

HAPTIC EXPLORATION OF UNKNOWN OBJECTS

A DISSERTATION

SUBMITTED TO THE DEPARTMENT OF MECHANICAL ENGINEERING

AND THE COMMITTEE ON GRADUATE STUDIES

OF STANFORD UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

Allison Mariko Okamura

June 2000

© Copyright by Allison Mariko Okamura 2000
All Rights Reserved

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.

Mark R. Cutkosky
(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.

Bernard Roth

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
(Computer Science)

Approved for the University Committee on Graduate Studies:

Abstract

Haptic exploration is a key mechanism humans use to learn about the surface properties of unknown objects. With specialized fingers and sensors, and the appropriate planning and control, robots can also be enabled to explore the world through touch. Haptic exploration has applications in many areas, including planetary exploration, undersea salvage, and other operations in remote or hazardous environments.

This thesis develops an approach for haptic exploration of unknown objects by robotic fingers. Because haptic exploration is coupled with manipulation, a procedure for combined manipulation and exploration using a sequence of phases is presented. Fingers alternately grasp and stabilize the object while other fingers explore the surface with rolling and sliding motions. During an exploratory phase, the goal is to move a fingers tactile sensors over the surface in a way that will elicit useful data.

There exist many possible objectives for haptic exploration. This work concentrates on the detection and identification of fine surface features. In the context of exploration with spherical robotic fingertips, fine surface features and macro features such as bumps, cracks and ridges are defined. Using different types of sensor data, various algorithms and experimental results for fine feature detection are presented. There are also many potential methods for actively exploring a feature on a surface in three dimensions. After a feature has been encountered on a surface, tactile sensor and position data may be used to determine the next direction of finger travel, guiding the finger around and over the feature in a way that will efficiently extract surface properties. Shape skeletons are used to create a map of features and regions on a surface.

Acknowledgements

There are many people who have helped make my time at Stanford enjoyable, and I am deeply grateful to each one of them. First and foremost, I thank my research advisor, Professor Mark Cutkosky, for his guidance over the past six years. His keen insights made research exciting, and his supportiveness made him an excellent advisor. The other members of the reading committee, Professors Bernard Roth and Oussama Khatib, also contributed advice in both research and life matters that I will always appreciate. Bernie's patient and thorough editing has greatly improved this work. Professor Ken Salisbury also provided useful feedback as a member of my defense committee, and I am grateful to Professor Carlo Tomasi for acting as Chair.

I am indebted to the National Science Foundation, the Office of Naval Research, and Immersion Corporation for supporting my research. The Mechanical Engineering Lab in Tsukuba gave me the opportunity to visit Japan and provided me with some of the vital hardware used in this work. Special thanks are due to Drs. Kazuo Tanie and Hitoshi Maekawa of MEL. I also thank Professor Jack Dennerlein of Harvard University for several enjoyable collaborations.

The camaraderie and good-natured ribbing between the members of the Dexterous Manipulation Laboratory made working in the lab even more amusing than Jorge Cham's Ph.D. comic strip. I especially appreciate lab mates Michael Costa, Christopher Richard, and Michael Turner for years of lab banter, and whipping me at Yahoo! Spades. I thank "The Brotherhood" (Sean Bailey, Jorge Cham and Weston Griffin), for being wonderful company and great mountain biking/ice hockey coaches. I hope they continue to eat meat often. I am grateful to Matt Hage for his assistance in the last year and wish the best of luck to him and the rest of the aspiring academics in the DML: Wendy Cheng, Jonathan Clark, Chris Hasser, Moto Hatanaka, Michael Koenig, Pratik Nahata, Will Provancher, and Neils Smaby. I feel lucky to have been a member of such a fun lab.

I will also miss my friends and colleagues at the Center for Design Research. "The

Secret CDR Sisterhood” (Krista Donaldson, Beth Pruitt and Maria Yang) taught me value of listening and was always there for me through good times and bad. Sanjay Rajagopalan was also a caring friend. Over the years, I came to fully appreciate the hard work of the administration of CDR, particularly Noelle Rudolph, Debra Frank, and Jeff Aldrich.

Outside of the lab, I was lucky to get to know Dr. Sheri Sheppard and Dr. Laura Demsetz, and travel down the Ph.D. path with Dr. Sile O’Modhrain. The Mechanical Engineering Graduate Women’s Group helped me think about issues larger than research, and working out with the Stanford Wushu Club was a necessary challenge twice a week. Other good friends along the path include: Al Cho, Aaron Barzilai, Elizabeth Lobo-Polefka, Priti Brahma, Tuni Kundu, Mark Evans, Professor Dennis Lieu, and Jim Young. My childhood companion Jennifer King was crazy enough to agree to proofread my thesis; this is a sure sign of a true friend.

Last, but certainly not least, I am thankful beyond words for the encouragement I have received from my parents, Bill and Judy, and my sister, Cindy. Thank you for supporting and cheering me on in all the tasks I undertake.

Contents

Abstract	iv
Acknowledgements	v
1 Introduction	1
1.1 Motivation and Challenges	4
1.1.1 Motivation	4
1.1.2 Challenges	5
1.2 Related Work	6
1.3 Contributions	8
1.4 Thesis Overview	8
2 A Procedure for Multifingered Exploration	10
2.1 Related Work	12
2.2 Exploratory Procedure Algorithm	13
2.2.1 Algorithm and States	13
2.2.2 Phase 1 - Cooperative Manipulation	18
2.2.3 Phases 2 and 3 - Exploration	25
2.3 Simulations and Experiments	27
2.3.1 Simulations	27
2.3.2 Experiments	29
2.4 Conclusion	32
3 A Feature Definition for Haptic Exploration	34
3.1 Previous Work	35
3.2 Defining Surface Features	36
3.2.1 Differential Geometry Surface Descriptions	36

3.2.2	A New Feature Definition	42
3.2.3	Example: A Bump Feature	46
3.3	Feature Detection	48
3.3.1	Using Tactile Sensor Data	49
3.3.2	Using Fingertip Center Position Data	50
3.3.3	Tactile Data Smoothing	51
3.4	Simulation and Experiments	52
3.4.1	Simulated Data for Surface Estimation	52
3.4.2	Experiments for Feature Detection	53
3.5	Conclusion	56
4	Feature-Guided Exploration	58
4.1	Previous Work	59
4.1.1	Tactile Servoing and Hybrid Control	59
4.1.2	Object Modeling from Data	59
4.1.3	Skeletonization	60
4.2	A Procedure for Feature-Guided Exploration	61
4.2.1	Goals and Assumptions	63
4.3	Control for Haptic Exploration	65
4.3.1	Hardware	65
4.3.2	Coordinate Systems	65
4.3.3	PD and Normal Force Control	68
4.4	Local Exploration	71
4.4.1	Feature Type Detection	71
4.4.2	Feature Tracing	71
4.4.3	Feature Modeling with Shape Skeletons	73
4.5	Global Exploration	79
4.5.1	Goal Feature Considerations	79
4.5.2	Feature Spacing Considerations	82
4.5.3	Partitioning using Feature and Global Skeletons	83
4.6	Conclusion	86
5	Conclusions	87
5.1	Summary of Results	87
5.2	Review of Contributions	88

5.3	Future Work	89
A	Kinematics of Contact	91
A.1	A Differential Geometry Surface Description	91
A.2	Contact Kinematics	95
B	Smoothing Tactile Data Using Noise Type	98
B.1	Noise Analysis	98
B.2	Curvature Calculation and Feature Detection Algorithm	102
B.3	Smoothing Techniques	103
B.4	Filter Experimental Results	107
B.5	Summary	107
C	A 3DOF Robotic Finger with Tactile Sensing	109
C.1	The Optical Waveguide Tactile Sensor	109
C.1.1	Sensor Function	109
C.1.2	Calculating Contact Point	110
C.1.3	Advantages and Disadvantages	112
C.2	The 3GM as a Robotic Finger	113
C.2.1	Hardware	113
C.2.2	Device Control	113
	Bibliography	119

List of Tables

2.1	Contact velocity constraints for pure rolling in the $x - z$ plane.	20
3.1	A partial list of possible macro features. In the 2D picture (a plan view of the surface), the gray regions indicate curvature features, with the necessary principal curvatures labeled. In this table, + indicates a positive curvature feature, - indicates a negative curvature feature, and ε is no curvature feature ($ k_i < \frac{1}{r_f}$). It is assumed that $ k_1 \geq k_2 $. The white regions between the curvature feature regions have a maximum width of r_f , the radius of the fingertip.	44
B.1	Noise analysis parameters for tactile data.	102

List of Figures

1.1	Exploration and manipulation are necessarily coupled during the exploration of unknown or partially known objects.	2
1.2	Klatzky and Lederman demonstrated that humans use a number of exploratory procedures (EPs) to determine unknown object properties. (Adapted from [41].)	3
2.1	Combined manipulation and exploration of an object using multiple fingers.	11
2.2	Two robotic fingers and an object with haptic features.	12
2.3	States and transitions of the exploratory procedure for two active fingers and a passive palm.	14
2.4	A circular object and the corresponding grasp map for two-fingered grasps. The workspace map regions corresponding to different pairs of fingers are overlaid on the grasp map. The configuration is two active fingers and a passive palm, and the coefficient of friction between the object and the fingers is 1.	18
2.5	Common contact types and their associated forces/moments.	19
2.6	The friction cone is a geometric description of the point contact friction constraint.	20
2.7	Control block diagram including rolling feedback control.	22
2.8	Explicit rolling can be commanded when there are sufficient degrees-of-freedom in the robot hand, such as this system of two fingers with three degrees-of-freedom each.	23
2.9	Model of explicit rolling during a simulation performed using Matlab.	24
2.10	Explicit rolling control experimental results.	24
2.11	Examples of (a) unstable and (b) stable two-fingered grasps.	26

2.12	Simulation of an exploratory procedure. Dashed lines show the positions of the fingers and objects at the beginning of or during a phase, and solid lines indicate the end of a phase.	28
2.13	Marvin, the testbed robotic hand for haptic exploratory procedures.	30
2.14	Tactile array and force sensor data taken during two phases of exploration. The left finger is shown grasping the object with the palm during phase 2, then exploring the surface during phase 3.	31
2.15	A ball with a ridge feature.	31
3.1	A series of parallel curves for an ellipse with a major axis of length 5 and a minor axis of length 2. The length of the normal line varies between -1 and 1.	40
3.2	2-D slices showing the (a) <i>parallel surface</i> , the locus of endpoints of equal-length lines drawn normal to the original surface, and (b) <i>traced surface</i> , the envelope of spheres whose centers are on the original surface. The radii of the spheres and the length of the normal lines are equal.	41
3.3	2-D slice of a spherical fingertip with original, parallel, and traced surfaces with interference and caustic points identified.	42
3.4	2-D slice of a spherical fingertip with original, traced, and estimated surfaces with the interference point and unobservable region identified.	42
3.5	The path of the spherical fingertip in this picture does not encounter any high curvature regions as it travels over a ridge at a shallow angle. Because of such cases, where a feature exists but is not detected, macro feature definitions cannot contain any assumptions about the path of the fingertip.	45
3.6	A bump feature (bottom), with parallel surface (top).	47
3.7	Curvature features are identified where $ k_i(S) > \frac{1}{r_f}$	48
3.8	Algorithms for feature detection.	49
3.9	Simulated data of a finger tracing over a step.	53
3.10	The experimental apparatus, a two-degree-of-freedom robot finger equipped with a tactile sensor.	54
3.11	Magnified view of unfiltered estimated object surface data.	55

3.12	Feature detection with a robotic finger rolling/sliding on a 45° surface with a 0.65 mm bump feature. The finger size is shown for scale. (See Figure 3.11 for a magnified view of original data points.)	56
4.1	An overview of the phases for local and global haptic exploration. . .	61
4.2	The first phase in exploration is moving the finger over the object during manipulation or while searching for a feature.	61
4.3	The second phase in exploration is moving the finger over and around the feature to determine type and record shape.	62
4.4	As the finger explores more features, the skeletons of the features are used to build a global skeleton that divides the surface into regions. .	63
4.5	The 3GM as a robotic finger, equipped with an Optical Waveguide Tactile Sensor. The finger is exploring a flat surface with a single ridge feature.	66
4.6	Coordinate systems for haptic exploration.	66
4.7	Flow chart for feature type identification. The possible features in this example are a cusp, a step, and a ridge.	72
4.8	The direction of contact point travel during feature tracing is determined by the intersection of a plane parallel to the local surface and a plane orthogonal to the contact normal.	73
4.9	Example of the medial axis for a simple 2D shape.	75
4.10	Example of the medial axis for a feature.	76
4.11	As the number of boundary samples increases, the discrete Voronoi medial axis (DVMA) approaches the continuous Voronoi diagram. The DVMA is then pruned to get a more accurate object skeleton. In this example, the boundary samples are noisy and cause unwanted arcs in the final skeleton. These can be removed using a length threshold.	78
4.12	Examples of branching and non-branching features. If only non-branching features are considered, unwanted skeleton arcs can be easily pruned away.	79
4.13	When a feature turns sharply, extra skeletons are extracted using a simple edge length threshold. The number of edges of disconnected skeletons can be compared, and the shorter ones discarded.	80
4.14	The global skeleton creates regions around features.	83

4.15	The global skeleton is approximated using the Voronoi diagram of the medial axes of the features. It is extracted from the Voronoi diagram by considering only the polygon edges which correspond to medial axis points from two different features.	85
A.1	A typical dexterous manipulation problem: Moving an object from configuration A to configuration B while keeping track of contact point locations.	92
A.2	A coordinate system (f, U) for coordinate patch S_0 of surface S	92
A.3	The Gauss frames for some points on a surface and a view of the Gauss frame coordinate axes.	93
A.4	The motion of contact frames for rolling and sliding contact. The contact frames were at \mathbf{u}_1 and \mathbf{u}_2 for objects 1 and 2 before motion occurred. The new contact frames are \mathbf{u}'_1 and \mathbf{u}'_2	96
A.5	Contact variables. (Adapted from Montana[57].)	96
B.1	Tactile data.	99
B.2	Magnified view of tactile data.	99
B.3	Skew in noise resulting from a linear fit to the tactile data.	100
B.4	Accurate noise resulting from a second order polynomial fit to the tactile data.	101
B.5	Top: A square bump feature, before and after Gaussian filtering. Bottom: Feature detection for the square bump feature.	104
B.6	Feature detection with real data smoothed with a Gaussian filter.	108
C.1	Interior structure of the Optical Waveguide Tactile Sensor. (Adapted from [50].)	110
C.2	Exterior of the Optical Waveguide Tactile Sensor, with and without the silicone rubber cover.	110
C.3	The sensor and detection coordinate systems, adapted from [50].	111
C.4	A side view of the 3GM robotic finger.	114
C.5	3GM kinematic variables.	114

Chapter 1

Introduction

Haptic exploration is an important mechanism by which humans learn about the surface properties of unknown objects. Through the sense of touch, we are able to learn about attributes such as object shape, surface texture, stiffness, and temperature. Although most current robotic systems are designed to operate in well-known, structured environments, robots can explore to learn about unknown environments. Unlike vision or audition, tactile, or haptic, exploration involves direct interaction with the object being explored, which presents significant challenges in both control and sensing. However, recent developments in tactile sensing and dexterous manipulation have made it possible for multi-fingered robotic hands to better manipulate and explore unknown objects.

This thesis develops an approach for haptic exploration of unknown objects with robotic fingers. Exploration and manipulation are inherently coupled, as shown in Figure 1.1. In order to manipulate an object, some information about the object properties and position/orientation must be known. The more that is known about an object, the more manipulation can be planned in advance. In cases where complete object information is not readily available, exploration must be performed to learn about the object's properties, or, given a known shape, the object's pose. In addition, manipulation is necessary for thorough exploration. While portions of a fixtured object may be explored without manipulation, it is often desired to explore all the surfaces of an object. Robotic fingers have limited workspaces, so holding an object stationary during exploration is of limited value. Rather, manipulation is used to move the object into appropriate poses for exploration by multiple fingers.

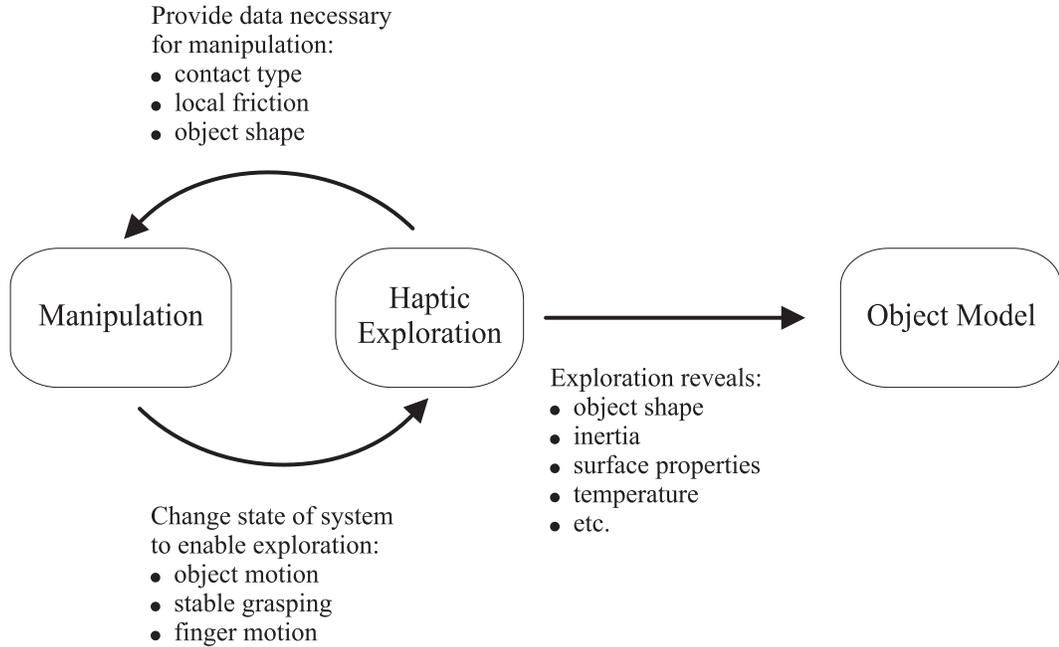


Figure 1.1: Exploration and manipulation are necessarily coupled during the exploration of unknown or partially known objects.

Because haptic exploration is coupled with manipulation, a procedure for combined manipulation and exploration using a sequence of phases is presented in this thesis. Fingers alternately grasp and stabilize the object while other fingers explore the surface with rolling and sliding motions. During an exploratory phase, a hybrid force/velocity control law is used to keep an exploring finger in rolling or sliding contact, moving the fingers tactile sensors over the surface in a way that will elicit useful data.

Humans perform a variety of specific exploratory procedures when attempting to learn about object properties. Psychophysicists Klatzky and Lederman[41] mapped human hand motions to object properties such as texture, hardness, temperature, weight, and shape. Figure 1.2 shows a number of these exploratory procedures. Thus, there exist many possible objectives for haptic exploration. This thesis concentrates on the detection and identification of fine surface features.

In the context of exploration with spherical robotic fingertips, curvature features are defined as areas where the local object surface curvature is greater than that of the fingertip. Patterns of negative and positive curvature features are then used to define macro features, such as bumps, cracks and ridges. Based on different types of sensor data, such as contact location, surface normal direction and fingertip-center

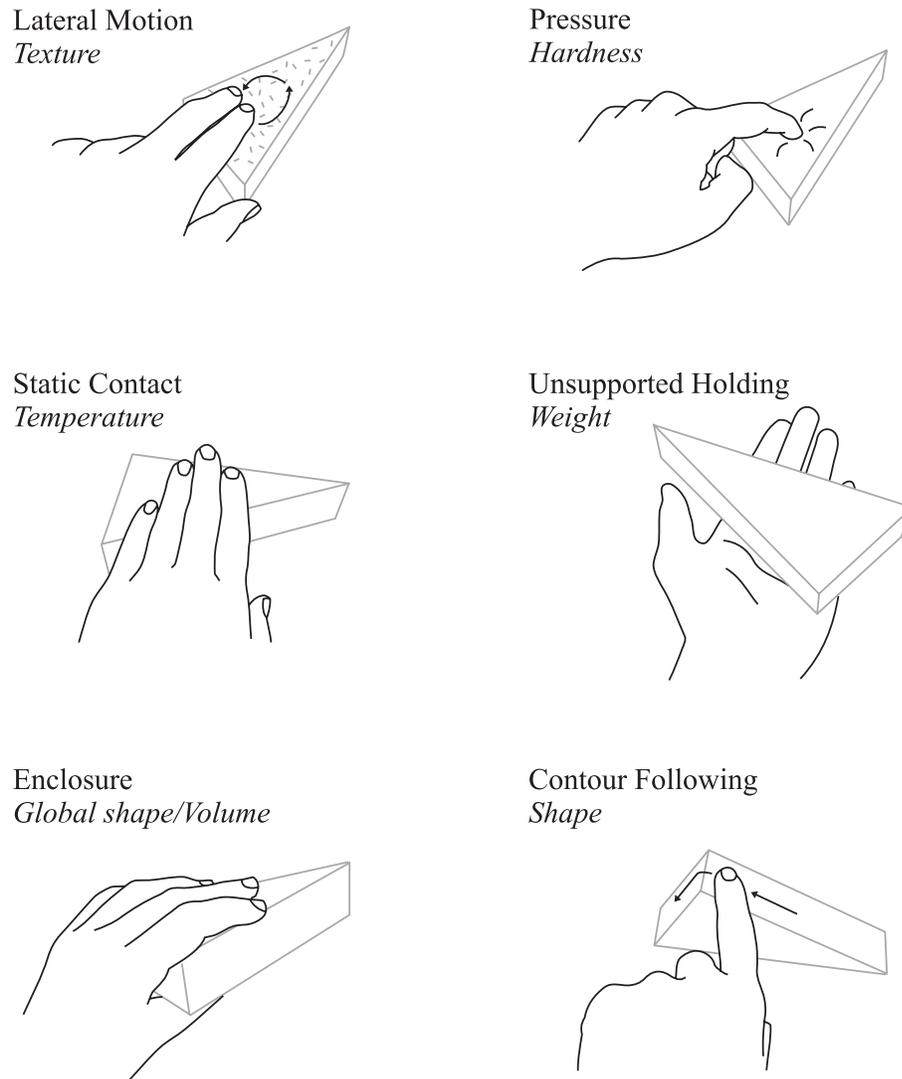


Figure 1.2: Klatzky and Lederman demonstrated that humans use a number of exploratory procedures (EPs) to determine unknown object properties. (Adapted from [41].)

position, various algorithms and experimental results for fine feature detection are presented. The final subject considered in this thesis is active exploration of surface features. With a three-degree-of-freedom robotic finger, there exist many possible methods for exploring a feature. After a feature has been encountered on a surface, tactile sensor and position data may be used to determine the next direction of finger travel, guiding the finger around and over the feature in a way that will efficiently extract surface properties.

Haptic exploration has applications in areas such as planetary exploration, undersea salvage, and other operations in remote or hazardous environments. Autonomous detection and identification of features allow for the development of realistic object models based on tactile data, which can then be displayed with virtual reality systems. These object models will be especially useful in creating realistic haptic displays, where human operators can explore models of actual surfaces obtained by remote robotic exploration.

1.1 Motivation and Challenges

1.1.1 Motivation

The primary motivation for this work is the development of object models from haptic exploration with robotic fingers. One use for such a model is to provide information for reality-based modeling systems. Reality-based modeling uses physical measurements to develop models for analysis or virtual reality displays. Taking these physical measurements is a particular challenge in remote or hazardous environments. In such an environment, a reality-based modeling system requires some autonomous or semi-autonomous device to obtain data. Robot fingers provide this ability, as well as the flexibility and dexterity to handle objects. Related applications include remote planetary exploration, undersea work, and operations in hazardous environments.

In addition, there are some tasks that are simply performed better by autonomous systems than by humans. While a person may have the ability to explore an unknown surface more completely and robustly than a robot, it is difficult to make a map based on human description. A robot can accurately measure points and store contact information automatically. For example, computer-numerically-controlled coordinate measuring machines (CNC CMMs) are used to make accurate measurements of surfaces for the purposes of component inspection or quality control. Such a machine

can accurately detect dimensional flaws with computerized data processing.

Another motivation is the general enhancement of robotic hands, in order to emulate the human capabilities of touch and manipulation. While robots and humans have differing strengths (i.e., robots may be designed to have very good proprioception, while humans have excellent tactile sensing), the diversity of tasks that humans can accomplish with their hands has been an inspiration for robotics. Humans can quickly manipulate and explore an object in order to identify many object properties, even without visual feedback. Through research in control, dexterous manipulation, computer vision, and haptic exploration, robots are being improved to emulate this ability.

1.1.2 Challenges

The general goal for haptic exploration is to have a robotic hand explore an object through the sense of touch in order to determine object properties. It is necessary to create some order from this vague problem statement. The problem can be divided into several sub-problems, each of which presents particular challenges.

- **Manipulation and Exploration.** The theory for robust manipulation and grasping has been extensively researched in previous work. However, active exploration, that uses tactile data and extracts object properties to change exploration and manipulation strategies, has not yet been achieved.
- **Surface Properties.** There are many object properties that can be detected, including geometry, mass, stiffness, reflectivity, acoustic characteristics, and texture. In order to program a robot to explore, there must be a goal to find specific properties. Each property must be defined in order to provide a robot with instructions for data collection. The data collection technique will depend on the type of sensing available.
- **Algorithms for exploration.** Exploration can be divided into two levels, local and global. Local exploratory procedures are used to determine particular object properties, such as texture or stiffness. Global exploratory procedures are methods for exploring the entire surface. If the goal is to explore the whole object, the algorithm must make sure that no area will be missed. If the goal is to find a particular feature, the algorithm must be able to find that feature and obtain sufficient information to identify it.

- **Data Storage and Modeling.** One of the great advantages of robotic exploration over human exploration is that data may be automatically stored for analysis or display. Efficient ways of storing the information about object properties must be developed.

This thesis deals with a subset of these challenges – essentially the minimum subset necessary to explore objects for surface feature detection. In the area of manipulation, I developed a simple procedure that alternates manipulation and exploration. In the area of surface properties, I defined the curvature feature, a geometric object property. Algorithms for detecting and identifying these features are developed. Finally, for data storage and modeling, I have examined the use of shape skeletons for storing information about detected features and their locations on a surface.

1.2 Related Work

Haptic exploration builds on many fields, including tactile sensing, dexterous manipulation, and robot control. This section describes work relevant to the general area of haptic exploration. Work related to the specific topics of each chapter is discussed in that chapter.

Much of the inspiration for haptic exploration comes from the ability of humans to explore the world through touch. As mentioned, psychophysics researchers Klatzky and Lederman[41] have examined the human haptic exploration process with an eye towards developing algorithms for robots. They found that humans use a number of specific exploratory procedures (EPs) for determining object properties such as temperature, texture, and shape.

For robotic exploration, a system for integrating vision and touch for object recognition tasks has been developed by Allen[1, 3, 4]. In this work, he combined passive stereo vision with active exploratory tactile sensing for the discovery of the 3D structure of objects. Vision processing provides sparse 3D data, while a robotic finger further explores regions of interest, particularly those that are occluded from the visual sensor. A hierarchical procedure is used to create surface and feature primitives, which can be compared with model objects in a matching algorithm. Allen has used several types of shape descriptions such as Coons' patches and NURBS (Non-Uniform Rational B-Splines). Allen has also investigated mappings between exploratory procedures and shape representations[2].

Stansfield[85] also used a combination of vision and touch to create a robotic perceptual system. A single finger is used to explore an unknown object and detect the features of an object and the relations between them. Vision is used to segment the object and obtain position information. The features defined by Stansfield differ from the features in this thesis in that they are defined primarily by the EPs used to explore them. Intermediate-level features are surface shape, edges, corners, and semi-parts, and high-level features are extent, contour, surface patch, and feature part. Thus, Stansfield's feature definitions are much broader and are not described by specific definitions based on surface and fingertip geometry. The major issues addressed in Stansfield's work are the structure of a perceptual system and the definitions of primitives, features, and representations extracted and created by the system. The system was implemented using a tactile array, a force/torque sensor, a PUMA robot arm, and a pair of CCD cameras.

Hemami and collaborators[31] have developed a conceptual framework for tactually guided exploration and shape perception. Their framework identifies the necessary sensory information, spatial and temporal transformations of this information, and control mechanisms. They have also investigated hybrid control and learning controllers for moving a robotic finger over globally unknown objects[70, 6, 26]. The low-level hybrid control law used in this thesis for moving a robotic finger over a surface is similar to that in Hemami's work.

Another area related to haptic exploration is surface metrology. A characterization of defect shapes on surfaces, including dimensional characteristics and types, is provided by Whitehouse[92]. These characterizations are not precise, but do give a general categorization of the different types of defects and their appearances.

An interesting note on the previous work in this field is that exploration with robotic fingers has received little attention in the past decade. Most of the work mentioned above was performed in the late 1980s when tactile sensing and grasping/manipulation theory was not as well developed as it is at the time of this writing (2000). In 1988, Hemami states: "The available tactile sensors to date... are not adequate for fast and efficient execution of rolling and gliding manipulations." Combined manipulation and exploration was an especially formidable task with the lack of technology in this area. The work presented in this thesis shows that haptic exploration is feasible, given improvements in tactile sensing, robot finger hardware, and algorithms for manipulation and exploration.

1.3 Contributions

The major contributions of this thesis are:

- A procedure for combined manipulation and exploration of an unknown object.
- A definition of features in the context of haptic exploration with robotic fingers.
- Algorithms for performing feature-guided exploration for the detection and identification of features with robotic fingers in three dimensions.
- Algorithms for using shape skeletons for storing feature information and global surface mapping.

1.4 Thesis Overview

This thesis is organized into five chapters. Chapter 1 is this introduction; it presents the motivations and background for this work, as well as a list of the major contributions.

Chapter 2 presents a procedure for combined manipulation and exploration with a multi-fingered robotic hand. In this chapter, the rolling and sliding kinematics necessary for planning exploration are developed. A sequence of phases is used to alternately manipulate and explore. Three distinct phases are used during manipulation, with transitions between phases triggered by either force closure or workspace constraints. Simulated and experimental results using this framework are shown for a planar robotic hand consisting of two fingers and a passive “palm.”

Chapter 3 discusses the details of using haptic exploration to identify surface features. Definitions of features and macro features are given for the context of exploration with robotic fingers. Various algorithms are presented for feature detection, using tactile and position sensors. Experimental results are presented for the use of a single finger to find features.

Chapter 4 explores the idea of feature-based exploration, whereby the initial detection of a feature leads a robotic finger to change its path in order to more efficiently extract feature properties. A control system for implementing exploratory procedures with a three-degree-of-freedom finger is presented. In addition, shape skeletons are presented as a method for storing feature information and providing a global map of features, to be used in exploration planning.

Finally, Chapter 5 summarizes the results of the research and offers suggestions for future work.

This thesis also contains three appendices: the first introduces a differential geometry description of surfaces and contact kinematics, the second explains smoothing algorithms used for interpreting tactile sensor data for feature detection, and the third describes the testbed (robotic finger and tactile sensor) used for exploration in 3D.

Chapter 2

A Procedure for Multifingered Exploration

A distinguishing characteristic of haptic exploration is that it is coupled with manipulation. Haptic sensing provides us with information, such as object weight and surface friction, needed for stable manipulation, and manipulation lets us explore an entire surface with our fingertips. In addition, control of the sensors' contact force, position, and orientation are required. Thus, precise manipulation control is a prerequisite for tactile exploration. Studies with human subjects have also underscored the coupling between manipulation and sensing; a combination of efferent (active sensing) and afferent (passive sensing) activity helps us integrate the information we obtain[41].

In this Chapter, I present an approach for haptic exploration of unknown objects with a multifingered robotic hand. The emphasis is on developing a robust manipulation process that allows a finger to traverse the surface of an object. The process consists of a sequence of phases in which some fingers are responsible for grasping and manipulating an object while others roll and slide over the object's surface. This procedure allows the rolling and sliding fingers to utilize sensors for detecting and identifying small surface features such as grooves and ridges. This approach builds on recent developments in several areas, including event-driven control of robotic hands, motion planning with rolling and sliding, and sensor integration. Simulations and experiments with a two-fingered hand with a planar palm were conducted to investigate the robustness of the approach for exploring various object shapes.

The basic approach is as follows: exploration proceeds as a sequence of phases in

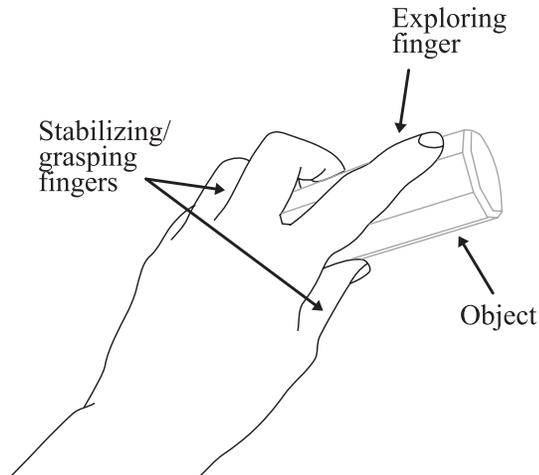


Figure 2.1: Combined manipulation and exploration of an object using multiple fingers.

which a subset of the fingertips is used to stabilize and reorient the object while the remaining fingertips roll or slide over the surface in search of both surface information and suitable contact locations for manipulating the object in the next phase. The fingertips can employ a combination of static and dynamic sensors to determine surface properties and locate features. The reader can confirm that humans take essentially the same approach when manipulating and exploring a small object (Figure 2.1):

... part of the hand typically stabilizes and part explores. For example, the fingers may hold the object while it is surveyed by the thumb.[41]

In the following sections, I first review the related literature, including the main technologies on which this work builds. Then, the exploration algorithm is presented. In this thesis, the conceptual approach described above is specialized to the simplest case: manipulation with two fingers and a palm, as shown in Figure 2.2. In this case, a modest set of states and transitions results. However, this minimal configuration is sufficient for exploring problems associated with ensuring robust and smooth exploration of arbitrary objects. These issues are discussed in the context of simulations and experiments for round and polygonal objects manipulated with a two-fingered hand.

