The Australian National University

Research School of Information Sciences and Engineering

&

The Faculty of Engineering and Information Technology

Advanced Human Hand Model with Dynamic Constraints

Priscilla Kan John

u3928684

Supervisor

Dr. Jochen Trumpf

A thesis submitted in part fulfillment for the degree of Honours in Bachelor of Engineering

September, 2006

Declaration

This thesis is an account of research undertaken between January 2006 and September 2006 at the Research School of Information Sciences and Engineering, The Australian National University, Canberra, Australia.

Except where acknowledged in the customary manner, the material presented in this thesis is, to the best of my knowledge, original and has not been submitted in whole or part for a degree in any university.

Priscilla Fong Line Kan John

September, 2006

Acknowledgments

I wish to thank the following people for having helped me in one way or another during the course of my honours thesis:

- First of all, my supervisor, Dr. Jochen Trumpf, for having been the "Dream Supervisor" anyone could wish for. Jochen made sure Desmond and I had regular meetings with him and I have enjoyed the insightful discussions during which humour was *au rendez-vous*. Jochen is also very approachable and friendly. He has always been very encouraging in his approach and that meant I set out to do things I had no idea how to do, but believing I could do them - I survived somehow :). His constructive criticisms and positive encouragements make me feel I have learnt a lot during this project.

- Desmond, for being a great technical advisor, and of course an awesome friend. It was a pleasure to collaborate with him during this project and tap into his large repository of Computer Graphics knowledge. I know he wished to be doing my project himself because it's "fun", like he said, and his enthusiasm for working with 3D graphics was contagious and soothed my frustration at various stages when I did not know what to do.

- Dave Ferrari, for helping enormously in teaching me how to use the 3D laser scanner and PolyWorks. Dave's been a great friend with whom to have discussions about aspects of the project since then too, despite him moving to Perth at some stage.

- Andreas, who shared an office at RSISE with me. Andi has been an awesome friend. He has been there to help me with coding and with lateX. It has also been a fruitful to discuss aspects of my work with him. He has provided me with great moral support in difficult times too.

- Pascal Vuylsteker, from CSIT, for helping to arrange access to Maya for me at the CNMA lab.

- Peter and Silas, from the IT Support team at RSISE, for being always ready to help with IT stuff and for being friendly and entertaining.

- Caroline, for her friendship and support. Caroline has been like a big sister looking after me and making sure I eat dinner and have coffee in the morning. Thanks a lot for all the morning calls for coffee!

- Rasa, for referring me to C++ and LateX guides. Rasa has been a great friend too, bringing me cake when I am stressed.

- Lee and Lucy, for their friendship and moral support. They also gave me food and made sure I don't go insane. Lee also helped with computer and software related issues.

- My parents and family for their love and support, even with an ocean between us. Mum has nurtured my love for Science since my early years, and Dad has always said that nothing is impossible in life - Reach for the stars, and if you miss you might just catch the moon.

- Chris, for his love, and for believing in me.

Abstract

In this thesis, we present the creation process of a virtual 3D hand model with sufficient accuracy and flexibility for the tracking of a human hand. The model consists of an underlying skeleton and a skin surface. The skeleton is implemented as a hierarchical joint structure with 17 joints and 26 degrees of freedom. We define an extra imaginary joint at the palm to model the movement of the CMC joint of the thumb. The skin is rendered from a point cloud joint by a triangular mesh. Our model has 46,140 points and 92,063 triangles. A set of constraints is also defined on motion of the hand at its joints. This helps in reducing the search space during tracking as poses disallowed by those constraints do not have to be taken into account. We used static and kinematic constraints on the hand and present a method for a future implementation of dynamic constraints. Detailed investigation of one kinematic constraint ($\theta_{DIP} = 2/3\,\theta_{PIP}$) defining a relationship between the PIP and the DIP joints of a finger was carried out. When used, this constraint reduces the number of degrees of freedom of the hand from 26 to 22. We confirm the validity of the constraint for natural hand motion and show that it may help tracking in difficult motions like a grip where different finger joints move at once. It also has potential to help track faster motion, possibly if used with other constraints or refined. We tested our model in the tracking framework developed by Mr. Desmond Chik, and compared its performance to a model from ETH Zurich[1] which Desmond initially used to develop his algorithms. The tracking results show that using a hand model that is better fitted to the hand being tracked does not make significant improvement on the tracking although it could reduce the amount of work done by the tracker. This indicates that the tracking framework is robust enough to be used to track different hand shapes as long as the key dimensions of the hand to be tracked are fitted to the model. Good tracking results of the thumb movement validate the inclusion of the extra imaginary joint for the thumb CMC to help reproduce natural motion of the thumb.

[1]Swiss Federal Institute

Contents

Chapter 1

Introduction

What is essential is invisible to the eye... The Little Prince

1.1 Context

As computers in one form or another pervade the modern world, it has become increasingly important to find new and better ways for humans to interact with them. The field of Human Computer Interaction (HCI) studies how humans design, implement and interact with computer systems, and the ways those systems affect humans at the individual and societal level. The aim is to have user interfaces that are efficient and easy to use, providing adequate support for user tasks, better access to information and more powerful forms of communication (Mischitz 2001). The importance of HCI should not be underestimated. The erroneous downing of an Iranian passenger plane, the IR655, in 1988, was partly attributed to the complicated user interface of the Aegis tracking system on the guided missile cruiser, USS Vincennes (Wikipedia, Iran Air Flight 655 2006). User interfaces is one big area within the vast HCI field. User interfaces involve input and output (I/O) devices and interaction techniques.

1.2 Motivation

User input methods to a computer have traditionally involved using the keyboard for text-based input and the mouse for Graphical User Interface (GUI). The standard way of user interaction with the computer is by typing, pointing and clicking. The advent of the GUI improved on command-line style input, its predecessor, in that a GUI provided the user with more direct control over their actions. The GUI has brought about windows, icons, menus and pointers (the WIMP paradigm) as interaction techniques (Turk & Robertson 2000). However, GUI-based interaction is becoming more and more limiting as our use of computers evolve. Computer systems now pervade our world in a variety of forms: small mobile devices, desktop applications, and embedded household or automo-

bile systems. The interaction needs for the various devices vary. Moreover, as increasing computing power improves graphics support capabilities, the trend is towards 3D virtual environments in HCI. This brings about the need for perceptual user interfaces (PUIs), which can meet the requirements of the diverse range of devices while providing better means to interact with the computer.
PUIs aim to provide interaction techniques that are more natural by combining an understanding of human capabilities (communication, motor, cognitive and perceptual), computer I/O devices and machine perception and reasoning. Devices and sensors are to be as transparent and as passive as possible. Machines should be able to perceive relevant human communication and produce easily understood output. To obtain this sort of interaction, integration of different technologies is required: speech/sound recognition and generation, computer vision and object tracking, graphical animation and visualisation, language understanding, touch-based sensing and feedback (haptics), machine learning, user modeling, and dialogue management (Turk & Robertson 2000).
This thesis focuses on the area of computer vision and object tracking. The use of hand gestures as a user input method is a promising PUI application. This has motivated research in visual analysis of highly articulated hand movement. To use human hands as a natural interface requires a means of capturing the hand motion. One option that has been explored is the use of glove-based devices, with attached sensors to measure joint angles and spatial positions of the hand. However, this method proves to be expensive and cumbersome. A possible more transparent alternative is the use of a non-contact vision-based technique that only needs affordable camera settings (Ying & Wu 1999). The system should recognise natural hand gestures, e.g. those used in sign languages, by tracking articulated hand motion in real time. This remains an open problem due to high degrees of freedom of the hand, self-occlusion, variable views and lighting (Zhou 2005).

1.3 Our Approach

We want to track an articulate structure in real-time. The articulate structure in question here is a human hand. One approach to tracking in real-time of an articulate structure involves using a computer model of the structure to reproduce the real-time movements. The model is part of a tracking system that comprises two video cameras to capture real-time hand motions, tracking algorithms that give the computer the ability to ”perceive” what the movements are, and the hand gestures of the human subject (the input method to the system).
There are various ways of building a hand model. We chose to use data from the real world to construct a human hand model to be used for tracking of hand movements in real-time.

1.4 Thesis Overview and Contribution

This project involves creating a model of a human hand with sufficient accuracy and flexibility for the tracking of a human hand. The hand consists of an underlying skeleton and a skin surface. The skeleton is implemented as a hierarchical joint structure. The skin is rendered from a point cloud joint by a triangular mesh. A set of constraints is also defined on motion of the hand at its joints. This helps in reducing the search space during tracking as poses disallowed by those constraints do not have to be taken into account. We used static and kinematic constraints on the hand and present a method for a future implementation of dynamic constraints. Detailed investigation of one kinematic constraint ($\theta_{DIP} = 2/3\theta_{PIP}$) defining a relationship between the PIP and the DIP joints of a finger was carried out. When used, this constraint reduces the number of degrees of freedom of the hand from 26 to 22. Dynamics of hand motion was also studied to produce a dynamic hand model. A dynamic hand model would have added constraints that reflect the action of muscles and tendons at the joints. This would hopefully improve on results obtained with a kinematic model by disallowing physically forbidden movements of the hand currently permitted with a kinematic model. It is expected that a better parameterized hand model can improve tracking efficiency by penalizing unlikely movements.

This honours work is relevant to NICTA's Advanced Nonlinear Gradient Methods - (ANGIE) - research project. The hand model was tested and used with tracking algorithms developed by a PhD student, Mr. Desmond Chik, who has agreed to be a second supervisor for the honours project. His algorithms have initially been tested using a kinematic hand model developed at the ETH Zurich[1]. Tracking results obtained from using the two different hand models were compared.

We confirm the validity of the PIP-DIP constraint for natural hand motion and show that it may help tracking in difficult motions like a grip where different finger joints move at once, and has potential to help track faster motion possibly if used with other constraints or refined. Comparing tracking results obtained with the ETH hand model and those obtained with our model shows that using a hand model that is better fitted to the hand being tracked does not make significant improvement on the tracking although it could reduce the amount of work done by the tracker. This means the tracking framework is robust enough to be used to track different hand shapes as long as the key dimensions of the hand to be tracked are fitted to the model. Good tracking results of the thumb movement validated the inclusion of the extra imaginary joint for the thumb CMC to help reproduce natural motion of the thumb.

[1]Swiss Federal Institute

Chapter 2

Literature Review

This chapter gives an overview of the body of work on modeling and tracking of a human hand. The two main areas where hand modeling has been investigated are computer graphics and biomechanics. More recently, hand modeling has become of relevance to the field of HCI. The application areas within HCI are hand animation (graphical and visualisation component of HCI) and hand motion analysis (computer vision and object tracking component of HCI). For graphics and visualisation, the main aim of hand modeling is to obtain realistic rendering. This is also desirable for the tracking part of HCI, as we want to match the computer model to the real hand using some methods. In the biomechanics community, the main aim is to understand the anatomical details of hand articulation for disease treatment, reconstructive surgery and the design of prosthetic joints (Hollister *et al.* 1992).

This thesis is concerned with the creation of a virtual three-dimensional hand model for the purpose of tracking a real human hand in real-time using stereo images. Literature was surveyed from the computer graphics, HCI and biomechanics communities. It is believed that a hand model with a high level of details and capable of deformation that is as "natural" as possible, will give higher accuracy in the reproduction of actual hand shapes. Hence, such a model would provide better tracking results due to its ability to produce subtle effects (e.g. the deformation of the palm, the interdepedence of finger joint angles) (Bray *et al.* 2004). However, there is a trade-off to be taken into consideration between the level of complexity of the model and speed of tracking.

2.1 Tracking of an articulated structure

The tracking of an object by a computer system refers to the ability of the system to recognise the movements of that object accurately, with sufficient information to reproduce the object's state at the different instants of its motion. Tracking is not a simple problem and different approaches have been tried. The range of tracking devices includes mechanical, optical, electromagnetic and acoustic devices (Baratoff & Blanksteen 1993).

For practical use in HCI, a tracking device needs to have high accuracy and resolution, very short lag time and high update rate as well as being easy to use, non-cumbersome and inexpensive (Baratoff & Blanksteen 1993). When the object in question is an articulated structure, tracking is complicated by the many degrees of freedom of the object. Most of the work done in the area of articulated structure tracking relates to human body tracking. Aggarwal *et al.* (Aggarwal & Cai 1997) provides a survey of early results in the field. A hand is a special case of an articulated structure with a relatively large number of degrees of freedom. Capturing hand gestures involves finding global hand movement as well as local finger motion (Lin & Huang 2001).

General solutions to tracking of a hand fall into two main categories: tracking using mechanical devices and visual tracking (i.e. using optical devices and computer vision techniques) (Guan & Chua &Ho 1999). Electromagnetic and acoustic devices lack robustness to be used for that purpose. Mechanical devices are accurate and fast enough but are cumbersome and tend to be expensive. We want sensing devices to be as passive and transparent as possible. In this respect, vision-based techniques offer a promising solution since they are non-contact, use affordable camera settings (passive sensing) and capture natural hand motion (Guan & Chua &Ho 1999), (Ying & Wu 1999).

2.1.1 Visual tracking: Appearance-based versus Model-based methods

Visual tracking methods can be classified further into appearance-based methods and model-based methods. Appearance-based methods employ matching techniques to map the image feature space onto the hand configuration space, emphasizing the analysis of hand shapes in images (Zhou 2005), (Duffy 1999), (Lin & Huang 2001). The main disadvantage of the appearance-based methods is that local hand motion is harder to estimate due to lack of precise spatial information (Lin & Huang 2001), (Guan & Chua &Ho 1999). Model-based methods estimate hand motion parameters by fitting a 3D hand model to observation images from a single calibrated camera or multiple cameras (Ying & Wu 1999). The use of a 3D model helps enhance tracking performance by alleviating the problem of depth ambiguity because the structure of the hand is inherent in the model (Ying & Wu 1999). The main challenge in visual tracking of the hand is the accurate recovery of hand posture from real world images due to the high number of degrees of freedom of the hand and noise in the observation.

To date, the arsenal of tracking approaches that can accurately track a human hand in a fast and reliable manner is still very small, and much work remains to be done (Bray

et al. 2004). We take a visual model-based approach to tracking in the work done as part of this thesis. We use Stochastic Gradient Descent (SMD) as a tracking method. The tracking framework will be further described in the next chapter. Other methods that have been investigated are tracking using an annealed particle filter (Deutscher & Blake & Reid 2000), simple gradient descent (Regh & Kanade 1995), (Sminchiesescu & Triggs 2002), second order gradient techniques (Press *et al.* 1988), (Levenberg 1944), (Marquardt 1963), the Powell method (Press *et al.* 1988), non-parametric belief propagation (Sudderth *et al.* 2004), among others. Some of these techniques are compared and evaluated by Bray *et al.* (Bray *et al.* 2004).

2.2 Aspects of Human Hand Modeling

A 3D hand model of some sort is required in using model-based methods for visual tracking of a hand. A hand can be modeled in several aspects: shape, kinematics, dynamics. Different models are suitable for different HCI applications. The two main application areas are hand animation (graphical animation and visualisation component of HCI), and hand motion analysis (computer vision and object tracking component of HCI). Hand motion analysis uses an "analysis by synthesis approach" method: hand motion is reproduced with a model in order to analyse it. Hand motion analysis is the application area that is looked into as part of the work done for this thesis. In animation applications, the shape aspect has the greatest importance and we want it to be as fine as possible. In motion analysis, all aspects matter depending on the depth of investigation we want (Ying & Wu 1999).

2.3 Hand Shape Modeling

The types of hand shape models can be grouped into geometrical, physical and statistical models (Ying & Wu 1999).

2.3.1 Geometrical Hand models

Geometrical hand models attempt to model the geometry of the surface of the hand. Splines, parameterized geometric shapes (cylinders, quadrics, cones) and free-form models (defined on a set of 3D points) have been the main methods used for building a geometric surface to model hand shape (Ying & Wu 1999). Parameterized geometric shape models

can achieve equally good surface approximation to spline models while being less complex (Ying & Wu 1999). In free-form hand models, hand shape is approximated in a computationally efficient manner by polygonal meshes that are formed on the set of 3D points. Cyber Scanner, MRI techniques or other space digitizers may be used to get the range data directly. The model we developed is a geometric free-form model. A laser space digitizer was used to obtain the 3D points directly.

2.3.2 Physical Hand models

Physical hand shape models try to reproduce the deformation of hand shape under the action of internal and/or external forces. Newtonian dynamics is employed to define the motion of the model (Ying & Wu 1999). The idea is that using physical modeling makes it possible to obtain realistic motion. Examples are models built using the Finite Element Method (FEM) (Tsap & Goldgof & Sarkar 1997) or rigid body dynamics (Durikovic & Numata 2004). Physical models tend to be complex to describe, especially in the case where we want to model the action of muscles and tendons. There is no simple method to describe the very complex muscle and tendon network that controls the articulation of the fingers of the hand. This is an area under current study and a promising attempt is being made by (Miller *et al.* 2004).

2.3.3 Statistical Hand models

A statistical model learns the deformation of hand shape from a set of training examples that can be 2D images or range images. Principal Component Analysis (PCA) is used to find mean shape and modes of variation. To obtain a hand shape, a linear combination of some significant modes of variation is added to the mean shape. An example is the point distribution model (PCM) (Ying & Wu 1999), (Heap & Hogg 1996). Heap and Hogg (Heap & Hogg 1996) report that the major problem with their model is its inability to cope with occlusion. Zhou (Zhou 2005) also takes a statistical approach in the modeling a a human hand and proposes a method he calls *eigen dynamics analysis* (EDA). The system learns the dynamics of natural hand motion as a *"high order stochastic linear dynamic system consisting of five lower-order subsystems"*.

2.4 Hand Skeleton and Kinematics

Kinematics refers to the study of motion that treats the subject without regard to the forces that cause it. Kinematics is concerned with the position of an object, its velocity and acceleration (and all higher derivatives with respect to time or any other variable). A manipulator can be defined as a set of bodies (links) connected in a chain by joints (Craig 2005). The skeleton of the hand can be treated as a complex manipulator consisting of a hierarchy of kinematical chains (the fingers). The skeleton thus defines the underlying kinematics of the hand and finger motion by constraining movement at the joints. There has been numerous attempts to model the kinematics of the hand by defining constraint sets on the angles of rotation of the joints (McDonald *et al.* 2001), (Lee & Tosiyasu 1995), (Rijpkema & Girard 1991), (Lin & Huang 2001).

An anatomical diagram of the skeletal structure of a human hand is shown in figure 2.1. A basic understanding of the hand skeletal structure is important in order to abstract a kinematic model from it. A more detailed description is contained in (Landsmeer 1955) and (Landsmeer 1976). We provide a short description in this section based on the one found in (McDonald *et al.* 2001). The carpals at the base of the hand are mainly involved with the flexion and abduction of the wrist. Most of them can be ignored when abstracting a kinematic model because they lie inside the palm and their contribution to the overall deformation of hand shape is negligible. However, we take into account the trapezium which articulates with the base of the thumb as its shape has considerable impact on the movement of the thumb.

Four long bones, the metacarpals, make up the main body of the palm. These articulate with the carpals at the carpometacarpal (CMC) joints and form the base of the fingers. An additional metacarpal bone, more mobile than the other four, forms the base of the thumb, and articulates with the trapezium.

Each finger, except the thumb, has three additional bones; the proximal, middle and distal phalanges. The metacarpal-phalangeal (MCP) joint is the joint between the metacarpal and proximal phalanges. The proximal interphalangeal (PIP) joint is between the proximal and middle phalanges, and the distal interphalangeal (DIP) joint is between the middle and distal phalanges. The interphalangeal joints are hinge joints (allowing flexion-extension). The MCP joint allows flexion-extension as well as abduction-adduction.

The thumb is the finger that gives the hand most of its expressiveness and is also the most complex of the digits. Its metacarpal bone articulates with the trapezium at the CMC joint. The articular surfaces of the thumb CMC joint are saddle shaped on both

sides. The articular capsule that joins the bones at this joint are loose but strong, which allows for a small amount of passive rotary movement in the thumb, enabling opposition (Netter 1987) *in* (McDonald *et al.* 2001). The thumb is also capable of flexion-extension and abduction-adduction at this same joint. The thumb also has an MCP joint connecting the metacarpal to the proximal phalanx, and an interphalangeal (IP) joint connecting its proximal and distal phalanges. The notable difference with the thumb is that its MCP joint is a hinge joint, unlike the MCP joints of the other fingers, and it has no middle phalanx.

Our kinematic model of the hand is provided in the next chapter, based on empirical and anthropomorphic data. We use descriptions from (McDonald *et al.* 2001) and (Rijpkema & Girard 1991) with some modifications.

