DATA-BASED MODELS FOR DEFORMABLE OBJECTS: SENSING, ACQUISITION, AND INTERACTIVE PLAYBACK

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Philip Fong
June 2007

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Jean-Claude Latombe) Principal Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Kenneth Salisbury)

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

(Oussama Khatib)

Approved for the University Committee on Graduate Studies.

Abstract

Many haptics-enabled virtual environments require the inclusion of deformable solids for increased realism. For example, some medical procedure simulations require deformable organs. The feel of an organ can indicate the location of critical underlying structures or the presence of disease. The models currently used in these simulations tend to be created by hand in a time consuming manner and/or are based on expensive computational methods (e.g. finite elements).

This thesis describes the design and implementation of an automated system to build data-based models of elastic deformable objects and to render these models in an interactive virtual environment. By automating model creation, more realistic models in greater numbers can be included in these interactive simulations.

The system consists of two components: a data acquisition system and a model creation and rendering system. In the data acquisition system, the geometry of an object is sensed by a novel structured-light sensor. The sensor is designed to recover depth from single images which allows it to work on fast moving and deforming objects. The behavior of the object and haptic information is captured through active probing using a haptic device as a robot. Techniques for contact detection, slip detection on deformable surfaces, and enhanced open-loop force control enable the robot to explore the force field at various contact locations on the object's surface.

In the model creation and rendering system, measurements are transformed into a data-based model and efficiently rendered in an interactive haptic environment. Discrete samples are interpolated into a continuous force field, and friction

properties are extracted. The running time of a rendering loop iteration scales logarithmically to large data sets and is empirically shown to be less than 50 µs for test data sets with measurements from about one hundred contact locations.

The system has been implemented on hardware readily available to haptics researchers and demonstrated with nonlinear inhomogeneous test objects.

Acknowledgements

I have been fortunate to have many people help me in my journey through graduate school.

First, Jean-Claude Latombe has been a wonderful adviser. I thank him for giving me so much freedom in the direction of my work and for his advice and support. Not only have I benefited for his technical knowledge, but he also has taught me a lot about writing and presenting. I admire his standards in what he expects of himself and others. He is truly a model of how to be a researcher and faculty member.

Ken Salisbury has also been instrumental in my journey. His knowledge and ideas about haptics and its applications are invaluable. In many ways, he is the perfect complement to Jean-Claude Latombe.

Oussama Khatib, my third reader, has asked many insightful questions and pushed me to think about topics that I had not considered. Those discussions have made this thesis much more complete.

The Surgsim Group has provided many helpful discussions. Through it I met Nik Blevins who served on my defense committee and gave me many insights on medical applications.

I thank Günter Niemeyer for being the chair of my defense committee even though I had met him only briefly once before asking him to be chair.

I am also grateful to all my fellow students who have enriched my time here both in the lab and out of it. Florian Buron worked with me on the range sensor. Chris Sewell is a great friend and has answered so many of my questions. Dan Morris is always an exceptional technical resource and an incredibly enthusiastic

intramural sports team leader. It has been a pleasure to share an office with Mitul Saha. Sean Walker helped me take measurements using his force-torque sensor setup. I would not have been able to laser cut the box for the range sensor without Keenan Wyrobek, and Justin Durack worked with me on developing a Bayesian classifier for the range sensor.

I also thank other collaborators including James Davis who gave me critical assistance with building the range sensor, Micheal Wand for his work on deformable geometry processing, and Vincent Hayward for tolerating my many questions.

This work was financially supported by NIH grant R33 LM07295 and NSF grant ACI-0205671. My first year was supported by a Stanford School of Engineering Fellowship. I could not have navigated the administrative labyrinth without the CS department and AI Lab staff, especially Liliana Rivera.

Sebastian Thrun and the Stanford Racing Team gave me an opportunity of a lifetime. I will never forget working on Stanley.

My family, especially my parents, have been very supportive of me. I thank them for understanding why I traveled across the country for graduate school.

Robert Pavlica who passed away this January deserves mention as well. He founded and ran the Authentic Science Research program at my high school through which I had my first significant experience with scientific research.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

A virtual environment strives to be an extension of the real world by allowing a user to interact with it in the same manner he would interact with the real world. The user perceives the virtual world through his senses and performs actions through touch. A typical virtual world has sound and graphics. Haptic environments add touch for both feedback and input. In current virtual worlds, smell and taste are rarely simulated.

Virtual environments are used for many purposes ranging from entertainment to education. Many of them are simulations in that they attempt to replicate an experience in the way that it would occur in the real world. The application that is of particular interest and that serves as the motivating application for the research presented in this thesis is real-time interactive simulation of medical procedures and surgery.

Using simulation for surgical training is becoming increasingly accepted [26]. Surgery is highly variable and errors are potentially catastrophic. Simulation allows someone to gain experience and learn without putting himself or a patient at risk, and a simulator is more readily available and less costly than animal models. A computerized simulator also has capabilities that are not available with other training means. Since it has complete knowledge of what occurs within the virtual environment, it can provide built-in feedback and metrics allowing a user to learn independently of an instructor [52].

Deformable models are an important component needed to build these simulations. The sense of touch conveys important information to a doctor. It is used for diagnosis of disease and for locating structures within the body or organs. During a physical exam, it is common practice to palpate various parts of the body. In the neck, lymph nodes and the thyroid gland are located through touch, and the feel of these structures can indicate the presence or absence of disease [23]. During lung surgery, palpation can find metastases that were not visible in pre-operative imaging [44]. In the liver, harder areas indicate scarring and disease [16].

1.1 Challenges in Deformable Modeling

The first challenge in modeling deformable objects is the inherent complexity of a deformable object. Compared to a rigid object, the description of a deformable object must be richer. A deformable object has more complex behavior. Interacting with it causes it to change shape and the reaction forces felt by the user can be nonlinear and vary over the object. As a consequence, the creation of deformable object models is correspondingly more complex.

The second challenge comes from our application. In an interactive environment, the computation needed for the simulation must occur in real time. For haptic feedback, the typical update loop rate is 1000 Hz. We must be able to compute the behavior and response of an object in less than a millisecond. Combined with the first challenge, this means we must be able to simulate a complicated object quickly.

1.2 Research Contributions

This thesis addresses these challenges through the design and implementation of an automated system to build models of elastic deformable objects from measurements of real objects and to render these models in a virtual world. Automating the creation process will allow more deformable object models and

more realistic models to be included in interactive simulations. In contrast, manually constructing models is a time consuming and tedious process.

The contributions of this thesis are the following:

Data Acquisition A framework and techniques to automatically collect the information needed to describe the behavior of real elastic deformable objects.

The data acquisition system consists of a component to sense geometry and a component to collect behavior and haptic information. These are implemented on hardware readily available to haptics researchers and users. Geometry is captured using a novel structured-light range sensor which is constructed from a projector and camera. It uses a pattern designed to enable depth extraction from single images allowing it to capture the geometry of fast moving and deforming objects. An active probing system using a haptic display device acting as a robotic manipulator captures force and friction information from objects. Previous research has not dealt with capturing friction behavior over a deformable object.

Model Creation Algorithms to build a data-based model from measurements.

The measurements must be organized into a model to allow the efficient computation of responses during playback. This process must account for noise present in the measurements and the playback constraints.

Model Rendering Algorithms to synthesize responses to user interaction in the virtual environment.

Previous research has demonstrated the synthesis of responses from a deformable object using artificial data and geometry [37]. This thesis develops rendering techniques for the models created from measurements, examines more closely the motion of contact points on a deformable surface, and develops new techniques to compute this motion at runtime.

Chapter 2

System Design Overview

There are two aspects to modeling objects for virtual environments, model creation and model rendering. Model creation is describing a particular object in the language of the model while model rendering generates the representation of the object in the virtual world in response to a user's interaction, based on the description. For this work, we focus on models with graphics and haptics. Accordingly, there are components that capture the geometry and feel of an object and components which transform the captured data into graphics and forces in the virtual environment.

2.1 Modeling Approaches

A modeling approach can be broadly characterized as parametric or data-based depending on the relative complexity of the data it contains and how it synthesizes new behaviors.

Parametric approaches have an explicit underlying model (e.g., in the form of equations) whose behavior is controlled by a collection of parameters. In the virtual environment, the parameters are transformed into behaviors by dynamically simulating the model. The complexity of the behaviors the model can produce is determined by the complexity of the model. For example, a linear model cannot exhibit non-linear responses. The power of a parametric model is its ability to

extend behavior to new situations. For example, such models can often generate forces for varying numbers of contact points.

In contrast, a data-based approach does not contain an explicit model. The behavior of the model is specified by the collection of response examples stored in it. The response examples are usually more complex than the parameters in a parametric model. This data can be generated through simulation, analytical specification, or measurement of real objects. However, the runtime synthesis techniques are much simpler. In the virtual environment, responses are generated through blending and interpolating the stored examples. This limits the model's ability to generalize behavior to new situations. For example, it usually cannot generate responses to two contact points if all the stored examples only had one contact point.

2.2 Related Work

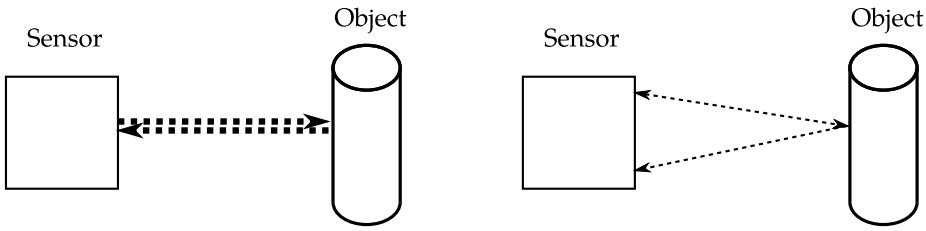
Work from which we can draw upon for this thesis can be grouped into two areas: capturing and rendering the geometry and appearance of objects; and capturing and rendering the behavior of objects. Capturing and rendering geometry involves only graphics while capturing and rendering behavior has both haptic and graphic aspects.

2.2.1 Geometry Capture and Rendering

There is extensive work on building visual models of real world objects. We are interested in techniques to sense the geometry of a physical object. In the discussion below, I divide sensing methods to acquire geometry into two categories: those based on time of flight and those based on triangulation.

2.2.1.1 Time-of-Flight Sensors

Time-of-flight sensors transmit energy such as light (LIDAR and shaped light pulse), sound (SONAR), or radio waves (RADAR). Distances from the transmission



(a) Time of flight (b) Triangulation

Figure 2.1: Geometry sensor layouts

point to points on target objects which reflect the energy are computed from the time it takes for the energy to travel to and from the object (Figure 2.1(a)). RADAR, SONAR, and LIDAR typically only transmit and receive energy along a narrow beam and only give distances to a point or small number of points per measurement. To capture the geometry of an object, the beam must be scanned over the object. This makes them unsuitable for moving or deforming objects since the object will move or change shape during the scanning.

Shaped light pulse systems do not need to be scanned over the object. They give a 2D array of points similar to an image and work at relatively high frame rates. However, the resolution in the depth direction tends to be limited [24].

Because they need to transmit specialized energy and make precise time measurements, these systems all require specialized hardware.

2.2.1.2 Triangulation Sensors

In triangulation-based systems, corresponding features on the surface of an object in different viewpoints are found and their 3D positions are computed by intersecting rays from each viewpoint (Figure 2.1(b)). One of the rays is light received from the object by the sensor and the other is either received or transmitted. The various methods differ in how the correspondences are determined.

Stereo vision takes images from two different viewpoints and searches for correspondences in the image [60]. It has the advantage of being passive. Only

observation is needed; no energy is transmitted. However, this process is computationally intensive and works poorly if there is no texture on the surface to match or if the texture repeats causing many possible matches. It does give dense range maps over a 2D area and has been successfully applied to moving and deforming textured cloth [45].

A structured-light system transmits a pattern from one of the two viewpoints. Laser scan systems project a single plane of light onto the object. In an image taken from a properly aligned but different viewpoint, the stripe of light will appear in each scanline only once. The correspondence problem is trivial but the laser stripe must be scanned across the object (thus the name). Again, this is not suitable for moving or deforming objects.

Two-dimensional pattern-based structured-light systems can produce dense range maps over an area [5, 6, 31, 34, 49, 50, 56, 57, 63]. The correspondences are found by locating spatially and/or temporally unique features encoded into the projected pattern. For example, a pattern can be composed of a non-repeating sequence of color stripes. The edges between the stripes are spatially unique features. In methods that only rely on single images, range maps can be produced at the video frame rate (in real-time or post-processed) [5, 49, 56, 57]. Multi-image techniques typically give higher quality range maps, but rely on temporal coherence limiting the speed of motions in the scene and often producing range maps at a rate lower than the video frame rate. This limits their use to rigid and slowly moving objects.

Pattern-based techniques differ in the way information is encoded into the pattern. Intensity ramp patterns [5] are sensitive to noise and reflectance variations. Techniques using Moiré gratings and sinusoids [49, 56, 57] are much more resistant to noise and reflectance variations, but require both knowledge of the topology and points of known depth in the scene. In systems using black and white stripes [31, 50] or colored stripes [6, 34, 63] the resolution of the range map obtained from a single image is limited by the discrete number of possible pattern encodings.

Spacetime stereo [13] is a method that uses unstructured light. It performs

stereo matching over space and time windows on scenes illuminated by arbitrarily varying patterns. This allows it to work on moving and deforming objects, but the motion speed is limited by the length of the time window.

2.2.1.3 Geometry Rendering

The sensing techniques described above produce a collection of point samples on the surface of objects. The point cloud is usually transformed into a triangle mesh for graphical rendering via standard mesh rendering techniques. Laser scanning, stereo vision, and spacetime stereo can additionally capture images of the scene which can be used to associate color with the point samples or as a texture map for the resulting mesh [13]. The simplest approach is to create a triangulation of the points by connecting a point with its neighbors as determined by the sensor layout. For example, a point corresponding to a pixel in the camera of a structured light, stereo vision, or shaped light pulse system would be connected to points corresponding to the neighboring pixels.

More advanced and complex procedures can fill holes, align, and merge multiple depth maps of the same object [10]. This creates a model that has more complete geometry. A single depth map will only show the geometry of one side and will have holes wherever there is occlusion. Multiple depth maps from different viewpoints will reduce occlusion and the merging algorithm can also guess to fill in holes [12]. Unfortunately, most of these techniques do not work with deformable objects because aligning the depth maps depends on the object being rigid. A recent advance in geometry processing is able to extract correspondences and merge point clouds of deforming objects [61]. However, the size of the dataset that can be processed is limited.

2.2.1.4 Other Techniques

It is also possible to create geometric models without directly observing the surface of an object. For example, volume or surface models can be extracted from medical imaging data such as magnetic resonance imaging (MRI) or computer

aided tomography (CAT) scans. However, segmenting the object of interest from the data is sometimes difficult [9, 38, 52].

2.2.2 Behavior Capture and Rendering

The behavior of an object is how the object interacts with and responds to other objects. In particular, in the virtual environment we need to know what happens when objects come into contact with each other. For example, the user could press a virtual tool against the object.

A deformable object is much more complex than a rigid object. The behavior of a rigid object can be characterized by relatively simple surface properties such as friction while a deformable object has additional properties that depend on its innards. A deformable object's behavior includes how it changes shape (deforms) and what response forces it exerts. The response forces can be nonlinear and vary throughout the object (inhomogeneity).

2.2.2.1 Parametric Models

Parametric approaches can be roughly categorized as constitutive or nonconstitutive. A constitutive model has structure and parameters which are based on real physical constants and quantities. Typically the volume of the object is decomposed into elements, and differential equations describe the behavior of the elements in terms of physical constants such as Young's Modulus and Poisson's Ratio. Techniques such as finite element models (FEM) and boundary element models (BEM) are often used as analysis models, and there are many interactive models based on simplifications of them or reductions of their evaluation time by precomputation [21]. One example is the ArtDefo method which precomputes a set of system responses to a BEM for linear homogeneous objects. The stored solutions are recombined and updated to generate responses in the virtual world [29].

In contrast, the parameters of a nonconstitutive model are not directly related to physical quantities and constants. They are generally simpler and as

a result less physically accurate but run faster than constitutive models. Mass-spring models decompose an object into point masses connected by springs and dampers. The parameters of the model are the masses, spring constants, and damping constants. In [8], a variation of the mass-spring concept is shown modeling organs. Their technique uses space filling spheres which have mass and are connected by springs and dampers. The surface is represented by nodes connected to each other and the spheres. There are also methods which model the deformation behavior based on a resistance to change in geometry. In [58] and [59], the object changes shape to minimize an energy function. Terms of the energy function increase with changes in the geometry from the rest configuration. The energy function of [59] has terms related to changes in the distance between nodes in the model, the surface area of elements composed of the nodes, and the volume of the elements . The parameters of the model are the coefficients of the terms.

To create models of specific objects, the parameters are often set by hand. A modeler can iteratively adjust parameters and observe the results until he is satisfied. In constitutive models, since the parameters represent physical constants, the influence of the parameters is usually straightforward. Good starting points for the parameters can be found in references for material property constants. For nonconstitutive modes, this process is more difficult since the parameters are only indirectly related to physical properties. To address this, Morris developed a procedure to set the parameters of a nonconstitutive model based on an FEM simulation [38]. In either case, manual modeling can be a long and tedious process. A complex model can have a large number of parameters. For example, in mass-spring models each mass, spring constant, and damping constant can be controlled.

Parameters can also be adjusted based on measurements of real objects. Cotin et al. matched the behavior of their model to force displacement measurements of a cylindrical sample of brain tissue [9]. Similar force-displacement measurements were used to set the parameters of a tumor palpation simulator [16]. The Tissue Material Sampling Tool (TeMPeST) is a device that measures linear visco-elastic

properties from in vivo tissues. It applies small one-dimensional deflections over a range of frequencies relevant to haptic feedback in surgical simulation. Material properties extracted are from the measurements [42].

In [43], the parameters of an ArtDefo model are set with an automated robotic system. An industrial robot arm equipped with a force sensor actively probed an object and made measurements while stereo vision recorded the geometry. Using the same hardware setup, [43] and [35] also captured the friction properties of rigid objects.

For inhomogeneous objects, the idea of haptically scanning an object by probing it at various points was proposed in [48]. A haptic display device was used as a robot to capture textures from rigid surfaces. The forces needed to move the end effector across a textured surface were used to estimate instantaneous friction coefficients which were described by a stochastic model. Measurements taken in a similar manner while moving the end effector through soil were also used to set parameters in a model of soil behavior [25].

2.2.2.2 Data-based Approaches

In the event-based haptics framework of [32], a hybrid data-based and parametric model was used. The realism of tapping on rigid objects is improved by overlaying transients on the standard spring restorative force used in rigid object haptic rendering. The transients are generated from acceleration profiles recorded while a person tapped on an actual object using the haptic device.

Motion capture is a popular technique to record the motion of real objects. It is most often applied to articulated bodies such as humans. A collection of markers are placed on the object of interest, and the motion capture system records their 3D trajectories. The markers are specially designed so that an external sensor can track them optically, magnetically, or sonically; or each marker may have internal inertial sensors [27].

A relatively simple way to use motion capture data is to animate a character by having the data control its motion. The joint angles of the articulated body are extracted from the data and used to set the character's joint angles. The

result can show subtle elements of a performer's style that can be very difficult to reproduce with hand animation [41]. By blending various motion sequences, the playback can be interactive. For example, a video game can show a character running and jumping in response to the player's inputs. In [33], a control policy developed with reinforcement learning blends segments of motion capture data to animate a boxer in response to a moving target.

The mesh inverse kinematics (MeshIK) technique synthesizes the geometry of a deformed object based on a collection of sample geometry of the object deformed in various ways [14, 55]. However, it is purely based on geometry and does not generate forces for haptic feedback.

James et al. [28] stored precomputed impulse response functions (IRF) which describe the motion of an object's vertices after an impulse followed by a continuous force were applied. In the virtual world, when a user applies similar forces, the closest IRF to the current object state is played back. To prevent jumping artifacts, the current object state is blended with the selected IRF by an exponential decay.

Mahvash and Hayward [37] described an interactive quasi-static deformable object model for haptics that uses stored data. The forces felt by the user are a force field $\vec{f}(\vec{c}, \vec{\delta})$ that is a function of the initial contact position $\vec{c}$ with the undeformed object and the displacement of the tool location $\vec{x}$ from the contact, i.e. $\vec{\delta} = \vec{x} - \vec{c}$. The model was demonstrated using an analytic force field and synthetic geometry. They focused on the haptics and did not address how to generate graphics.

2.3 System Design

There are differences in the nature of force feedback and visual feedback. Force feedback is a local phenomena while visual feedback is global. Force feedback provides information about the internals of an object while its appearance is only dependent on the surface. In the type of virtual environments we are considering, the user only feels forces at the point of interaction while he can

see the whole side of the object. This difference is seen in sensing methods for capturing data from real objects as well. Sensors for capturing geometry or appearance give information over a whole side of an object while force sensors only give information about the point of interaction.

The nature of force sensing and the fact that we are interested in the behavior of an object when a user interacts with it means the capture process must be active. We must apply a range of forces over the surface and observe the results in order to capture the behavior of the object. It is not possible to only observe without action.

Both the model creation and model rendering components must be designed with these characteristics in mind. In our system, the model creation component captures the graphical and haptic behavior of an object through active measurement and exploration. In addition to giving information about the geometry of the object while it is deformed, geometry sensing guides the probing process. In model rendering, visual feedback comes from the geometry. The geometry also provides structure necessary to turn the measurements into a model that synthesizes response forces from the captured forces.

2.3.1 Geometry Capture and Rendering

Geometry capture is an important component of the data acquisition system. It is used to guide the force capture process, and geometry is needed for the graphical representation of the object in the virtual environment. As discussed in Section 2.2.1, sensors which need to be scanned over an object are unsuitable because we are interested in deforming objects.

Our geometry capture process is a novel structured-light range sensor. Many previous structured-light approaches traded temporal resolution for spatial resolution by assuming temporal coherence and using a sequence of patterns. Our technique imposes no temporal coherence restriction and works with moving and deforming objects.

Using a sensing method that requires line-of-sight means we can only capture

one side of an object at a time. However, we could perform data collections from different orientations and merge them into one model. The line-of-sight requirement also means we can only capture exposed objects. However, we need to probe the object which also requires accessibility. It is also the case that in many situations natural interactions only occur from one side, such as pressing on objects on a table or organs which have only been exposed on one side.

For appearance rendering, we render surfaces extracted from depth maps captured by the range sensor. This simple technique is sufficient to demonstrate our data-based model. Section 5.6.2 discusses some more advanced alternatives.