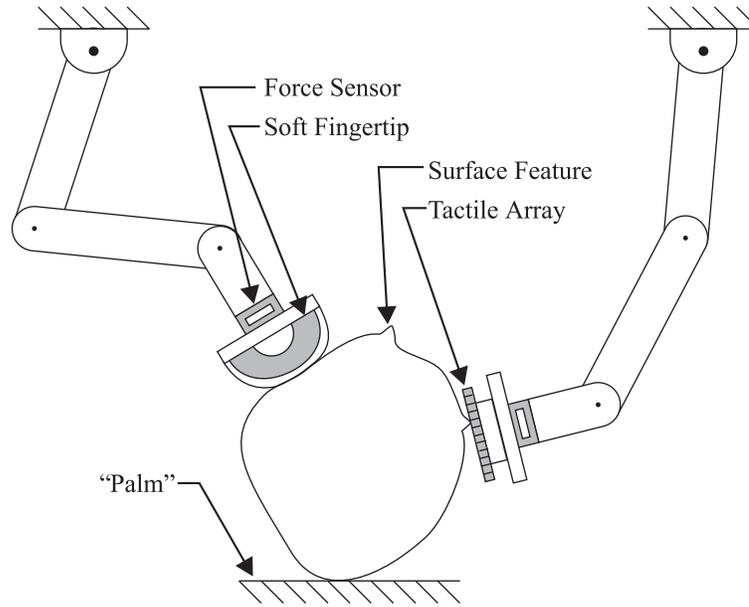


Figure 2.2: Two robotic fingers and an object with haptic features.

2.1 Related Work

The relevant literature includes investigations of tactile sensing and dexterous manipulation for robots and telemanipulators.

Tactile sensing is essential for robust dexterous manipulation, especially when the shape of the object to be manipulated is unknown. Theoretical work on grasping and manipulation with robotic fingers shows that contact location affects grasp stability and the mapping between a finger and an object[19, 40, 54]. In addition, a significant research effort has been directed towards the analysis of how contact locations move during manipulation[18, 57].

Surveys of tactile sensing can be found in [29, 34, 62]. Tactile sensing technology is divided into two main categories: intrinsic sensors (i.e., force-torque sensors) that infer contact location based on global measurements and extrinsic sensors (i.e., array sensors) that obtain contact location from a localized sensor at the contact point[82]. Nicolson and Fearing[63] consider the accuracy limits that can be obtained with array sensors. Son, *et al.*[82] compare methods for obtaining contact point location with tactile arrays and intrinsic force/torque sensors. Zhang, *et al.*[94] compare methods for obtaining object curvature from tactile sensors during manipulation.

Studies of human haptic perception reveal that edge or contour following is one of a set of common “exploratory procedures” that people use for determining object

geometry[41]. A number of investigators, including [5, 8, 15, 60, 70, 72], have developed robotic edge tracking algorithms for use with tactile sensors. Dario, *et al.*[20] describe algorithms for palpation. Others, including Allen[1] and Caselli, *et al.*[13], have developed exploratory strategies for determining object geometry with haptic sensing. However, most of this work focuses on using a minimal set of contacts to determine overall geometry rather than on moving the tactile sensors over the surface in order to discern fine surface features or properties such as texture or coefficient of friction.

A large body of work exists regarding dexterous manipulation. The relevant topics include nonholonomic motion planning, grasp stability analysis and optimization, finger gaiting, and coordinated control of external and internal grasp forces. A few notable examples are [32, 58, 74] and a survey by Shimoga[77]. Most implementations are based upon the kinematic models of Montana[57] or Cai and Roth[12].

Maekawa, *et al.*[49] showed that dexterous manipulation with rolling can be executed using only instantaneous kinematics if a tactile sensor provides continuous updates of the contact location. This is an advantage when manipulating and exploring unknown objects. However, for motion planning, the object curvature is needed to predict how far the fingertips can travel before they will encounter joint-space or grasp stability limitations, which would necessitate regrasping. Thus, there is a tradeoff between the amount of knowledge required about the object and the efficiency with which exploratory motions can be performed. I will return to this issue when discussing the algorithm and results in the following sections.

In other work related to dexterous manipulation with tactile sensors, Son, *et al.*[83] present results of using a tactile array sensor and intrinsic sensing to improve the accuracy of peg-in-hole assembly using a two-fingered hand with rolling contact. Fearing[25] and Sarkar, *et al.*[74] also demonstrate manipulation with tactile sensing.

2.2 Exploratory Procedure Algorithm

2.2.1 Algorithm and States

As described, object exploration proceeds as a repeated series of phases in which some fingers manipulate the object while others explore the object surface. The goal of the manipulation algorithm is to ensure complete and smooth exploration of a wide range of object shapes.

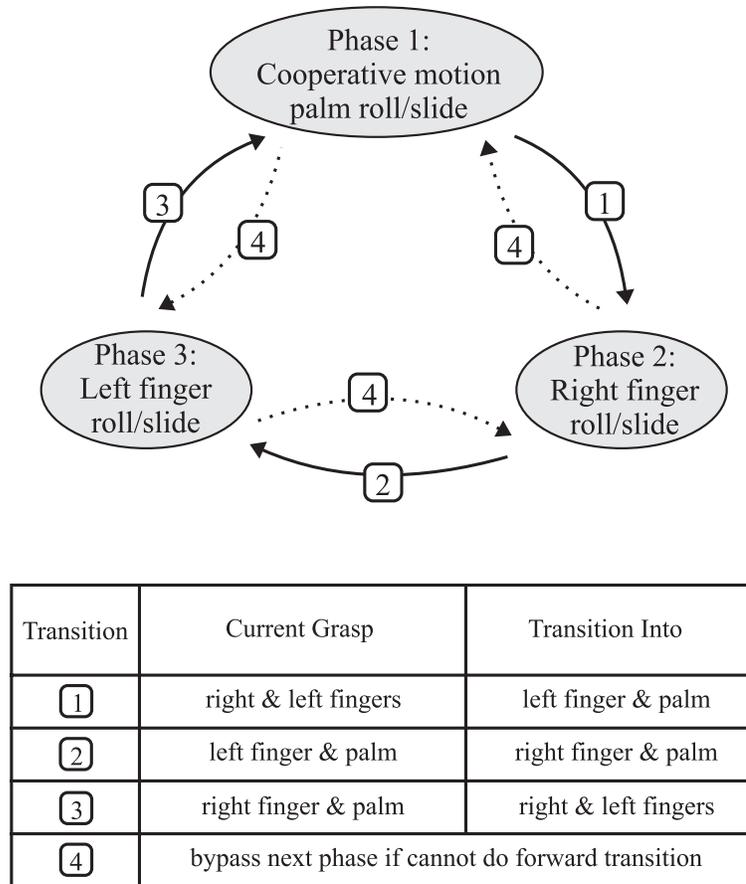


Figure 2.3: States and transitions of the exploratory procedure for two active fingers and a passive palm.

For this exploratory procedure, let us consider manipulation with two three-degree-of-freedom fingers and a fixed “palm,” as shown in Figure 2.2. This is the minimal configuration for our exploratory procedure: there must be at least three possible contact points so that, during manipulation, two contacts can be used for grasping and a third used for exploring. Two of these contacts must be able to impart arbitrary forces (subject to kinematic and friction limitations) to the object in order to move it to a desired location. Thus, the third contact may be passive rather than active. With this minimal configuration, three distinct states can be defined. Using these, a state transition diagram can be represented as a three phase cycle (Figure 2.3).

To simplify the discussion we will assume that we start the cycle in the cooperative motion phase (Phase 1) and rotate the object. In practice, the cycle can begin with any phase that admits a stable grasp. To further facilitate discussion, let us consider

a planar manipulation, and assume that it is desired to rotate the object clockwise. In general, manipulation could proceed by rotating the object in any direction (yaw, pitch, or roll). Considering a planar hand, we can call the two active fingers “left” and “right,” as labeled in Figure 2.3.

In the first phase, the object is grasped by the two fingers and rotated clockwise while optionally maintaining contact with the palm. (This may be useful if the palm is equipped with sensors.) The objectives are to reorient the object and to bring the left finger to a suitable location for holding the object against the palm. Grasp stability is computed using the method of Yoshikawa and Nagai[93]; this is discussed further in the following sections.

In the second phase, the left finger holds the object against the palm while the right finger rolls and/or slides over the surface to a location that will allow it to hold the object in the next phase. The third phase is similar to the second: the object is held between the palm and the right finger while the left finger rolls and/or slides to a location suitable for the next phase: stable cooperative two-fingered manipulation (Phase 1).

Motion planning for this exploratory procedure is based primarily on workspace and grasp stability limitations. The usual ending condition of each phase occurs when a finger reaches a workspace limitation. However, a phase will also terminate if the grasp is starting to become unstable. Depending on the size and shape of the object, a phase may reach a workspace limitation before reaching a stable grasp for the next phase. In this case, it becomes necessary to modify the sequence and bypass the nominal next phase in an effort to reorient the part and obtain better contact locations. This is shown as transition (4) in Figure 2.3.

For planar manipulation, a planar slice of the object can be completely explored using the series of phases described above. Once the object has been rotated 360° and the fingers return to their original contact locations on the object, the entire surface slice must have been traversed by the fingertips if contact was maintained. While complete coverage is straightforward in the planar case, this process of alternating manipulation and exploration is appropriate for general 3D manipulation as well. In this case, complete exploration of the object will require rotating the object about multiple axes. One simple procedure is to perform a series of full planar explorations, while rotating the object slightly between each one. (This results in an exploration path similar to a high-frequency Lissajous curve.) There are other strategies that

may be used to perform global 3D exploration, which are addressed in Chapter 4.

A Minimal Configuration for Exploration

In this section, we will show that the selected configuration is minimal for performing haptic exploration.

Let us first consider the necessary requirements for haptic exploration. In general, limited finger workspaces will require that the object be manipulated in order to do complete surface exploration. The minimal number of contacts for non-dynamic manipulation or grasping is two. For force closure under planar motion, these fingers will need to provide at least two point contacts with friction. (More are needed for general 3D motion.) While these two contacts are used to grasp the object, a third contact can be used for exploration. In order for the exploring finger to be independently controlled to perform any desired local exploratory procedure, it is necessary to have a total of three fingers. More fingers are not required because alternating pairs can be used to grasp and manipulate.

We can further minimize the system by making one of the three fingers passive. At least two fingers must be active to perform stable, non-sliding manipulation. The third finger, however, can serve as a passive surface against which the object can be pushed for a two-contact grasp. The active fingers must have at least two-degrees-of-freedom in order to move the fingertips to arbitrary positions for planar motion. Thus, the true minimal configuration is two two-degree-of-freedom fingers and a passive finger (or “palm”).

Given this three-contact configuration, there are three states (or phases) which correspond to the object being grasped/manipulated with a pair of contacts (since three different pairs can be created). During the phases in which one of the active fingers is the free finger, exploration can be performed. This corresponds to two of the three states. With these three phases, there are six possible transitions. As shown in Figure 2.3, three of these transitions are equivalent to bypassing the next phase while continuing an ordered series of phase transitions.

This minimal configuration of two two-degree-of-freedom fingers and a “palm” is similar to the system considered in the experiments and simulations in this chapter, with two differences. In the experiment, the active fingers have three degrees of freedom, which increases the workspace and allows for independent control of contact location. This allows for the planning of contact point motions for exploration. In

addition, it is also useful to have soft contacts rather than point contacts for improved friction during manipulation and for housing the tactile sensors used during exploration, as shown in Figure 2.2.

Grasp Gaiting

One can consider the series of phases and transitions in this procedure as a grasp gait. A grasp gait is a series of separate grasps used to reorient an object. In order to develop a grasp gait for manipulating an object, the *grasp map* is used to observe the set of stable grasps for a given object. Leveroni[45] constructs the grasp map by solving numerically for contact locations that represent force-closure grasps. (Most of the discussion in this section is based on Leveroni's work.) The grasp map for a two-fingered grasp is represented by points on a plot where the axes are two variables that specify the angles of the contacts with respect to a frame attached to the object. This is a similar concept to the contact configuration space suggested by Chen and Burdick[16].

Grasp maps are useful for determining whether an object can be fully explored using the exploratory procedure developed in this chapter. By overlaying the grasp map with a workspace map, which shows the possible contact locations on the object based on finger kinematics, one can find feasible grasps as regions that satisfy both the grasp stability and workspace constraints. Figure 2.4 shows a sample grasp map overlaid with a workspace map for the configuration of two active fingers and a passive palm.

Leveroni observed that a rotation of the object results in a sliding of the grasp map along a diagonal in the workspace map. With this knowledge, the problem of finding a gait can be posed in map space. In order to find a grasp gait, it is first necessary to find points which satisfy both the grasp stability and workspace constraints. Next, because only one finger may be released at a given time, these points must be connected by moves parallel to the grasp map axes. In addition, consecutive points should be in different regions of the workspace map. Otherwise, the grasp would not switch to a different pair of fingers.

Several different methods have been considered for finding feasible grasp gaits. Brute force search, breadth first search, depth first search, and best first search (using an evaluation function to select the next node) were all tested by Leveroni, but most of these methods were too computationally expensive and inefficient. Instead, she

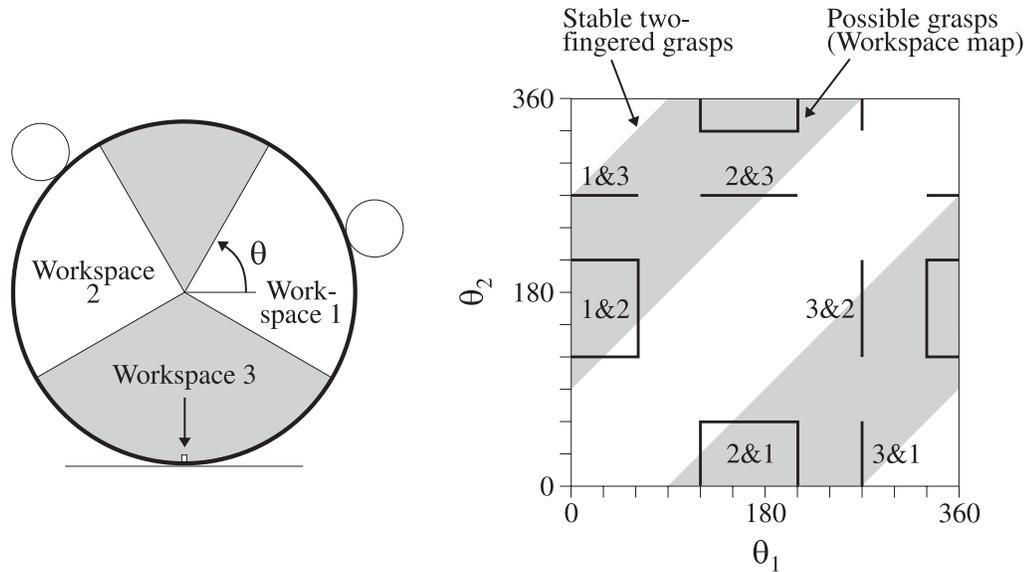


Figure 2.4: A circular object and the corresponding grasp map for two-fingered grasps. The workspace map regions corresponding to different pairs of fingers are overlaid on the grasp map. The configuration is two active fingers and a passive palm, and the coefficient of friction between the object and the fingers is 1.

observed that there are some prototype gaits that can be slightly modified to fit different objects. One of the basic prototype gaits defined, the “forward gate,” is similar to the series of phases and transitions in this chapter. While Leveroni does not consider that the fingers may be exploring, rather than simply detaching from the object while the other fingers are grasping or manipulating, the concept is the same.

Using grasp gait analysis, one can determine what object shapes and sizes can be successfully explored given a particular robotic hand configuration. In general, grasp gait planning will not be feasible because the object shape is not known in advance; a grasp map cannot be constructed without a priori knowledge of shape. However, the grasp and workspace maps could be used to plan detailed explorations based on a rough initial exploration that tests the ability of a hand system to perform complete explorations of different object shapes.

2.2.2 Phase 1 - Cooperative Manipulation

During phase 1, the cooperative manipulation phase, the motion planning and control variables are the object position and orientation and the distance that each contact

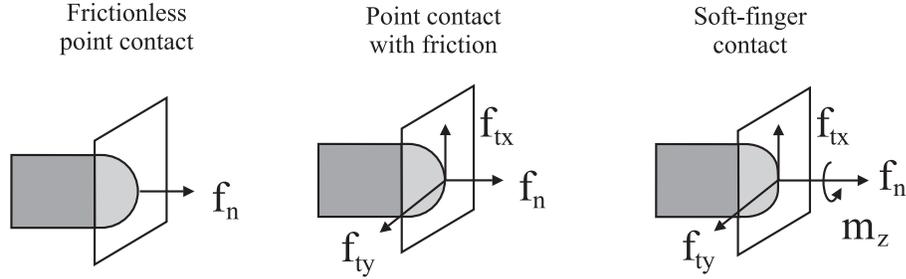


Figure 2.5: Common contact types and their associated forces/moments.

moves over the object surface. The object is rotated as far as possible, subject to finger workspace and grasp stability limits. Trajectory planning is done assuming pure rolling with point contact. This is an idealization since the fingertips are soft and have a distributed contact patch. However, as shown by Chang and Cutkosky [14], the deviations in rolling distances are negligible if contact forces are light.

Motion Planning

The general kinematics of contact is developed in Appendix A. The kinematics yields a set of equations that relate the relative motion of two bodies in point contact to the motion of the contact points on the bodies. These equations can be simplified for particular cases. We will consider planar contact with friction; this situation matches the experimental apparatus used to develop a procedure for exploration and manipulation with rolling and sliding.

First, let us examine the different types of contact and the friction constraints. There are three main contact types to consider, as shown in Figure 2.5: frictionless point contact, point contact with friction, and soft-finger contact.

In order for prevent sliding between two bodies, two friction constraints must be met:

$$f_{tx}^2 + f_{ty}^2 \leq c_1 f_n^2 \quad (2.1)$$

$$g(f_{tx}, f_{ty}, m_z) \leq c_2 f_n, \quad (2.2)$$

where f_{tx} and f_{ty} are the applied force components in orthogonal directions tangential to the surfaces at the point of contact, f_n is the applied normal force between the two objects, m_z is the applied moment about an axis normal the to the surfaces, and c_1 and c_2 are constants related to the coefficient of friction for the two materials considered.

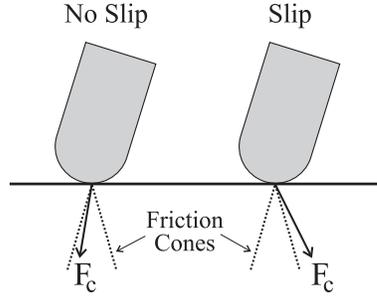


Figure 2.6: The friction cone is a geometric description of the point contact friction constraint.

Maintain contact	$v_z = 0$
No sliding	$v_x = 0, v_y = 0$
No spin	$\omega_z = 0$
Planar rolling	$\omega_x = 0$

Table 2.1: Contact velocity constraints for pure rolling in the $x - z$ plane.

Equation 2.1 describes the point contact friction constraint. Figure 2.6 illustrates the friction cone resulting from point contacts. Equation 2.2 is an analogous relation for soft finger contacts where the component m_z exists. The function g relating f_{tx} , f_{ty} , and m_z to f_n (a friction limit surface) was developed by Kao and Cutkosky[38]. Sliding and friction constraints are also addressed by Howe and Cutkosky[36] and Cole, *et al.*[18].

For pure rolling in a plane, the constraints on the relative motions between two objects are shown in Table 2.1. The plane considered is the x - z plane, and the contact is assumed at a point where the relative translational velocities are v_x , v_y , and v_z , and the relative angular velocities are ω_x , ω_y , and ω_z .

We will consider a case that matches our experimental setup: soft finger contact with friction. The contact force is high enough that the three-dimensional friction constraint (Equation 2.2) is satisfied. In this case, the spin $\dot{\psi}$ must be zero. Thus, the contact equations developed in Appendix A, Equation A.12 reduce to:

$$\begin{aligned}
 \dot{\mathbf{u}}_1 &= M_1^{-1}(K_1 + \tilde{K}_2)^{-1} \begin{bmatrix} -\omega_y \\ 0 \end{bmatrix} \\
 \dot{\mathbf{u}}_2 &= M_2^{-1}R_\psi(K_1 + \tilde{K}_2)^{-1} \begin{bmatrix} -\omega_y \\ 0 \end{bmatrix}
 \end{aligned} \tag{2.3}$$

$$\dot{\psi} = 0$$

Because they are defined by the surface geometry and not contact motion, the curvature matrix K , torsion form T , and metric M must remain constant, as long as the soft fingertip does not change its deflection. In addition, R_ψ and \tilde{K}_2 must be constant because $\dot{\psi} = 0$. Thus, the equations for $\dot{\mathbf{u}}_1$ and $\dot{\mathbf{u}}_2$ reduce to a constant (C_1 or C_2) times the angular velocity. These equations are now holonomic.

$$\begin{aligned} \dot{\mathbf{u}}_1 &= C_1 \begin{bmatrix} -\omega_y \\ 0 \end{bmatrix} \\ \dot{\mathbf{u}}_2 &= C_2 \begin{bmatrix} -\omega_y \\ 0 \end{bmatrix} \\ \dot{\psi} &= 0 \end{aligned} \tag{2.4}$$

Integrating these equations, one obtains the well-known definition of arclength: $s = r\theta$. In the procedure for haptic exploration, the manipulation phase is performed with pure, planar rolling of the fingertips on the object. With knowledge of the local curvature of the fingertips and the object, rolling motions can be planned in advance of actual exploration.

Control

During phase 1 of the exploratory procedure, control of object orientation and internal force between the fingers is performed using dynamic object impedance control[33, 75]. A block diagram of the total control scheme, including rolling control, is shown in Figure 2.7. The commands are the desired body position and orientation, contact locations on the object (or fingers), and the internal force. These commands control six variables that completely define the grasp. The robot hand used for experimental validation of the exploratory procedure has six-degrees-of-freedom. (There are two fingers with three-degrees-of-freedom each.) Thus, the number of freedoms in the fingers is equal to the number of grasp variables we wish to control. We will consider each of the six input commands as a separate control problem.

The first control problem is the object impedance controller, used for object position and orientation. Transformations of motion and force describing this control problem are presented by Mason and Salisbury[54] and Murray, Li and Sastry[59].

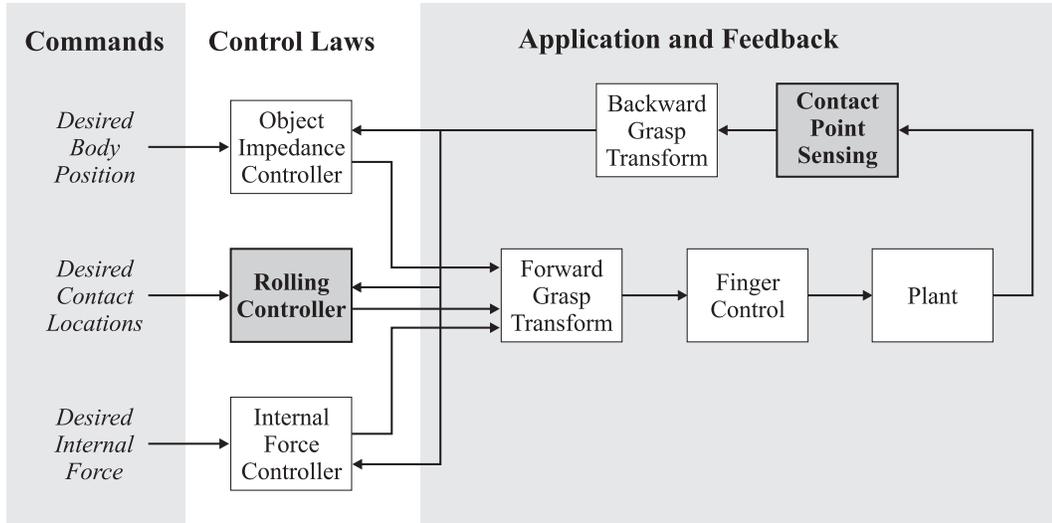


Figure 2.7: Control block diagram including rolling feedback control.

One can transform the velocity of the joint angles of the robot (measured with encoders) to the velocity of the fingertips with the Hand Jacobian, J_h .

$$\dot{\mathbf{x}}_c = J_h \dot{\boldsymbol{\theta}}, \quad (2.5)$$

where $\dot{\mathbf{x}}_c$ is a vector of operational space (Cartesian) fingertip velocities at the contact point with the object and $\dot{\boldsymbol{\theta}}$ is a vector of joint velocities.

Similarly, the Grasp Jacobian, G is used to calculate the motion of the object from the motion of the fingers.

$$\dot{\mathbf{x}}_{obj} = G^T \dot{\mathbf{x}}_c, \quad (2.6)$$

where $\dot{\mathbf{x}}_{obj}$ is a vector of operational space object velocities and $\dot{\mathbf{x}}_c$ is a vector of operational space fingertip velocities at the contact points. These Jacobians can be used in force/torque space as well as velocity space.

The fingertip forces that move the object are considered to be the *external forces*. These forces are calculated based on a proportional-derivative (PD) position and velocity servo (Equation 2.7). With this impedance control, the external force applied by the fingertips on the object is proportional to the position error of the object. The masses of the fingers are also fed forward to provide inertial and gravity compensation.

$$F = G + M\ddot{x} + K_p(x_d - x) + K_v(\dot{x}_d - \dot{x}) \quad (2.7)$$

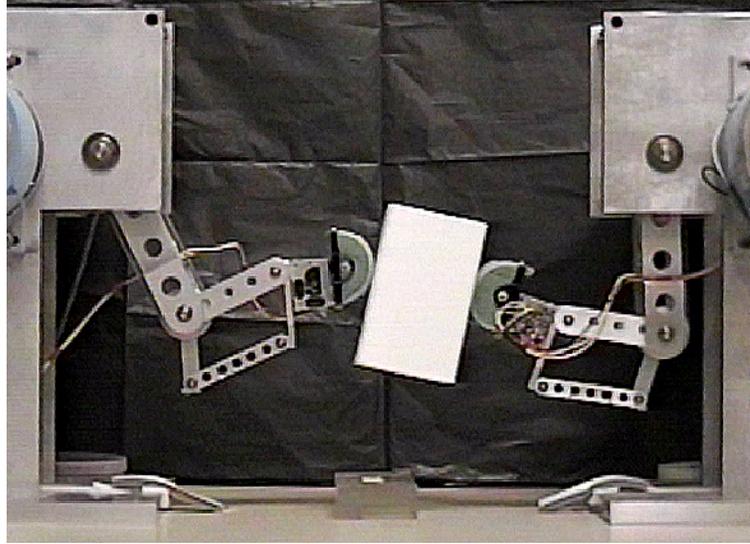


Figure 2.8: Explicit rolling can be commanded when there are sufficient degrees-of-freedom in the robot hand, such as this system of two fingers with three degrees-of-freedom each.

The second control problem is concerned with moving the contact locations, changing where the fingers touch the object. The rolling control law in the work was designed to augment the traditional object impedance control law. “Contact point sensing,” shown in the block diagram in Figure 2.7, can be done using intrinsic or extrinsic tactile sensors, or using assumptions based on fingertip motion history if the local object shape is known.

The type of rolling control law that can be used is highly dependent on the number of degrees-of-freedom of the system and the application. In this thesis, we define *explicit rolling* as the case where one can (and desires to) specify the location of the contact points on the object or the fingertips independent of the object position and orientation. This can only be accomplished when there are sufficient degrees-of-freedom in the robotic hand, such as in Figure 2.8.

The explicit rolling control torques are added to the general external force control torques. The rolling control torques are calculated using a proportional-derivative control on the contact position and velocity error. The control law is

$$\tau = K_{p_{roll}}(xc_{des} - xc) + K_{v_{roll}}(\dot{xc}_{des} - \dot{xc}), \quad (2.8)$$

where $K_{p_{roll}}$ and $K_{v_{roll}}$ are the feedback gains, xc_{des} is the desired contact location, and xc is the actual contact location.

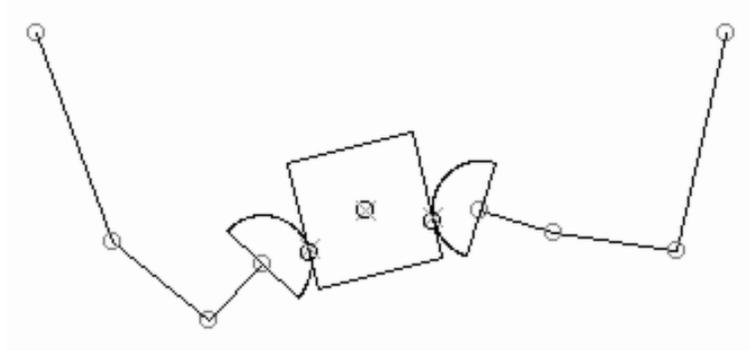


Figure 2.9: Model of explicit rolling during a simulation performed using Matlab.

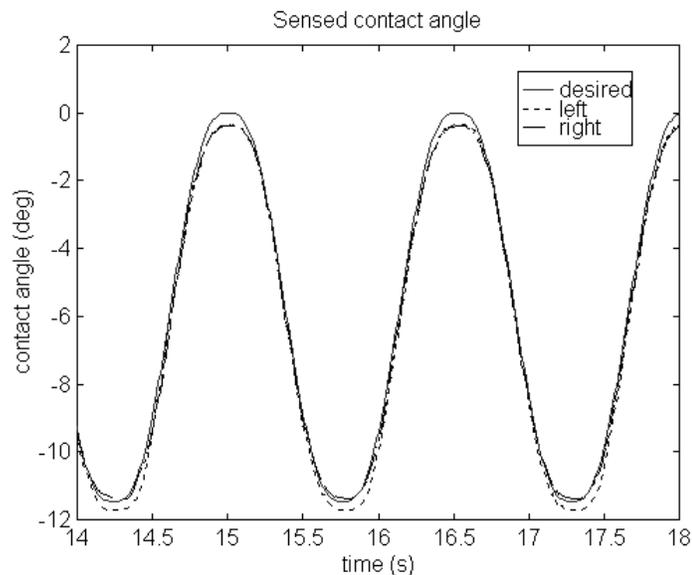


Figure 2.10: Explicit rolling control experimental results.

Explicit rolling simulations (in Matlab, Figure 2.9) and experiments (using the testbed in Figure 2.8) were performed to verify that the explicit rolling control law was effective. A plot of commanded versus actual contact locations for the experimental work is shown in Figure 2.10.

Implicit rolling can occur when there are not enough degrees-of-freedom to explicitly define the locations of the contact points. For example, if two two-degree-of-freedom fingers are moving an object, the contact points must move if the object moves, for general motions. However, even if there are sufficient degrees of freedom to command contact point motions, there may be reasons for not doing so. For example, there may be parameters one would like to minimize, such as distance of the robot fingers from the center of the workspace or the distance of contacts from the edge of

a square object. In this case, implicit rolling would occur as the contact locations move due to the control scheme rather than from an explicit calculation.

The last control problem addressed in cooperative motion is the *internal force* on the object. The internal force servo is a straightforward proportional-integral (PI) control on the sensed internal force on the object. The internal force is calculated by taking the minimum of the force sensed by each finger along a line through the two contact points.

2.2.3 Phases 2 and 3 - Exploration

During phases 2 and 3, the motion planning and control variables for the moving finger include the trajectory of the contact and the orientation of the fingertip. The specification of the fingertip orientation determines the amount of relative sliding that takes place. At one extreme, the orientation can be made consistent with pure rolling, and at the other the fingertip orientation can be kept constant, so that the relative motion is pure translation.

Motion Planning

In practice, the duration of each phase is mainly a function of the workspace of the fingers. Therefore, the fingertip orientations (or equivalently, the amount of rolling at each fingertip) are planned using a simple heuristic that attempts to keep the fingers within their workspaces for as long as possible. The Cartesian workspace of each finger is divided into four regions, each of which has a “preferred” fingertip orientation - one that maximizes the local configuration space. As each phase is planned, the approximate final position of the finger is mapped to one of the four workspace regions and the corresponding preferred orientation is found. The orientation is then interpolated between the initial value and this final value.

The planning is done dynamically at the start of each phase, using a current estimate of the object curvature and surface orientation. When the exploration task is just beginning, this estimate may be poor (for example, there may be an unexpected corner), in which case the phase will end quickly as the finger reaches the edge of its workspace. If the fingertip has not moved far enough to grasp the object stably in the next phase, the algorithm reverts to the previous phase (transition (4) in Figure 2.3). If a stable grasp within the finger workspace still cannot be found, the algorithm terminates.

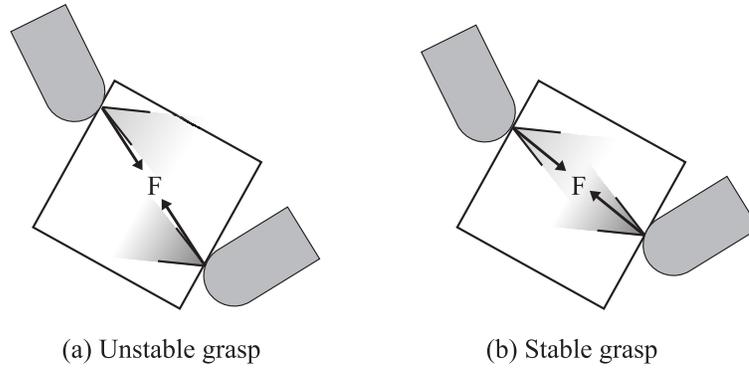


Figure 2.11: Examples of (a) unstable and (b) stable two-fingered grasps.

Grasp stability is computed using the method of Yoshikawa and Nagai[93]. A stable grasp occurs when the grasping forces at the two contacts each lie within a friction cone at the contact. The grasp force from each contact must lie along the same line if there are only two contacts. The friction cone is defined by the limit

$$f_t \leq \mu f_n \quad (2.9)$$

where f_t is the total force tangential to the surfaces, f_n is the total force normal to the surfaces, and μ is the coefficient of friction. Figure 2.11 shows examples of unstable and stable two-fingered grasps.

