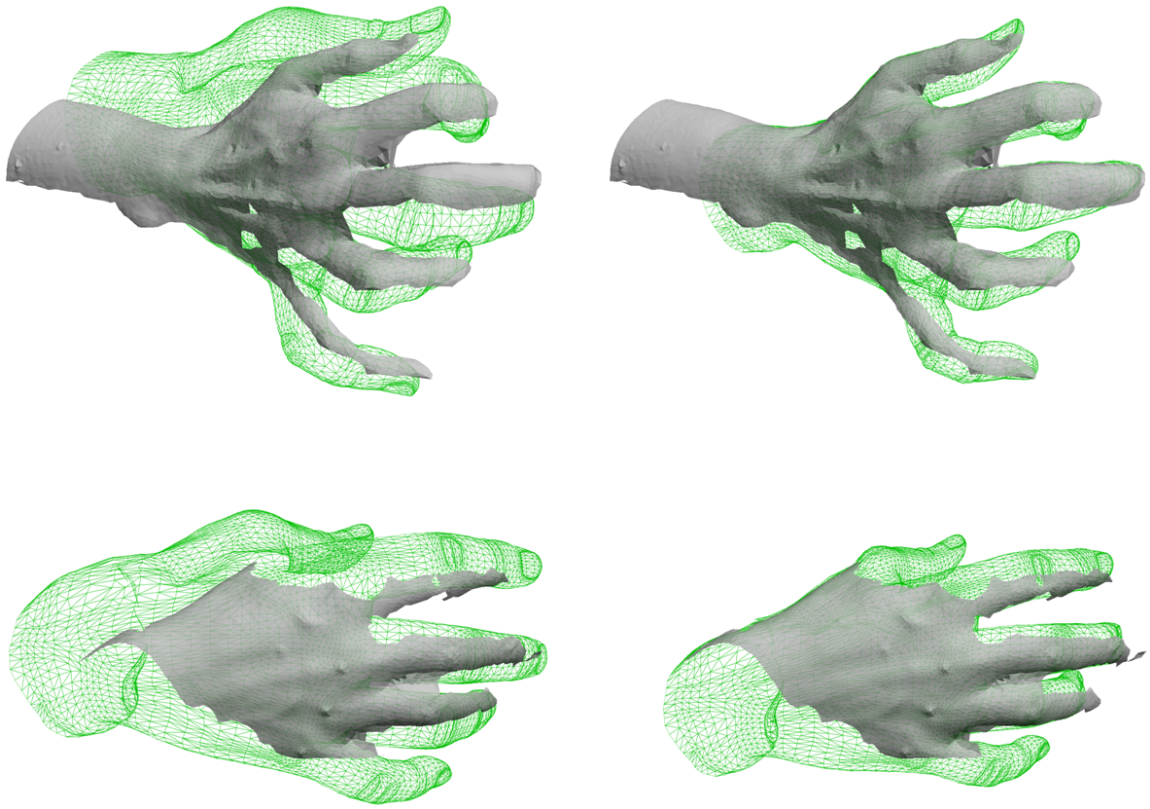


Figure 7.14: Before and after fitting the posed template to scan data.

is in determining the joint supports and deformation responses. In other words, given only general poses, we must approximate the function $u(\theta)$, which gives the corrections (displacements) required for a multi-dimensional configuration, θ. Since we expect the deformations and the corresponding corrections for an individual joint to be generally localized at that joint, we should be able to approximate the function as a linear superposition of individual joint displacements,

$$u(\theta) \approx \sum_i u_i(\theta_i). \tag{7.3}$$

The terms of this sum can be simplified with reduced bases, approximating the displacement variations observed for motions of a single joint. For general poses, however, the hard problem is finding each function $u_i(\theta_i)$. We discuss this problem further here, followed by a discussion of methods for acquiring the initial data points (samples of $u(\theta)$).

The problem has similarities to a signal separation problem. Given a set of observations, u, we must find mixing coefficients B and original signal s such that $u = Bs$. This is known as Blind Source Separation or Blind Signal Separation (BSS), and is often considered synonymous with Independent Component Analysis (ICA) [Matei

2000]. These techniques not only assume the signals are independent, but more importantly that they are statistically non-Gaussian. Given this second assumption it is possible to solve for the mixing matrix by finding the matrix which maximizes a measure of signal non-gaussianity [Hyvrinen and Oja 2000]. Unfortunately, in the case of human articulation, the desired signals are simple functions of joint angles; the joint angles of measured poses are not, in general, independent due to biological constraints. A more plausible approach is to consider the decomposition into independent signals as sufficient, provided they have limited support and that a reasonable relationship can be found between the joint angles and the signals. For example, we could possibly find correlations mapping the multi-dimensional signals with our multi-dimensional Vicon-measured joint configurations using Canonical Correlation Analysis (CCA) [Borga 1998]. Using a kernel based approach, we could possibly solve directly for the eigendisplacements and interpolating functions by assuming that the interpolating functions were smooth low degree polynomials. This is not an unreasonable assumption based on our observation of the shapes of RBF interpolation functions computed for the finite element model hand data (see Figure 7.7).

Chapter 8

Conclusions

This thesis has outlined a framework for interaction capture and synthesis that focuses on the difficulties of dealing with contact during motion capture. Our approach models the effect of contact forces on the captured system of interest. We demonstrate our framework on interactions such as touching, grasping, and manipulation motions of human hands. The key insight is that contact force measurements are important. When combined with the motion, they provide a means of estimating the passive behaviour of the system, which in our case is a hand interacting with an object or surface.

We first demonstrated methods for acquiring synchronized measurements of hand motion and contact forces. This addressed the issues of calibration, force estimation, and synchronization for a variety of different sensors. Although many different sensors can be used to capture interaction, they are best suited to particular tasks. Force torque sensors provide excellent data for compliance estimation during single point of contact interactions. The Tango presents different interesting possibilities as a self-contained interaction capture device, while in contrast, it is difficult to use for compliance estimation. The fingertip-mounted force sensitive resistors provide both good data for compliance estimation and allow a wide range of precision grasps to be easily captured, while the Mascaro Asada fingernail sensors proved to be trickier for this application. Although the use of fingertip-mounted sensors may alter the subject's motion, the effect of this is less disruptive than the perturbation in traditional system identification methods.

Compliance estimation makes our approach to reusing interaction possible. We have presented an effective technique that estimates the joint compliances for interactions captured at high temporal rates by observing the behaviour at the time of contact. This approach has the important advantage that we do not perturb the subject during the capture. We validated this approach by comparing our compliance estimation results to values reported in previous work, as well as to our own perturbation based estimations. We require compliance values for simulation, and although these values are available in the previous work on modelling hands and fingers, it is

preferable to obtain values for the specific interaction in question; The impedance used by a subject during an interaction depends on many factors including the task, intent, geometry, as well as who is performing it. Because of this, our approach is desirable in the same way that motion capture is desirable in traditional settings; it provides compliance estimates for a given captured interaction of interest.