2.3.2 Behavior Capture and Modeling

We restrict ourselves to capturing and rendering elastic objects. The elastic nature of the objects is an essential quality that allows collecting data through active probing. Since we are interested in how the object responds to various forces, application of various forces is required. Elastic objects return to the same equilibrium state whenever no force is applied. This makes subsequent applications of forces independent of previous probes. Without this property, the state of the object would be changed by the probing process and it would be much more difficult or impossible to capture the complete behavior. We would need to design and apply a series of complex manipulations that accounts for the changes in object state.

Although elastic deformable objects only make up a portion of the universe of deformable objects, it includes objects that are useful in surgical or medical procedure simulations. Many organs can be modeled as inhomogeneous nonlinear elastic deformable objects for the interactions in which we are interested. In many cases, if applying a force causes plastic deformation or other permanent state change, it probably also causes injury.

In interactive haptic environments, the limited computation time available and the complex behavior desired from deformable models favors data-based modeling. The underlying model of a parametric model must be sufficiently simple

that it can be simulated in real-time. Alternatively, its solutions must be wholly or partially precomputed and stored which yields an approach that is more similar to a data-based approach. The interpolation of a data-based model can easily happen within the time limits, and complex behavior can be obtained as long as we store sufficiently complex response examples. As computers become faster it will be possible to do more computation in the same amount of time. This will enable more complex parametric models to be used. However, these computers are also likely to have larger storage capabilities allowing more data to be stored in a data-based model.

As discussed in Section 2.2.2.2, data-based modeling has been successfully applied to rigid objects. We are interested in applying it to deformable objects. For our behavior model, we developed a data-based technique that extends the model in [37] to use data from measurements. In contrast to most interactive parametric models, this will enable our model to represent nonlinear and inhomogeneous objects.

Hand modeling of a complex object is tedious and time consuming. Using measurements of real objects for our data source will be easier on the modeler and more realistic.

In designing a data-based model for haptics, it is worth examining the successful application of this technique to visual modeling. Light field capture records the appearance of an object and image based rendering generates new views of objects by interpolating the appearance [53].

The plenoptic function describes all the rays of light passing through points in space in all directions over time for all wavelengths. A *light field* originally referred to the four dimensional slice of the plenoptic function of all the rays passing through a bounding box where time is constant and all wavelengths of light behave the same, but is sometimes used to refer more generally to the plenoptic function.

Light field capture directly samples the plenoptic function by taking multiple images from different locations and in different directions. If a sequence of images from a single moving camera is used, the object must be static. An array

of cameras or a camera with a microlens array must be used if there is motion or deformation in the scene [40, 62].

Image based rendering techniques synthesize new views of the object from the light field without explicitly modeling the geometry or any object surface properties [53]. Since the synthesis is done from captured appearances (images), the results are photo-realistic and can show complex geometry and lighting effects. In contrast, the results of conventional graphic rendering are limited by the model geometry and the complexity of the model describing the interaction of light with the geometry and surface properties.

We can define a haptic force field function $\vec{F} = \vec{f}(\vec{c}_{0...t}, \vec{d}_{0...t}, \vec{x}_{0...t}, \vec{\phi}_{0...t}, t)$ that is analogous to the plenoptic function. The function describes the forces felt on a tool given the contact positions $\vec{c}_{0...t}$, the point of contacts on the tool $\vec{d}_{0...t}$, the positions of the tool $\vec{x}_{0...t}$, and the relative orientations of the tool and object $\vec{\phi}_{0...t}$ over time t.

We can choose a slice of this function to explore, just as research in data-based visual modeling has explored specific slices of the plenoptic function. By restricting ourselves to a quasi-static model of elastic objects, the force field function is simplified to $\vec{f}(\vec{c}, \vec{\delta})$, the force field function used in [37].

Our data capture system seeks to sample the force field function by probing the object at a collection of surface contact points. It records the displacements from each of the contact points while applying various forces. Due to the nature of force sensing, we cannot collect data from many positions in parallel like a light field camera. To render the model, we interpolate the capture force fields without explicitly modeling the object's properties.

2.3.3 Hardware Overview

Figure 2.2 shows the hardware setup used to capture data. The structured-light based range scanner is at the top and points down at a shelf holding the object of interest (an inflated glove in the figure) and the robotic arm.

Similar to [25], a PHANToM haptic device is used as a robotic arm. Since

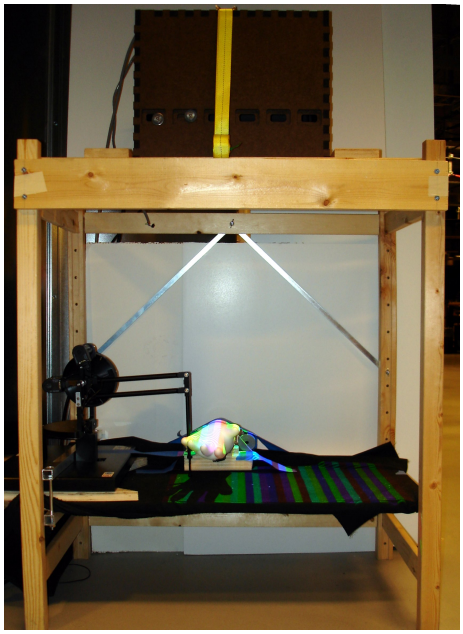


Figure 2.2: Hardware setup overview

users of interactive haptic simulators will have a haptic device, they can use our data capture techniques to build models. Previous work in automatically taking measurements and building a model required specialized hardware [43]. In order to remove the effect of orientation on the interaction with target objects, a rounded probe tip replaces the usual handle and orientation gimbal of the PHANToM.

2.3.4 Alternatives

The system design discussed above is a complete data-based modeling pipeline. Of course, there are many possible alternate designs which have relative strengths and weaknesses. Those designs may be more suited to some applications. Some extensions and alternatives to some components of the design presented in this chapter are discussed in Chapter 6.

Chapter 3

Geometry Acquisition

When presented with an unfamiliar object, your first impulse is likely to be to look at it. If it does not appear dangerous, you may decide to touch it. Where you touch it is probably based on what you see. Vision is a powerful sense that guides many of our actions.

Geometry sensing serves a similar purpose for our data acquisition system. When presented with an object, it is necessary to know its geometry before we probe it. Without knowledge of the geometry, it would be like feeling around in the dark.

In addition to helping us collect and organize force information, the geometry itself is data that is needed in the behavior model. The user needs to see a visual representation of the object in the virtual environment.

This chapter describes the design and implementation of a structured-light range sensor. Using these algorithms, a camera and a projector are all that is needed to sense geometry.

3.1 Design Goals

The overriding goal in the design of this range sensor is to be able to capture the geometry of moving and deforming objects. There should be no restrictions on how or how fast an object changes shape or moves.

A secondary goal is to use commercial-off-the-shelf (COTS) hardware. This makes the system much more practical.

3.2 Sensor Overview

3.2.1 Operating Principle

If depth is computed from a single image, there is no restriction on the motion of objects in the scene. In order to recover a dense depth map, it must be possible to determine from a single image which position in the projected pattern corresponds to each pixel in the image. This can be done by designing a pattern that contains both repeating and nonrepeating elements. When a nonrepeating feature is found in the image, its location in the projected pattern is uniquely known. In a discrete encoding there are only a finite number of possible features. In a continuous encoding, the features must be sufficiently different so that they are reliably identified. This again leads to a finite number of possible features. Therefore, we must have repeating features in order to densely cover the pattern. If the repeating features give us information about their relative locations, we can use the nonrepeating feature locations to disambiguate among their possible locations.

The technique described here combines aspects of color stripe and sinusoidal triangulation systems. In the projected pattern, unique features are created with colored stripes and repeating features are a sinusoid. We use color stripes to get sparse depth values and a sinusoid to propagate these values to get a high-resolution range map. To guide the propagation we segment the image into regions that do not contain any depth discontinuities.

Compared to patterns used in previous work, which employ either color or luminance alone, this pattern is more robust because the color and sinusoidal components provide complementary information. The colored stripe boundaries give sparse but absolute depths, while the sinusoid gives dense relative depths. The sinusoidal component is resistant to reflectance variations in the scene, and

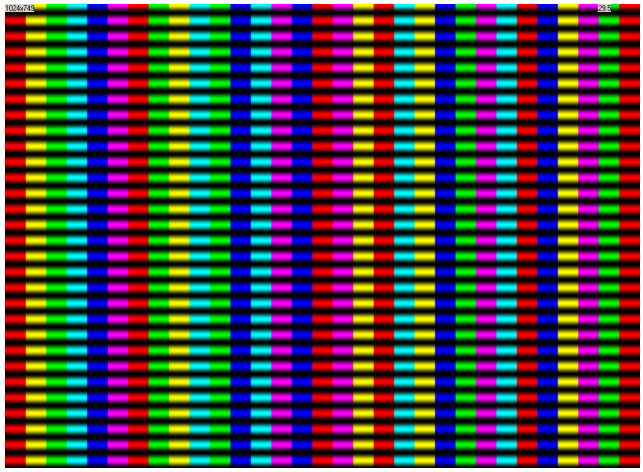


Figure 3.1: The projected pattern

only a few correctly sensed color transitions (also known as color edges) are needed as start points. Using a sinusoidal pattern also avoids the need to do any sub-pixel location estimation of pattern features (e.g. transitions). The depths are computed at pixel locations.

A pattern of vertical colored stripes with a horizontal sinusoid overlaid (Figure 3.1) is projected onto the scene by a standard liquid crystal display (LCD) projector, and images are captured from a different viewpoint with a color camera. The sinusoid of intensity is embedded in the value channel and the colored stripes are embedded in the hue channel in HSV color space.

Each frame can be processed independently to recover depth information. Pixels are given color labels and scores by a Bayesian classifier. The stripe transitions are identified, labeled, and scored according to the color labels of pixels adjacent to each boundary. The luminance component is processed to give relative depth information, and the scene is segmented into regions using a snake technique [30]. In each region, the dense depth map is recovered from the relative depths by using the depths obtained from the color transitions as starting points.

Processing each frame independently eliminates the need for temporal coherence. The scene can change significantly between frames without affecting the depth recovery.

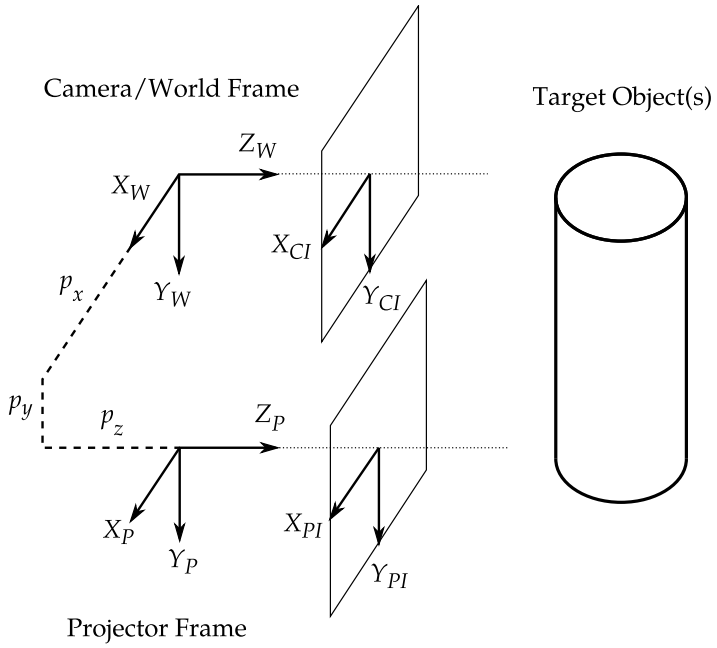


Figure 3.2: Sensor layout

3.2.2 Sensor Layout

Figure 3.2 shows the geometric configuration and coordinate frames of the camera and projector. The world coordinate frame W coincides with the camera coordinate frame. Both are centered at the camera lens optical center with the X_W-axis pointing to the right, the Y_W-axis pointing down, and the Z_W-axis pointing out through the camera lens along its optical axis. The projector frame P is centered at the projector lens optical center which is located at ${}^W[p_x, p_y, p_z]$, and its axes are aligned with those of the world frame. In practice, the optical axis of the projector and camera may not be parallel, but we can create a parallel virtual projector optical axis by rectifying the projected pattern. In fact, it is often useful for the actual optical axes to be verging together at a point near the object of interest.

Three-dimensional points in the world are mapped onto two-dimensional locations on the normalized virtual image planes (located at $z = 1$) of the projector

and camera using a pinhole lens model:

$$
\begin{aligned}
{}^{CI}x &= \frac{{}^{W}x}{{}^{W}z}, {}^{CI}y = \frac{{}^{W}y}{{}^{W}z} \\
{}^{PI}x &= \frac{{}^{P}x}{{}^{P}z}, {}^{PI}y = \frac{{}^{P}y}{{}^{P}z}.
\end{aligned}
\tag{3.1}
$$

Note that ${}^{CI}[x,y] = {}^{W}[x,y,1]$ and ${}^{PI}[x,y] = {}^{P}[x,y,1]$. The mapping of actual image plane coordinates in the camera and projector to the normalized virtual image plane coordinates is done using the model in the Camera Calibration Toolbox for Matlab which accounts for lens distortion [3]. Computing depth is equivalent to computing the ${}^{W}z$ coordinate for each pixel in the camera image.

3.3 Computing Depth

3.3.1 Depth from Color Stripe Transitions

The absolute depths of points in the scene can be found by detecting the color stripe transitions in the acquired image and triangulating with the locations of the stripe transitions in the projected pattern.

The color portion of the projected pattern consists of a set of colors $C = \{c_0, \ldots, c_n\}$ arranged into a sequence $(b_0, \ldots, b_m) \mid b_i \in C$ of vertical stripes such that:

1. $b_j \neq b_{j+1}$, i.e. consecutive colors are different.

2. $b_j \neq b_k$ or $b_{j+1} \neq b_{k+1}$ for $j \neq k$, i.e. each transition appears once.

Using more colors leads to a greater number of possible transitions, but the colors become more difficult to distinguish. In our case, since the sinusoid is used to get the dense range map, having a large number of transitions is not critical. For most data collections, I used variations of red, yellow, green, cyan, blue, and magenta. This allows for 30 transitions.

In the projected pattern, each color transition uniquely identifies a vertical plane that contains the projector optical center. On each scanline (constant ${}^{CI}y$), if a transition is found at ${}^{CI}x$ and it corresponds to the transition located at ${}^{PI}x$ in the projected pattern, we intersect the ray that starts at ${}^{W}[0,0,0]$ and passes through ${}^{CI}[x,y]$ with the plane identified by the transition. The ${}^{W}z$ value of the point in the scene corresponding to the pixel at ${}^{CI}[x,y]$ is

$$ {}^{W}z({}^{CI}x, {}^{CI}y) = \frac{{}^{W}p_x - {}^{PI}x\,{}^{W}p_z}{{}^{CI}x - {}^{PI}x}. \tag{3.2} $$

3.3.2 Recovering Stripe Transitions

Identifying color stripe transitions occurs through two steps. First, the transitions are localized. Then each is labeled according to the colors of pixels adjacent to the transition.

The colors are identified using a Bayesian classifier.[1] Probability distributions for each color are estimated by fitting Gaussian distributions to training images. The training images for the classifier are images of the scene with a projected pattern of only one color. The images are converted to HSV space giving hues $h({}^{CI}x, {}^{CI}y)$, saturations $s({}^{CI}x, {}^{CI}y)$, and values $v({}^{CI}x, {}^{CI}y)$. For each color, c, the mean, μ_c, and variance, σ_c, of the hues are computed and stored. Colors in the pattern should be chosen so that the overlap in the Gaussian distributions is minimized. Our technique is robust to incorrectly classified pixels because all transitions detected in a region are aggregated in phase unwrapping (see Section 3.3.5).

A captured image is processed one scanline (constant ${}^{CI}y$) at a time to locate color transitions. There are several techniques to locate color edges, such as the consistency measure used in [63]. We take a simpler approach by looking at the difference in hue between consecutive pixels. For each group of pixels whose absolute hue difference (taking into account the circular nature of hue) exceeds a set threshold, we pick the pixel with the maximum difference to be the location

[1] Justin Durack developed an initial implementation of the Bayesian classifier.

of a transition.

Once an edge is found, it is given a label and score based on the color scores of windows of pixels on each side of the transition. For an edge at pixel ${}^{CI}k$, the color score of the left and right pixel windows are

$$LCS_c = \sum_{x=k-win}^{k} \frac{CS_c(x,y)}{\sum_{h \in C} CS_h(x,y)},\ RCS_c = \sum_{x=k+1}^{k+win} \frac{CS_c(x,y)}{\sum_{h \in C} CS_h(x,y)}. \tag{3.3}$$

The color score is the estimated probability of observing the hue given that the projected color is c:

$$CS_c(x,y) = \frac{1}{\sigma_c \sqrt{2\pi}} e^{-(h(x,y)-\mu_c)^2/(2\sigma_c^2)}. \tag{3.4}$$

The colors of the left and right pixel windows around the transition are chosen to be the ones with the highest score:

$$lcolor = \arg\max_{c \in C} LCS_c,\ rcolor = \arg\max_{c \in C} RCS_c. \tag{3.5}$$

If $(lcolor, rcolor)$ is consistent with a transition in the projected pattern ($b_i = lcolor$ and $b_{i+1} = rcolor$ for some i) and LCS_{lcolor} and RCS_{rcolor} are greater than a set threshold, the transition is labeled with the ${}^{PI}x$ coordinate of the transition in the projected pattern and given a score, $ES = LCS_{lcolor} + RCS_{rcolor}$. The threshold on the color scores prevents labeling a transition based on colors that were not confidently identified. Validating the transitions against the projected pattern provides further robustness to transitions detected as a result of noise and/or reflectance variations in the scene.

3.3.3 Depth from the Sinusoid

The depth at each pixel is computed from the apparent phase shift between the projected and observed sinusoids. The phase shift wrapped to between $-\pi$ and π is recovered using digital demodulation. After it is unwrapped using the

procedure described in Section 3.3.5, absolute depth can be recovered.

On the projector image plane, the luminance of the projected pattern is

$$v({}^{PI}x, {}^{PI}y) = A\cos(\omega\, {}^{PI}y) + K \tag{3.6}$$

where ω is the frequency of the sinusoid, A the amplitude, and K an offset. The amplitude and offset are chosen to maximize the dynamic range of the projector, and the frequency is set to the smaller of the camera's and projector's Nyquist frequencies. Converting to world coordinates and projecting into the world yields

$$v({}^{W}x, {}^{W}y, {}^{W}z) = A\cos\left(\omega\frac{{}^{W}y - {}^{W}p_y}{{}^{W}z - {}^{W}p_z}\right) + K. \tag{3.7}$$

The camera sees on its image plane the reflection of the pattern on objects in the scene

$$\begin{aligned} v({}^{CI}x, {}^{CI}y) = {} & R({}^{W}x, {}^{W}y, {}^{W}z({}^{CI}x, {}^{CY}y))* \\ & \cos\left(\omega\frac{{}^{W}z({}^{CI}x, {}^{CY}y)\,{}^{CI}y - {}^{W}p_y}{{}^{W}z({}^{CI}x, {}^{CY}y) - {}^{W}p_z}\right) + N({}^{W}x, {}^{W}y, {}^{W}z({}^{CI}x, {}^{CY}y)) \end{aligned} \tag{3.8}$$

where $R(x,y,z)$ is the reflectance of the surface at $[x,y,z]$, ${}^{W}z({}^{CI}x, {}^{CY}y)$ is the depth of the surface at pixel $({}^{CI}x, {}^{CI}y)$ and $N(x,y,z)$ accounts for noise and ambient light. Computing the depth map is recovering ${}^{W}z({}^{CI}x, {}^{CY}y)$. We can rearrange Equation (3.8) into

$$v({}^{CI}x, {}^{CI}y) = R\cos(\omega {}^{CI}y - \theta({}^{CI}x, {}^{CI}y)) \text{ with} \tag{3.9}$$

$$\theta({}^{CI}x, {}^{CI}y) = \omega\left(\frac{-{}^{W}p_z\,{}^{CI}y + {}^{W}p_y}{{}^{W}z({}^{CI}x, {}^{CI}y) - {}^{W}p_z}\right). \tag{3.10}$$

In this formulation θ can be seen as modulating the phase or frequency of the original sinusoid. I will refer to it as the phase. The wrapped version of the phase, $\theta_w = \arctan 2(\sin(\theta), \cos(\theta))$, can be recovered through digital demodulation, multiplying each column of the image $v({}^{CI}x, {}^{CI}y)$ by $\cos(\omega\, {}^{CI}y)$ and low-pass

filtering [49, 57]. Section 3.3.5 describes the procedure to recover the unwrapped phase.

Given $\theta({}^{CI}x, {}^{CI}y)$, we can compute the depth map

$$ {}^{W}z({}^{CI}x, {}^{CI}y) = \frac{\omega(-{}^{W}p_z\,{}^{CI}y + {}^{W}p_y)}{\theta({}^{CI}x, {}^{CI}y)} + {}^{W}p_z. \tag{3.11} $$

The 3D locations of the points are found by projecting back into the world coordinate frame (Equation (3.1)).

3.3.4 Segmenting using Snakes[2]

The depth cannot be computed from the wrapped phase θ_w found with the sinusoid pattern processing. We need $\theta = \theta_w + 2\pi k$ to compute the depth, and the integer k cannot be recovered from the sinusoid. This ambiguity is due to the periodic nature of a sinusoid. However, if the phase (or the depth) of one pixel is known (for example from using the color stripes), we can assume that the adjacent pixels have the same offset, $2\pi k = \theta - \theta_w$, and compute the value of θ for the adjacent pixels. Performing this propagation is called phase unwrapping [20].

The main assumption made in phase unwrapping is that the phase changes by less than 2π between two adjacent pixels along the unwrapping path. So, we need to segment our image in regions such that in each region the difference in the value of θ_w between two adjacent pixels is less than 2π.

There exist various techniques to segment images. We chose the snake (or active contour) technique [30]. A snake is a closed contour, which deforms to fit the borders of the region we want to segment. Each point of the snake is moved iteratively based on the various forces applied to it until it reaches an equilibrium. This technique is known to give good results for many problems and has the advantage of being able to bridge across small gaps in the boundary of a region we want to segment. This property is very important in our case. It

[2]The segmentation technique was developed primarily by Florian Buron and is published in [4, 17, 18].

will enable us to segment a region even if its borders contain gaps that correspond to a local difference of phase that is a multiple of 2π.

The segmentation of the wrapped phase is based on the phase variance (defined in [20]) of the image. The phase variance is a measure of how fast the phase of a pixel changes compared to its neighbors. A large change in phase corresponds to a discontinuity in depth and is interpreted as a boundary in our image. Phase variance has been effectively employed in phase unwrapping for applications such as synthetic aperture radar and magnetic resonance imaging [20].