Control

In phases 2 and 3, there are two control laws in effect: one to grasp the object between two contacts, and the other to move the exploring fingertip. The grasping control of the object is accomplished using the internal force control as described in the previous section. Forces applied by the moving fingertip are considered external disturbances (which are compensated for by the integral portion of the control) and are not explicitly accounted for in the control law.

The moving fingertip is controlled independently using dynamic impedance control in the tip workspace. The impedance control is similar to that used for object control in cooperative motion, but now we specify only the impedance of one fingertip. The variables modified in this control law are the $x-z$ position of the contact point on the object and the orientation of the fingertip, completely defining the control problem for a three degree-of-freedom finger.

Phase Transitions

Transitions from one phase to the next occur in response to events. In the case of the exploratory procedure described in this chapter, the events are fingers reaching the limits of their workspaces and a computed loss of grasp stability.

Some care is needed to ensure smooth transitions that will not excite the tactile and force sensors. For example, in the transition from phase 3 to phase 1 the control changes from fingertip impedance to object impedance. Smooth ramping is provided by an explicit “startup” segment that is part of the phase definition. When this startup is a change from force to position control of a finger, a 5th order position spline is used to command smooth positions, velocities and accelerations. The commanded internal and external forces on the object are initially computed to be consistent with the commanded tip forces at the end of the previous phase and gradually ramped to their desired values for object manipulation. This phase/event/transition system is described in[89].

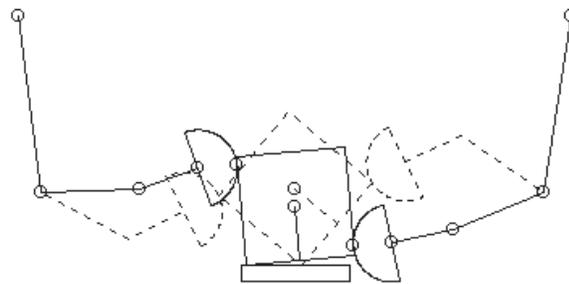
2.3 Simulations and Experiments

2.3.1 Simulations

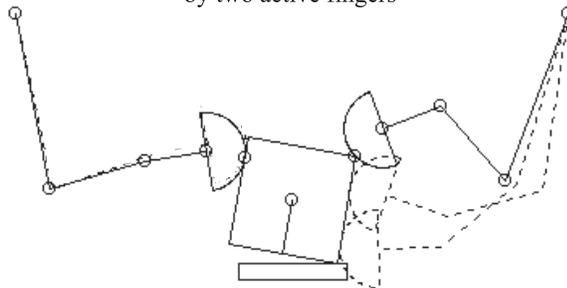
The exploratory procedure was first simulated numerically to determine how well the algorithm would traverse a range of object shapes, including round and square objects, and to test the sensitivity of the approach to workspace limits. The simulation modeled the grasp kinematics and included forces and friction coefficients in testing the grasp stability but did not include inertial terms.

Figure 2.12 shows three phases of the simulation during a clockwise rotation of a square object. The dotted lines show the finger positions at the start of each phase, and the solid lines show the final positions (the final position of one phase becomes the starting position of the next). The coefficient of friction between the object and fingertips and palm was assumed to be 1.0, a typical value for the rubber-coated fingertips used in subsequent experiments.

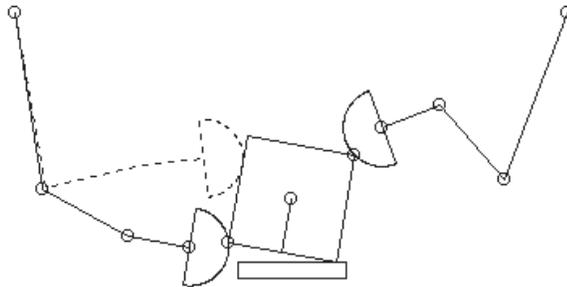
In general, the robot transitions from one stable configuration into the next. In some cases, rotating about a sharp corner on an object would require the simulated right finger to move outside of its workspace in phase 2 before reaching a stable configuration for phase 3. In several of these cases, the robot was able to recover by



Phase 1: Object rotated
by two active fingers



Phase 2: Right finger
rolls/slides on object



Phase 3: Left finger
rolls/slides on object

Figure 2.12: Simulation of an exploratory procedure. Dashed lines show the positions of the fingers and objects at the beginning of or during a phase, and solid lines indicate the end of a phase.

skipping phase 3 (transition 4 in Figure 2.3) and rotating the object with two fingers again.

The simulation revealed that finger workspaces were the most serious limitation and led to the development of the heuristic, mentioned in the previous section, for specifying the fingertip orientation at the end of each phase. Even so, the algorithm cannot handle long, thin objects, relative to the shape of the finger workspace. The workspace and grasp stability limitations of the exploratory procedure can be predicted using the grasp and workspace maps described in Section 2.2.1.

The simulation did not account for the effects of noisy sensors and imperfect control of the fingertip trajectories and forces. These effects were evaluated in experiments described in the next section.

This procedure works for a number of different object shapes, but its success is highly dependent on the workspace of the fingers. By allowing some phases to be skipped when a transition will result in an unstable grasp, it is assured that any object can be manipulated, as long as a pair of contacts can be found within the workspace to grasp the object stably in any orientation.

2.3.2 Experiments

The experimental test bed is a two-fingered planar robot with a passive palm, the minimal configuration for multifingered manipulation and exploration (Figure 2.13). Strain gauge force sensors and an 8x8 tactile array were used to collect haptic information. Contacts can be located to within approximately 1mm using the array and forces can be measured to an accuracy of approximately 0.02N[82]. The robot was controlled using dynamic object impedance control[33, 75] and the phase/event/transition framework of Hyde, *et al.*[89].

Figure 2.14 shows results obtained from the left finger sensors, while exploring a 10cm diameter plastic ball with a ridge approximately 3mm high and 4mm wide on its surface. The ball, shown in Figure 2.15, is manipulated clockwise so the tactile array slides over the ridge in the “snapshots” at the top of Figure 2.14.

During the first 3.8 seconds (Phase 2) the right finger slides over the object while the left finger holds it against the palm. The left finger tangential force is slightly positive, where the direction of positive tangential force corresponds to opposing the direction of motion of the moving finger. The tangential force is non-zero because the finger is grasping the ball against the external disturbance force from the right finger’s

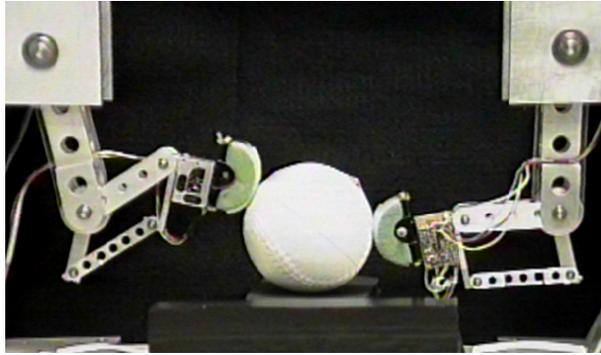


Figure 2.13: Marvin, the testbed robotic hand for haptic exploratory procedures.

motion. The force data reveals little noise during this phase, indicating that the right finger is sliding smoothly without exciting vibrations in the ball. At 3.8 seconds the right finger workspace limits are reached and a transition to phase 3 occurs. At this point, the left finger tangential force becomes negative, indicating that the tangential force has changed directions. As the fingertip slides over the feature the tangential force increases and becomes subject to stick/slip vibrations.

The model of the object used for motion planning is a sphere with approximately the same diameter as the actual ball, but with no features. Using the grasp map described in Section 2.2.1, it can be verified that it is possible to find a grasp gait that can be used to fully explore this object. The normal force is controlled to remain at approximately 0.3N but, as the plot reveals, the tangential force varies as the finger passes over the surface feature.

In Figure 2.14, the first and last tactile images show the pressure distribution produced by contact with the plastic ball. The middle images show the presence of the ridge. The pressure distribution near the feature becomes significantly sharper and changes from the characteristic pattern of a spherical contact to a ridge.

Specific to this experiment, there are several changes that could be made to enhance the manipulation and exploration capabilities of this robotic hand.

- The workspace of the fingers is a main limitation. The fingers employed in these experiments have joint angle ranges of approximately 110° at each joint. A full 180° range of motion would significantly increase the range of sizes and shapes that could be handled. With more fingers, however, the motions required of each finger can be reduced. Another way to increase the useful workspace of the fingers is to change the fingertip geometry, which changes the rolling distance for given joint motions. The combination of one flat fingertip and one

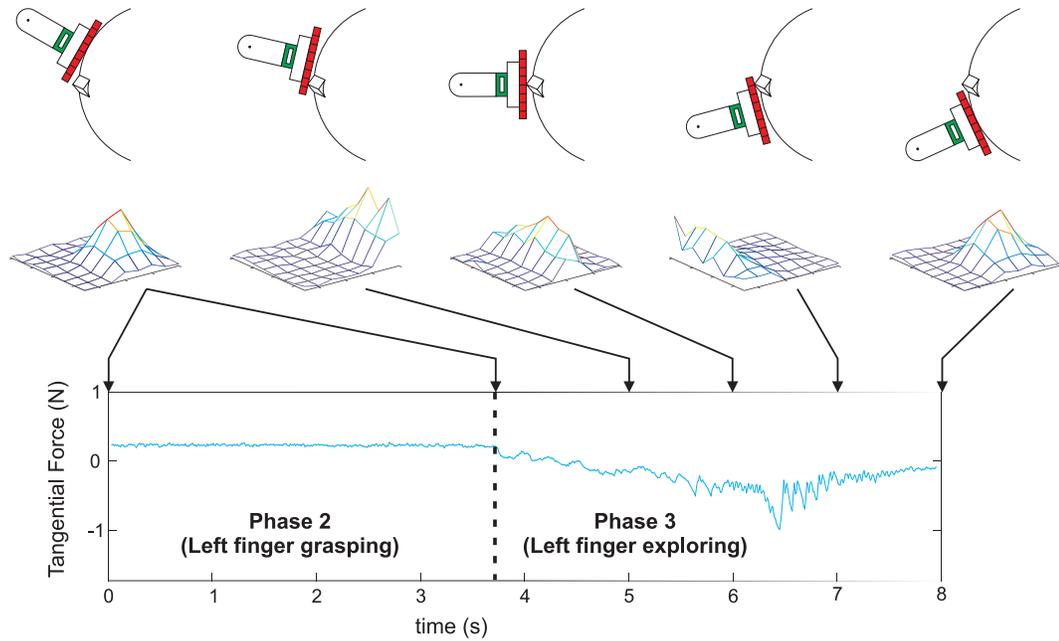


Figure 2.14: Tactile array and force sensor data taken during two phases of exploration. The left finger is shown grasping the object with the palm during phase 2, then exploring the surface during phase 3.

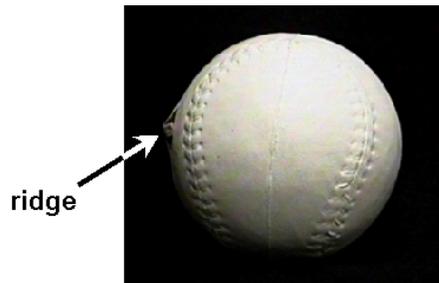


Figure 2.15: A ball with a ridge feature.

2.5cm radius fingertip was fairly restrictive. A tactile array that can fit over a curved fingertip with a small radius (i.e., 1.5cm) would be useful.

- Although the soft, rounded right fingertip, equipped with a texture of rubber “nibs” slid easily, the tactile array produced stick-slip vibrations, especially when passing over features. This is not surprising because the array was designed for rolling rather than sliding. Sliding difficulties have been observed with smooth skins in the past[35]; a new array skin design would permit easier sliding without distorting the sensor readings. It may also be possible to suppress stick-slip vibrations through better control.

2.4 Conclusion

The simulation and experiments confirm our basic conviction that the state of the art in tactile sensing and dexterous manipulation planning control are reaching the point at which autonomous haptic exploration becomes feasible. Thus, an exploratory procedure where some fingers hold the object while others explore is a valid paradigm. The main advantage is that the exploring finger does not need to contribute to grasp stability and the disturbing forces that finger applies to the object can be accommodated as external forces on the object.

Although the results presented are for the simple case of two independently controlled fingers, the basic approach can be applied to additional fingers and three-dimensional exploration. The two-fingered manipulations revealed several issues in haptic exploration, motivating the rest of the work addressed in this thesis:

- While it is apparent that there exists some perturbation on the surface of the object, as shown from the tactile and force data in Figure 2.14, there needs to be a method for extracting features automatically. To accomplish this, precise feature definitions and specific local exploratory procedures are necessary.
- In general, finger motion will not be constrained to planar motion as was the case in this experiment. Thus, the upcoming chapters address the definition of features in 3D and methods for exploring features on 2D surfaces in 3D space.
- The purpose of the exploratory procedure is the recording of object properties, which must be stored and identified to be of use. Data from the various sensors must be integrated into an object model. This information should be updated during manipulation, so that the exploration can be modified to explore interesting features in more detail. The structure of the object model should be simple enough that object features may readily be extracted, but flexible enough to incorporate multiple types of haptic information. The structure will also depend on the projected use of the information, for example, whether for played back through a haptic interface or displayed visually.

In this Chapter, we presented a procedure for haptic exploration coupled with manipulation. Now that it has been verified that combined manipulation/haptic exploration with multiple fingers is a viable method for obtaining surface data, we

will turn our attention to the feature detection and identification abilities of a single finger exploring a surface.

Chapter 3

A Feature Definition for Haptic Exploration

The purpose of any exploratory procedure must be defined in order to determine the nature of the exploration. Without such a goal, there is no particular direction or method to the exploratory procedure. For many applications, including reality-based object modeling, searching for object properties, and object identification, a key goal is the detection of object features. Primarily, features are a useful way to organize the information obtained while exploring an object. A number of investigators including Ellis[24] and Stansfield[85] consider features as intrinsically relevant for robotic haptic sensing. Features can also be used to segment object data that has been obtained, such as in registration procedures for pattern matching[56]. This chapter focuses on a feature definition and detection method that is particularly suited for haptic exploration of objects with small surface features such as bumps, ridges, and grooves.

A key question in this work is the definition of a feature. Brown, *et al.*[11], define features as “application and viewer dependent interpretations of geometry.” As will be shown, the identification of a feature is not only dependent on the geometric properties of the object being explored, but also on the properties of the robotic finger (the “viewer”) performing the exploration.

Some types of features, particularly small ones, cannot be sensed accurately through static touch; motion is required. To excite the fast-acting, vibration sensitive mechanoreceptors embedded in the fingertips, humans roll and/or slide their fingertips over a surface[41]. Emulating this behavior presents a challenge in that manipulation must move the object to a desired location with a stable grasp and

move the fingers and sensors over the features. The exploratory procedure in the previous chapter enabled this combination of manipulation and exploration. Using this procedure as a way to move robotic fingers over a surface, we can now consider the mechanics of detecting features.

This chapter begins with a review of previous work on feature detection and the representation of object surface geometry. Next, surface feature definitions are presented based on surface curvature. These definitions depend on the geometries of both the robot fingertip and the object being explored. It is also shown that the trajectory traced by a round fingertip rolling or sliding over the object surface has some intrinsic properties that facilitate feature detection. Then, several algorithms based on the feature definitions are presented and compared. Finally, simulated and experimental results are presented for feature detection using a hemispherical fingertip equipped with a tactile sensor.

3.1 Previous Work

As described in Chapter 1, a number of investigators have addressed the problem of using robotic fingers in exploratory procedures. Examples include work by Allen[3] and Pribadi[70], which focus on the sensing of global object shapes and on fitting shapes to object models. The integration of tactile sensing and dexterous manipulation with rolling or sliding has also been observed in recent work by Li[46] and Maekawa[49]. However, none of the previous work with robotic fingers has addressed the definition and detection of features for haptic exploration.

Much of the work on identification of features has come from the vision research community. In early work, the definitions of features in applications such as topography were often ambiguous because they were based on natural language. More recently, researchers have developed definitions based on various mathematical models, including local maxima of pixels in a discrete 2D image and height or intensity graphs.

Research using a differential geometry approach includes finding the local extrema of principal curvatures[56] and curvature properties of level sets of smooth functions[22]. In another approach, Kunii, *et al.*[43] extended the idea of the Medial Axis Transform to develop skeletons of object shape from image data and used caustic singularities in these skeletons to determine the locations of ridges. Applications of

ridge detection in images include medical imaging[53] and the analysis of topographic data[44].

While these feature definitions are applicable to understanding image features, they are not appropriate for haptic exploration. Exploration with a robotic finger is inherently different from vision because it is active rather than passive, and the sensing modality is tactile rather than optical. In addition, one in general has access only to the path of the robotic finger, rather than a full image of the object. Because the definition of a feature is context-sensitive, it was necessary to create a new feature definition for the context of exploration of surfaces with robotic fingers.

3.2 Defining Surface Features

This thesis will take a differential geometry-based approach to surface feature definition. This section briefly reviews a mathematical description of an object surface and defines the curves associated with the path of a round finger rolling or sliding over the object surface. Then, the definitions of two major feature types are provided for the context of robotic haptic exploration. Finally, examples are given of different kinds of macro features that use the basic feature definitions.

3.2.1 Differential Geometry Surface Descriptions

As described in Appendix A, Montana[57] defines a *Gauss map* for a surface patch and a normalized *Gauss frame* at a point on the surface patch. Recall that the axes of the Gauss frame are

$$\begin{aligned} \mathbf{x}(\mathbf{u}) &= \frac{f_u(\mathbf{u})}{\|f_u(\mathbf{u})\|} \\ \mathbf{y}(\mathbf{u}) &= \frac{f_v(\mathbf{u})}{\|f_v(\mathbf{u})\|} \\ \mathbf{z}(\mathbf{u}) &= g(f(\mathbf{u})), \end{aligned} \tag{3.1}$$

and they can be used to calculate the curvature matrix K :

$$K = \begin{bmatrix} \mathbf{x}(\mathbf{u})^T \\ \mathbf{y}(\mathbf{u})^T \end{bmatrix} \begin{bmatrix} \frac{\mathbf{z}_u(\mathbf{u})}{\|f_u(\mathbf{u})\|} & \frac{\mathbf{z}_v(\mathbf{u})}{\|f_v(\mathbf{u})\|} \end{bmatrix}. \tag{3.2}$$

This equation is a 2×3 matrix right-multiplied by a 3×2 matrix, thus K is always a square 2×2 matrix.

As shown in the following section, the curvature matrix can be used to find the principal curvatures and directions of a surface. The Gauss frame is also used to introduce some additional surfaces that are useful for the purposes of feature definition and detection.

Principal Curvatures and Directions

Principal curvatures and their associated principal directions are classical measures that can be used to describe the local shape of a surface. Given a smoothly curving surface (at least C^2 continuous), there will be a direction in which the curvature is greatest, called the *first principal direction*. The curvature of the surface in this direction is called the *first principal curvature*. The vector that is orthogonal to both the first principal direction and the surface normal ($\mathbf{z}(\mathbf{u})$) is called the *second principal direction* and the curvature in that direction is the *second principal curvature*. The second principal direction represents the direction in which the surface is least curved.

There are two cases for which the principal directions are undefined: when the surface is locally spherical or locally planar (at an *umbilic* or *navel* point[84]). The principal curvatures are defined for all smooth surfaces: in the cases of spherical and planar regions they are identical, and zero in the planar case. Similarly, the second principal curvature will be small if the point is near a feature that is a three-dimensional ridge or crack where the curvature is not high in one direction.

If the curvature matrix K is a diagonal matrix (Equation 3.3), then the diagonal elements k_1 and k_2 (where $|k_1| \geq |k_2|$) are the principal curvatures of the surface at the location of the Gauss frame. The principal directions correspond to the $\mathbf{x}(\mathbf{u})$ and $\mathbf{y}(\mathbf{u})$ directions of the Gauss Frame.

$$K = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix} \quad (3.3)$$

One can also calculate the principal curvatures and directions from the *Second Fundamental Form*. Consider the axes of the Gauss frame to be an orthogonal frame

$(\mathbf{x}, \mathbf{y}, \mathbf{z}) = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3)$. The Second Fundamental Form is a matrix of partial derivatives

$$A = \begin{bmatrix} \omega_{11} & \omega_{12} \\ \omega_{21} & \omega_{22} \end{bmatrix}, \quad (3.4)$$

where the elements ω_{ij} are computed as the dot product of e_j and the first derivative of the gradient in the e_i direction. As shown by Interrante[37], it is possible to extract the principal curvatures from A by creating the diagonalized matrix D .

$$A = PDP^{-1} \quad (3.5)$$

$$D = P^{-1}AP \quad (3.6)$$

where

$$D = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix} \quad (3.7)$$

$$P = \begin{bmatrix} p_{1u} & p_{2u} \\ p_{1v} & p_{2v} \end{bmatrix} = [\mathbf{p}_1 \quad \mathbf{p}_2]. \quad (3.8)$$

Again, k_1 and k_2 are the principal curvatures where $|k_1| \geq |k_2|$. The principal directions are the corresponding eigenvectors \mathbf{p}_1 and \mathbf{p}_2 , which are the columns of the matrix P .

As a simple example, consider a robotic finger with a fingertip that is locally spherical (surface S_f) at the contact point with radius r_f . Using the formulation for the curvature matrix above, K_f for the fingertip is

$$K_f = \begin{bmatrix} \frac{1}{r_f} & 0 \\ 0 & \frac{1}{r_f} \end{bmatrix}. \quad (3.9)$$

The principal curvatures for the fingertip are $k_1 = \frac{1}{r_f}$ and $k_2 = \frac{1}{r_f}$. Because the fingertip is (at least locally) spherical, all points on the surface are umbilic points. Thus, the principal curvatures are equal and the principal directions are undefined.

Parallel Surfaces

Now consider that a spherical robot finger is tracing over the surface of an object, keeping point contact at all times. The finger may be rolling and/or sliding. As the

finger moves over the surface of an object, the center point of the finger generates an *offset surface*. (This is the same concept as the offset surface defined in CNC tool path planning and other manufacturing applications. In such applications, one must consider that the commanded position of the tool is not where the cutting will occur, because of the radius of the bit[73].)

The concept of a *parallel surface* can be used to develop a description of the offset surface traced by the fingertip's center of curvature. A parallel surface is constructed by drawing a normal line (determined by the z-axis of the Gauss frame, $\mathbf{z}(\mathbf{u})$) with length r_f at each point on the surface. The locus of the end points of these normal lines is the parallel surface. $\mathbf{s}_p(\mathbf{u}) \in S_p$ maps U to \mathcal{R}^3 :

$$\mathbf{s}_p(\mathbf{u}) = f(\mathbf{u}) + r_f \mathbf{z}(\mathbf{u}), \quad (3.10)$$

where $\mathbf{z}(\mathbf{u})$ is the unit vector in the z-direction of the Gauss frame, r_f is the radius of the fingertip, and \mathbf{u} parameterizes the surface.

Parallel curves can be developed for curves in a plane. Now, at each point, one uses the unit principal normal to the curve (which lies in the plane of the curve) instead of the surface normal. Figure 3.1 shows a series of parallel curves (where different radii r_f are considered), created for an ellipse. Inside the ellipse in Figure 3.1, there are discontinuities in the parallel curves for certain values of the length of the normal line, r_f . These discontinuities are called *caustic points*[42], and occur when the radius of curvature of the object is smaller than $-r_f$. As one can see from the figure, caustic points will occur even when the original curve is smooth.

Similarly, caustic points can occur when generating parallel surfaces. Caustic points are particularly important when considering the path of the center point of a spherical fingertip tracing over the surface, because they represent regions where the fingertip cannot trace out the theoretical parallel surface. Thus, in the region bracketed by caustic points, the finger cannot exactly follow the contours of the object because of interference between the object and fingertip. The implication of this result is that caustic points will be associated with features having high local curvature.

To better illustrate this concept, let us present another surface definition, the *traced surface*. The traced surface is defined as the envelope of spheres with centers on the original surface. Figure 3.2 shows that the only difference between parallel and traced surfaces is at the region surrounding a point where the radius of curvature

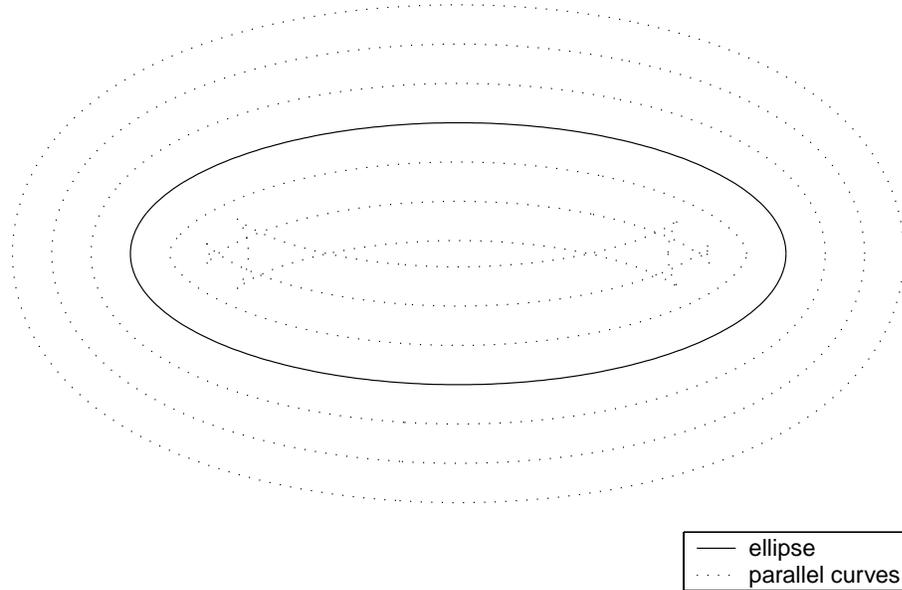


Figure 3.1: A series of parallel curves for an ellipse with a major axis of length 5 and a minor axis of length 2. The length of the normal line varies between -1 and 1.

of the surface is less than the negative of the radius of curvature of the fingertip.

Because the finger travels along a path that is a curve over the original surface, the parallel and traced surfaces may also be discussed in terms of parallel and traced curves. As shown in Figure 3.2, the caustic points bracket the area of the parallel curve where the original curvature, $k_i(S)$, is less than the negative of the finger curvature, $-\frac{1}{r_f}$. Due to interference, however, the path of an actual finger tracing over the surface is limited before the region bracketed by the caustics. The *interference point* is defined as the point where the finger can no longer follow the parallel curve when entering a region of $k_i(S) < -\frac{1}{r_f}$. A locus of interference points creates an *interference curve* for a surface. Any curve on the parallel surface that travels in and out of a continuous area with $k_i(S) < -\frac{1}{r_f}$ must travel through two interference points. Thus, the traced surface, S_t , is the parallel surface without the portions that the fingertip cannot access due to curvature and interference limitations. As the finger moves along a surface, a traced surface will always have a discontinuity at an interference point. The interference points (or curve) can be calculated by finding the location where the traced curve (or surface) intersects itself.

Figure 3.3 shows interference points (discontinuities in the traced surface) and caustic points (discontinuities in the parallel surface). This figure shows that, in addition to the curvature limit that causes caustic points and interference, there is

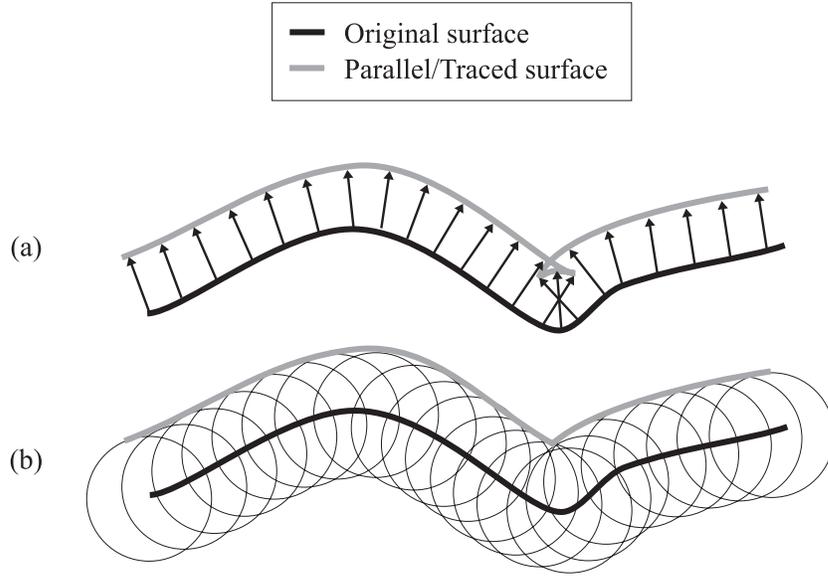


Figure 3.2: 2-D slices showing the (a) *parallel surface*, the locus of endpoints of equal-length lines drawn normal to the original surface, and (b) *traced surface*, the envelope of spheres whose centers are on the original surface. The radii of the spheres and the length of the normal lines are equal.

another phenomenon that occurs when the positive curvature of the object is greater than that of the fingertip. When $k_i(S) > \frac{1}{r_f}$, the curvature of the path of the fingertip is limited by the curvature of the fingertip, $\frac{1}{r_f}$. Figure 3.3 indicates a “cusp” region where this is the case. At a true cusp, the curvature is infinite and the normal is undefined, however, given the measurement capabilities of real sensors, we can consider a cusp to have finite curvature and a unit normal. The curvature limitation ($\frac{1}{r_f}$) of the traced surface over a cusp region has a tendency to round out the traced surface compared to the original surface.

We will now consider one additional surface that will be useful in the description of surface features. The *estimated surface* is the surface obtained by taking the negative traced surface of the traced surface. Shown in Figure 3.4, the estimated surface clearly indicates that there are unobservable regions of the original surface at locations near the interference point. The curvature of the estimated surface in that region is limited by the curvature of the fingertip $\frac{1}{r_f}$. The estimated surface points $\mathbf{s}_e(\mathbf{u}) \in \mathcal{R}^3$ can be determined from the traced surface using

$$\mathbf{s}_e(\mathbf{u}) = \mathbf{s}_t(\mathbf{u}) - r_f \mathbf{z}_t(\mathbf{u}), \quad (3.11)$$

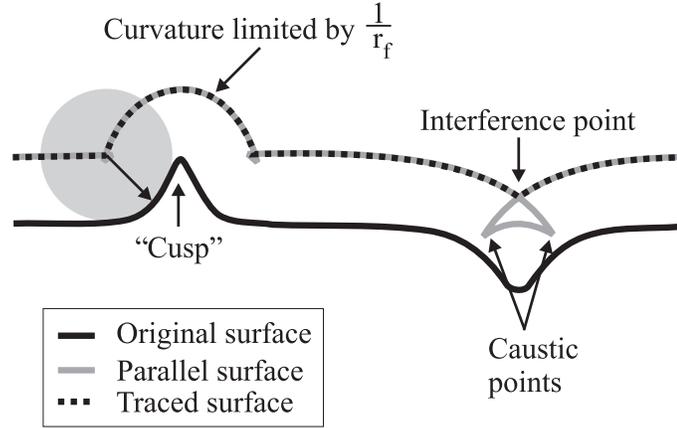


Figure 3.3: 2-D slice of a spherical fingertip with original, parallel, and traced surfaces with interference and caustic points identified.

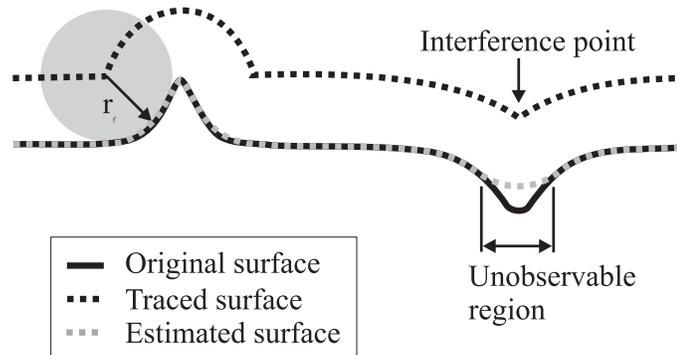


Figure 3.4: 2-D slice of a spherical fingertip with original, traced, and estimated surfaces with the interference point and unobservable region identified.

where $\mathbf{s}_t(\mathbf{u})$ are points on the traced surface, $\mathbf{z}_t(\mathbf{u})$ is the z -direction of the Gauss frame of the traced surface (which is the same as that of the original surface, except at interference points and true cusps on the original surface), r_f is the radius of the fingertip, and \mathbf{u} parameterizes the surface.

3.2.2 A New Feature Definition

Using the surface descriptions developed in the previous section, a new feature definition can be created for the purposes of feature identification. The concept of a feature in the context of haptic exploration with robotic fingers is not only dependent on the surface of the object, but also on the size and shape of the finger. In this work, we assume a spherical fingertip. We begin by defining a *curvature feature*, then use this as a building block to define *macro features*.

Suppose that an object can be locally fit with a surface S with principal curvatures $k_1(S)$ and $k_2(S)$. The basic criterion for defining a feature is maximum principal curvature:

Definition 1 *A curvature feature, as detected by a spherical robotic fingertip with radius r_f tracing over a surface S with an outward normal and curvature K , is a compact region where at least one of the principal curvatures satisfies $k_i > \frac{1}{r_f}$ or $k_i < -\frac{1}{r_f}$. These are positive curvature (convex) and negative curvature (concave) features, respectively.*

This means that the magnitude of the radius of curvature of the fingertip is larger than that of the curvature feature. The simple definition above can be used to define a macro feature, which consists of a pattern of curvature features.

Definition 2 *A macro feature is an compact region on surface S containing one or more curvature features. A macro feature is only formed from multiple curvature features if each curvature feature is near another curvature feature, within a distance equal to the radius of the fingertip.*

There are specific reasons for defining curvature and macro features in this way. For the purposes of haptic exploration with robotic fingers, it is not enough to know the locations of maximal curvature, because the region, or extent, of the feature may be an important property. When creating models of objects using tactile data, it may be desired to create separate local (feature) and global (object) models. Curvature and/or macro features may also be considered as perturbations from the global model. As presented in the definition, macro features are patterns of curvature features that are close enough to each other to be considered part of the same entity.