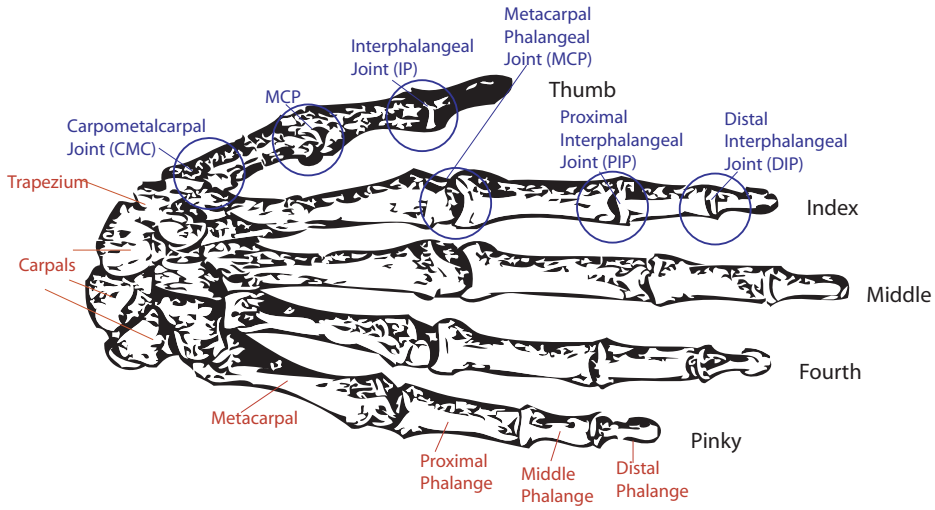


Figure 2.1: Hand skeleton and joints (WCSN 1997), (McDonald *et al.* 2001)

2.5 Hand Dynamics

On top of the skeleton, the hand has a complex network of muscles and tendons that are used to control its movements. Forces in those muscles and tendons are the cause of movement at the joints. The complexity of these movements have been extensively studied in the Biomechanics community. Landsmeer (Landsmeer 1955) studied the anatomy and spatial relationships between tendons and muscles and their associated joints in the hand. He proposed various models of tendon-joint displacement relationships. However,

there is a lack of quantitative description in these studies, in addition to the information presented being restricted to two-dimensions. An *et al.* (An *et al.* 1979) made up for that by conducting experiments to build a normative human hand model in three-dimensions for biomechanical analysis. This model gives force and moment potential under neutral pose. Determining exact muscle or tendon forces for dynamic modeling is extremely difficult. Firstly, there is the difficulty of measuring the exact forces in the different tendons and muscles as various forces come into action to produce a particular movement. This can lead to an indeterminate problem in the three-dimensional force analysis of the musculoskeltal system under analysis (the hand). Various methods have been attempted to solve such a system, but models derived would only be valid for certain hand configurations. Secondly, the exact manner of force transmission between muscles and tendinous bands has not been determined (An *et al.* 1985). However, we can find bounds on those forces, which would translate into dynamic constraints on the movement of the joints.

It is worth mentioning here that attempts are currently being made to model dynamics by using prediction methods. These attempts try to capture complex hand motion and recognise continuous hand gestures. Methods used include (Ying & Wu 1999):

1. Kalman filtering and Extended Kalman filtering (EKF)
   This is based on small motion assumption, which often fails to hold in hand motion.
2. Finite state machine
   This can be used for small hand gestures but proves insufficient for complex gestures.
3. Hidden Markov Model (HMM) and its variants
4. Rule-based approach
5. Bayesian networks
6. Neural networks

We will not consider prediction methods in this thesis due to its restricted timeline.

Chapter 3

Background

This chapter presents the background concepts and ideas used for developing our virtual hand model. The hand model is to be used for tracking of a hand in real-time using a model-based approach.

3.1 The tracking framework

Figure 3.1 depicts the tracking framework within which the hand model we developed is to be used. The tracking framework was built by Mr. Desmond Chik as part of his phD project. A pair of calibrated SONY XCD-X710CR firewire video cameras are used to capture images (at a resolution of 640x480 pixels) of a real human hand that is moving in their vision fields. These images serve as input to the tracking system. The tracking system compares those images to data from the internal computer model of the hand and try to match the model to the input images to estimate a position and configuration for the real hand using its tracking algorithms (SMD in our case). The output from the system is an estimated position and configuration for the real hand. The aim is to minimise the error between this estimation and the actual position and configuration of the real hand. At the moment, offline tracking is used with video sequences captured at a rate of 30 frames per second. Eventually, the aim is to achieve tracking in real-time.

3.1.1 Stochastic Meta-Descent (SMD)

The tracker we use is based on an optimization method called stochastic meta-descent (SMD), similar to the one proposed by (Bray *et al.* 2004) and described in details in their work. SMD has been developed to deal with high-dimensional state spaces like the state space of the hand. It is basically a gradient-descent method of optimisation that adapts its step-sizes over time and between dimensions to minimise a cost function. The cost function used by Desmond has two components: a silhouette component and a brightness consistency component. The overall cost function is evaluated by first taking N sample

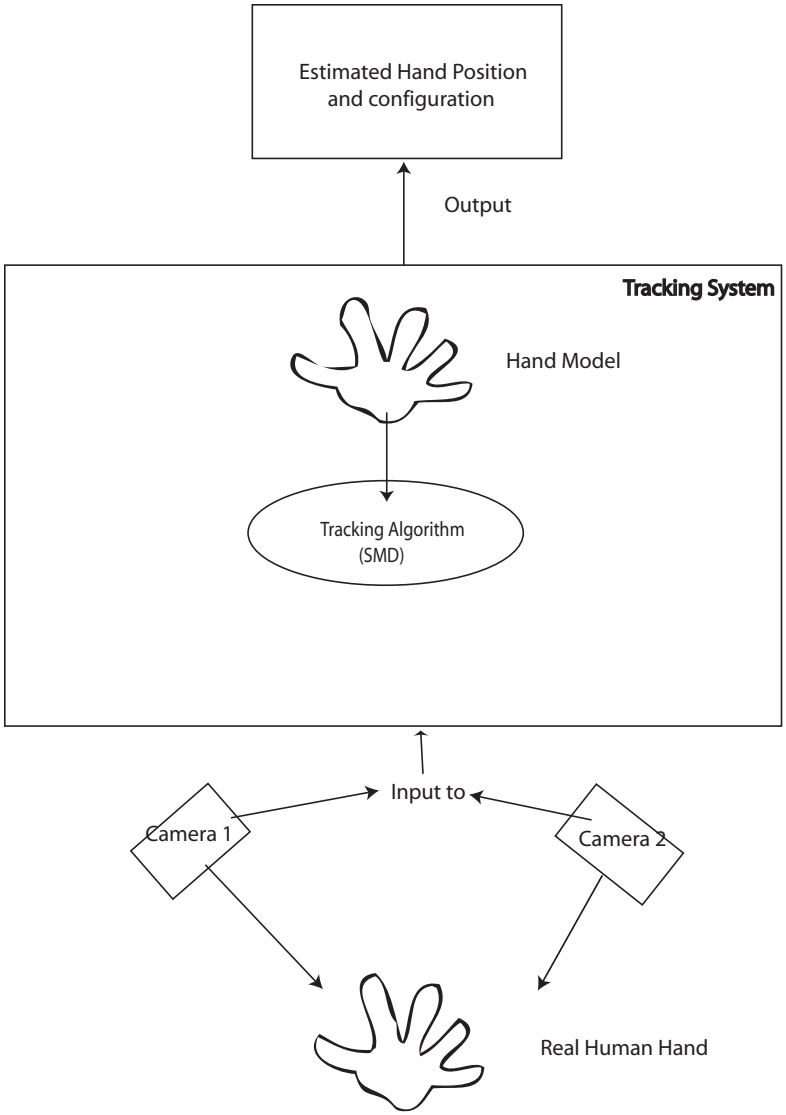


Figure 3.1: Tracking framework

points from the hand model and projecting them onto the two camera image planes. The individual sample points are then used in the computation of the two cost function components. Desmond presents his method in a paper that has been submitted for publication at the time of writing, which makes it inappropriate to discuss it further here.

The tracking algorithm uses a function F defined on the state of the hand $\boldsymbol{p}$ for relating 3D coordinates of the model to 2d coordiates of the cameras. $\boldsymbol{p}$ is given in terms of the translation and rotation of the palm and the joint angles of the finger joints. F, for a given $\boldsymbol{p}$ and intrinsic model parameters M, maps a point $\hat{x}_m$ given in hand model coordinates to predicted position $\hat{x}_c$ in camera coordinates, assuming a pseudo-orthographic position:

$$\hat{x}_c = F(\hat{x}_m, p, M) \tag{3.1}$$

3.2 Kinematic Modeling of the Hand

The kinematics of the hand is modeled by defining a skeleton to be used to control movements at all the hand joints. We are only concerned with forward kinematics when modeling the hand. We are interested using our model to specify joint angles to move the fingers at their joints to achieve a certain pose. In the hand modeling stage, we are not concerned with inverse kinematics where we are given a certain pose in world coordinates and have to find the joint angles that can achieve that. We abstract the skeleton of the hand as a stick figure. This reduces the dimension of a finger bone to its length (the link length). Each finger makes up a kinematic chain linked at the joints. The five finger kinematic chains are grouped by having a common reference frame at the palm. Hence, the skeleton of the hand can be modeled as a hierarchical model of 16 joints as shown in figure 3.2.

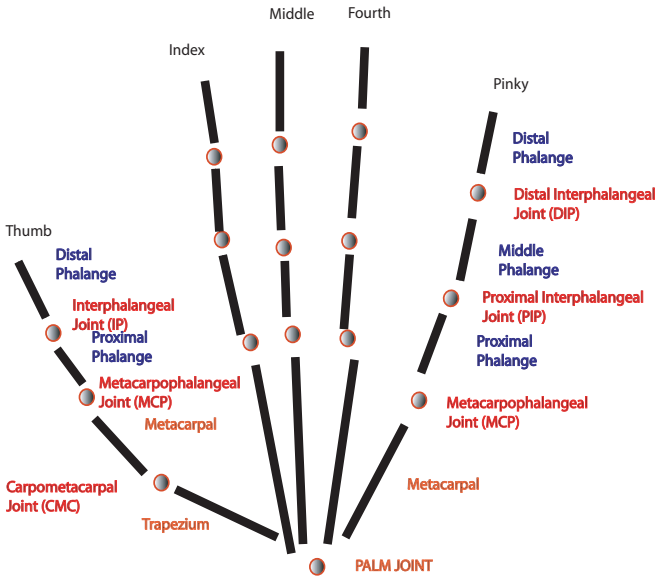


Figure 3.2: Abstracted Hand Skeleton

We based our modeling on the descriptions found in (McDonald *et al.* 2001). Our basic kinematic model has 26 degrees of freedom (DOF). The palm has 6 DOF, 3 for rotation and 3 for translation. The four fingers each have four DOF: one at the DIP, one at the PIP, for flexion, 2 at the MCP, for flexion and abduction. The thumb has a special structure, as mentioned in the previous chapter. The CMC joint of the thumb moves along a saddle joint, producing complex motion. We chose to have an extra joint for the thumb in our

model (an imaginary joint) to approximate the rotation of the thumb at the CMC that brings it into opposition with the other four fingers. We defined one degree of freedom for flexion at the IP, MCP and CMC for flexion. Thus our model has 4 DOF for the thumb and a total of 17 joints. The skeleton for our 26 DOF kinematic model is shown in figure 3.3.

In other work, the opposition movement of the thumb is modeled by placing an additional degree of freedom at the MP joint of the thumb (Rijpkema & Girard 1991), (Regh & Kanade 1995), (Lin & Huang 2001). This gives five DOF for the thumb, resulting in 27 DOF for their models.

The number of degrees of freedom of the hand model determines the dimensionality of the state space that defines the hand. Hence we want to minimise this number. We can restrict the degrees of freedom of the hand or decrease their number by defining constraints on the joint angles, reducing the search space for tracking. Constraints we apply are described in section 4.7.

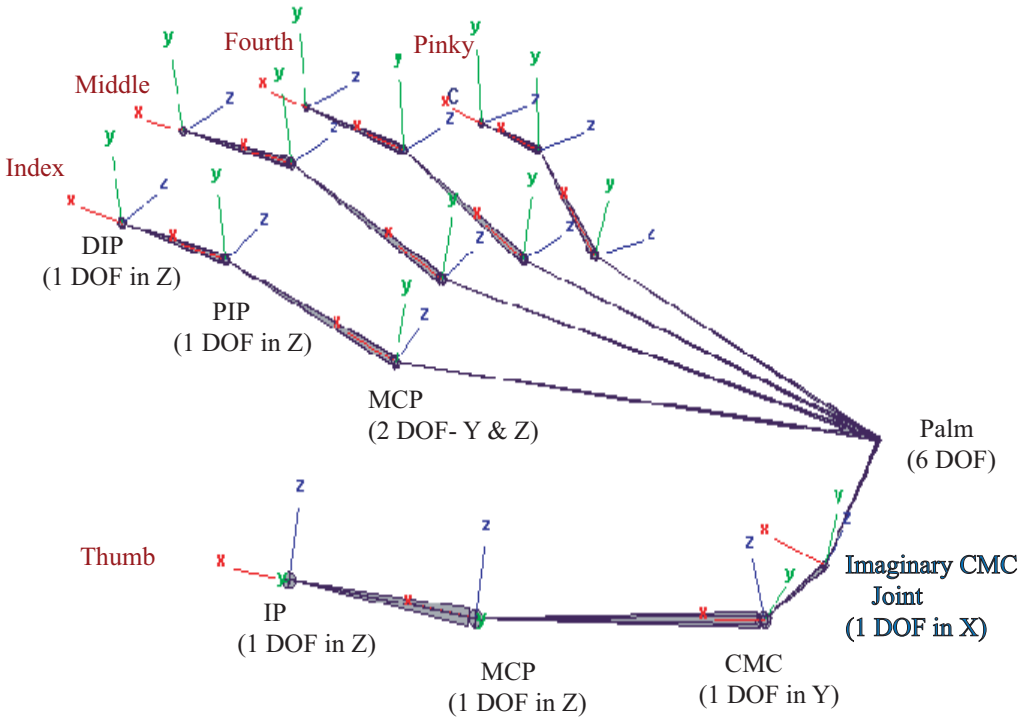


Figure 3.3: Hand Skeleton Model in Maya *(last phalanges of the fingers are not shown)*

3.3 Dynamic Modeling of the Hand

3.3.1 The complex musculotendinous structure of the finger

The network of muscles and tendons that controls the movement of the fingers is a complex system. An anatomical description is depicted in figure 3.4. To the best of our knowledge, the exact manner of force transmission within this network is not fully understood yet, and is still being researched. A novel approach to modeling the finger extensor mechanism is being taken by (Miller *et al.* 2004) whereby a computer-controlled system is used for actuating multiple muscles in cadaveric hands to measure finger forces and torques for given input tendon forces or excursion. The data thus collected is then to be used with model-based estimation techniques to determine the statistically best suited representation of the finger extensor mechanism.

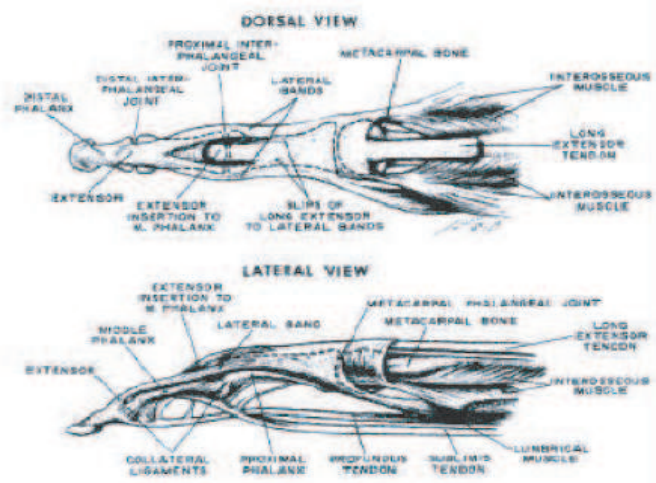


Figure 3.4: Musculotendinous structure of the finger from (An *et al.* 1985)

An adapted version of Winslow's tendinous rhombus, dating from the 17th century, is presented in (Valero-Cuevas & Zajac & Burgar 1998), shown in figure 3.5. The distribution of tension in the network branches depends on their geometric configuration which changes with finger posture (a floating net). This model improves on previously proposed models that assumed a fixed distribution of tension among its elements (Miller *et al.* 2004). The floating net model of the extensor mechanism has been employed in more recent studies of finger movement (Sancho-Bru *et al.* 2001), (Valero-Cuevas & Johanson & Towles 2003). However, there is a lack of a general-purpose model in the literature so far. Those proposed are not valid for all finger postures as a general model is very hard to define. There is also no mechanics-based description that can be easily simulated on a computer (Miller *et al.* 2004).

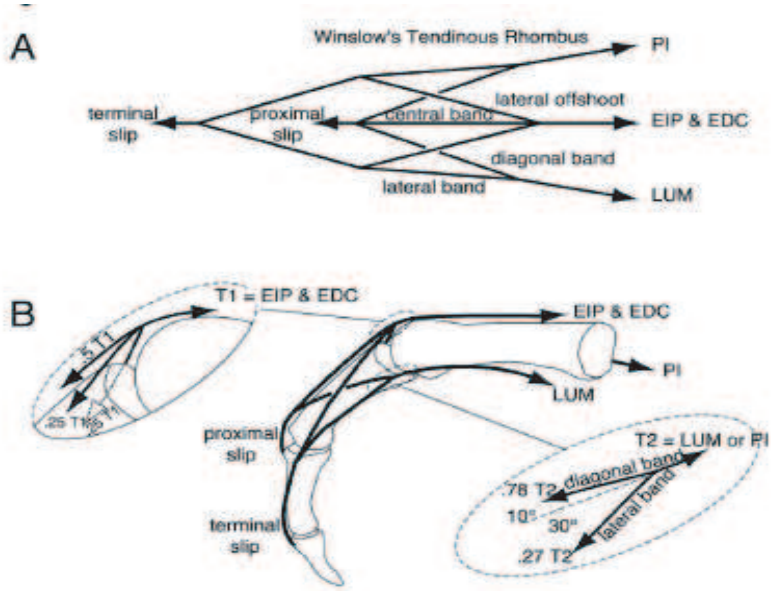


Figure 3.5: Extensor Mechanism Model (Valero-Cuevas & Zajac & Burgar 1998)

3.3.2 Relationship between muscle force and joint torque

We study a simple model from (Shadmehr & Wise 2005) consisting of one muscle acting at one joint to produce flexion-extension to illustrate the relationship relating muscle forces to joint torques. The joint in question can be any of the finger joints and the muscle could be one of the extensor or flexor muscles that line the finger. When the muscle contracts, it produces a force $\vec{F}$ (with magnitude f) that results in a torque T on the finger joint (figure 3.6). This torque causes the joint angle to extend by an amount Δq in a given time interval Δt (we use discrete time notation here to express what happens on a micro-level). In the convention chosen, q increases when the joint flexes, hence flexion gives positive angular velocity in that case and positive torque. Contraction of the muscle results in a shortening of its length by an amount $\Delta\lambda$. By the **principle of virtual work**, the amount of work that the muscle does in shortening its length is the same as the virtual work it performs in rotating the joint.

$$Work\ done\ in\ muscle = -f\Delta\lambda$$

$$Work\ done\ in\ rotating\ joint = \tau\Delta q$$

Hence we can equate the two equations above to give:

$$\begin{aligned} T\Delta q &= -f\Delta\lambda \\ T &= -\tfrac{\Delta\lambda}{\Delta q} f \\ T &= -\tfrac{d\lambda}{dq} f \end{aligned} \tag{3.2}$$

if we let the the length and angular changes approach zero.
The important idea expressed in equation 3.2 is that the torque a muscle produces on a joint depends on how its length changes with respect to the angle of the joint. In the given reference frame, an increase in the angle q results in an increase in the length of the muscle. This relationship depends on the choice of reference frame. To find $\frac{d\lambda}{dq}$, we consider the geometry of the finger around its joint (see figure 3.6). We can thus write:

$$\lambda = \sqrt{a^2 + b^2 - 2abcos(q)} \tag{3.3}$$

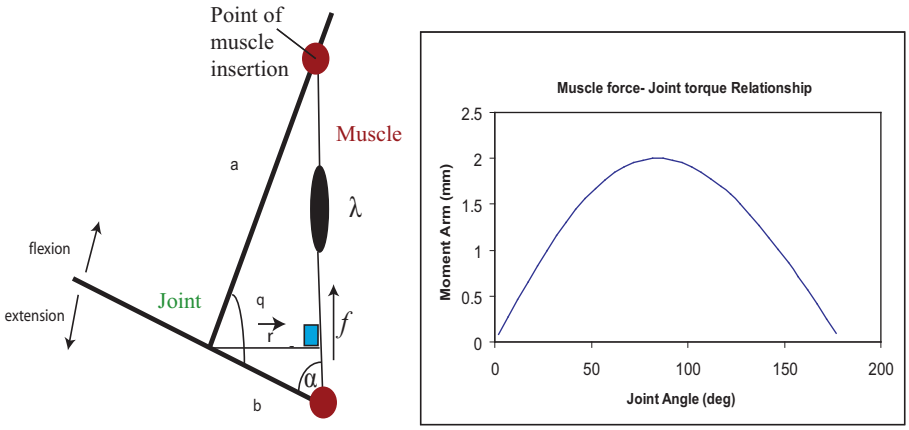
Differentiating the above with respect to q we get:

$$\frac{d\lambda}{dq} = \frac{absin(q)}{\sqrt{a^2 + b^2 - 2abcos(q)}} \tag{3.4}$$

Combining equations (3.2) and (3.4), we get:

$$T = -\frac{absin(q)}{\sqrt{a^2 + b^2 - 2abcos(q)}}\ f \tag{3.5}$$

We can use trigonometry to show that $\frac{d\lambda}{dq} = \vec{r}$, the moment arm of the muscle at that particular joint. The moment arm is the length of the perpendicular that connects the centre of rotation of the joint with the line of action of the muscle (figure 3.6). Hence, torque increases with an increase in force and/or an increase in moment arm. Figure 3.6(b) shows how moment arm varies with changes in joint angle q. Values used for a, b and f are hypothetical and for illustration purpose only, and are not based on anatomy. For the given parameter values, the muscle exerts its greatest torque on the joint when the angle is approximately 84^o. At extreme positions of flexion or extension, i.e. q very large or very small, the value of the moment arm becomes very small. We therefore deduce that at these extreme angles, the muscle force results in very little joint torque. The moment arm provides a measure of the effect of muscle tension at a joint.



