Shape Reconstruction Using Active Tactile Sensors

Mark Moll

July 22, 2002

CMU-CS-02-157

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy.

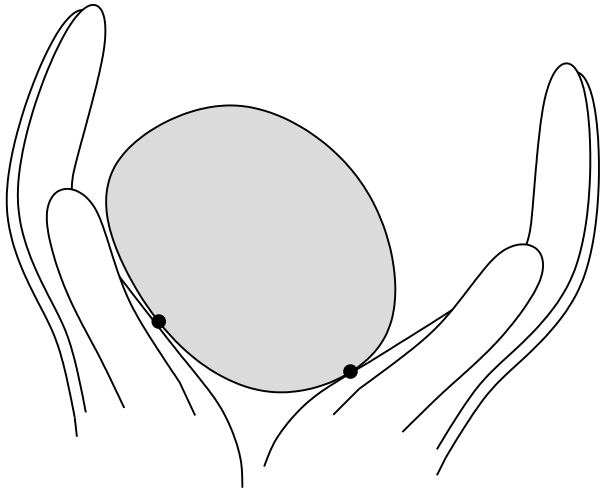
Thesis Committee:
Michael A. Erdmann, Chair
Kenneth Y. Goldberg, University of California, Berkeley
Matthew T. Mason
Alfred A. Rizzi

This work was supported in part by the National Science Foundation under grant IIS-9820180

Keywords: Tactile sensing, shape reconstruction, nonprehensile manipulation, contact kinematics

Shape Reconstruction Using Active Tactile Sensors

Mark Moll



Abstract

We present a new method to reconstruct the shape of an unknown object using tactile sensors, without requiring object immobilization. Instead, sensing and nonprehensile manipulation occur simultaneously. The robot infers the shape, motion and center of mass of the object based on the motion of the contact points as measured by the tactile sensors. This allows for a natural, continuous interaction between manipulation and sensing. We analyze the planar case first by assuming quasistatic dynamics, and present simulation results and experimental results obtained using this analysis. We extend this analysis to the full dynamics and prove observability of the nonlinear system describing the shape and motion of the object being manipulated. In our simulations, a simple observer based on Newton's method for root finding can recover unknown shapes with almost negligible errors. Using the same framework we can also describe the shape and dynamics of three-dimensional objects. However, there are some fundamental differences between the planar and three-dimensional case, due to increased tangent dimensionality. Also, perfect global shape reconstruction is impossible in the 3D case, but it is almost trivial to obtain upper and lower bounds on the shape. The 3D shape reconstruction method has also been implemented and we present some simulation results.

Acknowledgments

First and foremost, I would like to thank the members of my thesis committee for all their time and efforts. I consider myself very lucky having had Michael Erdmann as my advisor. He has given me a tremendous amount of freedom in exploring different directions. During our meetings he was always able to ask the right questions. Matt Mason has also been a great source of good ideas. Especially his intuitive mechanical insights have been really useful. Al Rizzi has been very helpful in advising me on some control and engineering problems. Ken Goldberg, my external committee member, has given me excellent feedback. I am also grateful to him and Michael Erdmann for having given me the opportunity to work in Ken Goldberg's lab on a side project.

Before coming to Carnegie Mellon I have been fortunate to have worked with two great researchers. I am greatly indebted to Anton Nijholt, who has supported me in many ways during my graduate studies at the University of Twente in the Netherlands and long afterward. I also enjoyed my collaboration with Risto Miikkulainen at the University of Texas in Austin. Through him I learned a lot about doing research. He also encouraged me to pursue a PhD degree.

Finally, I would like to thank members of the Manipulation Lab for their support and brutal honesty during many of my lab presentations: Yan-Bin Jia, Garth Zeglin, Devin Balkcom, Siddartha Srinivasa, and Ravi Balasubramanian. Many thanks also to honorary Manipulation Lab member Howard Choset for his support and advice.

Contents

List of Figures

Notation

The notation in each chapter relies as much as possible on the notation used in previous chapters. Below we only define notation introduced in the corresponding chapter.

Chapter 3

s_1, s_2	sensor values on palm 1 and palm 2, respectively
ϕ_1	angle between X-axis of the world frame and palm 1
ϕ_2	angle between palm 1 and palm 2
ϕ_0	orientation of the object held by the palms
R_0	rotation matrix that maps points in the object's body coordinates to world coordinates
$\boldsymbol{c}_m$	center of mass of the object
$\boldsymbol{c}_r$	center of rotation of the object
$\boldsymbol{x}(\theta)$	curve describing the shape of the object
$v(\theta)$	radius of curvature at $\boldsymbol{x}(\theta)$
$\boldsymbol{n}(\theta), \boldsymbol{t}(\theta)$	normal and tangent at $\boldsymbol{x}(\theta)$
$\bar{\boldsymbol{n}}_i, \bar{\boldsymbol{t}}_i$	normal and tangent at the contact point on palm i in world coordinates
$(r(\theta), d(\theta))$	contact support function
$(\tilde{r}_i(\theta), \tilde{d}_i(\theta))$	generalized contact support function relative to contact point i
e_f	error in force/torque balance constraint
$\boldsymbol{e}_c$	error in two-point contact constraint

Chapter 4

$\boldsymbol{q}$	vector describing the state of the system formed by the palms and the object
$\boldsymbol{F}_z$	the gravity vector acting on the object
g	the gravity constant, equal to $-9.81\mathrm{m/s}^2$
$\boldsymbol{F}_{c_i}$	contact force at contact point i acting on the object

f_{c_i}	magnitude of $\boldsymbol{F}_{c_i}$
τ_{c_i}	torque generated by the contact force at contact point i on the object
τ_i	torque exerted by the motor of palm i
$\boldsymbol{f}(\boldsymbol{q})$	drift vector field; the rate of change of the system if no torques are applied
$\boldsymbol{g}_i(\boldsymbol{q})$	control vector field i; the rate of change of the system due to τ_i
$\boldsymbol{h}(\boldsymbol{q})$	output function of the system
$\boldsymbol{y}$	output vector of the system; $\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{q})$
$\omega_i,\ i = 0,1,2$	rotational velocity of the object, palm 1, and palm 2
$\alpha_i,\ i = 0,1,2$	angular acceleration of the object, palm 1, and palm 2
$\boldsymbol{a}_0$	acceleration of the object
$I_i,\ i = 0,1,2$	moment of inertia of the object, palm 1, and palm 2
m	mass of the object
ρ	radius of gyration of the object
$d\phi(\boldsymbol{q})$	differential of a function ϕ; $d\phi(\boldsymbol{q}) = \left(\frac{\partial \phi}{\partial q_1}, \ldots, \frac{\partial \phi}{\partial q_n}\right)$
$L_X\phi$	Lie derivative of a function ϕ along a vector field X; $L_X\phi = d\phi \cdot X$
$\boldsymbol{c}_i$	contact point i in world coordinates
b_i	radius of circular palm i
R_i	matrix describing the orientation of palm i

Chapter 5

$\boldsymbol{x}(\boldsymbol{\theta})$	surface describing the shape of the object
$[\boldsymbol{t}_1(\boldsymbol{\theta}),\boldsymbol{t}_2(\boldsymbol{\theta}),\boldsymbol{n}(\boldsymbol{\theta})]$	local coordinate frame at $\boldsymbol{x}(\boldsymbol{\theta})$
$(r(\boldsymbol{\theta}),d(\boldsymbol{\theta}),e(\boldsymbol{\theta}))$	3D contact support function
$[\bar{\boldsymbol{t}}_{i1},\bar{\boldsymbol{t}}_{i2},\bar{\boldsymbol{n}}_i]$	coordinate frame at the contact point on palm i in world coordinates.
$\boldsymbol{\beta}_i(t)$	curve in object coordinates traced out by contact point i on the surface of the object.
$\boldsymbol{s}_i = (s_{i1}, s_{i2}, 0)^T$	vector of sensor values at palm i
$\mathcal{I}$	inertia matrix of the object
ϕ_i	angle between palm i and the horizontal plane

Chapter 1

Introduction

1.1 Motivation

There are many examples in everyday life of robotic systems that have almost nothing in common with the walking and talking humanoids of popular culture: robots assemble cars, assist in telesurgery, and can be used to assemble nano-structures. In all these examples it is critical that the robot can manipulate, sense, and reason about its environment. We can create a better understanding of fundamental manipulation and sensing strategies through rigorous exploration of the algorithmic and geometric aspects of robotics. For many tasks there is a trade-off between manipulation and sensing. For example, a robot can establish the orientation of an object with a vision system or by pushing and aligning the object with a fixture. There is not only a trade-off, but there are also interactions between action and sensing. Sensor information can be used to guide certain actions, and manipulation can be used to reduce uncertainty about the state of the environment. There are many seemingly simple tasks such as tying shoe laces or recognizing an object by touching it, that pose enormous problems for automated systems today. The difficulty of these tasks is caused in part by the uncertainty in the knowledge about the environment and by the difficulty of tightly integrating sensing with manipulation. In our research we aim to tighten this integration as well as explore what the minimal manipulation and sensing requirements are to perform a given task.

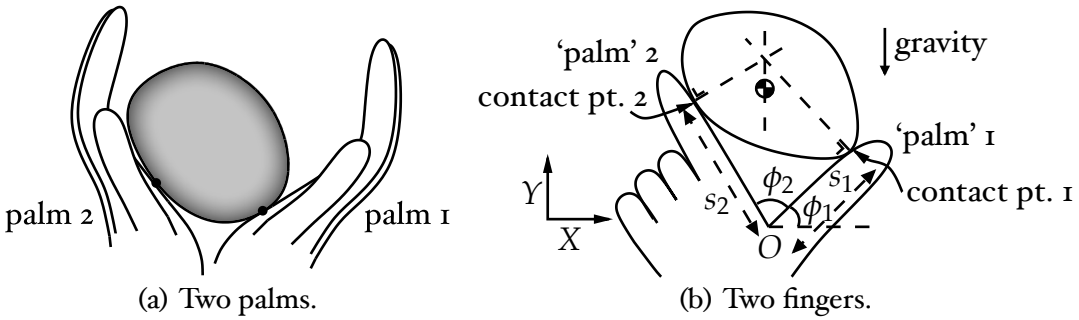
Robotic manipulators typically cannot deal very well with objects of partially unknown shape and weight. Humans, on the other hand, seem to have few problems with manipulating objects of unknown shape and weight. For example, Klatzky et al. (1985) showed that blindfolded human observers identified 100 common objects with over 96% accuracy, in only 1 to 2 seconds for most objects. Besides recognizing shape and size, humans also use touch to determine vari-

ous features such as texture, hardness, thermal qualities, weight and movement (Lederman and Browse, 1988).

It seems unlikely that people mentally keep track of the *exact* position, shape and mass properties of the objects in their environment. So somehow during the manipulation of an unknown object the tactile sensors in the human hand give enough information to find the pose and shape of that object. At the same time some mass properties of the object are inferred to determine a good grasp. These observations are an important motivation for our research on tactile shape reconstruction. In most research on tactile shape reconstruction it is assumed that the object being touched is in a fixture, or at least does not move as result of being touched by a robot (Fearing and Binford, 1988; Montana, 1988; Allen and Michelman, 1990; Okamura and Cutkosky, 1999; Kaneko and Tsuji, 2001). This makes the shape reconstruction problem significantly easier, but it introduces another problem: how to immobilize an unknown object. In this thesis we don not assume the object is immobilized. Instead, we will solve for the motion and shape of object simultaneously. In the process we will also recover the mass properties of the object. We will show that the local shape and motion of an unknown object can be expressed as a function of the tactile sensor data and the motion of the manipulators. By 'local shape' we mean the shape at the contact points. Our approach allows for a natural, continuous interaction between manipulation and sensing.

As tactile sensors become more and more reliable and inexpensive, it seems inevitable that touch sensors will be mounted on more general purpose robot hands and manipulators. Being able to manipulate unknown objects is very valuable for robots that interact with the real world. It is often impossible to predict what the exact position of a robot and the exact forces should be to manipulate an object. Being able to handle these uncertainties allows a robot to execute tasks that may only specify approximate descriptions of shape and position of an object of interest. Pushing this idea even further, tactile data can add another dimension to simultaneous localization and mapping (SLAM), an important problem for mobile robots. But also in settings where the shapes and positions of objects in the robot's workspace are known, tactile sensors could provide useful information. It is possible that there are slight variations in the shape and position of the objects, or that there are errors in the robot's position. With the information from tactile sensors the robot can adjust its position and refine its grasp. Tactile sensing may also be useful in manipulating deformable objects. This would have applications in, e.g., medical robotics.