The curvature feature definition differs from definitions in previous work, which define a “feature” or “ridge” as a location of maximal curvature. The definition here is not a single point with locally maximal curvature, but a region where the curvature is higher than a threshold (the curvature of the fingertip). In addition, previous work in computer vision literature (e.g., Monga, *et al.*[56]) considered a “ridge” feature to occur at each locally maximal point. The macro feature definition here requires a pattern of curvature features and is bracketed by the outermost curvature features in the pattern.

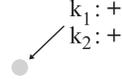
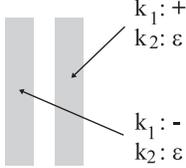
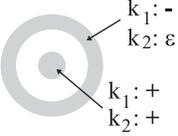
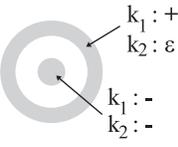
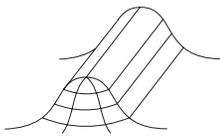
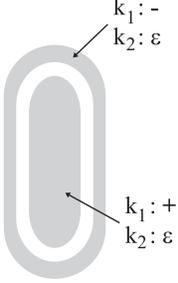
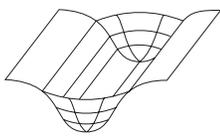
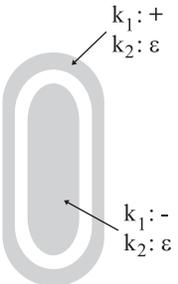
Feature Name	3D picture	2D picture
Convex "Cusp"		
Step		
Bump		
Pit		
Ridge		
Ravine		

Table 3.1: A partial list of possible macro features. In the 2D picture (a plan view of the surface), the gray regions indicate curvature features, with the necessary principal curvatures labeled. In this table, + indicates a positive curvature feature, - indicates a negative curvature feature, and ε is no curvature feature ($|k_i| < \frac{1}{r_f}$). It is assumed that $|k_1| \geq |k_2|$. The white regions between the curvature feature regions have a maximum width of r_f , the radius of the fingertip.

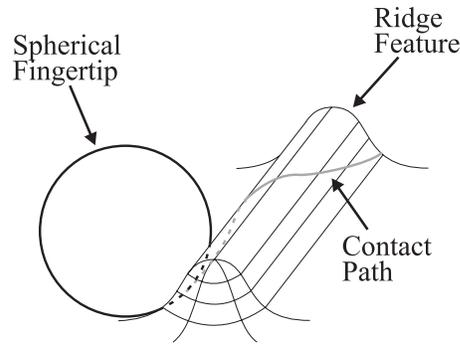


Figure 3.5: The path of the spherical fingertip in this picture does not encounter any high curvature regions as it travels over a ridge at a shallow angle. Because of such cases, where a feature exists but is not detected, macro feature definitions cannot contain any assumptions about the path of the fingertip.

Macro Features

It is generally recognized that defining features is a difficult problem, particularly for the unstructured surfaces we would like to consider for exploration. Using the basic curvature feature definition presented earlier, an infinite number of macro features may be defined. While the minimum number of curvature features required for a macro feature is one (creating a concave or convex cusp), additional curvature features can be used to define more complicated macro features. Table 3.1 shows several possible macro features and their definitions in terms of first and second principal curvatures. While Table 3.1 shows one possible definition for each feature type, it is important to recognize that these definitions and the corresponding macro feature names may change depending on the application. Thus, this list of macro features is meant to demonstrate several possible features, rather than provide definitions for all applications. The general definition of macro features as combinations of curvature features, however, will not change.

It should also be recognized that the macro feature definitions depend only on the surface geometry and the radius of the robotic fingertip. While it is tempting to consider the path of a fingertip as it travels over the feature, there is no guarantee that any given exploratory procedure will cause the finger to travel in a direction that will encounter and detect curvature features. For example, if a finger crosses a ridge at a very shallow angle (Figure 3.5), no curvature features will be detected. Thus, the macro feature definitions do not use any assumptions about the fingertip path.

As an example of a macro feature, a *bump feature* is a region where the first

principal curvature creates a positive curvature feature in a ring. Inside that ring, there must be a region where both the first and second principal curvatures create positive curvature features. Because curvature features must be close enough to each other to consider them part of the same macro feature, the largest distance between the two regions is the radius of the fingertip, r_f . One possible definition of a *ridge feature* is a region where the first principal curvature creates a negative curvature feature region around the border, and a positive curvature feature region on the inside. The ridge is differentiated from the bump by the fact the the second principal curvature at the inner region is less than the curvature of the fingertip, or approximately zero. For all the features shown in Table 3.1, a curvature of ε corresponds to any curvature less than that of the fingertip. Based on these feature definitions, there is nothing preventing features that close upon themselves, such a ridge in a ring shape, or a step feature that is revolved to create a plateau.

An interesting application of these feature definitions is that one can perform multiscale observations of features on a surface. Given a surface explored with an actual fingertip of radius r_f , certain regions on the surface will or will not be considered curvature features. After a robotic finger has already explored a surface and calculated an estimated surface, it is possible to assume different fingertip radii $r_i > r_f$ that will allow for simulated feature detection on different scales. r_i must be greater than or equal to r_f because the unobservable regions of the original exploration cannot be used to detect any additional features.

3.2.3 Example: A Bump Feature

Consider the three-dimensional parabolic surface of revolution “bump” shown in Figure 3.6. The bump surface is defined by the set

$$U = \{(u, v) \mid -1 < u < 1, 0 < v < \pi\} \quad (3.12)$$

and the map

$$\begin{aligned} f : U &\rightarrow \mathcal{R}^3, \\ (u, v) &\mapsto \left(u \cos v, u \sin v, \frac{1}{1 + au^2}\right) \end{aligned} \quad (3.13)$$

for some $a \in \mathcal{R}$.

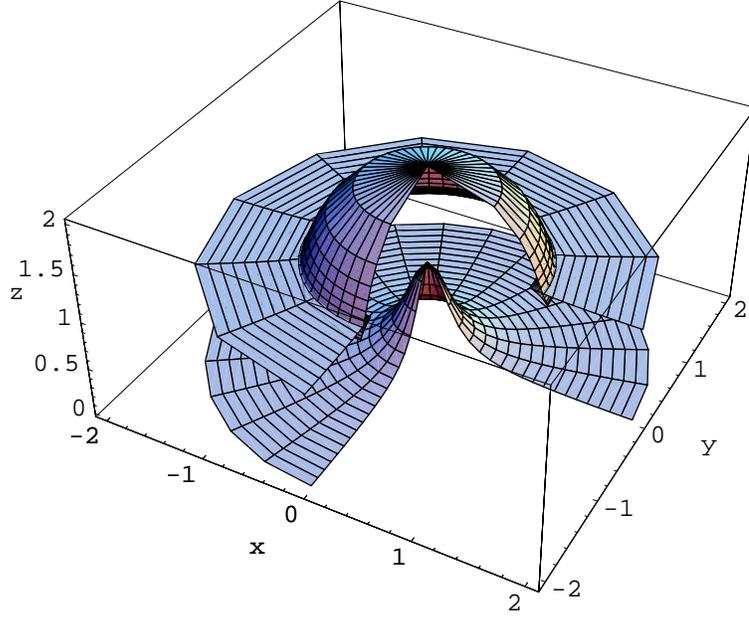


Figure 3.6: A bump feature (bottom), with parallel surface (top).

It can be shown that (f, U) is an orthogonal coordinate system and thus the normalized Gauss frame exists for all $\mathbf{u} = (u, v) \in U$. For the bump example, the Gauss frame is

$$\begin{aligned}
 \mathbf{x}(\mathbf{u}) &= \left[\frac{1}{c} \cos v \quad \frac{1}{c} \sin v \quad \frac{2au}{cd^2} \right]^T \\
 \mathbf{y}(\mathbf{u}) &= \left[-\sin v \quad \cos v \quad 0 \right]^T \\
 \mathbf{z}(\mathbf{u}) &= \left[\frac{2au}{cd^2} \cos v \quad \frac{2au}{cd^2} \sin v \quad \frac{1}{c} \right]^T,
 \end{aligned} \tag{3.14}$$

where

$$c = \sqrt{1 + \frac{4a^2u^2}{(1+au^2)^4}} \tag{3.15}$$

$$d = 1 + au^2. \tag{3.16}$$

The curvature matrix is

$$K = \begin{bmatrix} \frac{2ad(1-3au^2)}{c(d^4+4a^2u^2)} & 0 \\ 0 & \frac{2a}{cd^2} \end{bmatrix}. \tag{3.17}$$

Using the simplification variables c and d defined in Equations (3.15) and (3.16),

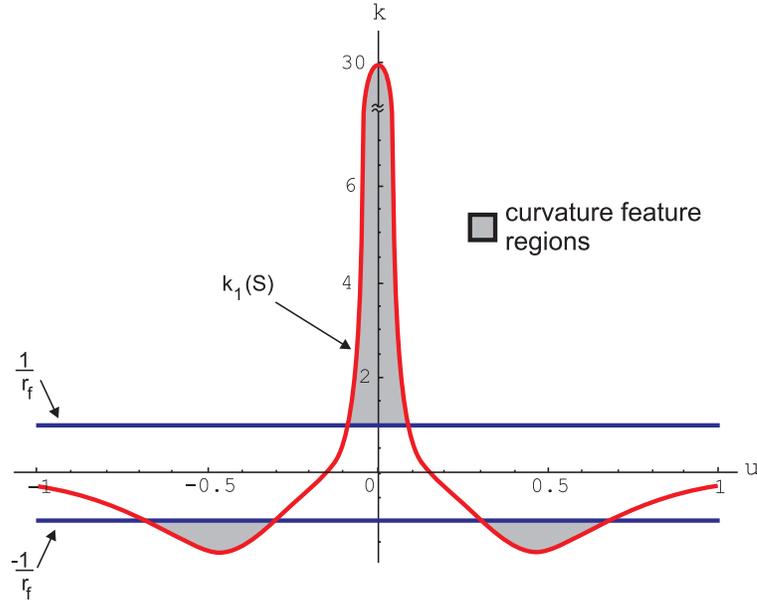


Figure 3.7: Curvature features are identified where $|k_i(S)| > \frac{1}{r_f}$.

the parallel surface is defined by points

$$\mathbf{s}_p(\mathbf{u}) = \begin{bmatrix} \left(u + \frac{2ar_f u}{d^2 c}\right) \cos v \\ \left(u + \frac{2ar_f u \sin v}{d^2 c}\right) \sin v \\ \frac{1}{d} + \frac{r_f}{c} \end{bmatrix}. \quad (3.18)$$

To determine the locations of the curvature features, we can plot the principal curvatures of the object surface and the robotic fingertip with respect to the variable u . Figure 3.7 shows the principal curvature $k_1(S)$ (the K_{11} element of the matrix in Equation 3.17) and $\pm \frac{1}{r_f}$, where $r_f = 1$, plotted against u for $a = 15$. There are three curvature features: two negative, and one positive. The pattern of curvature features can be identified as a bump as presented in Table 3.1.

3.3 Feature Detection

In this section, algorithms are presented for the detection of surface features using the curvature and macro feature definitions.

Figure 3.8 shows how position and tactile sensor data from the fingertip can be used to identify features. Different chains on the diagram result in different identification algorithms. The algorithms can be divided into two groups: those that use

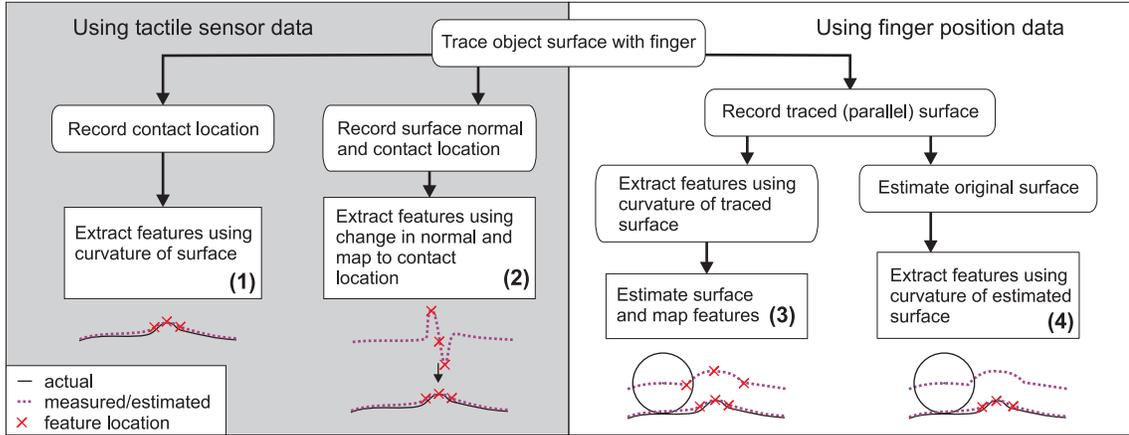


Figure 3.8: Algorithms for feature detection.

tactile sensor data to find the object surface and features, and those that use finger position data.

3.3.1 Using Tactile Sensor Data

Using the tactile sensor data and a model of the robot finger, one can compute the trajectory of the finger/object contact point in space and build a model of the surface. (This model is the estimated surface.) This method works for any fingertip shape, as long as the contact point can accurately be sensed. However, this method is limited because most tactile sensors are somewhat noisy and of low resolution and bandwidth compared to joint angle and force sensors. Contact location data may not be accurate enough.

The first algorithm (1) in Figure 3.8 calculates the curvature at each point on the estimated surface using a numerical difference method. Depending on the noise in the data, the points on the estimated surface may first need to be smoothed or fit to an analytical model before differentiation. Sets of points on the smoothed or modeled surface with a curvature greater than that of the fingertip may then be identified as features. Methods for smoothing data and calculating curvature are described in Section 3.3.3 and Appendix B.

The second algorithm (2) in Figure 3.8 requires that the normal of the surface be recorded over time. With a spherical fingertip, this is easily determined from the contact point on the finger.

$$\mathbf{n} = \frac{\mathbf{c}_f - \mathbf{c}_s}{\|\mathbf{c}_f - \mathbf{c}_s\|} \quad (3.19)$$

where \mathbf{n} is the unit normal vector to the surface, \mathbf{c}_f is the center point of the spherical fingertip, and \mathbf{c}_s is the contact point between the finger and the surface. With some tactile sensors[51], the surface normal is the directly sensed quantity, so obtaining the surface normal does not require any fingertip position data.

Again, due to noise in the sensor or position of the robot finger, the normal may need to be smoothed. The normal is differentiated with respect to arclength of the contact path, providing a spike at the location of the feature. This location can then be mapped to the estimated surface. This method uses the same information as the first algorithm, but may improve feature detection because, with some tactile sensors, the surface normal is directly sensed. This is superior to taking the derivative of a noisy surface estimation.

3.3.2 Using Fingertip Center Position Data

Feature detection can also be accomplished without tactile sensor data, using the traced surface. If the fingertip is spherical (in 3D) or circular (in 2D), the traced surface created by the center of the fingertip can be used to estimate the original surface without tactile sensor data. Estimation of the original surface can be done using Equation (3.11). This method, however, is very susceptible to noise in $\mathbf{z}_t(\mathbf{u})$. Before calculating $\mathbf{z}_t(\mathbf{u})$, an improved estimation of the surface can be obtained by fitting a curve to or smoothing the traced surface.

The third algorithm (3) in Figure 3.8 extracts the features based only on the curvature of the traced surface. This has an advantage over the previous methods because the traced surface is likely to be less noisy than the estimated surface which used tactile sensor data. In addition, the traced surface will show interference points, which are a good indicators of features because they are discontinuities in the traced surface. This serves to enhance negative curvature features.

The fourth algorithm (4) in Figure 3.8 is similar to the previous algorithm, although the order of surface estimation and feature detection has been reversed. Now we use the traced surface to estimate the original surface first, and then perform feature detection using the curvature of the estimated surface.

A key observation in the last two algorithms is that tactile sensor data is not necessary. While contact point information may be necessary for control, this data is not used in the last two feature detection algorithms. Instead, these algorithms rely on the relatively accurate and non-noisy proprioceptive information from the

robot's joint angle sensors. (Proprioception is the ability to sense of the absolute position of the body or limbs.) While humans in general have poorer proprioception than robots, we do have superior tactile sensing. The algorithms developed here for robots are certainly not the way that humans detect features. It is important in the development of these algorithms to capitalize on the strengths of the system doing the exploration; for the case of robotic fingers, this means using position rather than tactile sensing.

3.3.3 Tactile Data Smoothing

Because both tactile and fingertip position data are inherently noisy, filtering algorithms can be used to create smoother data for curvature calculation in the feature detection algorithms. Curvature calculation is a second derivative computation, thus it requires smooth data sets for accurate curvature estimation. There are two main approaches to smoothing that will be addressed: smoothing using curvature limitations, and smoothing based on noise type.

Smoothing Using Curvature Limitations

For algorithms that use only the position of the fingertip, two nonlinear smoothing algorithms based on curvature limitations can be used to smooth noisy data in the traced and estimated surfaces. The algorithms apply the observations in Section 3.2.1 that the traced surface must round sharp cusps or corners on the object surface and the estimated surface must fillet sharp indentations.

At each point on the traced and estimated surfaces, the principal curvatures are calculated using a numerical difference method. On the traced surface, any points for which $k_i(S_t) > \frac{1}{r_f}$ are invalid (i.e., they cannot correspond to motion along an object surface) and should be deleted. After removing these points, the traced surface can be smoothed by any standard method. After the traced surface is smoothed, the estimated object surface is calculated using the *envelope* of r_f circles centered on the traced surface. In the estimated surface, the principal curvature $k_i(S_t)$ must be at least $-\frac{1}{r_f}$. Although the actual object surface could have regions with larger negative curvature, it is impossible for the fingertip to detect such regions. Again, such points should be deleted and the remaining surface can be smoothed using any standard method.

Smoothing Using Noise Type

The previous smoothing method was specific to the expected geometry of the tactile data. Smoothing using noise type, however, can be used on any kind of data provided that the noise type matches an existing smoothing technique.

In order to determine which filters might be appropriate for smoothing tactile and position sensor data, a χ^2 (Chi-squared) test was implemented. The χ^2 test is a goodness-of-fit measure that can be used to test the probability that a sample of data is or is not normally distributed[7, 55]. It was determined that the noise was normally distributed, so Gaussian or Wiener filters were targeted as the best types of filters for smoothing the tactile data. Appendix B describes the process for determining noise type and a comparison of several filtering techniques. The smoothing algorithms were evaluated based on their ability to minimize the number of false positive and false negative feature identifications.

3.4 Simulation and Experiments

3.4.1 Simulated Data for Surface Estimation

A realistic simulation of a spherical fingertip traveling over a step shows the curvature limitations in surface estimation. For the purposes of simulation, a realistic traced surface must be calculated. First, the parallel surface is calculated from the original surface, then the interference points are identified and the unreachable points are removed to form the traced surface. Next, because points on the parallel surface are not spaced equally, the surface is re-calculated using equal arclengths between the points. Finally, Gaussian noise with a variance (σ) of 0.01 is added for realism.

Nonlinear smoothing algorithms are then invoked to limit the curvature to realistic values. The first curvature limitation algorithm is used to remove unreachable points from the traced surface, then it is smoothed. Then, the original surface is estimated using the envelope of circles with centers on the smoothed traced surface. Next, the second curvature limitation algorithm is used to remove unreachable points from the estimated surface. As a final step, the surface can be fit to create an analytical model. Figure 3.9 shows a 2D slice of the following four surfaces for this example: the object surface, the traced surface with noise, the smoothed traced surface, and the smoothed estimated object surface.

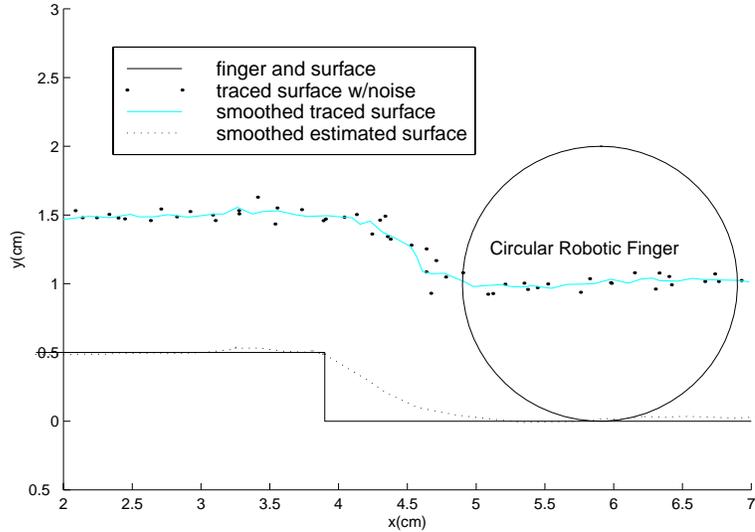


Figure 3.9: Simulated data of a finger tracing over a step.

3.4.2 Experiments for Feature Detection

Apparatus

Manipulation experiments were performed using two-degree-of-freedom robotic fingers and tactile sensors developed by Maekawa, *et al.*[50, 51]. The optical waveguide tactile sensors on the fingers form a hemispherical fingertip and provide analog signals that can be used to calculate the intensity and centroid of the contact point(s). The 20 mm diameter sensors can be sampled at 5 kHz, and have a field of detection of ± 75 degrees from the sensor pole. The error is approximately ± 1 degree, although this can change when the contact area increases during sliding. A calibration was performed to characterize and remove the nonlinearity of the sensor. Appendix C provides a detailed description of this tactile sensor.

As is typical of many robotic fingers, the workspace was limited and thus a combination of rolling and sliding was necessary to move the fingers over the surface of the object. A hybrid force/velocity control was used to obtain smooth sliding over flat surfaces with bump features (0.5-1.5mm diameter wires placed on the surface).

During motion, joint-angle potentiometers were used to determine the position of the center of the hemispherical fingertip. The location of the contact point on the fingertip was then used to determine the direction of the contact normal, which was filtered to reduce noise. The center of the fingertip can be sensed to within ± 0.2 mm

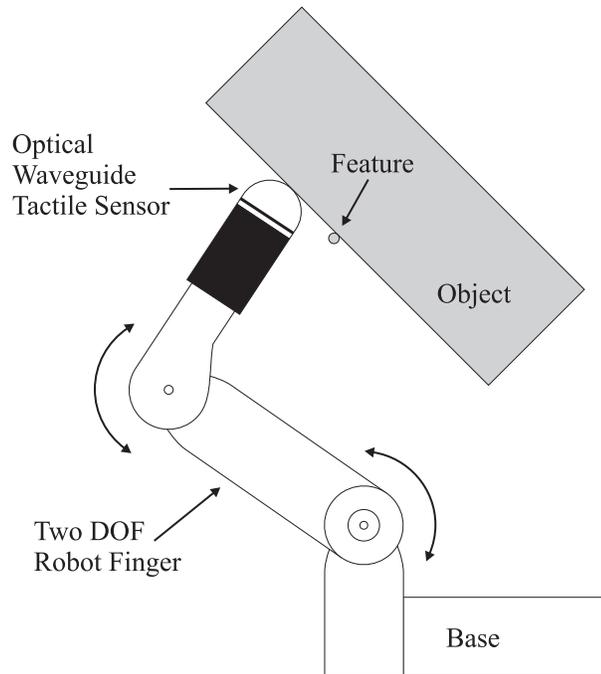


Figure 3.10: The experimental apparatus, a two-degree-of-freedom robot finger equipped with a tactile sensor.

and the contact location can be estimated within ± 0.5 mm. A tension differential-type torque sensor is used to measure torque in the joints and calculate the Cartesian force at the fingertip.

Because the fingertip is spherical, the contact location on the finger gives the tangent and normal of the rigid surface. The velocity of the fingertip tangent to the surface and the force normal to the surface were controlled using a simple proportional law. The finger moved with an average speed of 0.03 m/sec and a normal force of 1 ± 0.01 N.

Results

Each algorithm outlined in Figure 3.8 was tested using the data from these experiments. Algorithm (1) often resulted in false negative identifications because the estimated surface was obtained directly from the contact point. Concave curvature features on the object were overlooked because of curvature limitations; the data for these points are automatically smoothed to the radius of curvature of the fingertip. Using algorithm (2), spikes in the contact normal indicated the presence of negative

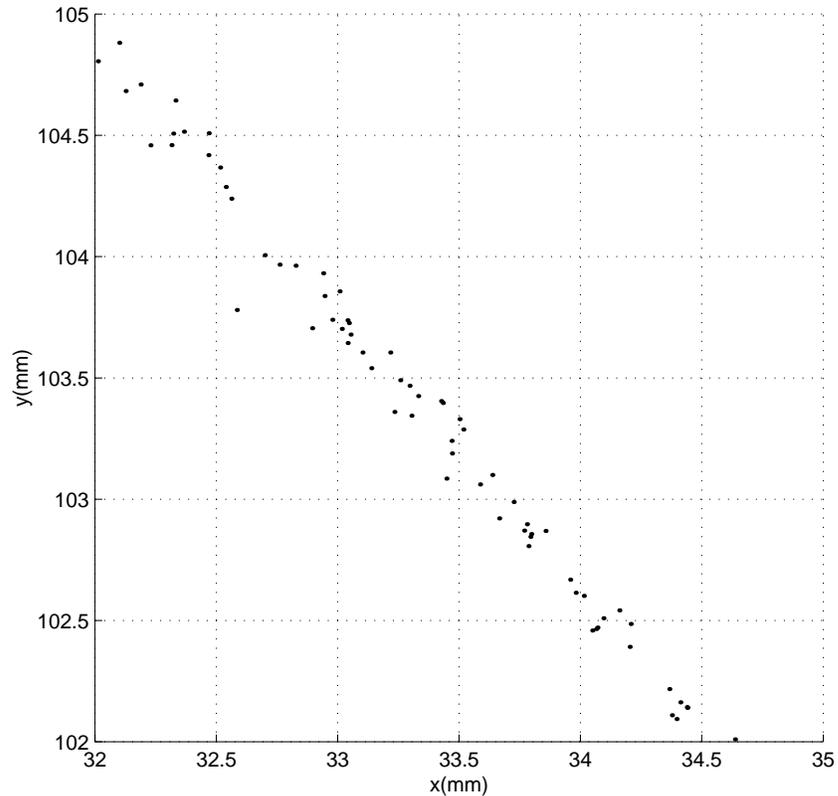


Figure 3.11: Magnified view of unfiltered estimated object surface data.

curvature (concave) features. This method does not provide good detection of positive (convex) features, because the normal direction does not change as quickly at those points.

Using algorithm (3), features were extracted using the curvature of the traced surface, then mapped to an estimated surface. This algorithm performed the best for the application of small bump features on a flat surface. Figure 3.11 shows a magnified view of the unsmoothed estimated object surface data in a region with no features. Figure 3.12 shows the traced and estimated surfaces and features detected for a bump on a flat surface angled at 45 degrees, with a 0.65 mm diameter wire stretched across the surface. The orientation of the object is the same as that in Figure 3.10. Algorithm (4) resulted in false negatives for the same reason as algorithm (1): using the estimated surface to determine locations with high curvature is not feasible because of the curvature limitations.

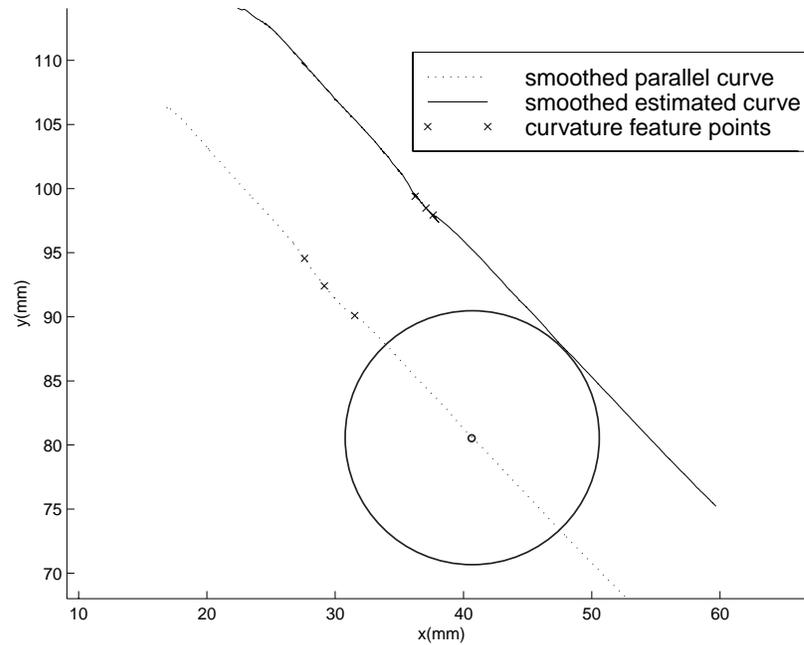


Figure 3.12: Feature detection with a robotic finger rolling/sliding on a 45° surface with a 0.65 mm bump feature. The finger size is shown for scale. (See Figure 3.11 for a magnified view of original data points.)

3.5 Conclusion

In this Chapter, features were defined for the context of robotic haptic exploration with curved fingertips. Features may be identified using the curvatures of traced or estimated surfaces, with or without tactile sensor data. In particular, the traced surface described by a spherical fingertip accentuates concave features on the object surface. This is an interesting result because, while contact location data may be necessary for stable manipulation and exploration control, such data is not needed to recreate the object shape or to identify small surface features. Definitions for curvature and macro surface features were distinguished; macro features are patterns of curvature features that may be used to define 3-D and compound features such as bumps, ridges, and ravines.

Methods for filtering were also addressed based on the inherent curvature limitations of the physical system and the type of noise present in the tactile or position data. Because the calculation of curvature requires second-order derivatives, it is essential that the data be smooth.

While these feature definitions are useful for detecting and identifying features, the problem of detailed exploration of features still remains. The experimental work

for this chapter used a two-degree-of-freedom robot finger. In this case, the finger was able to move back and forth over a feature, but there was no opportunity to do further exploration in other directions. In addition, it is possible that the feature may not be oriented in such a way to allow the finger to travel in a principal direction. Thus, the next chapter addresses the development of 3D haptic exploratory procedures that use the curvature and macro feature definitions to efficiently explore features and develop a global map of features on an object.

Chapter 4

Feature-Guided Exploration

In the previous chapters, a procedure for multifingered haptic exploration was developed and surface features were defined for detection and identification purposes. We now consider the situation where a robotic finger has sufficient degrees of freedom to actively explore a surface in three dimensions. This is termed “feature-guided” exploration because the discovery of a feature leads to changes in the fingertip path. Features may be detected when a finger’s path simply moves the contact point over the feature, however, complete feature type and shape identification may require further exploration. For local exploration, it is assumed that a feature has been detected and further exploration is desired. In this case, the robot finger’s trajectory is modified in order to collect additional data about the feature. While planning for global exploration is not discussed in this work, we present a method for partitioning the surface that can be used to choose the next direction of exploration based on the features already found. In both local and global exploratory procedures, the finger uses feedback about discovered surface properties to actively plan new explorations.

Feature-guided haptic exploration as addressed in this thesis has two levels. The first is local exploration: Upon discovering a feature, how should the finger navigate around or over it in order to extract desired feature properties? This involves issues of control and tactile sensor data interpretation, and uses the feature definitions presented in Chapter 3. The second problem is how to model the features on an object. This model can be used in the planning and execution of global exploration by mapping the features on the surface of the object.

4.1 Previous Work

Although there have been several investigations of haptic exploration, active or feature-guided exploration in three dimensions has received relatively little attention in the literature. The relevant areas of research are tactile servoing, surface modeling from point data, and skeletonization algorithms for object modeling.

4.1.1 Tactile Servoing and Hybrid Control

Tactile servoing is the process of using data from tactile sensors to control the motion of a robotic hand and the object it is manipulating. Hemami, *et al.*[6, 70, 31, 30] performed surface tracking with a robotic finger equipped with a tactile sensor. Using a low cost, low resolution end-effector force sensor, they developed a learning controller that was designed to reject sensor noise and perform surface estimation for a generalized quadric shape. They also used a hybrid velocity/force control to move a robot end-effector over the surface of an object, although geometries were known a priori so tactile data was not necessary. In other work, they showed that tactile data can be used with a controller to track edges of a certain sharpness. Sikka, *et al.*[79] also performed tactile servoing. Drawing an analogy to image-based visual servoing, they monitor the progress of a manipulation task and recover object geometry using tactile images on an array-type sensor. A similar system was used by Allen[3] to explore regions of an object occluded from a vision system. Maekawa, *et al.*[49] stably manipulated an unknown object while using tactile sensor feedback to observe the motion of contact points during rolling. The feature tracing work presented in this thesis differs from these tactile servoing techniques because the goal is for the finger to trace the sides of a feature on a surface rather than along a sharp edge.

4.1.2 Object Modeling from Data

Object modeling from data has been used primarily in the computer vision and graphics communities to build up global object models from cloud points of data. Solina and Bajcsy[81] used data from laser range finders for fitting superquadrics. Modified superquadric shapes were obtained by bending and tapering. Chek, *et al.*[47] also used implicit surfaces to model data sets consisting of cloud points. Algebraic functions that closely approximated the data were obtained by Implicit Solid Modeling, a constructive scheme for approximating Boolean volume set operations on implicitly

defined primitive volumes. However, none of the previous work has used tactile data or focused on the skeletonization and mapping of features on a surface. One goal of the procedure for haptic exploration described in this chapter is to obtain data for such a modeling algorithm.

4.1.3 Skeletonization

Shape skeletons are geometric abstractions of curves, surfaces, or solids that are useful as lower-dimensional representations. The skeleton is known in 2D as the *medial axis* and in 3D as the *medial surface*. The Medial Axis Transform (MAT) is a skeletonization technique that was first developed by Blum[9] as an alternate shape description for biological applications. In a famous analogy, Blum proposed that the medial axis can be found by considering the object as a patch of grass whose boundary is set on fire. As the fire-front propagates to the interior of the patch, multiple flame fronts will meet at minimal paths to the boundary, forming the medial axis. Some of the first MAT algorithms used this approach; in this case, the medial axis is created by uniform shrinking of the boundaries of the object[88].