Most active contour algorithms require the user to create a contour around the object to be segmented and then the snake shrinks to fit the object boundaries. To make the initialization automatic we use a growing snake which is initialized randomly in the image instead. This technique works because the regions we want to segment are characterized by their uniformity (smooth changes of depth). A growing snake which starts inside an object region will expand until it reaches the object boundaries.

We initialize the snake to a small circle centered randomly in the yet unsegmented region of the phase variance image. Then we adjust the location of the circle so that the circle covers a uniform area of small phase variance values. Notice the trade-off on the size of the initial circle: a large circle will guarantee that we are starting inside a uniform region, but will prevent us from segmenting small or narrow regions.

The motion of the contour is controlled by three forces:

Expansion Force: Each snake point P is subject to a constant force pointing toward the outside of the region encircled by the snake and orthogonal to the tangent at P. This causes the basic growing motion.

Image Force: The snake expansion is locally stopped when it reaches regions of high phase variance. The force is proportional to the gradient of the phase variance. To avoid having discontinuities in this force (due to the phase

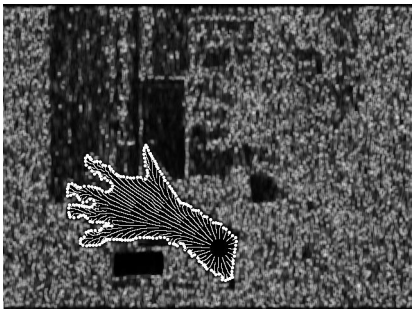


Figure 3.3: Segmenting based on phase variance

being computed at pixel locations), it is interpolated from the four pixels surrounding the current snake point.

Rigidity Force: Each snake point is subject to a force that resists high contour curvatures. This prevents the snake from flooding into other regions through small gaps in the phase variance where the local difference of phase is a multiple of 2π.

In addition to these forces, other events affect the deformation of the snake. To control the discretization, snake points are added or removed when they get too far or too close to each other. The snake is not allowed to self-intersect. It is also prevented from entering regions that have been already segmented.

Once segmentation by the snakes is complete, regions that do not contain any detected color edges can be merged with neighboring regions. The depth found in the merged area will be incorrect if it is truly a separate region. If a region without detected color edges were incorrectly identified as a separate region, merging corrects the error and allows us to compute the depth in that area. On the other hand, without merging it would not be possible to compute the depth in that area at all.

Since this segmentation is based only on phase variance, it could fail if there is a long depth discontinuity where the phase difference is a multiple of 2π. But this is unlikely to happen because the depth difference corresponding to a phase difference of 2π varies with position.

Figure 3.3 shows the segmenting process on the phase variance of a scene containing a hand. Dark pixels are areas with low phase variance. The initial location of the snake was in the wrist area and it traveled outward to the outline of the hand.

3.3.5 Phase Unwrapping

Before unwrapping begins, guesses for θ are computed from the detected color edges by applying Equations (3.2) and (3.10) to each of the transitions labeled according to the process in Section 3.3.2. The score of each guess is the score, ES, of the transition that produced it.

The unwrapping of each region starts at the center of the initial circle of the snake. Then the pixel with the lowest phase variance among pixels that border unwrapped pixels is unwrapped. This proceeds until the entire region is unwrapped in a way similar to the quality map algorithm of [20]. This produces an unwrapped phase θ_u that is offset from the actual phase θ. Once a region is unwrapped, the offset for the region needs to be computed from the guesses encountered in the region.

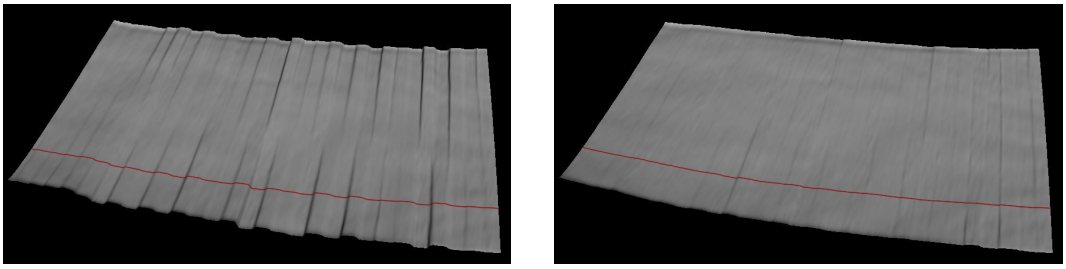
$$\theta(x,y) = \theta_u(x,y) + 2\pi k \tag{3.12}$$

The integer k is computed by rounding the weighted median of the difference between guesses encountered in the region and θ_u divided by 2π. The guesses are weighted by their scores. Using a weighted median provides robustness against bad guesses from misclassified colors or falsely detected edges.

3.4 Using Commercial-off-the-shelf Hardware

3.4.1 Challenges

Use of color presents a special challenge in using COTS projectors and cameras. Color misalignment caused noticeable artifacts in our range maps. The colored



(a) Without compensation (b) With compensation

Figure 3.4: 3D reconstruction of a surface from about 1000 mm away

stripe processing is robust to some color misalignment because the results are only used as guesses for the phase unwrapping and only need to be within 2π of the correct result. But, if the misalignment causes the sinusoidal pattern to be shifted, it will result in a systematic error in the depth. This is especially evident when the shift varies between different colored stripes causing ridges to appear in the range map. Figure 3.4(a) shows the reconstruction of a surface at about 1000 mm from the sensor. The artifacts are clearly visible.

Several sources contribute to the color misalignment. Both the projector and camera lenses exhibit chromatic aberration. A typical C-mount lens such as the one used in our experiments is designed for monochrome closed-circuit TV (security camera) or industrial machine vision applications where chromatic aberration is irrelevant. LCD projectors are normally used to show presentations where audience members will not notice small amounts of chromatic distortion. It is not intended to be used as a precision instrument. Specialized lenses that compensate for chromatic aberration could be used in the camera but would be much more expensive. Using such a lens in the projector is not possible since its lens is not designed to be replaceable. Additional color distortion in the projector is due to color being produced by three separate LCD panels. The panels could be misaligned and their associated optical paths may not be perfectly matched.

Chromatic distortion can be divided into traverse (lateral) and longitudinal components. Longitudinal distortion results in the colors not being in focus in the same plane and cannot be easily compensated for after the image has been

captured. Our depth sensing technique is not very sensitive to this effect because the sinusoid has no sharp edges. Lateral chromatic aberration can be modeled by focal length and distortion parameters that vary with wavelength.

3.4.2 Compensation for Chromatic Distortions

If we assume each color channel of the captured image has an associated focal length, stretching the different color channels appropriately can compensate for the distortion. In the projector, we can stretch the color channels to compensate for focal length changes and shift the color channels relative to each other to compensate for misalignment before projecting the pattern.

In many cameras (including ours), color is sensed by putting a Bayer pattern color filter in the optical train, which reduces the effective resolution. This makes it difficult to find the focal length and distortion parameters for each color channel directly. Additionally, the projector can only be calibrated through images captured through the camera lens, which will show the combined distortions of both the projector and camera.

In our system we chose the green image channel as the reference. Green has a refraction index between red and blue, the camera has the highest effective resolution in green channel due to the Bayer pattern filter, and it is the most sensitive channel. Differences observed in the red (blue) channels relative to green are modeled with three parameters: α the ratio of the green camera focal length to the red (blue) camera focal length, β the ratio of the green projector focal length to the red (blue) projector focal length and γ the shift in direction of the Y-axis of the red (blue) channel relative to the green channel in the projector. A shift does not need to be considered in the camera because color is sensed through a Bayer pattern filter in the camera which means the color channels are always aligned in the camera. We only consider a shift in the Y-axis direction because shifts in the X-direction do not affect the projected sinusoid. Shifts in X do affect the color transitions, but as previously discussed they do not need to be accurately located.

If these parameters are known we can compensate for the distortion. The projected sinusoid (Equation (3.6)) is modified for each color channel by

$$v({}^{PI}x, {}^{PI}y) = A\cos(\omega(\beta\,{}^{PI}y + \gamma)) + K. \tag{3.13}$$

For the camera, we scale the acquired images' red (blue) channel by α and resample

$$(x, y) = (\alpha\,{}^{CI}x, \alpha\,{}^{CI}y). \tag{3.14}$$

The parameters α, β, and γ can be estimated from the measured phases computed from the green channel θ_g and the red (blue) channel θ_r (θ_b) of an image taken of a projected white sinusoid. In the compensated images, each channel should have the same phase at every point. We minimize

$$\sum_{Q}(\theta_s(\alpha x, \alpha y) - \theta_g(x, y))^2 \tag{3.15}$$

where Q is the set of points in the camera images which have reliable phase measurements and θ_s simulates the effect of compensating for the shift and focal length difference. It is computed from θ_r (θ_b) by

$$\theta_s(x, y) = \beta(\theta_r(x, y) - \omega((\frac{1}{\alpha} - 1)y - \gamma)). \tag{3.16}$$

The $(\frac{1}{\alpha} - 1)$ term accounts for the effective change in sampling period of the camera due to the change in focal length.

In the results discussed in this chapter, the optimization was done over measurements made of surfaces about 540 mm, 690 mm, 825 mm, and 1075 mm from the sensor.

3.4.3 Effectiveness of Compensation

The effectiveness of the distortion compensation technique was evaluated by taking range maps of a set of four surfaces sensed at about 500 mm, 700 mm,

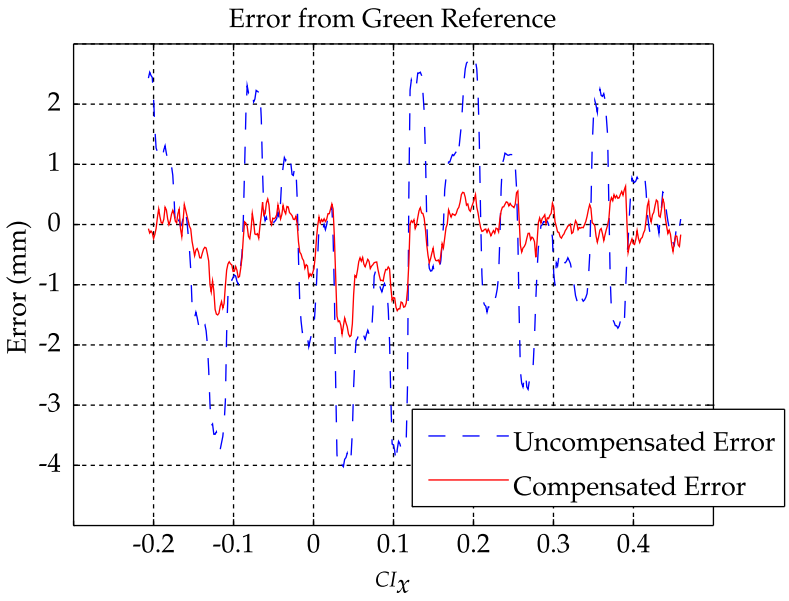


Figure 3.5: Error on cross-section of surface

1000 mm, and 1200 mm from the sensor using the hardware setup described in Section 3.5. Each surface was imaged using a white sinusoid for reference and the multi-colored sinusoid with and without compensation for comparison.

We defined the error to be the difference between the depths computed from the green channel of the white sinusoid and the depths computed from the colored stripe image. Figure 3.5 shows the error on a cross-section of the surface sensed at about 1000 mm. This cross-section is along the red line in Figure 3.4 where the combined effects of the camera and projector are fairly large. Figure 3.4(b) shows the reconstruction of the same surface sensed using compensation. Applying correction significantly reduces the artifacts. Over the whole data set, the compensated root mean square (RMS) error is on average 36% of the uncorrected RMS error. The compensated RMS error is 0.05% of the average measured distance to each surface. For instance, on the surface at about 1000 mm, the RMS error is 0.48 mm. The remaining errors are likely due to higher order variation in the lens distortion between colors in the camera and projector and unmodeled effects of the projector's optics (such as rotation of the LCD panels relative to each

other).

These results were obtained by optimizing α, β, and γ over a fairly large working volume (over 700 mm in ${}^{W}z$). It is likely that the error would be further reduced if only a small working volume was desired.

3.5 Experiments

For the results shown here, we projected patterns with a Viewsonic PJ551 LCD projector at a resolution of 1024x768 pixels. Images were captured with a Basler A602fc camera at 640x480. A pattern similar to Figure 3.1 was used. In each case, the colors were adjusted as described in Section 3.3.2 and ω was selected based on the distance the object of interest was from the sensor.

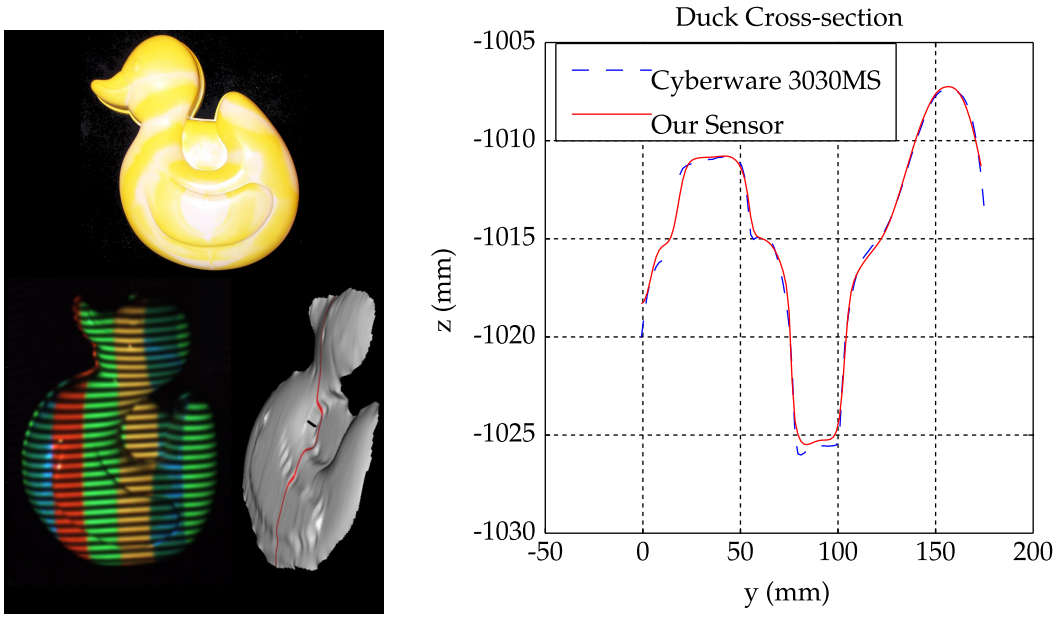
3.5.1 Accuracy Tests

To evaluate the accuracy of the sensor, we captured range maps of a plastic toy duck and a computer speaker. The range maps were compared to scans of the same objects made using a Cyberware 3030MS laser stripe scanner. The laser scans were aligned to the range maps using a variant of the Iterated Closest Point (ICP) algorithm [46].

Figure 3.6(a) shows a photograph of the duck along with the image from the camera used in the sensor and the extracted range map. Figure 3.6(b) shows a plot of the cross-section of the range map along the red line in Figure 3.6(a). There are gaps in the cross-section for the laser scan because it was manually edited to remove spurious points caused by specularity.

The range map from our sensor captures the general shape but has some small distortions where specular highlights were in the captured image. When aligned with the reference range map the RMS error over the parts present in both is about 0.48 mm or 0.1% of the average distance to the object.

We captured a sequence where a computer speaker was being rotated and translated. The speaker was about 650 mm from the sensor. Figure 3.7(a) shows



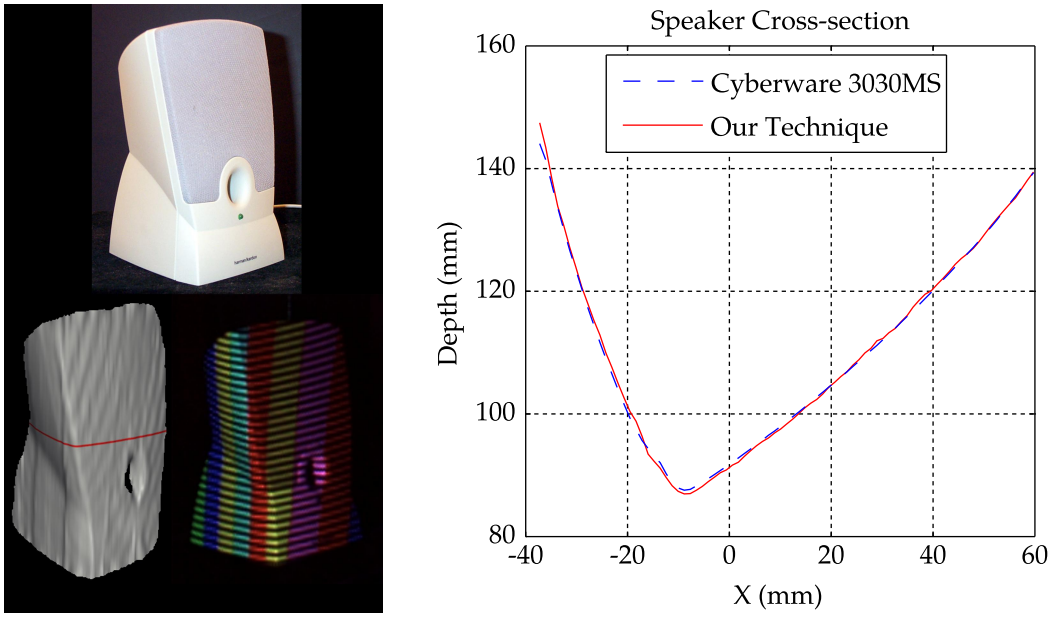
(a) (b)

Figure 3.6: (a) Photograph, image from sensor camera, range map, and (b) cross-section of range map of a plastic toy duck

a photograph, an image from the range sensor camera, and the corresponding range map from the sequence of the computer speaker. Figure 3.7(b) shows the cross-section along the red line in Figure 3.7(a) of the range map and a laser scan. The range map shows the bend in the left side and the hole on the front. Note that no surfaces on the actual speaker are flat. There are small ridge artifacts along colored stripe boundaries due to the color misalignment problem. The ridges on the front appear to be a Moiré-like pattern caused by interaction between the sinusoidal component of the pattern and the speaker's mesh surface. In all cases, the ridges and noise are less than ± 1 mm from the laser scan surface reference. The RMS error between the range map and reference is about 0.7 mm.

3.5.2 Deformable Object Tests

We also captured sequences of a variety of deforming objects. The slowest moving object was a piece of foam attached to the front of a piece of more rigid



(a) (b)

Figure 3.7: (a) Photograph, image from sensor camera, range map, and (b) cross-section of range map of a computer speaker

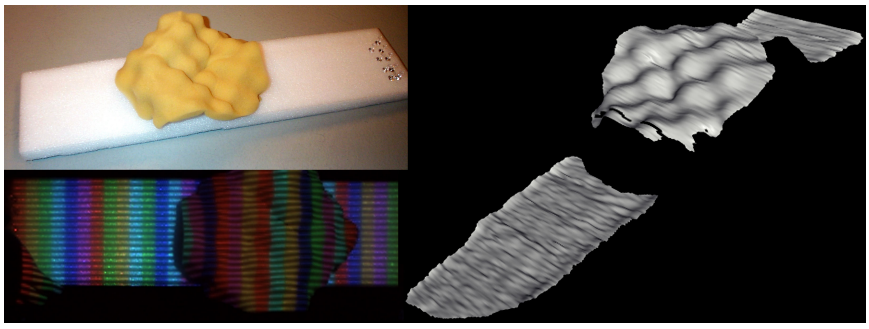


Figure 3.8: Photograph (upper left), image from sensor camera (lower left), and range map (right) of foam test object

Figure 3.9: Photograph of popcorn tin (upper left), image from sensor camera (upper right), and range maps of person moving a popcorn tin (bottom)

foam. We captured a sequence while deforming the softer foam by pulling on wires connected to the back of it. Figure 3.8 shows a photograph of the foam and a frame from the sequence captured about 700 mm from the foam. The sensor is able to correctly separate the deforming foam from the background foam. The rougher appearance of the holder foam is due to the actual surface being rougher and its open-cell structure causing subsurface scattering. Also, notice that the colors in the stripes look different when reflected by two types of foam. The sensor is able to deal with this through a combination of the robustness of the Bayesian color classifier and the aggregation of guesses in computing the offset.

Figure 3.9 shows two frames from a sequence that contains both rigid motion and deformation. A person is holding a popcorn tin and moving it around in front of his body. The picture on the popcorn tin affects the range map by adding some small distortions and, in very dark areas, holes. However, the sensor correctly senses the person and the overall shape of the popcorn tin. In

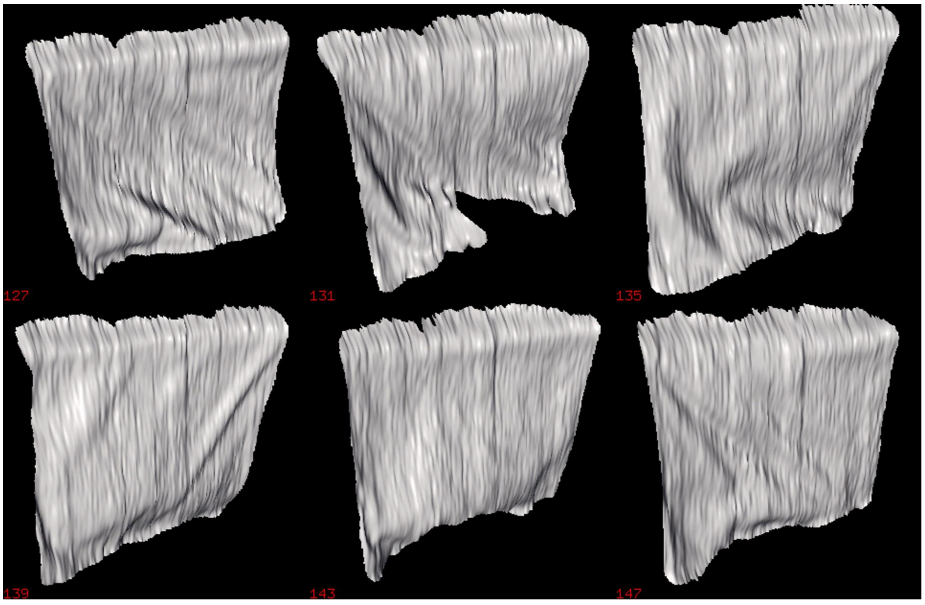


Figure 3.10: Range map frames 127, 131, 135 (top row), 139, 143, and 147 (bottom row) of a waving flag

particular, the shape and motion of the shirt worn by the person comes out very nicely.