(a) (b)

Figure 3.6: Schematic for a muscle acting on a single joint. **(a)**a,b: length of bone from joint to insertion point of muscle; q: supplementary joint angle; f: linear force applied by muscle; λ: length of muscle; $\vec{r}$: moment arm of joint-muscle system. **(b)**Moment arm as a function of joint angle for system in (a). *hypothetical values a=20mm, b=2 mm, f=1N.* Adapted from (Shadmehr & Wise 2005).

When the muscle (or tendon) in question spans more than one joint, the muscle force to joint torques relationship becomes more complicated. Numerical methods are then required to solve the system of equations. It should be noted that the tension along the muscle or tendon remains the same, even as it changes direction when crossing different joints. However, its moment arms at the different joints will differ, determining its relative effect at each joint.

3.3.3 Forward dynamics to find dynamics constraints

Forward dynamics is a physics-based modeling approach. It takes the forces and torques that cause the motion of an articulated object and uses mechanics equations to calculate the object's spatial parameters (position, velocity and acceleration). It is a technique that is increasingly used in computer animation to obtain realistic movements of characters that conform to the laws of physics. The calculated spatial parameters are used to update the pose of the characters at each time step. Common methods employed to perform forward dynamics calculations are the Lagrange equations or the Newton-Euler equations, where velocity and acceleration calculations are propagated from link to link in a step-wise fashion (Huang 1996). Huang (Huang 1996) gives a more detailed treatment

of the Newton-Euler method in describing his implementation which uses a forward and inverse dynamics in a closed-loop set-up. We adopt here a different approach, presented in (Albrecht & Haber & Seidel 2003), that is more intuitive to the understanding of the action of muscle forces at the joints. This approach can be used to find dynamics constraints for the motion at a particular joint, given that we know the maximum force that can act at that joint, and assuming constant moment arm. This approach does not use force propagation from link to link but treats the structure in question as a single object on which a muscle force is acting at a given joint. The complexity of force propagation along the links is delegated to the calculation of the moment of inertia of the structure, explained further below.

A muscle can exert tension in two ways: contraction due to nerve stimulus (active contraction) or contraction resulting from the muscles elastic behaviour (passive contraction). We denote the time-varying neural control signal (the actuation value of the muscle) by c(t). The highest tension, $\vec{F}_{max}$ during active contraction is produced when the muscle is at its resting length l_o. The tension-length relationship for passive and active muscle contractions is depicted in figure 3.7, which is known as a *Blix* curve. We approximate the two tension-length curves by the quadratic fit given in (Albrecht & Haber & Seidel 2003) to obtain a continuous function for the force exerted by a muscle.

Let $l \in [0.5l_o, 1.5l_o]$ be the current fibre length of a muscle with rest length l_o. The muscle contraction value is given by $c \in [0, 1]$. The contraction force $\vec{F}_c$ and stretch force $\vec{F}_s$ are given by:

$$\vec{F}_c(l) = \left[1 - 4\left(\frac{l}{l_o} - 1.1\right)^2\right]\vec{F}_{max} \tag{3.6}$$

$$\vec{F}_s(l) = \begin{cases} 2.77\left(\frac{l}{l_o} - 1\right)^2 \vec{F}_{max}, & l \geq l_o \\ 0, & l < l_o \end{cases} \tag{3.7}$$

The total force is hence

$$\vec{F}(l) = c\vec{F}_c(l) + \vec{F}_s(l) \tag{3.8}$$

where c is the neural control signal (or muscle contraction value).
We consider exactly one muscle acting at the last joint of a finger, i.e we consider the effect on a distal phalange. This is the simplest case because it simplifies the computation of moment of inertia of the body undergoing rotation.
Let $\vec{r}$ be the moment arm of the muscle at the distal joint. We note here that $\vec{r}$ is also

given by equation (3.5). The torque produced is

$$T = sgn\left(\left\langle \vec{a}, \vec{r} \times \vec{F} \right\rangle\right) . \left\| \vec{r} \times \vec{F} \right\| \tag{3.9}$$

where $\left\langle \vec{a}, \vec{r} \times \vec{F} \right\rangle$ denotes the dot product of the rotation axis and the vector-valued torque, the latter being the cross-product of the moment arm and the force.
We also know the relationship between torque, T, angular velocity ω, and moment of inertia J is given by

$$T = J\frac{d\omega}{dt} \tag{3.10}$$

The angular velocity ω is the first derivative of the rotation angle ϕ.

$$\omega = \frac{d\phi}{dt} \tag{3.11}$$

We discretise time and evaluate:

$$\begin{aligned} \Delta\omega &= \Delta t.J^{-1} \cdot T \\ \omega_{new} &= \omega_{old} + \Delta\omega \\ \Delta\phi &= \Delta t.\omega_{new} \\ l \leftarrow l - \Delta\phi. \left\|\vec{r}\right\| & \end{aligned} \tag{3.12}$$

We can use this approach to find a bound on $\Delta\phi$ for a given contraction value c and time step Δt and a maximum force $\vec{F}_{max}$. Δt is given by the frame rate of the video sequence that the tracking system is using.
To find J for the distal phalange, we approximate the segment of the finger above the distal joint to a solid cylinder of length l and mass m being rotated about an axis orthogonal to its length axis and passing through its end. J is thus given by

$$J = \frac{1}{3}ml^2 \tag{3.13}$$

The calculation of moment of inertia becomes more complicated when we want to rotate

a structure with more than one link. The value of J then depends on the position of all the links in the chain. This situation is illustrated in figure 3.8. The moment of inertia J_i of the rotated link is constant and can be computed using equation(3.13). However, the total moment of inertia for a rotation about joint J_{i-1} is composed of J_{i-1} and the moment of inertia of the end segment. The latter is not simply J_i in this situation because the rotation axis does not pass through the end of the rotated end segment as required for equation(3.13). To obtain the moment of inertia J_i^* of the end segment, we have to transform its position into the coordinate frame of J_{i-1} and sum up the squared distances of the transformed segment vertices to the rotation axis multiplied by the mass of the segment. This computation gets more complex with more segments in a chain.

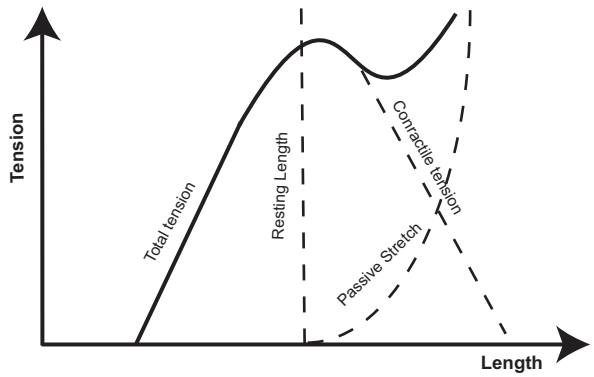


Figure 3.7: Blix diagram showing tension-length relationship for a muscle under active and passive contraction

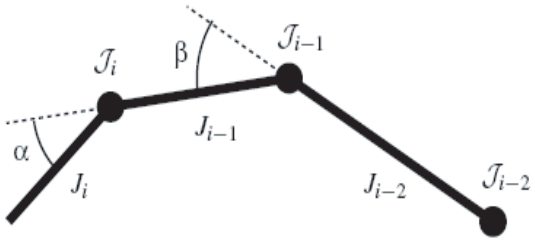


Figure 3.8: Moments of inertia for a chain of links

We present a "recipe" for a proposed method to implement dynamic constraints in our model in Appendix B.

3.4 Modeling the skin

A free-form geometric shape from a set of 3D points (a 3D point cloud) produces a model with a higher level of details. This gives greater accuracy (than using a model built

from basic primitives) in the reproduction of hand shape and therefore better supports tracking by exhibiting more subtle effects like the deformation of the palm. The 3D points we obtain from a laser scanner are joint together by triangles to form a polygonal mesh. This mesh surface is in a "static" pose (termed the binding pose in Maya), where the points have undergone no transformation to move their coordinates. We place a skeleton inside this model to drive the deformation of the skin by defining a relationship between the movement of the skeleton joints and the position of the points forming the skin surface. We use a technique commonly employed in computer graphics called linear blend skinning, or linear blending ("smooth skinning" in Maya), to define this relationship. This augments the static polygonal model with extra attributes: every vertex is given a set of bending weights that corresponds to the influence of the joints affecting them (one weight for each affecting joint). These weights are used to compute a transformed position for each of the vertices when the skeleton is moved.
The position of a vertex $\mathbf{v}_c$, influenced by n joints, at an arbitrary skeletal configuration c is calculated as:

$$\mathbf{v}_c = \sum_{i=1}^{n} \alpha_i M_{i,c} M_{i,b}^{-1} \mathbf{v}_b \tag{3.14}$$

α_i are the blending weights, $\mathbf{v}_b$ is the binding pose position of some vertex $\mathbf{v}$, $M_{i,c}$ is the global transformation matrix of the ith joint in configuration c and $M_{i,b}^{-1}$ is the inverse of the global transformation matrix of the ith joint in binding pose b. When $c = b$, we obtain the same location of $\mathbf{v}$ ($\mathbf{v}_c = \mathbf{v}_b$) in its binding pose (Bray *et al.* 2004). More details are given in (Lewis & Cordner & Fong 2000).

3.5 3D Scanning System

3.5.1 3D Scanners

A scanning system is used to capture range images - arrays of depth values for points on the object from a particular viewpoint - from which a point cloud is built. To generate an accurate model efficiently, it is useful to understand the physical principles by which a scanning system works. A 3D scanner works in an analogous fashion to a camera. It has a cone-like field of view and collects distance information about surfaces within its field of view - by analogy, a camera collects colour information within its field of view. We define a spherical coordinate system with the scanner at the origin and the vector coming out from the front of the scanner at $\phi = 0$ and $\theta = 0$. Each point captured by the scanner is associated with a set of (ϕ,θ,r). These spherical coordinates fully describe the

three-dimensional position of each captured point in a local coordinate system relative to the scanner.

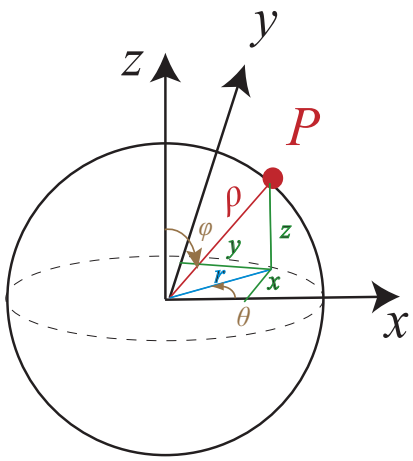


Figure 3.9: Spherical Coordinate System: *Scanner is at origin, point P is described by ϕ, θ and r*

A 3D scanner can be of contact or non-contact type - the former needs to be in physical contact with the object to be scanned, and the latter does not. Non-contact scanners can be further categorised into active and passive types - the former emits radiation and the latter does not. We are interested here in non-contact active 3D scanners; the most common of which are time-of-flight, triangulation and structured light systems. We used a laser triangulation system to obtain 3D scans for building our hand model and will therefore elaborate more on this particular type of scanner in this section. More details about the other types can be obtained from (Wikipedia, 3D Scanner 2006).

3.5.2 Principle of a Laser Triangulation System

A laser triangulation system consists of a laser projector that projects a pattern of light (a laser dot or stripe) onto the object to be scanned, and a sensor (a CCD camera) that senses the reflected light from the object (figure C.1, in Appendix C). This technique is called triangulation because the laser projector, the pattern on the object and the sensor form a triangle. The laser dot or line appears at different locations in the CCD camera field of view depending on the distance of the laser projector from the object's surface. The location of the laser dot or stripe projected onto the object provides us with distance information concerning the object. We consider a point P on the stripe of light projected by the laser onto the object (see figure C.1). We know the distance A between the laser and sensor, and the angle b formed by the laser beam. Angle c, formed by the reflected laser light can be determined by looking at the location of the laser dot corresponding to

point P in the camera's field of view. These three values (A,b,c) fully determine the shape and size of the triangle, and thus the location of the object (distance C) is known. This distance calculation is automatically done for all points on the laser stripe by software provided with the scanner, generating an array of depth values which can be converted to 3D point positions in the scanner coordinate system, using the calibrated position and orientation of the laser projector and sensor.
The main advantage of triangulation systems is their relatively high accuracy (order of 10^{-5}m) - accuracy of time-of-flight systems is in the order of 10^{-3}m. However, triangulation systems need to be used at close range (a few metres), which is fine for our purpose but would not be suitable for applications requiring long range like surveying. Other important aspects of laser triangulation scanning are:

- a clear view is required for both the source and sensor to see the surface points being scanned.
- surface reflectance properties affect the quality of data obtained - materials that are shiny, have low surface albedo or significant subsurface scattering give poor results.

Range scanners provide implicit surface normal information about the scanned object. We know the direction of the reflected laser ray from a 3D point on the object to the CCD sensor. This indicates that there are no occluding surfaces along that ray and thus tells us which side of the point form the exterior of the object. Range images are organised as 2D arrays so that an estimate of the surface normal at each point can be obtained by computing the vector cross product for vectors from each point to its immediate neighbours.

Chapter 4

Experimental Design

We present in this chapter the process used to obtain a virtual 3D hand model which is to be used for tracking purposes. The virtual model was constructed from data obtained from a real human hand with a 3D laser scanner (ModelMaker x70 from 3D Scanners) mounted on a coordinate measuring machine (CMM) (a Platinum FaroArm). The model built is a free-form geometric surface model consisting of a skeleton covered by a skin surface. We first outline the construction process of the skin surface using what is commonly termed the 3D model acquisition pipeline in computer graphics. We then explain how a skeleton was estimated for the model. We used linear blending to obtain realistic deformation of the skin surface when the skeleton is bent at its joints. Linear blending assigns a set of weights for each point on the skin surface that correspond to the effect of different joints acting on the point. A set of constraints was defined on movement at the different joints of the hand, with the aim of making hand motion as realistic as possible. Finally, we tested the performance of our hand model in different tracking scenarios.

In this work, when we talk of the top side of the hand, we mean the dorsal side (in an anatomical sense). Accordingly, the bottom side is the palmar side.

4.1 From Real to Virtual Hand: The 3D Acquisition Pipeline

We go through the 3D acquisition pipeline to obtain the skin surface for a 3D virtual hand model from real human hand data, a procedure that is commonly used to obtain models for computer graphics applications. A 3D model, in the computer graphics sense, is a "numerical description of an object that can be used to render images of the object from arbitrary viewpoints and under arbitrary lighting conditions" (Bernardini & Rushmeier 2002). The basic skin surface thus obtained is eventually augmented with the attachment of a skeleton and definition of constraints at the finger joints to obtain a model to be used for tracking.
The 3D acquisition pipeline used in our work is depicted in figure 4.1, adapted from

(Bernardini & Rushmeier 2002). We do not consider texture mapping for our model since we are only interested in the 3D shape and motion of a hand in the tracking framework we use. We use a range scanner to obtain a set of ranges images (arrays of depth values for points on the object from a particular viewpoint) from which to build a 3D point cloud. We then bring the images into a common reference frame (registration) before merging them to create a complete 3D model by obtaining a polygonal mesh that connects the 3D points. We postprocess the polygonal mesh to refine the model before it can be used. We discuss how we implemented the different stages of the pipeline in the subsequent sections of this chapter.

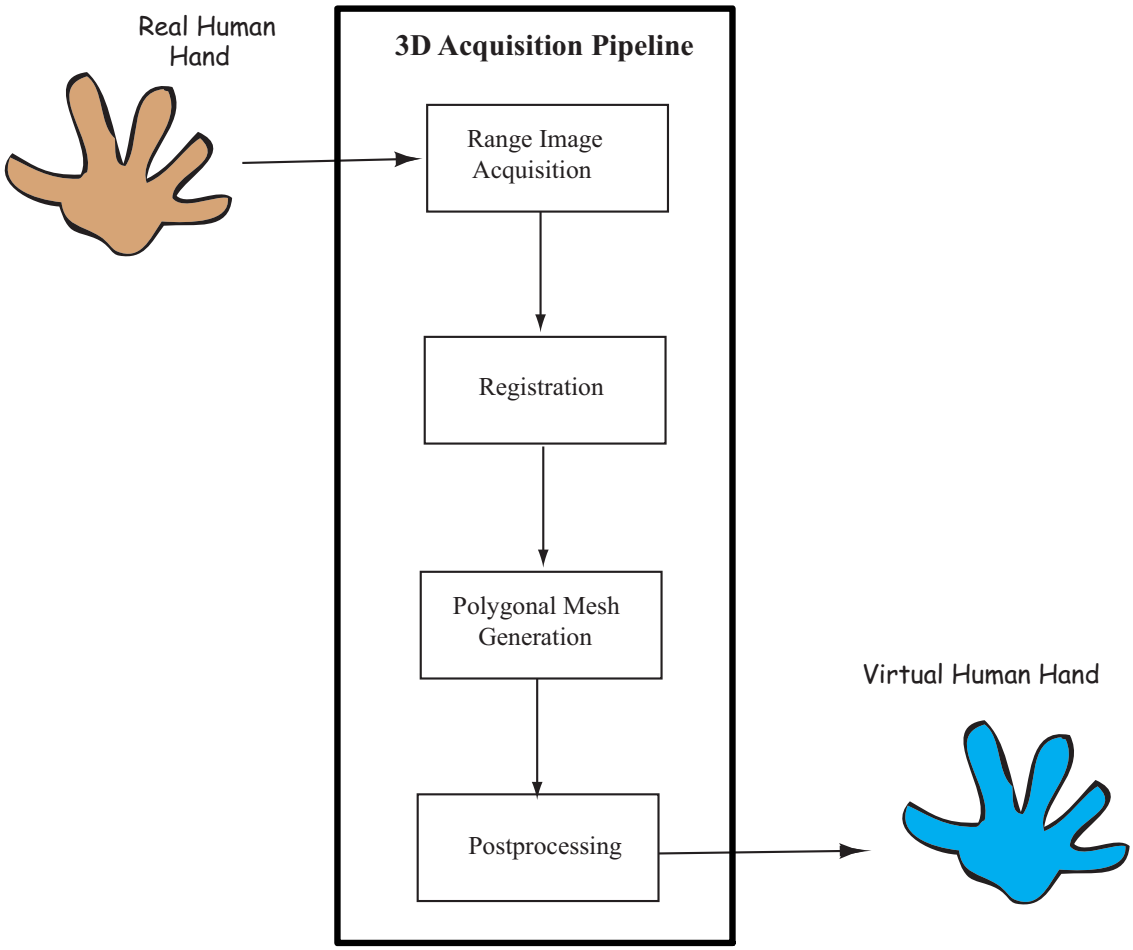


Figure 4.1: The 3D Acquisition Pipeline used to construct a 3D model from scanner data

4.2 Obtaining range images

4.2.1 Scanning Equipment

The chosen method of modeling the skin surface of our virtual hand model is by defining a free-form geometric shape from a set of 3D points (a 3D point cloud). The obtention of this point cloud constitutes the first stage of the virtual model creation. Range images from a 3D laser scanner are used to construct this point cloud. We used a laser triangulation set-up (see section 3.5) to perform the scanning. This consisted of a laser projector and its matching CCD sensor - ModelMaker x70 from 3D Scanners - mounted on an articulated robotic arm - platinum FAROArm from FARO Technologies - that acts as a coordinate measuring machine (CMM). The 3D scanning equipment we used is illustrated in figure D.1 in Appendix D. The CMM captures Cartesian coordinates of the laser system mounted on its end, which are used in determining the position of points on the object being scanned. The Platinum FAROArm has 7 axes of rotation and a scanning accuracy of ± 0.025 mm at close range (1.2m) (FARO Technologies 2006).

4.2.2 Reference Model

The object to be scanned needs to be very still during the scanning process because any slight movement will cause distortion in the point cloud obtained. It is quasi-impossible to get our test person (Desmond) to hold his hand absolutely still for the duration of scanning (we tried). Therefore we need a means to fix the physical shape of a human hand and use the created mold for scanning. We casted the bottom half of Desmond's hand in pottery plaster, and scanned the resulting mold. We then had Desmond place his actual hand into the cast and scanned the top part of his hand. This ensured his hand did not move during scanning. Pictures of the plaster mold and scanning process are included in Appendix D. A set of range images were thus obtained for each half of the hand. These were further processed along the 3D acquisition pipeline using the software suite Polyworks (v8) from InnovMetric.