There are many other algorithms for extracting the MAT of a 2D image. Chiang[17] provides a good overview by describing five different methods. Niblack, *et al.*[61] developed another algorithm, for which they prove connectivity and use a distance transform, which is a good approximation to the Euclidean distance. Another MAT algorithm was designed by Shih and Pu[76] that trims branches to make the skeleton simpler and more useful for object recognition. Sudhakar, *et al.*[88] identified the properties of the MAT which are of primary interest to engineering design: dimensional reduction, homotopic equivalence, and invertibility. Two maps are homotopic if one can be deformed into the other. For discrete objects, they define a new skeleton which shares these properties with the MAT for 2D and 3D cases. Skeleton-based modeling operations on solids and methods for computing skeletons of free-form solids are also considered by Storti, *et al.*[86] and Turkiyyah, *et al.*[91]. A more detailed description of the MAT and how it can be calculated from tactile data is provided in Section 4.4.3.

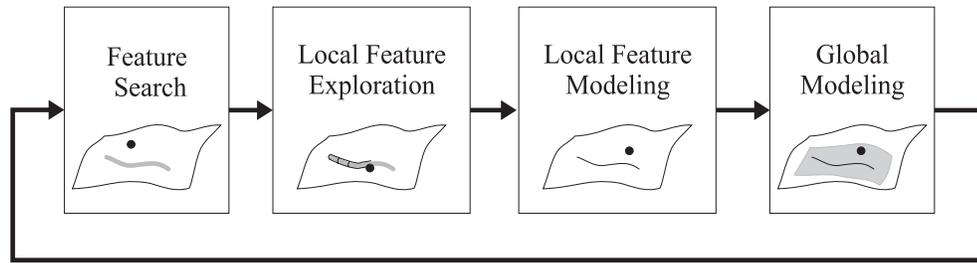


Figure 4.1: An overview of the phases for local and global haptic exploration.

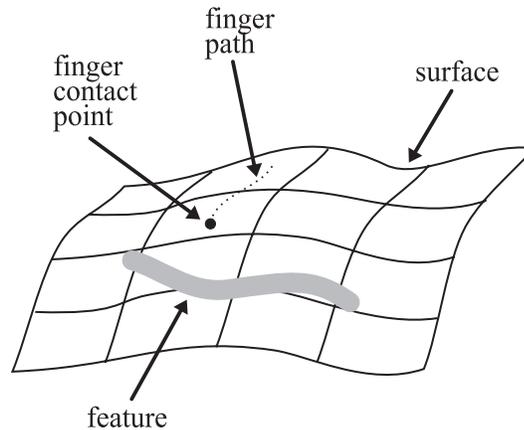


Figure 4.2: The first phase in exploration is moving the finger over the object during manipulation or while searching for a feature.

4.2 A Procedure for Feature-Guided Exploration

This section describes a high-level procedure for feature-guided haptic exploration. Details about the phases of active haptic exploration are presented in the other sections of this chapter. Figure 4.1 shows the progression of exploration using these phases. In this figure, the black dot represents the contact point of the finger performing the exploration.

A feature may be discovered on the surface of an object during a manipulation task, a random walk of the fingertip over the surface, or a specific multifingered exploratory procedure as described in Chapter 1. For the purposes of this work, let us assume that one of these three finger-motions results in the finger encountering a feature while rolling and sliding over the surface. Control for this motion on a surface in 3D space is described in Section 4.3. This fingertip motion corresponds to the “feature search” phase of exploration as shown in Figure 4.2.

The first step in a feature encounter is identification of the feature type. A basic algorithm for feature detection was provided in the previous chapter. In this chapter,

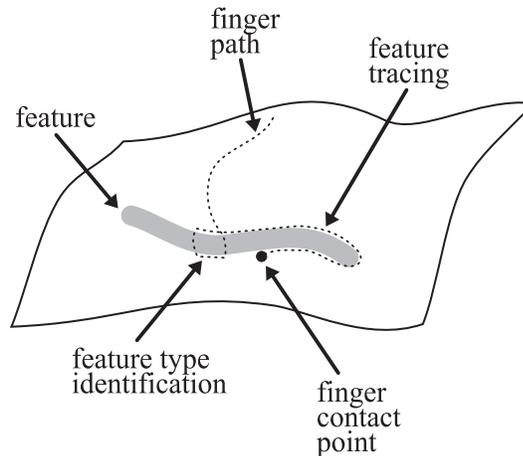


Figure 4.3: The second phase in exploration is moving the finger over and around the feature to determine type and record shape.

that work is expanded to include active exploration of three-dimensional features. In order to identify feature type, the finger must go through a local exploratory procedure that will extract feature properties for type identification. Then, the feature can be modeled by tracing to find its boundary. For each feature, a shape skeleton, or 2D medial axis, may be created in a coordinate system fixed on the surface of the object. This is known as a *feature skeleton*. The procedures for determining feature type, tracing the boundary, and modeling using skeletons are described in Section 4.4 and correspond to the “local feature exploration” phase in Figure 4.3.

Another phase of feature-guided exploration uses the feature shape skeletons to create a global map of features on the surface, called a *global skeleton* (Figure 4.4). This “global modeling” is shown in the last block of Figure 4.1, and can be done for the entire object surface or for any explored region. By dividing the surface into regions for different features, one can plan manipulation and further exploration. The mechanics of planning for global exploration using the global skeleton are not addressed in this work, however, we do consider several possible paradigms for global feature search.

Depending on the application and spatial frequency of the features, there is a trade-off between complete exploration of individual features (performing the local exploratory procedures described above) and continuing the search to find the next feature. As will be shown, for a general application where knowledge of each feature is important, the most efficient subsequent motion is to continue exploring any feature until it has been completely traced. However, there may be specific applications in

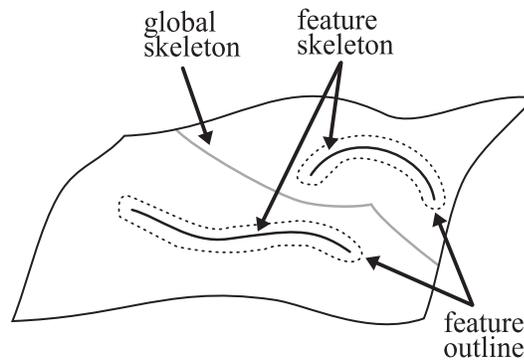


Figure 4.4: As the finger explores more features, the skeletons of the features are used to build a global skeleton that divides the surface into regions.

which exploration is performed to find one specific feature; as soon as a feature is detected, a decision can be made about the likelihood of that feature being the goal.

4.2.1 Goals and Assumptions

No exploratory procedure will work in all situations; depending on the nature of the surface, the features, and the robotic hand doing the exploration, the general procedure for exploration will perform with varying degrees of success. Thus, this section states the goals of exploration and assumptions used in this work.

Goals

The goals of active exploration as described in this chapter are:

- For each feature explored, identify feature type and create a skeleton of its shape.
- Create a global skeleton based on the features.

In addition, we consider methods for performing global exploration depending on the nature of the features and their spatial distribution, but do not formulate a global exploration goal in this work.

Assumptions

There are a number of assumptions that are necessary to guarantee success of feature-guided exploration, where success is defined as meeting the goals stated above. The assumptions are:

- The viewer is either a single robotic finger or a multifingered hand for which at least one finger can report contact location while exploring. The reported contact location and the local exploratory procedures (type identification and boundary tracing) discussed in the next section can be used to control the finger to trace around features that have boundaries, such as ridges and ravines. The finger must have a shape that will allow it to either accurately record contact location for calculating the curvature of the surface, or be spherical so that an offset surface can be used to identify curvature features as discussed in Chapter 3.
- The item is held stationary by a fixture or other fingers.
- The region of the object to be explored is a surface patch that is accessible by a single finger (with contact sensing) without manipulation or regrasping. The surface is bounded by a limit curve defined by the workspace of the finger. By “global exploration,” we refer to the search and exploration of multiple features in this region.
- If a feature is not completely contained within the region under consideration, the accessible portion of the feature may be explored. Later, models of different regions of the surface may be joined together to connect features that were partially explored.
- Due to the hardware used in the experiments for local feature exploration, we also provide two simplifying assumptions. The first is that the closest points on any two features are at least one fingertip diameter apart. This is so that the finger can trace the boundary of one feature without being distracted by other feature contacts. The second is that each surface feature is one of the basic types shown in Figure 3.1. In general, it is possible to identify and trace any macro feature type, however, we simplify our approach by considering only a few examples. Compound features, such as a ridge that suddenly dips down and becomes a ravine, would be especially difficult to trace with only contact point sensing.

4.3 Control for Haptic Exploration

Given the goals of exploration and assumptions about the features and their distribution, we now develop the control for haptic exploration. If we are to use the feature definitions presented in Chapter 3, this control law must cause the finger to travel parallel to the surface while maintaining contact. Tactile sensor feedback is used to determine the current surface normal and provide information for the feature detection algorithms.

4.3.1 Hardware

Experiments for controlling a finger in 3D using tactile feedback to detect features require a three-degree-of-freedom robotic finger with a tactile sensor. The tactile sensor can be any shape, but a spherical one is most useful for feature tracing.

The robotic finger used was the 3GM haptic interface from Immersion Corporation[27]. While the device was originally created as a haptic interface, the software for controlling it was easily modified to make it into a robotic finger. The 3GM has three degrees-of-freedom of motion at the end effector, and three independent actuators. Because the device is backdrivable and lightweight, gravity compensation and open-loop force control can be used in the device control. Details about the 3GM can be found in Appendix C.

The tactile sensor used was the Optical Waveguide Tactile Sensor from Maekawa, *et al.*[51] Details of its function are also described in Appendix C. This tactile sensor uses an analog position-sensitive device (PSD) to measure the reflection of light from the point(s) of contact. It can report contact centroid and contact intensity, which is related to the amount of light reflected. Because the sensor is analog and fast, data can be obtained at the 1kHz rate used for controlling the finger. Figure 4.5 shows a picture of the robotic finger and tactile sensor exploring a flat surface with a ridge feature.

4.3.2 Coordinate Systems

The basic control law uses proportional-derivative (PD) control parallel to the surface and open-loop force control perpendicular to the surface. Thus, several coordinate systems must be considered in the control for surface tracking, as shown in Figure 4.6. The World Coordinate System (WCS) is a fixed coordinate system. For the purposes

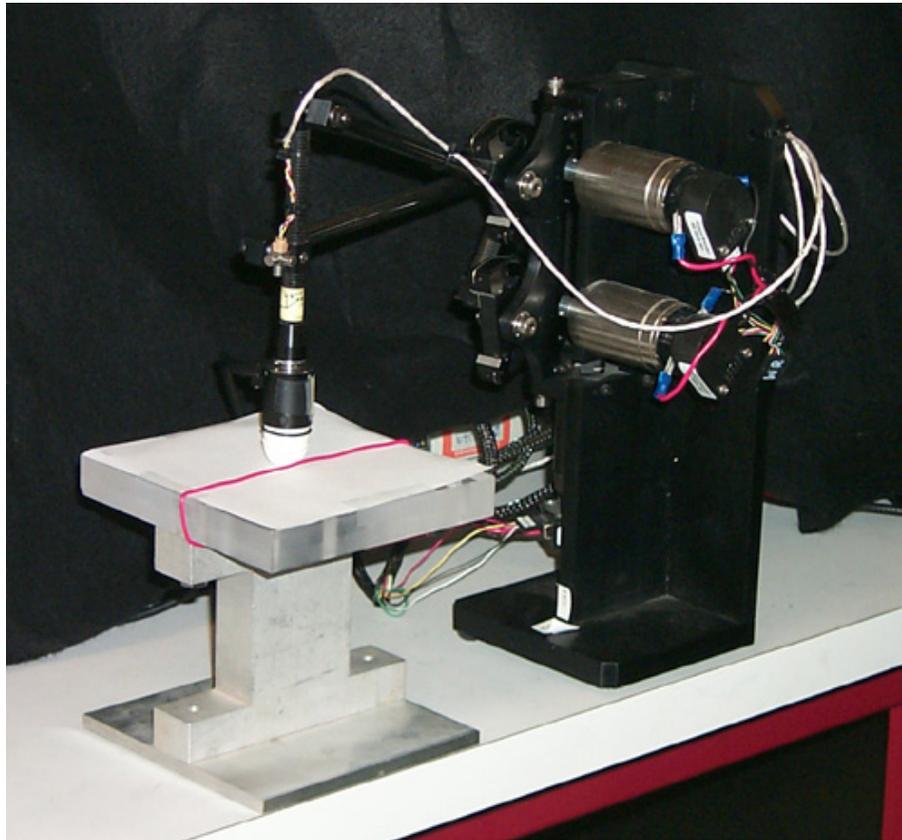


Figure 4.5: The 3GM as a robotic finger, equipped with an Optical Waveguide Tactile Sensor. The finger is exploring a flat surface with a single ridge feature.

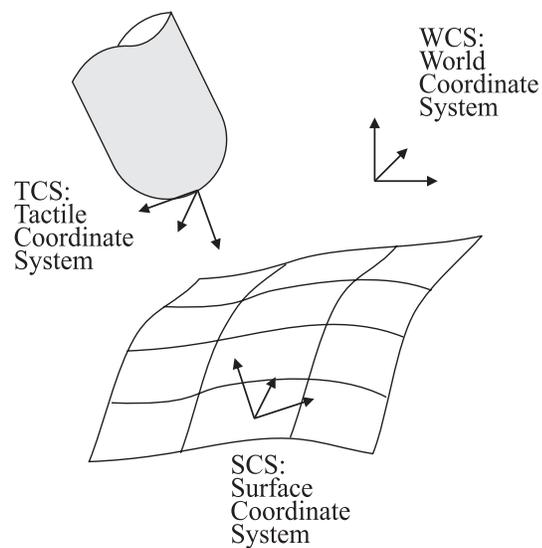


Figure 4.6: Coordinate systems for haptic exploration.

of single-fingered exploration, the base of the finger is stationary and can be assigned to the WCS. The Tactile Coordinate System (TCS) is attached to the finger, at the center of the sensor if the fingertip is spherical. Using the kinematics of the finger, the location of the TCS can always be calculated relative to the WCS. The z -axis of the TCS always points normal to the finger at the pole of the hemisphere. The rotation matrix from the TCS to the WCS is

$$\begin{aligned}\alpha &= \frac{\pi}{2} + \gamma_1 \\ \beta &= \frac{\pi}{2} - \theta_3 \\ R_{TW} &= \begin{bmatrix} \cos(\alpha) & -\sin(\alpha)\cos(\beta) & \sin(\alpha)\sin(\beta) \\ \sin(\alpha) & \cos(\alpha)\cos(\beta) & -\cos(\alpha)\sin(\beta) \\ 0 & \sin(\beta) & \cos(\beta) \end{bmatrix},\end{aligned}\tag{4.1}$$

where γ_1 and θ_3 are two of the joint angles of the robot finger as defined in Appendix C.

The Surface Coordinate System (SCS) is constantly updated at the point of contact. The SCS is calculated using the contact point information from the tactile sensor. When location data is obtained from the sensor, the direction of the contact normal is recorded by the unit vector \mathbf{n}_T in the TCS. The normal direction is then converted to the world coordinate system using the rotation matrix R_{TW} to become \mathbf{n}_W and low pass filtered to remove noise.

$$\mathbf{n}_{W_k} = 0.05R_{TW}\mathbf{n}_T + 0.95\mathbf{n}_{W_{k-1}}\tag{4.2}$$

If there is no contact, the normal returned is $\mathbf{n}_T = \{0, 0, 0\}$. When this case is detected, the assumed contact normal is the z -axis of the TCS.

The z -axis of the SCS is assigned to be equal and opposite to the contact normal, \mathbf{n}_T . While the normal is directly defined, there remains some choice about the directions of the x - and y -axes. To simplify matters, we have chosen the y -component of \mathbf{x}_S to be zero, so that \mathbf{x}_S will always lie in the world x - z plane. The axes of the SCS in the WCS are:

$$\mathbf{z}_S = \begin{bmatrix} z_{Sx} \\ z_{Sy} \\ z_{Sz} \end{bmatrix} = -\mathbf{n}_W,$$

$$\begin{aligned} \mathbf{x}_S &= \begin{bmatrix} z_{Sz} \\ z_{Sx} \\ 0 \end{bmatrix}, \\ \mathbf{y}_S &= \mathbf{z}_S \times \mathbf{x}_S. \end{aligned} \quad (4.3)$$

A singularity will occur when z_{Sx} and z_{Sz} are both zero. In and near this case, one can change the SCS to require that \mathbf{y}_S lies in the world y - z plane. An alternative local SCS would be to define the x -direction as the last direction of travel, projected onto a plane perpendicular to the z -axis. In this case, a singularity occurs if the finger moves in the direction of the z -axis, although this would not be allowed to happen using the control law for moving the finger parallel to the surface. The SCS axes are normalized to provide a unit vector description of the SCS orientation. Using these vectors, the rotation matrix between the SCS and WCS can be constructed.

$$R_{TW} = \begin{bmatrix} \mathbf{x}_S & \mathbf{y}_S & \mathbf{z}_S \end{bmatrix} \quad (4.4)$$

Further rotation matrices, R_{WS} (World to Surface) and R_{TS} (Tactile to Surface) can be obtained through matrix multiplication.

$$R_{WS} = R_{SW}^T \quad (4.5)$$

$$R_{TS} = R_{TW}R_{WS} \quad (4.6)$$

4.3.3 PD and Normal Force Control

The combination of proportional-derivative (PD) and normal force control allows the finger to slide over the surface and travel over features while following the contours of the object. The normal force control causes the finger to maintain a constant force applied to the object. This force is light enough so that friction does not impede the motion of the fingertip, but strong enough to allow the tactile sensor to measure the contact point (this requires a minimal amount of pressure). Because friction during sliding caused difficulties with the tactile sensor (as discussed in Appendix C), graphite was spread on the surface to lower the coefficient of friction so that higher normal forces could be applied. Tactile sensor data is used to determine the direction of the normal to the surface, so when the surface curves, the force applied to the fingertip changes direction. The PD control causes the finger to move parallel to the

surface with a particular velocity. (Desired positions are selected using the current position and desired velocity.) For this control, the contact point must be known in the WCS.

$$\mathbf{x} = \mathbf{x}_f + R_f \mathbf{n}_W, \quad (4.7)$$

where \mathbf{x} is the contact point, \mathbf{x}_f is the position of the center of the fingertip, R_f is the radius of the fingertip, and \mathbf{n}_W is the contact normal. All the vectors are in the WCS.

The following vector equation expresses the force applied to the motors.

$$\mathbf{f}_{pd} = K_p(\mathbf{x}_{des} - \mathbf{x}) + K_v(\dot{\mathbf{x}}_{des} - \dot{\mathbf{x}}) \quad (4.8)$$

In this equation, \mathbf{f}_{pd} is the force due to the proportional-derivative control, \mathbf{x}_{des} and \mathbf{x} are the desired and actual contact point positions, and $\dot{\mathbf{x}}_{des}$ and $\dot{\mathbf{x}}$ are the desired and actual velocities.

When exploring a surface, the goal is not necessarily to reach certain locations on the surface. When searching for a feature, the goal is to move the finger around on the surface and go in different directions. Thus, rather than specifying a particular position that the contact point should attain, we define a desired velocity and the direction of travel. The desired position is determined from the desired velocity; at each servo loop, the corresponding position can be calculated.

$$\mathbf{x}_{des_k} = \mathbf{x}_{des_{k-1}} + \frac{\dot{\mathbf{x}}_{des}}{\omega}, \quad (4.9)$$

where ω is the servo frequency. When the desired velocity is changed, a linear ramp is used to slowly increase the velocity to the desired value.

K_p and K_v are the Cartesian stiffness and damping matrices. They are diagonal matrices whose entries are the stiffness in each of the Cartesian directions (x, y, and z). The gain matrices are originally developed in surface coordinates so the that direction perpendicular to the surface will have a gain of zero. By defining the matrices this way, we are effectively including a selection matrix to apply the PD control law in particular directions. k_p and k_v are the chosen scalar gains.

$$\begin{aligned}
Kp_S &= \begin{bmatrix} k_p & 0 & 0 \\ 0 & k_p & 0 \\ 0 & 0 & 0 \end{bmatrix} \\
Kv_S &= \begin{bmatrix} k_v & 0 & 0 \\ 0 & k_v & 0 \\ 0 & 0 & 0 \end{bmatrix}
\end{aligned} \tag{4.10}$$

Then, a similarity transform is used to convert the gain matrices into world coordinates.

$$\begin{aligned}
Kp_W &= R_{SW}^T Kp_S R_{SW} \\
Kv_W &= R_{SW}^T Kv_S R_{SW}
\end{aligned} \tag{4.11}$$

The desired normal force magnitude, f_{des} , is also assigned. The desired force vector is then determined from the contact normal:

$$\mathbf{f}_n = f_{des} \mathbf{n}_W. \tag{4.12}$$

The open-loop normal force is added to the PD force to determine the total force at the contact point:

$$\mathbf{f}_{total} = \mathbf{f}_{pd} + \mathbf{f}_n. \tag{4.13}$$

In this equation, \mathbf{f}_{pd} is the PD force, \mathbf{f}_n is the desired normal force, and \mathbf{n} is the direction normal to the surface in the world coordinate system. A control loop could certainly be used to regulate the normal force, however, this was unnecessary with the current experimental setup. Gravity compensation is also applied, as discussed in Appendix C.

With this control law in place, we can now consider the desired directions of motion for the contact point. In local exploration, decisions about this motion are based on sensed feature information.

4.4 Local Exploration

Using the control law described in the previous section, a finger can travel across a surface with a constant velocity and normal force. During this motion, the finger may encounter a feature. There are two main phases of local feature exploration: type detection and tracing. Type detection is a series of motions used to determine whether the feature is a ridge, step, ravine, etc. Tracing is used to get an “outline” of the feature so its overall geometry can be recorded.

4.4.1 Feature Type Detection

Let us assume that a finger has encountered a feature and detection has been performed using the method in Chapter 3, so that curvature features are recorded. It is important to note that the direction of finger travel during a search may not correspond to the first principal direction of a feature, so the finger can actually travel over a feature without detecting it. Thus, it may be desired to perform type detection even if it is uncertain whether the curvature is sharp enough to warrant designation as a feature.

A flow chart can be used to describe a type detection algorithm, where the branching direction depends on previous discoveries. In this example, let us consider three possible feature types, a convex cusp, a step, and a ridge. Because it is possible to define an infinite number of macro features, we will restrict the possible feature types for simplicity. Figure 4.7 shows an example of a feature type identification flow chart using these three feature types.

Rather than using an explicit flow chart for identification, the finger can also move over the surface (perhaps in a raster-scanning fashion) and record all the curvature feature regions detected. These regions can then be pattern-matched against the possible feature types for identification purposes.

4.4.2 Feature Tracing

After the feature type has been identified as a ridge, step, or cusp, the geometry of the feature is further examined by tracing the feature boundary. Tracing is accomplished by moving in the plane perpendicular to the contact normal when the finger is on a negative curvature feature. With only contact location sensing, it is difficult to decide which direction to travel on this plane. However, when the nominal surface is

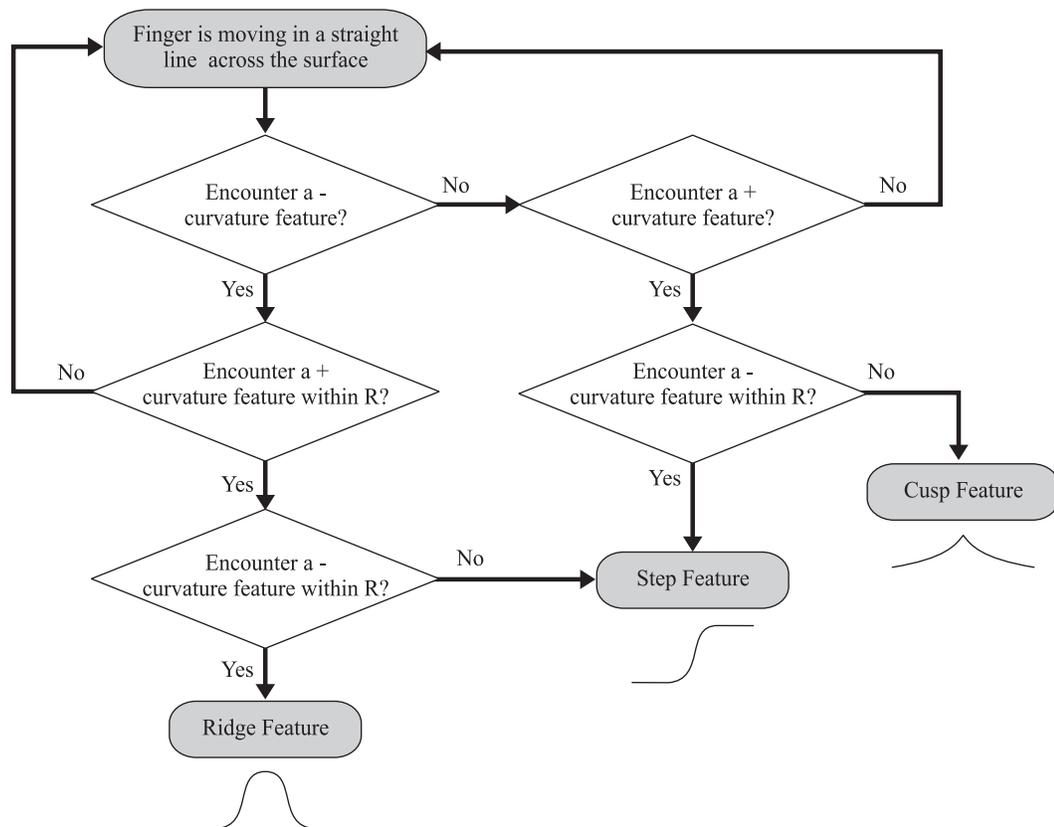


Figure 4.7: Flow chart for feature type identification. The possible features in this example are a cusp, a step, and a ridge.

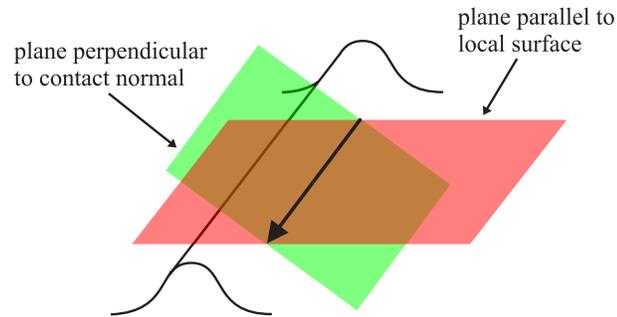


Figure 4.8: The direction of contact point travel during feature tracing is determined by the intersection of a plane parallel to the local surface and a plane orthogonal to the contact normal.

gently curving, one can also include the constraint of staying in the plane of travel before the feature was encountered. The intersection of these two planes creates a path through space that also traces along a negative curvature feature, as shown in Figure 4.8. When tracing a feature, these planes should not be close to parallel, thus, the path of travel will always be defined. The direction along this path can be chosen arbitrarily.

While performing this tracing, it is possible for the finger to lose contact with the feature. At intervals, the finger can re-approach the feature and find the curvature feature regions again. During the tracing, it is important to keep track of locations that represent the boundary of the feature. When the finger moves away from the feature in order to relocate a curvature feature, this data should be removed.

4.4.3 Feature Modeling with Shape Skeletons

Using feature data points obtained by tracing, the Medial Axis Transform (MAT) of the feature can be identified for macro features. The MAT is useful for modeling and object identification purposes, and, as we will see in the Section 4.5, for creating global maps of features and feature regions.

The Medial Axis Transform

The medial axis is defined as the loci of centers of locally maximal balls (in 3D) or disks (in 2D) inside an object. In other words, it is the set of the centers of balls or disks that fit tightly into the object, touching the boundary at at least two contact points. The medial axis is also known as a shape skeleton. Each point on the medial

axis is associated with the radius of a locally maximal ball or disk. These medial axis points, together with their associated radii, define the MAT of an object. In this work, the medial axis is used to define the locations of features and is the most important part of the transform. If the features are to be reconstructed, the entire MAT is needed. In addition, only the 2D MAT of a surface will be considered here. While it is possible to construct a 3D MAT of an object, it is not as useful in detecting and mapping features for the context of haptic surface exploration with robotic fingers.

According to the above definition, medial axis points are equidistant to the object boundary and represent the innermost interior points. In 2D, these medial axis points represent a set of continuous curves. The points where three or more of these curves meet are called branch points. In 3D, medial axis points form a set of continuous 1D or 2D points known as medial curves or medial surfaces. The edges where more than two medial surfaces meet are called seams. A number of definitions related to the terminology of the MAT are provided by Turkiyyah, *et al.*[91]. The medial axis for a simple 2D object is illustrated in Figure 4.9.

An example of a medial axis that may be created for a feature is shown in Figure 4.10, although the maximal disk algorithm is not the way the medial axis is created in practice for simulated noisy data.

It should be noted that while a medial axis can be found for macro features such as ridges, ravines, bumps, and pits, it is not necessary to create a skeleton for features with two or less curvature features. For a cusp or step feature, the finger cannot trace “around” the feature, it can only trace “along” it. The path obtained by tracing along such a feature can be used as a pseudo-skeleton because this path basically represents the shape of the feature.

Creating the Skeleton from Feature Data

There are two major skeletonization methods that can be used to extract the medial axis transform of feature data, taken from medial axis algorithms for 2D image data[10]. The first is region-based, where the input is usually an array of filled image data. In this “thinning” approach, pixels of images are iteratively thinned, or equivalently, redundant pixels are successively deleted until a final skeleton is derived. The second method is based on the boundaries of images. The boundaries are extracted from the image by edge detection and skeletons are generated directly from the boundary data. The boundary technique is in general more efficient than the

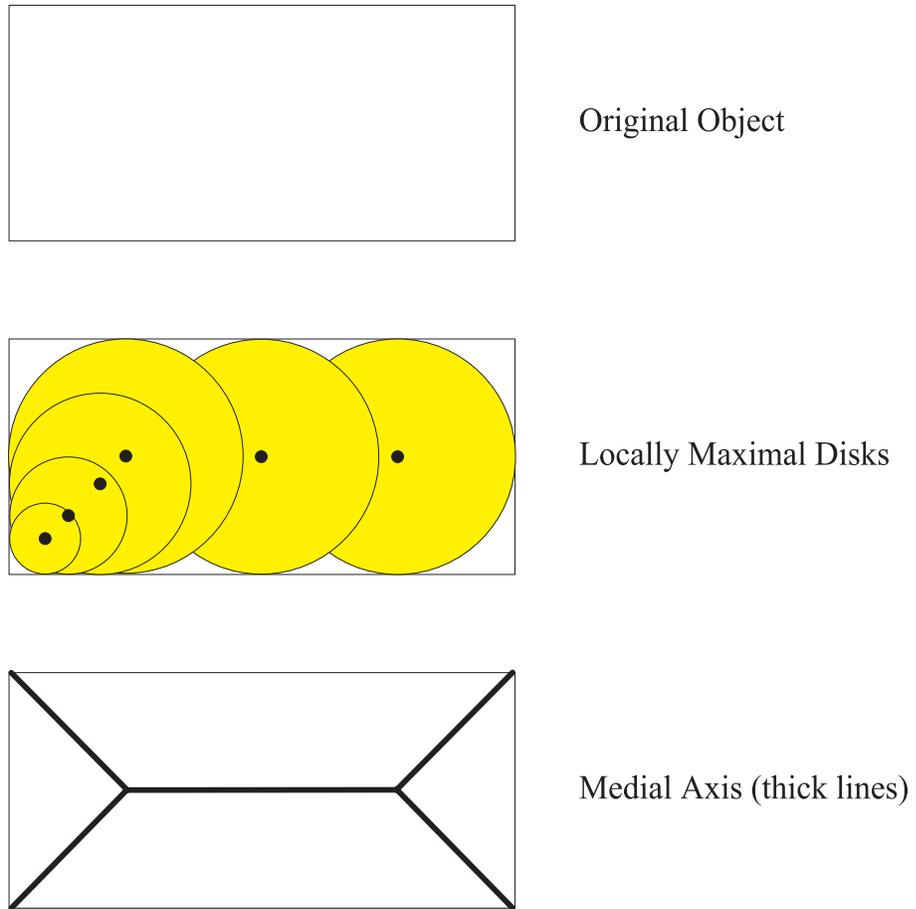


Figure 4.9: Example of the medial axis for a simple 2D shape.

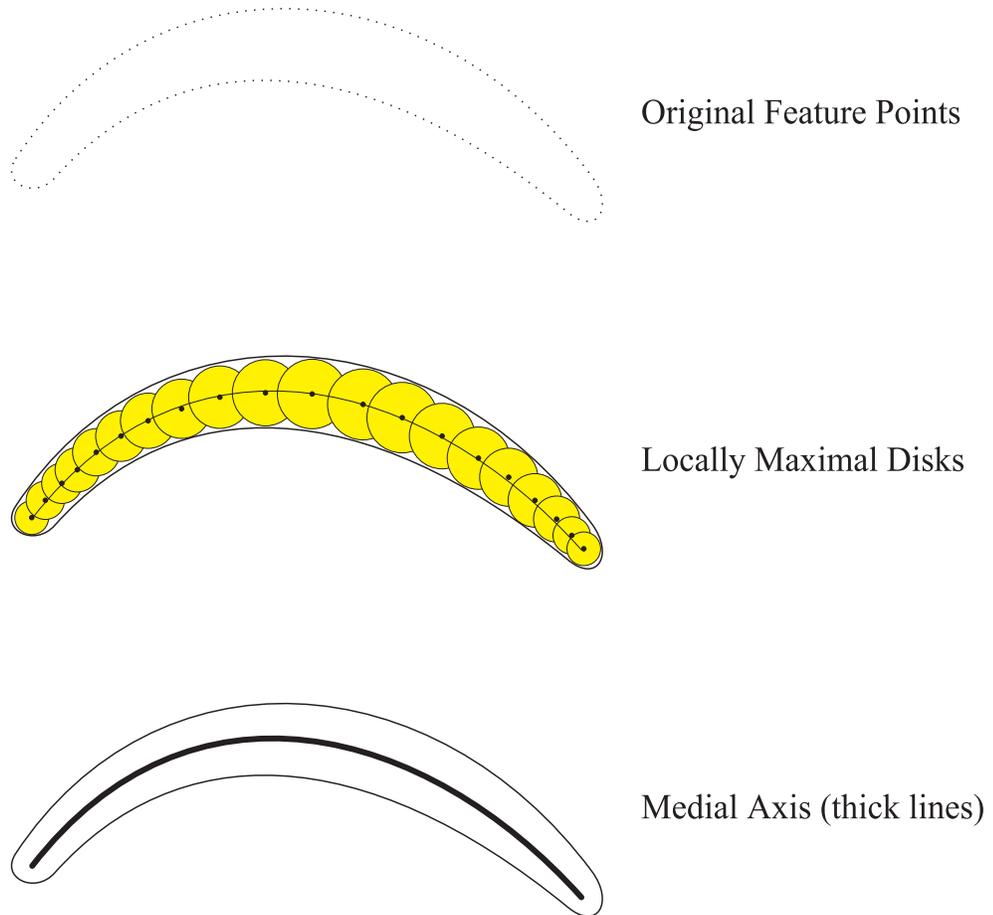


Figure 4.10: Example of the medial axis for a feature.

region-based approach, however, there are issues due to noisy boundary data input.