The sensor also works well with fast deformation. Figure 3.10 shows six frames from a sequence of a waving flag. A piece of fabric attached to a wooden board was waved in a side to side motion. The noise is larger than in the foam and speaker sequences because in order to fit the flag in the field of view it was about 1300 mm from the sensor. In frames 127, 131, and 147 the wooden board is moving leftward and in 139 and 143 it is moving rightward. In frame 131, part of the fabric is missing because it was at a steep angle relative to the camera and as a result very dark. In frame 135, one can see the folds in the fabric making a C-like shape, which is a result of the board changing directions.

The fastest moving object captured was a balloon filled with water bouncing on a table. Figure 3.11 shows selected frames from a sequence captured at 100 fps. It shows the balloon deforming as it hits the surface and bounces several times. Frame 25 shows the balloon just before hitting the surface when it is still spherical. The following frames show the balloon deforming as it hits the surface and

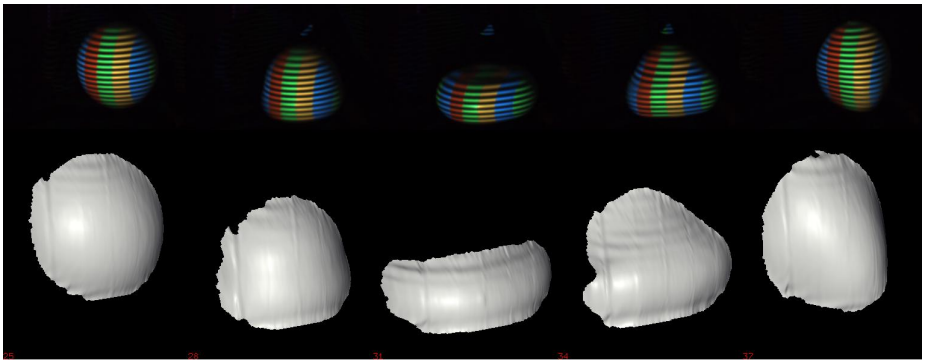


Figure 3.11: Frames 5, 28, 31, 34 and 37 of the water balloon sequence

bounces up. There is significant deformation and motion between every frame. A small amount of artifacting due to chromatic aberration is visible. There is also some artifacting near the top edge of the visible surface where the apparent frequency of the sinusoid is lower. This frequency is nearing the cutoff limit of the filter used in demodulating the sinusoid.

3.6 Discussion

The approach described in this chapter has a number of advantages. The sensor produces good quality range maps showing the effectiveness of a combination of a discrete color pattern and continuous sinusoidal intensity pattern. This combination pattern avoids the trade-off between spatial and temporal resolution made by previous approaches. The spatial resolution can be scaled by using a higher resolution camera and higher resolution projector.

It works on objects moving much faster than what has been shown with multi-frame techniques. Because only a single image is needed to extract a range map and the pattern is static, the range map frame rate matches the camera frame rate. Systems using pattern sequences are limited by the speed of both the projector and camera. Cameras faster than 500 fps are commercially available, while almost all projectors work at 60 fps or less. Additionally, in cases where adapting the projected pattern to the scene is unnecessary, the LCD projector can be eliminated

saving cost and weight.

Objects were successfully sensed over a wide range of distances without adjusting the sensor setup (lens focus, baseline, etc.). The distortion compensation technique is also effective. It reduces artifacting and allows the use of COTS hardware.

There are, however, some disadvantages. The unwrapping process requires at least one correct phase guess in each region. The colored stripe processing often produces many more than needed. But in objects that have strongly saturated colors there will be fewer correct guesses making the system more sensitive to segmenting errors. For example, the sensor will have difficulties on a bright red object. This can be mitigated by choosing the right set of colors in the pattern in some cases. However, if the scene contains many saturated colors it then could be very difficult to find a good set of pattern colors.

Also, since we need to project and see the pattern, this approach will not work in situations where the ambient light is brighter than the projector. For example, this makes it difficult to use it outdoors in sunlight. This is not a problem when the range sensor is used as part of our data acquisition system. It is indoors and we can control the environment.

Chapter 4

Capturing Force and Friction Data

The second data acquisition system capability needed is capturing the behavior of the object and haptic information. The range sensor tells us the shape of an object but not how it feels. This chapter describes a system to collect information about how an object feels and how it responds to interactions by actively probing it.

4.1 Design Goals

As discussed in Section 2.3.2, the forces felt by a tool at position $\vec{x}$ is given by a haptic force field function $\vec{f}(\vec{c}, \vec{\delta})$ where $\vec{c}$ is the initial contact location on the object's surface and $\vec{\delta} = \vec{x} - \vec{c}$ is the displacement of the tool from the contact location. Motion of the contact location depends on the friction properties at the contact location. The data acquisition system seeks to sample the force fields at various contact locations for various displacements and record the maximum force that can be applied in various directions without slipping (i.e. exceeding the friction limit). In addition, while the object is being probed, the deformed geometry should be captured for visual rendering.

As with the range sensor, for practical reasons minimizing complexity and utilizing available hardware were secondary goals.

4.2 Selecting Contact Locations

The current implementation of the probing system uses a simple strategy to select contact locations at which to make measurements. Points are selected from alternating grid lines on a range map of the undeformed geometry based on their distance to the edge of the range map. A point must not be too close to the boundary of the range map to decrease the chance of the probe tip failing to make contact. The distance between grid lines controls the overall number and density of contact locations. This produces a set of contact locations that is approximately uniformly distributed over the surface. Note that the range sensor cannot sense surfaces that are occluded or surfaces with very oblique normals. However, these are also likely to be locations at which it is hard to position the probe tip without collisions between the links of the robot and the object due to the location of the PHANToM and its kinematics.

A discussion of more complex contact location selection strategies is in 6.2.1.1.

4.3 Probing at a Contact Location

For each of the selected contact locations, samples of the force field function $\vec{f}(\vec{c}, \vec{\delta})$ can be obtained by repeatedly applying different forces at the contact point and recording the resulting displacement. While exploring the force field, we also want to capture the friction properties at $\vec{c}$ by determining the maximum tangential force component $\vec{f}_T$ that can be applied without the probe tip slipping from $\vec{c}$. Since the surface deforms when a force is applied, the true normal and tangential directions to the surface are not known. The probe tip hides the area of the surface where it is touching from the view of the range sensor. Instead, the normal and tangential directions obtained from the undeformed geometry are used as references. A variety of directions in the tangent plane needs to be explored to ensure the applied forces will have components in various tangential directions on the deformed geometry. This will give samples on the friction limit and allow us to extract the average normal direction (see Section 5.4.1).

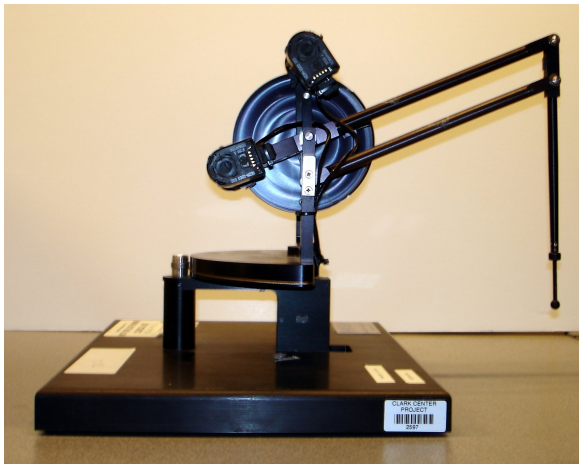


Figure 4.1: PHANToM haptic device with probe tip

This is done by applying forces with the same normal force $\vec{f}_N$ component but with tangential components which vary in a binary search pattern. The upper limit $\vec{f}_{Tupper}$ is initially set such that the total force does not exceed the maximum continuous force of the robot. The lower limit $\vec{f}_{Tlower}$ is initially 0 N. A test force with a tangential component $\vec{f}_T = \frac{\vec{f}_{Tupper}+\vec{f}_{Tlower}}{2}$ at the midpoint of the interval is applied. If the contact slips, $\vec{f}_{Tupper}$ is set to the test force. Otherwise $\vec{f}_{Tlower}$ is set to the test force. This is repeated until $\vec{f}_{Tupper} - \vec{f}_{Tlower}$ is less than a set threshold. The lower limit is conservatively taken to be on the friction boundary for the given normal force component.

This probing process produces a set of sample points of $\vec{f}(\vec{c}, \vec{\delta})$ and samples of the friction limit boundary for each contact point.

4.4 Probing with a PHANToM

The PHANToM haptic device is a three degree-of-freedom robot arm with high resolution encoders and open-loop force control (Figure 4.1). It does not have a force sensor or any type of slip sensor. It is a natural choice to use for probing the object. Since a haptic device is needed for interaction with the virtual world, it is available to anyone who wishes to use models resulting from

the data acquisition system. In addition, any limitations such as workspace or maximum force limits do not need to be exceeded since the same limitations exist in the playback environment. Since our model is quasi-static, measurements can be made by applying a force and waiting for equilibrium. At equilibrium, the applied force is equal to the response force of the object.

In the procedure described in Section 4.3, the ability to detect contact with the object, detect slipping, and accurately apply forces is critical. Each of these can be done with the techniques described below without the addition of any hardware.

Adding sensors was considered but rejected. A force-torque sensor would exacerbate occlusion of the object from the range sensor, add mass to the robot arm, and increase cost and complexity. It is not clear what type of sensor would be suitable to detect slipping on the surface of deformable objects. Tactile sensing slip sensors measure micro-slip or micro-accelerations and produce signals which are difficult to interpret even on rigid surfaces [11]. On a deformable surface, motion from deformation of the surface adds to the difficulty. Also tactile sensors typically need to be embedded in soft finger tips. The data collected by probing implicitly incorporates all aspects of the probe tip object interaction. Therefore, the probe tip should be similarly shaped to the virtual tool that will be interacting with our model in the virtual world. If the virtual world is not simulating interaction via a soft finger tip, the tactile sensor could not be used.

4.4.1 Detecting Contact

We must be able to distinguish when the end effector is moving through free space from when it is moving while in contact with the object (i.e. deforming it). Information from the range sensor enables us to place the end effector near the desired point of contact on the surface, but the exact point of contact must be determined by the robot.

The contact location can be determined by examining the difference in acceleration of the probe tip while in free space from when it is in contact. When the

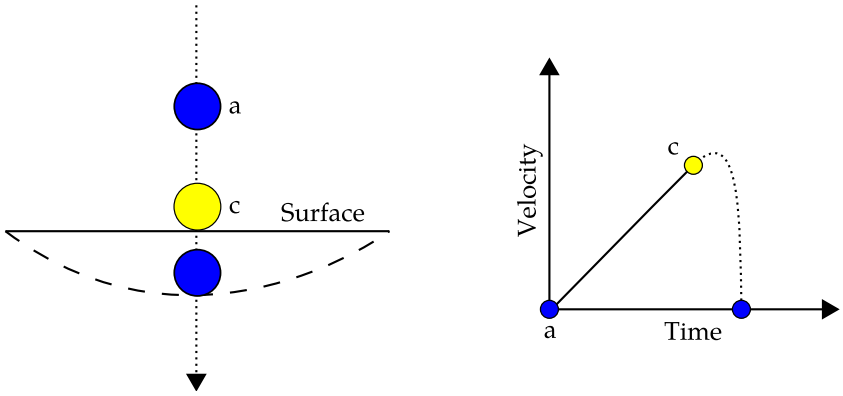


Figure 4.2: Detecting contact

end effector is near the desired contact point, Point a in Figure 4.2, we apply a constant force towards the surface. While it is in free space (between Points a and c) the velocity increases at a constant rate. Once it contacts the surface (at Point c), it will start to slow down. The contact position can be determined by detecting when the acceleration is no longer constant.

Directly estimating acceleration from the Cartesian positions of the end effector is difficult. The positions are non-uniformly quantized because they are derived from joint encoder positions. Instead, velocity is estimated by a linear least squares fitting of parabolas to the position data. A global estimate is derived using all the data points since the beginning of the trajectory to the current point and a local fit is derived using a window of points around the current point. After the point of contact the global fit will tend to overestimate the velocity magnitude because it assumes a parabolic trajectory while the local fit will adapt more quickly. The contact point is determined by comparing these two velocity estimates and looking at the position when the estimates deviate beyond a set threshold.

Figure 4.3 shows data collected while making contact with a piece of foam. In this data, the local and global velocity estimates start deviating at around 0.03 secs from the beginning of the trajectory, indicating initial contact was made at about 27.3 mm.

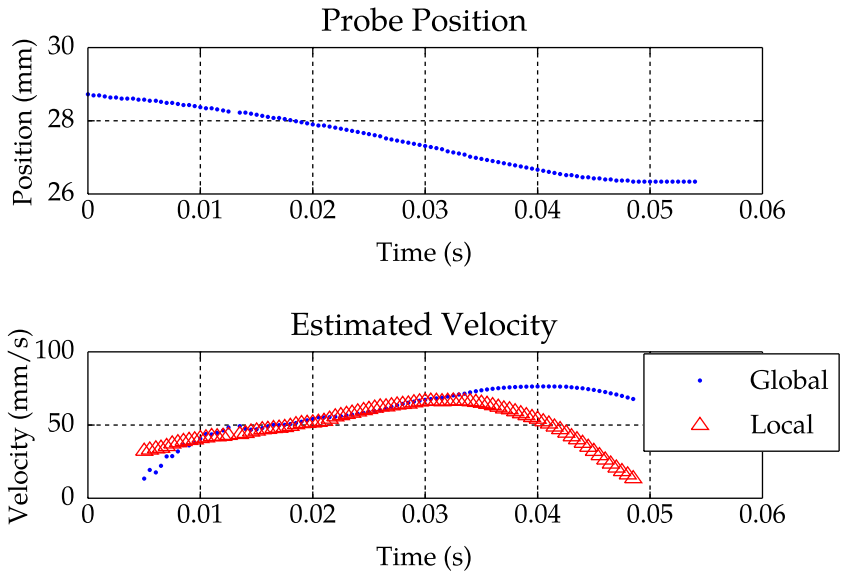


Figure 4.3: Making contact with foam

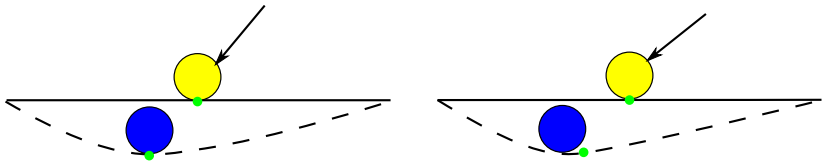


Figure 4.4: Motion with deformation only (left) and with deformation and slipping (right)

4.4.2 Detecting Slipping

Whether the probe tip slips when a force is applied while it is in contact with the surface is dependent on the friction properties of the surface and probe tip. For Coulomb friction, the maximum tangential force magnitude that can be applied without slipping is related to the normal component of the force by $\left\|\vec{f}_{Tmax}\right\| = \mu \left\|\vec{f}_N\right\|$. If the magnitude of the tangential component of the applied force $\vec{f}_T$, is less than the friction limit for the applied normal component $\vec{f}_N$, the probe will not slip and any motion results in deformation (Figure 4.4 left). If the tangential component exceeds the friction limit, the probe may slip off the object or slip to another equilibrium position on the surface while also deforming

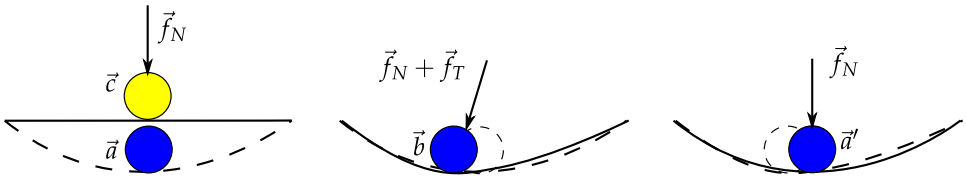


Figure 4.5: Detecting slip step 1 (left), 2 (middle), 3 (right)

the object (Figure 4.4 right). With rigid objects slipping can be easily detected because any motion will result in slipping. With deformable objects it is necessary to distinguish motion without slipping from motion with slipping.

Once again, we may take advantage of the elastic nature of the object. If a force is applied and then removed, the probe will return to the same location if no slipping occured. If it does not return to the same position, it slipped on the surface. The following procedure to apply forces exploits this property by comparing the positions of the probe tip before and after applying the tangential component of the force (Figure 4.5):

1. Make contact with the object while applying $\vec{f}_N$. Wait for equilibrium and record position $\vec{a}$. If the probe tip has fallen off the object, it slipped.

2. Apply $\vec{f}_T + \vec{f}_N$. Wait for equilibrium and record position $\vec{b}$. The displacement for this force field sample is $\vec{\delta} = \vec{b} - \vec{c}$. If the probe tip has fallen off the object, it slipped.

3. Remove $\vec{f}_T$. At equilibrium compare the new position, $\vec{a}'$with $\vec{a}$. If $\|\vec{a} - \vec{a}'\|$ exceeds a set threshold, slipping has occurred.

4.4.3 Force Control

There is no force sensor in the PHANToM, and force commands are mapped to motor currents in an open-loop manner. As a result, the force applied at the end effector is a function of commanded force, gravity, and internal forces such as friction:

$$\vec{f}_{app} = \vec{f}_{cmd} - \vec{f}_g(x_t) - \vec{f}_{int}(\vec{x}_0, \ldots, \vec{x}_t). \tag{4.1}$$

The force resisting gravity $\vec{f}_g(\vec{x}_t)$ is dependent on the current position of the device $\vec{x}_t$, while the force needed to resist internal forces $\vec{f}_{int}(\vec{x}_0, \ldots, \vec{x}_t)$ is dependent on both the current position and (some) past history. To compensate for these effects, $\vec{f}_g$ and $\vec{f}_{int}$ need to be estimated and added to the desired force, i.e. $\vec{f}_{comp} = \vec{f}_{des} + \vec{f}_g + \vec{f}_{int}$. This compensated force is commanded instead of directly using the desire force $\vec{f}_{des}$.

4.4.3.1 Gravity Compensation

Since the force needed to resist gravity only depends on the current PHANToM state, it is straight forward to build a static curve that maps position to compensation force. Only the last two joints of the PHANToM are affected by gravity when the base is on a level surface. Each joint is moved through its range of motion using a PID controller, and the torque needed to hold position at various points is recorded.

If we only consider the last two joints, the Jacobian for the position of the probe tip is

$$J(\vec{\theta}) = \begin{bmatrix} l_1 c_1 & l_2 s_2 \\ -l_1 s_1 & l_2 c_2 \end{bmatrix} \tag{4.2}$$

where l_i is the length of link i, c_i is $\cos\theta_i$, s_i is $\sin\theta_i$, and θ_i is the angle of Joint i. This means the torque for each joint can be fit to

$$\tau_i(\theta_i) = a_i \cos(\theta_i + \phi_i)$$

where a_i and ϕ_i are parameters of the fit. Static torques and forces are related by $\vec{f}_g(\vec{x}) = J(\vec{\theta})^{-T}\tau(\vec{\theta})$ with $\vec{\theta}$ related to $\vec{x}$ by the kinematics.

Because of friction, the torque needed to hold positions while moving a joint will differ with the direction of the joint motion. To isolate the effect of these internal forces from the effect of gravity, the average difference between the torques recorded while moving up and while moving down is computed. Half this average is subtracted from the torques recorded while moving up and added to the torques while moving down before performing the torque function fit.

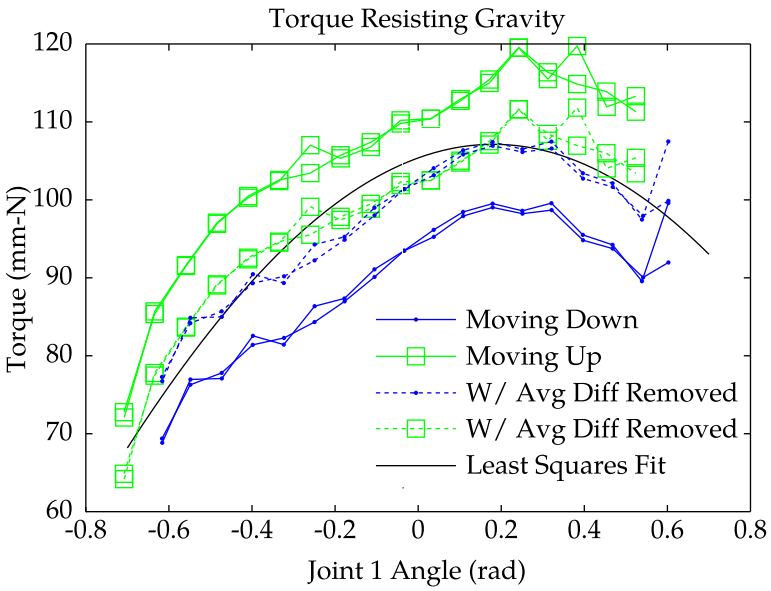


Figure 4.6: Joint 1 torques required to resist gravity

Figure 4.6 shows example data collected while moving Joint 1 (second to last joint) and holding at various positions. The joint was moved through its range of motion twice. The lines with box markers are moving the joint through increasing angles, and lines with dot markers are moving through decreasing angles. The dashed lines are the torques required to hold position with the average difference between directions removed. The solid line shows the fitted torque function.

4.4.3.2 Internal Force Compensation

Forces internal to the device which affect the applied force are primarily friction forces. They are observed as a hysteretic effect on the commanded force versus the applied force. As can be seen in the gravity compensation data (Figure 4.6) the force needed to hold the probe tip in a position varies with the direction from which it was approached. These internal forces are dependent on both the current state and some state history. As a result, the force needed to hold the end effector at a point depends on the point and the path to the point. Unlike gravity forces, it cannot be easily expressed by a static compensation curve.