We could also reconstruct an unknown shape with a camera by taking a series of pictures and reconstruct the shape from that. However, recovering a shape



(a) Two palms. (b) Two fingers.

Figure 1.1: Two possible arrangements of a smooth convex object resting on palms that are covered with tactile sensors.

from a series of images is not necessarily easier than recovering a shape from a set of curves traced out by the contact points. Often lighting and occlusion are beyond control of the robot and make the reconstruction problem much harder. Also, from camera images it is not possible to infer mass properties of the unknown object. These mass properties are important if we want to manipulate the object. As we will describe later in this thesis, it *is* possible to infer these mass properties from tactile data. Of course, cameras and tactile sensors are not mutually exclusive. Ideally, the information obtained from all sensors would be combined to guide the robot. In this thesis we will focus exclusively on the information that can be extracted from tactile data.

1.2 Problem Statement

Let a *palm* be defined as a *surface covered with tactile sensors.* Suppose we have an unknown smooth convex object in contact with a number of moving palms (two palms in two dimensions, three in three dimensions); the only forces acting on the object are gravity and the contact forces. The curvature along the surface of the object is assumed to be strictly greater than zero, so that we have only one contact point on each palm. For simplicity we assume that there is no friction. The central problem addressed in this thesis is the identification of the mapping from the motion and sensor values of the palms to the local shape, motion, and mass properties of an unknown object.

Figure 1.1 illustrates the basic idea. There are two palms that each have one rotational degree of freedom at the point where they connect. That allows us to change the angle between palm 1 and palm 2 and between the palms and the global frame. As we change the palm angles we keep track of the contact points through tactile elements on the palms. We are using touchpads as tactile sensors

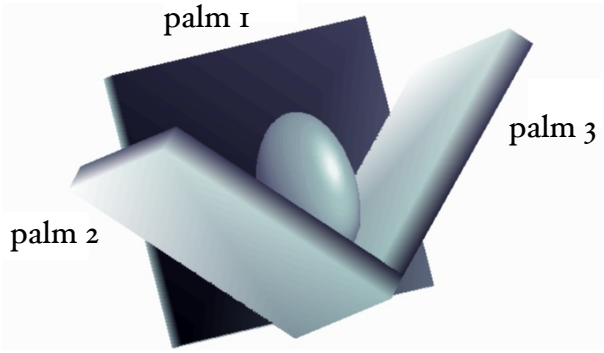


Figure 1.2: An object resting on three palms.

on the palms. In our experiments we found that we can track a contact point on a touchpad at approximately 0.1mm resolution and at 100Hz. Figure 1.2 shows a possible arrangement of three palms in three dimensions. Palm 2 and palm 3 are free to rotate around their line of intersection. This axis is rigidly connected to palm 1. Palm 1 can rotate around its bottom edge.

1.3 Thesis Outline

In the next chapter we will give an overview of related work. In chapter 3 we derive the shape and motion of an unknown smooth convex planar object in frictionless contact with two palms assuming quasistatic dynamics. That is, we will assume that the palms move slowly enough so that the object is always in force/torque balance. Another way to think about it is that the object rotates to minimize potential energy. We take the following approach in reconstructing the shape. We first derive expressions for the local shape at the contact points as a function of the sensor values, the motion of palms and the motion of the object. We then solve for the motion of the object by considering the dynamics.

The local shape is represented by the values of the *radius function* at the contact points. The radius function returns the projection onto the normal of the vector from the center of mass to a point on the surface of the object. Although shape is typically thought of as time invariant, we reparametrize the radius function with respect to time for each contact point. So for a given t the recovered radius function returns the values of the radius function at a contact point. The values of the radius function do not vary arbitrarily. Firstly, the values are constrained because we assume the object is smooth and convex. Secondly, the rate of change of the radius function depends on the motion of the object, which in turn depends on the mass properties. These mass properties are constant.

The solution for the shape and motion of the object are in the form of a system of differential equations, so the shape and motion of the object are obtained by integration of these equations. We demonstrate the feasibility of this approach with simulations and experimental results. We conjecture that, in general, it is possible to reconstruct the *entire* shape. This may seem obvious, but it could be, for instance, that the robot can only reconstruct one side of the shape and that it is impossible to reach the other side. To support this conjecture we analyze the stable orientations in the configuration space formed by the palm angles and object orientation. We show that the stable poses generally form a connected surface, so that there exists a path between any orientation of the object to any other orientation. The analysis of the stable pose surface gives rise to an open-loop manipulation strategy that has been used successfully in simulations. It is possible that the motion of the object has discontinuities, even if the motion of the palms is continuous. This results in different disconnected or overlapping pieces of the shape. We will present a method for piecing these different segments together.

In chapter 4 we will remove the quasistatic dynamics assumption. We will model the forces and torques exerted by the palms and by gravity. We can then solve for the acceleration and angular acceleration of the object. By integrating twice we can then obtain the position of the object at any given time. We will use results from non-linear control theory to prove observability for the system formed by the palms and the object. If a system is observable, then it is possible to correct errors as time progresses. We first analyze the general case and then consider some special cases: (1) moving both palms at the same rate, (2) moving only one palm, and (3) keeping both palms fixed. We prove that each case is observable, but in the general case and in case 3 we rely on the effects of the controls to prove observability. There is no general method to construct an observer for the general case and case 3. We *did* construct an observer for case 1 and 2. With an observer it is possible to not only detect errors, but also drastically reduce errors in the state estimate. This means that if the estimate for the initial conditions for the system of differential equations is slightly wrong, an observer will correct for that. More importantly, it filters out noise and numerical error that might otherwise accumulate. The observer we implemented is based on Newton's method for finding roots of a function. This observer works well in simulation.

Chapter 5 describes the generalization of the results in two dimensions to three dimensions. The same approach carries over to 3D, but there are some fundamental differences. Most importantly, the contact points now trace out curves on a *surface*, so it is impossible to create a complete reconstruction of

the shape in finite time. We will show that these curves are less constrained than the contact point curves in 2D. This makes it impossible to construct an observer within the current framework. Nevertheless, our simulations show that good results can still be obtained by integrating out the differential equations that describe the shape and motion of a 3D object. Although the system is not constrained enough to prove observability, we have derived additional constraints that can be used to minimize an error measure. But since the system is not observable, there is no guarantee that by minimizing this error measure the state estimate converges to the true state.

Finally, in chapter 6 we summarize the main contributions of this thesis and outline different future directions. Specifically, we describe how to remove some of the assumptions we made in this thesis. We also describe ways this work can be extended to different sensing modalities.

Chapter 2

Related Work

Our research builds on many different areas in robotics. These areas can be roughly divided into four different categories: probing, nonprehensile manipulation, grasping, and tactile sensing. We can divide the related work in tactile sensing further into three subcategories: shape and pose recognition with tactile sensors, tactile exploration, and tactile sensor design. We now briefly discuss some of the research in these areas.

2.1 Probing

Shape sensing can be approached purely geometrically and algorithmically. Sensing is then often called probing. One can define different kinds of probes that correspond to abstractions of sensor devices. For instance, a *finger probe* corresponds to a robotic finger moving along a line until it contacts an object (or misses the object). The probe outcome is then the point where the probe contacted the object. Typical questions are:

- How many probes are sufficient to reconstruct the shape of an object?
- How many probes are sufficient to recognize the pose of a known object?

Often these problems are restricted to a class of shapes (such as polygons). We can relax the questions above by trying to solve for the number of probes needed for a bounded error approximation of the exact answers. Cole and Yap (1987) showed that the answer to the first question using finger probes is $3n$ for a convex n-sided polygon. Furthermore, they showed that $3n - 1$ probes are necessary. If we assume that a finger probe outcome is never exactly a vertex of the polygon, then $3n$ probes are necessary. Shortly after (Cole and Yap, 1987) Dobkin et al. (1986) investigated the complexity of determining the shape and pose of convex polytopes for a variety of different probes, including probes with

errors. Boissonnat and Yvinec (1992) extended the probe model of Cole and Yap: their probe outcome includes the normal at the contact point. With this probe model they showed that at most $3n - 3$ probes are needed for simple *non-convex* polygons with no collinear edges. Their results can be extended to probe a set of polygons and to probe a set of polyhedra.

Li (1988) gave algorithms that reconstruct convex polygons with $3n + 1$ *line probes* or with $3n - 2$ *projection probes*. Line probes slide a straight line in a particular direction over the plane until it hits the object. Each probe reveals a tangent line to the object. Projection probes consist of two line probes that move in opposite directions towards each other. Lindenbaum and Bruckstein (1994) gave an approximation algorithm for *arbitrary* planar convex shapes using line probes. They showed that for an object with perimeter L no more than $O(\sqrt{L/\varepsilon}\log\frac{L}{\varepsilon})$ probes are needed to get an approximation error of ε. Kölzow et al. (1989) presented an approximation algorithm using projection probes, but their projection probes are defined as the length of the intersection of a line with the object. In (Lindenbaum and Bruckstein, 1991) bounds were given on the number of *parallel* probes that are necessary to recover the shape of a planar polygon. With parallel probes, k probes ($k > 1$) are performed at the same time. Skiena (1989) observed that the line probe can be generalized to a new kind of probe which is the dual of the finger probe, so that there is a one-to-one correspondence between algorithms that use finger probes and ones that use this generalized line probe.

Rao and Goldberg (1994) studied the problem of determining the shape of a convex polygon using diameter measurements from a parallel jaw gripper. They showed that there is an infinite set of polygonal shapes for a given set of diameter measurements. However, it is possible to recognize a shape from a known (finite) set of shapes. Rao and Goldberg presented sensing plans that require no more than n measurements, where n is the number of stable faces. Arkin et al. (1998) proved that finding a minimal length sensing plan is *NP*-hard and gave a polynomial-time approximation algorithm with a good performance guarantee. Akella and Mason (1999) showed how to orient and distinguish (sets of) polygonal parts using diameter measurements.

Skiena (1989) described many different probes and many (open) problems in probing. An overview of research on probing can be found in (Romanik, 1995).

2.2 Nonprehensile Manipulation

The basic idea behind nonprehensile manipulation is that robots can manipulate objects even if the robots do not have full control over these objects. This idea

was pioneered by Mason. In his Ph.D. thesis (Mason, 1982) and the companion paper (Mason, 1985) nonprehensile manipulation took the form of pushing an object in the plane to reduce uncertainty about the object's pose. Further work by Peshkin and colleagues (Peshkin and Sanderson, 1988; Wiegley et al., 1996) analyzed the pushing problem and showed how to design fences for a conveyor belt system. Berretty et al. (1998) proved the conjecture of Wiegley et al. (1996) that any polygonal part can be oriented by a sequence of fences and presented an $O(n^3 \log n)$ algorithm to compute the shortest such sequence. The work on fence design has recently been generalized to polyhedral objects by Berretty (2000). Berretty described a system where parts were fed from one conveyor belt to another, each belt with a sequence of fences. Akella et al. (2000) described a system where a sequence of fences was replaced with a one joint robot. The robot was basically a movable fence with one rotational degree of freedom. The robot could push a part up a conveyor belt and let it drift back. Akella et al. presented an algorithm that finds a sequence of pushes to orient a given polygonal part without sensing. Lynch (1997) further built on Mason's work. In his Ph.D. thesis Lynch described a path planner for quasistatically pushing objects among obstacles. He also investigated controllability of dynamic nonprehensile manipulation such as throwing and catching a part. Lynch et al. (1998) showed how to make a robotic manipulator perform a certain juggling motion with a suitable parameterization of the shape and motion of the manipulator. Much research on juggling balls has been done in Koditschek's research group (see e.g. (Rizzi and Koditschek, 1993) and (Whitcomb et al., 1993)). Rizzi and Koditschek (1993) described a system consisting of a robot arm and a camera that can juggle two balls. In (Abell and Erdmann, 1995) nonprehensile manipulation took the (abstract) form of moving two frictionless contacts on a polygonal part in a planar gravitational field. Abell and Erdmann presented an algorithm to orient such a polygonal part by moving the contact points and performing hand-offs between two pairs of contact points.