Because the feature data is obtained by tracing, the data that must be used in the MAT is the feature boundary. This is advantageous because boundary-based approaches are known to be more stable and less problematic than thinning methods[21]. Much of the overhead of the boundary approach is in the detection of a boundary of the image and removal of excessive small branches due to noisy data[39]. Because the portion of local exploration which is identified as a “tracing” phase is well-defined, the data taken during that phase can be separated from the rest of the tactile or position data. This creates a clean group of boundary samples. Thus, the boundary is already calculated, although we must still deal with the issue of noisy data.

Brandt and Algazi[10] and Ogniewicz[64] have used approaches in which boundary points are used to find Voronoi diagrams in order to generate a discrete medial axis. Pruning procedures are necessary to handle noise, errors in sampling, and for hierarchical skeleton representation. We will now step through the basic medial axis extraction algorithm from their work.

The algorithm begins with the creation of a Voronoi diagram from the boundary data points. The Voronoi diagram is the partitioning of a plane with n points into n convex polygons. Each polygon of the Voronoi diagram (also known as a Voronoi polygon) contains exactly one data point and every point in a polygon is closer to this point than to any other data point. Ogniewicz calls the Voronoi diagram of a discrete set of boundary points the *discrete Voronoi medial axis (DVMA)*. Brandt showed formally that the DVMA approaches the continuous Voronoi diagram as the number of boundary samples increases (Figure 4.11).

Once the DVMA has been found, the skeleton must still be extracted by pruning away undesirable Voronoi polygon edges. There are many pruning algorithms that can be used for this purpose. Brandt[10] presents a method where polygon segments are deleted based on the absolute regeneration error and an empirically-determined threshold. For feature detection based on tactile and position data, however, there is a simpler pruning algorithm based on the length of the Voronoi polygon edges. Given the velocity of the robotic fingertip performing the exploration and the rate at which data is sampled, there will be a minimum distance between samples at the boundary of the feature. The results of pruning using this method are shown in Figure 4.11. One can see that where there are data points that are particularly noisy and far away from the main feature shape, unwanted skeleton arcs appear. These

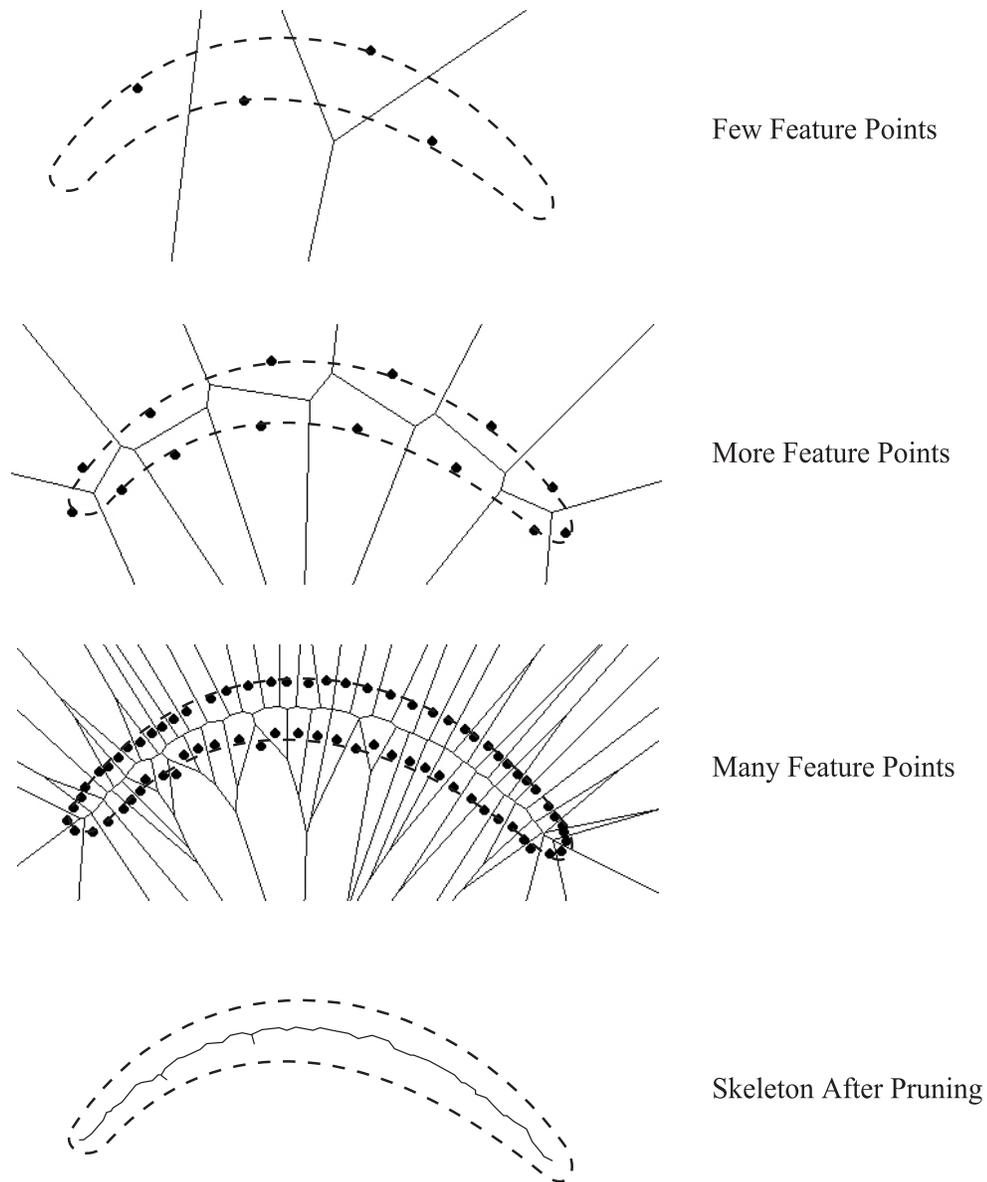


Figure 4.11: As the number of boundary samples increases, the discrete Voronoi medial axis (DVMA) approaches the continuous Voronoi diagram. The DVMA is then pruned to get a more accurate object skeleton. In this example, the boundary samples are noisy and cause unwanted arcs in the final skeleton. These can be removed using a length threshold.

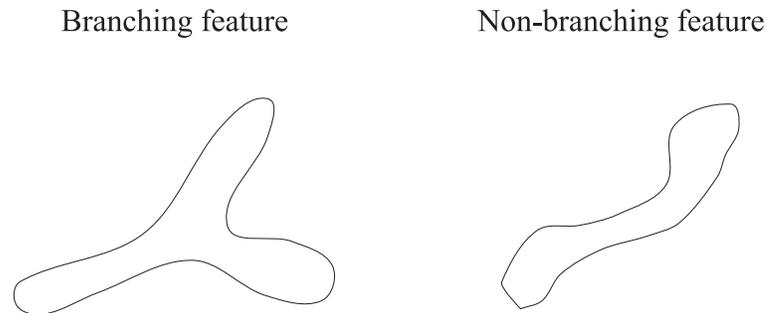


Figure 4.12: Examples of branching and non-branching features. If only non-branching features are considered, unwanted skeleton arcs can be easily pruned away.

could also be pruned away, provided that the features under consideration do not branch (Figure 4.12). Pruning is accomplished by removing the skeleton arcs which are smaller than a threshold length, defined using the noise present in the data. Some researchers[68] use the medial axis itself to determine if an object “branches,” so the choice of threshold will depend on the application.

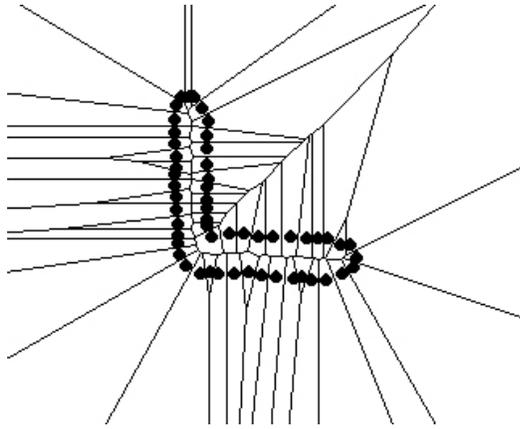
This basic algorithm results in a single medial axis for features that are not significantly curved on the surface. The threshold can be chosen to make the skeleton inside of the boundary samples completely connected for a given sample spacing. However, if the feature curves sharply on the surface, the algorithm will also find part of the skeleton outside the region of the feature, as shown in Figure 4.13. In this case, the connected skeleton with the most edges can be taken as the medial axis. With these feature skeletons created, we can now consider methods for global exploration.

4.5 Global Exploration

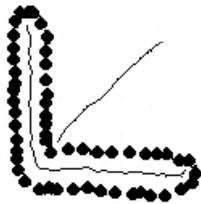
In choosing the method for global exploration, one can consider the efficiency and amount of information obtained from different algorithms. Depending on the goal of the exploration and the nature of the features, various feature search methods will perform differently. In this section, we address some issues for future work on global exploration and the search for features.

4.5.1 Goal Feature Considerations

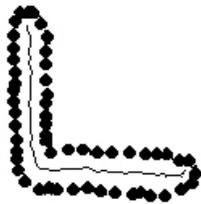
The goal of haptic exploration for finding object features can fall into many categories. For example, we may be looking for a type of feature, a specific feature, any feature



Voronoi diagram of
feature boundary samples



Extracted skeletons using
edge length threshold



Final skeleton determined by
comparison of number of edges

Figure 4.13: When a feature turns sharply, extra skeletons are extracted using a simple edge length threshold. The number of edges of disconnected skeletons can be compared, and the shorter ones discarded.

in a particular region, or all features. There is also the case where the finger desires to move from one point to another, and find any features along the way. For this scenario, a motion based on Lumelsky's bug algorithms (1, 2 or 3)[48] can be used to move towards a goal while tracing any features found along the way.

When searching for a particular type of feature, it is usually advantageous to explore each feature that is detected until that feature is either identified as the goal feature, or there is enough evidence to reject it as a possible match. This can be shown by counter example: rejecting features that have not been identified as the incorrect type would waste any time that has already been spent exploring that feature. A probability parameter could be updated as the feature is explored: when the probability that the feature is of the desired type becomes lower than the probability that the goal feature would be more quickly found by moving to a new feature, that feature could be temporarily rejected.

This would modify the general local exploratory procedures described in the previous section because the exploration needs to be carried out only far enough to make a positive or negative type identification. A similar approach can be taken when looking for a specific feature (of a particular type and shape). First, some features can be rejected automatically if the type does not match. Features with matching type must be explored in more detail by feature tracing. One possible way of matching features to a specific goal feature is pattern matching using the feature shape skeleton. One could also compare the principal curvatures and spacing of curvature features on the macro feature. Because the shape skeleton alone would not capture the "thickness" of a ridge, this curvature feature region size and space could be used as additional information for pattern matching.

When searching for a feature in a specific location, the global search should lead the finger to explore only in the area under consideration. Depending on the specific application, one may want to explore fully all features that have any portion of their boundaries in a particular area. For finding all features, every feature should be fully explored. However, the efficiency of exploration may depend on the spacing of the features on the object in comparison to the size of the robot finger.

4.5.2 Feature Spacing Considerations

Depending on the application and spatial frequency of the features, there is a trade-off between complete exploration of individual features (performing both of the exploratory procedures described above) and continuing exploration to find the next feature. For a general application, where complete knowledge of each feature is important, the most efficient subsequent motion is to continue exploring each feature that has been detected until its boundary has been completely traced. Given the macro feature definition that the curvatures must be within a certain distance to be considered part of the same feature, all curvature features on other macro features will be further away than those on the current macro feature. Thus, more feature information can be obtained by continuing the current exploration, rather than setting out on a search for new macro features. In addition, it may be desired to accomplish complete feature tracing for creating skeletons for global feature mapping.

One way to compare the efficiency of local versus global feature exploration is to consider both large and small spatial distributions of features. For the context of exploring with spherical robotic fingers, a large distribution occurs when no two different features have points closer to each other than one fingertip diameter. This means that more information can always be gained by continuing to explore a current feature until it has been fully traced. The robot finger size can also be considered as a measure of the quantization error in feature exploration. If features are too close together, it is possible that the finger will not be able to differentiate them, especially if only contact location centroid data is available.

In other contexts, it may be desired to use the features themselves as way of determining spatial distribution. By comparing average feature length to average feature spacing (shortest distance between closest points on two different features), another metric for spatial distribution may be obtained. While this method of assigning a distribution parameter may be desirable for object identification and pattern matching, it is not always directly relevant for the context of haptic exploration. From the point of view of a robotic finger, the only goal is to obtain as much relevant information as quickly as possible.

Even for large spatial distributions, one can consider various weighting techniques for deciding whether to continue exploring a current feature or move on to a new search. For example, feature tracing may be computationally more expensive or slower than the motion required to find a new feature. A weighted sum of the expected

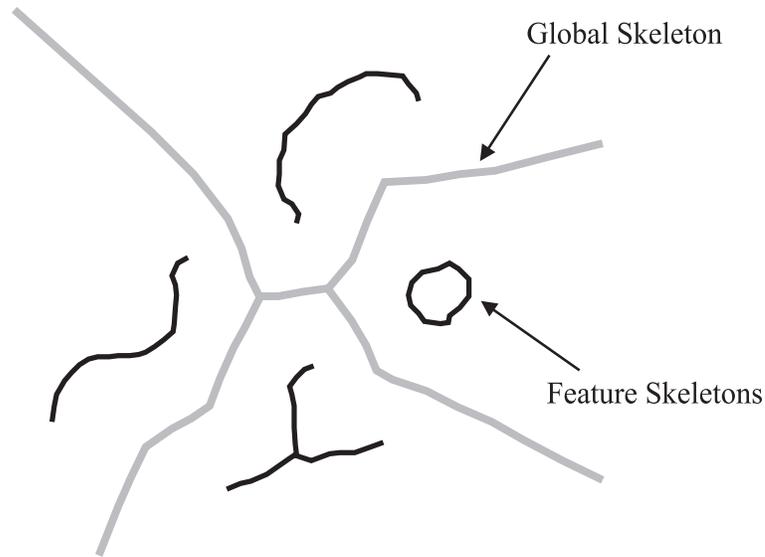


Figure 4.14: The global skeleton creates regions around features.

value of the new information found from continued exploration and the penalty for the cost of doing tracing rather than searching for a new feature can be used to create a decision parameter. A positive value of this parameter indicates that the best move is to continue exploration, and a negative value indicates that a new feature should be found. Such a parameter could also be used for deciding when a feature has been completely traced. When the finger moves near points that have already been traced, the gain from continuing exploration in that area is low or zero.

In general, the robotic hand performing the exploration will not have a priori knowledge of feature distribution. As exploration continues, a distribution parameter can be updated and the exploration strategy modified based on feature spacing.

4.5.3 Partitioning using Feature and Global Skeletons

After a complete local feature exploration has been completed, tracing data may be used to start building a global map of features. As additional features are discovered and explored, a *global skeleton* may be developed to partition the surface of the object into regions as shown in Figure 4.14. This global skeleton is similar to the concept of a *negative medial axis*, which is the medial axis formed from data points consisting of the previously defined (feature) medial axes.

Planning using the global medial axis is left as future work. However, there are several key planning applications that will be described here. One important use of

this partitioning is to evaluate the spacing of the features. The minimum radius of the disks along the medial axis represents the smallest clearance, and therefore the minimum spacing, among objects. Given different feature spacings, various global exploration or manipulation strategies may be invoked. In addition, the global medial axis may be such that it bounds regions on the surface. Larger regions that do not closely bound a feature (to within a fingertip diameter) may need to be explored further to ensure that no features were missed.

If the goal is to avoid features, the trajectory of the global skeleton depicts the safest navigation path that is the farthest from all features. Thus, the relationship between the size of the finger and the radius for each locally maximal disk tells us whether the finger can navigate without hitting any features.

Creating a Global Skeleton

Given the medial axis for each feature, the negative MAT can be calculated by taking the locally maximal balls between the medial axes. This amounts to taking the MAT of the feature medial axes. As shown in Section 4.4.3, however, algorithms based on the Voronoi diagram of sample data are particularly straightforward for boundary data. Thus, we will consider another simple algorithm based on the Voronoi diagram to determine a simple approximation to the negative MAT of the features. The basic steps in this algorithm are shown in Figure 4.15.

The algorithm for obtaining the global medial axis begins with the points on the DVMA's of the features obtained as described in Section 4.4.3. The Voronoi diagram of these medial axes is constructed. Similar to the method for determining the feature medial axes, a number of pruning algorithms can be used to extract the skeleton from the Voronoi diagram. However, the nature of the contact and position data obtained during exploration allows for a new, simple extraction method.

One advantageous property of the local feature exploration algorithm is that features are naturally segmented during the exploration process. When the finger encounters a feature, that feature is explored fully and all the data points observed during this exploration can be attached to a structure for that feature. When a different feature is explored, the new data points are associated with the new feature. Therefore, each boundary sample is associated with a single feature. This property is used in the extraction of the global medial axis from the Voronoi diagram by considering only the Voronoi polygon edges which are shared by two sample points associated

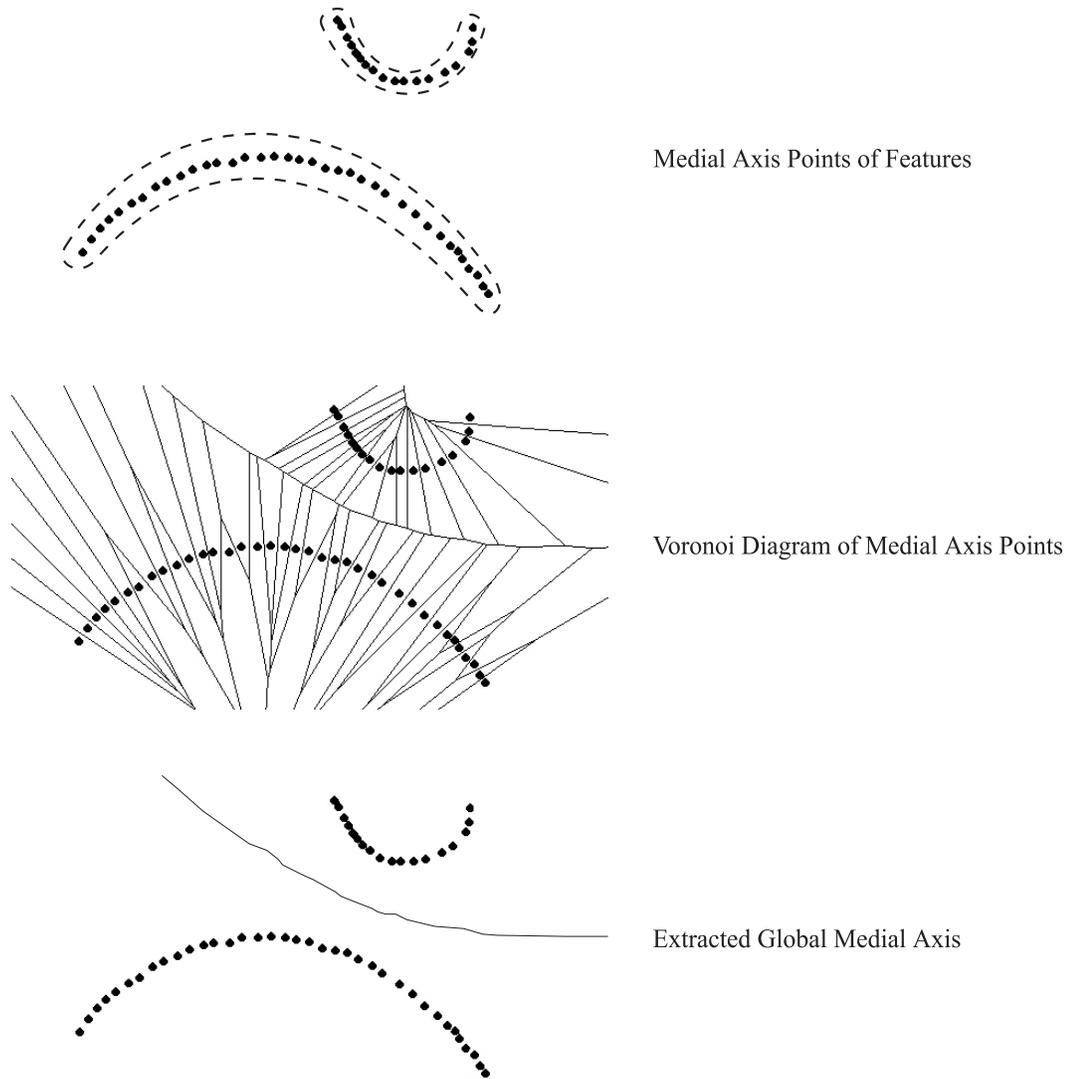


Figure 4.15: The global skeleton is approximated using the Voronoi diagram of the medial axes of the features. It is extracted from the Voronoi diagram by considering only the polygon edges which correspond to medial axis points from two different features.

with different features. By using only these polygon edges, only segments which are equidistant from two different features are considered.

This technique also eliminates some medial axis arcs that might be created using the traditional negative MAT calculation. A true negative MAT calculation would include points that are equidistant from multiple points on a single feature. While this information may be useful for some applications, such as navigation, it would distort the description of feature spacing or the division of a surface into discrete regions surrounding different features. By using Voronoi polygon edges that are only shared between points on different features, we obtain a simpler extraction algorithm and a useful global medial axis.

4.6 Conclusion

This chapter addressed issues in the planning and execution of local and global exploratory procedures for finding features on a surface in 3D. Using the feature definitions presented in Chapter 3 and local exploratory procedures, features can be identified by type and geometry. Feature geometry is recorded by tracing around or along the length of the feature.

The control for moving the finger over the surface at a constant velocity and normal force is presented. This control can be used both during the search for a feature and during feature tracing. Using tactile sensor data, the surface normal is continuously updated, and the motion and normal force trajectories are modified. During feature tracing, the contact location and nominal surface orientation are used to determine the direction of motion when only contact location is available for feedback.

Using the control described above, feature type identification can be performed by using a series of motions over and around the feature. Then, the feature is completely traced in order to build a shape skeleton that can be used for pattern matching or global feature mapping. Using this global feature mapping, various methods for global exploration have been discussed, their use depending on the goal of the exploration and spatial feature distribution.

Chapter 5

Conclusions

This Chapter concludes the thesis by providing a summary of results obtained in the preceding chapters, a description of the major contributions and conclusions drawn from those results, and suggestions of areas for future work.

5.1 Summary of Results

The most significant result produced by this research is a method for detecting and identifying surface features for the context of haptic exploration with robotic fingers. This included the formulation of the definition of a feature, and the development of feature-guided exploration. Two levels of feature are defined: curvature features and macro features. Curvature features are areas where the curvature of the surface exceeds the curvature of a round robotic fingertip exploring the surface. Macro features are patterns of curvature features that can describe ridges, bumps, ravines, and more complicated 3D features. It was shown that the parallel surface traced by a fingertip traveling over a feature can be used to detect and identify features, and that this path has some intrinsic properties that facilitate feature detection.

It is noted that the definition of a feature is highly context-sensitive, depending on both the application and the viewer. While we have tried to leave the application flexible, the viewer is limited to the context of robotic haptic exploration. Robotic fingers can come in many shapes, however, a spherical fingertip is an excellent shape for exploration and mimics the geometry of the human fingertip. By choosing this shape of fingertip, our “viewer” is selected and features can be precisely defined.

Based on the definition for certain macro features, algorithms were developed for

active, feature-guided exploration of these features. As the finger explores a feature, a skeleton representation may be constructed to describe the feature shape and its location on the surface. An algorithm was presented for extracting the skeleton, or medial axis approximation, from the Voronoi diagram of the feature boundary samples.

Global exploration techniques for a single robotic finger were also considered. Depending on the spatial frequency and size of features, different exploration strategies may be invoked. When the features are spatially distributed so that there is more than one fingertip diameter between them, it is most advantageous to continue exploring any feature that is found in order to extract the most feature information in the shortest amount of time. When features are very close, it may be more efficient to “raster-scan” a surface with the finger. Assuming the features are distributed so that they can be explored individually, a global skeleton may be developed using the Voronoi diagram of the skeleton points of the features. A useful global skeleton can be extracted from the Voronoi diagram by considering only polygon edges that are shared by data points from different features. This allows the surface to be divided into discrete regions for mapping, planning, or pattern matching purposes.

This research also recognized the importance of combining manipulation and exploration. Manipulation cannot proceed without exploration to determine unknown object properties, primarily shape and friction. Global exploration cannot be accomplished with most robotic hands unless some manipulation is used to reposition or reorient the object to access different areas of the surface.

5.2 Review of Contributions

The major contributions of this research are summarized here, with rationales based on the above summary of results.

- A procedure for combined manipulation and exploration of an unknown object. Chapter 1 developed a method for accomplishing this with a minimal configuration, two active fingers and a passive palm. The procedure is extendable to multiple fingers and different object shapes, restricted only by workspace and grasp stability limitations. This procedure emphasizes the importance of the link between manipulation and exploration.

- The definition of a feature in the context of haptic exploration with robotic fingers. A feature is defined as a region, or pattern of regions, where at least one of the principal curvatures of a surface is greater than that of the robotic finger. It was shown through simulations and experiments that this definition can be used to identify features. In addition, the parallel surface created by a robotic finger traveling over a feature can be used for feature detection and identification, thus, tactile sensor data is not necessary for this purpose.
- Algorithms for performing feature-guided exploration for the detection and identification of features with robotic fingers in three dimensions. A local exploration procedure was developed and tested that enabled a robotic finger to identify and trace around a ridge feature.
- Algorithms for using shape skeletons for storing feature information and global surface mapping. Voronoi diagrams are used to develop an approximate medial axis for the features. In addition, the global distribution of features can be mapped by a global skeleton, created by taking the Voronoi diagram of the feature skeleton points and removing Voronoi polygon edges that are not shared by skeleton points from different features.

The above results and contributions form a cohesive procedure for haptic exploration of surfaces for the purpose of feature identification. The general procedure and feature definitions can be extended to many different types of robotic hands and objects. Feature-guided exploration allows for the creation of a skeleton model, useful for object identification, pattern matching, and visual or haptic representation.

5.3 Future Work

There are many intriguing avenues for this research. Some possibilities are direct extensions of the work in this thesis, while others move some of the concepts developed here into different domains.

Following directly from this thesis, it will be important to develop planning methods for global exploration. Using the global skeleton that describes the distribution of features, the surface may be divided into regions that logically divide the surface. If there are regions that are larger than expected, or regions which have a feature that is not near a region boundary, it may be desired to continue local exploration in

that region to look for undiscovered features. One may also desire to develop a global exploration algorithm that proves that all features of a minimum size will be found. Efficiency of various feature search techniques can also be investigated.

In performing the experiments for this thesis, it became evident that, while the state of tactile sensing has certainly improved over the last decade, there is still no existing tactile sensor that is ideal for haptic exploration. In particular, haptic exploration involves sliding or dragging a fingertip over a surface. While there exist tactile sensors that are spherical and tactile sensors that can withstand dragging, there is no single sensor that is accurate and has both properties. Thus, new tactile sensors could be designed with exploration in mind. In addition, the application of exploration may also point towards appropriate robotic hand or finger designs.

Moving on to broad extensions of this work, one particularly interesting topic is the development of a reality-based modeling system that can measure other types of object properties. While this thesis focuses on surface shape and features, there are many other properties that can be explored, such as mass/inertia, friction, temperature, texture, and stiffness. Some of these properties, such as texture, are concepts as difficult to define as features. While roughness has been characterized using fractal[65] and stochastic[28, 78] methods, texture is still a nebulous concept and is therefore difficult to detect, identify, and model. Some other properties, such as friction[71] and impact dynamics[23, 66], have been automatically identified, however, they have not been included as part of a single data acquisition system.

In performing this reality-based modeling, one should consider the application of the model. One use of particular interest is haptic display, for virtual environments that display to the human sense of touch. The shapes and features explored in this thesis could be modeled in such a way as to be easily used for haptic rendering. For a full reality-based modeling system, multiple modes (including sight, vision, touch, etc.) can be integrated for data acquisition and display. Even with touch alone, there are a number of possible properties to integrate. For example, features may be combined with a global shape model as local shape perturbations. The feature exploration techniques addressed in this thesis have many possible extensions into the areas of exploration planning and object modeling.

Appendix A

Kinematics of Contact

For the haptic exploration procedure described in Chapter 2, it is essential to understand the kinematics of rolling and sliding manipulation. During manipulation phases, there is rolling contact between the fingers and the object. During exploration phases, there is a combination of rolling and sliding with one finger and static contact with the others. During dexterous manipulation, it is necessary to keep track of the contact points between the object and the finger (Figure A.1). This appendix presents a summary of the kinematics of manipulation with rolling and sliding contacts, based largely on the work of Montana[57, 58], Kerr and Roth[40], and Cai and Roth[12].

In addition, the feature definition presented in Chapter 3 requires a differential geometry surface description. In the development of the kinematics of contact, a surface description is also described that will be useful for the development of a feature definition.

A.1 A Differential Geometry Surface Description

The kinematics of rolling and sliding motion begin with the equations of contact. From differential geometry, we obtain a surface description that can be used to understand the object geometry for a coordinate patch near the contact point. The following definitions are useful in describing contact kinematics[84, 58].

Definition 1 *A coordinate patch S_0 for a surface $S \subset \mathbb{R}^3$ is an open, connected subset of S . There exists an open subset U of \mathbb{R}^2 and an invertible map $f : U \rightarrow$*

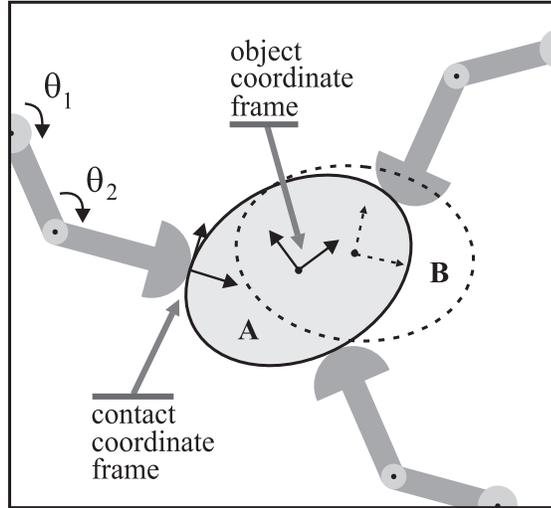


Figure A.1: A typical dexterous manipulation problem: Moving an object from configuration A to configuration B while keeping track of contact point locations.

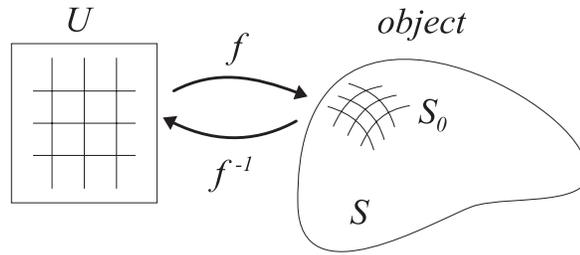


Figure A.2: A coordinate system (f, U) for coordinate patch S_0 of surface S .

$S_0 \subset \mathbb{R}^3$ (Figure A.2) such that the partial derivatives $f_u(\mathbf{u})$ and $f_v(\mathbf{v})$ are linearly independent for all $\mathbf{u} = (u, v) \in U$. The pair (f, U) is a coordinate system for S_0 .

Definition 2 A Gauss map for a manifold S is a continuous map $g : S \rightarrow S^2 \subset \mathbb{R}^3$ such that for every $s \in S$, $g(s)$ is perpendicular to S at s . When S is the surface of a solid object, the Gauss map is an outward normal map if it points outward.

Definition 3 Consider a manifold S with a coordinate patch S_0 that has a coordinate system (f, U) . This coordinate system is orthogonal if $f_u(\mathbf{u}) \cdot f_v(\mathbf{u}) = 0$ for all $\mathbf{u} \in U$. When the coordinate system is orthogonal, the normalized Gauss frame (Figure A.3) is defined at a point $\mathbf{u} \in U$ as the coordinate frame with origin at $f(\mathbf{u})$ and coordinate

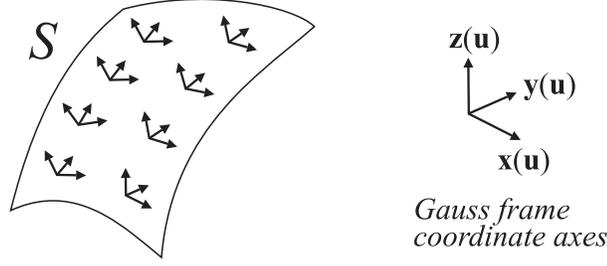


Figure A.3: The Gauss frames for some points on a surface and a view of the Gauss frame coordinate axes.

axes

$$\begin{aligned}
 \mathbf{x}(\mathbf{u}) &= \frac{f_u(\mathbf{u})}{\|f_u(\mathbf{u})\|} \\
 \mathbf{y}(\mathbf{u}) &= \frac{f_v(\mathbf{u})}{\|f_v(\mathbf{u})\|} \\
 \mathbf{z}(\mathbf{u}) &= g(f(\mathbf{u})),
 \end{aligned} \tag{A.1}$$

where $g()$ gives a normalized vector.