4.3 Registration

4.3.1 The Registration Process

The next step is to align the range images from the scanner to get them to be in the same coordinate system, i.e. perform image registration. This was done using IMAlign, a Polyworks feature. We have a set of sab2 files (3D Scanners file format) which contain range image information in the form of unorganised point clouds. These images are imported into IMAlign and manually divided into subgroups defined by a viewing direction. This assigns surface normal information to the points - if surface normal information is available from the scanner, IMAlign can automatically delimit the subgroups. IMAlign then triangulates a surface mesh for each subset and interpolates a new 3D image. Two considerations to take into account when selecting and editing each subgroup are:

1. A subgroup should not contain data points that are behind other data points.
2. Overlap between the image subgroups is important for the proper working of the merging algorithm that is applied downstream in IMMerge to obtain a polygonal mesh.

IMAlign performs shape-based alignment so that images need to meet two criteria for the alignment methods to produce good results:

1. Each image must share some redundant information (overlap) with adjacent images.
2. Each image should contain some shape variation of the object.

IMAlign offers three mouse-driven tools for specifying an approximate alignment between selected images. Next, an iterative best fit algorithm is applied to refine the approximate alignment. The three approximate alignment techniques are:

- Manual alignment between two groups of images, one selected and one unselected. Rotations and translations are applied, using the mouse, to give the selected images an orientation similar to the one of the non-selected images.
- One-point pair alignment.
  Two corresponding points are picked, one from each of the two sets of images to be aligned. IMAlign tries to match the point pair to align the two sets of images.

- N-point pairs alignment.
  This is similar to the One-point pair but with N matching point pairs chosen by the user.

IMAlign's iterative best fit algorithm refines on the approximate alignment specified by the user. It tries to compute an optimal alignment by minimizing the 3D distances between surface overlaps in a set of 3D images. With each iteration, the algorithm multiplies each 3D image transformation matrix by an incremental transformation matrix that improves the image alignment with respect to the other 3D images. A linear least-squares technique is used to calculate this incremental matrix which results from averaging the best alignment parameters of each 3D image point. This can be formulated as follows: Given two sets of matching points $P = \{p_1, ..., p_n\}$, $Q = \{q_1, ..., q_n\}$, we want to compute a rotation matrix R and translation vector T such that the sum of squares of pairwise distances

$$e = \sum_{i=1}^{n} \|p_i - (Rq_i + T)\|^2 \tag{4.1}$$

is minimised (figure 4.2). This process is iterated until some stopping condition is satisfied. In IMAlign, two types of stopping condition can be specified: number of iterations and/or a convergence criterion. Convergence of the optimisation algorithm occurs when all incremental transformation matrices computed during one iteration are identity matrices - multiplication by an identity matrix has no effect on resulting alignment. Since perfect convergence is unlikely to occur, we can specify a convergence value that will stop the iteration process. This value is calculated for each incremental matrix and is defined as the sum of the squared differences between the identity matrix elements and the incremental matrix elements. It converges towards zero as the incremental matrices get close to identity form. We chose a convergence value of 0.0001 in our work.

It has been proved (Besl & McKay 1992) that the best-fit algorithm - also known as the Iterative Closest Point (ICP) algorithm in the literature - converges to a local minimum. It can achieve convergence in a few steps in good implementations. However, convergence to a global minimum is not guaranteed and is dependent on the initial configuration. Problems arise with images that do not have enough overlap or with object surfaces that do not have enough shape variation. A cylindrical surface would be an example of the latter. Two aligned partial scans of a cylindrical surface could slide relative to each other while the distance between corresponding points remain zero (Bernardini & Rushmeier 2002). IMAlign requires that at least one 3D image be "locked" during the

alignment process. The transformation matrix of a locked image is not modified during the optimisation process. The convergence process can be sped up by locking one central image that has a significant degree of overlap with other images.

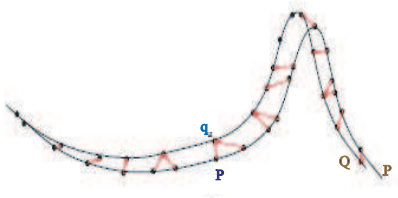


Figure 4.2: One step of Best Fit Alignment Algorithm. Point matches are defined based on shortest Euclidean distance. A least-squares method is used to transform scan P in order to minimise the length of the displacement vectors.

4.3.2 Alignment procedure and issues

We built our 3D virtual hand model by scanning a plaster cast of the bottom of a real hand and the real upper part of the hand. This produced data for the two halves of a human hand which need to be aligned. One issue we faced is that the two halves only overlap at the edges formed by the shape of the hand, and on the smooth flat surface of the plaster which is outside of the hand shape. The criteria for shape-based alignment are insufficient here as there is too little overlap in the scans of the two halves, and the smooth flat plaster surface does not have enough shape variation. A solution to the problem was to place round-headed pins into the plaster cast between the shape of the fingers and around the outline of the hand (see figure 4.3). This would provide additional overlapping surfaces in the scans of each half of the hand, which will help the alignment process. Because the pins have a shiny metal surface (high reflectivity), they had to be sprayed with white paint before scanning with the laser scanner.

The normals on the scans of the plaster cast were flipped to give the proper normal direction for the bottom surface of a real hand. The normals of the pin heads in the scan images were not flipped however as we want to use them for N-point pair alignment. N-point pair alignment was performed by specifying corresponding point pairs on five pin head surfaces and the fingertips. Then,a separate iterative best-fit alignment process was run for the images comprising each half of the hand. This was done by locking images of the other half when trying to align one of them. This gave reasonable alignment results

with a specified convergence value of 0.0001. Pictures from our alignment process are included in Appendix E.

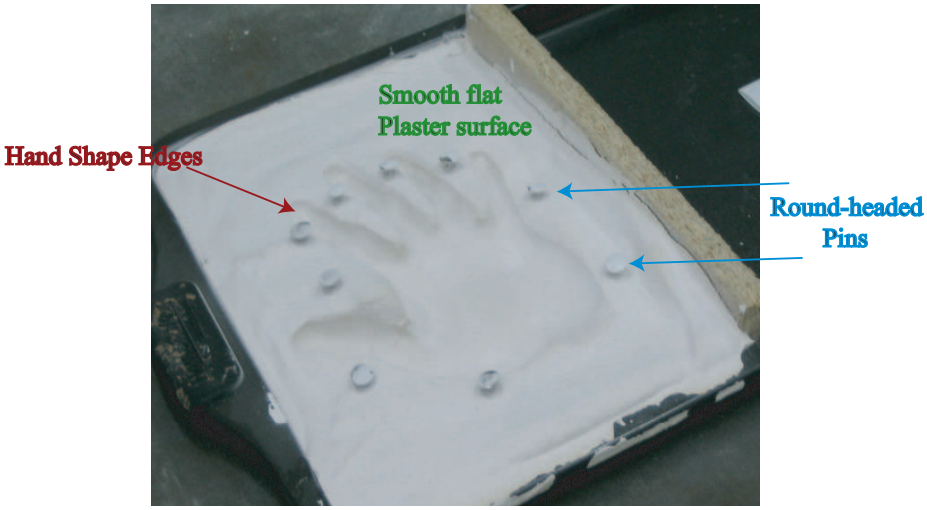


Figure 4.3: Plaster cast fitted with round-headed pins used for scanning.

4.4 Polygonal Mesh Construction and Postprocessing

4.4.1 Merging aligned images and editing resulting polygonal mesh

The resulting aligned scan images are merged to produce a polygonal mesh using the IMMerge module in Polyworks. IMMerge performs a global triangulation over all surface data, connecting points into triangles to produce a polygonal mesh. This step is straightforward and default merging parameters detected by IMMerge were used. For meshing, a maximum distance of 0.1000 and surface sampling step of 0.524 were used (both in mm, same units as scanner data). Maximum distance specifies the maximum acceptable distance between two overlapping distances belonging to two distinct scans. The surface value sets the triangle edge length that the merging algorithm tries to keep more or less constant. the triangle edge length specified should have a value reflecting the point cloud density (IMAlign is able to estimate a default value). The meshed hand produced by IMMerge is shown in Appendix F.

Postprocessing was then carried out on the obtained polygonal mesh using IMEdit. Surfaces that do not belong to the hand had to be clipped. The produced mesh also contained

holes that needed to be filled by interpolating a surface over them. The surfaces where the top and bottom halves of the hand join are irregular and had to be reconstructed. The patchy triangles were removed and a surface reconstructed by interpolating between the triangles on the two sides of the gap formed. These editing steps produced a reasonable hand model with 20,344 points and 36,473 [1] triangles. This was then reduced to decrease the number of triangles on its surface, producing a model has 6894 points and 13,675 triangles. The reduced hand model is shown in Appendix G.

4.4.2 Issues with the polygonal mesh and correction

The arrangement of triangles produced by the merging algorithm is not uniform. The density and direction of the triangles are different on different parts of the hand, as can be seen in Appendices F and G. The mesh is denser where there is more variation in surface shape (along curves) and less dense on flatter surface (the palm). At that point, it did not seem that this would have any impact on the application of the model (rendering on-screen and tracking). Moreover, the idea behind producing a model with a reasonable number of points is that it will make rendering faster. However, when we got to the skinning stage - to define how the skin deforms with movement of the joints - we found that the irregularity of the mesh and the sparsity of points on the palm made it difficult to obtain good results with linear blending. We therefore decided to subdivide and regularise the original polygonal mesh using the powerful subdivision tool from IMEdit. We picked the "*By Edge Length*" subdivision type that aims at uniformly sampling a surface by subdividing triangles until their average edge length reaches a specified value (we used a value of 1; the default is 2). We also enabled the "textitOptimise equilarity" option which enables an algorithm that produces equilateral triangles.
IMEdit offers two interpolation method to subdivide the triangles on a mesh: linear and cubic. The linear method creates new vertices on the surface of existing triangles. With the cubic method, a NURBS surface (made up of splines) is fitted to the surface of existing triangles, and new triangles are mapped onto the NURBS surface producing smooth interpolation. We found that using the linear interpolation method produced a "wrinkle effect" on the skin of the skin which was not acceptable for our purpose. Hence, we chose the cubic interpolation method and obtained a smooth regular mesh that is significantly denser than the original (46,140 points and 92,063 triangles).

The high density of the final mesh is not an issue for our tracking framework since points from the hand are sampled before being used in tracking and rendering of the whole hand is not required at each step of the tracking algorithm. Therefore having a dense skin mesh

[1] note that InnovMetric's estimating algorithm overestimates number of triangles by 10%

has no impact on tracking speed. Actually, it helps tracking in the sense that there are more points to sample from.

4.5 Porting the 3D model to C++/OpenGL

The final hand model obtained from Polyworks was saved in VRML format for export to C++/OpenGL. A perl script was written to extract from the VRML file information required to construct the model in C++/OpenGL: the coordinates of the 3D points on the surface of the hand, the values of the normal at each point, and the set of connected points that make up the triangles. The perl script is provided on the attached CD-ROM.

4.6 Building a skeleton model for the Hand

4.6.1 Estimation of joint positions from a real human hand

We built a skeleton model to define movement of the hand at its joints. This skeleton is constructed as a hierarchical model of joints as explained in Chapter 3, section 3.2. The basic skeleton has 16 joints whose positions need to be estimated from the actual placement of the joints in a real hand. We did this estimation "*by eye*" and present the method we used in this section. We discuss the estimation of the extra imaginary joint of the thumb a bit later.

On each of the four fingers, 7 points (P1 to P7) are picked as shown in figure 4.4 - IMEdit has an option that gives the coordinate values of a selected point. P1 to P5 are straightforward to choose as they are found on the contours of the finger. P6 and P7 are used for estimating the MCP joint which is placed inside the palm. Hence we pick a point P6 on the top (dorsal) surface of the hand and a corresponding point P7 on the bottom side (palmar). We calculate an origin postion for the different joints as follows:

$$DIP_o = P1 - 1/2(P1 - P2)$$

$$PIP_o = P4 - 1/2(P4 - P5)$$

$$MCP_o = P6 - 1/2(P6 - P7)$$

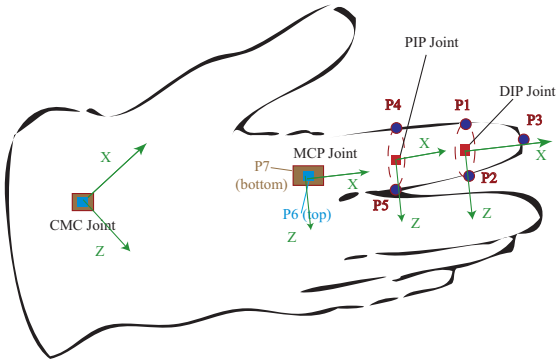


Figure 4.4: Method for estimating joint placement in a real hand.(Hand outline obtained from (Quarl 2006)

The same procedure was used to calculate an origin position for the three joints of the thumb. The position of the CMC joint was chosen to be inside the little groove at the base of the thumb that is noticeable on the dorsal side (top) of the hand - see figure 4.4.

4.6.2 Estimation of a local coordinate frame for each joint

We then want to define local coordinate frames associated with each joint by estimating the directions of the x, y and z axes. We use a right-hand coordinate system and take the x-axis to be pointing in the direction of the bone towards the tip of the finger. The z-axis is thus in the plane of the hand, perpendicular to the x-axis. The y-axis follows from definition of a right-hand coordinate system - see figure 4.4. Given the point P3 at the tip of the finger and let the origin of the joint be P_o, we obtain an estimate of the direction of the x-axis at the joint as follows:

$$\hat{X} = \frac{P3 - P_o}{|P3 - P_o|} \tag{4.2}$$

For the DIP and PIP joint of the four fingers, we obtain an estimate of the direction of

the z-axis using the point pairs P1-P2 and P4-P5. Equation (4.3) shows the method for the DIP joint.

$$\hat{Z}_{DIP(approx)} = \frac{P2 - P1}{|P2 - P1|} \tag{4.3}$$

The same method applies for the PIP joint by replacing P1-P2 by P4-P5 in the calculation. For last two joints of the thumb (the IP and MCP joints), the same procedure is used to get their z-directions.

We need to correct for the z-direction we find because it will not be exactly perpendicular to its corresponding x-axis. This is done by finding the error vector $\vec{z}_e$ and subtracting it from the approximate value for z-direction - see figure 4.5.

$$\begin{aligned} \vec{z}_e &= (\hat{Z}_{approx} \cdot \hat{X})\hat{X} \\ \hat{Z} &= \hat{Z}_{approx} - \vec{z}_e \end{aligned} \tag{4.4}$$

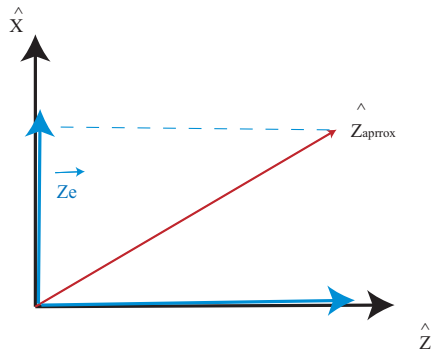


Figure 4.5: Error correction in estimation of z-axis direction.

Once we have calculated a perpendicular set of x and z axes, we obtain the corresponding y-xis by taking the cross-product of the former two.

We cannot directly estimate a z-axis for the MCP joints of the four fingers or the CMC joint of the thumb using the above method due to those joints lying inside of the palm. Instead, normal information that comes with the 3D points is used to estimate a direction for the y-axis. The idea is that the y-axis at points on the surface of the skin right above the joint would point in approximately the same direction as our defined y-axis.

A perl script was written to take in a point specified by the user and give back its index. The index of the point can then be used to look up the value of the normal at that point. We found the normals for a few points lying on the skin surface above and below the MCP/CMC joints and averaged them to give a y-direction. This y-direction is also approximate because it will not be exactly perpendicular to the previously found x-direction. Hence, we need to apply a similar correction to that described for the z-axis of the other joints which were not inside the palm. The z-axis for the MCP joints of the four fingers and CMC joint of the thumb was then obtained by taking the cross-product of the x-direction and y-direction found.

4.7 Defining constraints on finger motion

It is important that a model of the human hand incorporates a set of constraints on the motion of the fingers for two main reasons (Lee & Tosiyasu 1995):

1. avoiding unrealistic motions during hand animation
2. reducing the search space in model-based analysis of hand images

The second reason is of primary relevance to tracking in real-time, which is what we are aiming to achieve with our system eventually. We define three sets of constraints on the hand: static, kinematic and dynamic constraints.

4.7.1 Static Constraints

Static constraints arise due to physical rotation limits on the joints due to the anatomy of the hand. We only consider range of motion of a joint that can be achieved without applying external forces. We derive joint angle limits defined in (McDonald *et al.* 2001) in our model and apply slight modifications judged from empirical observation of Desmond's hand movement limits. These values are given in table 4.6. No static limits have been applied to the two CMC joints (real and imaginary ones) because we were interested in studying how they would model skin deformation around the joint. Eventually, tracking experiments could help determine a suitable static range for the two CMC joints.

Normal Fingers	Y	Z
MCP	[-25, 30]	[-15, 90]
PIP		[-20, 115]
DIP		[-20-90]

Thumb	Y	Z
IP		[-15, 90]
MCP		[0, 90]
CMC		

Figure 4.6: Table showing static angle limits.

4.7.2 Kinematic Constraints

Kinematic constraints are constraints arising from observation of finger motion and do not take into account the forces that are causing them. These can be constraints between joints of the same finger or between rotations of the same joint about two different axes. Inter-finger constraints (between joints of different fingers) also exist but are hard to explicitly represent in equations (Lin & Huang 2001). We defined an inter-joint constraint for our model as follows (Rijpkema & Girard 1991):

$$\theta_{DIP} = 2/3\theta_{PIP} \tag{4.5}$$

This constraint allows us to reduce the number of degrees of freedom of the hand from 26 to 22.

We also defined a constraint between rotations around the y and z axes of the MCP joint of the four fingers (McDonald *et al.* 2001). As the angle of z-rotation increases in the MCP, the amount by which the MCP can rotate in y decreases. Since no data - to the best of our knowledge - is available on the exact relationship between the z and y rotations of the MCP, we used a linear relationship. As the rotation in z increases towards 90 degrees, the rotation range in y decreases linearly towards zero. This gave reasonable motion in the model.

We were very much interested in investigating the relationship defined by equation (4.5). We were suspicious of its veracity due to unsupported evidence from data published in the biomechanics community - see for example data obtained from (Becker & Thakor 1988). We thought that maybe this relationship held for some situations but not in others (i.e. it could have a dynamic range that would depend on some other parameters). We present the brief investigation we did from data obtained in the biomechanics literature in Appendix A.

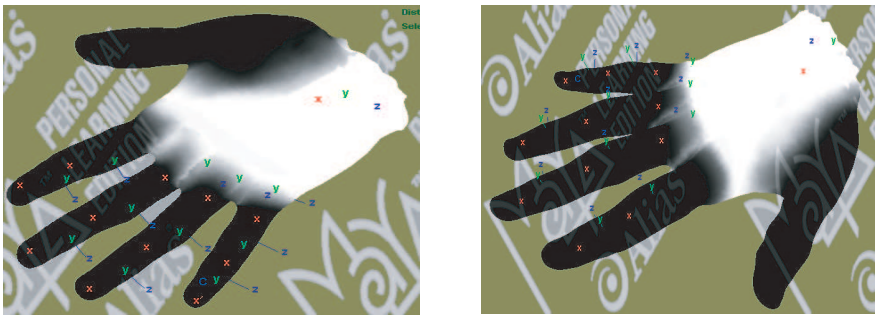
4.7.3 Dynamic Constraints

In specifying dynamic constraints, we want to use knowledge of the forces that are causing motion of the fingers to impose limits on the values that joint angles can take within some time-frame where that force has an effect. We can use force information to calculate a bound on the change of angle $\Delta\phi$ of a joint in a given time step Δt given that we have information about the maximum force that can act at that joint (as explained in details in section 3.3). We present the outline of a method that could be used for implementing dynamic constraints in our model in Appendix B.

4.8 Authoring the skin

We used linear blending in Maya to model the deformation of the skin as the joints of the skeleton move - a process known as skin authoring in computer graphics. This defines a relationship between the position of the points on the skin surface and the movement of the joints as given by equation 3.14 from the previous chapter. We defined a set of 3 bending weights for each point; i.e. each point in our model can be affected by a maximum of 3 joints. This produced realistic deformation of the skin in our model. It should be noted that the linear nature of the linear blending algorithm prevents complex deformations like wrinkles. However, this does not pose a problem for our application. We show pictures of skin deformation before and after linear blending in Appendix J.2.

We comment here that painting skin weights using Maya was probably the most painful and most time-consuming part of the project. We grouped points on the hand into subgroups of 3 types: those affected by one joint only, those affected by 2 joints, and those affected by 3 joints (figure I.1 in Appendix I). This issue arose because for points that only have one joint influence (so that the other 2 weights should be zero), Maya tries to be smart and fills up values for those, which muck up the overall skinning. We first obtained an overall rough blending weight definition on the hand. Then, we activate those subgroups and force weights due to non-influencing weights to zero, starting from the type with 3 influences (regions between fingers), then the type with 2 influences (regions on the joints of a finger), then the type with 1 influence (region on the finger bone between the joints). This made sure the weights we set to zero do not subsequently get changed as we move to painting other regions.



(a) Bottom View

(b) Top View

Figure 4.7: Skin weight painted on the hand with respect to the palm joint; the darker the area the less the influence of the joint.