Our novel interaction resynthesis technique incorporates a quasi-static compliant kinematic model into a dynamic environment with friction. This formulation consists of a position based linear complementarity problem that incorporates friction, breaking contact, and the compliant coupling between contacts at different fingers. For the small number of contacts that occur in precision grasps (i.e., grasps which only involve the fingertips), our resynthesis via simulation runs in real time.

Captured interaction also affords possible applications in real time interaction. Complementary to our primary contribution is our work on whole hand interaction in virtual environments. We presented methods that use a novel self-contained interaction capture device called the Tango. This consisted of two parts. First, we presented a method for approximating grasp hand-shapes from previously observed data through rotationally invariant comparison of pressure measurements. Second, we introduced methods involving heuristics and thresholds that allow for reliable drift-free position control with the Tango.

Finally, articulated movement of human hands results in complex skin shapes due to the deformation of soft tissues. Rendering these deformations is an important problem for computer animation of human hands. Addressing this problem, we have presented a technique called EigenSkin for rendering approximations of complex physically based deformations of hands and other articulated characters. EigenSkin applies pose dependent corrections to the common SSD skinning technique. Corrections are described in a memory efficient basis, allowing EigenSkin deformations to be rendered in real-time on consumer graphics hardware.

8.1 Future Work

There are a number of promising directions for future research. First, concerning compliance estimation, there are factors that we do not currently take into account that may improve our estimates. For example, correlations exist between the compliances of different joints, as well as between the compliances and the forces used during contact. These effects may be biological or neurological (e.g., Santello and Soechting [2000] examines force production synergies), and could be modelled or used as heuristics in a holistic approach to improving the compliance estimates at joints whose values are difficult to measure. Likewise, methods for determining plausible compliance variations during contact should be investigated. A possible approach to this is to look at the muscles and tendons that create the joint properties we measure at

the time of contact. Muscle tissue, in particular, has quite complex behaviour for which models exist in the literature [McMahon 1984]. It may be possible to determine muscle activations during movement, and in turn formulate expressions for the compliance during movement as a function of pose and force production.

Though we had success with single axis force measurements, it is possible that these measurements are introducing large errors in our compliance estimates. It is important investigate estimation methods further using full 6 axis force torque measurements. These measurements could be made on the fingerpad, but likewise, a different avenue of future work is to estimate the impedance of captured tool use. Examples of tools that we are currently examining include drawing instruments, such as pencils and paintbrushes, and assembly tools, such as screwdrivers.

Our quasi-static compliant kinematic model can fail to be an appropriate model for fast interactions when damping and inertia become important. For these cases, we would like to consider simulating the full dynamics of the hand. This requires the estimation of damping and inertia properties. We observe that damping is even important in the quasi-static setting. We believe that modifying our contact model to include damping forces will produce results that are more favourable. Furthermore, modifications to emulate soft fingerpads will produce the frictional torques necessary for correct simulation of two finger interactions.

We would like to explore more fully the variety of shapes to which different interaction capture examples can be successfully applied. Likewise, warping the reference trajectory will allow interactions to be reused with a much wider variety of objects. For example, we could take a two-finger grasp of a pea and apply the same interaction to a ball provided the trajectory was modified to have the fingers open wide enough for the ball to fit within the fingers. Likewise, when mapping a grasp on a large object to a small object, the trajectory would need modification such that the fingers establish contact with the surface. Previous work, such as that of Santello and Soechting [1998], will likely be useful in determining these trajectory transformations. Additionally, simulation forces may provide useful feedback for tuning the parameters used to warp the trajectory.

Note that all the hands in this thesis were the author's. The goal of this work is not to characterize all human hands, but instead to investigate methods for capturing and synthesizing a given motion of a given individual. Nevertheless, in the case of whole hand interaction with the Tango, our grasp recognition with the Tango is based on examples from one user. We would like to quantify the success of grasp recognition with both the original user and other users. It would also be interesting to use a particle filter approach for tracking grasp hand-shapes using a dimension-reduced model of observed hand shapes.

Lastly, we have taken steps towards building an EigenSkin hand model from laser scans. As mentioned in Section 7.7, the problem of missing data, laser scans of general poses, and approximate kinematics make this an interesting avenue for continued

work.

Bibliography

Albrecht, I., Haber, J., and Seidel, H.-P. 2003. Construction and animation of anatomically based human hand models. In *ACM SIGGRAPH Symposium on Computer Animation*, 98–109

Allen, B., Curless, B., and Popovic, Z. 2002. Articulated body deformation from range scan data. In *SIGGRAPH 02 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH

Allen, B., Curless, B., and Popovic, Z. 2003. The space of human body shapes: reconstruction and parameterization from range scans. In *ACM Trans. Graph.*, ACM Press, vol. 22, 587–594

Anitescu, M., and Potra, F. A. 2002. A time-stepping method for stiff multibody dynamics with contact and friction. *International Journal of Numerical Methods in Engineering 55*, 753–784.

Arbib, M. A., Iberall, T., and Lyons, D. M. 1985. Coordinated control programs for movements of the hand. *Hand Function and the Neocortex, Experimental Brain Research Supplemental 10*, 111–129.

Arikan, O., and Forsyth, D. A. 2002. Interactive motion generation from examples. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, ACM Press, 483–490

Asada, H., and Asari, Y. 1988. The direct teaching of tool manipulation skills via the impedance identification of human motions. In *IEEE International Conference on Robotics and Automation*, vol. 2, 1269–1274.

ATI Industrial Automation. Nano Force/Torque sensor, http://www.ati-ia.com/

Atkeson, C., and Hollerbach, J. 1985. Kinematic features of unrestrained vertical arm movements. *Journal of Neuroscience 5*, 9, 2318–2330

Baraff, D. 1994. Fast contact force computation for nonpenetrating rigid bodies. In *Proceedings of SIGGRAPH '94 (Orlando, Florida, July 24–29, 1994)*, ACM Press, A. Glassner, Ed., Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, 23–34. ISBN 0-89791-667-0.

Barbagli, F., Frisoli, A., Salisbury, K., and Bergamasco, M. 2004. Simulating human fingers: a soft finger proxy model and algorithm. In *Haptics Symposium*, vol. 1, 9–17

Bartlett, J. F. 2000. Rock 'n' Scroll is here to stay. *IEEE Comput. Graph. Appl. 20*, 3, 40–45.