Erdmann and Mason (1988) described sensorless manipulation within the formal framework of the preimage methodology (Lozano-Pérez et al., 1984). In particular, Erdmann and Mason showed how to orient a planar object by a tray tilting device: first, the object is placed in a random unknown pose in the tray and, second, the tray is tilted at a sequence of angles to bring the object in a unique pose. In (Erdmann et al., 1993) the tray tilting idea was extended to polyhedra.

The pushing and tilting primitives can be formulated as parametrized functions that map orientations to orientations. The parameters of such a function are then the push direction and the tilt angle, respectively. By composing these functions one can orient a part without sensing. Eppstein (1990) presented a very general algorithm that takes a set of functions as input. Given these functions

the algorithm computes a shortest sequence of these functions that will orient a polygonal part. Goldberg (1993) introduced another primitive: squeezing with a parallel-jaw gripper. By making one jaw compliant in the tangential direction, the contacts with the part are effectively frictionless. Goldberg proved that for every n-sided polygonal part, a sequence of 'squeezes' can be computed in $O(n^2 \log n)$ time that will orient the part up to symmetry. The length of such a sequence is bounded by $O(n^2)$. Chen and Ierardi (1995) improved this bound to $O(n)$ and showed that the algorithm runs in $O(n^2)$. (Van der Stappen et al., 2000) presented improved bounds that depend on the *geometric eccentricity* (intuitively, how "fat" or "thin" a part is). Their analysis also applies to curved parts. Recently, Moll et al. (2002) introduced a new primitive for orienting micro-scale parts using a parallel jaw gripper. Moll et al. showed that any polygonal part can be oriented with a sequence of *rolling* motions, where the part is rolled between the two jaws. With this primitive the gripper does not need to be reoriented.

One of the first papers in palmar manipulation is (Salisbury, 1987). Salisbury suggested a new approach to manipulation in which the whole robot arm is used as opposed to just the fingertips. Paljug et al. (1994) investigated the problem of multi-arm manipulation. Paljug et al. presented a nonlinear feedback scheme for simultaneous control of the trajectory of the object being manipulated as well as the contact conditions. Erdmann (1998) showed how to manipulate a known object with two palms. He also presented methods for determining the contact modes of each palm: rolling, sliding and holding the object. Zumel (1997) described a palmar system like the one shown in figure 1.1(b), but without tactile sensors. Zumel derived sufficient conditions for orienting known polygonal parts with these palms. She also showed that an orienting plan for a polygon can be computed in $O(N^2)$ and that the length is $O(N)$, where N is the number of stable edges of the polygon.

Another way to orient parts is to design a manipulator shape specifically for a given part. This approach was first considered for the Sony APOS system (Hitakawa, 1988). The design was done mainly by ad-hoc trial and error. Later, Moll and Erdmann (2002) explored a way to automate this process.

In recent years a lot of work has been done on programmable force fields to orient parts (Böhringer et al., 2000a, 1999; Kavraki, 1997; Reznik et al., 1999) The idea is that an abstract force field (implemented using e.g. MEMS actuator arrays) can be used to push the part into a certain orientation. Böhringer et al. used Goldberg's algorithm (1993) to define a sequence of 'squeeze fields' to orient a part. They also gave an example how programmable vector fields can be used to simultaneously sort different parts and orient them. Kavraki (1997) presented a vector field that induced two stable configurations for most parts. In 2000,

Böhringer et al. proved a long-standing conjecture that the vector field proposed in (Böhringer et al., 1996) is a universal feeder/orienter device, i.e., it induces a *unique* stable configuration for most parts. Recently, Sudsang and Kavraki (2001) introduced another vector field that has the universal feeder property.

2.3 Grasping

The problem of grasping has been widely studied. This section will not try to give a complete overview of the results in this area, but instead just mention some of the work that is most important to our problem. Much of the grasp research focuses on computing grasps that establish *force-closure* (the ability to resist external forces) and *form-closure* (a kinematic constraint condition that prevents all motion). Important work includes (Salisbury, 1982), (Cutkosky, 1985), (Fearing, 1984), (Kerr and Roth, 1986), (Mishra et al., 1987), (Montana, 1988), (Nguyen, 1988), (Trinkle et al., 1988), (Hong et al., 1990), (Markenscoff et al., 1990), and (Ponce et al., 1997). For an overview of grasp synthesis algorithms see e.g. (Shimoga, 1996).

To grasp an object one needs to understand the kinematics of contact. Independently, Montana (1988) and Cai and Roth (1986, 1987) derived the relationship between the relative motion of two objects and the motion of their contact point. In (Montana, 1995) these results were extended to multi-fingered manipulation.

Sudsang et al. (2000) looked at the problem of manipulating three-dimensional objects with a reconfigurable gripper. The gripper consisted of two horizontal plates, of which the top one had a regular grid of actuated pins. They presented a planner that computed a sequence of pin configurations that brought an object from one configuration to another using so-called immobility regions. For each (intermediate) configuration only three pins were needed. Plans were restricted to ones where the object maintains the same set of contact points with the bottom plate. Rao et al. (1994, 1995) showed how to reorient a polyhedral object with *pivoting grasps*: the object was grasped with two hard finger contacts so that it pivoted under gravity when lifted. Often only one pivot grasp was sufficient to bring the object from one stable pose to another (provided the friction coefficient was large enough).

Trinkle and colleagues (Trinkle et al., 1993; Trinkle and Hunter, 1991; Trinkle and Paul, 1990; Trinkle et al., 1988) investigated the problem of dexterous manipulation with frictionless contact. They analyzed the problem of lifting and manipulating an object with enveloping grasps. Kao and Cutkosky (1992) and Yoshikawa et al. (1993) did not assume frictionless contacts. Whereas Kao and Cutkosky focused on modeling sliding contact with compliance, Yoshikawa et al.

showed how to regrasp an object using quasistatic slip motion. Nagata et al. (1993) described a method of repeatedly regrasping an object to build up a model of its shape.

Grupen and Coelho Jr. (1993; 1996) proposed an on-line grasp synthesis method for a robot hand equipped with sensors. The controllers of the hand used sensor data to refine a grasp. Teichmann and Mishra (2000) presented an algorithm that determines a good grasp for an unknown object using a parallel-jaw gripper equipped with light beam sensors. This paper also presented a tight integration of sensing and manipulation. Interestingly, the object is not disturbed until good grasp points are found.

Erdmann (1995) proposed a method for automatically designing sensors from the specification of the robot's task. Erdmann gives the example of grasping an ellipse. By sensing some aspect of the local geometry at the contact points, it is possible to define a feedback loop that guides the fingers toward a stable grasp. Recently, Jia (2000) extended these results and showed how to achieve an antipodal grasp of a curved planar object with two fingers. By rolling the fingers around the object the pose of the object is determined and then the fingers are rolled to two antipodal points.

2.4 Shape and Pose Recognition

The problem of shape and pose recognition can be stated as follows: suppose we have a known set of objects, how can we recognize one of the objects if it is in an unknown pose? For an infinite set of objects the problem is often phrased as: suppose we have a class of parametrized shapes, can we establish the parameters for an object from that class in an unknown pose? Schneiter and Sheridan (1990) and Ellis (1992) developed methods for determining sensor paths to solve the first problem. In Siegel (1991) a different approach is taken: the pose of an object is determined by using an enveloping grasp. This method uses only joint angle and torque sensing.

Grimson and Lozano-Pérez (1984) used measurements of positions and surface normals to recognize and locate objects from among a set of known polyhedral objects. They phrased the recognition problem as a constraint satisfaction problem using an interpretation tree. Interpretation trees were introduced by Gaston and Lozano-Pérez (1984) as a way to match sets of contact points with edges of an object.

Kang and Goldberg (1995) used a Bayesian approach to recognizing arbitrary planar parts from a known set. Their approach consists of randomly grasping a part with a parallel-jaw gripper. Each grasp returns a diameter measurement,

which can be used to update a probability for each part in the set of known parts. Using a statistical measure of similarity it is possible to predict the expected number of grasps to recognize a part.

Jia and Erdmann (1996) proposed another 'probing-style' solution: they determined possible poses for polygons from a finite set of possible poses. One can think of this finite set as the stable poses (for some sense of stable). One method determines the pose by bounding the polygon by supporting lines. The second method they propose is to sense by point sampling. They prove that solving this problem is *NP*-complete and present a polynomial time approximation algorithm.

Keren et al. (1998) proposed a method for recognizing three-dimensional objects using curve invariants. This idea was motivated by the fact that tactile sensor data often takes the form of a curve on the object. They apply their method to geometric primitives like spheres and cylinders.

Jia and Erdmann (1999) investigated the problem of determining not only the pose, but also the motion of a known object. The motion of the object is induced by having a robotic finger push the object. By tracking the contact point on the finger, they were able to recover the pose and motion using nonlinear observer theory.

2.5 Tactile Shape Reconstruction

With tactile exploration the goal is to build up an accurate model of the shape of an unknown object. One early paper by Goldberg and Bajcsy (1984) described a system requiring very little information to reconstruct an unknown shape. The system consisted of a cylindrical finger covered with 133 tactile elements. The finger could translate and tap different parts of an object.

Often the unknown shape is assumed to be a member of a parametrized class of shapes, so one could argue that this is really just shape recognition. Nevertheless, with some parametrized shape models, a large variety of shapes can still be characterized. In (Fearing, 1990), for instance, results are given for recovering generalized cylinders. Allen and Roberts (1989) model objects as superquadrics. Roberts (1990) proposed a tactile exploration method for polyhedra. In (Chen et al., 1996) tactile data are fit to a general quadratic form. Finally, Liu and Hasegawa (2001) use a network of triangular B-spline patches.

Allen (1988) presents a tight integration of vision and tactile sensing. The vision processing provides an estimate of the shape of areas of interest, which are then further explored by a tactile sensor. Allen presents a procedure for robustly integrating noisy visual and tactile data into 3D surface and feature primitives.

Allen and Michelman (1990) presented methods for exploring shapes in three stages, from coarse to fine: grasping by containment, planar surface exploring and surface contour following. Montana (1988) described a method to estimate curvature based on a number of probes. Montana also presented a control law for contour following. Charlebois et al. (1996, 1997) introduced two different tactile exploration methods. The first method is based on rolling a finger around the object to estimate the curvature using Montana's contact equations. Charlebois et al. analyze the sensitivity of this method to noise. With the second method a B-spline surface is fitted to the contact points and normals obtained by sliding multiple fingers along an unknown object.

Marigo et al. (1997) showed how to manipulate a known polyhedral part by rolling it between the two palms of a parallel-jaw gripper. Bicchi et al. (1999) extended these results to tactile exploration of unknown objects with a parallel-jaw gripper equipped with tactile sensors. The two palms of the gripper roll the object without slipping and track the contact points. Using tools from regularization theory they produce spline-like models that best fit the sensor data. The work by Bicchi and colleagues is different from most other work on tactile shape reconstruction in that the object being sensed is not immobilized. With our approach the object is not immobilized either, but whereas Bicchi and colleagues assumed pure rolling we assume pure sliding.

A different approach is taken by Kaneko and Tsuji (2001), who try to recover the shape by pulling a finger over the surface. With this finger they can also probe concavities. In (Okamura and Cutkosky, 1999; Okamura et al., 1999; Okamura and Cutkosky, 2001) the emphasis is on detecting fine surface features such as bumps and ridges. Sensing is done by rolling a finger around the object. (Okamura et al., 1999) show how one can measure friction by dragging a block over a surface at different velocities, measure the forces and solve for the unknowns. This work builds forth on previous work by Cutkosky and Hyde (1993), who propose an event-driven approach to dextrous manipulation. During manipulation of an object there are several events that can be detected with tactile sensing. Examples of such events are: making and breaking contact, slipping, change in friction, texture, stiffness, etc. Based on these events it is possible to infer some of the object's properties (such as its shape and mass distribution) and adjust the grasp.