4.9 Using the hand for tracking

The resulting hand model we obtained - which we will call the "NICTA hand" - was tested in different offline tracking scenarios. At the moment, real-time tracking has not yet been achieved using the tracker we have. We used 3 basic tracking scenarios:

- A grip motion.
  All fingers close in towards the palm.
- A spin motion.
  The hand is rotated around the x-axis of the palm and most of the fingers gradually get occluded behind the pinky.
- A translate motion.
  The hand is translated along the axes of the palm with no rotation at the finger joints.

Desmond collaborates with people from the ETH Zürich and has been using a hand model they made in the development of his tracking algorithms. The 3 scenarios described above have been used with the ETH hand as well. We can therefore compare how the two models perform in each situation.

There are two basic differences between the NICTA hand model and the ETH hand model:

1. The NICTA hand model has an additional PIP-DIP kinematic constraint ($\theta_{DIP} = 2/3\theta_{PIP}$).

2. The modeling of the thumb is different.
   The NICTA hand has an extra imaginary CMC joint to model opposition motion of the thumb. The ETH hand has no extra joint for the CMC but has an extra degree of freedom at the MCP of the thumb.

Tracking Steps in the 3 basic scenarios:

- The new hand is used in tracking without the extra PIP-DIP constraint.
- The PIP-DIP constraint is added and tracking is performed with th augmented model.
- Results are compared with those obtained with the ETH hand.

Testing and analysis of extra features:

- A natural finger curling motion is used to analyse the effect of the PIP-DIP kinematic constraint. The index finger is curled by bending movements at the PIP and DIP joints in natural fashion.
- A forced finger curling motion is used to analyse the effect of the PIP-DIP kinematic constraint. The index finger is curled by bending movements at the PIP and DIP joints and the DIP angle is forced to remain close to zero.
- The movement at the thumb CMC joint (with both the actual and imaginary joints) is tracked and results are compared to ETH hand.

Chapter 5

Results and Discussion

5.1 The Hand class

The resulting hand class we implemented is shown in figure 5.1. The hand is defined by a skin, a skeleton and a set of constraints. The skin consists of 46,140 surface skin points and 92,063 triangles. The skeleton has 17 joints - one of which is an imaginary joint define to model realistic movement of the thumb CMC joint. We defined a set of static and kinematic constraints on the hand as described in the previous section and provide a method to implement dynamic constraints. The code for the hand model[1] is included on the attached CD-ROM, along with the definition of local joint coordinate frame transformations that describe the skeleton hierarchy.

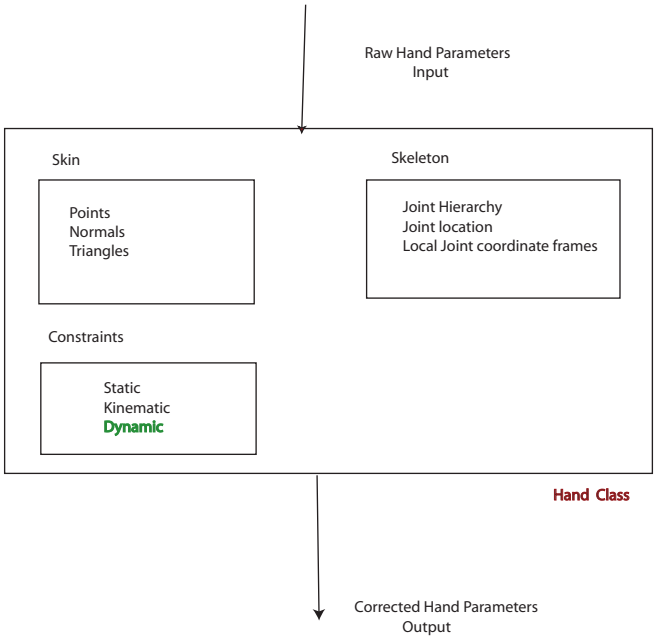


Figure 5.1: The resulting hand class we implemented (except for the dynamic constraints)

[1]the VRML model (.wrl file) can be viewed in a web browser - it works in firefox but a plugin might need to be downloaded.

5.2 Tracking Results and Analysis

All tracking videos are provided with the attached CD-ROM.

5.2.1 Tracking results and analysis in the 3 basic scenarios

The tracking frames for the 3 basic scenarios described in section 4.9 are presented in Appendix K. The results are summarised in the table below:

	GRIP	SPIN	TRANSLATE
ETH Hand	Fails on Middle and Fourth fingers	Fails	Tracks (noticeable drift in palm joint)
NICTA Hand (no extra constraint)	Fails on Middle finger	Fails	Tracks (less drift of palm joint)
NICTA Hand (with extra constraint)	Tracks well in 2 out of 9 cases	Fails	Tracks (less drift of palm joint)

Figure 5.2: Summary of tracking results in 3 basic scenarios: Grip,Spin, Translate.

Grip Motion

The grip motion is a difficult motion to track due to many joints moving at the same time and partial occlusion of finger links. In the grip motion (figure K.1 in section K.1) the NICTA hand with no extra constraint and the ETH hand both fail to track the middle finger. Significant deviation from the actual finger motion is observed in both cases. The ETH model also fails to track the fourth finger in the grip situation, while the NICTA hand shows acceptable tracking of the fourth finger. Interestingly, adding the extra PIP-DIP constraint produces good tracking results in 2 out of 9 runs. This indicates that having the constraint might help although the results were not accurately reproducible.

Spin Motion

Both hand models could not track the spin motion in which a large amount of occlusion occurs (figure K.2 in section K.2). The extra constraint did not help in this situation. We can therefore deduce that adding that extra constraint has no significant effect on

performance in the case where a lot of occlusion occurs. Another point to note is that the constraint does not however make the tracking worse.

Translate Motion

Both hand models could track the translation movement of the palm (figure K.3 in section K.3). The ETH hand shows a noticeable drift in the tracked position of the palm joint during the tracking. This drift is not as significant in the NICTA model case. It could be caused by the fact that the ETH hand is not exactly fitted to Desmond's hand. This indicates that modeling more accurate hand shape could reduce the amount of work done by the tracker. However, it also shows that the tracker can work with a non-exact model, as long as the dimensions of the model (length and width of the fingers) are fitted to those of the hand being tracked.

5.2.2 Testing and analysis of extra features

Natural Index curling motion

We analysed the effect of adding the PIP-DIP kinematic constraint by bending the index finger at the PIP and DIP joints in a natural fashion. The finger movement is shown in figure 5.3. Tracking was performed at three different speeds: the normal captured speed of 30 frames/s (30 Hz), by skipping each two frames from the original sequence giving a speed of 90 Hz and by skipping each three frames from the original sequence giving a speed of 121 Hz.

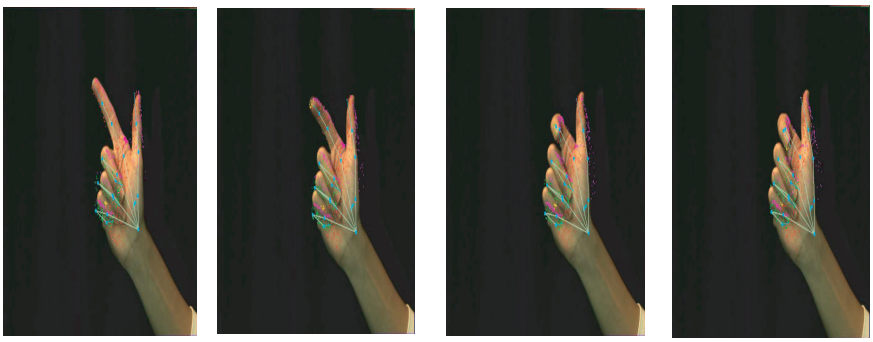


Figure 5.3: Tracking of natural finger curling of Index finger over 37 frames (frames 0-20-30-36 are depicted).

To analyse what is happening in the tracking, we plotted graphs for all tracking situations at the three tested speeds and using the NICTA hand model with and without the

PIP-DIP constraint. These are shown in figures 5.4 to 5.6.

Figure 5.4 shows the variation of PIP angle over frame number as the tracking sequence progresses, for all the six situations investigated. We take the original sequence as the benchmark to compare with the faster sequences. We note that having the extra PIP-DIP constraint provides no significant improvement on the tracking result at a tracking speed of 30 Hz. In the second situation where tracking is performed at 90 Hz, reasonable tracking results are also obtained with and without the extra constraint. In the fastest tracking case, at 121 Hz, tracking of the finger movement deviates significantly in both situations where constraints are not used and used. This result indicates that the PIP-DIP constraint does not have a major impact on the PIP angle; a reasonable outcome since the constraint is defined on the DIP angle with respect to the PIP angle. We can also observe that tracker is not robust enough yet to cope with speeds above 90 Hz for this particular motion.

Similarly, figure 5.5 shows the variation of DIP angle over frame number as the tracking sequence progresses, for all the six situations investigated. The tracking is reasonable for both situations of constraint and no constraint at 30 Hz and 90 Hz but fails at 121 Hz. However, we note that the final DIP angle reached, for the case where the constraint is applied, deviates significantly less from the benchmark angle (around -50 at 30 Hz) than for the case where the constraint is not applied - DIP angle reaches around 43 degrees with the constraint applied compared to 25 degrees when the constraint is not applied. This is an interesting result and also suggests that having the PIP-DIP constraint can help tracking at faster speeds. The indication here is that it is worthwhile investigating how to refine the set of kinematic constraints on the hand.

Figure 5.6 shows a plot of DIP versus PIP angle. The first thing to note is that in the case of tracking at 30 Hz (the benchmark case with good tracking results), the DIP-PIP relationship has the same linear relationship as in the cases where constraints are applied ($\theta_{DIP} = 2/3\theta_{PIP}$)- the curve lies on the same line where the 2/3 relationship holds. This validates the DIP-PIP 2/3 relationship and indicates that it indeed holds for natural finger motion - at least in our testing situations. The deviation of the DIP-PIP relationship from the 2/3 constraint increases as tracking speed increases. Therefore, this also suggests that the constraint may help track fast motion.

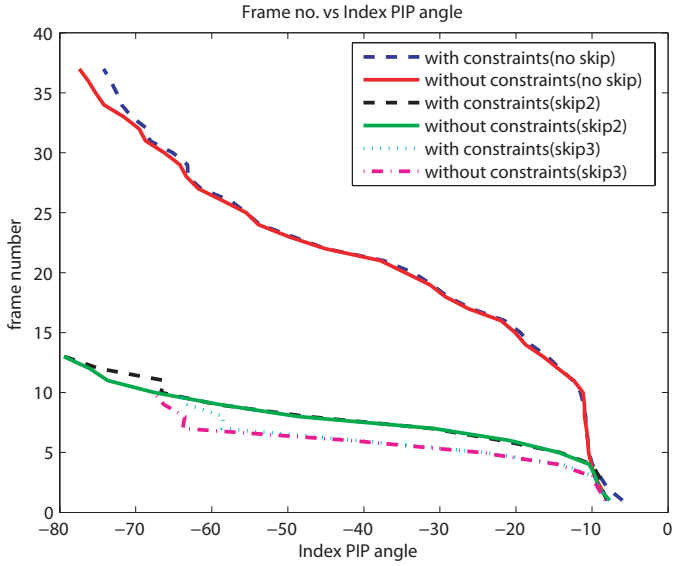


Figure 5.4: Graph of PIP angle as a function of frame number for 3 tracking situations: at 30 Hz (no skip), at 90 Hz (skip 2) and at 121 Hz (skip 3) - frame rate without skipping is 30 frames/s.

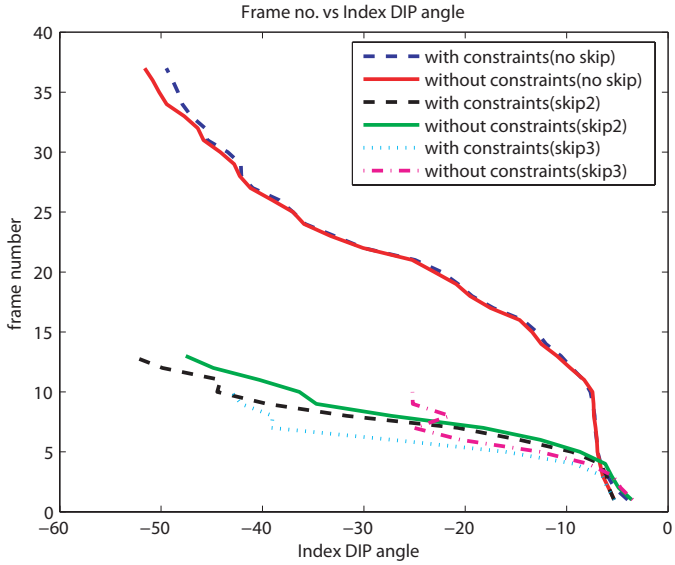


Figure 5.5: Graph of DIP angle as a function of frame number for 3 tracking situations: at 30 Hz (no skip), at 90 Hz (skip 2) and at 121 Hz (skip 3) - frame rate without skipping is 30 frames/s.

Forced Index curling motion

We also investigated what would happen in the tracking if the actual hand motion is not following the 2/3 relationship of the PIP and DIP joints. Desmond forced his index finger to curl in such a way as to maintain the DIP angle as small as possible while the PIP

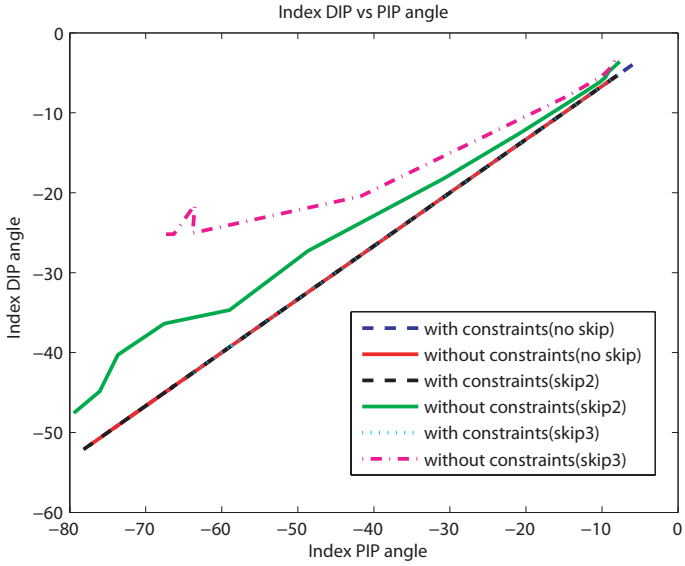


Figure 5.6: Graph of DIP angle as a function of PIP angle for 3 tracking situations: at 30 Hz (no skip), at 90 Hz (skip 2) and at 121 Hz (skip 3) - frame rate without skipping is 30 frames/s.

bends. This forced finger movement is shown in figure 5.7. Tracking was performed with and without the PIP-DIP constraint.

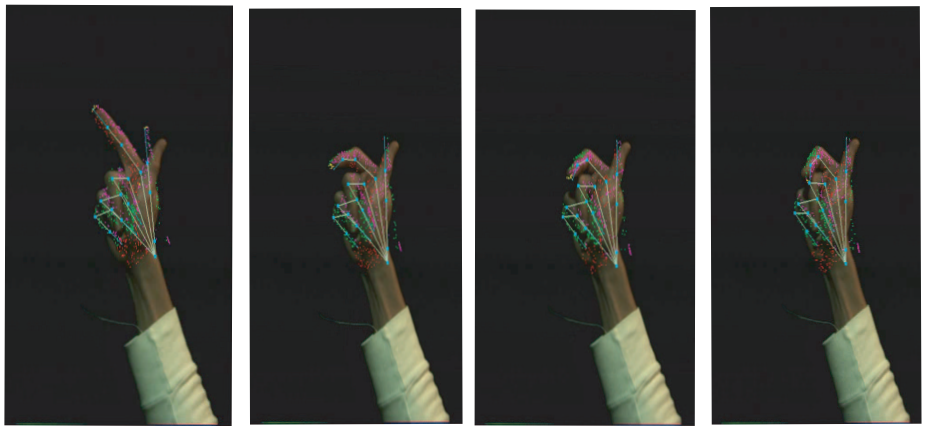


Figure 5.7: Tracking of forced finger curling of Index finger over 37 frames (frames 0-20-30-36 are depicted).

We plotted graphs , shown in figures 5.8 to 5.10, to analyse what is happening to the PIP and DIP angles in this forced motion situation. As expected, having the PIP-DIP constraint did not have much impact on the tracking of the PIP angle. This can be seen in figure 5.8 where the deviation between the cases with and without the added constraint is quite small. Figure 5.9 indicates that the deviation between the constraint and no constraint cases increases as the DIP angle increases (we ignore the sign here as it arises from the way the local coordinate frame is defined). This means that the 2/3 relationship is more likely to hold for small DIP angles, and increasing deviates as DIP angle increases.

Figure 5.10 shows a plot of PIP versus DIP angles for tracking of the forced finger motion with and without the additional PIP-DIP constraint applied. Again, we can deduce that the 2/3 relationship is close to reality for small DIP and PIP angles. Obviously, in this case of forced motion, the deviation then becomes increasingly significant as the PIP or DIP angle increases. However, it is interesting that even when we are trying to force the 2/3 PIP-DIP relationship to be untrue, it is still close to 2/3 for small angles.

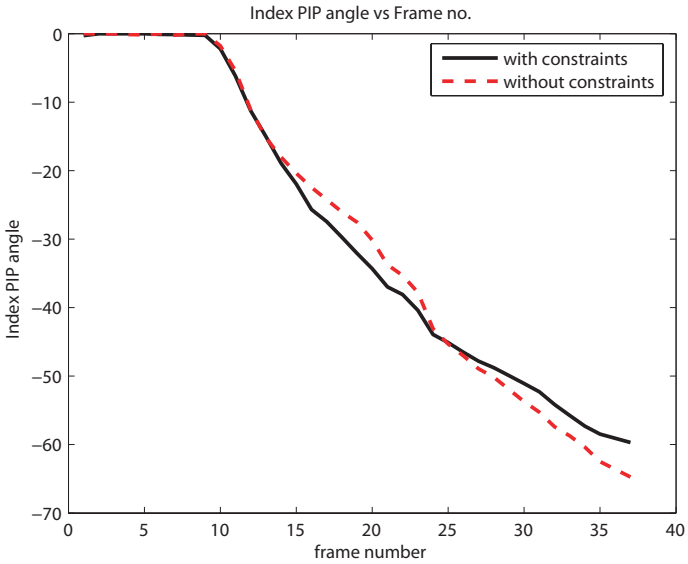


Figure 5.8: Graph of PIP angle as a function of frame number during a forced curling motion of the index finger.

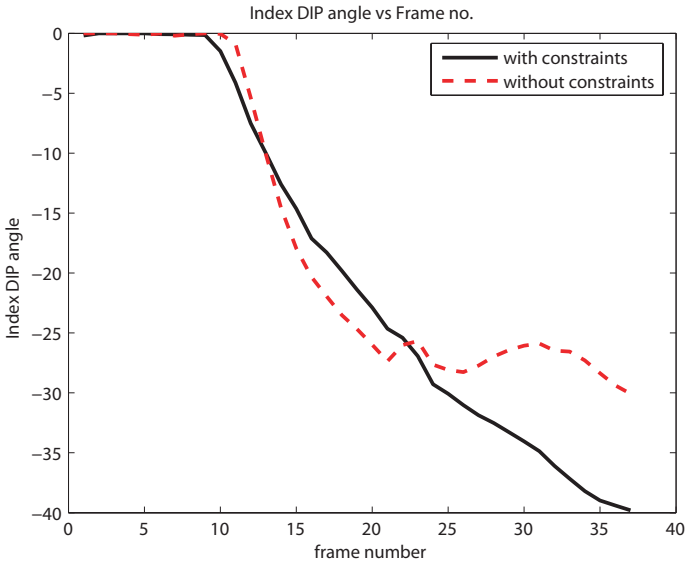


Figure 5.9: Graph of DIP angle as a function of frame number during a forced curling motion of the index finger.

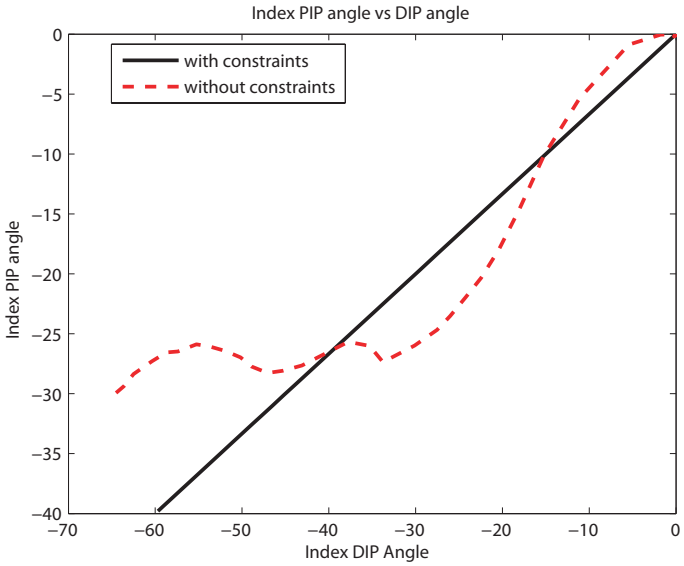


Figure 5.10: Graph of PIP angle VS DIP angle during a forced curling motion of the index finger.