Baud-Bovy, G., and Soechting, J. F. 2001. Two virtual fingers in the control of the tripod grasp. *Journal of Neurophysiology 86*, 604–615.

Bekey, G. A., Liu, H., Tomovic, R., and Karplus, W. J. 1993. Knowledge-based control of grasping in robot hands using heuristics from human motor skills. In *IEEE Transactions on Robotics and Automation*

Bernardin, K., Ogawara, K., Ikeuchi, K., and Dillmann, R. 2003. A hidden markov model based approach for recognizing continuous human grasping sequences. In *IEEE International Conference on Humanoid Robots*.

Bernardin, K., Ogawara, K., Ikeuchi, K., and Dillmann, R. 2005. A sensor fusion approach for recognizing continuous human grasping sequences using hidden Markov models. *IEEE Transactions on Robotics 21*, 1 (February), 47–57

Bicchi, A. 2000. Hands for dexterous manipulation and robust grasping: A difficult road towards simplicity. *IEEE Transactions on Robotics and Automation*

Bindiganavale, R., and Badler, N. 1998. Motion absraction and mapping with spatial constraints. In *Modelling and Motion Capture Techniques for Virtual Environments, CAPTECH'98*

Birch, A. S., and Srinivasan, M. A. 1999. Experimental determination of the viscoelastic properties of the human fingerpad. Tech. Rep. 632, Massachusetts Innstitute of Technology, September.

Bizzi, E., Hogan, N., Mussa-Ivaldi, F. A., and Giszter, A. 1992. Does the nervous system use equilibrium-point control to guide single and multiple joint movements? *Behavioral and Brain Sciences 15*, 603–613.

Blanz, V., and Vetter, T. 1999. A morphable model for the synthesis of 3D faces. In *SIGGRAPH 99 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH

Borga, M. 1998. *Learning Multidimensional Signal Processing*. PhD thesis, Linkoping University

Borshukov, G., Piponi, D., Larsen, O., Lewis, J. P., and Tempelaar-Lietz, C. 2003. Universal capture – image-based facial animation for "The Matrix Reloaded". In *SIGGRAPH 2003 Conference Abstracts and Applications*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH.

Borst, C., Fischer, M., and Hirzinger, G. 2004. Grasp planning: How to choose a suitable task wrench space. In *Proc. of the IEEE Intl. Conf. on Robotics and Automation*, 319– 325

Bouzit, M., Burdea, G., Popescu, G., and Boian, R. 2002. The Rutgers Master II–New design force-feedback glove. *IEEE/ASME Transactions on Mechatronics 7*, 2 (June)

Bowden, R. 2000. Learning statistical models of human motion. In *IEEE Workshop*

on Human Modelling, Analysis and Synthesis, CVPR2000

Boyd, S., and Vandenberghe, L. 2004. *Convex Optimization.* Cambridge University Press, New York, NY, USA

Bregler, C., Hertzmann, A., and Biermann, H. 2000. Recovering non-rigid 3D shape from image streams. In *IEEE Conference on Computer Vision and Pattern Recognition*, 690–696.

Burdet, E., Osu, R., Franklin, D. W., Milner, T. E., and Kawato, M. 2001. The central nervous system stabilizes unstable dynamics by learning optimal impedance. *Nature 414* (November), 446–449.

Cani-Gascuel, M.-P. 1998. Layered Deformable Models with Implicit Surfaces. In *Graphics Interface*, 201–208

Capell, S., Green, S., Curless, B., Duchamp, T., and Popovic, Z. 2002. A multiresolution framework for dynamic deformations. In *ACM SIGGRAPH Symposium on Computer Animation*

Catmull, E. 1972. A system for computer generated movies. In *Proceedings of the ACM Annual Conference*, 422–431

Chen, D. T., and Zeltzer, D. 1992. Pump it up: Computer animation based model of muscle using the finite element method. In *Computer Graphics (SIGGRAPH 92 Conference Proceedings)*, Addison Wesley, vol. 26, 89–98.

Ciocarlie, M., Miller, A., and Allen, P. 2005. Grasp analysis using deformable fingers. In *Int. Conf. Intelligent Robots and Systems*

Cline, M. B., and Pai, D. K. 2003. Post-stabilization for rigid body simulation with contact and constraints. In *IEEE International Conference on Robotics and Automation*, 3744–3751.

Cotin, S., Delingette, H., and Ayache, N. 1999. Realtime Elastic Deformations of Soft Tissues for Surgery Simulation. *IEEE Transactions On Visualization and Computer Graphics 5*, 1, 62–73.

Crossan, A., and Murray-Smith, R. 2004. Variability in wrist-tilt accelerometer-based gesture interfaces. In *MobileHCI 2004: 6th International Symposium*, 144–155.

Curious Labs Inc. Poser 4, Santa Cruz, CA., http://www.curiouslabs.com/products/poser4

Debunne, G., Desbrun, M., Barr, A., and Cani, M.-P. 2001. Dynamic Real-Time Deformations Using Space and Time Adaptive Sampling. In *SIGGRAPH 01 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH.

Dennerlein, J. T., Martin, D. B., and Hasser, C. 2000. Force-feedback improves performance for steering and combined steering-targeting tasks. In *Proc. of the Conference of Human Factors in Computing Systems (CHI)*

Dhondt, G., and Wittig, K. CalculiX: A Free Software Three-Dimensional Struc-

tural Finite Element Program, http://www.calculix.de.

Douville, B., Levison, L., and Badler, N. 1996. Task-level object grasping for simulated agents. In *Presence*

Ekvall, S., and Kragic, D. 2004. Interactive grasp learning based on human demonstration. In *IEEE International Conference on Robotics and Automation*

ElKoura, G., and Singh, K. 2003. Handrix: Animating the human hand. In *ACM SIGGRAPH Symposium on Computer Animation*, 110–119

Fang, A. C., and Pollard, N. S. 2003. Efficient synthesis of physically valid human motion. *ACM Trans. Graph. 22*, 3, 417–426

Feldman, A. G. 1986. Once more on the equilibrium-point hypothesis (lambda model) for motor control. *Journal of Motor Behavior 18*, 1 (March), 17–54.

Fitts, P. M. 1954. The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology 47*, 381–391.

Flash, T., and Hogan, N. 1985. The coordination of arm movements: an experimentally confirmed mathematical model. *Journal of Neuroscience 5*, 1688–1703

Flash, T., and Sejnowski, T. J. 2001. Computational approaches to motor control. *Current Opinion in Neurobiology 11*, 655–662.