Much of our work builds forth on (Erdmann, 1999). There, the shape of planar objects is recognized by three palms; two palms are at a fixed angle, the third palm can translate compliantly, ensuring that the object touches all three palms. Erdmann derives the shape of an unknown object with an unknown motion as a function of the sensor values. In our work we no longer assume that the motion

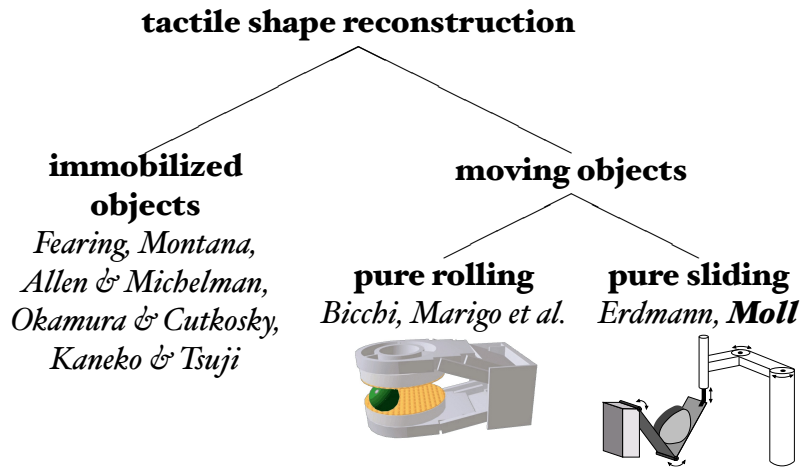


Figure 2.1: Related work in tactile shape reconstruction.

of the object is completely arbitrary. Instead, we model the dynamics of the object as it is manipulated by the palms. Only gravity and the contact forces are acting on the object. As a result we can recover the shape with fewer sensors. By modeling the dynamics we need one palm less in 2D. In the 3D case, instead of six palms, we need only three. Figure 2.1 shows where our research fits within the field of tactile shape reconstruction. In the long term we plan to develop a unified framework for reconstructing the shape and motion of unknown object with varying contact modes.

2.6 Tactile Sensor Design

Despite the large body of work in tactile sensing and haptics, making reliable and accurate tactile sensors has proven to be very hard. Many different designs have been proposed. This section will mention just a few. For an overview of sensing technologies, see e.g. (Lee, 2000), (Howe and Cutkosky, 1992) and (Nicholls and Lee, 1989). Fearing and Binford (1988) describe a cylindrical tactile sensor to determine the curvature of convex unknown shapes. Russell (1992) introduces so-called whisker sensors: curved rods, whose deflection is measured with a potentiometer. Kaneko and Tsuji (2001) have developed planning algorithms to recover the shape of an object with such whiskers. Speeter (1990) describes a tactile sensing system consisting of up to 16 arrays of 256 tactile elements that can be accessed in parallel. He discusses the implementation issues involved

with using these sensors with the Utah/MIT Hand. The underlying tactile technology is based on force sensing resistors from Interlink Electronics. Choi et al. (1998) present a different design for tactile sensors for multifingered robots based on capacitive tactile elements. They compare their experimental results with Montana's contact equations (Montana, 1988).

Ando and Shinoda (1995) describe a tactile sensor based on ultrasonic emission. Their system consists of a flexible spherical fingertip and a sound sensor at the center of the fingertip. Contact points act as emission sources and the sensor works as a direction-sensitive, wideband acoustic emission transducer. Although the error in the position estimate of the contacts can be as large as a few millimeters, with this sensor one can distinguish multiple contacts, which is impossible with most other sensor technologies.

There is also a fair amount of research on compliant tactile sensors. A compliant tactile sensor deforms to take on the local object shape. An obvious advantage of this approach is that instantaneously the sensor recovers an entire patch of an object as opposed to just one contact point. One idea is to use an optical waveguide to create an image of the pressure distribution over the sensor surface (Maekawa et al., 1993, 1997; Kinoshita et al., 1999). The light of an internal light source reflects on the inside of the sensor surface. Deformation causes the light to refract differently, which is detected by a camera inside the sensor. It is possible to infer the contact points from the camera image. Hristu et al. (2000) use a slightly different approach. Their deformable sensor surface is painted on the inside with a pattern of dots. From a camera image of these dots it is possible to infer the deformation of the surface. (Shinoda and Oasa, 2000) describe an elastic skin in which many sensing elements are embedded. These elements are resonator chips whose frequency reflects the stress around the chip. The resonant frequency is read out by a ground coil under the skin. Finally, Teramoto and Watanabe (2000) combine a deformable 'skin' with an acoustic sensor array. This sensor array uses acoustic transceivers to measure the shape of the skin (and the shape of the object touching the skin) from the inside.

In our own experiments we are relying on off-the-shelf components. The tactile sensors are touchpads from Interlink Electronics (http://www.interlinkelec.com), as found on many notebooks. Most touchpads use capacitive technology, but the ones we are using are based on force-sensing resistors. These touchpads return the centroid of contact, have a resolution of approximately 0.1mm and report readings at a frequency of 100Hz. This technology has already successfully been used before as a tactile sensor (Mingrino et al., 1994; Liu et al., 1995; Jockusch et al., 1997).

Chapter 3

Quasistatic Shape Reconstruction

In this chapter we will present a quasistatic method for reconstructing the local shape of an unknown smooth convex object. By 'local shape' we mean the shape of the object at the contact points. The object is placed between the two palms, and we can vary the angles between the palms and the world frame. The object will move as a result of the contact forces and gravity acting on it. We say that the object is in *force/torque balance* if and only if all forces and torques acting on the object add up to 0. Below, we will show that if we assume that the object is always in force/torque balance and if there is no friction between the object and the palms, then we can reconstruct the local shape with two palms.

Figure 3.1 shows the two inputs and the two sensor outputs. The inputs are

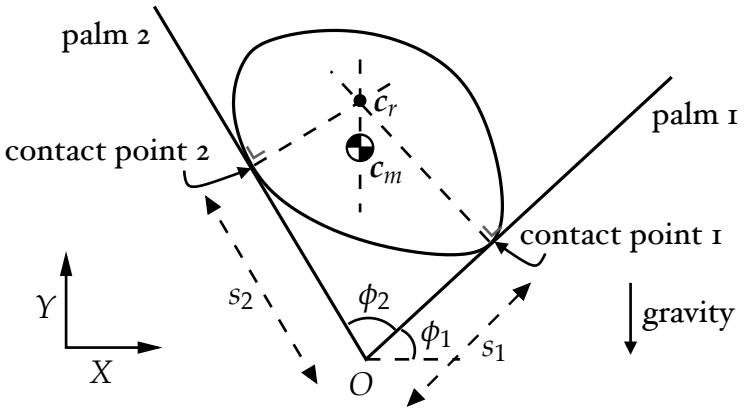


Figure 3.1: Inputs and outputs of the system formed by the palms and the object. Input values are ϕ_1 and ϕ_2, output values are the contact point locations s_1 and s_2. The contact normals intersect at the center of rotation (c_r), which lies on the vertical line through the center of mass (c_m).

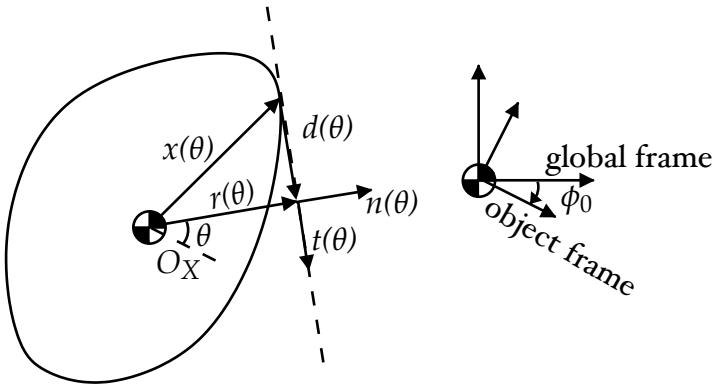


Figure 3.2: The contact support function $(r(\theta), d(\theta))$ and the object frame. O_X denotes the X-axis of the object frame.

ϕ_1, the angle between palm 1 and the X-axis of the global frame, and ϕ_2, the angle between palm 1 and 2. The tactile sensor elements return the contact points s_1 and s_2 on palm 1 and 2, respectively. Gravity acts in the negative Y direction. If the object is at rest, there is force/torque balance. In that case, since we assume there is no friction, the lines through the normal forces at the contact points and gravity acting on the center of mass intersect at a common point. In other words, the sensor values tell us where the X-coordinate of the center of mass is in the global frame. Below we will show that this constraint on the position of the center of mass and the constraints induced by the sensor values will allow us to derive an expression for the curvature at the contact points. However, this expression depends on the initial position of the center of mass. We can search for this position with an initial pose observer that minimizes the error between what the curvature expression predicts and what the sensor values tell us.

3.1 Notation

Frames We assume that the object is smooth and convex. We also assume that the origin of the object frame is at the center of mass, and that the center of mass is in the interior of the object. For every angle θ there exists a unique point $\boldsymbol{x}(\theta)$ on the surface of the object such that the outward pointing normal $\boldsymbol{n}(\theta)$ at that point is $(\cos\theta, \sin\theta)^T$. This follows from the convexity and smoothness assumptions. Let the tangent $\boldsymbol{t}(\theta)$ be equal to $(\sin\theta, -\cos\theta)^T$ so that $[\boldsymbol{t}, \boldsymbol{n}]$ constitutes a right-handed frame. Figure 3.2 shows the basic idea. We can also

define right-handed frames at the contact points with respect to the palms:

$$\bar{\boldsymbol{n}}_1 = \big(-\sin\phi_1, \cos\phi_1\big)^T$$
$$\bar{\boldsymbol{t}}_1 = \big(\cos\phi_1, \sin\phi_1\big)^T$$

and

$$\bar{\boldsymbol{n}}_2 = \big(\sin(\phi_1+\phi_2), -\cos(\phi_1+\phi_2)\big)^T$$
$$\bar{\boldsymbol{t}}_2 = \big(-\cos(\phi_1+\phi_2), -\sin(\phi_1+\phi_2)\big)^T$$

Note that $\bar{\boldsymbol{n}}_1$ and $\bar{\boldsymbol{n}}_2$ point into the free space between the palms. Let ϕ_0 be the angle between the object frame and the global frame, such that a rotation matrix $R(\phi_0)$ maps a point from the object frame to the global frame:

$$R(\phi_0) = \begin{pmatrix} \cos\phi_0 & -\sin\phi_0 \\ \sin\phi_0 & \cos\phi_0 \end{pmatrix}$$

The object and palm frames are then related in the following way:

$$\begin{pmatrix} \bar{\boldsymbol{n}}_1 & \bar{\boldsymbol{t}}_1 \end{pmatrix} = -R(\phi_0)\begin{pmatrix} \boldsymbol{n}(\theta) & \boldsymbol{t}(\theta) \end{pmatrix}$$
$$\begin{pmatrix} \bar{\boldsymbol{n}}_2 & \bar{\boldsymbol{t}}_2 \end{pmatrix} = -R(\phi_0)\begin{pmatrix} \boldsymbol{n}(\theta+\phi_2-\pi) & \boldsymbol{t}(\theta+\phi_2-\pi) \end{pmatrix}$$

The different frames are shown in figure 3.3. From these relationships it follows that

$$\theta = \phi_1 - \phi_0 - \tfrac{\pi}{2} \tag{3.1}$$

Differentiation We will use ‘˙’ to represent differentiation with respect to time t and ‘′’ to represent differentiation with respect to a function’s parameter. So, for instance, $\dot{\boldsymbol{x}}(\theta) = \boldsymbol{x}'(\theta)\dot{\theta}$. Let $v(\theta) = \|\boldsymbol{x}'(\theta)\|$ be the *parameterization velocity* of the curve $\boldsymbol{x}$. We can write $v(\theta)$ as $-\boldsymbol{x}'(\theta)\cdot\boldsymbol{t}(\theta)$ and $\boldsymbol{x}'(\theta)$ as $-v(\theta)\boldsymbol{t}(\theta)$.

The *curvature* $\kappa(\theta)$ is defined as the turning rate of the curve $\boldsymbol{x}(\theta)$. For an arbitrary curve $\boldsymbol{x}$ the following equality holds (Spivak, 1999a):

$$\boldsymbol{t}'(\theta) = \kappa(\theta)v(\theta)\boldsymbol{n}(\theta) \tag{3.2}$$

In our case we have that $\boldsymbol{t}'(\theta) = \boldsymbol{n}(\theta)$ so it follows that the parameterization velocity $v(\theta)$ is equal to the *radius of curvature* $\frac{1}{\kappa(\theta)}$ at the point $\boldsymbol{x}(\theta)$.