Movement of the thumb at its CMC joint

Movement of the thumb at its CMC joint is tracked using the ETH hand and the NICTA hand. Results are shown in Appendix K, figure K.4. Both models provide equally good results of tracking the the thumb movement at the CMC. We can hence say that the imaginary joint helps define motion at the CMC that is at least as good as the motion in the ETH model where there is no extra joint, but the CMC is given an extra degree of rotation in the same way described in (Rijpkema & Girard 1991) and (McDonald *et al.* 2001).

Chapter 6

Conclusion and Further Work

6.1 Conclusion

In this work, the creation process of a virtual 3D model for a human hand was presented. The primary application for which the virtual hand model was developed is for tracking of a human hand. Current work in this area of research - tracking of a hand as a highly articulated object - has not achieved real-time performance yet, to the best of our knowledge. Therefore, we tested the use of the hand in offline tracking scenarios using video sequences captured at a resolution of 640x480 pixels and a rate of 30 frames per second.

Our hand model (the "NICTA hand") consists of a skin surface and an underlying skeleton. The skin surface is a triangular mesh defined by 46,140 points and 92,063 triangles specifying how those points are connected. The skeleton has 17 joints and 26 degrees of freedom that are used to define movement of the hand. Our skeleton has an extra imaginary joint to model realistic motion of the CMC joint of the thumb. A set of 3 bending weights, corresponding to the influence of 3 joints, are defined on each vertex of the skin mesh using linear blending. These weights are used to compute a transformed position for each skin point as the skeleton is moved such that realistic deformation of the skin is obtained. The NICTA hand gave good results, comparable to the ETH model, in the tracking of the thumb motion, which indicates that the extra joint is giving good reproduction of thumb movement.

A set of constraints is also defined on motion of the hand at its joints. This helps in reducing the search space during tracking as poses disallowed by those constraints do not have to be taken into account. We used static and kinematic constraints on the hand and present a method for a future implementation of dynamic constraints. We investigated in detail one particular kinematic constraint defining a relationship between the PIP and DIP angles of the digits ($\theta_{DIP} = 2/3\theta_{PIP}$). When used, this constraint reduces the number of degrees of freedom of the hand from 26 to 22. Tracking results obtained with a natural index finger curling motion (at 30 Hz) demonstrated that the DIP angle was indeed 2/3 of the PIP angle during the motion in the case where this relationship was

not forced (i.e. the constraint was not applied). We then made this curling motion faster by skipping frames in the sequence, obtaining two set of sequences at 90 Hz and 121 Hz. The tracking produced reasonable results at 90 Hz but failed at 121 Hz. This suggests that using a PIP-DIP constraint could help in tracking faster motion but the relationship might need further refinement. We also investigated a case of forced index curling motion where Desmond tried to keep the DIP angle of his finger as small as possible in an attempt to invalidate the PIP-DIP relationship. The tracking results interestingly showed that although the motion was forced, θ_{DIP} was still close to $2/3$ θ_{PIP} for small values of θ_{DIP} and θ_{PIP}. This further confirms the validity of the constraint in natural hand movement.

Tracking was also performed using three different motions:grip, spin and translate. We compared tracking in those scenarios using the NICTA hand with and without the PIP-DIP constraint, to tracking using the ETH hand. In the grip motion tracking, the ETH hand and the NICTA hand without the extra constraint both fail on the third finger, although the ETH hand also fails on the fourth. The NICTA hand with the constraint gave good tracking results 2 out of 9 tested cases. Although this gave results that were not accurately reproducible, it served to show that having constraints defined on finger motion could help in the tracking of difficult motion like a grip where many joints are moving at the same time. In the spin motion, there is heavy occlusion as most of the fingers gradually move behind the pinky. Tracking with both hand models failed in that case, even with the constraint added to the NICTA hand, suggesting that the extra constraint does not help in the case of heavy occlusion but does not seem to make the results worse. The translate motion tracked well with both models, with and without the extra constraint. The palm joint was noticed to drift less during the tracking when the NICTA model is used. This suggests that having a more accurate hand shape model could help in reducing the amount of work done by the tracker. It also indicates that exactness of the model does not matter in the tracking framework we used, as long as the dimensions of the model (length and width of the fingers) are fitted to those of the hand being tracked.

Hence, we conclude that having an extra PIP-DIP constraint is more likely to help tracking than to make it worse, at least in the tracking scenarios we used. Our results show that the constraint we used is accurate enough for predicting natural hand motion and could also help in tracking faster motion. For the latter case, the constraint would have to be refined and other constraints added to the hand. The slight improvement obtained with the PIP-DIP constraint suggests that it is worthwhile investigating other constraints on hand motion to help produce better tracking results. Due to time restrictions, dynamic constraints investigated were not implemented in the model. Implementing those would constitute the obvious next step.

6.2 Further Work

6.2.1 Implementing dynamic constraints

Dynamic constraints were investigated but given the timeframe of this thesis, they were not implemented on the hand model we built. We have strong reasons to believe that adding those constraints would significantly improve tracking, based on the results we obtained just by adding one extra kinematic constraint defining a relationship between the PIP and DIP joints of a finger. The results served to indicate that refining the set of constraints on the hand to reflect more realistic movement has the potential to improve tracking. Dynamic constraints would be an improvement on kinematic ones, in the sense that they could more accurately model hand movements by taking into account the forces producing them. The kinematic constraints we used was just based on empirical observation of finger motion as mentioned in (Rijpkema & Girard 1991) and (McDonald *et al.* 2001).

6.2.2 Other Constraints

Finding other kinematic constraints that could be applied to the hand could be another area of investigation. At the moment, we are using two types of kinematic constraints; an inter-joint constraint between joints of the same finger (PIP and DIP) and a constraint on rotation about two different axes of the MCP joint of the digits. Observation of natural hand motion suggests that dependencies exist between the motion of the different fingers. For example, the middle finger cannot be bent without the fourth finger also bending. Those relationships could be very complex and difficult to define in closed form (Lin & Huang 2001). An attempt is made by (Lee & Tosiyasu 1995) to model the dependencies of MCP joint angle limits on those of neighbouring fingers by expressing those limits as a set of inequalities. This could be a starting point for further refinement of kinematic constraints on our hand model.

Lin *et al.* (Lin & Huang 2001) also mentions constraints that are imposed by the naturalness of hand motion, for which a closed form definition would not exist. One example of such motions is the order by which the finger joints are bent when closing in the fingers to make a fist. They propose a method by which those constraints arising from natural hand motion are learnt from a large and representative set of training samples. A learning-based approach to constraint modeling and definition could be another area of further work.

At the moment, no constraint has been applied yet to the both the actual and imaginary CMC joints of the thumb in our model. We were interested in studying how those two joints would model movement at the CMC of the thumb and therefore wanted to have them unconstrained initially. The next step would be to try and find appropriate static and kinematic constraints for those two CMC joints. One approach to find static limits could be to use some arbitrary initial values and test the effect on tracking. A whole set of those values in a given range could be tested to determine which ones are best suited to define the bounds of the range of motion at the two joints.

6.2.3 Making an "eigen-hand"

We eventually want to use the hand model to track arbitrary hands belonging to different persons. A method for achieving this was briefly analysed with the aim of implementing it if time permits after tracking experiments on the NICTA model have been performed. To fit an arbitrary hand automatically to our hand model, a pre-tracking module would be implemented. In this module, the person whose hand we want to track places her hand in front of the pair of calibrated cameras currently used for tracking and the basic dimensions of her hand are obtained using 3D reconstruction from the camera images. In this module, cylindrical coordinates are used to define the position of points on the hand. This makes it easier to then fit the dimensions of the person's hand to the virtual model as the fingers are roughly cylindrical in shape. The spherical coordinate values are then converted to Cartesian values before tracking is done.

6.2.4 Improvements on the tracking end

At the moment, the actual values of the joint angles of the hand are not known during tracking. Having those values would help gauge performance of tracking as it would allow us to measure deviations of the movement predicted by the tracking from actual angle values. A glove can be worn with sensors that can directly measure the joint angle values - an example of such a glove is the cyberglove used by (Zhou 2005). However, this device has a prohibitive price. Another method would be to use stereo techniques to reconstruct the actual hand pose from images captured.

Desmond has investigated a novel approach of initialising step-size in his stochastic optimisation setting for hand tracking. His new method, termed "optical flow", has been submitted for publication at the time of writing which makes it inappropriate to discuss it here. However, we can say that this method gave reproducible results when applied to

a grip motion when tested with the ETH hand. The method is yet to be tested with the NICTA hand but it looks likely that it will provide improvement on the tracking with our model as well.

Appendix A

Investigation of PIP-DIP relationship

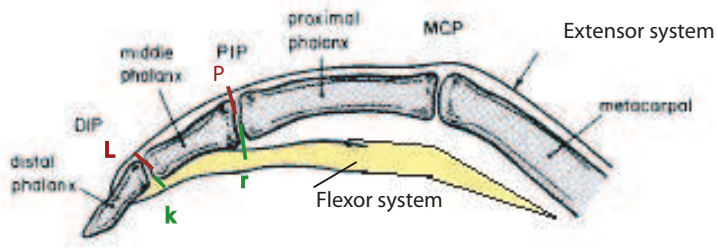


Figure A.1: Tendon System in the finger (Becker & Thakor 1988): P is the distance of extensor system from PIP, L is the distance of extensor system from DIP, r is the distance of flexor system from PIP, k is the distance of flexor system from DIP (by distance we mean perpendicular distance)

Landsmeer (Landsmeer 1955) defined a relationship stating that $\theta_{DIP} = P/L\ \theta_{PIP}$ (see figure A.1) but provided no numerical data. We tried investigating this relationship by trying to find numerical data in the biomechanics literature. We were unable to find specific extensor data relevant to our study, but found some flexor data from (Armstrong & Chaffin 1977).

Landsmeer extensively studied tendon displacement in the fingers. He proposes a model defining a linear relationship between joint angle θ and tendon displacement x.

$$x = r\theta \tag{A.1}$$

where r is the moment arm of the tendon from the joint.

We reasoned that if this relationship held, then for a tendon spanning both the PIP and

DIP joint, and being displaced by an amount x, we have

$$x = r_{pip}\theta_{pip} \tag{A.2}$$

$$x = r_{dip}\theta_{dip} \tag{A.3}$$

Equating the two equations we obtain that $\frac{\theta_{dip}}{\theta_{pip}} = \frac{r_{pip}}{r_{dip}}$.

We used flexor data from (Armstrong & Chaffin 1977) to plot graphs of tendon displacement of the FDP (flexor profundus) tendon versus joint angle for the PIP and the DIP (figure A.2). We fitted a line through the points in each case and obtained the description of the line - shown in figure A.2. We ignore the offset of each line from the origin and assume the gradient is equal to the moment arm at that joint. The ratio of the PIP to DIP gradient is 2.6 (a value far off from 2/3). This suggests that there might be the effect of more than one tendon to take into account in modeling the flexor system as defined by Landsmeer.

We also show a maximal flexion values for the fingers in the table shown in figure A.3. We compute ratios for the MCP to PIP angles and DIP/PIP angles. In (Kuch & Huang 1995), a possible linear relationship between MCP and PIP is defined ($\theta_{mcp} = 1/2\ \theta_{pip}$). Values in table A.3 falsify this assumption, and we decided not to use it in our model.

However, for the DIP/PIP ratios shown in table A.3 are close to 2/3, which suggests that it might be accurate in some situations.

We had to stop our review of the biomechanics literature due to time restrictions, but this could constitute further work.

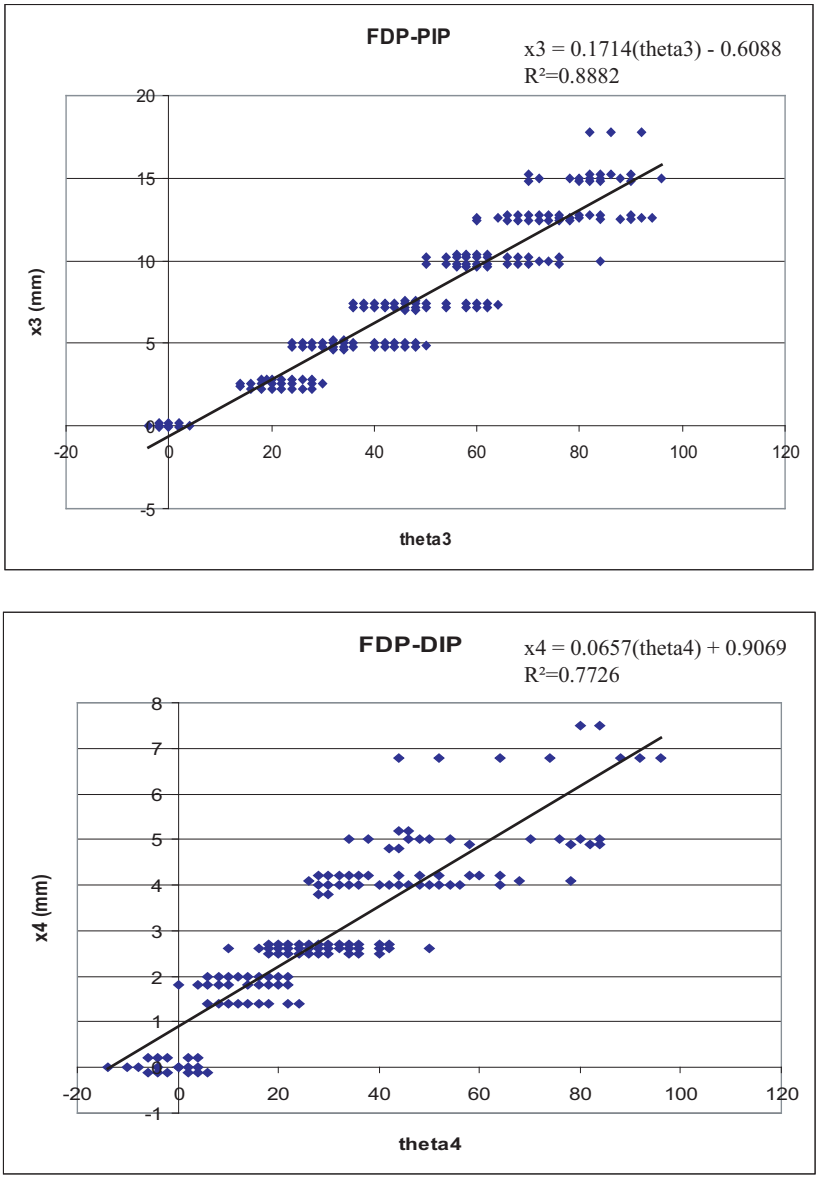


Figure A.2: Graph of tendon excursion (moment arm) versus PIP and DIP joint angles - R^2 is the mean-squared error

	Index	Middle	Fourth	Pinky
MCP	70.83 (11.09)	85.03 (9.87)	85.09 (14.46)	85..58 (18.09)
PIP	103.87 (7.79)	103.98 (8.98)	107.15 (13.49)	98.95 (11.20)
DIP	61.17 (12.71)	73.64 (16.30)	66.96 (15.77)	70.79 (15.84)
MCP/PIP	0.68	0.82	0.79	0.86
DIP/PIP	0.59	0.71	0.62	0.72

Figure A.3: Table showing means and standard deviations (in bracket) for maximal flexion values collected from 15 subjects (Becker & Thakor 1988). Ratios of MCP/PIP and DIP/PIP are also shown.

Appendix B

Dynamic constraints recipe

We propose a method, based on the work by (Albrecht & Haber & Seidel 2003) and (Tsang & Singh & Fiume 2005), for an initial implementation of dynamic considerations in our model by defining dynamic constraints on the joint motions. The dynamic constraints we have in mind are limits on the change of joint angles ($\Delta\phi$) that can occur in a given time step Δt.

Our starting point is the force-torque equation as given by equation(3.9), which we reproduce here:

$$T = sgn\left(\left\langle \vec{a}, \vec{r} \times \vec{F} \right\rangle\right) . \left\| \vec{r} \times \vec{F} \right\|$$

Equation(3.9) does not consider friction yet. This means that a muscle contraction would cause a rotation that will not stop. Therefore, we modify the equation to account for friction by subtracting the torque due to friction:

$$T = sgn\left(\left\langle \vec{a}, \vec{r} \times \vec{F} \right\rangle\right) . \left[\left\| \vec{r} \times \vec{F} \right\| - \mu . \left|\omega_{old}\right|\right] \tag{B.1}$$

where μ is the coefficient of friction. We use $\mu = 0.015$ as suggested in (Albrecht & Haber & Seidel 2003).

A joint is spanned by more than one muscle, therefore the total torque acting at a particular joint is given by the sum of the torques due to every muscle that spans the joint. The net torque at a given joint that is spanned by n muscles is given by:

$$T_{net} = \sum_{i=1}^{n} T_i \tag{B.2}$$

where T_i is the torque contributed by muscle i.

We use equations (3.6) to (3.8) to find the maximum total force that could act at a

given joint. We know that the highest tension, $\vec{F}_{max}$ during active contraction is produced when the muscle is at its resting length (Tsang & Singh & Fiume 2005). Therefore the total maximum force that can act at a joint is given by $0.96\vec{F}_{max}$ for a contraction value c of 1.

In (Albrecht & Haber & Seidel 2003), we are referred to (Brand & Beach & Thompson 1981) for relative values of $\vec{F}_{max}$ for all relevant hand muscles. It is not clear to us how Albrecht *et al.* obtained their values for $\vec{F}_{max}$ from the data presented in (Brand & Beach & Thompson 1981). Brand *et al.* present values of "tension fraction" for the muscles of the hand. What they term "tension fraction" is a percentage of the single-contraction tension capability of all muscles below the elbow. Those values were found from the mass of the muscles. The reason behind stating a percentage value rather than an absolute numerical value is the large variability of tension values in the muscles of the hand for different people. It was however stated that each 1% of tension fraction is equivalent to about 5 kg (for the data used by the authors) (Brand & Beach & Thompson 1981). We interprete this as meaning that each 1% can sustain a force of $5g$ where g is the value of gravity. We can therefore try using this to convert values of tension fraction presented into absolute values that we can use in our computation.

Next we need a force direction taken to be along the line of action of the force. Tsang *et al.* (Tsang & Singh & Fiume 2005) propose a method to estimate force direction by representing the line of action as a sequence of piecewise linear segments extending through one or more joints. They estimated the origin and insertion points for every muscle at every joint it crosses. This was done by manual fitting of digitized muscle fibres and tendon data to their 3D skeleton model using Maya. A similar estimation method could be developed for our model. The force direction $\hat{F}$ is then given by:

$$\hat{F} = \frac{\vec{i} - \vec{o}}{\left\| \vec{i} - \vec{o} \right\|} \tag{B.3}$$

Moment arm ($\vec{r}$) data for the muscles of the hand are provided in (Albrecht & Haber & Seidel 2003). Therefore, we can calculate a value for T_{net} at each of the joints we are interested in. We now plug this into the set of equations(3.12) to give:

$$\Delta\omega = \Delta t.J^{-1} \cdot T_{net}$$

$$\omega_{new} = \omega_{old} + \Delta\omega$$

$$\Delta\phi = \Delta t.\omega_{new}$$

$$l \leftarrow l - \Delta\phi.\,\|\vec{r}\|$$

We know the value of Δt (the video frame rate) so that we can obtain a value for $\Delta\phi$, which gives the bound on maximum change of angle of a particular joint that can occur in a time Δt (i.e. between two consecutive video frames). This gives us our dynamic constraint defined on a joint.