Forsyth, B. 2004. *Intelligent Support of Interactive Manual Control: Design, Implementation and Evaluation of Look-Ahead Haptic Guidance.* Master's thesis, University of British Columbia

Gleicher, M. 1997. Motion editing with spacetime constraints. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics*

Gleicher, M. 1998. Retargeting motion to new characters. In *SIGGRAPH 1998.*

Golub, G. H., and van Loan, C. F. 1996. *Matrix Computations*, third ed. Johns Hopkins University Press, Baltimore and London.

Gomi, H., and Kawato, M. 1997. Human arm stiffness and equilibrium-point trajectory during multi-joint movement. *Biological Cybernetics 76*, 163–171.

Gourret, J., Magnenat-Thalmann, N., and Thalmann, D. 1989. Simulation of Object and Human Skin Deformations in a Grasping Task. In *Computer Graphics (SIGGRAPH 89 Conference Proceedings)*, Addison Wesley

Goyal, S., Ruina, A., and Papadopaulos, J. 1991. Planar sliding with dry friction part 1: Limit surface and moment function. *Wear 142*, 2, 307–330.

Gulati, R. J., and Srinivasan, M. A. 1995. Human fingerpad under indentation I: Static and dynamic force response. In *Bioengineering Conference ASME*, vol. 29, 261–262.

Gulati, R. J., and Srinivasan, M. A. 1997. Determination of mechanical properties of the human fingerpad in vivo using a tactile stimujlator. Tech. Rep. 605, Massachusetts Innstitute of Technology, January.

Gyration. Gyromouse, http://www.gyration.com/gyromouse.htm

Hager-Ross, C., and Schieber, M. H. 2000. Quantifying the independence of

human finger movments: Comparisons of digits, hands and movement frequencies. *The Journal of Neuroscience 20*, 22, 8542–8550.

Haggard, P., and Wing, A. 1998. Coordination of hand aperture with the spatial path of hand trasport. *Exp Brain Res*, 118, 286–292.

Hajian, A. Z. 1997. *A Characterization of the Mechanical Impedance of Human Hands*. PhD thesis, Harvard.

Han, H.-Y., Shimada, A., and Kawamura, S. 1996. Analysis of friction on human fingers and design of artificial fingers. In *IEEE International Conference on Robotics and Automation*, 3061–3066

Harris, C. M., and Wolpert, D. M. 1998. Signal-dependent noise determines motor planning. *Nature 394*, 780–784

Hasser, C. J., and Cutkosky, M. R. 2002. System identificaiton of the human hand grasping a haptic knob. In *Proceedings of the 10th Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems (HAPTICS'02)*

Hodgins, J. K., and Pollard, N. S. 1997. Adapting simulated behaviors for new characters. In *SIGGRAPH '97: Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 153–162.

Hogan, N. 1984. An organizing principle for a class of voluntary movements. *Journal of Neuroscience 4*, 2745–2754.

Huang, Z., Boulic, R., Thalmann, N. M., and Thalmann, D. 1995. A multi-sensor approach for grasping and 3D interaction. In *Computer Graphics: Developments in Virtual Environments*, Academic Press Ltd., London, UK, 235–253

Hyvrinen, A., and Oja, E. 2000. Independent component analysis: Algorithms and applications. *Neural Networks*, 411–430.

Immersion Corporation. CyberGlove and CyberGrasp, http://www.immersion.com/

Insko, B., Meehan, M., Whitton, M., and Jr, F. P. B. 2001. Passive haptics significantly enhances virtual environments. Tech. rep., Computer Science Technical Report 01-010, University of North Carolina, Chapel Hill, NC.

Interlink Electronics. Force Sensing Resistors, http://www.micronavlink.com/technology.html

Irving, G., Teran, J., and Fedkiw, R. 2004. Invertible finite elements for robust simulation of large deformation. In *ACM SIGGRAPH Symposium on Computer Animation*, 131–140.

Isard, M., and Blake, A. 1998. Condensation – conditional density propagation for visual tracking. *International Journal of Computer Vision 28*, 1, 5–28

James, D. L., and Pai, D. K. 1999. ArtDefo: Accurate Real Time Deformable Objects. In *SIGGRAPH 99 Conference Proceedings*, Addison Wesley, Annual Con-

ference Series, ACM SIGGRAPH, 65–72.

James, D. L., and Pai, D. K. 2002. DyRT: Dynamic response textures for real time deformation simulation with graphics hardware. In *SIGGRAPH 02 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH

James, D. L., and Pai, D. K. 2002. Real time simulation of elastokinematic models. In *ICRA2002: IEEE International Conference on Robotics and Automation*.

Jerde, T. E., Soechting, J. F., and Flanders, M. 2003. Biological constraints simplify the recognition of hand shapes. *IEEE Transactions on Biomedical Engineering 50*, 2 (February), 265–269.

Johansson, R. S. 1996. *Sensory and memory information in the control of dextrous manipulation*. Kluwer Academic, 205–260

Johnson, K. L. 1985. *Contact Mechanics*. Cambridge University Press.

Jolliffe, I. 1986. *Principal Component Analysis*. Springer Verlag.

Kang, S., and Ikeuchi, K. 1994. Grasp recognition and manipulative motion characterization from human hand motion sequences. In *IEEE International Conference on Robotics and Automation*, 1759–1764

Kang, S. B., and Ikeuchi, K. 1997. Toward automatic robot instruction from perception – mapping human grasps to manipulator grasps. *IEEE Transactions on Robotics and Automation 13*, 1

Kang, S. B. 1994. *Robot Instruction by Human Demonstration*. PhD thesis, Carnigie Mellon University

Kazhdan, M., Funkhouser, T., and Rusinkiewicz, S. 2003. Rotation invariant spherical harmonic representation of 3D shape descriptors. In *Proceedings of the Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, Eurographics Association, 156–164

Kim, J., Cordier, F., and Magnenat-Thalmann, N. 2000. Neural network-based violinist's hand animation. In *Proceedings of the International Conference on Computer Graphics*, IEEE Computer Society, 37.

Koga, Y., Kondo, K., Kuffner, J., and Latombe, J.-C. 1994. Planning motions with intentions. In *SIGGRAPH 94*

Kovar, L., Gleicher, M., and Pighin, F. 2002. Motion graphs. In *SIGGRAPH 2002*

Kovar, L., Schreiner, J., and Gleicher, M. 2002. Footskate cleanup for motion capture editing. In *ACM SIGGRAPH Symposium on Computer Animation*

Kry, P. G., and Pai, D. K. 2003. Continuous contact simulation for smooth surfaces. *ACM Trans. Graph. 22*, 1, 106–129

Kry, P. G., James, D. L., and Pai, D. K. 2002. Eigenskin: Real time large deformation character skinning in hardware. In *ACM SIGGRAPH Symposium on Computer Animation*

Kurihara, T., and Miyata, N. 2004. Modeling deformable human hands from

medical images. In *ACM SIGGRAPH Symposium on Computer Animation*, 357–365.