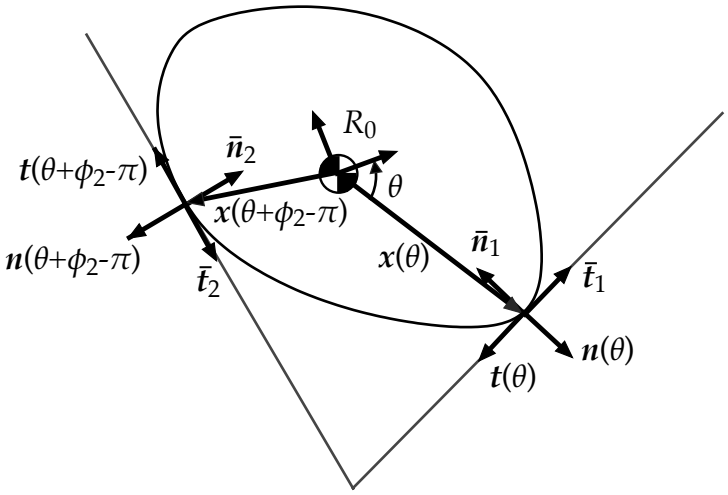


Figure 3.3: The different coordinate frames.

Support Functions The radius function is a useful tool for recovering the shape of an object (see e.g. (Santaló, 1976)). We define $r(\theta)$ to be the projection of the contact point $\boldsymbol{x}(\theta)$ onto the normal $\boldsymbol{n}(\theta)$:

$$r(\theta) = \boldsymbol{x}(\theta) \cdot \boldsymbol{n}(\theta)$$

This function is called a *radius function* or *support function.* For our shape recovery analysis it will be useful to define another function, $d(\theta)$, to be the projection of the contact point $\boldsymbol{x}(\theta)$ onto the tangent $\boldsymbol{t}(\theta)$:

$$d(\theta) = \boldsymbol{x}(\theta) \cdot \boldsymbol{t}(\theta)$$

We will refer to the pair $(r(\theta), d(\theta))$ as a *contact support function.* The goal is now to derive a solution for $\boldsymbol{x}(\theta)$ as we change the palm angles ϕ_1 and ϕ_2. As we will see below, it is actually sufficient to derive a solution for $r(\theta)$. We will derive a solution for $\dot{r}(t) = r'(\theta)\dot{\theta}$, which we can integrate to obtain a solution for $r(\theta)$.

One final bit of notation we need is a generalization of the contact support function, which we will define as a projection of the vector between the two contact points. We define *the generalized contact support function relative to contact point 1* as:

$$\tilde{r}_1(\theta) = \big(\boldsymbol{x}(\theta) - \boldsymbol{x}(\theta + \phi_2 - \pi)\big) \cdot \boldsymbol{n}(\theta) \tag{3.3}$$

$$\tilde{d}_1(\theta) = \big(\boldsymbol{x}(\theta) - \boldsymbol{x}(\theta + \phi_2 - \pi)\big) \cdot \boldsymbol{t}(\theta) \tag{3.4}$$

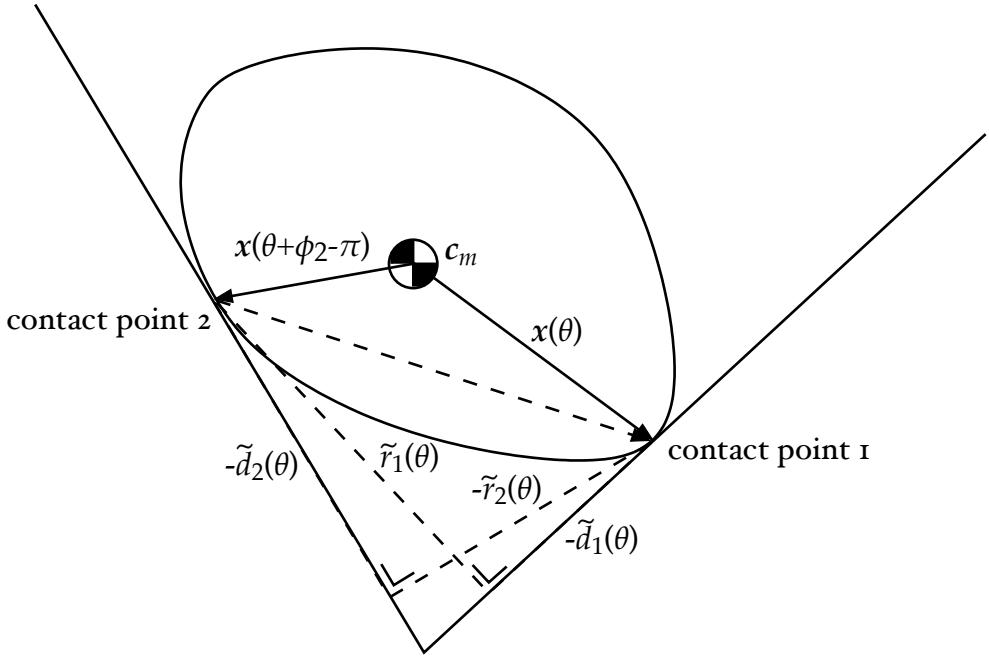


Figure 3.4: The generalized contact support functions.

Similarly, we can define *the generalized contact support function relative to contact point 2* as:

$$\tilde{r}_2(\theta) = \big(\boldsymbol{x}(\theta) - \boldsymbol{x}(\theta + \phi_2 - \pi)\big) \cdot \boldsymbol{n}(\theta + \phi_2 - \pi) \tag{3.5}$$

$$\tilde{d}_2(\theta) = \big(\boldsymbol{x}(\theta) - \boldsymbol{x}(\theta + \phi_2 - \pi)\big) \cdot \boldsymbol{t}(\theta + \phi_2 - \pi) \tag{3.6}$$

Below we drop the function arguments where it does not lead to confusion, and instead use subscripts '0', '1' and '2' to refer to the center of mass of the object, the contact point on palm 1, and the contact point on palm 2, respectively. So we will write e.g. $R_0\boldsymbol{n}_2$ for $R(\phi_0)\boldsymbol{n}(\theta + \phi_2 - \pi)$.

The generalized contact support functions have the property that they can be expressed directly in terms of the palm angles and sensor values (assuming the object is in two-point contact):

$$\begin{cases} \tilde{r}_1 = s_2 \sin\phi_2 \\ \tilde{d}_1 = s_2 \cos\phi_2 - s_1 \end{cases} \quad \text{or} \quad \begin{cases} \tilde{r}_2 = -s_1 \sin\phi_2 \\ \tilde{d}_2 = s_1 \cos\phi_2 - s_2 \end{cases} \tag{3.7}$$

These equalities can be obtained by inspection from figures 3.1 and 3.4. We can also obtain these equalities analytically. First, we write the constraints that

two-point contact induces as

$$s_1 \bar{\boldsymbol{t}}_1 = \boldsymbol{c}_m + R_0 \boldsymbol{x}_1 \tag{3.8}$$
$$-s_2 \bar{\boldsymbol{t}}_2 = \boldsymbol{c}_m + R_0 \boldsymbol{x}_2, \tag{3.9}$$

where $\boldsymbol{c}_m$ is the position of the center of mass. Next, we can eliminate $\boldsymbol{c}_m$ from these equations and write

$$R_0 \left(\boldsymbol{x}_1 - \boldsymbol{x}_2 \right) = s_1 \bar{\boldsymbol{t}}_1 + s_2 \bar{\boldsymbol{t}}_2 \tag{3.10}$$

The expressions in 3.7 then follow by computing the dot product on both sides of expression 3.10 with the palm normals and tangents:

$$\begin{aligned} \tilde{r}_1 &= (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{n}_1 = R_0 (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot R_0 \boldsymbol{n}_1 = -(s_1 \bar{\boldsymbol{t}}_1 + s_2 \bar{\boldsymbol{t}}_2) \cdot \bar{\boldsymbol{n}}_1 \\ &= s_2 \sin \phi_2 \end{aligned} \tag{3.11}$$
$$\tilde{d}_1 = (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{t}_1 = -(s_1 \bar{\boldsymbol{t}}_1 + s_2 \bar{\boldsymbol{t}}_2) \cdot \bar{\boldsymbol{t}}_1 = s_2 \cos \phi_2 - s_1 \tag{3.12}$$
$$\tilde{r}_2 = (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{n}_2 = -(s_1 \bar{\boldsymbol{t}}_1 + s_2 \bar{\boldsymbol{t}}_2) \cdot \bar{\boldsymbol{n}}_2 = -s_1 \sin \phi_2 \tag{3.13}$$
$$\tilde{d}_2 = (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{t}_2 = -(s_1 \bar{\boldsymbol{t}}_1 + s_2 \bar{\boldsymbol{t}}_2) \cdot \bar{\boldsymbol{t}}_2 = s_1 \cos \phi_2 - s_2 \tag{3.14}$$

Above we have shown that although the generalized contact support functions were defined in the object frame, we can also express them directly in terms of sensor values and palm angles. This is useful because it can be shown (Erdmann, 1999) that the radii of curvature at the contact points can be written in terms of the generalized contact support functions as

$$v_1 = -\frac{\tilde{r}_2' + \tilde{d}_2}{\sin \phi_2} \tag{3.15}$$
$$v_2 = -\frac{\tilde{r}_1' + \tilde{d}_1}{\sin \phi_2} \tag{3.16}$$

The derivation of these expressions is included in the appendix. Note that these expressions are *not* sufficient to observe the local shape, even though the generalized support functions are directly observable. To observe the shape we will also need an expression for the *time* derivative of the function parameters. This is the topic of section 3.3.

Equation 3.10 can also be rewritten in terms of the contact support function:

$$-(r_1 \bar{\boldsymbol{n}}_1 + d_1 \bar{\boldsymbol{t}}_1) + (r_2 \bar{\boldsymbol{n}}_2 + d_2 \bar{\boldsymbol{t}}_2) = s_1 \bar{\boldsymbol{t}}_1 + s_2 \bar{\boldsymbol{t}}_2 \tag{3.17}$$

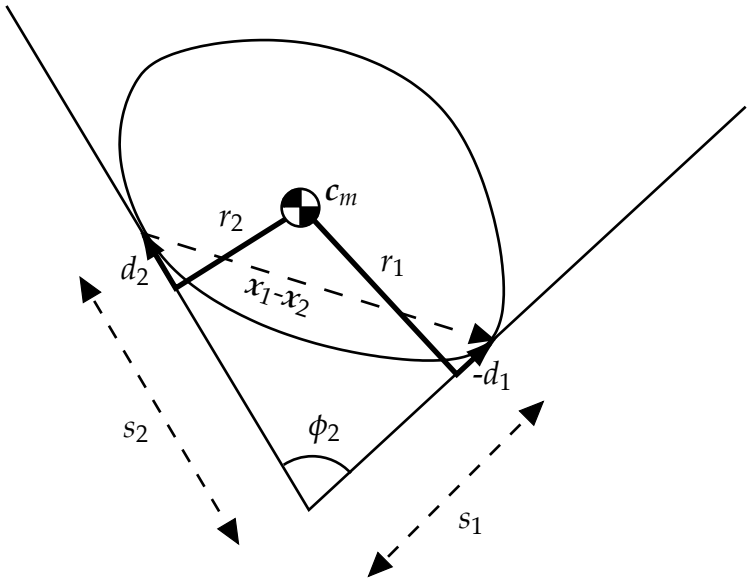


Figure 3.5: The dependencies between sensor values, the support function and the angle between the palms when the object makes two-point contact.

Solving this constraint for d_1 and d_2 we get:

$$d_1 = \frac{r_1 \cos\phi_2 + r_2}{\sin\phi_2} - s_1 \tag{3.18}$$

$$d_2 = -\frac{r_2 \cos\phi_2 + r_1}{\sin\phi_2} + s_2 \tag{3.19}$$

See also figure 3.5. Note that by construction $r'(\theta) = \boldsymbol{x}'(\theta) \cdot \boldsymbol{n}(\theta) + \boldsymbol{x}(\theta) \cdot \boldsymbol{t}(\theta) = -d(\theta)$. So a solution for $r(\theta)$ can be used in two ways to arrive at a solution for $d(\theta)$: (1) using the property $d(\theta) = -r'(\theta)$ of the radius function, or (2) using expressions 3.18 and 3.19. In other words, to recover the shape it is sufficient to reconstruct the radius function.