Appendix C

Laser Triangulation system

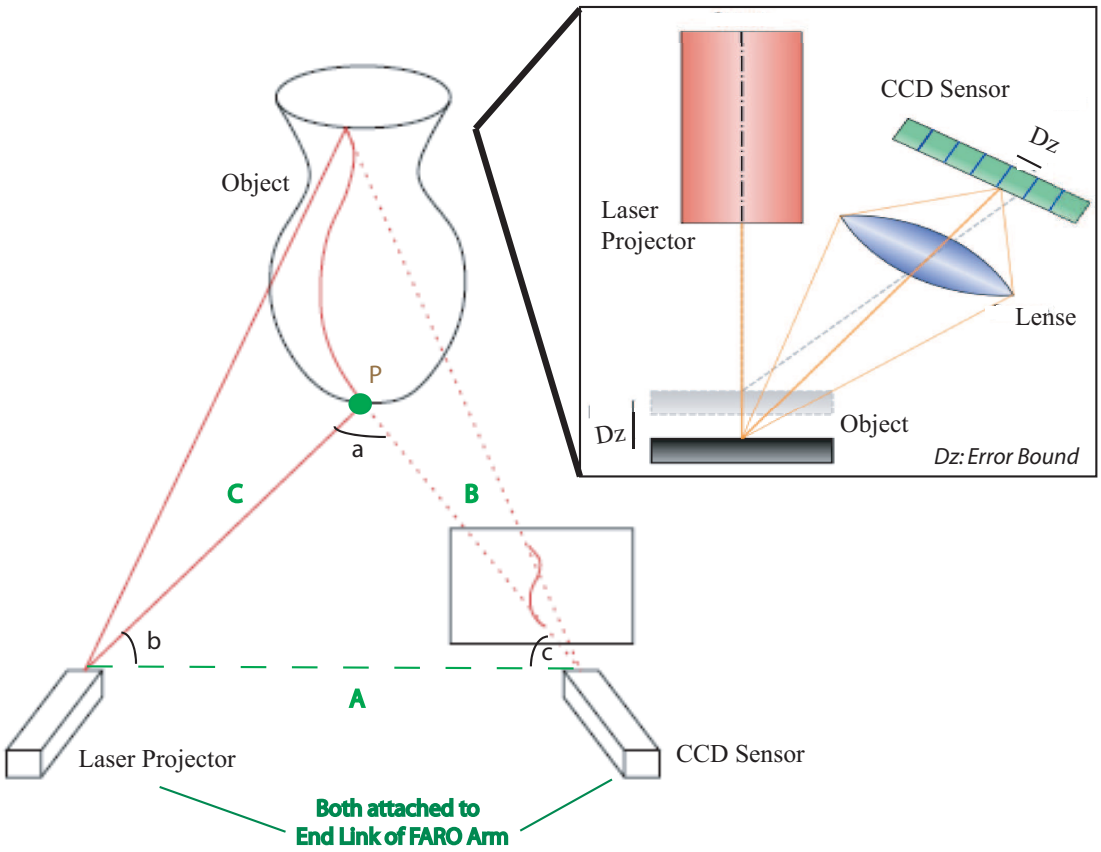


Figure C.1: Laser Range Scanner System using triangulation method (Wikipedia, 3D Scanner 2006), (Bernardini & Rushmeier 2002)

Appendix D

Scanning Process and Equipment

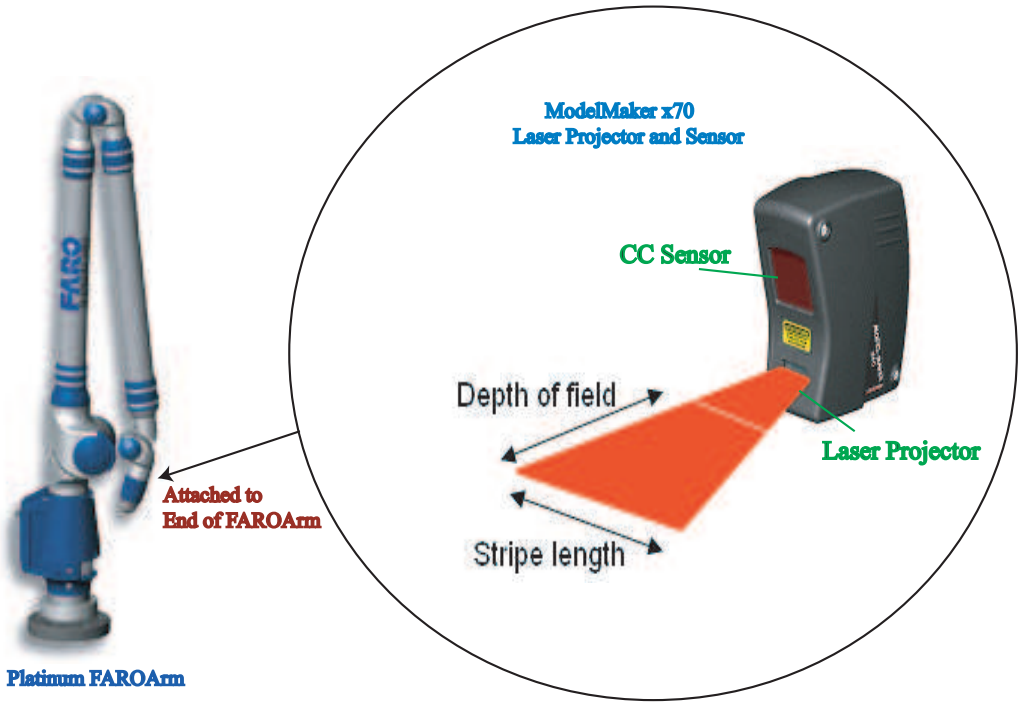
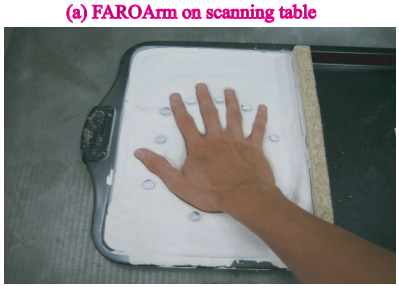


Figure D.1: 3D Scanning Equipment: 7-axes of rotation FAROArm on which a ModelMaker x70 laser scanner system is mounted.

(a) FAROArm on scanning table

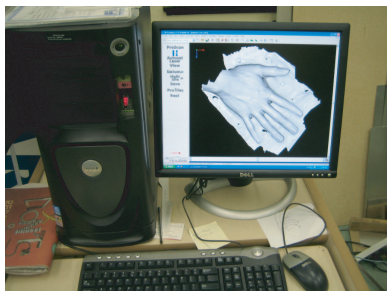
(b) Plaster Cast of Bottom Half of Hand



(c) Top Half of Desmond's actual hand

(d) The Scanner System on FAROArm

(e) Laser stripe onto the Hand



(f) Overlaid range images from scanner

(g) Resulting Polygonal Mesh after postprocessing

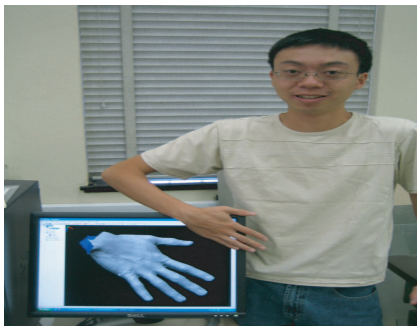


Figure D.2: Scanning Equipment and Process

Alignment Process

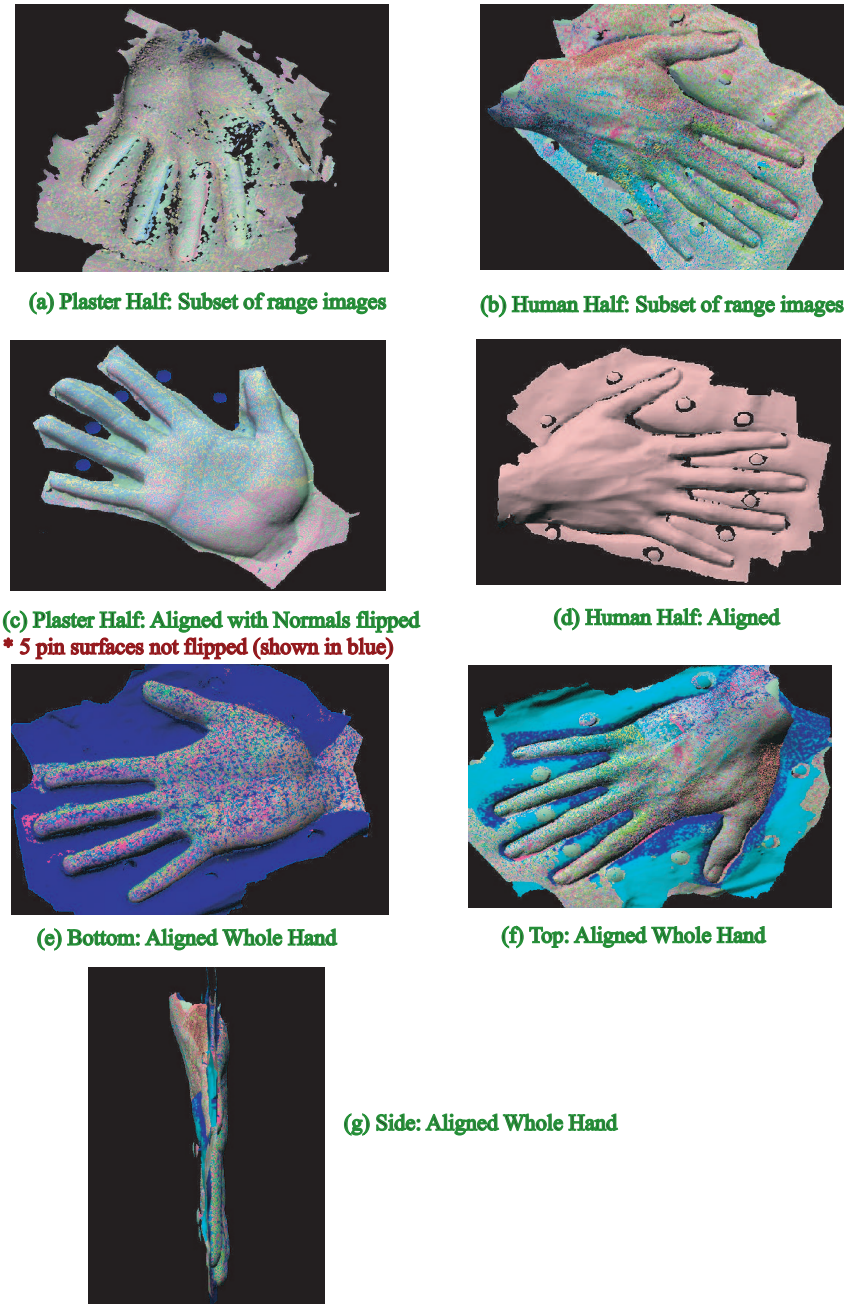
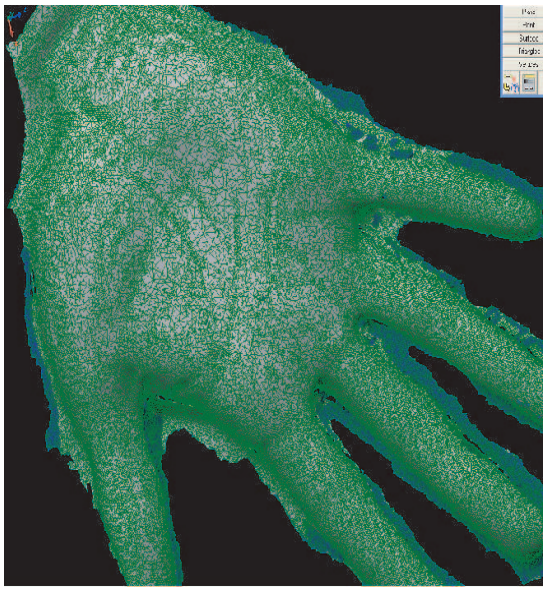


Figure E.1: Alignment Procedure

Appendix F

Meshed Hand after merging

(a) Top part of meshed hand

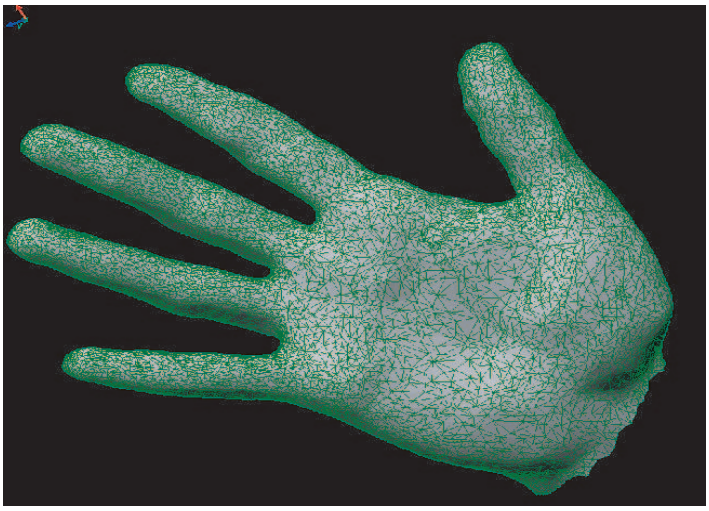


(b) Bottom part of meshed hand

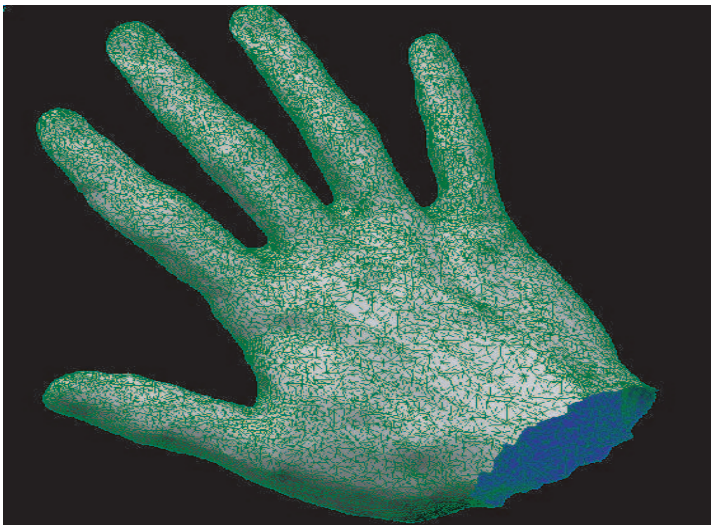
Figure F.1: Meshed Hand obtained after merging in IMMerge.

Appendix G

Reduced Hand Model



(a) Reduced Hand Model: Bottom

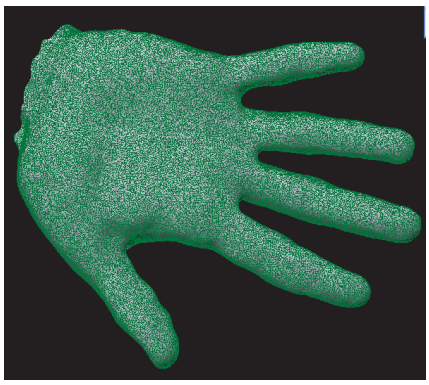


(b) Reduced Hand Model: Top

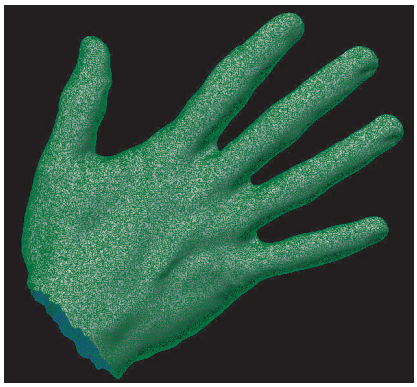
Figure G.1: Reduced hand model with 6894 points and 13, 675 triangles.

Appendix H

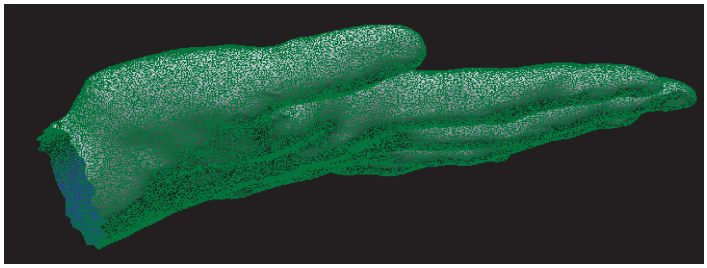
Final Hand Model



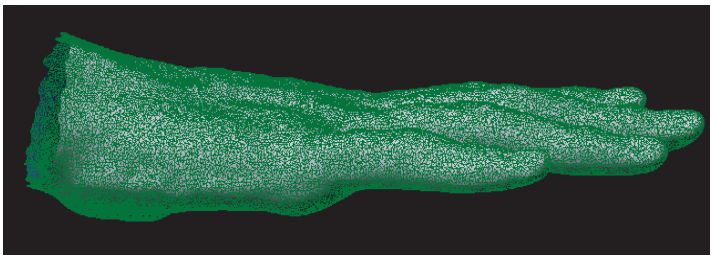
(a) Final Hand: Bottom View



(b) Final Hand: Top View



(c)(i) Final Hand: Side View



(c)(ii) Final Hand: Side View

Figure H.1: Final hand model with 46, 140 points and 92 063 triangles.

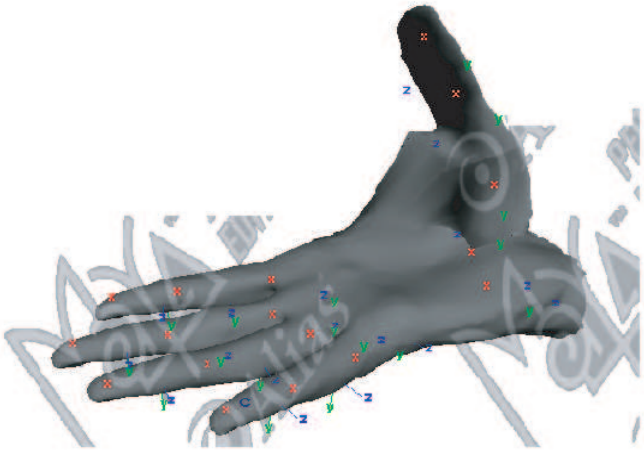
Appendix I

Strategy for painting skin weights

Figure I.1: Strategy for painting skin weights in Maya by grouping of points: Green - Points affected by a max of One joint; Red - Points Affected by a max of Two joints; Yellow: Points affected by a max of Three joints.

Appendix J

Skin Deformation

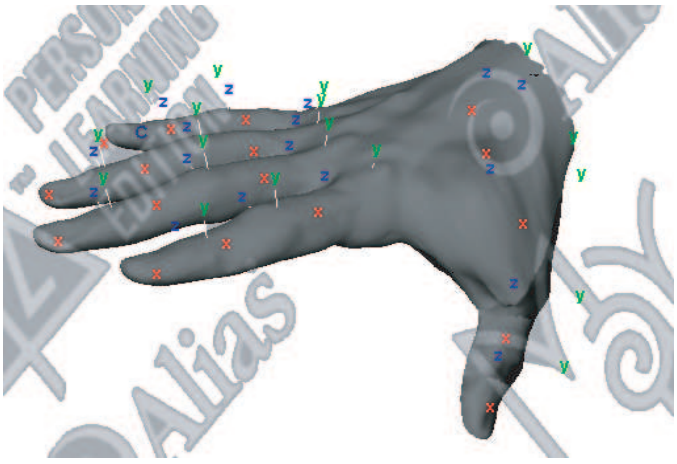


(a) Skin deformation before linear blending: Bottom

(b) Skin deformation before linear blending: Top

Figure J.1: Skin deformation before linear blending.

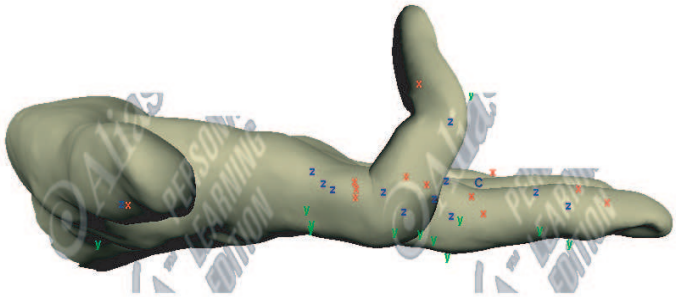
(a) Skin deformation after linear blending : Bottom



(b) Skin deformation after linear blending : Top

Figure J.2: Skin deformation after linear blending.