Lang, J., Pai, D. K., and Seidel, H.-P. 2003. Scanning large-scale articulated deformations. In *Grapics Interface*

Larsen, E., Gottschalk, S., Lin, M., and Manocha, D. 1999. Fast proximity queries with swept sphere volumes. Tech. rep., TR99-018, Department of Computer Science, University of North Carolina

Lee, P., Wei, S., Zhao, J., and Badler, N. I. 1990. Strength guided motion. In *SIGGRAPH '90: Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, ACM Press, New York, NY, USA, 253–262.

Lee, Y., Terzopoulos, D., and Walters, K. 1995. Realistic Modeling for Facial Animation. In *SIGGRAPH 95 Conference Proceedings*, Addison Wesley, vol. 29 of *Annual Conference Series*, ACM SIGGRAPH, 55–62

Lee, J., Chai, J., Reitsma, P., Hodgins, J. K., and Pollard, N. 2002. Interactive control of avatars animated with human motion data. In *Proceedings of SIGGRAPH 2002*

Lewis, J. P., Cordner, M., and Fong, N. 2000. Pose Space Deformations: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation. In *SIGGRAPH 00 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH

Lin, J., We, Y., and Huang, T. S. 2000. Modeling the constraints of human hand motion. In *IEEE Workshop on Human Motion*.

Lin, J. Y., Wu, Y., and Huang, T. S. 2004. 3D model-based hand tracking using stochastic direct search method. In *Sixth IEEE International Conference on Automatic Face and Gesture Recognition*, 693–698

Lindholm, E., J.Kilgard, M., and Moreton, H. 2001. A User-Programmable Vertex Engine. In *SIGGRAPH 01 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH.

Liu, Y., and Badler, N. I. 2003. Real-time reach planning for animated characters using hardware acceleration. In *Proceedings of the 16th International Conference on Computer Animation and Social Agents (CASA'03)*, 86–93.

Lloyd, J. E. 2005. Fast implementation of Lemke's algorithm for rigid body contact simulation. In *IEEE International Conference on Robotics and Automation*.

Loop, C. 1987. *Smooth Subdivision Surfaces Based on Triangles*. Master's thesis, University of Utah, Department of Mathematics.

Lotstedt, P. 1981. Coulomb friction in two-dimensional rigid-body systems. In *Zeitschrift fur Angewandte Mathematik und Mechanik*, 605–615.

Lu, S., Metaxas, D., Samaras, D., and Oliensis, J. 2003. Using multiple cues for hand tracking and model refinement. In *IEEE International Conference on Computer Vision*, vol. II, 443–450

MacKenzie, C. L., and Iberall, T. 1994. *The Grasping Hand*. North-Holland.

Maestri, G. 1999. *Digital Character Animation 2, Vol. 1*. New Rider, Indianapolis.

Magnenat-Thalmann, N., and Thalmann, D. 1991. Human Body Deformations Using Joint-dependent Local Operators and Finite Element Theory. In *Making Them Move: Mechanics, Control, and Animation of Articulated Figures*. Morgan Kaufmann, 243–262.

Magnenat-Thalmann, N., Laperriere, R., and Thalmann, D. 1988. Joint-dependent Local Deformations for Hand Animation and Object Grasping. In *Proc. of Graphics Interface '88*, 26–33.

Mascaro, S., and Asada, H. 2001. Photoplethysmograph fingernail sensors for measuring finger forces without haptic obstruction. *IEEE Transactions on Robotics and Automation 17*, 5, 698–708

Mascaro, S., and Asada, H. 2004. Measurement of finger posture and three-axis fingertip touch force using fingernail sensors. *IEEE Transactions on Robotics and Automation 20*, 1 (February), 26–35

Matei, B., 2000. A review of independent component analysis techniques. http://www.caip.rutgers.edu/riul/research/tutorials/tutorialica.pdf

McMahon, T. A. 1984. *Muscles, Reflexes, and Locomotion*. Princeton University Press, Princeton, New Jersey.

Metaxas, D., and Terzopoulos, D. 1992. Dynamic Deformation of Solid Primitives with Constraints. In *Computer Graphics (SIGGRAPH 92 Conference Proceedings)*, Addison Wesley, vol. 26, 309–312.

Miller, A. T., and Allen, P. K. 2000. Graspit!: A versatile simulator for grasp analysis. In *ASME International Mechanical Engineering Congress & Exposition*.

Miller, A. T., and Christensen, H. I. 2003. Implementation of multi-rigid-body dynamics within a robotic grasping simulator. In *IEEE International Conference on Robotics and Automation*, vol. 2, 2262–2268.

Milner, T. E., and Franklin, D. W. 1998. Characterization of multijoint finger stiffness: dependence on finger posture and force direction. *IEEE Transactions on Biomedical Engineering 45*, 11 (November), 1363–1375

Mohr, A., and Gleicher, M. 2003. Building efficient, accurate character skins from examples. *ACM Trans. Graph. 22*, 3, 562–568

Moore, K. L., and Dalley, A. F. 1999. *Clinically Oriented Anatomy*, fourth ed. Williams & Wilkins, April.

Muller, M., Dorsey, J., McMillan, L., Jagnow, R., and Cutler, B. 2002. Stable real-time deformations. In *ACM SIGGRAPH Symposium on Computer Animation*

Murty, K. G. 1988. *Linear Complementarity, Linear and Nonlinear Programming*. Heldermann Verlag, Berlin

Neff, M., and Fiume, E. 2002. Modeling tension and relaxation for computer

animation. In *ACM SIGGRAPH Symposium on Computer Animation*

Northern Digital Inc. Aurora, http://www.ndigital.com/aurora.php

O'Brien, J. F., and Hodgins, J. K. 1999. Graphical Modeling and Animation of Brittle Fracture. In *SIGGRAPH 99 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH, 111–120.

Pai, D. K., and Rizun, P. 2003. The WHaT: a Wireless Haptic Texture Sensor. In *Eleventh Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*.

Pai, D. K., van den Doel, K., James, D. L., Lang, J., Lloyd, J. E., Richmond, J. L., and Yau, S. H. 2001. Scanning Physical Interaction Behavior of 3D Objects. In *SIGGRAPH 01 Conference Proceedings*, Addison Wesley, Annual Conference Series, ACM SIGGRAPH

Pai, D. K., Sueda, S., and Wei, Q. 2005. Fast physically based musculoskeletal simulation. *conditionally accepted for publication in ACM Transactions on Graphics* (April).

Pai, D. K., VanDerLoo, E. W., Sadhukan, S., and Kry, P. G. 2005. The Tango: A tangible tangoreceptive whole-hand human interface. In *Proceedings of WorldHaptics (Joint Eurohaptics Conference and IEEE Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems)*.