3.2 A Geometric Interpretation of Force/Torque Balance

Before deriving the equations for the local shape of the object as a function of the palm angles and sensor values, let us first consider the dynamics of the system formed by the palms and the object. We assume the dynamics are quasistatic, which means that all the forces and torques balance each other. A necessary and

sufficient condition for force/torque balance is that the lines through the contact points along the contact forces and the line through the center of mass along the gravity vector intersect at one point called the *center of rotation* (see figure 3.1). If we move the palms the object will instantaneously rotate around the center of rotation. The lines through the normals can be described by:

$$\ell_1 : q_1 \mapsto s_1 \bar{\boldsymbol{t}}_1 + q_1 \bar{\boldsymbol{n}}_1 \tag{3.20}$$
$$\ell_2 : q_2 \mapsto -s_2 \bar{\boldsymbol{t}}_2 + q_2 \bar{\boldsymbol{n}}_2 \tag{3.21}$$

These lines intersect if and only if

$$q_1 = \frac{s_2 - s_1 \cos\phi_2}{\sin\phi_2} \quad \text{and} \quad q_2 = \frac{s_1 - s_2 \cos\phi_2}{\sin\phi_2}.$$

Using the generalized contact support functions we can simplify this to $q_1 = -\tilde{d}_2 / \sin\phi_2$ and $q_2 = -\tilde{d}_1 / \sin\phi_2$. So we can write the following equations for the center of mass, $\boldsymbol{c}_m$, and the center or rotation, $\boldsymbol{c}_r$:

$$\begin{aligned} \boldsymbol{c}_m(\phi_0, \phi_1, \phi_2) &= s_1 \bar{\boldsymbol{t}}_1 - R_0 \boldsymbol{x}_1 \\ &= -\tilde{r}_2 \bar{\boldsymbol{t}}_1 / \sin\phi_2 - R_0 \boldsymbol{x}_1 \end{aligned} \tag{3.22}$$
$$\begin{aligned} \boldsymbol{c}_r(\phi_0, \phi_1, \phi_2) &= s_1 \bar{\boldsymbol{t}}_1 + q_1 \bar{\boldsymbol{n}}_1 \\ &= s_1 \bar{\boldsymbol{t}}_1 - \tilde{d}_2 \bar{\boldsymbol{n}}_1 / \sin\phi_2 \\ &= -(\tilde{r}_2 \bar{\boldsymbol{t}}_1 + \tilde{d}_2 \bar{\boldsymbol{n}}_1) / \sin\phi_2 \end{aligned} \tag{3.23}$$

In appendix A.1 it is shown that the partial derivatives of $\boldsymbol{c}_m$ and $\boldsymbol{c}_r$ can be written as

$$\frac{\partial \boldsymbol{c}_m}{\partial \phi_0} = -\frac{\tilde{d}_2 \bar{\boldsymbol{t}}_1}{\sin\phi_2} - \Big(\frac{\partial}{\partial \phi_0} \boldsymbol{R}_0\Big) \boldsymbol{x}_1, \tag{3.24}$$
$$\frac{\partial \boldsymbol{c}_r}{\partial \phi_0} = -\big(v_1 \bar{\boldsymbol{n}}_2 - v_2 \bar{\boldsymbol{n}}_1 - \tilde{r}_2 \bar{\boldsymbol{n}}_1 + \tilde{d}_2 \bar{\boldsymbol{t}}_1\big) / \sin\phi_2. \tag{3.25}$$

and that we can rewrite equation 3.24 as a function of the relative distance between the center of mass and the center of rotation:

$$\frac{\partial \boldsymbol{c}_m}{\partial \phi_0} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} (\boldsymbol{c}_m - \boldsymbol{c}_r). \tag{3.26}$$

This last equation should not surprise us. It says that instantaneously the center of mass rotates around the center of rotation, which is true by definition for *any* point on the object.

With the results above we can easily describe all the stable poses of an object. We define a *stable pose* as a local minimum of the potential energy function with respect to ϕ_0. The potential energy of an object in two-point contact with the palms is simply the Y coordinate of $\boldsymbol{c}_m$, which can be written as $\boldsymbol{c}_m \cdot \left(\begin{smallmatrix}0\\1\end{smallmatrix}\right)$. At a local minimum the first derivative with respect to ϕ_0 of this expression will be equal to 0. We can write this condition using equation 3.26 as $(\boldsymbol{c}_m - \boldsymbol{c}_r) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) = 0$. In other words, at the minima of the potential energy function the X coordinates of $\boldsymbol{c}_m$ and $\boldsymbol{c}_r$ have to be equal. Since we assume that the object is always in force/torque balance and, hence, at a minimum of the potential energy function, we can directly observe the X coordinate of the center of mass. Or, equivalently, we can directly observe the projection onto the X-axis of the vector from the center of mass to contact point 1 by using expressions 3.22 and 3.23:

$$\begin{aligned} & (\boldsymbol{c}_m - \boldsymbol{c}_r) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) = 0 \\ \Rightarrow \quad & \left(-\tilde{r}_2 \bar{\boldsymbol{t}}_1 / \sin\phi_2 - R_0 \boldsymbol{x}_1 + (\tilde{r}_2 \bar{\boldsymbol{t}}_1 + \tilde{d}_2 \bar{\boldsymbol{n}}_1) / \sin\phi_2 \right) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) = 0 \\ \Rightarrow \quad & (R_0 \boldsymbol{x}_1) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) = -\tilde{d}_2 \frac{\sin\phi_1}{\sin\phi_2} \end{aligned} \tag{3.27}$$

For a local minimum of the potential energy function the Y coordinate of the second partial derivative of $\boldsymbol{c}_m$ with respect to ϕ_0 has to be greater than 0, i.e., $\frac{\partial}{\partial \phi_0}\left((\boldsymbol{c}_m - \boldsymbol{c}_r) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right)\right) > 0$.

3.3 Recovering Shape

We can write the derivative $\dot{\boldsymbol{x}}(\theta)$ of the function $\boldsymbol{x}(\theta)$ that describes the shape as $\dot{\theta} v(\theta) \boldsymbol{t}(\theta)$. So if we can solve for $\dot{\theta}$, $v(\theta)$ and the initial conditions, we can find the shape by integrating $\dot{\boldsymbol{x}}$. Recall that θ is a curve parameter, so $\dot{\theta}$ is in general not equal to the rotational velocity of the object. We can obtain a simple relationship between $\dot{\theta}$ and $\dot{\phi}_0$ by differentiating equation 3.1:

$$\dot{\theta} = \dot{\phi}_1 - \dot{\phi}_0 \tag{3.28}$$

Since we know $\dot{\phi}_1$ solving for $\dot{\theta}$ is equivalent to solving for $\dot{\phi}_0$. In other words, if we can observe the curvature at the contact points and the rotational velocity of the object, we can recover the shape of an unknown object. By differentiating the generalized support functions with respect to time, we can rewrite expressions 3.15

and 3.16 for the radii of curvature at the contact points as

$$v_1 = -\frac{\dot{\tilde{r}}_2 + (\dot{\theta} + \dot{\phi}_2)\tilde{d}_2}{\dot{\theta}\sin\phi_2} \tag{3.29}$$

$$v_2 = -\frac{\dot{\tilde{r}}_1 + \dot{\theta}\tilde{d}_1}{(\dot{\theta} + \dot{\phi}_2)\sin\phi_2} \tag{3.30}$$

See appendix A.1 for details. So to observe the curvature at the contact points, we need to derive an expression for the rotational velocity of the object that depends only on palm angles, sensor values and their derivatives. Note that we can not observe the curvature at the contact point 1 or contact point 2 if $\dot{\theta} = 0$ or $\dot{\theta} + \dot{\phi}_2 = 0$, respectively. If we can not observe the curvature at a contact point, that point is instantaneously not moving in the object frame.

We can recover the rotational velocity by looking at the constraint the force/torque balance imposes on the motion of the object. Recall equation 3.27:

$$(R_0\boldsymbol{x}_1) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) = -\tilde{d}_2\frac{\sin\phi_1}{\sin\phi_2} \tag{3.31}$$

The left-hand side of this equation can be rewritten as

$$(R_0\boldsymbol{x}_1) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) = (R_0(r_1\boldsymbol{n}_1 + d_1\boldsymbol{t}_1)) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) \tag{3.32}$$

$$= r_1\sin\phi_1 - d_1\cos\phi_1 \tag{3.33}$$

This expression (implicitly) depends on the orientation of the object. In appendix A.1 it is shown how by differentiating this expression and the right-hand side of equation 3.31 we can obtain the following expression for the rotational velocity of the object:

$$\dot{\phi}_0 = \frac{\dot{\tilde{r}}_2\cos\phi_1 - \dot{\tilde{d}}_2\sin\phi_1 + \tilde{d}_2\dot{\phi}_2\frac{\sin\phi_{12}}{\sin\phi_2}}{r_1\sin\phi_{12} + (r_2 + \tilde{r}_2)\sin\phi_1 + \tilde{d}_2\cos\phi_1}, \tag{3.34}$$

where $\phi_{12} = \phi_1 + \phi_2$. This expression for $\dot{\phi}_0$ depends on the control inputs, the sensor values, their derivatives and the current values of radius function at the contact points. The system of differential equations describing the (sensed) shape and motion can be summarized as follows:

$$\dot{r}_1 = -d_1(\dot{\phi}_1 - \dot{\phi}_0) \tag{3.35}$$

$$\dot{r}_2 = -d_2(\dot{\phi}_{12} - \dot{\phi}_0) \tag{3.36}$$

$$\dot{\phi}_0 = \frac{\dot{\tilde{r}}_2\cos\phi_1 - \dot{\tilde{d}}_2\sin\phi_1 + \tilde{d}_2\dot{\phi}_2\frac{\sin\phi_{12}}{\sin\phi_2}}{r_1\sin\phi_{12} + (r_2 + \tilde{r}_2)\sin\phi_1 + \tilde{d}_2\cos\phi_1} \tag{3.37}$$

The first two equations describe the rate of change of the radius function at the contact points. The radius function uniquely defines the shape of the object. Equation 3.37 describes the motion of the object. If the object remains in contact with both palms, it has only one degree of freedom. So knowing just the rotational velocity is sufficient.

So far we have assumed that we have sensor data that is continuous and without any error. In practice sensors will be discrete, both in time and space, and there will also be errors. We would like to recover the shape of an unknown object in such a setting as well. There are two main directly observable error terms at each time step. First, one can check the error in the force/torque balance constraint (equation 3.31). Let that error be denoted by e_f. So at a given time t, $e_f(t)$ is equal to

$$e_f(t) = \left((R_0(\hat{\phi}_0)\hat{\boldsymbol{x}}_1) \cdot \left(\begin{smallmatrix}1\\0\end{smallmatrix}\right) + \tilde{d}_2 \frac{\sin\phi_1}{\sin\phi_2}\right), \tag{3.38}$$

where "$\hat{}$" denotes the estimated value of a variable. The second observable error is the error in the two-point contact constraint (equation 3.10). Let this error be denoted by $\boldsymbol{e}_c$. In other words,

$$\boldsymbol{e}_c(t) = \left(R_0(\hat{\phi}_0)(\hat{\boldsymbol{x}}_1 - \hat{\boldsymbol{x}}_2) - s_1\bar{\boldsymbol{t}}_1 - s_2\bar{\boldsymbol{t}}_2\right) \tag{3.39}$$

Our program searches for the initial conditions of our system by minimizing the sum of all directly observable errors.

In our current implementation we use a fourth-order Adams-Bashforth-Moulton predictor-corrector method to integrate equations 3.35–3.37. Let the state be defined as the vector $(r_1, r_2, \phi_0)^T$. In the prediction step the state at time $t-1$ and derivatives at time $t-3$, $t-2$, and $t-1$ are used to predict the state at time t. In the correction step, we use the state estimate at $t-1$ and derivatives at $t-3, \ldots, t$ to refine the state estimate at time t. The correction step is repeated until the relative change in the estimate is below a certain threshold or if a maximum number of iterations has been reached. This high-order method tends to filter out most of the noise and numerical errors. In our simulation results hardly any error accumulates during integration (see section 3.4).