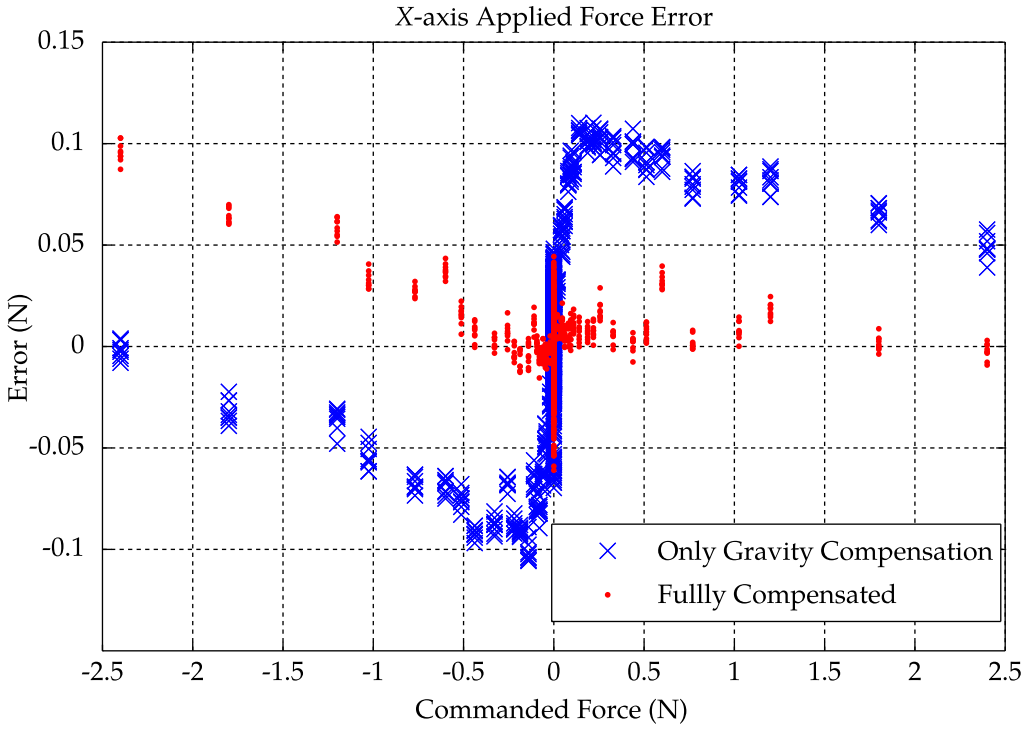


Figure 4.7: Error in force applied along the *X*-axis of the PHANToM

However, once we have accounted for the gravity resistance force, the force needed to resist these internal forces can be determined by looking at the integral term of the PID controller used to move the probe tip in free space. When the end effector is in equilibrium (not moving) in free space, the force applied by the integral term of the controller is equal to the internal forces of the robot. While we cannot use this technique after contact is made, we can assume that internal forces are similar for nearby points. We can approach the point of contact from the same direction as the desired force vector to be applied, stop before making contact, and take the force from the integral term to be an estimate of $\vec{f}_{int}$.

To evaluate the effectiveness of compensating for internal forces, forces applied by the PHANToM with and without internal force compensation were measured using a six degree-of-freedom force-torque sensor. Table 4.1 shows a summary of the data. The measurements were scaled and rotated to match the PHANToM coordinate frame by taking the rotation and scaling which minimized

Axis	X (tangent)	Y (normal)	Z (tangent)
Only Gravity Compensation			
Mean	0.0100	0.0323	-0.0261
Std-Dev	0.0453	0.0146	0.0190
Full Compensation			
Mean	0.0049	0.0212	-0.0059
Std-Dev	0.004	0.009	0.004

Table 4.1: Summary of applied force errors

the RMS error between commanded forces and measured forces. The Y-axis was the normal direction. Figure 4.7 shows the error between commanded forces and measured forces along the X-axis.

Without compensation the internal forces cause a large relative error of approximately 100% for small commanded forces in the X-axis. The error for a command of 0.1 N is about 0.1 N. The ability to accurately apply small tangential forces is important to estimating the friction properties. If the coefficient of friction is small, even a small tangential force will cause slipping.

When fully compensated, the largest errors in the X-axis occur with the largest commanded forces making the relative error small. In addition, the overall mean error is reduced for all axes and the variation of the forces is decreased as shown by smaller standard deviations.

4.5 Output of Data Acquisition

Performing the probing procedure in Section 4.3 using the procedures described in Section 4.4 produces samples of the the force field at various contact points. For each contact point, there is a set of data triples $(\vec{p}, \vec{a}, \vec{b})$ where $\vec{p}$ is the total force applied, $\vec{a}$ is the position with only the normal component applied, and $\vec{b}$ is the position with the total force applied. Samples in which the component of $\vec{p}$ tangent to the undeformed surface were the largest that could be applied without slipping for that direction and normal force component are marked as being on the friction boundary. Range maps of the deformed geometry for these

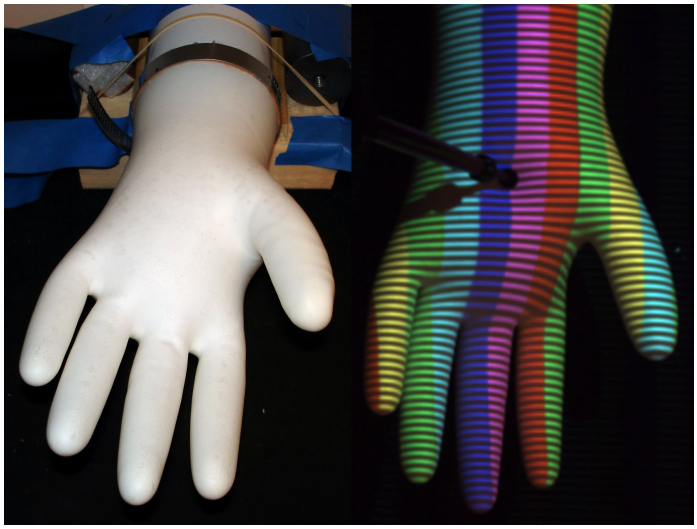


Figure 4.8: Photo of inflated vinyl glove (left) and view from range sensor camera (right)

samples and samples with no tangential force component are also saved.

4.6 Example Measurement Results

4.6.1 Inflated Vinyl Glove

An inflated vinyl glove was mounted at the wrist and probed. Figure 4.8 shows a photo of the glove and an image taken from the range sensor during the probing process. The last link of the PHANToM, the probe tip, and the structured-light pattern are visible. Areas where the projection or view of the pattern is occluded are missing from the geometry data.

Figure 4.9 shows visualizations of two samples in the data set at the same contact location on the undeformed object. The left side shows the location of the probe and geometry of the object while 0.1 N of force is applied normal to the surface, and the right side shows the same information while 3 N of force is applied. The bounding box of the undeformed surface is given for reference, and a grid is projected from the range sensor camera center point to show the

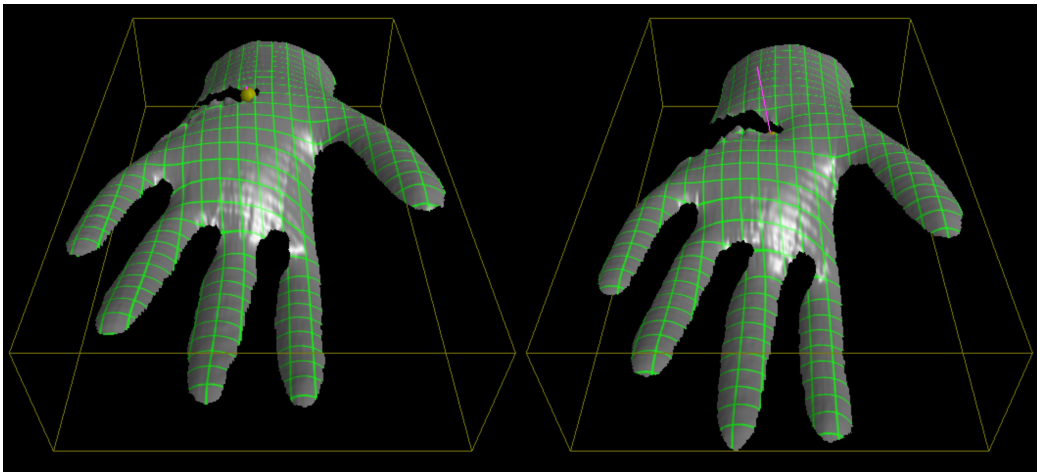


Figure 4.9: Deformation with 0.1 N of normal force (left) and 3 N (right)

curvature of the surfaces. A magenta line extends from the sphere representing the probe in the direction of the applied force with a length proportional to the magnitude of the force. Unsurprisingly, the glove deforms more when a larger force is applied, and the tips of the fingers move farther than the wrist area which is closer to the mounting area.

Figure 4.10 shows an orthographic view of the data set. Each circle is centered at a contact point where measurements were made. The diameters of the circles are proportional to the displacement in the direction normal to the undeformed surface when 0.547 N was applied normal to the undeformed surface at that point. It shows an overall pattern of it being easier to deform the glove when pressing on points farther from the wrist.

4.6.2 Latex Sheet

A 1/16 inch sheet of latex was suspended over a gap while being supported on two edges by wooden supports. Figure 4.11(a) gives a photo of it and Figure 4.11(b) shows contact locations where measurements were made and an outline of its depth map. The test object is larger than the intersection of the workspace of the robot and the range sensor. Blue tape added after the data collection in the

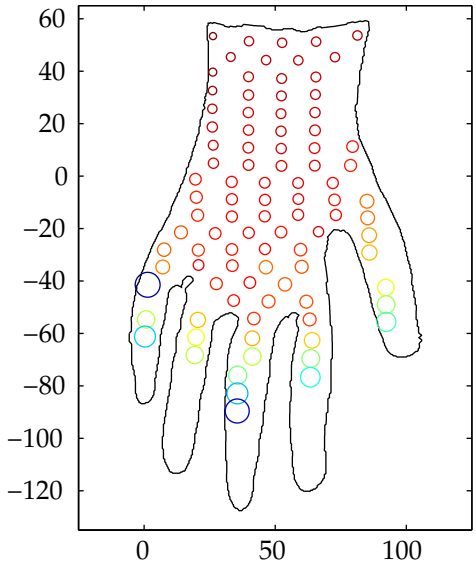
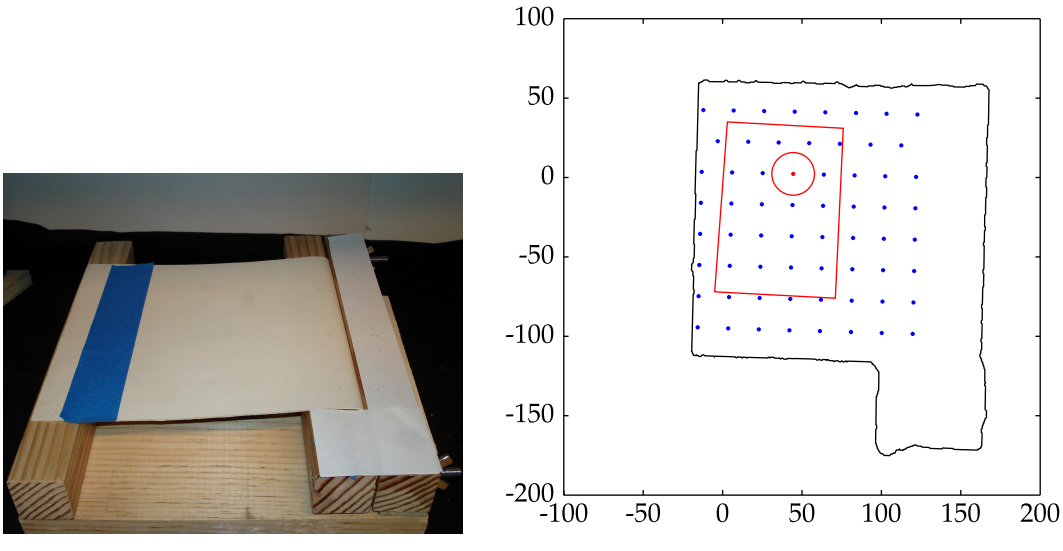


Figure 4.10: Displacement in direction normal to undeformed surface of glove



(a) (b)

Figure 4.11: Latex sheet test object

photo shows approximately where the left edge of the depth map is on the latex surface. Measurements were also made with a wooden block placed underneath the sheet in the gap. The top of the block was about 15 mm below the latex sheet. This creates an object with more variation in force and is somewhat similar to feeling a structure underneath skin. The approximate location of the block is shown in Figure 4.11(b).

Plots of the measurements made at the circled contact location in Figure 4.11(b) are shown in Figure 4.12(a) for data captured without a block and Figure 4.12(c) for data with a block. The Z-axis is normal to the undeformed surface. The base of each arrow is plotted at the displacement $(\vec{b} - \vec{c})$ when a force $\vec{p}$ proportional to the length of the arrow and in the direction of the arrow was applied. The green arrows are the samples which lie on the friction boundary. There are no green arrows on the bottom layer because the PHANToM cannot apply more than 3 N for any useful amount of time. It was unable to apply enough tangential force to reach the friction limit. When the block is present at around 16 mm of displacement, the force magnitudes increase quickly without much increase in displacement. This is where the latex sheet made contact with the block.

The displacement normal to the undeformed surface when 3 N of force is applied normal to the undeformed surface with no block present is visualized in Figure 4.12(b), and displacements when a block are present is shown in Figure 4.12(d). Unsurprisingly, the displacements are larger when no block is present and in areas not over the block. Displacements in areas over the wooden support are even smaller than areas over the block.

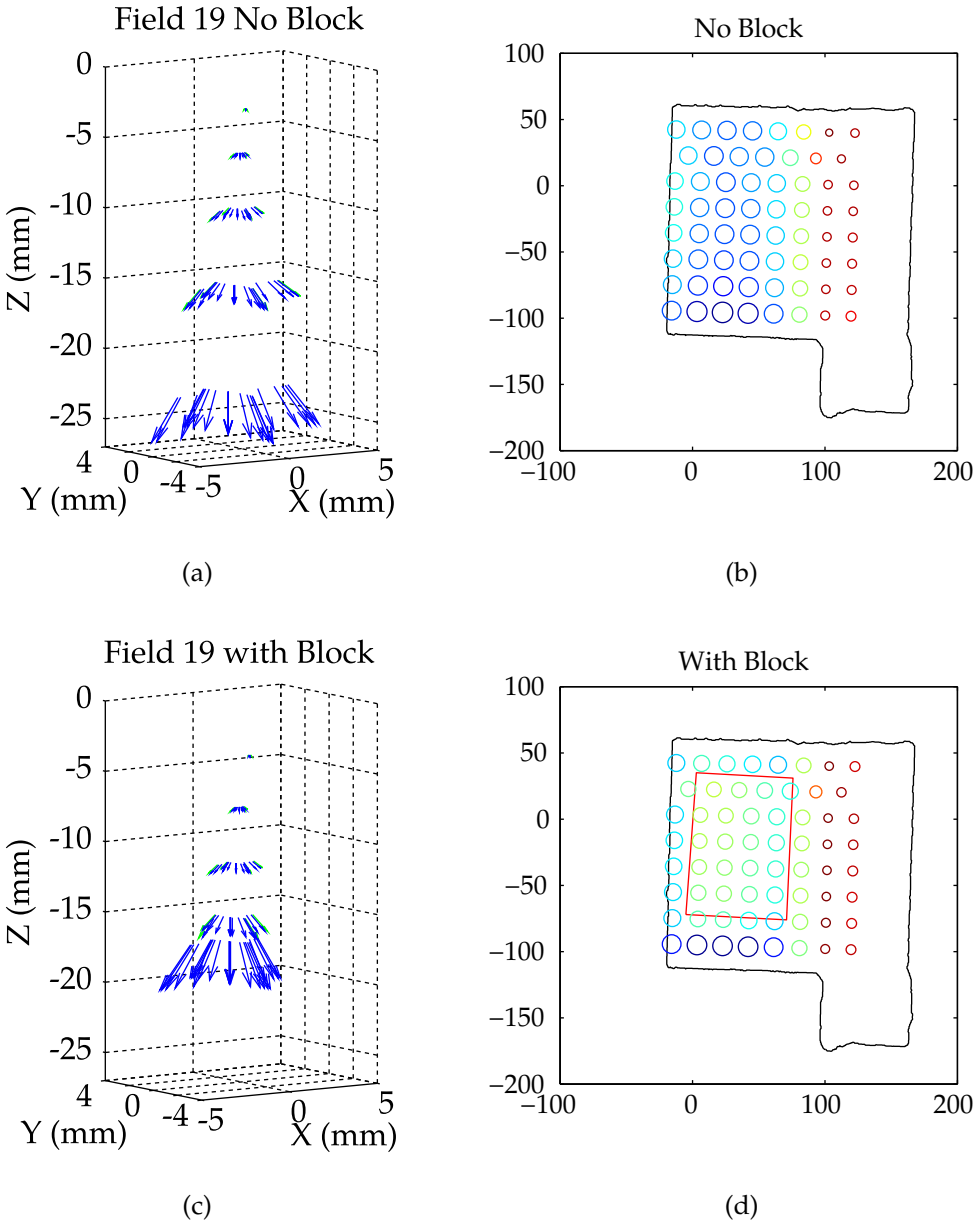


Figure 4.12: Latex sheet test object measurements

Chapter 5

Model Creation and Rendering

The central task of the techniques described in this chapter is to extract information from the data set captured by the data acquisition system, store it in a structure, and render the modeled object in a virtual environment from the contents of this structure. Organizing the data into a data-based model is necessary because we cannot efficiently render from raw data. It would be like trying to plan a road trip using only a list of latitude and longitude coordinates of road intersections. Without a map to show how these intersections are connected by roads, it may not even be possible to follow the sequence of selected intersections.

The techniques presented in this chapter are illustrated using data sets described in Section 4.6 and described using the same notation for data set quantities.

5.1 Requirements and Challenges

For haptic rendering, two questions need to be answered:

1. What is the value of the force field at an arbitrary contact and for an arbitrary displacement?

2. Where does the contact point move when the user moves outside the friction limits?

Our data-based model must not only answer these questions but also be designed with the following challenges in mind. First, the data set is noisy because it is composed of measurements of a real object. These measurements are obtained by applying a series of probes to the object, and each of the probes will vary. They may not be applying exactly the same force and may not make contact at exactly the same location. The object may also respond differently to each probe. The object itself may not be exactly elastic while our measurement techniques and modeling approach assume perfect elasticity. This will introduce errors that depend on the order of the probes. Damping in the object will affect the measurements depending on how long we waited for equilibrium after applying a force before recording the displacement. Both too much and too little damping require longer waits for equilibrium.

Second, there is limited computation time at playback. As discussed in Section 1.1, the computation at playback must occur in less than 1 ms for a haptic loop that runs at 1000 Hz. The organization of our model must allow us to efficiently answer the questions posed above.

5.2 Data Pair Realignment

We can mitigate the effects of probes not making contact at the exact same location by realigning the data pairs. The measurement process records two positions for each probe, the position with only the tangential component of the force applied $\vec{a}_i$, and the position with the whole force applied $\vec{b}_i$. If the object were perfectly elastic and the measurement system were ideal, applying the same force at the same contact location would result in the same displacement.

In the data acquisition process, a number of probes with the same normal force component but different tangential force components are applied at the same contact location to explore the friction limits (see Section 4.3). In practice, the positions of the probe tip after applying only the normal component of the force vary slightly. This indicates that the object may be responding slightly differently or the probe tip may be making contact at slightly varying locations.

For example, in the data set of the latex sheet without a block, the standard deviation of the $\vec{a}_i$ positions measured with the same normal force applied is 0.017 mm and in the data set with the block it is 0.024 mm.

For each group of positions that were recorded with the same normal force component applied, we can realign $\vec{a}_i$ to the mean position $\overline{\vec{a}} = \frac{1}{N}\sum_{i=1}^{N}\vec{a}_i$ while maintaining the positions with the complete force applied relative to the positions with only the normal force applied, i.e. $\vec{b}'_i = \vec{b}_i - \vec{a}_i + \overline{\vec{a}}$. This realignment process does not adjust the recorded force. However, as long as the displacement due to a force is similar for contact locations that are close together, adjustment of the force is not needed.

5.3 Force Field Function Approximation

To answer the first question of Section 5.1, we need to compute $\vec{f}(\vec{c}, \vec{\delta})$ for $\vec{c}$ and $\vec{\delta}$ that are not in the measurement set. This is done by interpolating from the data set. Interpolating the force field from the sample points can be decomposed into two parts. If the contact point is fixed, we need to interpolate within the force field at that contact point to generate values at arbitrary displacements. This will enable us to evaluate the force field function for arbitrary displacements at all of the contact locations where measurements were made. Then to generate force field values for contact points between points where measurements were made, we can interpolate from the force field functions at the the fixed contact locations.

5.3.1 Approximation within a Force Field

When interpolating within a force field, the contact point $\vec{c}$ is fixed. We want to find a vector valued function, the force field function, that is a function of the displacement: $\vec{f}(\vec{\delta})$. Rather than exactly interpolate all of the force field samples, we construct an approximation that is close to the samples $\vec{f}(\vec{c} - \vec{b}_i) \approx \vec{p}_i$. Exact interpolation would reproduce noise that is present in the data and is difficult

to achieve because there can be nearly repeated samples. If the same force were applied at the same contact location multiple times, the resulting displacements should be averaged. Approximation smoothes out noise and allows constraints which give us desirable properties to be enforced. Since the samples are noisy measurements, it is unlikely they would exactly satisfy a constraint.

5.3.1.1 Desirable Force Field Properties

There are a number of desirable properties that a force field function should satisfy:

Conservative $\nabla \times \vec{f}(\vec{\delta}) = 0$: Since we are modeling elastic objects, the force field should be conservative.

Zero at initial contact $\vec{f}(\vec{0}) = \vec{0}$: There should be no force until the tool penetrates the object. This ensures continuity of forces since there is no force when the tool is not in contact with the object.

Passive $\int_{\vec{0}}^{\vec{\delta}} \vec{f}(\vec{x}) \cdot d\vec{x} \geq 0$ for $\vec{\delta}$ reachable by the tool: If the real object is passive, the user should not gain energy while interacting with the virtual object.

These properties are also the assumed properties of the force fields used in the proof of passivity in the playback system of [37].

5.3.1.2 Radial Basis Fields

Radial basis function interpolation is a popular method to approximate scalar functions and interpolate scalar data. Given a set of points $\vec{x}_1, \ldots, \vec{x}_N$ and function values $y_1, \ldots, y_N$ at those points, an exact interpolator is a function which passes through the data at the given points, i.e. $p(\vec{x}_i) = y_i$. A radial basis func-

tion is a radially symmetric function of distance $\phi(\|\vec{r}\|)$. A radial basis function interpolator is formed by the sum of radial basis functions at a set of centers with a set of weights:

$$p(\vec{x}) = \sum_{k=1}^{M} \alpha_k \phi(\|\vec{x} - \vec{g}_k\|). \tag{5.1}$$

For exact interpolation, the solution for the weights exists and is unique if the centers $\vec{g}_k$ are the data location points $\vec{x}_i$ and the interpolator satisfies certain conditions [22]. For approximation, the number of basis functions can be smaller than the number of data pairs ($M < N$), but the criteria for center locations, solution existence, and uniqueness have not been thoroughly studied.