Park, K.-H., Kim, B.-H., and Hirai, S. 2003. Development of a soft-fingertip and its modeling based on force distribution. In *IEEE International Conference on Robotics and Automation*, vol. 3, 3169–3174.

Parke, F. I., and Waters, K. 1996. *Computer Facial Animation*. A. K. Peters Ltd.

Pauly, M., Pai, D. K., and Guibas, L. J. 2004. Quasi-rigid objects in contact. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM Press, New York, NY, USA, 109–119

Pawluk, D. 1997. *A Viscoelastic Model of the Human Fingerpad and a Holistic Model of Human Touch*. PhD thesis, Harvard.

Picinbono, G., Delingette, H., and Ayache, N. 2001. Non-linear and anisotropic elastic soft tissue models for medical simulation. In *ICRA2001: IEEE International Conference on Robotics and Automation*

Pollard, N. S., and Hodgins, J. K. 2002. Generalizing demonstrated manipulation tasks. In *Workshop on the Algorithmic Foundations of Robotics (WAFR '02)*.

Pollard, N. S., and Wolf, A. 2004. Grasp synthesis from example: tuning the example to a task or object. In *Workshop on Multi-point Interaction in Robotics and Virtual Reality, IEEE International Conference on Robotics and Automation*

Pollard, N. S., and Zordan, V. B. 2005. Physically based grasping control from example. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics*

Symposium on Computer Animation, ACM Press, New York, NY, USA, 311–318

Pollard, N. S. 2004. Closure and quality equivalence for efficient synthesis of grasps from examples. *International Journal of Robotics Research 23*, 6, 596–614

Popovic, Z., and Witkin, A. 1999. Physically based motion transformation. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, ACM Press/Addison-Wesley Publishing Co., 11–20.

Powell, M. J. D. 1987. Radial basis functions for multivariate interpolation: A review. In *Algorithms for Approximation*, J. C. Mason and M. G. Cox, Eds. Clarendon Press, Oxford.

Pratscher, M., Coleman, P., Laszlo, J., and Singh, K. 2005. Outside-in anatomy based character rigging. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, ACM Press, New York, NY, USA, 329–338

Pressure Profile Systems Inc. ConTacts, http://www.pressure-profile.com

Quinlan, S., and Khatib, O. 1993. Elastic bands: Connecting path planning and control. In *IEEE International Conference on Robotics and Automation*, 802–807.

Quinlan, S. 1994. Efficient distance computation between non-convex objects. In *IEEE International Conference on Robotics and Automation*, 3324–3330.

Rancourt, D., and Hogan, N. 2001. Stability in force-production tasks. *Journal of Motor Behavior 33*, 2, 193–204.

Rearick, M. P., and Santello, M. 2002. Force synergies for multifingered grasping: effect of predictability in object center of mass and handedness. *Exp Brain Res*, 144, 38–49.

Rezzonico, S., Boulic, R., Huang, Z., Magnenat-Thalmann, N., and Thalmann, D. 1995. Consistent grasping interactions with virtual actors based on the multi-sensor hand model. In *EGVE95*

Richardson, M. J. E., and Flash, T. 2002. Comparing smooth arm movements with the two-thirds power law and related segmented-control hypothesis. *Journal of Neuroscience 22*, 18, 8201–8211

Rijpkema, H., and Girard, M. 1991. Computer animation of knowledge-based human grasping. In *SIGGRAPH 91*

Rose, C., Guenter, B., Bodenheimer, B., and Cohen, M. F. 1996. Efficient generation of motion transitions using spacetime constraints. In *SIGGRAPH 1996*

Rose, C. F., Sloan, P. J., and Cohen, M. F. 2001. Artist-directed inverse-kinematics using radial basis function interpolation. In *Eurographics*, vol. 20

Rosenbaum, D. A., Meulenbroek, R. J., Vaughan, J., and Jansen, C. 2001. Posture-based motion planning: applications to grasping. *Psychological Review 108*, 4, 709–734.

Safonova, A., Hodgins, J. K., and Pollard, N. S. 2004. Synthesizing physi-

cally realistic human motion in low-dimensional, b. *ACM Trans. Graph. 23*, 3

Sand, P., McMillan, L., and Popovic, J. 2003. Continuous capture of skin deformation. In *ACM Transactions on Graphics*, vol. 22, 578–586

Sanso, R. M., and Thalmann, D. 1994. A hand control and automatic grasping system for synthetic actors. In *Eurographics, Computer Graphics Forum*.

Santello, M., and Soechting, J. F. 1997. Matching object size by controlling finger span and hand shape. *Somatosensory and Motor Research 14*, 3, 203–212.

Santello, M., and Soechting, J. F. 1998. Gradual molding of the hand to object contours. *The Journal of Neurophysiology 79*, 3 (March), 1307–1320.

Santello, M., and Soechting, J. F. 2000. Force synergies for multifingered grasping. *Exp Brain Res*, 133, 457–467.

Santello, M., Flanders, M., and Soechting, J. 1998. Postural hand synergies for tool use. In *The Journal of Neuroscience*

Santello, M., Flanders, M., and Soechting, J. F. 2002. Patterns of hand motion during grasping and the influence of sensory guidance. *Journal of Neuroscience 22*, 4, 1426–1435

Scheepers, F., Parent, R. E., Carlson, W. E., and May, S. F. 1997. Anatomy-Based Modeling of the Human Musculature. In *SIGGRAPH 97 Conference Proceedings*, Addison Wesley, vol. 31 of *Annual Conference Series*, ACM SIGGRAPH, 163–172

Schoberl, J. 1997. NETGEN - An advancing front 2D/3D-mesh generator based on abstract rules. *Comput.Visual.Sci 1*, 41–52.

Shimoga, K. B. 1996. Robot grasp synthesis algorithms: A survey. *International Journal of Robotics Research 15*, 3, 230–266.

Singh, K., and Kokkevis, E. 2000. Skinning characters using surface-oriented free-form deformations. In *Graphics Interface 2000*, 35–42.

Sloan, P.-P. J., Rose, C. F., and Cohen, M. F. 2001. Shape by example. In *2001 Symposium on Interactive 3D Graphics*, 135–143

Soechting, J. F., and Flanders, M. 1989. Sensorimotor representations for pointing to targets in three dimensionsal space. *Journal of Neurophysiology 62*, 2, 582–594.