Definition 4 Given a manifold S with Gauss map g , coordinate patch S_0 , and an orthogonal coordinate system (f, U) , we can define the curvature form K (or curvature matrix), torsion form T , and metric M . At a point $s \in S_0$, the curvature form K is defined as the 2×2 matrix

$$K = \begin{bmatrix} \mathbf{x}(\mathbf{u})^T \\ \mathbf{y}(\mathbf{u})^T \end{bmatrix} \begin{bmatrix} \frac{\mathbf{z}_u(\mathbf{u})}{\|f_u(\mathbf{u})\|} & \frac{\mathbf{z}_v(\mathbf{u})}{\|f_v(\mathbf{u})\|} \end{bmatrix}. \tag{A.2}$$

where $\mathbf{u} = f^{-1}(s)$. The torsion form T at s is the 1×2 matrix

$$T = \mathbf{y}(\mathbf{u})^T \begin{bmatrix} \frac{\mathbf{x}_u(\mathbf{u})}{\|f_u(\mathbf{u})\|} & \frac{\mathbf{x}_v(\mathbf{u})}{\|f_v(\mathbf{u})\|} \end{bmatrix}. \tag{A.3}$$

The metric M is defined at s as the 2×2 diagonal matrix

$$M = \begin{bmatrix} \|f_u(\mathbf{u})\| & 0 \\ 0 & \|f_v(\mathbf{u})\| \end{bmatrix} \tag{A.4}$$

For example, we will consider the calculation of these surface descriptors for a sphere, which can be defined by the set

$$U = \{(u, v) \mid -\frac{\pi}{2} < u < \frac{\pi}{2}, -\pi < v < \pi\} \quad (\text{A.5})$$

and the map

$$f : U \rightarrow \mathfrak{R}^3, \quad (\text{A.6})$$

$$(u, v) \mapsto (R \cos u \cos v, -R \cos u \sin v, R \sin u). \quad (\text{A.7})$$

The sphere can be viewed as the surface of a solid ball, so the outward normal map is

$$g : S \rightarrow S^2. \quad (\text{A.8})$$

Using the definitions of the coordinate axes of the Gauss frame, we can calculate these vectors as

$$\begin{aligned} \mathbf{x}(\mathbf{u}) &= \begin{bmatrix} -\sin u \cos v \\ \sin u \sin v \\ \cos u \end{bmatrix} \\ \mathbf{y}(\mathbf{u}) &= \begin{bmatrix} \cos u \cos v \\ -\cos u \sin v \\ \sin u \end{bmatrix} \\ \mathbf{z}(\mathbf{u}) &= \begin{bmatrix} -\sin v \\ -\cos v \\ 0 \end{bmatrix}. \end{aligned} \quad (\text{A.9})$$

If one considers the sphere to be the surface of the earth, the x - and y -directions are North and West, and the z -direction is up. The curvature matrix, torsion form, and metric are calculated from the map and the coordinate axes:

$$\begin{aligned} K &= \begin{bmatrix} \frac{1}{R} & 0 \\ 0 & \frac{1}{R} \end{bmatrix} \\ T &= \begin{bmatrix} 0 & \frac{-\tan u}{R} \end{bmatrix} \end{aligned}$$

$$M = \begin{bmatrix} R & 0 \\ 0 & R \cos u \end{bmatrix}. \quad (\text{A.10})$$

A.2 Contact Kinematics

In order to develop the equations of contact, we now consider two rigid bodies that are moving while maintaining a single contact with each other. Object i has a surface S_i (assuming that the surface is a surface patch for simplicity), a series of local coordinate systems (f_i, U_i) and a Gauss map g_i . Each object has a reference frame fixed on the object, C_{ri} , and a reference frame at the contact point C_{li} that is defined relative to the fixed frame. $c_i(t) \in S_i$ is the position of the contact point at time t relative to C_{ri} . In general, c_i will not remain in a single coordinate patch, so we will restrict motions to an interval for which $c_i(t)$ remains in a contact patch that can be described by a single local coordinate system and Gauss map.

Now consider that there are in general five degrees-of-freedom if a rigid body is to remain in single point contact with another rigid body. The total degrees-of-freedom for a rigid body is six, and the contact constraint subtracts one of those degrees-of-freedom. Therefore, the contact equations must account for five degrees-of-freedom.

The contact point coordinates relative to the coordinate system (f_i, U_i) can be calculated from c_i as

$$\mathbf{u}_i(t) = f_i^{-1}(c_i(t)) \in U_i. \quad (\text{A.11})$$

This equation constrains four degrees-of-freedom of the relative motion of the two objects because $\mathbf{u}_i(t)$ has two coordinates, and $i = 1, 2$ for two objects. As Figure A.4 illustrates, $\mathbf{u}_1(t)$ and $\mathbf{u}_2(t)$ change during rolling and sliding motion from \mathbf{u}_1 and \mathbf{u}_2 to \mathbf{u}'_1 and \mathbf{u}'_2 , respectively. The fifth degree-of-freedom is constrained by the angle of contact, ψ (spin), between the contact coordinate frames on the two objects.

The motion of an object relative to another can be described by the relative velocities of the contact frames on the two objects. The translational velocities are described by v_x , v_y , and v_z . Similarly, the components of rotational velocity are ω_x , ω_y , and ω_z . These velocities are measured in the contact coordinate system of one of the objects, and they will be equal and opposite to the velocities in the contact coordinate system of the other object. Once again, we have six degrees-of-freedom, but in order to maintain contact, there is a constraint that $v_z = 0$. Montana[57] developed a set of equations that can be used to transform relative object velocities

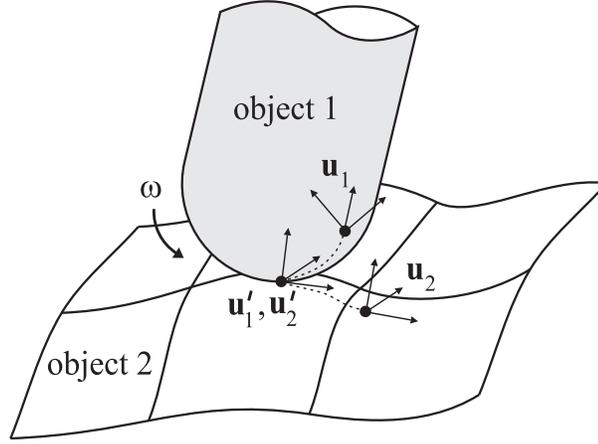


Figure A.4: The motion of contact frames for rolling and sliding contact. The contact frames were at \mathbf{u}_1 and \mathbf{u}_2 for objects 1 and 2 before motion occurred. The new contact frames are \mathbf{u}'_1 and \mathbf{u}'_2 .

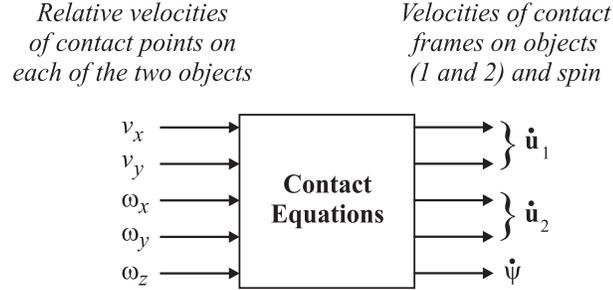


Figure A.5: Contact variables. (Adapted from Montana[57].)

to contact point motion (Figure A.5), where the subscripts on the K , T , and M matrices refer to the object number.

$$\begin{aligned}
 \dot{\mathbf{u}}_1 &= M_1^{-1}(K_1 + \tilde{K}_2)^{-1} \left(\begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} - \tilde{K}_2 \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) \\
 \dot{\mathbf{u}}_2 &= M_2^{-1}R_\psi(K_1 + \tilde{K}_2)^{-1} \left(\begin{bmatrix} -\omega_y \\ \omega_x \end{bmatrix} + K_1 \begin{bmatrix} v_x \\ v_y \end{bmatrix} \right) \\
 \dot{\psi} &= \omega_z + T_1 M_1 \dot{\mathbf{u}}_1 + T_2 M_2 \dot{\mathbf{u}}_2 \\
 0 &= v_z,
 \end{aligned} \tag{A.12}$$

where

$$\begin{aligned} R_\psi &= \begin{bmatrix} \cos \psi & -\sin \psi \\ -\sin \psi & -\cos \psi \end{bmatrix} \\ \tilde{K}_2 &= R_\psi K_2 R_\psi \end{aligned} \tag{A.13}$$

The contact equations are *non-holonomic*, meaning that that they are non-integrable: the constraint is on velocity rather than position. Such a non-holonomic system is problematic because it is difficult to plan manipulations without being able to integrate to find positions. Thus, much of the experimentally successful work in multi-fingered manipulation has been done for planar rolling and sliding, where the contact equations become holonomic. This reduction to a holonomic system is described in Chapter 2.

Appendix B

Smoothing Tactile Data Using Noise Type

The position data obtained from tactile sensors in robotic applications are typically noisy due to both the motion of the robot supporting the sensor and the construction of the sensors themselves. In order to extract the location of surface features from such data (requiring a second derivative operation), a smoothing filter must be applied. This Appendix presents a comparison of several smoothing techniques. First, a χ^2 (Chi-squared) test was performed to confirm that the noise in the data was Gaussian. Next, several different filters were considered: average, median, dual point, Gaussian, and Wiener. Finally, a feature detection algorithm was implemented for rating the success of the appropriate smoothing filters with different filter parameters. Results of the feature detection scheme on the filtered data are presented for comparison of the effectiveness of the various filters.

One can observe the noise in data from Figures B.1 and B.2. Data was obtained from sliding the finger over a small bump (created by stretching a 0.65mm diameter wire over an angled planar surface). From the zoomed out view (Figure B.1), it is obvious where the bump is located. However, the noise present in the zoomed in view (Figure B.2) makes automatic feature detection difficult.

B.1 Noise Analysis

In order to determine which filters might be appropriate for smoothing the tactile sensor data, a χ^2 test was implemented. The χ^2 test is a goodness-of-fit measure

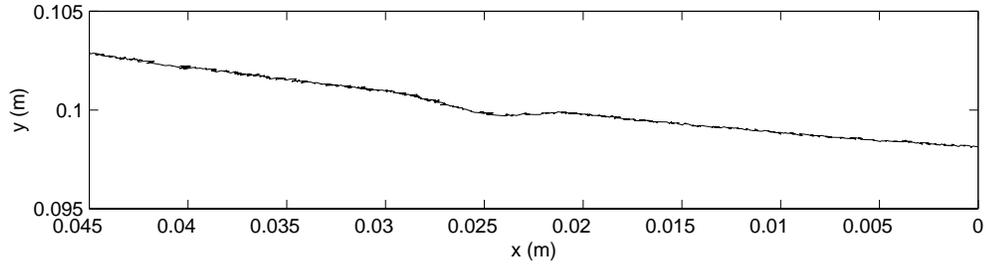


Figure B.1: Tactile data.

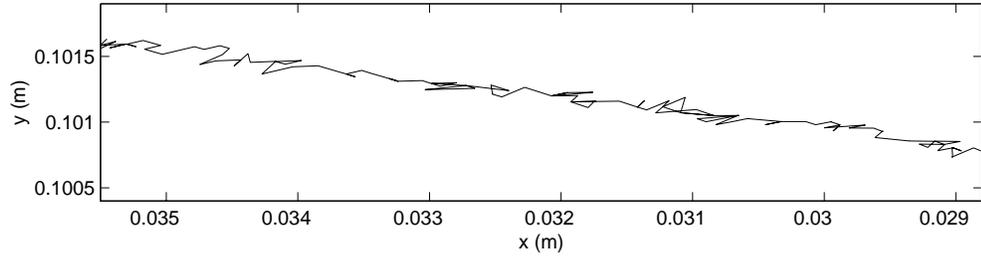


Figure B.2: Magnified view of tactile data.

that can be used to test the probability that a sample of data is or is not normally distributed[7, 55].

The first step in performing the χ^2 test is to extract the noise from the data. The noise may be observed as x position vs. y position or a combination of x vs. t (time) and y vs. t . For the tactile data in this work, filtering the position variables separately parameterized by time is more appropriate, given the way filters must be applied. Many of the filters, such as the Gaussian filter, require that a window is convolved along the signal. It is difficult to convolve at equidistant points along a 2D path such as the (x, y) tactile data. However, because the data was obtained at constant time intervals, the convolution (and therefore smoothing) can be done separately along the x and y directions with respect to t . In addition, the noise arises over time, not as a function of the relationship between x and y . Thus, all the noise analysis was performed twice, on both the x and the y data.

Beginning with a portion of the data with no features (which should be a straight line), a least-squares line fitting algorithm was used to fit the data. The noise was determined from the line coefficients (a , b and c) returned by the algorithm. The equation of the line is

$$c = au + bv, \quad (\text{B.1})$$

where u and v are the independent variables. The noise was calculated from the

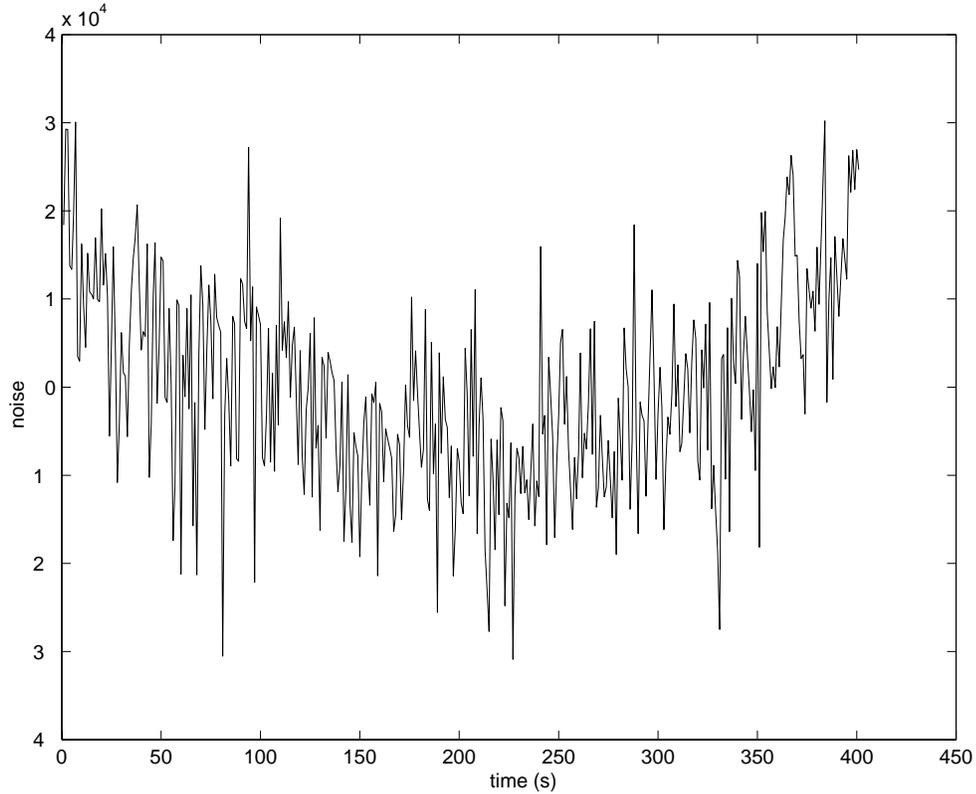


Figure B.3: Skew in noise resulting from a linear fit to the tactile data.

coefficients using the actual (u, v) data points in the equation

$$noise = au + bv - c. \quad (\text{B.2})$$

However, a linear fit was not appropriate for this data, as evidenced by the obvious skew in the noise in Figure B.3.

The fact that a linear fit was not appropriate can be explained by errors in the robot and tactile sensor calibration. Although the contact point moved in a straight line (because the surface was flat), the sensed path is slightly curved. This problem was solved by applying a second order polynomial fit. Using the coefficients returned from the fit, predicted values versus time were obtained and subtracted them from the original data to obtain the noise. The noise was no longer skewed, as is shown in Figure B.4.

Using the noise data, the mean and standard deviation (σ) values were obtained. Then the noise was divided into bins to create a histogram. A uniform bin size was used, estimated from the time scale and minimum and maximum noise values. After

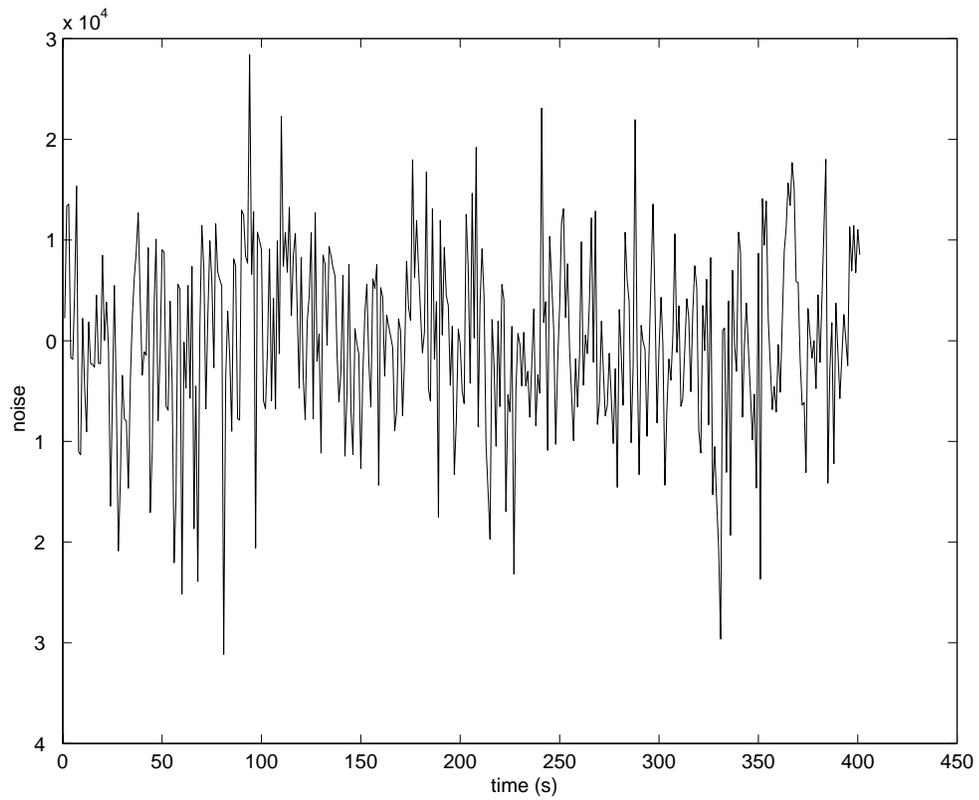


Figure B.4: Accurate noise resulting from a second order polynomial fit to the tactile data.

variable	mean	σ	χ^2 value	normal?
x	$1.1282e^{-17}$	$8.9044e^{-5}$	11.9320	95% sure
y	$-3.2531e^{-18}$	$1.0209e^{-4}$	10.6652	95% sure

Table B.1: Noise analysis parameters for tactile data.

calculating the observed frequency in each bin, the z -value was determined by

$$z = \frac{(x - \bar{x})}{\sigma}. \quad (\text{B.3})$$

Next, the distribution was calculated from

$$F = \int_{-\infty}^z \frac{e^{-\frac{z^2}{2}}}{\sqrt{2\pi}} dz. \quad (\text{B.4})$$

The expected frequency in each bin was then calculated and the χ^2 value summed. The results of the χ^2 tests were 11.9 for x and 10.7 for y . The 95% confidence value for a normal distribution is 11.07. Since both χ^2 values are near this, we can say that the noise is approximately Gaussian, or normally distributed. Table B.1 lists the values of various noise analysis parameters for the x and y variables.

The results of this noise analysis show that a smoothing technique aimed at normally distributed noise should work well on this data. The Gaussian and Wiener filters should be particularly effective, the average filter less so, and the median filter and dual-point smoothing techniques not very effective.

B.2 Curvature Calculation and Feature Detection Algorithm

To provide an objective comparison of the effectiveness of the various smoothing algorithms, a feature detection algorithm was implemented based on the algorithms developed in Chapter 3. The output of the algorithm is the number of features detected and their locations.

The calculation of curvature is extremely sensitive to noise. The computation was approached in several different ways. One (unsuccessful) method was to compute the circle passing through three points, and take the inverse of the circle's radius of curvature. This led to an extremely noisy curvature computation. A much better

way is to use the time derivatives of the x and y coordinates[80].

$$k = \frac{x'y'' - y'x''}{(x'^2 + y'^2)^{1.5}} \quad (\text{B.5})$$

Recall that the purpose of this curvature calculation is to compare smoothing techniques on the original data, so smoothing in the feature detection stage was avoided as much as possible. However, some curvature smoothing could not be avoided in order obtain reasonable curvature plots, as is shown in the feature detection steps below:

1. The inputs are the filtered x and y data.
2. Remove some of the points (every n points), because there is more data than necessary.
3. Calculate the first and second derivatives using surrounding points
 $x' = x(i + 1) - x(i - 1)$ and $x'' = x'(i + 1) - x'(i - 1)$
4. Determine a modified curvature: $k = x'y'' + y'x''$. The denominator of the regular curvature equation is eliminated because it causes a large amount of the noise and does not provide necessary information.
5. Smooth the curvature a small amount with a Gaussian filter.
6. Curvature thresholds are set for feature detection.
7. Positive and negative feature regions are detected, and their ranges and maximum curvature points are recorded.

The feature detection scheme was tested on two sets of simulated data with Gaussian filtering, a square bump (Figure B.5). The plot of the curvature is also shown, with the feature regions marked.

B.3 Smoothing Techniques

Several different soothing techniques were studied and compared as methods for filtering tactile data: the average filter, median filter, dual point filter, Gaussian filter, and Wiener filter.

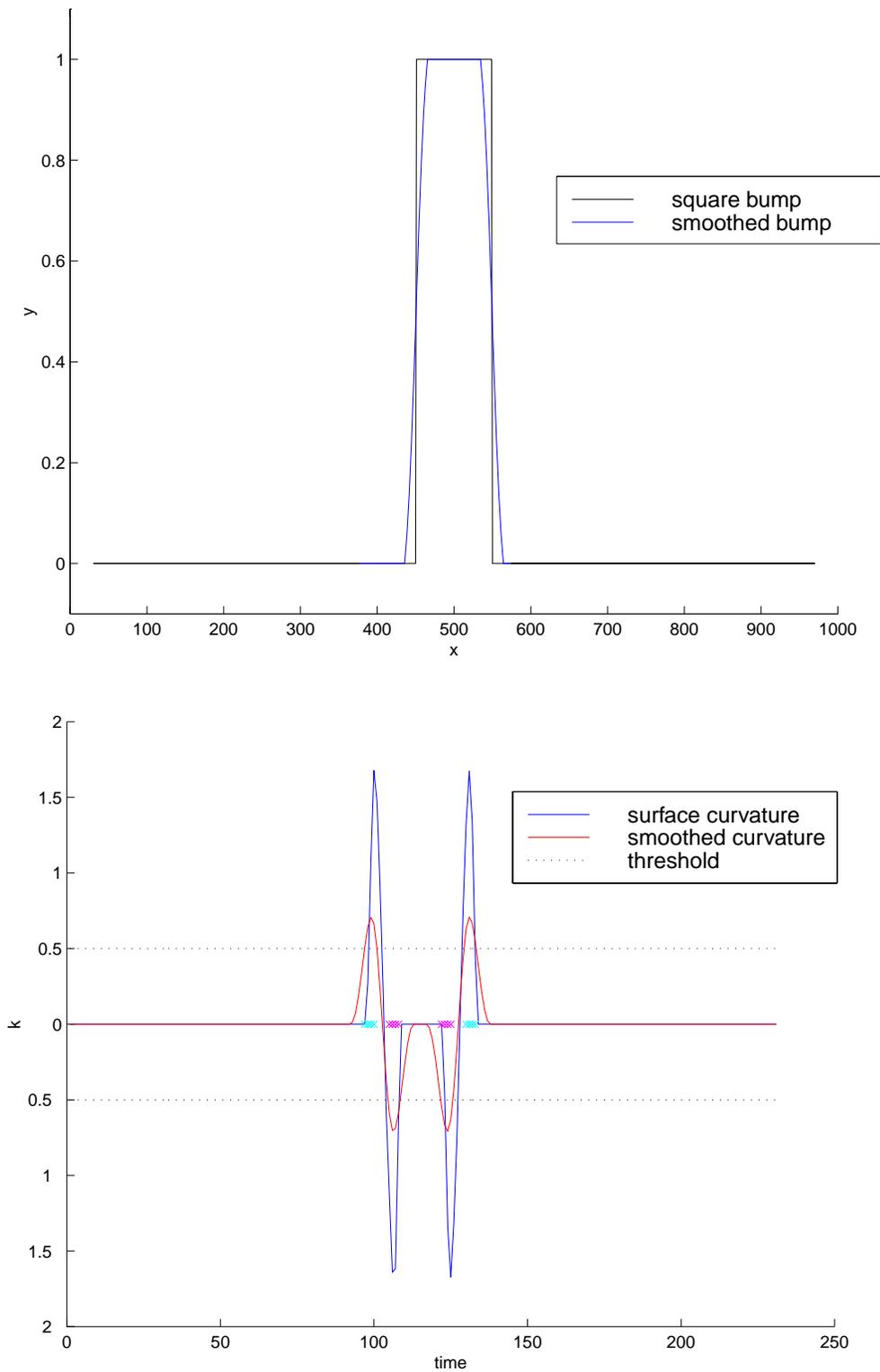


Figure B.5: Top: A square bump feature, before and after Gaussian filtering. Bottom: Feature detection for the square bump feature.

Average Filter

An averaging algorithm takes a 1D vector and a window size as inputs. It calculates the average of the points within the window, and uses the average value as the value of the new point. The window size can be adjusted to change the result to get a smoother curve. However, if the window size is too big, features in the tactile data may be lost, so the window size needs to be adjusted. For example, if the window size used is 3, this algorithm calculates the average between consecutive points a_0 , a_1 and a_2 , and the result is a'_1 .

$$a'_1 = \frac{a_0 + a_1 + a_2}{3} \quad (\text{B.6})$$

This algorithm is simple, fast, and is easy to implement. However, it smooths sharp corners more than other methods because of the equal weighting given to all the points within the window.

Median Filter

The median filter is a non-linear filter useful for removing spurious random noise[90]. This filter is most applicable to signals where the noise can be characterized as “salt-and-pepper” or impulsive noise. For example, suppose the position data is constant over time. If some large spurious noise is added to random position values, this will almost, if not completely, be removed by a median filter with a large enough window.

As shown from the noise analysis in the Section B.1, the noise in the tactile data is not multiplicative, it is additive Gaussian noise. Thus, the median filter does not perform as well as the Gaussian and Wiener filters, which are designed to filter additive noise.

Dual Point Filter

The dual point algorithm is a variant of the mid-point algorithm, and is studied in the environment of smoothing polygons. Dual point smoothing takes in the coordinates of neighboring points in space and replaces them with the $\frac{1}{4}$ and $\frac{3}{4}$ points on the line formed by the two original points. Advantages of this algorithm include that it is fast, simple, and the complexity is $O(kn)$ for k iterations. The disadvantages are that it increases the number of vertices and the space complexity is $O(2kn)$ for k iterations.

The dual point has poor performance on tactile data because it is meant for polygon smoothing, where it smooths out corners. In the intended application of

the dual point algorithm, the raw data will have distinct corners and the x and y coordinates of the points will both increase monotonically with respect to time. In tactile data, neither of these two are necessarily true. Thus, dual point filtering is not an appropriate method for smoothing tactile data.

Gaussian Filter

A Gaussian filter, essentially an averaging filter with weights, is characterized by σ , a standard deviation, and a filter window size. The Gaussian filter weights the current data point the most and assigns decreasing weights to the points farther away from the current point. The rate of this decrease in weights is defined by σ , creating a low-pass filter.

The Gaussian filter is a good match to data characterized by additive noise. However, the impulsive noise is only diffused, not removed. Additionally, the frequencies common to the signal and noise are lost[90]. This means that, since noise is normally high frequency (as it also is in tactile data), the high frequencies of the data are lost. Thus, sharp transitions are blurred. (However, due to the tapered shape of the Gaussian, sharp transitions are not normally smoothed as drastically as when using an average filter.) Thus, it is possible that the Gauss filter is not the optimal filter because the sharp transitions of the data near a feature can be smoothed beyond recognition for the feature detection algorithm.

Wiener Filter

A Wiener filter is characterized by the signal and the signal noise. As with the Gaussian filter, it essentially performs a low-pass filter—with the taper of the filter being defined by the noise and the signal. A Wiener filter is suited well for additive zero-mean noise. Since the data noise (see Section B.1) is characterized by additive zero-mean Gaussian noise[69], the Wiener filter was expected to perform well.

The Wiener filter should be superior to the Gaussian filter since, in the frequency spectrum, the Wiener filter immediately begins to taper off at the frequency where the noise over powers the signal. Additionally, the rate at which it tapers off is proportional to the power of the signal. (In contrast, the Gaussian filter tapers at a random frequency (as defined by σ), and the rate of taper is not appropriately matched to the signal.)

B.4 Filter Experimental Results

The feature detection scheme described in Section B.2 was applied to the appropriate smoothing techniques with different input parameters (window size, σ , etc.). Because the noise in the tactile data was identified as normal, or Gaussian, the Gaussian and Wiener filters were determined to be most appropriate.

The filters were all applied to the same set of data, which included a bump so that there were three curvature features: two negative and one positive. The actual locations of the features, with position parameterized by time, were 90 (negative), 105 (positive), and 112 (negative). The different smoothing techniques were rated by adding the number of false positive and false negative curvature feature identifications occurring from using the test filter and the curvature calculation/feature detection scheme.

The filter which allowed most accurate feature identification were a Gaussian filter with a window of 25-40 and a σ of 5-10 and a Wiener filter with a window of 35-40. An example of the successful feature detection is shown in Figure B.6 for the case with Gaussian filtering, a window of size 25 and $\sigma = 8$. This plot illustrates a borderline case, where one of the features is almost not detected.

B.5 Summary

In this appendix, algorithms were described for smoothing tactile data. A noise analysis was performed to verify that the noise was normally distributed. Several different smoothing algorithms were implemented and tested. A feature detection scheme was designed to facilitate an objective comparison between smoothing techniques. It was found that the Wiener and Gaussian filters were most appropriate given the noise in the tactile data, and performed best when used with particular parameter values for window size and σ .

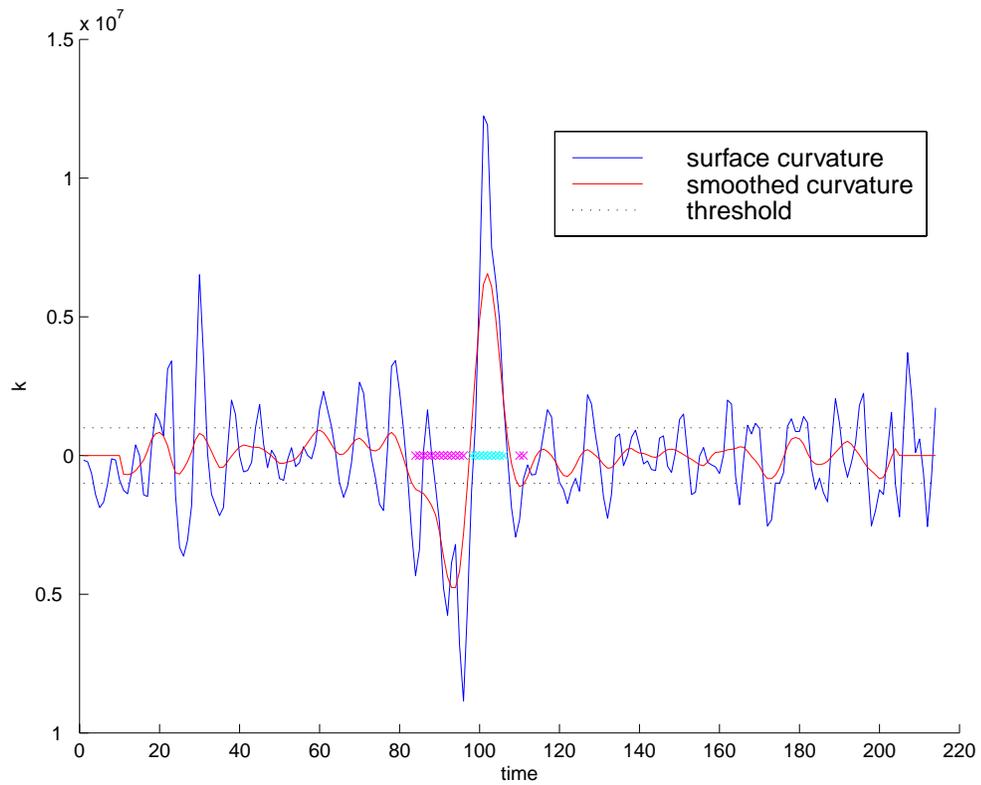


Figure B.6: Feature detection with real data smoothed with a Gaussian filter.

Appendix C

A 3DOF Robotic Finger with Tactile Sensing

This appendix describes the testbed hardware used for tactile sensing in Chapters 3 and 4, and the three-degree-of-freedom robotic finger used for active exploration in Chapter 4.

C.1 The Optical Waveguide Tactile Sensor

The Optical Waveguide Tactile Sensor (OWTS) was developed by Maekawa, *et al.*[50, 51, 52] at the Mechanical Engineering Laboratory in Tsukuba, Japan. Several generations of the sensor were developed, continually improving accuracy and miniaturization with each newly developed prototype. Based on the work by Maekawa, *et al.*, this section describes the function of the latest sensor and how it is used to calculate contact information for haptic exploration.