The radial basis fields technique is an extension of the radial basis functions to approximating vector fields [39]. Given a set of points $\vec{x}_1, \ldots, \vec{x}_N$ and vectors $\vec{y}_1, \ldots, \vec{y}_N$ at those points, we wish to find a vector function $\vec{g}(\vec{x}_i) \approx \vec{y}_i$ which approximates the data. If the vectors are values from an unknown conservative vector function as in our case, the gradient of a scalar function can be the approximating function. A radial basis field interpolator is formed by taking the gradient of a radial basis function interpolator

$$\vec{g}(\vec{x}) = \nabla p(\vec{x}) = \sum_{k=1}^{M} \alpha_k \frac{\vec{x} - \vec{g}_k}{\|\vec{x} - \vec{g}_k\|} \frac{\partial \phi(\|\vec{x} - \vec{g}_k\|)}{\partial \|\vec{x} - \vec{g}_k\|}. \tag{5.2}$$

Radial basis fields formed from multiquadratic radial basis functions are used to interpolate measured force field values. A multiquadratic radial basis function has the form

$$\phi(\|\vec{r}\|) = \sqrt{\|\vec{r}\|^2 + 1} \tag{5.3}$$

and the associated interpolator is

$$p(\vec{x}) = \sum_{k=1}^{M} \alpha_k \phi(\|\vec{x} - \vec{g}_k\|) + \vec{d} \cdot \vec{x} + K. \tag{5.4}$$

The linear term is needed to fulfill the conditions necessary for a unique solution in the exact interpolation case [22]. Applying Equation (5.2) gives the associated

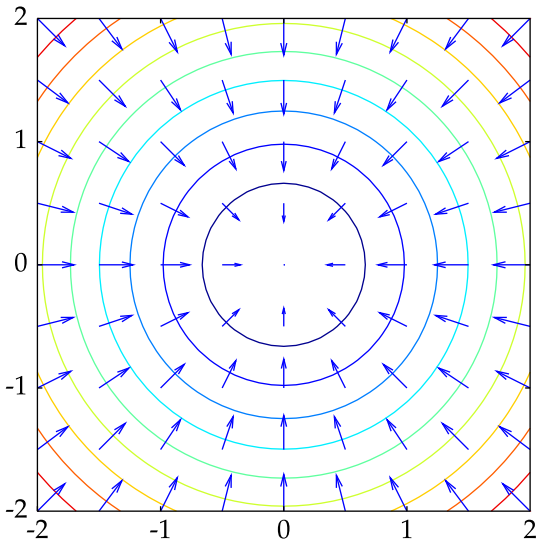


Figure 5.1: Multiquadratic radial basis field with potential function isocontours

radial basis field interpolator and the form of our force field function

$$\vec{g}(\vec{x}) = \sum_{k=1}^{M} \alpha_k \frac{\vec{x} - \vec{g}_k}{\sqrt{\|\vec{x} - \vec{g}_k\|^2 + 1}} + \vec{d}. \tag{5.5}$$

Figure 5.1 shows a two-dimensional plot of a multiquadratic radial basis field centered at the origin and isocontour lines of its potential function.

The parameters of the interpolator are set by minimizing

$$\sum_{i=1}^{N} \|\vec{g}(\vec{b}_i - \vec{c}) - \vec{p}_i\|^2 \tag{5.6}$$

where $\vec{b}_i$ are the displacements and $\vec{p}_i$ are the applied forces of the measured force field samples at $\vec{c}$. A starting point for the numerical optimization can be obtained by choosing the centers to be a subset of the data locations and solving the resulting linear least squares problem for the coefficients. Then nonlinear optimization can be done over the coefficients and center locations.

The radial basis field interpolator matches our desired force field function properties well. It smoothly interpolates the given data vectors. It is conservative

because it is formed by taking the gradient of a scalar potential function. We can enforce the constraint that the function is $\vec{0}$ at $\vec{0}$ by setting

$$\vec{d} = \sum_{k=1}^{M} \alpha_k \frac{\vec{g}_k}{\sqrt{\|\vec{g}_k\|^2 + 1}}. \tag{5.7}$$

This gives the force field function the form

$$\vec{g}(\vec{x}) = \sum_{k=1}^{M} \alpha_k \left(\frac{\vec{x} - \vec{g}_k}{\sqrt{\|\vec{x} - \vec{g}_k\|^2 + 1}} + \frac{\vec{g}_k}{\sqrt{\|\vec{g}_k\|^2 + 1}} \right). \tag{5.8}$$

The passivity condition is more difficult to achieve. It is equivalent to requiring that the potential function value at $\vec{0}$ be less than or equal to its value over the volume of points at which we will be computing the force field, i.e. $p(\vec{0}) \leq p(\vec{\delta})$. $\vec{0}$ is a critical point of the potential function since its gradient is constrained to be $\vec{0}$. There is no simple way to enforce it to be the global minimum, but we can ensure that is a local minimum.

A critical point of a function is a local minimum if the Hessian matrix at that point is positive definite. The Hessian matrix of Equation (5.8) at $\vec{0}$ is

$$H = \sum_{k=1}^{M} \frac{\alpha_k}{(\|\vec{g}_k\|^2 + 1)^{\frac{3}{2}}} \begin{bmatrix} g_{ky}^2 + g_{kz}^2 + 1 & -g_{kx}g_{ky} & -g_{kx}g_{kz} \\ -g_{kx}g_{ky} & g_{kx}^2 + g_{kz}^2 + 1 & -g_{ky}g_{kz} \\ -g_{kx}g_{kz} & -g_{ky}g_{kz} & g_{kx}^2 + g_{ky}^2 + 1 \end{bmatrix}. \tag{5.9}$$

A matrix is positive definite if the determinants of all its leading minors are strictly positive. This translates into nonlinear inequalities which can be enforced during the numerical optimization process that sets the coefficients and centers. Many nonlinear numerical optimizers such as those in the Matlab Optimization Toolbox accept nonlinear inequalities as constraints. Note that since we require that $p(\vec{0})$ be less than but not necessarily strictly less than $p(\vec{\delta})$ for reachable $\vec{\delta}$, the determinants need only be non-negative.

A radial basis field interpolator also has a useful property that was not in the list of Section 5.3.1.1. Its representational power is independent of coordinate

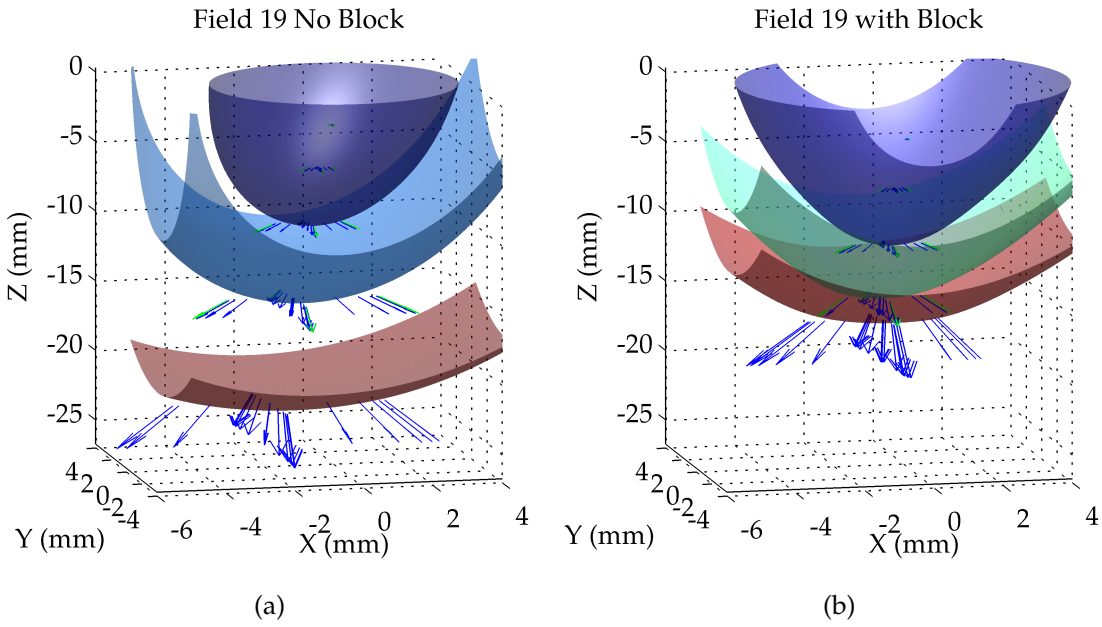


Figure 5.2: Six-basis-field fit

rotations. The radial basis functions only depend on distance which is rotationally invariant, and the linear portion can be represented in any rotated coordinate system by applying the same rotation to it. Independence from the coordinate frame allows the coordinate frame to be chosen based on other criteria. This is in contrast to interpolators which depend on the coordinate system such as fitting scalar functions to each coordinate independently. In [36, 37], the coordinate frame for the force field had to be chosen based on competing criteria of friction directions and force field representability.

5.3.1.3 Radial Basis Field Fit Results

In my test data sets, good quality fits were obtained using significantly fewer bases than data pairs. Figure 5.2 shows a six-basis-field fit to the force field samples measured at the circled contact location in Figure 4.11(b) for the data sets taken without and with a block underneath the latex sheet. The measured data are plotted for reference, and isosurfaces of the potential functions are shown.

The interpolated force field vectors are normal to the isosurfaces. In both cases, the fitted force field function captures the underlying force field well. The RMS errors between the interpolating function and the measurements are 0.0917 N and 0.101 N respectively. Compared with the force field without a block underneath, the force field with the block has larger forces at smaller displacements. This occurs because the probe is pushing on the block through the latex sheet. The maximum normal force of 3 N was not sufficient to cause the latex sheet to contact the bottom of the test object without a block underneath.

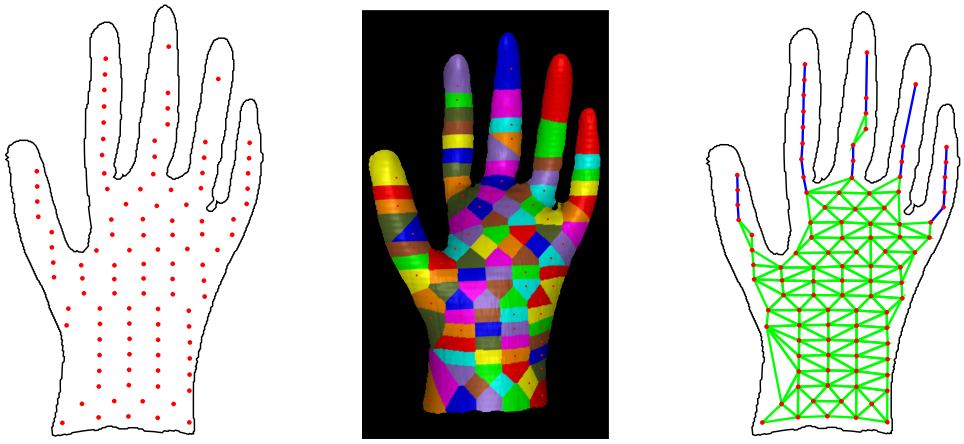
5.3.2 Interpolation between Contact Points

In our model, force fields are associated with contact points on the undeformed geometry. Before we can interpolate between contact points, the contact points must be spatially organized so that we have a parameterization to use as the basis of the interpolation. Otherwise, the notions of "between contact points" and "nearby" are not well defined. The spatial relationships between the contact points control how their associated force field functions will be blended together.

To generate a value at a contact point that was not in the measurement set, force field functions associated with nearby measured contact locations should be combined together. If the contact point is at a contact location in the measurements, no blending is needed.

The straight-line Euclidean distance is not a good metric to use. It can cause contact points on different sides or parts of the object such as different fingers of the glove to be considered near each other. Also, the Delaunay graph based on Euclidean distances may not be a manifold surface. Since the points are in 3D, it can contain tetrahedra. Using geodesic distances on the undeformed geometry avoids these difficulties.

Another desirable property is continuity in the interpolated force field as the contact point moves on the surface. When the set of force fields being blended changes, there should not be a discontinuity in the blended force field.



(a) (b) (c)

Figure 5.3: Computing geodesic Delaunay triangulation. (a) Contact locations, (b) geodesic Voronoi decomposition, and (c) Delaunay graph

Linear blending weighted by barycentric coordinates on the geodesic Delaunay triangulation of the contact points satisfies both our desired properties. For a point $\vec{c}$ with barycentric coordinates λ_D, λ_E, λ_F, inside triangle $\triangle DEF$, the interpolated force field is

$$\vec{f}(\vec{c},\vec{\delta}) = \sum_{K=D,E,F} \lambda_K \vec{f}(\vec{K},\vec{\delta}). \tag{5.10}$$

5.3.2.1 Computing the Approximate Geodesic Delaunay Triangulation

An approximation of the Delaunay triangulation can be produced by first computing the approximate Voronoi decomposition and connecting adjacent Voronoi cells with edges (Figure 5.3). The measured contact points are first registered to the range map of the undeformed geometry using the iterated closest point (ICP) algorithm [46]. Figure 5.3(a) shows an orthogonal projection of the contact points in the inflated glove data set and the outline of the undeformed geometry's depth map. After registration, the approximate Voronoi decomposition is efficiently computed on the 2D range map grid using a fast marching method (Figure 5.3(b)). The running time of the fast marching method is $O(N \log N)$ for

a range map with N points [51].

Range map points which are closest to the same contact point form a Voronoi cell. Edges are added between adjacent cells and the Delaunay graph is produced (Figure 5.3(c)). The graph is a triangulation everywhere except areas such as the fingers in the glove data set that do not have enough contact points.

5.3.2.2 Coordinate Frame Interpolation

The coordinate frame of the force field functions is also interpolated. Similar to interpolating normals in force shading or graphics rendering, interpolating the coordinate frames smooths the edges between triangles. Users will not feel a triangle edge as the contact point moves over it.

In [36, 37], the coordinate axes were independently linearly interpolated with barycentric weights. However, this does not guarantee that the resulting coordinate frame will be orthonormal. Interpolation with an exponential map will ensure an orthonormal coordinate frame [1]. For a contact point $\vec{c}$ with barycentric coordinates λ_D, λ_E, λ_F, inside triangle $\triangle DEF$, the interpolated coordinate frame is

$$R_c = \exp(\sum_{k=D,E,F} \lambda_k \log R_k) \tag{5.11}$$

where R_k is the coordinate frame at k.

This can be efficiently computed using the Rodrigues formula. For any rotation matrix R, there is a skew symmetric matrix B whose matrix exponential is the rotation matrix $R = e^B$. As given in [19], the logarithm of the rotation matrix has the form

$$B = \begin{bmatrix} 0 & -c & b \\ c & 0 & -a \\ -b & a & 0 \end{bmatrix} \tag{5.12}$$

where $[a, b, c]^T$ is the axis of rotation and $\theta = \sqrt{a^2 + b^2 + c^2}$ is the amount of rotation. Then

$$e^B = I + \frac{sin\theta}{\theta} + \frac{(1 - cos\theta)}{\theta^2} B^2. \tag{5.13}$$

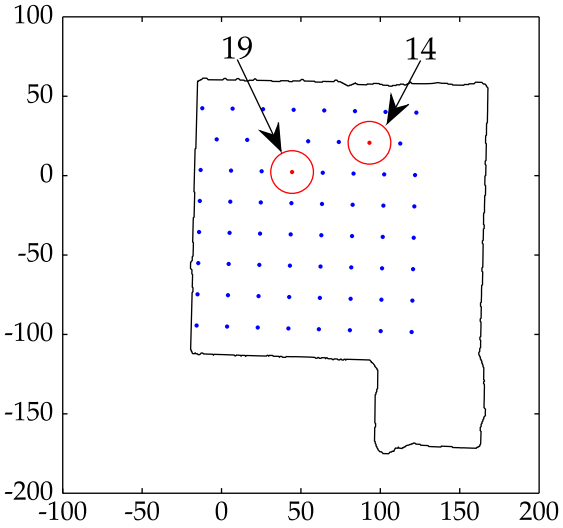


Figure 5.4: Location of removed measurement samples

Contact Location	14	19
RMS Difference (N)	1.38	0.196

Table 5.1: RMS difference between measurements and interpolation

5.3.3 Evaluation of Interpolation Performance

To evaluate the performance of using radial basis fields with barycentric contact point interpolation, the results of interpolation were compared with measured data. A model was constructed from a measurement data set with force field measurements at one contact location removed. The force field samples at the contact not used in the construction of the model were then compared with the interpolation results at those sample locations.

Our proposed technique works well as long as the model has stored similarly behaving locations surrounding the contact location where we are interpolating. Table 5.1 shows the RMS difference between the interpolated force field values and the measured force field values for the data set of the latex sheet without a block. Models were constructed without measurements at contact locations 14 and 19 (Figure 5.4). Location 14 is near the edge of a wooden support beneath the latex sheet. The interpolated values there do not match the measurements

very well because force field values from the nearly rigid area supported by the wood and the area suspended over the gap are being combined together. The interpolated values match the measured values much more closely at contact location 19. Location 19 is in the middle of the suspended part of the latex sheet and is surrounded by other locations that have similar behavior. They are all relatively far from the wooden supports.

5.4 Motion of Contact Point

5.4.1 Extracting Friction Properties

Friction properties are needed to compute the motion of the contact point during model playback. Since our model is quasi-static, there is a static friction coefficient $\mu(\vec{c})$ and normal direction for each contact point. The coefficient of friction and normal direction may change as the object deforms. However, we do not have a direct measurement of the normal since the contact location is obscured by the probe tip.

Instead, we can extract an average friction coefficient and average friction normal at each contact point by fitting a cone to the force field samples which lie on the friction boundary $\Gamma(\vec{c})$ for contact point $\vec{c}$. We find

$$\arg\min_{\alpha,\theta,\phi} \sum_{\vec{p}\in\Gamma(\vec{c})} \left(\cos(\alpha)\sqrt{\|\vec{p}\|^2 - (\vec{p}\cdot\vec{s})^2} - \sin(\alpha)(\vec{p}\cdot\vec{s}) \right) \tag{5.14}$$

$$\vec{s} = \begin{bmatrix} \sin(\theta)\cos(\phi) \\ \sin(\phi) \\ -\cos(\theta)\sin(\phi) \end{bmatrix} \tag{5.15}$$

where $\vec{s}$ is the axis of the cone, α is the angle of the cone's opening, and $\mu(\vec{c}) = \tan(\alpha)$. This will be a conservative estimate of the friction coefficient. Recall that the measurement process applies forces that have various components in the tangent plane to the undeformed surface in various directions. The samples with forces that have the largest tangential component in various directions in

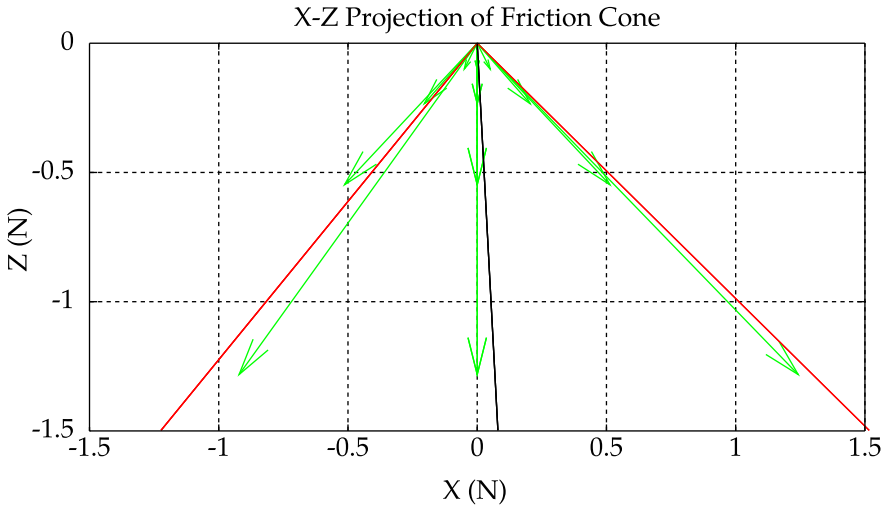


Figure 5.5: Projection friction cone

the tangent plane which do not slip are marked as lying on the friction boundary.

Figure 5.5 shows a two-dimensional projection of the force field samples collected at position 19 on the latex sheet with no block. The Z-axis direction is normal to the undeformed surface. The green arrows are the force samples, the fitted cone is shown in red, and the axis of the cone is the black line. The estimated coefficient of friction is 0.91.

As discussed in Section 5.3.1.2, we are free to choose any coordinate rotation without affecting the force field function. It is convenient to make the Z-axis of the force field function the direction of the friction normal. This simplifies computation of the friction constraint (Inequality (5.16)) and the coordinate frame interpolation of Section 5.3.2.2 will naturally interpolate the friction normal. Like the force field function, the friction coefficient can be linearly blended with barycentric weights.

5.4.2 Computing the Motion of the Contact Point

During playback, as long as the user's tool position $\vec{x}$ satisfies the friction constraint

$$\vec{f}_T(\vec{c}, \vec{x} - \vec{c}) \leq \mu(\vec{c}) \vec{f}_N(\vec{c}, \vec{x} - \vec{c}) \tag{5.16}$$

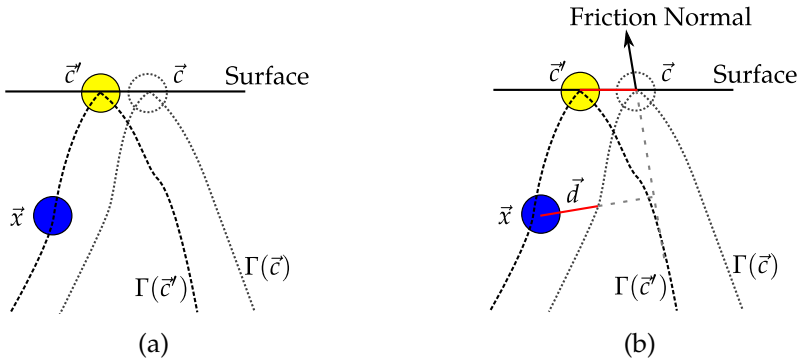


Figure 5.6: Contact point motion

where $\vec{f}_T$ is the tangential component of the force and $\vec{f}_N$ is its normal component, the contact point does not change. This defines a volume within which the tool position can move without moving the contact point. When the tool position moves such that the constraint is no longer satisfied, a new contact position $\vec{c}'$ must be found (Figure 5.6(a)). The tool position must be on the boundary of the friction volume at the new contact position $(\vec{x} - \vec{c}') \in \Gamma(\vec{c}')$ where

$$\Gamma(\vec{c}) = \{\vec{\delta} \mid \vec{f}_T(\vec{c}, \vec{\delta}) = \mu(\vec{c})\vec{f}_N(\vec{c}, \vec{\delta})\}. \tag{5.17}$$

There are multiple solutions for $\vec{c}'$ because $\Gamma(\vec{c}')$ is a surface. There are many contact positions (usually a continuum) where the tool position will be on the associated friction boundary. To simplify the problem and make it well posed, we can reduce it to a one-dimensional problem as follows.