Soechting, J. F., Buneo, C. A., Herrmann, U., and Flanders, M. 1995. Moving effortlessly in three dimensions: Does donders' law apply to arm movement? *The Journal of Neuroscience 15*, 9 (September), 6271–6280

Springer, S., and Ferrier, N. 1999. Design of a multi-finger haptic interface for teleoperational grasping. In *ASME International Mechanical Engineering Congress and Exposition*

Stewart, D. E., and Trinkle, J. C. 1996. An implicit time-stepping scheme for rigid body dynamics with inelastic collisions and coulomb friction. *International J. Numer. Methods Engineering 39*, 2673–2691.

Tactex Controls Inc. MTC Express, http://www.tactex.com

Teran, J., Blemker, S., Hing, V. N. T., and Fedkiw, R. 2003. Finite volume methods for the simulation of skeletal muscle. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 68–74.

Terzopoulos, D., and Fleischer, K. 1988. Deformable Models. *The Visual Computer 4*, 306–331.

Terzopoulos, D., and Witkin, A. 1988. Physically-based Models with Rigid and Deformable Components. *IEEE Computer Graphics and Applications 8*, 6, 41–51.

Todorov, E., and Jordan, M. I. 2002. Optimal feedback control as a theory of motor coordination. *Nature Neuroscience 5*, 1226–1235.

Tsang, W., Singh, K., and Fiume, E. 2005. Helping hand: an anatomically accurate inverse dynamics solution for unconstrained hand motion. In *SCA '05: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM Press, New York, NY, USA, 319–328

Tsang, W. 2005. *Helping Hand: An Anatomically Accurate Inverse Dynamics Solution for Unconstrained Hand Motion*. Master's thesis, University of Toronto.

Uno, Y., Kawato, M., and Suzuki, R. 1989. Formation and control of optimal trajectory in human multijoint arm movement. minimum torque change model. *Biol. Cybern. 61*, 89–102.

Vicon Motion Systems. http://www.vicon.com

Viviani, P., and Flash, T. 1995. Minimum-jerk, two-thirds power law, and isochrony: converging approaches to movement planning. *Journal of Experimental Psychology 21*, 1.

Wagner, C. R., Stylopoulos, N., and Howe, R. D. 2002. The role of force feedback in surgery: Analysis of blunt dissection. In *Tenth Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 73–79

Wang, X. C., and Phillips, C. 2002. Multi-weight enveloping: least-squares approximation techniques for skin animation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ACM Press, 129–138

Wang, Y., Huang, X., Lee, C.-S., Zhang, S., Li, Z., Samaras, D., Metaxas, D., Elgammal, A., and Huang, P. 2004. High resolution acquisition, learning and transfer of dynamic 3-D facial expressions. In *Eurographics, Computer Graphics Forum*, vol. 23.

Wigdor, D., and Balakrishnan, R. 2003. TiltText: using tilt for text input to mobile phones. In *UIST '03: Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology*, ACM Press, New York, NY, USA, 81–90.

Wilhelms, J., and van Gelder, A. 1997. Anatomically Based Modeling. In *SIGGRAPH 97 Conference Proceedings*, Addison Wesley, vol. 31 of *Annual Conference*

Series, ACM SIGGRAPH, 173–180

Wolpert, D. M., and Ghahramani, Z. 2000. Computational principles of movement neuroscience. *Nature Neuroscience Supplement 3*, 1212–1217.

Xu, Y., and Hollerbach, J. M. 1999. A robust ensemble data method for identification of human joint mechanical properties during movement. *IEEE Transactions on Biomedical Engineering 46*, 4 (April), 409–419

Yamane, K., and Nakamura, Y. 2003. Natural motion animation through constraining and deconstraining at will. *IEEE Transactions on Visualization and Computer Graphics 9*, 3, 352–360

Yamane, K., Kuffner, J. J., and Hodgins, J. K. 2004. Synthesizing animations of human manipulation tasks. In *ACM Trans. Graph.*

Yasumuro, Y., Chen, Q., and Chihara, K. 1997. 3D modeling of human hand with motion constraints. In *International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, 275–282

Yin, K., and Pai, D. K. 2003. Footsee: an interactive animation system. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, Eurographics Association, 329–338.

Yin, K., Cline, M. B., and Pai, D. K. 2003. Motion perturbation based on simple neuromotor control models. In *Proceedings of the 11th Pacific Conference on Computer Graphics and Applications*, 445–449.

Zajac, F. E. 1989. Muscle and tendon: properties, models, scaling and applicaiton to biomechanics and motor control. *CRC Critical Reviews in Biomedical Engineering 17*, 359–411.

Zatsiorsky, V. M., Gao, F., and Latash, M. L. 2003. Prehension synergies: Effects of object geometry and prescribed torques. *Experimental Brain Research*, 148, 77–87.

Zhai, S., Milgram, P., and Buxton, W. 1996. The influence of muscle groups on performance of multiple degree-of-freedom input. In *Proceedings of CHI '96*, 308–315.

Zhang, L., Snavely, N., Curless, B., and Seitz, S. M. 2004. Spacetime faces: High-resolution capture for modeling and animation. In *ACM Annual Conference on Computer Graphics*, to appear.

Zhao, J., and Badler, N. I. 1994. Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Trans. Graph. 13*, 4, 313–336.

Zhuang, Y., and Canny, J. 1999. Real-time Simulation of Physically Realistic Global Deformation. In *IEEE Vis'99 Late Breaking Hot Topics*.

Zienkiewicz, O. C. 1977. *The Finite Element Method*. McGraw-Hill Book Company (UK) Limited, Maidenhead, Berkshire, England.

Zordan, V. B., and Hodgins, J. K. 2002. Motion capture-driven simulations that hit and react. In *ACM SIGGRAPH Symposium on Computer Animation*

Appendix A

FNS Synchronization Circuit

Careful timing of fingernail sensor measurements is essential because the photodiodes in these sensors are sensitive to the light emitted by the Vicon strobes. The circuit shown in Figure A.1 solves this problem. It takes two readings for each Vicon frame, allowing for background light subtraction (just like that done by the Vicon system).

The circuit consists of an adjustable frequency generator, set to approximately 1 kHz, and a synchronous counter for frequency division. Figure A.2 shows a timing diagram and a summary of events. Once the counter reaches its terminal count, the frequency generator is stopped until another Vicon strobe occurs. During each count from zero to 15 (the terminal count), the start scan signal does a low to high transition twice. On each transition the DAQPad rapidly samples all 30 photodiodes. We use a double buffer data collection mode; we copy full buffers as they become available into an expandable data structure, allowing acquisition of continuous uninterrupted measurements for an arbitrary length of time (limited only by available memory or disk). Data from 30 sensors at 60 Hz (sampled at 120 Hz for light subtraction) consumes only 3.5 kB/s.