3.4 Simulation Results

The results in this section are based on numerical simulation in Matlab. We generate random test shapes in the following way. We pick three numbers q_1, q_2, and q_3 independently and uniformly random from the interval $[0.1, 2.5]$. A random shape is generated by computing a closed spline interpolation of the

points with polar coordinates $(0, q_1)$, $(\frac{2}{3}\pi, q_2)$ and $(\frac{4}{3}\pi, q_3)$. Figure 3.6 shows an example of the shape reconstruction process. 270 measurements were used to reconstruct the shape. In each frame the part of the shape that has been observed up to that point in time is shown. Also drawn are the contact points, the center of mass, and the palms. Notice how the (observed) shape sometimes intersects the palms. This means that there is a conflict between the currently observed shape and the previously observed shape, which could potentially be used to guide the search for initial conditions. The motion of the palms is open-loop. Initially palm 1 and palm 2 are nearly horizontal; the object is squeezed (but without friction!) between the palms. The motion of the palms can roughly be described as a sequence of squeeze-and-rotate motions and motions where one of the palms stays put and the other palm opens up. Notice how in the penultimate frame the simulator misgauges the shape, but has recovered in the last frame.

In figure 3.7 the differences are shown between the reconstructed and actual shape and motion of the object. We defined the object frame to be located at the center of mass. From the force/torque balance constraint we can solve for this position. The relative orientation of the object frame with respect to the world frame is arbitrary. We just assume that the initial orientation is 0° and use SVD to align the orientation of the generating shape and the observed shape (Golub and Loan, 1996, p. 601).

One can not directly observe the errors in $\dot{\phi}_0$ and ϕ_0, but one *can* observe the error in the X-coordinate of the center of mass and the error in the two-point contact constraint. These errors are shown in figure 3.8. Note that the big errors in error plot 3.8(d) occur at the same times as when the rotational speed of the object was misgauged. This suggests that our system could at least detect where the observed shape will be wrong. It is possible that the system could even detect that such a situation is approaching and maybe even prevent it by changing the motion of the palms. Also, the error in the norm of the contact point vector is very small, but does not appear to be completely random, suggesting that there is still room for improvement in the integration step.

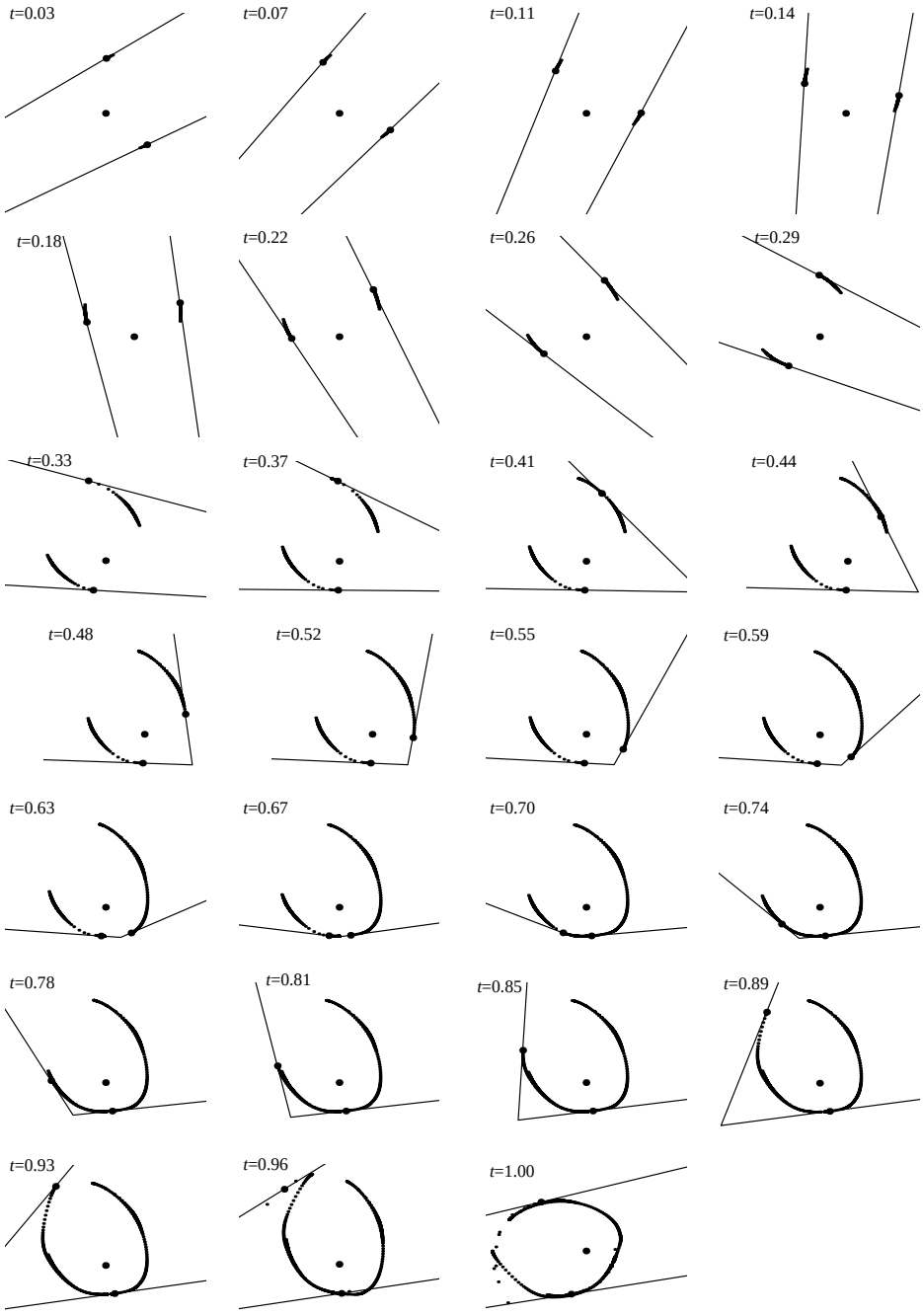
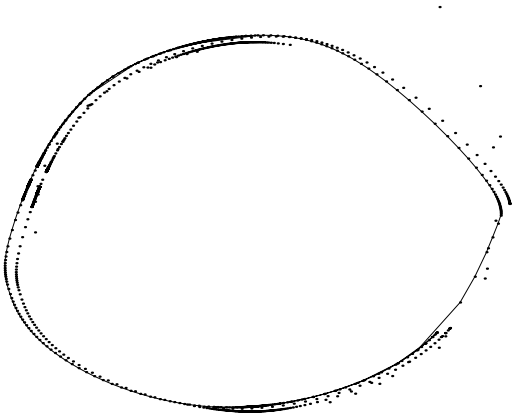
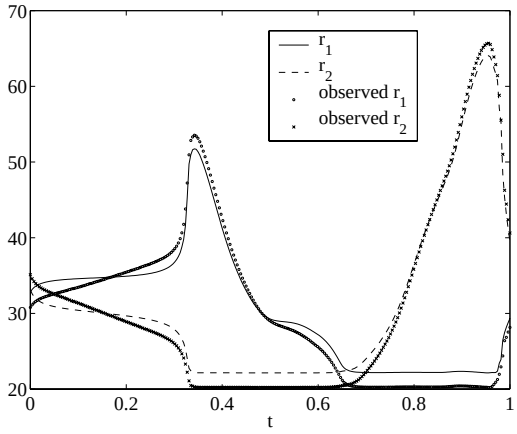


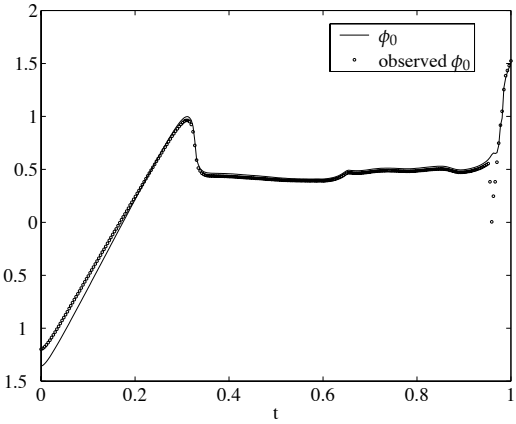
Figure 3.6: The frames show the reconstructed shape after 10, 20,. . . ,270 measurements. The three large dots indicate the center of mass and the contact points at each time, the smaller dots show the part of the shape that has been reconstructed at that time.



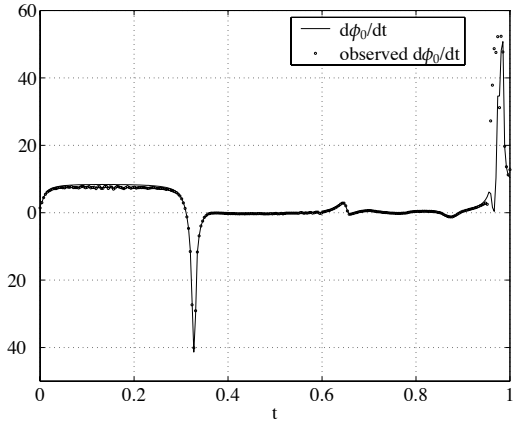
(a) The actual shape and the observed shape.



(b) The actual and observed values of the radius function (in mm).

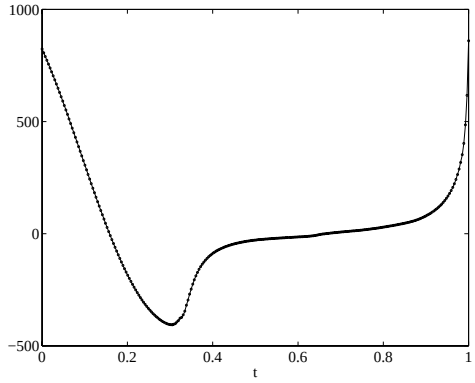


(c) The actual and observed orientation of the object.

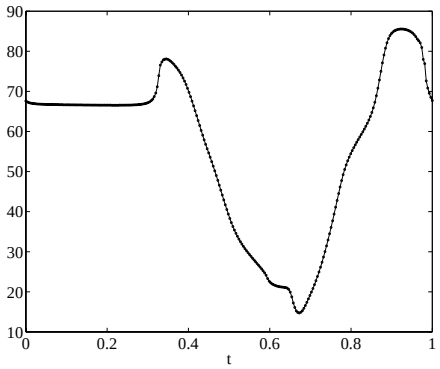


(d) The actual and observed rotational velocity of the object.

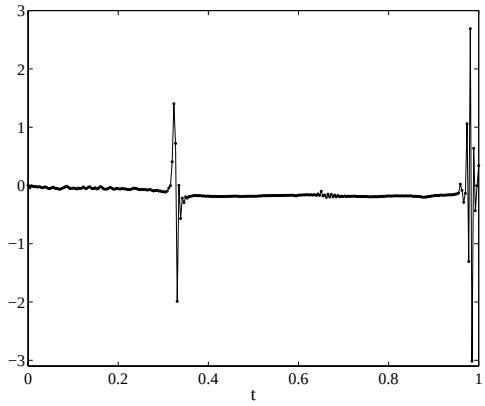
Figure 3.7: The differences between the actual and observed shape.