The solution proposed in [37] is to move the contact point based on the tool position and current friction boundary. The vector $\vec{d}$ runs from the friction boundary to the tool position along the line from the friction normal to the tool position. The contact position is moved by $\vec{d}$ projected onto the undeformed surface (Figure 5.6(b)). Essentially, this solution slides the friction boundary over to the virtual tool location. But it does not account for changes in the coefficient of friction, the force field, or friction normal due to moving the contact point which will cause the shape of the friction boundary to change. This is a solution to $(\vec{x} - \vec{c}') \in \Gamma(\vec{c})$.

As shown in Figure 5.6(b), if the friction boundary shape at the new contact position is different than its shape at the original contact position, the tool position may not lie on it. The computed position $\vec{c}'$ is farther to the left in this solution (Figure 5.6(b)) than correct solution (Figure 5.6(a)).

I propose a technique that accounts for these changes and more accurately models the friction properties. As discussed in Section 5.3.1.2, the friction normal in [37] is the same as the Z-axis of the force field interpolation function and is a compromise between modeling friction and modeling the force field. Instead of using the friction normal direction, an independent friction axis $\vec{N}_f$ is extracted from the friction boundary. This axis is found by fitting a cone to the force field sample displacements that are on the friction boundary. The boundary of the friction volume is generally not a cone, but we only need to approximate its axis. It is a cone in force space, but the displacements do not necessarily form a cone because motion of the surface during deformation is not always parallel to the direction of the force being applied. Also, the shape of the boundary is distorted because as the surface deforms the normal changes.

Figure 5.7 shows the friction boundary and extracted axis from the measurements taken at contact location 19 on the latex sheet object without a block. The force field samples are also plotted for reference. The Z-axis is the extracted average friction normal (see Section 5.4.1). The friction boundary shown in red is not a cone, and the extracted friction axis shown as a black line is not the same as the friction normal.

The motion of the contact point is constrained to be along the line that is the intersection of the plane containing the tool position and the friction axis with the undeformed surface. To find the new contact point we minimize

$$h(t) = \left\| \vec{f}_T(\vec{c}', \vec{x} - \vec{c}') - \mu(\vec{c}')\vec{f}_N(\vec{c}', \vec{x} - \vec{c}') \right\|^2 \tag{5.18}$$

$$\vec{c}' = \vec{c} + t\,\vec{l} \tag{5.19}$$

where $\vec{l}$ is a vector pointing in the direction of the line. Minimization is more robust than exactly solving for the location especially when the coefficient of

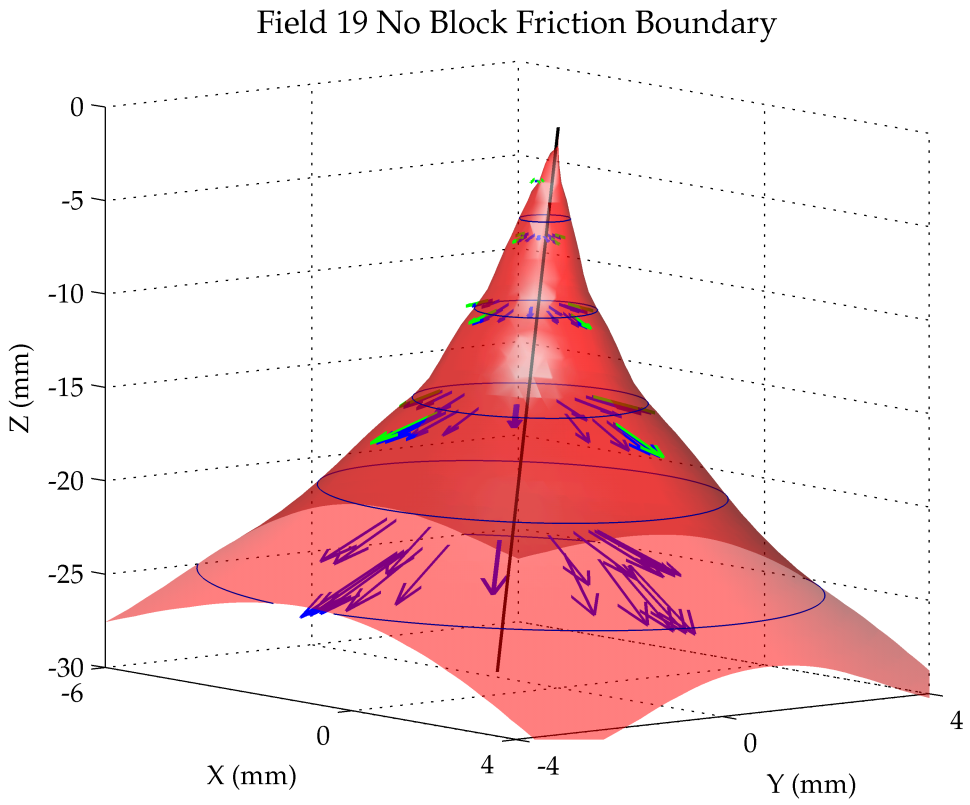


Figure 5.7: Friction constraint volume boundary and axis

friction is zero. There is always a minimum while an exact solution may not exist due to modeling errors or numerical effects. Our extracted friction boundary axis may not go through all points where the tangential component of the force is zero. The actual axis may not even be a straight line. Also, the friction axis will change as the contact position changes.

To ensure a bounded runtime, the minimization is performed using Brent's method. Brent's method is a minimum bracketing algorithm that combines the golden Section method with parabolic interpolation [2]. In each iteration, the interval decreases by at least a fixed amount. As a result, there is a maximum number of iterations for a specified tolerance.

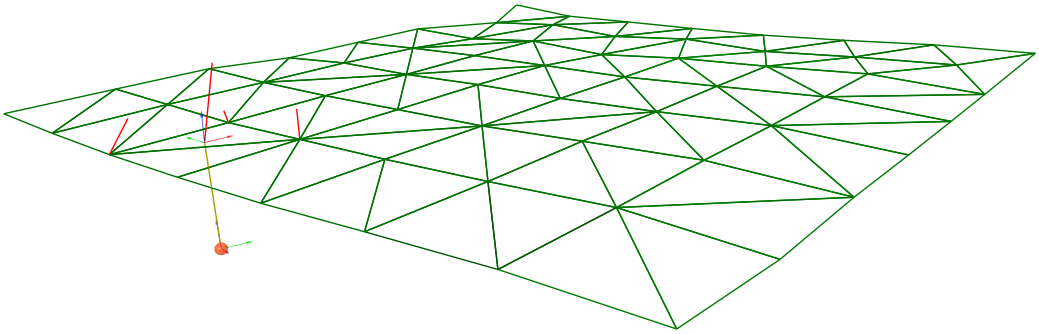
5.5 The Haptic Rendering Loop

5.5.1 Description

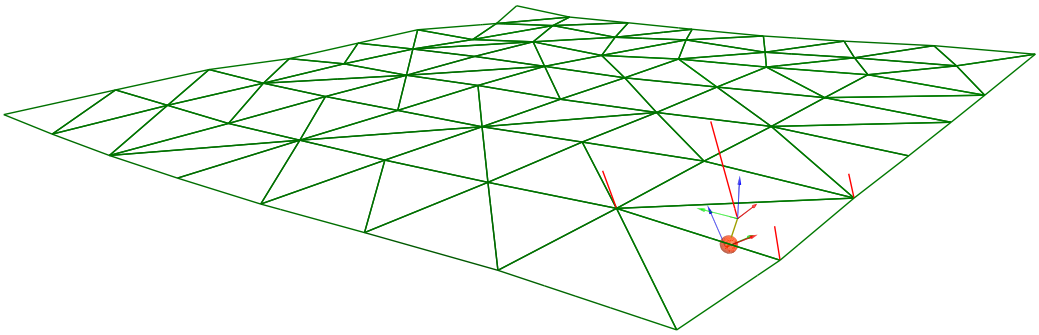
The haptic rendering loop of the data-based model has the same general structure as the proxy and god-object algorithms [47, 64]. Tracking the contact point on the undeformed geometry serves a similar purpose as the proxy or god-object. The analytic restorative spring force is replaced by the force field function and the friction model is replaced by the procedure given in Section 5.4.2.

The state of the model at haptic loop iteration t is the proxy location $\vec{c}_t$, the location of the virtual tool $\vec{x}_t$, and whether the tool is in contact with the object $Touch_t$. If the line connecting the proxy location at the previous iteration $\vec{c}_{t-1}$ and current tool location $\vec{x}_t$ does not intersect the triangulation created in Section 5.3.2.1, the proxy location is updated to be the tool location $\vec{c}_t = \vec{x}_t$, the rendered force is zero $\vec{F} = \vec{0}$, and $Touch_t =$ false. This intersection test is done efficiently using a bounding volume hierarchy.

If there is an intersection at point $\vec{k}$ and the tool was not in contact at the previous iteration ($Touch_{t-1} ==$ false), the proxy location is updated to be the intersection point $\vec{c}_t = \vec{k}$ and is now also the contact point ($Touch_t =$ true). The



(a)



(b)

Figure 5.8: Haptic rendering of latex sheet

force field function (Equation (5.10) and (5.8)) is computed and the friction constraint (Inequality (5.16)) checked. If the constraint is violated, the proxy location (contact point) is updated by minimizing the quantity $h(t)$ defined in Equation (5.18). The rendered force is the value of the force field function evaluated at the current contact position and displacement, i.e. $\vec{F} = \vec{f}(\vec{c}_t, \vec{x}_t - \vec{c}_t)$.

Figure 5.8 shows a visualization of the rendered forces for the latex sheet without a block object. The red lines show the force applied to the user and the force contributions of each force field. The directions of the forces are indicated by the directions of the lines and the magnitudes are proportional to the lengths of the lines. A yellow line connects the tool position to the contact position. When the user is touching an area of the latex sheet that is over the gap (Figure 5.8(a)), the displacement is much larger than when pressing on an area over a wooden support (Figure 5.8(b)) while the user feels a similar force magnitude.

Data set	Force Field Time (µs)	Contact Motion Time (µs)	Total Time (µs)	Mean Num Bases	Proxy Algorithm Time (µs)
Glove	1.03	42.0	48.7	2	7.19
Latex	1.75	38.2	45.6	6.01	10.2
Latex w/ Block	1.81	42.3	49.4	6.38	10.4

Table 5.2: Haptic loop iteration runtime

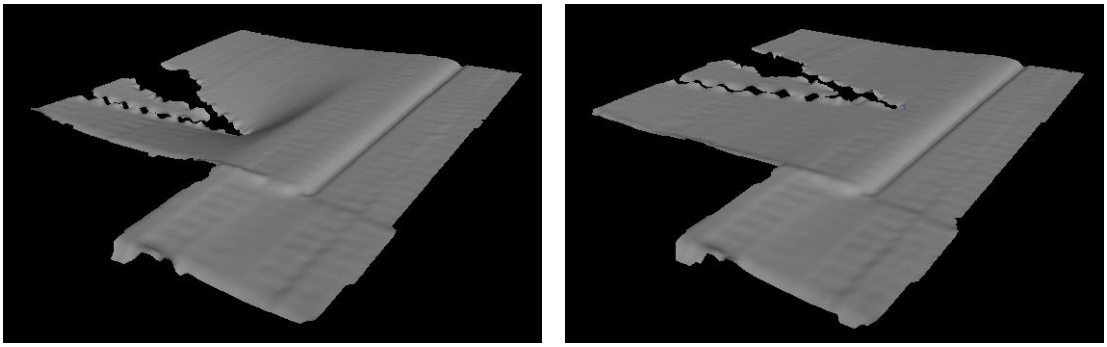
5.5.2 Running time

Table 5.2 shows the average running times of the haptic rendering algorithm for some data sets on a computer with two Xeon 3.06 GHz processors. The time to evaluate the force field function, the time to compute a new contact position, and total time including collision detection are shown. The contact motion times and total times are only averaged over loop iterations where the friction constraint was violated. For reference, the time it took to run the proxy algorithm on the triangulation of contact points was also measured.[1]

The force field function evaluation time is largely dependent on the number of basis fields used. This is expected since evaluating the radial basis function interpolator is $O(B)$ if it has B basis fields. The time to compute motion of the contact point does not show a clear trend. It is dependent on the motion of the tool, characteristics of the force field, and the time it takes to evaluate the force field function. However, the maximum number of iterations for the numerical optimizer is fixed based on the tolerance specified (see Section 5.4.2). In all cases, the times are well below the 1 ms allowed for running the update loop at 1000 Hz.

Collision detection is the only part of the haptic rendering loop whose running time depends on the number of force fields stored in our model. If it is done using a bounding volume hierarchy, its expected running time is $O(\log N)$for a mesh with N triangles. Since our approximate Delaunay triangulation is generated on the undeformed range map, it is a planar graph, and the number of triangles in it is $O(M)$ for a triangulation of M contact points. Thus the overall expected

[1]The proxy algorithm implementation and collision detection routines are from CHAI3D [7].



(a) (b)

Figure 5.9: Simple graphical rendering of latex sheet

running time will be $O(\log M)$ as the number stored force fields increases. In the proxy algorithm, collision detection is also the only part whose running time depends on the number of triangles in the mesh, and its running time will scale similarly.

5.6 Graphical Rendering

5.6.1 Basic Rendering

A simple visual representation of the object in the virtual environment was used in demonstrating the model discussed in this thesis. Whenever the tool is in contact with a triangle, the positions of the probe tip when depth maps associated with the vertex of the triangle that is closest to the current virtual contact location were captured are examined. The depth map acquired when the probe tip was closest to the current virtual tool location is displayed. There is no interpolation of the position of the geometry or filling in holes caused by occlusion. Figure 5.9(a) shows the latex sheet with the user pressing on a soft area, and Figure 5.9(b) shows the sheet when the user presses on the relatively rigid area over the wooden support. The rendering correctly shows deformation or lack of deformation for each case.

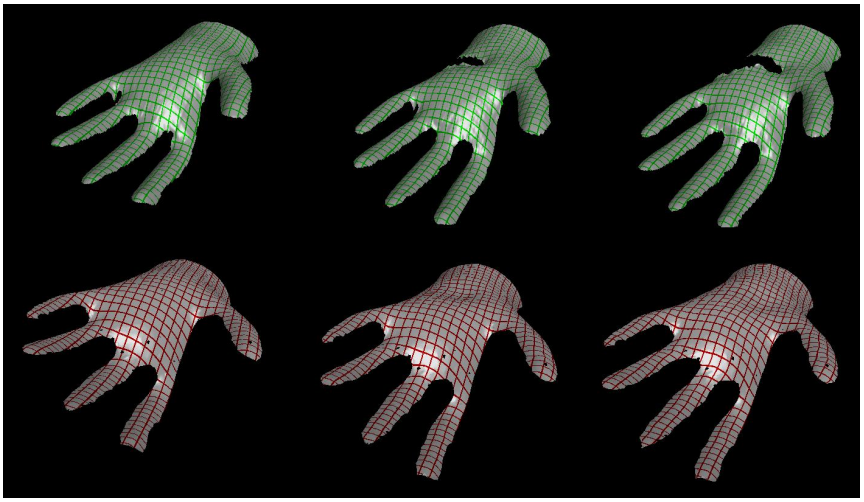


Figure 5.10: Glove depth maps (top row), Reconstructions (bottom row)

5.6.2 Advanced Rendering

In an actual simulator more realistic graphical feedback is desirable. One approach is to take advantage of recent advances in graphical modeling of deformable objects. It is possible to reconstruct the geometry of a deforming object based on surface point sample clouds and find correspondences between the point clouds. In the reconstructed geometry holes caused by occlusion are filled in based on the geometry seen in other frames [61]. Unfortunately, the technique currently does not scale to data sets with large numbers of frames because its global optimization across all frames does not scale well.

This technique was applied to eight depth maps from the glove data set.[2] Figure 5.10 shows renderings of the depth map of the undeformed geometry (left) and two depth maps of the deformed glove (middle and right) in the top row and renderings of the corresponding reconstructed geometry in the bottom row. The grid in the depth maps is not attached to the surface while in the reconstruction the known correspondences allow us to move the grid with the surface. The reconstruction reasonably fills in the occluded parts of the depth maps taken while the robot was deforming the glove. It may also be possible

[2]The data processing was performed by Micheal Wand.

to enhance the depression caused by the probe tip by incorporating the known location of the probe tip into the reconstruction algorithm. The correspondences also appear to be reasonable. Notice the location of the grid appears to be in the same place relative to the fingers in all the reconstructions.

Further research in this area shows promise in scaling it to larger data sets. To use the current algorithm, we could forgo global optimization by matching each depth map of deformed geometry only with the depth map of undeformed geometry. However, this would sacrifice any additional information about areas not seen in the undeformed geometry.

Once reconstructed geometry with correspondences is available, it is possible to take advantage of data-based visual models like the MeshIK method [14, 55]. It interpolates geometry from a set of reference poses given constraints on where some of the vertices should be located. For graphical rendering of our model during interaction, the vertex closest to the contact point on the undeformed geometry would be constrained to be located at the current virtual tool position.

Chapter 6

Conclusion

This thesis presents a complete integrated system for modeling elastic deformable objects. The system consists of two components.

- **Data Acquisition System**: The geometry of an object is captured using a novel structured-light range sensor. The design of the pattern in the sensor allows depth extraction to be made from single images. This allows it to work on fast moving and deforming objects. A method to compensate for chromatic distortions in camera and projector lenses allows the system to be implemented on standard commercial cameras and projectors.
  The behavior of the object and haptic information are captured using an active probing system. A haptic device acting as a robot probes the object to measure its response forces and friction properties. Techniques for slip detection on a deformable surface, contact detection, and enhanced open-loop force control enables the robot to explore the force field at various contact points on the object's surface and record the resulting displacements.

- **Model Creation and Rendering System**: Measurements from the data acquisition system are turned into a data-based model. Force field samples at a contact location are interpolated by a radial basis field interpolator to create a continuous force field function. A Delaunay triangulation of

the measurement contact locations built on the undeformed object's surface enables blending force field functions from distinct contact points to generate force fields at arbitrary contact locations. The force field functions are linearly blended, weighted by the barycentric coordinates of the contact location in the triangulation.

An average friction normal and average static friction coefficient for each contact location are extracted from the measurements to enable modeling of contact point sliding during playback. Together with the force field they define a volume within which the virtual tool position can move without the contact point moving. When the constraint is violated, the contact point must be moved to a location whose friction constraint boundary contains the virtual tool location.

Haptic rendering using the force field interpolation and contact point motion algorithms is computationally efficient. Empirical tests show that it takes less than 50 µs per loop iteration. In addition, the running time scales logarithmically with the number of force field contact locations. This allows it to handle large data sets.

The system has been implemented on hardware readily accessible to haptics researchers and demonstrated with nonlinear inhomogeneous test objects.

6.1 Potential Applications

This work has several potential applications. A few are highlighted here.

6.1.1 Medical Simulators

As discussed in the introduction to this thesis, force feedback provides important information during some medical procedures. Our data-based modeling system was developed with creating a simulator for such procedures in mind.

For example, to create a neck or breast exam simulator the data acquisition system could capture the geometry and force field of an actual neck or breast.

A data-based model would be generated from this data and embedded into the simulator. Although the current data acquisition system is too slow for in vivo measurement (see Section 6.2.1), it could be used on animal or cadaver organs in vitro. This approach could facilitate the implementation of realistic simulators for intra-operative liver or lung palpation. A liver palpation simulator was described in [16] but used a hand-created parametric model. Parameters of the model were set based on force data collected from rubber balls used to simulate the feel of a tumor.

6.1.2 Creation of Parametric Models

Parametric models and data-based models have different strengths and weaknesses (see Section 2.1). When the modeled object's behavior is simple enough for an interactive parametric model, a parametric model may be preferable for some applications such as situations requiring dynamics or generalization of behavior.

The data collected by the data acquisition system can be used to set the parameters of a parametric model. Previous work on automatically capturing a model from real objects used a complex setup of sensors and actuators including multiple robot arms, force sensors, microphones, and a trinocular camera [35, 43]. Aside from measuring sound, the data acquisition system described in Chapters 3 and 4 produces similar data. In [38], the parameters of a non-constitutive parametric model were automatically set based on a reference FEM simulation. With our data acquisition system, measurements from a real object could be substituted for the simulation.

6.1.3 Study of Deformable Object Motion

The range sensor is capable of capturing the motion of fast moving and deforming objects. It could be used to add depth information in situations where high-speed cameras are used. For example, high-speed cameras record vehicle collision tests. Such tests are costly, and it is desirable to get as much data as possible from each test. Adding a projected pattern and using the algorithms in

Chapter 3 would give additional information in the form of the geometry of the deformation of parts of the vehicle as it collides.

This information can be used many ways. One possibility is to improve or verify the performance of models used in vehicle collision simulations. Such simulations are less costly than actual crash tests and enable designers to test their designs without having to physically build them. These models and simulators can also provide information such as where stresses are located in the structure.

6.2 Limitations and Extensions

6.2.1 Data Acquisition Time

While the haptic rendering algorithm can scale to large data sets (see Section 5.5.2), acquiring data does not. It takes a fairly long time to collect force and friction information. The latex rubber sheet data sets each contain about 5300 force field samples, and it took about 17 hours to collect each data set. This is about 12 seconds per sample. The time to collect a force field sample can be broken down into three parts: time to position the probe tip, time spent waiting for equilibrium, and time spent waiting for the motors to cool. For the collection of the data sets discussed in this thesis, the trajectory generator in the controller of the robot was set to move the probe tip at 30 mm/s.

As discussed in Section 4.4, before measurement can be made after applying a force, we must wait for equilibrium because our model is quasi-static. The time it takes for the robot and object to reach equilibrium after applying a force depends on the object being probed. In general, since our model is quasi-static, the object should not have too much dynamic behavior. The latex sheet object has a small amount of damping and reaches equilibrium fairly quickly. However in tests with some foam objects, there was still motion after 8 seconds of waiting. This is far from quasi-static, and our model and data acquisition methods are not suitable for this type of object.

When the PHANToM applies forces over about 1.5 N, its motors heat up.