The photodiode signals are first amplified on the sensor circuit mounted at the fingernail. To reduce signal noise, Mascaro and Asada suggest a simple resistor-capacitor (RC) low pass filter. Because of interference from the Vicon strobe, we want the cut-off frequency to be high enough to allow the filtered voltage-spike to decay before we take measurements. Figure A.3 shows measured timing signals and photodiode signals filtered with two different filters. The cut-off frequency where the filter attenuates the signal power by half is computed as $(2\pi RC)^{-1}$; for R = 10 kΩ it is 138 Hz and for R = 5 kΩ it is 276 Hz.

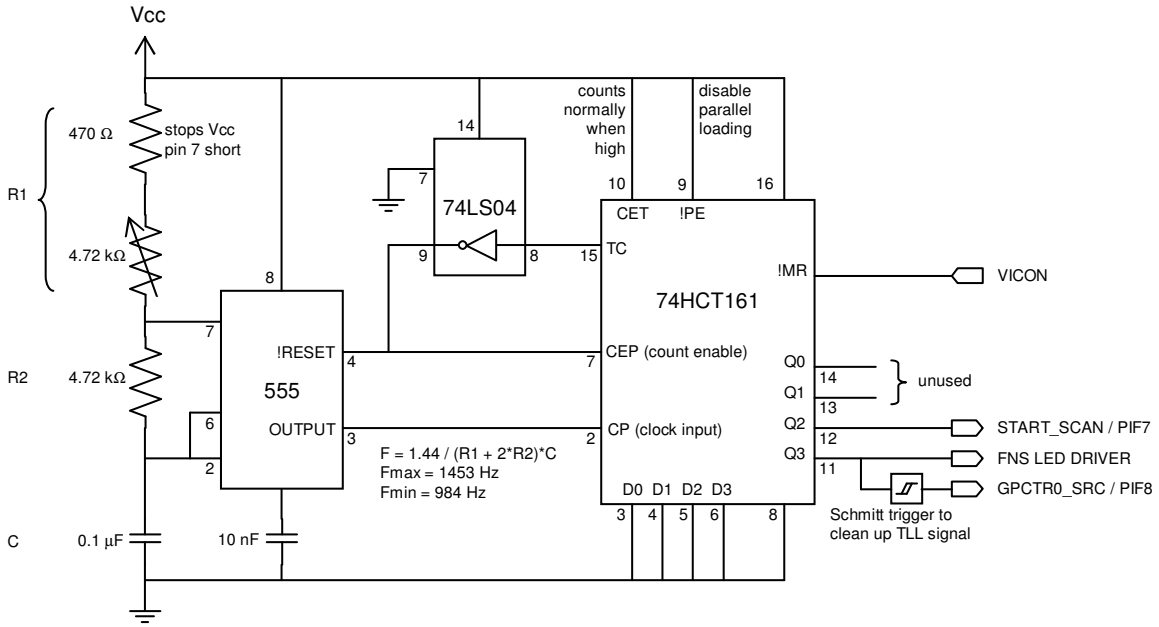
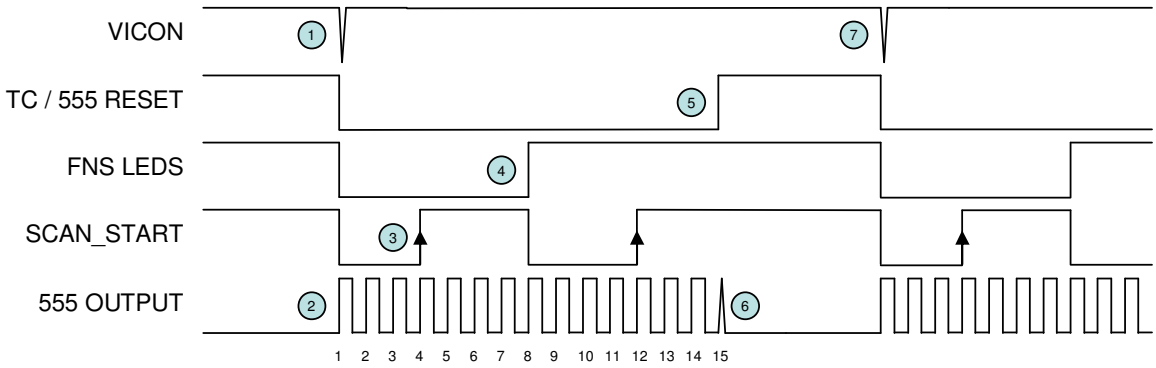


Figure A.1: Circuit for synchronizing FNS and Vicon data. Note that this circuit, the Vicon box, and the DAQPad must have a common ground.



1. Vicon frame capture is signalled by a brief low signal. This resets the counter, sending `TC` low, allowing the timer to come out of the reset state.
2. The timer resumes astable oscillation (the duty cycle is not 0.5 as drawn, but counting occurs on rising edges regardless the duty cycle).
3. Analog to digital conversion occurs on the rising edges of `START_SCAN`. This occurs when `Q2`, the fours bit, first goes high at count 4, and again at 12 as marked by the arrows.
4. The LEDs are illuminated when `Q3`, the eights bit of the counter, is high.
5. The counter's Terminal Count, `TC`, is reached at 15, and would normally go low on the next clock pulse when the count wraps to zero, except that `TC` resets the timer and holds the timer in reset until the counter is reset.
6. With `TC` high, `!RESET` on the 555 timer goes low and causes the output to become low.
7. The process repeats with another low pulse from the Vicon system.

Figure A.2: Timing diagram for the synchronization circuit with corresponding explanations of different events.

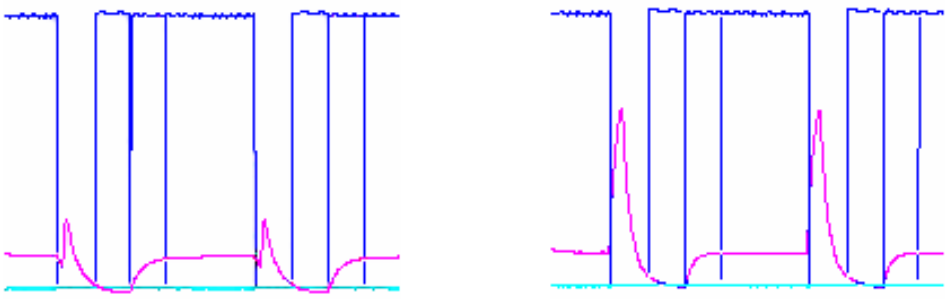


Figure A.3: Timing signals measured on the DAQPad with two different resistor-capacitor (RC) filters. Left shows a R = 10 kΩ, C = 115 nF and right shows R = 5 kΩ, C = 115 nF (twice the cutoff frequency). Blue corresponds to `START_SCAN` and LED signals, and magenta to the photodiode reading.