(a) Index bending before linear blending (shrunken effect observed)



(b) Index bending after linear blending (intersecting surface gives more natural shape)

Figure J.3: Effect of weights on shape of joint

Appendix K

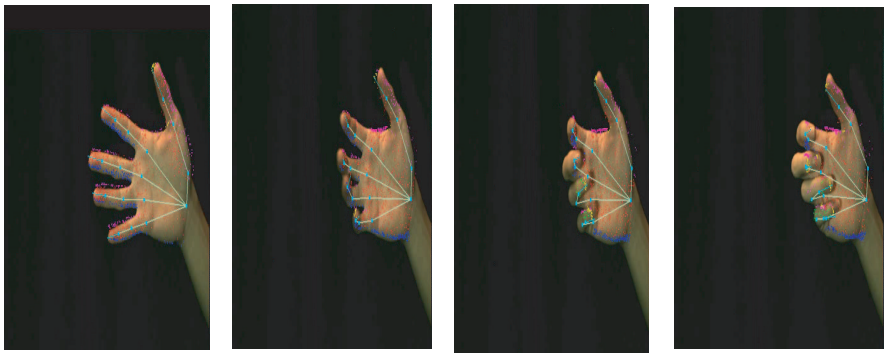
Tracking Results

K.1 Gripping Motion

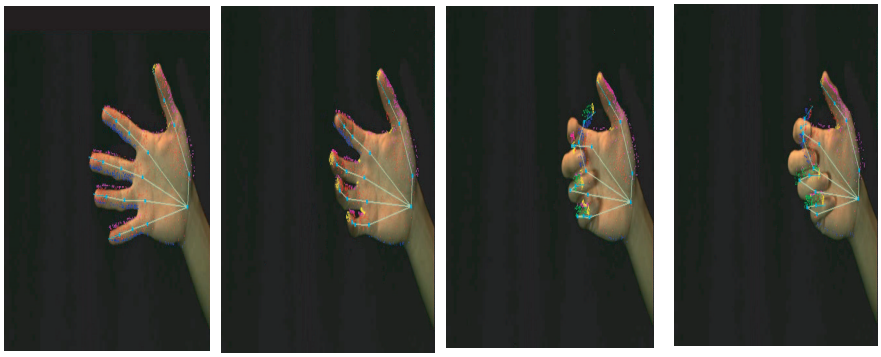
K.2 Spin Motion

K.3 Translate Motion

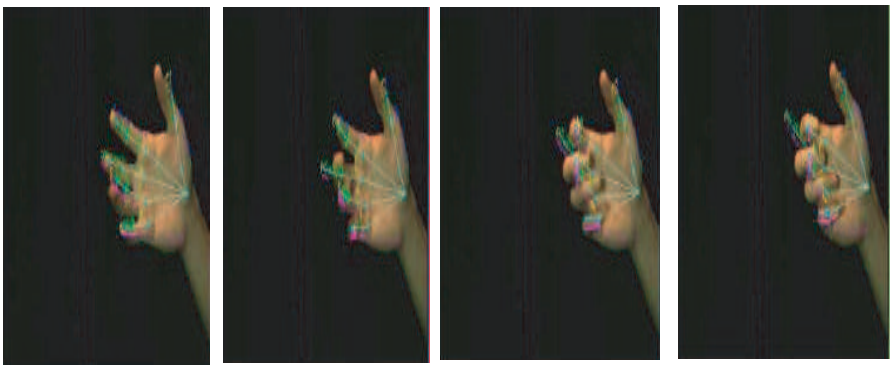
K.4 Thumb CMC Motion



(a) NICTA Hand - Grip (constraints): Tracks well 2 out of 9 cases (good results shown)

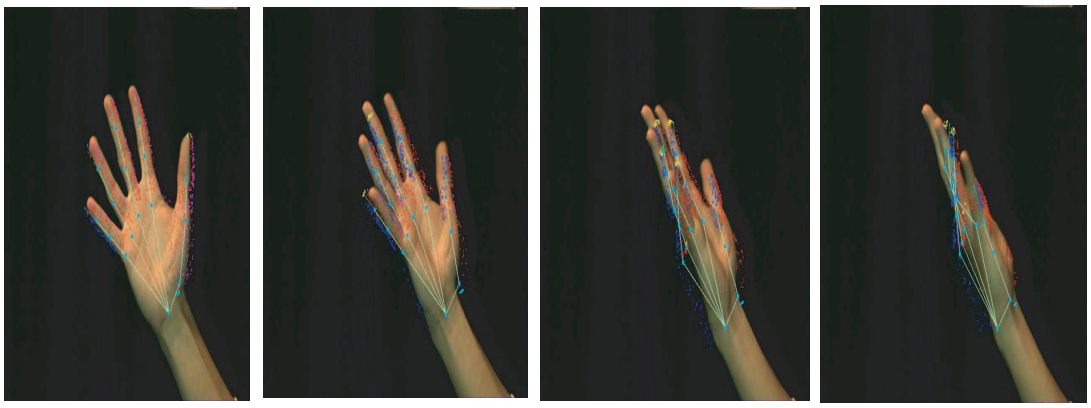


(b) NICTA Hand - Grip (no constraints)

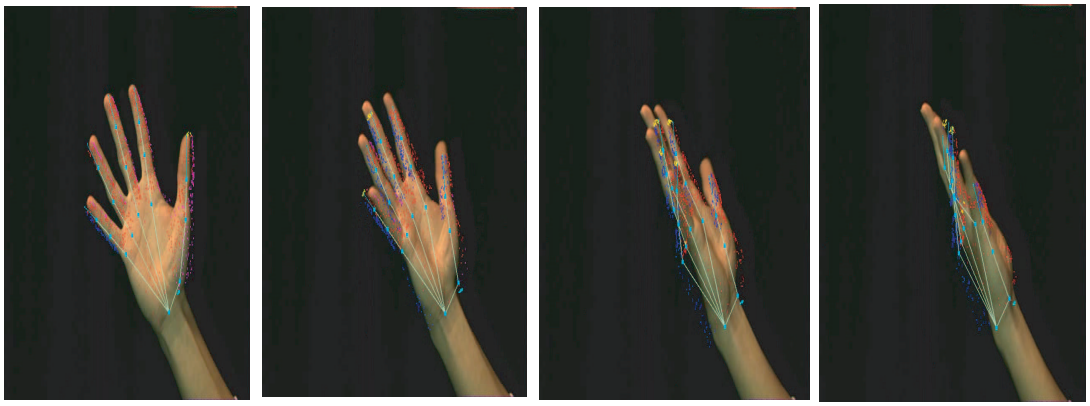


(c) ETH Hand - Grip

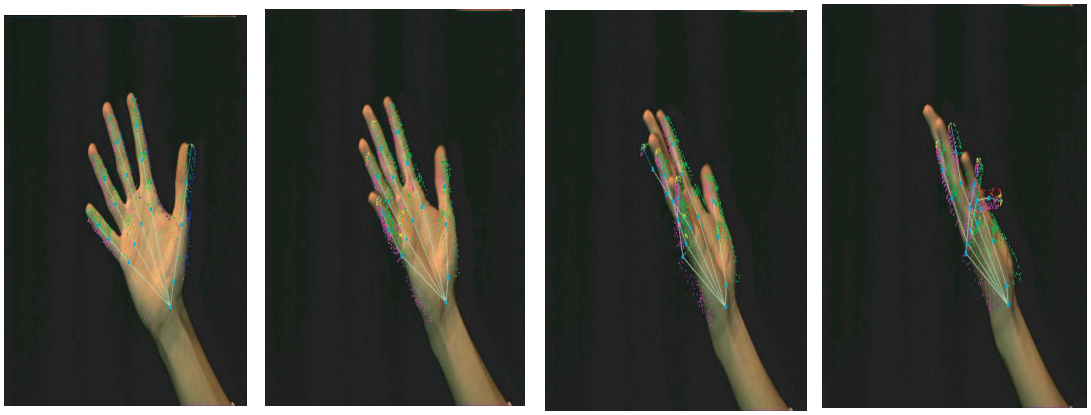
Figure K.1: Grip movement over 23 frames (showing frames 0-10-17-22).



(a) NICTA Hand - Spin (with constraints)

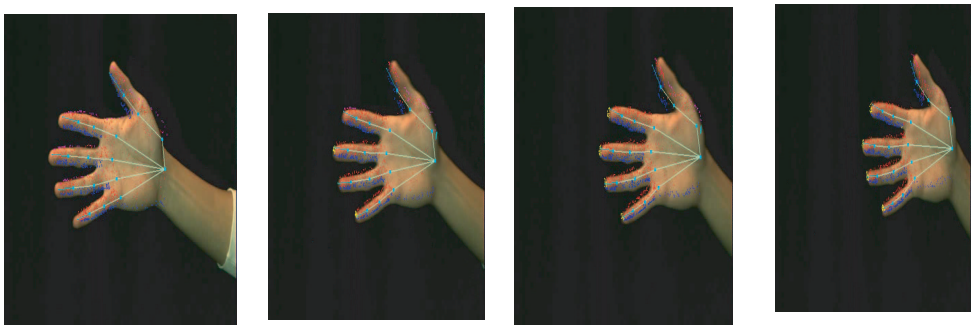


(b) NICTA Hand - Spin (without cocnstraints)

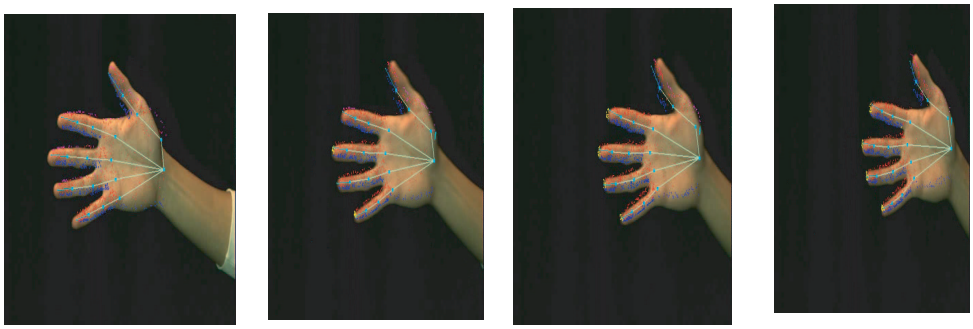


(c) ETH Hand - Spin

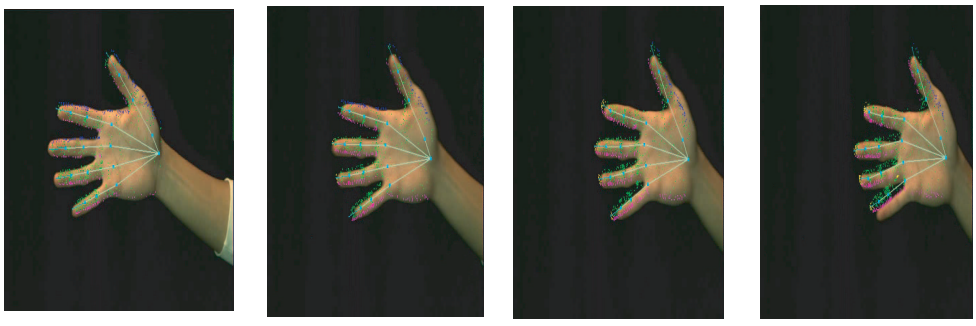
Figure K.2: Spinning movement over 22 frames (showing frames 0-10-15-21).



(a) NICTA Hand - Translate (with constraints)

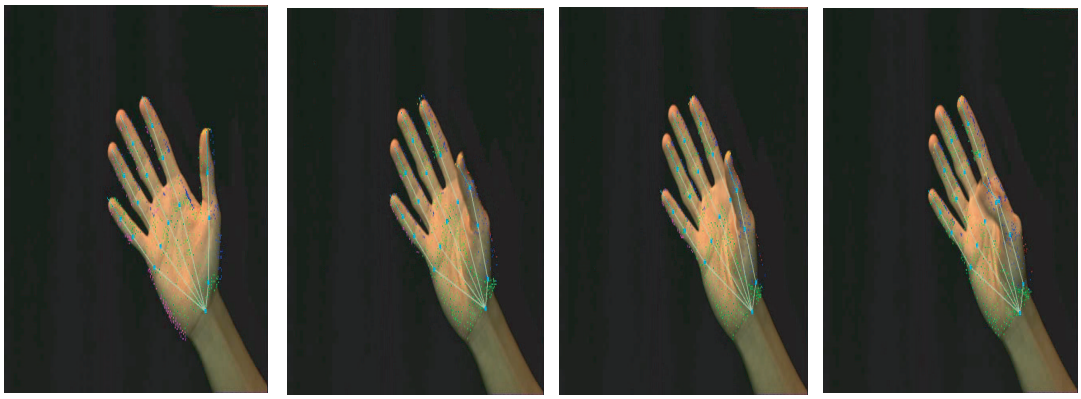


(b) NICTA Hand - Translate (no constraints)

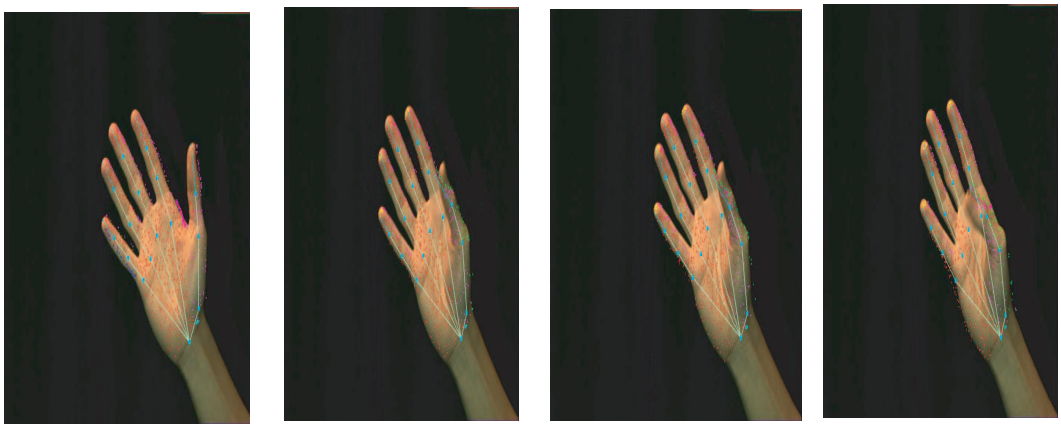


(c) ETH Hand - Translate

Figure K.3: Translation movement over 17 frames (showing frames 0-7-12-16).



(a) ETH Hand



(b) NICTA Hand

Figure K.4: Thumb CMC motion over 37 frames (showing frames 0-20-30-36).

Bibliography

Mischitz, G.E. (2001), *The History of Human Computer Interaction*, viewed 7 August 2006, <http://www2.iicm.edu/cguetl/education/projects/mischitz/Seminar.htm>

The Wikipedia (2006), *Iran Air Flight 655*, viewed 7 August 2006, <http://en.wikipedia.org/wiki/Iran_Air_Flight_655>

Turk, M. & Robertson, G. (2000), *Perceptual User Interfaces*, Communications of the ACM, Vol. 43, No. 3.

Wu, Y. & T.S. (1999), *Human Hand Modeling, Analysis and Animation in the Context of HCI*, IEEE Int'l Conf. on Image Processing, Kobe, Japan.

Zhou, H. (2005), *Visual Tracking and Recognition of the Human Hand*, PhD Thesis, University of Illinois at Urbana-Champaign.

Craig, J.J. (2005),*Introduction to Robotics, Mechanics and Control*, 3rd Edition, Pearson Prentice Hall.

McDonald, J. *et al.* (2001), *An Improved Articulated Model of the Human Hand.*, Proceedings of the 8 th International Conference in Central Europe on Computer Graphics, Visualization and Interactive Digital Media, pp. 306 - 313.

Hollister, A. & Buford, W.L. & Myers, L.M. & Giurintano, D.J. & Novick, A. (1992), *The axes of rotation of the thumb carpometacarpal joint* J Orthop Res 10, pp. 454Ű460.

Bray, M. & Koller-Meier, E. & Muller, P. & Van Gool, L. & Schraudolph, N.N. (2004), *3D Hand Tracking by Rapid Stochastic Gradient Descent using a skinning model*, 1st European Conference on Visual Media Production (CVMP), pp. 59-68.

Baratoff, G. & Blanksteen, S. (1993), *Tracking Devices*, viewed 22 August 2006, <http://www.hitl.washington.edu/scivw/EVE/I.D.1.b.TrackingDevices.html>

Aggarwal, J.K. & Cai, Q. (1997), *Human motion analysis: A review*, IEEE Nonrigid and Articulated Motion Workshop, San Juan, Puerto Rico, pp. 90-102.

Lin, J. & Huang, T.S. (2001), *Modeling Human Hand Constaints*, ARL Federated Laboratory 5th Annual Symposium, pp. 105-110.

Guan, H.Y. & Chua, C.S. & Ho, Y.K. (1999), *Hand Posture Estimation from 2D Monocular Image*, Proceedings, Second International Conference on 3D Digital Imaging and Modeling.

Duffy, N. (1999), *Overview of Appearance Based Methods in Computer Vision*, Technical Report, Available at <https://www.cs.tcd.ie/publications/tech-reports/reports.99/TCD-CS-1999-51.pdf>

Deutscher, J. & Blake, A. & Reid, I. (2000), *Articulated Body Motion Capture by Annealed Particle Filtering*, in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR),pp. 126-133.

Regh, J.M. & Kanade, T. (1995), *Model-based tracking of Self-Occluding Articulated Objects*, in Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 612-617.

Press, W.H. & Flannery, B.P. & Teukolsky, S.A. & Vetterling, W.T. (1988), *Numerical recipes in C*, Cambridge University Press.

Levenberg, K. (1944), *A Method for the Solution of Certain Non-Linear Problems in Least Squares*, Quarterly Journal of Applied Mathematics, II(2), pp. 164-168.

Marquardt, D.W. (1963), *An Algorithm for Least-Squares Estimation of Non-Linear Parameters*, Journal of the Society of Industrial and Applied Mathematics, 11(2),pp. 431-441.

Sminchisescu, C. & Triggs, B. (2002), *Covariance Scaled Sampling for Monocular 3D Body Tracking*, Conference on Computer Vision and Pattern Recognition (CVRP) , pp.447-454.

Sudderth, E.B. & Mandel, M. & Freeman, W.T. & Willsky A.S. (2004), *Visual Hand Tracking Using Nonparametric Belief Propagation*, Conference on Computer Vision and Pattern Recognition (CVRP) Workshop, pp. 189.

Tsap, L.V & Goldgof, D.B & Sarkar, S. (1997), *Human Skin and Hand Motion Analysis from Range Image Sequences Using Nonlinear FEM*, IEEE Proceedings, Nonrigid and Articulated Motion Workshop, pp. 80-88.

Durikovic, R. & Numata, K. (2004), *Human Hand Model based on Rigid Body Dynamics*, IEEE Proceedings, Eight International Conference on Information Visualisation.

Heap, T. & Hogg, D. (2004), *Towards 3D Hand Tracking Using a Deformable Model*, Proceedings, International Conference on Automatic Face and Gesture Recognition, Killington, Vt., pp. 140-145.

Lee, J. & Kosiyasu (1995), *Model-Based Analysis of Hand Posture*, IEEE Computer Graphics and Applications, Vol. 15, Issue 5, pp. 77-86.

Rijpkema, H. & Girard, M. (1991), *Computer Animation of Knowledge-Based Human Grasping*, Computer Graphics, Vol. 25, no. 4, pp. 339-348.

Landsmeer, J. (1955), *Anatomical and functional investigations on the articulation of the human fingers*, Acta Anatomica 24 supl 2, 25, pp. 1-69.

Landsmeer, J. (1976), *Atlas of Anatomy of the Hand*, Churchill Livingstone, Edinburgh, New York.

WCSN, Wcape School (1997), *The Human Skeleton*, viewed 25 August 2006, <http://www.wcape.school.za/subject/biology/skeleton/hand.gif>

Netter, F. (1987), *The CIBA collection of medical illustrations*, vol. 8, Musculoskeletal system, Ciba-geigy, Tom's River, New Jersey.

An, K.N. & Chao, E.Y. & Cooney III, W.P. & Linscheid, R.L. (1979), *Normative Model of Human Hand for Biomechanical Analysis*, J. Biomechanics, 12,10,pp. 775-788.

An, K.N. & Chao, E.Y. & Cooney III, W.P. & Linscheid, R.L. (1985), *Forces in the Normal and Abnormal Hand*, Journal of Ortheopaedic research, vol. 3, pp. 202-211.

Miller, A. & Allen, P. & Santos, V. & Valero-Cuevas, F. (2004), *From robotic hands to human hands: a visualization and simulation engine for grasping research*, Industrial Robot: An International Journal, Vol. 32, No. 1, pp. 55-63.

Valero-Cuevas, F.J. & Zajac, F.E. & Burgar, C.G. (1998), *Large index-fingertip forces are produced by subject-independent patterns of muscle excitation*, J. Biomech., Vol. 31, pp. 693-703.

Valero-Cuevas, F. & Johanson, M.E. & Towles, J.D. (2003), *Towards a realistic biomechanical model of the thumb: The choice of kinematic description is more critical than the solution method or the variability/uncertainty of musculoskeletal parameters*, J. Biomech., Vol. 36, No. 7, pp. 1019-30.

Sancho-Bru, J.L. & Perez-Gonzalez, A.& Vergara-Monedero, M. & Giurintano, D. (2001), *A 3-d dynamic model of human finger for studying free movements*, J. Biomech.,Vol. 34, pp. 1491-500.

Shadmehr, R .& Wise, P. (2005), Supplementary documents for the *Computational Neurobiology of Reaching and Pointing*, MIT Press, Available at <http://www.bme.jhu.edu/ reza/book/force_and_torque/forcetotorque.htm>

Huang, Z. (2005), *Motion Control for Human Animation*, PhD Thesis, Swiss Federal Institute of Technology, Lausanne (EPFL).

Albrecht, I. & Haber, J. & Seidel, H.P. (2003), *Construction and Animation of Anatomically Based Human Hand Models*, Eurographics/SIGGRAPH Symposium on Computer Animation.

Tsang, W. & Singh, K. & Fiume, E. (2003), *Helping Hand: An Anatomically Accurate Inverse Dynamics Solution For Unconstrained Hand Motion*, Eurographics/ACM SIGGRAPH Symposium on Computer Animation, pp. 1Ű10.

Bernardini, F. & Rushmeier, H. (2002), *The 3D Model Acquisition Pipeline*, Computer Graphics Forum, Vol. 21, no. 2, pp. 149-172.

Lewis, J.P. & Cordner, M. & Fong, N. (2000), *A Pose space deformations: A unified approach to shape interpolation and skeleton-driven deformation*, Annual Conference Series, ACM SIGGRAPH, pp. 165-172.

The Wikipedia (2006), *3D Scanner*, viewed 29 August 2006, <http://en.wikipedia.org/wiki/3D_scanner>

Besl, P.J. & McKay, N.D. (1992), *A method for registration of 3D shapes*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Issue 14, no. 2, pp. 239-256.

Quarl Web Page, (2006), *Scene Description Language Manual*, last accessed 3 Spetember 2006, <http://www.qarl.com/menu/class/cs323_fl98/mrmanual_2.0/scene.html>

Faro Technologies Web site, last accessed 3 September 2006, <http://www.faro.com>

Becker, J.C & Thankor, N.V (2006), *A Study of the Range of Motion of Human Fingers with Application to Anthropomorphic Designs*, IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING. VOL. 35. NO. 2, pp. 110-118.

Armstrong, T. & Chaffin D., (1977), *An investigation of the relationship between displacements of the finger and wrist joints and the extrinsic finger flexor tendons*, J. Biomechanics, Bol. 11, pp. 119-128.

Brand, P.W. & Beach, R.B. & Thompson, D.E. (1981), *Relative tension and potential excursion of muscles in the forearm and hand*, J. Hand surgery, Vol. 6, Issue 3, pp. 209-219.

Kuch, J. & Huang, T. (1995), *Vision Based Hand Modeling and Tracking for Virtual Teleconferencing and Telecollaboration*, IEEE Transaction on Computer Vision, vol. 8, pp. 666-671.