Although it is not equipped with temperature sensors, the programming interface maintains an internal estimate of the temperature. If this estimate exceeds a threshold, it shuts off the motors. In order to prevent damage to the motors and avoid this shutdown, the temperature estimate must be monitored, and a cool off period must be observed until it falls below a safe threshold. This limitation also means the PHANToM cannot be used to apply forces over 1.5 N for more than a few seconds.

Given these limitations, a better means to reducing the data acquisition time is to reduce the number of force field samples acquired. However, the behavior of a data-based model depends directly on the data stored in it (see Section 2.1). Ideally, we would collect the minimum amount of data necessary for the model to exhibit the desired range of behavior. Three possible approaches to doing so are adaptive sampling, more advanced response synthesis routines, and precomputation using parametric models.

6.2.1.1 Adaptive Sampling

In the current implementation of the data acquisition process, measurement contact locations are distributed approximately uniformly over the object's surface, and a fixed number of force field samples are acquired at each contact location. However, each additional piece of data does not add the same amount of additional behavior information to the model. Moreover, after collecting a certain quantity of data, collecting additional data provides diminishing returns. The force field interpolation experiment of Section 5.3.3 suggests that the value of a particular data sample depends on its location relative to other data samples and the behavior of the object.

The number of contact locations where measurements are made and the number of force field samples can be reduced by adaptive sampling. Measurement contact locations should be denser in areas of the object where the behavior varies more. For example, in the latex sheet test object, measurements should be made at more contact locations near the transition between the freely suspended area and the area supported by wood. This could be done by looking at how the

geometry deforms or looking at the error between the measurements and the interpolation.

The motion of the surface when the object deforms varies based on the object's composition. In the latex sheet object, the portion of the latex that is over open space moves farther than the portion over wood. The difference in motion indicates a possible difference in the force fields of the two areas. This kind of surface motion analysis could guide the placement of additional measurement contact locations. The current data acquisition system does not use information from deformed geometry to guide probing at all. Deformed geometry is merely data that is stored for visual rendering at playback.

The difference between interpolated values and measured values indicates how well the model approximates the force field (and hence behavior) of the object. Whenever a measurement is made it could be compared to the interpolated value generated based on all the previous measurements. If the interpolated value is similar to the measured value, it is reasonable to expect that additional measurements nearby are not needed. In most objects, nearby parts influence each other's behavior because they are connected together. If the interpolated value at a location matches with a measured value, it is likely that the behavior of locations between the measurement location and the locations on which the interpolation are based also will agree with the model. This could be applied both to taking samples at a particular contact location and to selecting new contact locations.

6.2.1.2 Advanced Synthesis Algorithms

A parametric model can generate complex behavior from relatively simple stored data (parameters) because its behavior synthesis routines are complex, and the underlying model limits the range of possible behaviors (see Section 2.1). These routines are more powerful because they incorporate (possibly non-physical) knowledge about how the world works. Our data-based model (and data-based models in general) which uses simple interpolation for rendering runs very fast (see Section 5.5.2). A more complex interpolation procedure that

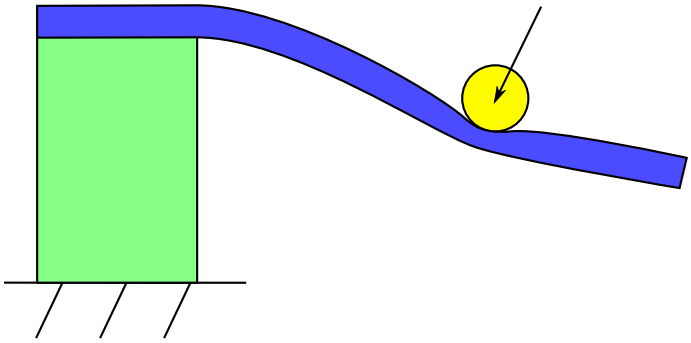


Figure 6.1: Bending a foam beam

incorporates prior physical knowledge about the object's behavior would require more runtime computation, but would need less data to be stored. Hence, by blending aspects of parametric models and data-based models, trade-offs could be made between acquisition and rendering times.

For example, the response synthesis routines could incorporate additional information about the structure of the force field. We could decompose the force field of an object into local and global components. Local forces are due to the object locally deforming while global forces are due to the global motion of the object. The two parts could be measured separately and combined during haptic rendering. If the local part is similar over the whole object, then fewer samples are needed to capture the local behavior. The data acquisition system could densely sample at a few contact locations to capture the local force field behavior while sampling less densely at other contact locations to capture the global behavior.

Imagine a foam beam mounted at one end and cantilevered outward (Figure 6.1). Pressing on it will cause both a deformation around the probe tip and overall bending. The forces due to the indentation near the probe tip are likely to be similar as long the contact location is on the cantilevered part while the forces due to the bending vary with the distance from the mounted end. The inflated glove test object behaves similarly (Figure 4.9). The indentation near the probe tip is similar at all contact locations while how much the glove bends depends on the contact location.

6.2.1.3 Precomputation using Parametric Models

The amount of data needed to determine the parameters of a model is related to the complexity of the model and number of parameters in the model. For example, a linear spring model has one parameter, the spring constant. Given one displacement and force measurement, we can determine the value of the spring constant. As discussed above (Section 6.1.2), if there is a parametric model that is capable of modeling the behaviors desired for an application, the parameters for the model could be extracted from measurements. The interactive simulation could then use the parametric model directly.

However, if the model is too slow to run interactively, we could capture its behavior in a data-based model and use the data-based model for rendering. Solutions to the parametric model could be precomputed and stored into the data-based model in the same manner as measurements. Essentially, the parametric model would be used to generalize from the measurements and generate data for the data-based model. Then the data-based model would act as a generic behavior interpolator. This would be faster than taking actual measurements as long as the precomputation is fast enough. Fortunately, it is easier to parallelize precomputation than measurement.

6.2.2 Geometry Model Data Size

Much more space is needed to store geometry than to store force and friction information. Since global geometry is needed for visual feedback, it is necessary to store geometry for the whole (side) of an object for various sample tool locations. Conversely, force feedback is local and only the force at the tool location is needed. In our model, the force field interpolator, friction properties, and contact point triangulation take less storage space than a single depth map. The force and friction information for the latex sheet take less than 100 KBytes of storage while a single depth map is about 730 KBytes. Downsampling the depth maps and using more advanced graphical rendering techniques such as those in Section 5.6.2 would reduce the storage requirements for the geometry data. But it

would still be much larger than the haptic information.

Another approach is to use a parametric model for visual feedback while using a data-based model for haptic feedback. A typical parametric visual model needs to store the undeformed geometry and some parameters. The number of parameters may be proportional to the number of elements needed to represent the geometry such as masses, spring constants, and damping constants in a mass-spring model. This is much less than storing the deformed geometry for a whole range of possible tool locations. Since graphics update rates are much lower than haptics update rates (30–60 Hz vs 1000 Hz), a parametric visual model can be more complex than a parametric haptics model. The challenge for this approach is to make the haptic and visual models consistent so that the user is not confused by conflicting feedback.

6.2.3 Multi-sided Models

The current implementation of the data-based modeling system produces single-sided models. These models are useful for situations where interaction is constrained to be from one side such as pressing on objects on a table or organs that have only been exposed on one side.

However, it is possible to create multi-sided models. The force field is parameterized by contact locations on the undeformed geometry. Multiple single-sided models could be combined by merging the undeformed geometry using existing techniques in geometric modeling (see Section 2.2.1.3) and applying the same transforms to the contact locations.

6.2.4 Generalizing Behavior

A disadvantage of data-based modeling approaches in general is their limited ability to generalize behavior in new situations. However, there are a few situations where the model could be easily extended.

6.2.4.1 Multiple Points of Contact

In general, if the data used to create a model only contains examples with a single point of contact, it is not possible to generalize the behavior to multiple points of contact. Interacting with an object at one location can globally change its behavior. Pressing on a water balloon at one point completely changes how pressing on another part at the same time feels. In addition, the second interaction changes what is felt at the first location. To capture this behavior, the location and displacement of the additional interactions would need to be added as parameters of the haptic force field function. This increases the dimensionality of the space in which we need to sample, and the number of measurements needed scales exponentially in the number of dimensions.

However, in some objects the effects of interactions are local. In a block of foam, pressing on one location generally only affects the area around that location. The behavior of the rest of the block is unchanged. In this case, we could merely use the same haptic force field function for the additional points of interaction as long as the points of interaction are not too close together. The higher dimensional force field function would still be needed if the contact points are close together, but this limits the area and volume in which additional measurements need to be made. If only a fixed additional amount of sampling is needed for each contact point, the number of measurements scales linearly with the number of interaction points.

6.2.4.2 Transferring Behavior to New Geometry

It is sometimes desirable to change the geometry of an object while retaining how it feels. There has already been work on transferring various properties such as geometric deformation and texture maps from one undeformed geometry to another [15, 54].

Transferring the feel can be done because our force field function is parameterized by contact location on the undeformed geometry. Transferring it to new

geometry means finding a mapping from locations on the original surface to locations on the new surface. Since the force field also implicitly depends on the interior of the object, whether such a mapping produces a reasonable result is dependent on the object. In some cases applying transforms to the force fields may also be necessary. In others where the force field is very dependent on underlying geometry, it might not be possible to create a sensible mapping at all.

Simple transformations are most likely to be successful. For example, if we have created a model of a foam cube and would like to extend the model to a larger foam cube, we could scale the cube and its force fields. We could also create a rectangular box by scaling only one dimension. However, mapping the cube onto a sphere may cause the areas of the sphere that were mapped to the corners of the cube to feel incorrect. A more practical example is taking generic models of organs and mapping them to the patient-specific geometry. If we had a liver model and wanted to simulate a particular patient, the force field of the original liver model would need to be mapped onto the geometry of the patient's liver. If their geometries are similar, the result is likely to be reasonable.

Bibliography

[1] Alexa, M. Linear combination of transformations. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2002), ACM Press, pp. 380–387.

[2] Bochkanov, S., and Bystritsky, V. ALGLIB. Website. `http://www.alglib.net/optimization/brent.php`.

[3] Bouguet, J.-Y. Camera calibration toolbox for matlab. Website. `http://www.vision.caltech.edu/bouguetj/calib_doc/`.

[4] Buron, F. High-resolution three-dimensional sensing of fast deforming objects. Master's thesis, Stanford University, 2005.

[5] Carrihill, B., and Hummel, R. Experiments with the intensity ratio data sensor. *Computer Vision, Graphics, and Image Processing 32*, 3 (Dec. 1985), 337–358.

[6] Caspi, D., Kiryati, N., and Shamir, J. Range imaging with adaptive color structured light. *IEEE Trans. Pattern Anal. Mach. Intell. 20*, 5 (1998), 470–480.

[7] Conti, F., Barbagli, F., Morris, D., and Sewell, C. CHAI: An open-source library for the rapid development of haptic scenes. In *IEEE World Haptics* (Mar. 2005).

[8] Conti, F., Khatib, O., and Baur, C. Interactive rendering of deformable objects based on a filling sphere modeling approach. In *Robotics and Automation, 2003. Proceedings. ICRA '03. IEEE International Conference on* (2003), vol. 3, pp. 3716–3721.

[9] Cotin, S., Delingette, H., and Ayache, N. Real-time elastic deformations of soft tissues for surgery simulation. *IEEE Transactions on Visualization and Computer Graphics 5*, 1 (1999), 62–73.

[10] Curless, B., and Levoy, M. A volumetric method for building complex models from range images. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1996), ACM Press, pp. 303–312.

[11] Cutkosky, M. R., and Hyde, J. Manipulation control with dynamic tactile sensing. In *Proceedings of the Sixth International Symposium on Robotics Research* (Oct. 1993), T. Kanade, Ed., M.I.T. Press.

[12] Davis, J., Marschner, S., Garr, M., and Levoy, M. Filling holes in complex surfaces using volumetric diffusion. In *3D Data Processing Visualization and Transmission. Proceedings. First International Symposium on* (2002), pp. 428–861.

[13] Davis, J., Ramamoorthi, R., and Rusinkiewicz, S. Spacetime stereo: a unifying framework for depth from triangulation. In *Computer Vision and Pattern Recognition. Proceedings. 2003 IEEE Computer Society Conference on* (2003), vol. 2, pp. II–359–66.

[14] Der, K. G., Sumner, R. W., and Popović, J. Inverse kinematics for reduced deformable models. In *SIGGRAPH '06: ACM SIGGRAPH 2006 Papers* (New York, NY, USA, 2006), ACM Press, pp. 1174–1179.

[15] Dinh, H. Q., Yezzi, A., and Turk, G. Texture transfer during shape transformation. *ACM Trans. Graph. 24*, 2 (2005), 289–310.

[16] Dinsmore, M., Langrana, N., Burdea, G., and Ladeji, J. Virtual reality training simulation for palpation of subsurface tumors. In *VRAIS '97: Proceedings of the 1997 Virtual Reality Annual International Symposium* (Washington, DC, USA, 1997), IEEE Computer Society, p. 54.

[17] Fong, P., and Buron, F. High-resolution three-dimensional sensing of fast deforming objects. In *Intelligent Robots and Systems. (IROS 2005). 2005 IEEE/RSJ International Conference on* (Aug. 2005), pp. 1606–1611.

[18] Fong, P., and Buron, F. Sensing deforming and moving objects with commercial off the shelf hardware. In *Computer Vision and Pattern Recognition (CVPR). 2005 IEEE Computer Society Conference on* (2005), vol. 3, pp. 101–108.

[19] Gallier, J., and Xu, D. Computing exponentials of skew symmetric matrices and logarithms of orthogonal matrices. *International Journal of Robotics and Automation 18*, 1 (2003), 10–20.

[20] Ghiglia, D., and Pritt, M. D. *Two-dimensional Phase Unwrapping: Theory, Algorithms, and Software*. Wiley, 1998.

[21] Gibson, S. F. F., and Mirtich, B. A survey of deformable modeling in computer graphics. Tech. Rep. TR-97-19, Mitsubishi Electric Research Laboratory, Nov. 1997.

[22] Girosi, F., Jones, M., and Poggia, T. Priors, stabilizers and basis functions: from regularization to radial, tensor and additive splines. Tech. Rep. 1430, Massachusetts Institute of Technology Artificial Intelligence Laboratory, June 1993.

[23] Goldberg, C., and Thompson, J. A practical guide to clinical medicine. Website, 2005. `http://medicine.ucsd.edu/clinicalmed/`.

[24] Gonzalez-Banos, H., and Davis, J. Computing depth under ambient illumination using multi-shuttered light. In *Computer Vision and Pattern Recognition.*

CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on (2004), vol. 2, pp. 234–241.

[25] Green, D. F. Haptic simulation of naturally occuring textures and soil properties. Master's thesis, Massachusetts Institute of Technology, 1998.

[26] Haluck, R. S., Marshall, R. L., Krummel, T. M., and Melkonian, M. G. Are surgery programs ready for virtual reality? A survey of program directors in general surgery. *Journal of the American College of Surgeons 193*, 6 (Dec. 2001), 600–665.

[27] Hodgins, J. K., O'Brien, J. F., and Bodenheimer, R. E. Computer animation. In *Wiley Encyclopedia of Electrical and Electronic Engineering*, J. G. Webster, Ed. John Wiley, 1999, pp. 686–690.

[28] James, D. L., and Fatahalian, K. Precomputing interactive dynamic deformable scenes. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers* (New York, NY, USA, 2003), ACM Press, pp. 879–887.

[29] James, D. L., and Pai, D. K. Artdefo: accurate real time deformable objects. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1999), ACM Press/Addison-Wesley Publishing Co., pp. 65–72.

[30] Kass, M., Witkin, A., and Terzopoulos, D. Snakes: Active contour models. *International Journal of Computer Vision 1*, 4 (Jan. 1988), 321–331.

[31] Koninckx, T. P., Griesser, A., and Gool, L. V. Real-time range scanning of deformable surfaces by adaptively coded structured light. In *Fourth Int. Conf. on 3-D Digital Imaging and Modeling - 3DIM03* (2003).

[32] Kuchenbecker, K. J., Fiene, J., and Niemeyer, G. Improving contact realism through event-based haptic feedback. *IEEE Transactions on Visualization and Computer Graphics 12*, 2 (2006), 219–230.

[33] Lee, J., and Lee, K. H. Precomputing avatar behavior from human motion data. In *SCA '04: Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation* (2004), Eurographics Association, pp. 79–87.

[34] Liu, W., Wang, Z., Mu, G., and Fang, Z. A novel profilometry with color-coded project grating and its application in 3d reconstruction. In *Communications. APCC/OECC '99. Fifth Asia-Pacific Conference on ... and Fourth Optoelectronics and Communications Conference* (1999), vol. 2, pp. 1039–1042.

[35] Lloyd, J. E., and Pai, D. K. Robotic mapping of friction and roughness for reality-based modeling. *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation 2* (2001), 1884–1890.

[36] Mahvash, M., and Hayward, V. High-fidelity haptic synthesis of contact with deformable bodies. *IEEE Computer Graphics and Applications 24*, 2 (2004), 48–55.

[37] Mahvash, M., and Hayward, V. High-fidelity passive force-reflecting virtual environments. *Robotics, IEEE Transactions on 21*, 1 (Feb. 2005), 38–46.

[38] Morris, D. *Haptics and Physical Simulation for Virtual Bone Surgery*. PhD thesis, Stanford University, Aug. 2006.

[39] Mussa-Ivaldi, F. A. From basis functions to basis fields: vector field approximation from sparse data. *Biological Cybernetics 67*, 6 (Oct. 1992), 479–489.

[40] Ng, R., Levoy, M., Brédif, M., Duval, G., Horowitz, M., and Hanrahan, P. Light field photography with a hand-held plenoptic camera. Tech. Rep. CSTR 2005-02, Stanford University Computer Science, Apr. 2005.

[41] O'Brien, J. F., Bodenheimer, R., Brostow, G. J., and Hodgins, J. K. Automatic joint parameter estimation from magnetic motion capture data. In *Graphics Interface* (May 2000), pp. 53–60.

[42] Ottensmeyer, M. *Minimally invasive instrument for in vivo measurement of solid organ mechanical impedance*. PhD thesis, Massachusetts Institute of Technology, 2001.

[43] Pai, D. K., van den Doel, K., James, D. L., Lang, J., Lloyd, J. E., Richmond, J. L., and Yau, S. H. Scanning physical interaction behavior of 3d objects. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 2001), ACM Press, pp. 87–96.

[44] Parsons, A. M., Detterbeck, F. C., and Parker, L. A. Accuracy of helical ct in the detection of pulmonary metastases: Is intraoperative palpation still necessary? *Ann Thorac Surg 78*, 6 (2004), 1910–1918.

[45] Pritchard, D., and Heidrich, W. Cloth motion capture. In *Eurographics* (Sept. 2003), pp. 263–271.

[46] Rusinkiewicz, S., and Levoy, M. Efficient variants of the ICP algorithm. In *Proceedings of the Third Intl. Conf. on 3D Digital Imaging and Modeling* (2001), pp. 145–152.

[47] Ruspini, D. C., Kolarov, K., and Khatib, O. The haptic display of complex graphical environments. In *Computer Graphics (SIGGRAPH 97 Conference Proceedings)* (1997), ACM SIGGRAPH, pp. 345–352.

[48] Salisbury, K., Brock, D., Massie, T., Swarup, N., and Zilles, C. Haptic rendering: programming touch interaction with virtual objects. In *SI3D '95: Proceedings of the 1995 symposium on Interactive 3D graphics* (New York, NY, USA, 1995), ACM Press, pp. 123–130.

[49] Sansoni, G., Biancardi, L., Docchio, F., and Minoni, U. Comparative analysis of low-pass filters for the demodulation of projected gratings in 3-d adaptive profilometry. *Instrumentation and Measurement, IEEE Transactions on 43*, 1 (1994), 50–55.

[50] Sansoni, G., Carocci, M., and Rodella, R. Calibration and performance evaluation of a 3-d imaging sensor based on the projection of structured light. *Instrumentation and Measurement, IEEE Transactions on 49*, 3 (2000), 628–636.

[51] Sethian, J. A. Fast marching methods. *SIAM Review 41*, 2 (1999), 199–235.

[52] Sewell, C. *Automatic Performance Evaluation in Surgical Simulation*. PhD thesis, Stanford University, Mar. 2007.

[53] Shum, H.-Y., and Kang, S. B. Review of image-based rendering techniques. In *Proc. SPIE Visual Communications and Image Processing 2000* (May 2000), K. N. Ngan, T. Sikora, and M.-T. Sun, Eds., vol. 4067, pp. 2–13.

[54] Sumner, R. W., and Popović, J. Deformation transfer for triangle meshes. In *SIGGRAPH '04: ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), ACM Press, pp. 399–405.

[55] Sumner, R. W., Zwicker, M., Gotsman, C., and Popović, J. Mesh-based inverse kinematics. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers* (New York, NY, USA, 2005), ACM Press, pp. 488–495.

[56] Takeda, M., and Mutoh, K. Fourier transform profilometry for the automatic measurement of 3-d object shapes. *Applied Optics 22* (Dec. 1983), 3977–3982.

[57] Tang, S., and Hung, Y. Y. Fast profilometer for the automatic measurement of 3-d object shapes. *Applied Optics 29* (July 1990), 3012–3018.

[58] Terzopoulos, D., Platt, J., Barr, A., and Fleischer, K. Elastically deformable models. In *SIGGRAPH '87: Proceedings of the 14th annual conference on Computer graphics and interactive techniques* (New York, NY, USA, 1987), ACM Press, pp. 205–214.

[59] Teschner, M., Heidelberger, B., Muller, M., and Gross, M. A versatile and robust model for geometrically complex deformable solids. In *CGI '04:*

Proceedings of the Computer Graphics International (CGI'04) (Washington, DC, USA, 2004), IEEE Computer Society, pp. 312–319.

[60] Trucco, E., and Verri, A. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[61] Wand, M., Jenke, P., Huang, Q., Bokeloh, M., Guibas, L., and Schilling, A. Reconstruction of deforming geometry from time-varying point clouds. In *Proc. 5th Eurographics Symposium on Geometry Processing (SGP) 2007* (July 2007).

[62] Wilburn, B., Joshi, N., Vaish, V., Talvala, E.-V., Antunez, E., Barth, A., Adams, A., Horowitz, M., and Levoy, M. High performance imaging using large camera arrays. *ACM Trans. Graph. 24*, 3 (2005), 765–776.

[63] Zhang, L., Curless, B., and Seitz, S. M. Rapid shape acquisition using color structured lightand multi-pass dynamic programming. In *The 1st IEEE International Symposium on 3D Data Processing, Visualization, and Transmission* (June 2002).

[64] Zilles, C., and Salisbury, J. A constraint-based god-object method for haptic display. In *Intelligent Robots and Systems. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on* (1995), vol. 3, pp. 146–151.