C.1.1 Sensor Function

The OWTS is an optical sensor that can detect the point where an object contacts its hemispherical surface. One of the major elements of this sensor is the hemispherical optical waveguide made of a glass shell. When light is injected by an LED at the edge of this shell, total internal reflection is restricted to the inner surfaces of the glass. Reflection conditions are modified when an object contacts the surface of the glass, resulting in light reflected toward the interior of the sensor. A silicone rubber cover over the glass shell protects the surface, improves elasticity for gripping, and

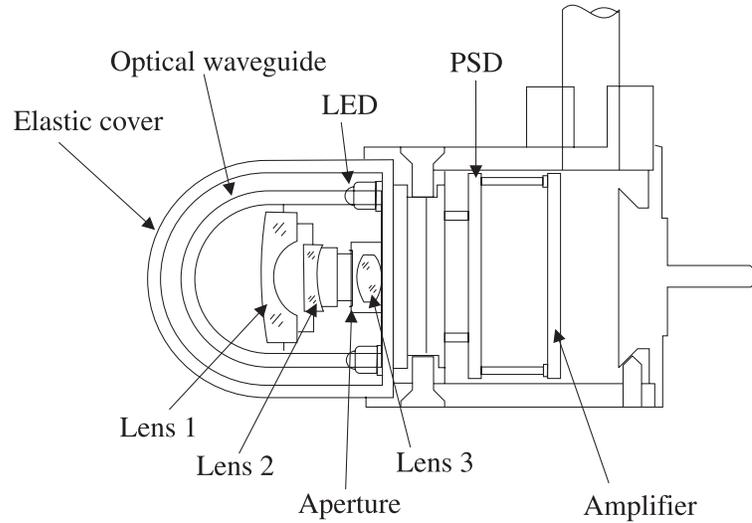


Figure C.1: Interior structure of the Optical Waveguide Tactile Sensor. (Adapted from [50].)



Figure C.2: Exterior of the Optical Waveguide Tactile Sensor, with and without the silicone rubber cover.

provides a uniform reflection that does not depend on the type or color of the object in contact. Inside the shell, light travels through three lenses and an aperture to the sensing mechanism, a position sensitive detector (PSD). The PSD is an analog sensor that measures the total incident light and the centroid of incident light in x - y Cartesian coordinates. Earlier versions of the tactile sensor used a CCD camera, however, this resulted in a larger fingertip and longer image processing time. Figures C.1 and C.2 show the inside and outside of the latest version of the sensor.

C.1.2 Calculating Contact Point

The optical input on the surface of the PSD is determined using the detection coordinate system shown in Figure C.3. The outputs of the PSD are voltages v_1 , v_2 , v_3 , and v_4 . These outputs are used to find the location of the contact in the detection

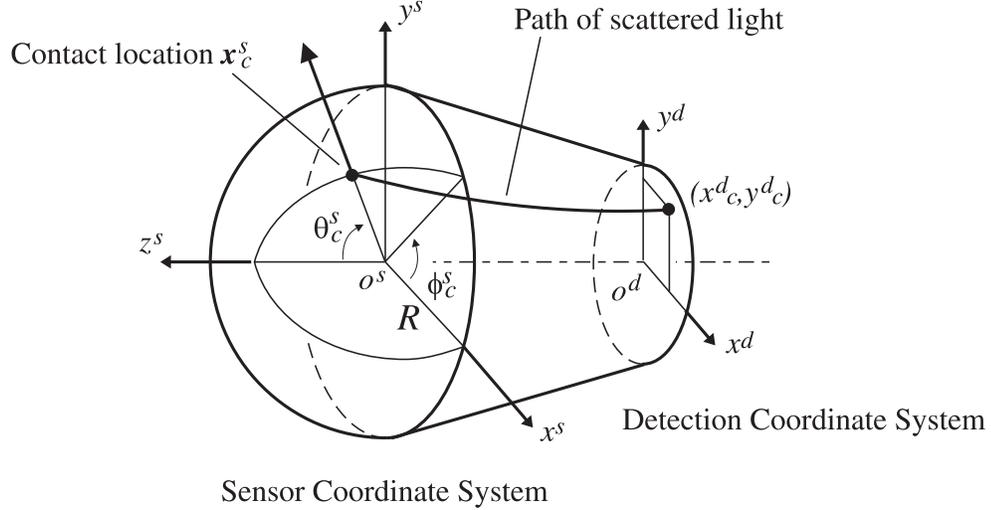


Figure C.3: The sensor and detection coordinate systems, adapted from [50].

coordinate system, (x_c^d, y_c^d) . Calibration factors x_o^d and y_o^d are used to compensate for the deviation between the location on the PSD surface that corresponds to the pole of the hemisphere and the electrical center of the PSD. L is the length of the side of the square PSD.

$$\begin{aligned} x_c^d &= \left(\frac{v_1 - v_2 + v_3 - v_4}{v_1 + v_2 + v_3 + v_4} - x_o^d \right) \frac{L}{2} \\ y_c^d &= \left(\frac{-v_1 + v_2 + v_3 - v_4}{v_1 + v_2 + v_3 + v_4} - y_o^d \right) \frac{L}{2} \end{aligned} \quad (\text{C.1})$$

The contact point on the spherical sensor surface can then be calculated from the contact point in the detection coordinate system in spherical coordinates.

$$\begin{aligned} \phi_c^s &= \tan^{-1} \left(\frac{y_c^d}{x_c^d} \right) \\ \theta_c^s &= K \sqrt{x_c^{d2} + y_c^{d2}}, \end{aligned} \quad (\text{C.2})$$

where K is a scale factor providing the relationship between the two coordinate systems.

For the purposes of haptic exploration, the contact point location on the surface of the sensor and the normal vector can be found using the angles defined above and the radius of the fingertip, R .

$$\begin{aligned} \mathbf{x}_c^s &= \begin{bmatrix} R \cos \phi_c^s \sin \theta_c^s \\ R \sin \phi_c^s \sin \theta_c^s \\ R \cos \theta_c^s \end{bmatrix} \\ \mathbf{n}_c^s &= \begin{bmatrix} \cos \phi_c^s \sin \theta_c^s \\ \sin \phi_c^s \sin \theta_c^s \\ \cos \theta_c^s \end{bmatrix} \end{aligned} \quad (\text{C.3})$$

Because the relationship between the sensor and detection coordinate systems is not linear with the latest version of the sensor, calibration is needed to compute \mathbf{n}_c^s from the detected point (x_c^d, y_c^d) . The maximum sampling rate is approximately 5kHz, well over the 1kHz servo loop used in the haptic exploration experiments in this thesis. The resolution of the sensor is limited by the 12-bit A/D converter used in reading the voltage signals. However, due to noise, the actual resolution is approximately $\pm 0.5\text{mm}$.

C.1.3 Advantages and Disadvantages

One of the primary advantages of the OWTS is its spherical shape which gives it 3D contact sensing capability. This makes the OWTS ideal for active explorations that require three-dimensional contact point information. Another major advantage of this device is size. With a diameter of approximately 16mm, the sensor is on the order of the size of a human fingertip. Due to the analog PSD used for sensing contact location by reflected light, the sensor can also read and transmit contact data very quickly, at over 1kHz. There is also no hysteresis, making fast readings for different contact locations very accurate. A final advantage is its low weight, 27g. This makes it ideal for mounting on the end of a small robot finger.

While the OWTS is excellent in comparison to many other tactile sensors at the time of this writing, it also has a number of areas that need improvement. First, the sensor is not linear; it must be calibrated. Calibration is not an easy task because spherical contact locations must be accurately determined. The most important problem with this sensor is the pulling of the plastic sheath when undergoing sliding motion over a surface. Due to friction between the fingertip and surface, the rubber gets pulled over a patch of the glass that is larger than, and not centered at, the actual

contact location. This can be considered a form of hysteresis, and was corrected for in the experiments by releasing the normal force holding the finger against the object at intervals to get a correct contact location reading. For some experiments, graphite was spread on the fingertip and object to lower the coefficient of friction between the two surfaces. This helped to prevent the pulling of the rubber cover that resulted in false contact location readings.

C.2 The 3GM as a Robotic Finger

For the haptic exploration experiments, the OWTS was mounted on a 3DOF robotic finger, the 3GM from Immersion Corporation[27]. This finger was originally developed as a haptic interface for endoscopic sinus surgery simulation[87], and has also been used in experiments for vibration feedback in virtual environments[67].

C.2.1 Hardware

The 3GM has three degrees of freedom (DOF), each one actuated with a DC motor and measured with an optical encoder. The well-conditioned workspace of the device is trapezoidal, approximately 4in high, 4in long, 4in wide in front, and 10in wide in back. This workspace, which avoids regions at or near singularities, gives a volume of 120 cubic inches. The position resolution is approximately 0.05mm, and the maximum force output is 9N (2lbs) at the center of the workspace.

Figure C.4 shows a side view of the device. The advantages of the 3GM as a haptic interface, including low inertia, low friction, backdriveability, and accurate force output, also make it ideal for use as a robot finger.

C.2.2 Device Control

Forward Kinematics

The device is composed of a planar five bar mechanism mounted on an extra rotational axis to change its plane of motion. Figure C.5 shows the link lengths and angle variables used to describe the 3GM kinematics.

The forward kinematics are calculated using the following intermediate variables:

$$a_1 = l_1 \cos \theta_1 - l_2 \cos \theta_2 \quad (\text{C.4})$$



Figure C.4: A side view of the 3GM robotic finger.

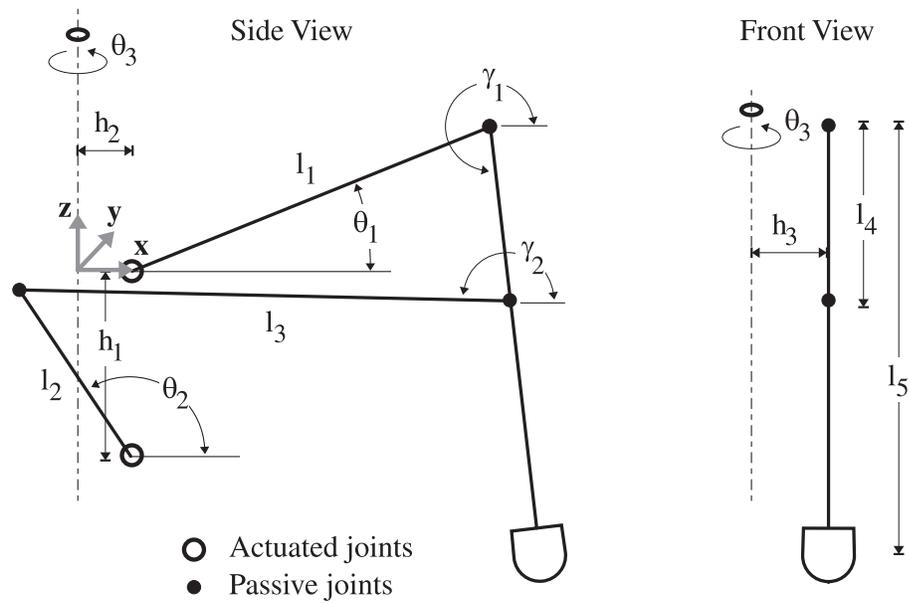


Figure C.5: 3GM kinematic variables.

$$\begin{aligned}
 a_2 &= l_1 \sin \theta_1 - l_2 \sin \theta_2 + h_1 \\
 b_1 &= \frac{l_3^2 - l_4^2 - a_1^2 - a_2^2}{2l_4} \\
 b_2 &= \frac{l_3^2 - l_4^2 + a_1^2 + a_2^2}{2l_3}; \\
 c &= \arctan \left(\frac{a_2}{\sqrt{a_1^2 + a_2^2}}, \frac{a_1}{\sqrt{a_1^2 + a_2^2}} \right).
 \end{aligned}$$

While the device is designed so that singularities are not encountered in the operational workspace, it is possible to reach them in free motion. One of the singularities occurs when $\frac{b_1}{\sqrt{a_1^2 + a_2^2}}$ or $\frac{b_2}{\sqrt{a_1^2 + a_2^2}}$ is ≥ 1 . This position also presents a problem in the forward kinematics computation because the following arccos calculations cannot be performed. If the finger is not in this singular position, the passive joint angles are calculated from

$$\begin{aligned}
 \gamma_1 &= c - \arccos \left(\frac{b_1}{\sqrt{a_1^2 + a_2^2}} \right) \\
 \gamma_2 &= c - \arccos \left(\frac{b_2}{\sqrt{a_1^2 + a_2^2}} \right).
 \end{aligned} \tag{C.5}$$

The selection of the arccos result (there are two solutions) in the calculation of γ_1 determines whether the robot is in “elbow up” or “elbow down” configuration. The robot was used in “elbow up” configuration in the work presented in this thesis. γ_2 is not used in any of the subsequent kinematics calculations. However, the correct value for γ_2 can be selected to match the elbow configuration determined by γ_1 .

The endpoint position in Cartesian Coordinates is then calculated using

$$\begin{aligned}
 r &= l_1 \cos \theta_1 + l_5 \cos \gamma_1 \\
 x &= (r + h_2) \cos \theta_3 + h_3 \sin \theta_3 \\
 y &= (r + h_2) \sin \theta_3 - h_3 \cos \theta_3 \\
 z &= l_1 \sin \theta_1 + l_5 \sin \gamma_1.
 \end{aligned} \tag{C.6}$$

Jacobian

The device Jacobian is calculated using partial derivatives of the forward kinematics.

$$J = \begin{bmatrix} \frac{\partial x}{\partial \theta_1} & \frac{\partial x}{\partial \theta_2} & \frac{\partial x}{\partial \theta_3} \\ \frac{\partial y}{\partial \theta_1} & \frac{\partial y}{\partial \theta_2} & \frac{\partial y}{\partial \theta_3} \\ \frac{\partial z}{\partial \theta_1} & \frac{\partial z}{\partial \theta_2} & \frac{\partial z}{\partial \theta_3} \end{bmatrix} \quad (\text{C.7})$$

Before writing out the final equation for the Jacobian, let us first calculate several intermediate variables.

$$\begin{aligned} c_1 &= \frac{l_1 l_5 \sin(\theta_1 - \gamma_1)}{l_4 \sin(\gamma_2 - \gamma_1)} \\ c_2 &= \frac{l_2 l_5 \sin(\gamma_2 - \theta_2)}{l_4 \sin(\gamma_2 - \gamma_1)} \\ \frac{\partial r}{\partial \theta_1} &= -l_1 \sin \theta_1 - c_1 \sin \gamma_1 \\ \frac{\partial r}{\partial \theta_2} &= -\cos \theta_3 \sin \theta_1 \end{aligned} \quad (\text{C.8})$$

$$J = \begin{bmatrix} \frac{\partial r}{\partial \theta_1} \cos \theta_3 & \frac{\partial r}{\partial \theta_2} \cos \theta_3 & -(r + h_2) \sin \theta_3 + h_3 \cos \theta_3 \\ \frac{\partial r}{\partial \theta_1} \sin \theta_3 & \frac{\partial r}{\partial \theta_2} \sin \theta_3 & (r + h_2) \cos \theta_3 + h_3 \sin \theta_3 \\ l_1 \cos \theta_1 + c_1 \cos \gamma_1 & c_2 \cos \gamma_1 & 0 \end{bmatrix} \quad (\text{C.9})$$

The Jacobian is used for two purposes in fingertip control: calculating endpoint velocity based on joint angle velocities, and calculating the necessary motor torques to output a desired Cartesian force.

$$\begin{aligned} \mathbf{v} &= J\dot{\boldsymbol{\theta}} \\ \boldsymbol{\tau} &= J^T \mathbf{f} \end{aligned} \quad (\text{C.10})$$

Gravity Compensation

Gravity compensation is used to add torque to the motors in order to balance the weight of the finger. While full gravity compensation requires exact measurement of the masses and locations of every component in the finger, a simple approximation can be used. Observing the kinematics of the fingertip in the pose used for exploration, the first motor (θ_1) applies most of the vertical force. Thus, this motor alone can be

used to apply gravity compensation. The compensation torque is approximated as

$$\tau_g = Ml_5 \cos \theta_1, \quad (\text{C.11})$$

where M is an empirically determined mass value for the system. While this method works well for the fingertip position in the haptic exploration experiments in Chapter 4, full gravity compensation would be necessary for general applications.

Software Modules

Control software for the finger is divided into several modules:

- **Main Program and Interrupt Service Routine.** This module runs the high-level control of the finger and the user interface. The interrupt reads the kinematics of the device, calculates desired output forces based on the exploration state and control law, and outputs torques to the motors. The interrupt is read at a constant rate, typically 1kHz.
- **Vector and Matrix Operations Methods.** Because matrix and vector multiplication, addition, and subtraction are commonly used in control of the finger, object-oriented programming was used create structures and handle basic vector and matrix operations.
- **Transformations Module.** There are several different coordinate systems used in the haptic exploration control. This module records the transformations between the World Coordinate System (WCS), Tactile Coordinate System (TCS) and Surface Coordinate System (SCS) in the form of rotation matrices.
- **Tactile Sensor.** The tactile sensor module handles the reading and interpretation of tactile sensor data to determine the angle of the contact normal in the Tactile Coordinate System. This information is passed to the Interrupt Service Routine for use in exploration.
- **Finger Kinematics and Force Output.** The finger kinematics and force output commands are located in a module that deals only with the kinematics and dynamics of the robotic finger. Based on measured joint angles and velocities, the state of the finger is calculated. When a desired output force is determined in the Main Program, the necessary motor torques are calculated in this module.

- D/A and A/D Methods. Low-level commands to read and output data, providing communication between the finger/tactile sensor and the computer boards, are located in this module.

Bibliography

- [1] Peter K. Allen. Integrating vision and touch for object recognition tasks. *International Journal of Robotics Research*, 7(6):15–33, 1988.
- [2] Peter K. Allen. Mapping haptic exploratory procedures to multiple shape representations. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1679–1684, 1990.
- [3] Peter K. Allen and Paul Michelman. Acquisition and interpretation of 3-d sensor data from touch. *IEEE Transactions on Robotics and Automation*, 6(4):397–404, 1990.
- [4] Peter K. Allen, Andrew T. Miller, Paul Y. Oh, and Brian S. Leibowitz. Using tactile and visual sensing with a robotic hand. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 676–681, 1997.
- [5] John S. Bay. Tactile shape sensing via single- and multi-fingered hands. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 290–295, 1989.
- [6] John S. Bay and Hooshang Hemami. Dynamics of a learning controller for surface tracking robots on unknown surfaces. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1910–1915, 1990.
- [7] Thomas G. Beckwith, Roy D. Marangoni, and John H. Lienhard. *Mechanical Measurements*. Addison-Wesley, 1993.
- [8] A. D. Berger and P. K. Khosla. Using tactile data for real-time feedback. *International Journal of Robotics Research*, 10(2):88–102, 1991.
- [9] H. Blum. A transformation for extracting new descriptors of shape. *Models for Perception of Speech and Visual Form*, pages 362–380, 1967.

- [10] J. W. Brandt and V. R. Algazi. Continuous skeleton computation by voronoi diagram. *CVGIP: Image Understanding*, 55(3):329–338, 1992.
- [11] K. N. Brown, C. A. McMahon, and J. H. Sims Williams. Features, aka the semantics of a formal language of manufacturing. *Research in Engineering Design*, 7(3):151–172, 1995.
- [12] C. Cai and B. Roth. On the spatial motion of a rigid body with point contact. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 686–695, 1987.
- [13] S. Caselli, C. Magnanini, F. Zanichelli, and E. Caraffi. Efficient exploration and recognition of convex objects based on haptic perception. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3508–3515, 1996.
- [14] Dean Chang and Mark R. Cutkosky. Rolling with deformable fingertips. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2:194–199, 1995.
- [15] N. Chang, H. Zhang, and R. Rink. Edge tracking using tactile servo. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2:84–89, 1995.
- [16] I-Ming Chen and Joel W. Burdick. Qualitative test for n-finger force-closure grasps on planar objects with applications to manipulation and finger gaits. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 814–820, 1993.
- [17] Ching-Shoei Chiang. Algorithms for extracting the medial axis transform of 2d images. *Proceedings of SPIE - The International Society for Optical Engineering, Visual Communications and Image Processing*, 2501(2):1173–1182, 1995.
- [18] Arlene A. Cole, Ping Hsu, and Shankar S. Sastry. Dynamic regrasping by coordinated control of sliding for a multifingered hand. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 781–786, 1989.
- [19] Mark R. Cutkosky and Imin Kao. Computing and controlling compliance of a robotic hand. *RIMANSY 88: Seventh CISM-IFTOMM Symposium on the Theory and Practice of Robots and Manipulators*, 1988.

- [20] P. Dario and M. Bergamasco. Advanced robot system for automated diagnostic tasks through palpation. *IEEE Transactions on Biomedical Engineering*, 35(2):118–126, 1988.
- [21] E. R. Davies and A. P. N. Plummer. Thinning algorithms: a critique and a new methodology. *Pattern Recognition*, 14(1-6):53–63, 1981.
- [22] D. Eberly, R. Gardner, B. Morse, S. Pizer, and C. Scharlach. Ridges for image analysis. *Journal of Mathematical Imaging and Vision*, 4(4):353–373, 1994.
- [23] B. Eberman and J. Kenneth Salisbury. Application of change detection to dynamic contact sensing. *International Journal of Robotics Research*, 13(5):369–394, 1994.
- [24] R. E. Ellis. Extraction of tactile features by passive and active sensing. *Intelligent Robots and Computer Vision, Proceedings of SPIE*, 521:289–295, 1984.
- [25] R. S. Fearing. Tactile sensing mechanisms. *International Journal of Robotics Research*, 9(3):3–23, 1990.
- [26] Ralph E. Goddard, Yuan F. Zheng, and Hooshang Hemami. Dynamic hybrid velocity/force control of robot compliant motion over globally unknown objects. *IEEE Transactions on Robotics and Automation*, 8(1):132–138, 1992.
- [27] Alex S. Goldenberg, Evan F. Wies, Kenneth Martin, and Christopher J. Hasser. Next-generation 3d haptic feedback system. Poster, American Society of Mechanical Engineers Haptics Symposium, November 1998.
- [28] Donald F. Green and J. Kenneth Salisbury. Texture sensing and simulation using the phantom: Towards remote sensing of soil properties. *Proceedings of the Second PHANToM Users Group Workshop*, 1997.
- [29] R. A. Grupen, T. C. Henderson, and I. D. McCammon. Survey of general purpose manipulation. *International Journal of Robotics Research*, 8(1):38–62, 1989.
- [30] Hooshang Hemami. Differential surface models for tactile perception of shape and on-line tracking of features. *IEEE Transactions on Systems, Man, and Cybernetics*, 18(2):312–316, 1988.

- [31] Hooshang Hemami, John S. Bay, and Ralph E. Goddard. A conceptual framework of tactually guided exploration and shape perception. *IEEE Transactions on Biomedical Engineering*, 35(2):99–109, 1988.
- [32] R. A. Hilhorst and K. Tanie. Dextrous manipulation of objects with unknown parameters by robot hands. *Proceedings of the IEEE International Conference on Robotics and Automation*, 4:3098–3103, 1994.
- [33] Neville Hogan. Impedance control: an approach to manipulation: parts i, ii, and iii. *ASME Journal of Dynamic Systems, Measurement, and Control*, 107:1–24, 1985.
- [34] Robert D. Howe and Mark R. Cutkosky. Touch sensing for robotic manipulation and recognition. In O. Khatib, J. Craig, and T. Lozano-Perez, editors, *The Robotics Review 2*, pages 55–112. MIT Press, 1992.
- [35] Robert D. Howe and Mark R. Cutkosky. Dynamic tactile sensing: Perception of fine surface features with stress rate sensing. *IEEE Transactions on Robotics and Automation*, 9(2):140–151, 1993.
- [36] Robert D. Howe and Mark R. Cutkosky. Practical force-motion models for sliding manipulation. *International Journal of Robotics Research*, 15(6):557–572, 1996.
- [37] Victoria Interrante. Illustrating surface shape in volume data via principal direction-driven 3d line integral convolution. *Proceedings of 24th International Conference on Computer Graphics and Interactive Techniques Los Angeles*, pages 109–116, 1997.
- [38] Imin Kao and Mark R. Cutkosky. Quasistatic manipulation with compliance and sliding. *International Journal of Robotics Research*, 11(1):20–40, 1992.
- [39] Ju-Hsien Kao. *Process Planning For Additive/Subtractive Solid Freeform Fabrication Using Medial Axis Transform*. PhD thesis, Stanford University, Department of Mechanical Engineering, 1999.
- [40] Jeffrey Kerr and Bernard Roth. Analysis of multifingered hands. *International Journal of Robotics Research*, 4(4):3–17, 1986.

- [41] Robert L. Klatzky and Susan Lederman. Intelligent exploration by the human hand. In Subramanian T. Venkataraman and Thea Iberall, editors, *Dextrous Robot Hands*, pages 66–81. Springer-Verlag, 1990.
- [42] Jan J. Koenderink. *Solid Shape*. The MIT Press, 1990.
- [43] T. L. Kunii, A. G. Belyaev, E. V. Anoshkina, S. Takahashi, R. Huang, and O. G. Okunev. Hierarchic shape description via singularity and multiscaling. *Proceedings of the 18th Annual International Computer Software and Applications Conference (COMPSAC 94)*, pages 242–251, 1994.
- [44] I. S. Kweon and T. Kanade. Extracting topographic terrain features from elevation maps. *CVGIP: Image Understanding*, 59(2):171–182, 1994.
- [45] Susanna R. Leveroni. *Grasp Gaits for Planar Object Manipulation*. PhD thesis, Massachusetts Institute of Technology, Department of Mechanical Engineering, 1997.
- [46] Z. X. Li, Z. Qin, S. Jiang, and L. Han. Coordinated motion generation and real-time grasping force control for multifingered manipulation. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3631–3638, 1998.
- [47] Chek T. Lim, George M. Turkiyyah, Mark A. Ganter, and Duane W. Storti. Implicit reconstruction of solids from cloud point sets. *ACM SIGGRAPH Symposium on Solid Modeling and Applications*, pages 393–402, 1995.
- [48] Vladimir J. Lumelsky and Tim Skewis. Incorporating range sensing in the robot navigation function. *IEEE Transactions on Systems, Man and Cybernetics*, 20(5):1058–1069, 1990.
- [49] H. Maekawa, K. Tanie, and K. Komoriya. Tactile sensor based manipulation of an unknown object by a multifingered hand with rolling contact. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 743–750, 1995.
- [50] H. Maekawa, K. Yokoi, K. Tanie, M. Keneko, N. Kimura, and N. Imamura. Development of a three-fingered robot hand with stiffness control capability. *Mechatronics*, 2(5):483–494, 1992.

- [51] Hitoshi Maekawa, Kazuo Tanie, and Kiyoshi Komoriya. A finger-shaped tactile sensor using an optical waveguide. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, pages 403–408, 1993.
- [52] Hitoshi Maekawa, Kazuo Tanie, and Kiyoshi Komoriya. Tactile feedback for multifingered dynamic grasping. *IEEE Control Systems Magazine*, 17(1):63–72, 1997.
- [53] J. B. A. Maintz, P. A. van den Elsen, and M. A. Viergever. Evaluation of ridge seeking operators for multimodality medical image matching. *IEEE Trans. of Pattern Analysis and Machine Intelligence*, 18(4):353–365, 1996.
- [54] Matthew T. Mason and J. Kenneth Salisbury. *Robot Hands and the Mechanics of Manipulation*. The MIT Press, 1985.
- [55] W. Mendenhall and T. Sincich. *Statistics for Engineering and the Sciences*. Prentice Hall, 1995.
- [56] Olivier Monga, Serge Benayoun, and Olivier D. Faugeras. From partial derivatives of 3d density images to ridge lines. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 354–359, 1992.
- [57] David J. Montana. The kinematics of contact and grasp. *International Journal of Robotics Research*, 7(3):17–32, 1988.
- [58] David J. Montana. The kinematics of multi-fingered manipulation. *IEEE Transactions on Robotics and Automation*, 11(4):491–503, 1995.
- [59] R. Murray, Zexiang Li, and Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.
- [60] C. Muthukrishnan, D. Smith, D. Meyers, J. Rebman, and A. Koivo. Edge detection in tactile images. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1500–1505, 1987.
- [61] C. Wayne Niblack, David W. Capson, and Phillip B. Gibbons. Generating skeletons and centerlines from the medial axis transform. *Proceedings of the International Conference on Pattern Recognition*, 1:881–885, 1990.

- [62] H. R. Nicholls and M. H. Lee. Survey of robot tactile sensing technology. *International Journal of Robotics Research*, 8(3):3–30, 1989.
- [63] E. J. Nicolson and R. S. Fearing. Reliability of curvature estimates from linear elastic tactile sensors. *Proceedings of the IEEE International Conference on Robotics and Automation*, 1:1126–1133, 1995.
- [64] R. L. Ogniewicz. Skeleton-space: a multiscale shape description combining region and boundary information. *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 746–751, 1994.
- [65] Allison M. Okamura, Michael A. Costa, Michael L. Turner, Christopher Richard, and Mark R. Cutkosky. Haptic exploration of surfaces. In Peter Corke and James Trevelyan, editors, *Experimental Robotics VI*, volume 250 of *Lecture Notes in Control and Information Sciences*, pages 423–432. Springer-Verlag, 2000.
- [66] Allison M. Okamura, Jack Tigh Dennerlein, and Robert D. Howe. Vibration feedback models for virtual environments. *Proceedings of the IEEE International Conference on Robotics and Automation*, 1:674–679, 1998.
- [67] Allison M. Okamura, Matthew W. Hage, Jack Tigh Dennerlein, and Mark R. Cutkosky. Improving reality-based models for vibration feedback. *Proceedings of the ASME Dynamic Systems and Control Division*, page to appear, 2000.
- [68] N. M. Patrikalakis and H. N. Gursoy. Shape interrogation by medial axis transform. *Proceedings of the ASME Design Technical Conferences, Design Engineering Division*, 3(1):77–88, 1990.
- [69] I. Pitas. *Digital Processing Algorithms*. Prentice Hall, 1995.
- [70] K. Pribadi, J. S. Bay, and H. Hemami. Exploration and dynamic shape estimation by a robotic probe. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(4):840–846, 1989.
- [71] Christopher Richard, Mark R. Cutkosky, and Karon MacLean. Friction identification for haptic display. *Proceedings of the ASME Dynamic Systems and Control Division*, pages 327–334, 1999.

- [72] K. Roberts. Robot active touch exploration: Constraints and strategies. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1679–1684, 1990.
- [73] J. R. Rossignac and A. A. G. Requicha. Offsetting operations in solid modeling. *Computer Aided Geometric Design*, 3(2):129–148, 1986.
- [74] Nilanjan Sarkar, Xiaoping Yun, and Vijay Kumar. Dynamic control of 3-d rolling contacts in two-arm manipulation. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 978–983, 1993.
- [75] S. Schneider and R. H. Cannon. Experimental object-level strategic control with cooperating manipulators. *International Journal of Robotics Research*, 12(4):338–350, 1993.
- [76] Frank Y. Shih and Christopher C. Pu. Medial axis transform with single-pixel and connectivity preservation using euclidean distance computation. *Proceedings of the International Conference on Pattern Recognition*, 1:723–725, 1990.
- [77] K. B. Shimoga. Robot grasp synthesis algorithms: a survey. *International Journal of Robotics Research*, 15(3):230–266, 1996.
- [78] Juhani Siira and Dinesh K. Pai. Haptic texturing - a stochastic approach. *Proceedings of the IEEE International Conference on Robotics and Automation*, 1:557–562, 1996.
- [79] Pavan Sikka, Hong Zhang, and Steve Sutphen. Tactile servo: Control of touch-driven robot motion. *Proceedings of the Third International Symposium on Experimental Robotics*, pages 155–161, 1993.
- [80] G. Simmons. *Calculus with Analytic Geometry*. McGraw-Hill, 1985.
- [81] Frac Solina and Ruzena Bajcsy. Recovery of parametric models from range images: the case for superquadrics with global deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(2):131–147, 1990.
- [82] J. S. Son, M. R. Cutkosky, and R. D. Howe. Comparison of contact sensor localization abilities during manipulation. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2:96–103, 1995.

- [83] J. S. Son and R. D. Howe. Tactile sensing and stiffness control with multifingered hands. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3228–3233, 1996.
- [84] Michael Spivak. *A comprehensive introduction to differential geometry, Volume Three*. Berkeley: Publish or Perish, Inc., 1979.
- [85] Sharon Stansfield. A robotic perceptual system utilizing passive vision and active touch. *International Journal of Robotics Research*, 7(6):138–161, 1988.
- [86] Duane W. Storti, George M. Turkiyyah, Mark A. Ganter, Chek T. Lim, and Derek M. Stal. Skeleton-based modeling operations on solids. *Proceedings of the Symposium on Solid Modeling and Applications*, pages 141–154, 1997.
- [87] Don Stredney, Gregory J. Wiet, Roni Yagel, Dennis Sessanna, Yair Kurzion, Mark Fontana, Naeem Shareef, Mike Levin, Kenneth Martin, and Allison Okamura. A comparative analysis of integrating visual representations with haptic displays. In J.D. Westwood, H.M. Hoffman, D. Stredney, and S.J. Weghorst, editors, *Medicine Meets Virtual Reality*, volume 50 of *Studies in Health Technology and Informatics*, pages 20–26. IOS Press and Ohmsha, 1998.
- [88] Atul Sudhalkar, Levent Gursoz, and Fritz Prinz. Continuous skeletons of discrete objects. *Proceedings of the Symposium on Solid Modeling and Applications*, pages 85–94, 1993.
- [89] Marc R. Tremblay, James M. Hyde, and Mark R. Cutkosky. An object-oriented framework for event-driven dextrous manipulation. In O. Khatib and J.K. Salisbury, editors, *Experimental Robotics IV*, pages 53–61. Springer, 1997.
- [90] E. Trucco and A. Verri. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.
- [91] George M. Turkiyyah, Duane W. Storti, Mark Ganter, Hao Chem, and Munikumar Vimawala. An accelerated triangulation method for computing the skeletons of free-form solid models. *Computer Aided Design*, pages 5–19, 1997.
- [92] David J. Whitehouse. *Handbook of Surface Metrology*. Bristol, Philadelphia: Institute of Physics Publishing, 1994.

- [93] Tsuneo Yoshikawa and Kiyoshi Nagai. Analysis of multi-fingered grasping and manipulation. In Subramanian T. Venkataraman and Thea Iberall, editors, *Dextrous Robot Hands*, pages 5–31. Springer-Verlag, 1990.
- [94] H. Zhang, H. Maekawa, and K. Tanie. Sensitivity analysis and experiments of curvature estimation based on rolling contact. *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3514–3519, 1996.