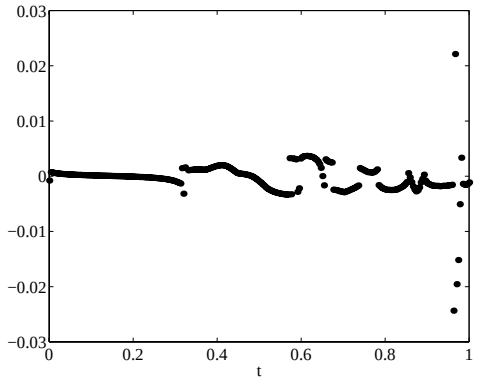
(a) The real and observed X-coordinate of the center of mass (in mm)



(b) The real and observed norm of the vector between the contact points (in mm)

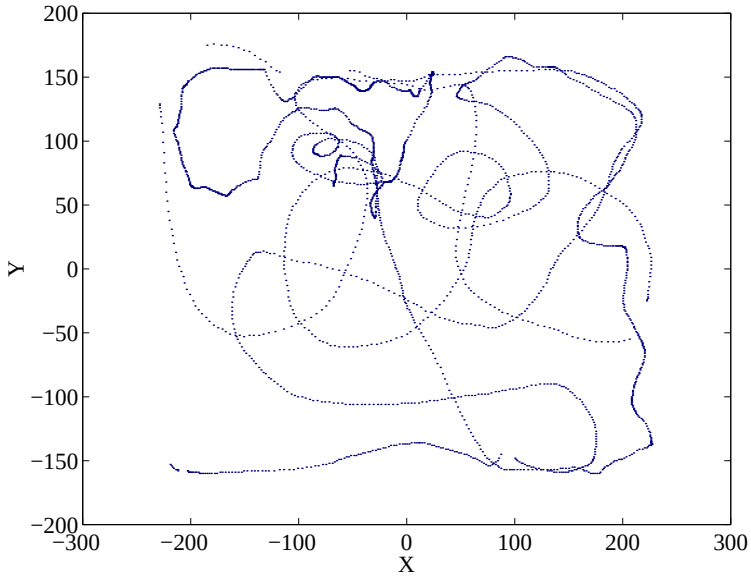


(c) The error in the X-coordinate of the center of mass (in mm)

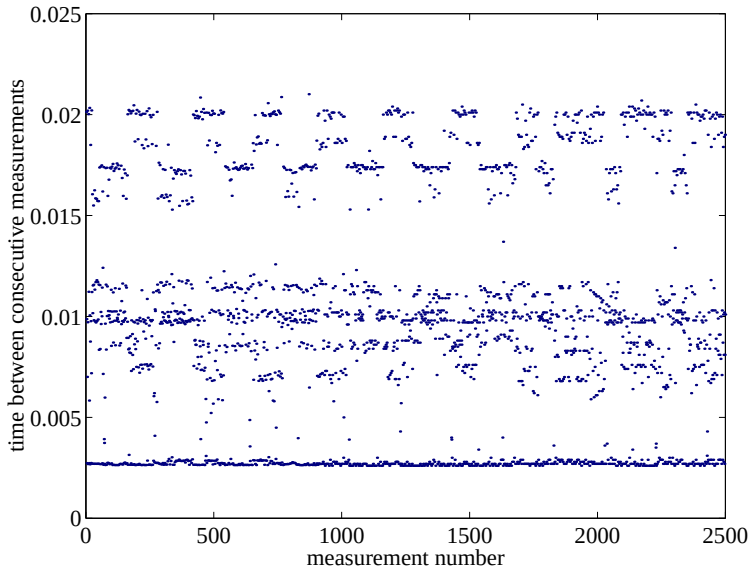


(d) The error in the norm of the vector between the contact points (in mm)

Figure 3.8: The observable error for the reconstructed shape.



(a) The contact point of a marble being rolled on a touchpad. X and Y are measured in 'tactels.'



(b) The speed at which measurements are reported. The average time between measurements is 0.010 seconds, the standard deviation is 0.0050.

Figure 3.9: Resolution and sensing frequency of the VersaPad.

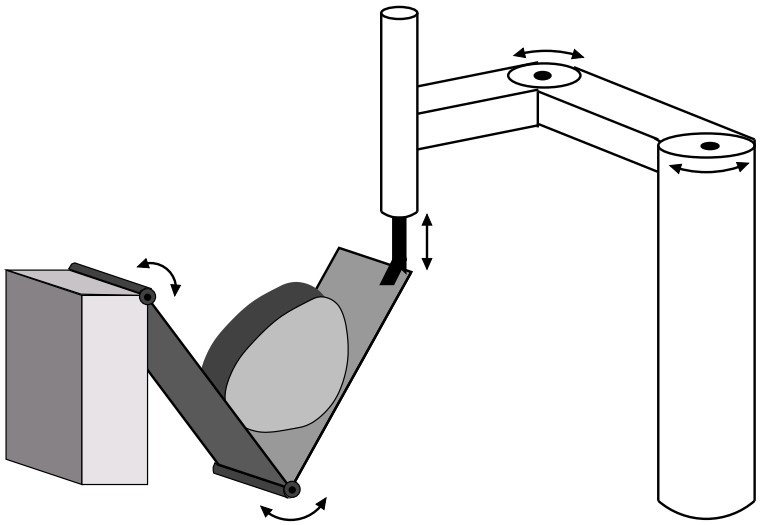


Figure 3.10: Setup of the palms.

3.5 Experimental Results

The tactile sensors we are using are touchpads made by Interlink Electronics (http://www.interlinkelec.com). These touchpads are most commonly used in notebook computers. They use so-called force sensing resistors to measure the location and the applied pressure at the contact point. One of the advantages of this technology, according to Interlink, is that it does not suffer as much from electrostatic contamination as capacitance-based touchpads. If there is more than one contact point, the pad returns the centroid. The physical pad has a resolution of 1000 counts per inch (CPI) in the X and Y direction, but the firmware limits the resolution to 200 CPI. It can report 128 pressure levels, but the pressure readings are not very reliable. In our experiments we only used the coordinates of the contact point and ignored the pressure data. The pad measures $55.5 \times 39.5\text{mm}^2$. Sensor data can be read out through a RS232 serial port connection.

Figure 3.9 shows the results of a simple test to establish the feasibility of the touchpad. The test consisted of rolling a marble around on the touchpad and tracking the contact point. Figure 3.9(a) shows the 'curve' traced out by the contact point. Figure 3.9(b) shows how fast we can get sensor readings from the touchpad. Notice how the times between measurements are roughly centered around 3 bands. This could be related to the way our driver polls the touchpad for data; further tweaking might increase the speed at which measurements are reported.

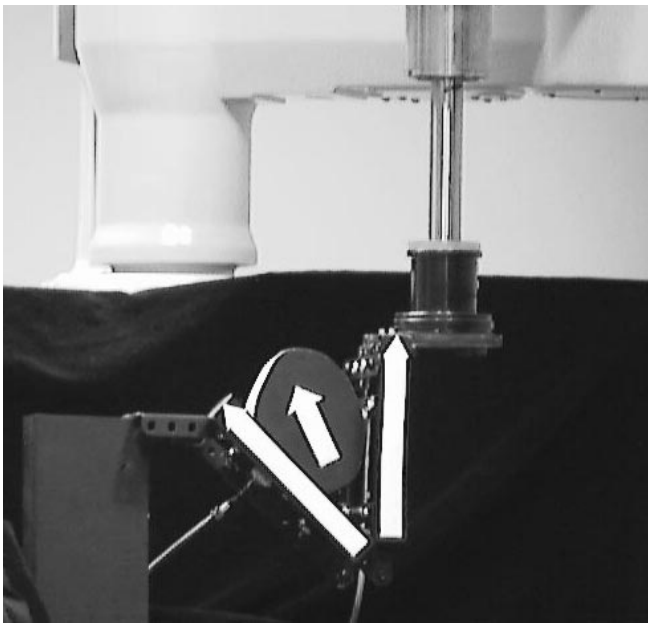
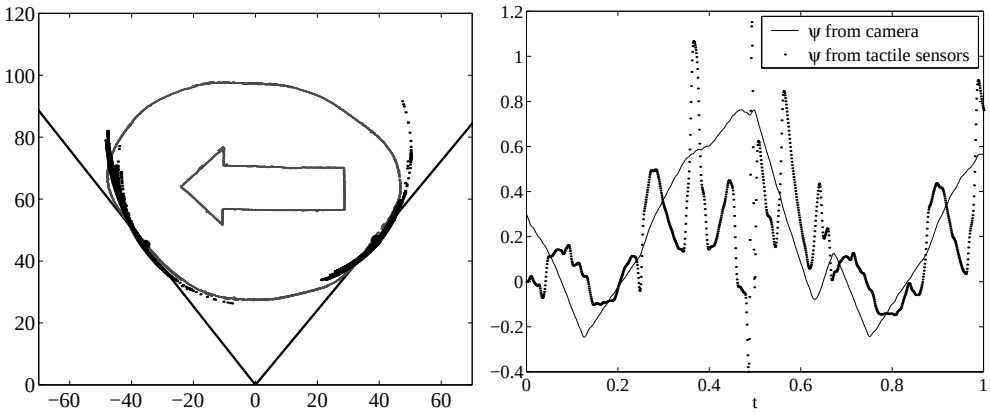


Figure 3.11: Experimental setup.

For the actual palms, we are using an Adept SCARA-type arm to control two planar surfaces connected with hinges. The Adept robot arm holds the endpoint of one palm. The endpoint of the other palm is attached to a fixed base. Figure 3.10 shows the basic idea (not to scale). The touchpads are mounted on the two surfaces and connected to a PC. It is important to realize that this experimental setup is just meant to be a proof of concept. Mechanically the sensing mechanism can be much simpler. More importantly, the palms do not need to be connected at all: the analysis only depends on the relative angle between the palms and the world. So in theory our proposed sensing strategies can also be applied to a robot hand equipped with tactile sensors. Our actual experimental setup is shown in figure 3.11. The white arrows on the object and the palms are tracked by an Adept vision system to establish 'ground truth', which can be compared with the shape and motion inferred from the tactile data.

Figures 3.12(a) and 3.12(b) show the reconstructed shape and motion, respectively. The observed motion is far from perfect, but the observed shape comes close to the actual shape. This seems to suggest that the system of differential equations 3.35–3.37 is fairly stable in this case, i.e., the errors in the motion did not cause the radius function to shoot off to infinity. The palms made back-and-forth motions to cover the shape several times. This means that we can prune those parts of the reconstructed shape that have been touched only once. For instance,



(a) Partially reconstructed shape

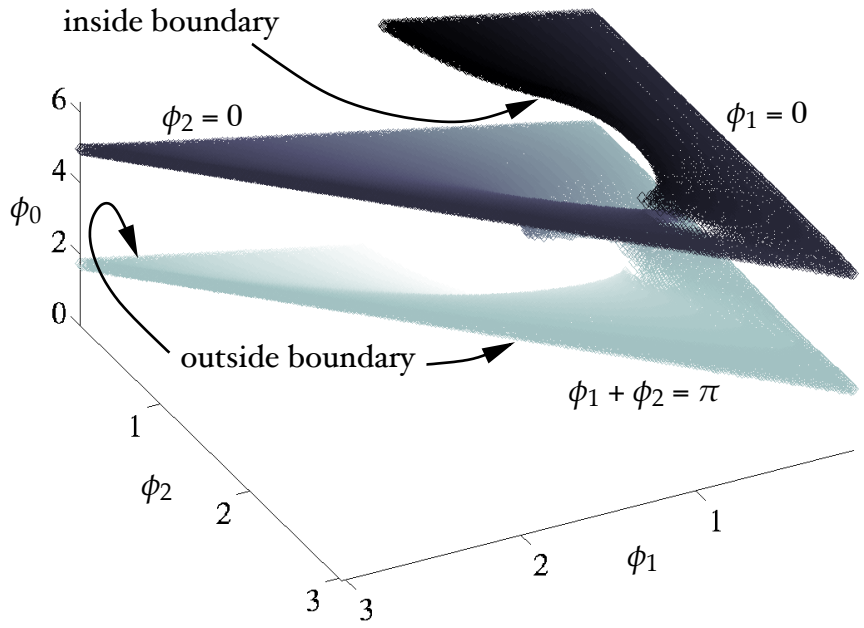
(b) Orientation measured by the vision system and the observed orientation

Figure 3.12: Experimental results.

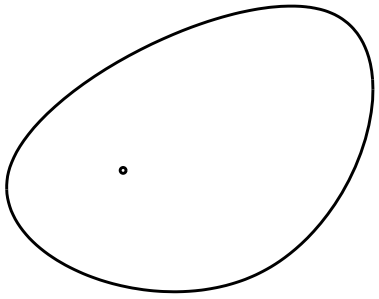
in figure 3.12(a) we can eliminate the sparse point distribution in the top right and bottom middle. To determine which parts to eliminate one can draw a curve interpolating the points $\big(t_i, r(\theta(t_i))\big), i = 1, \ldots$ The points we can eliminate are those for which $\big(t_i, r(\theta(t_i))\big)$ is the only intersection with the line $t = t_i$.

3.6 Global Observability

In the previous sections we showed that the curvature at the contact points and the motion of the object can be expressed as a function of the motion of the palms and the sensor values. By integrating out these expressions we can reconstruct a part of the shape. An important open problem is whether it is possible to reconstruct the *entire* shape. In this section we conjecture that, in general, it *is* possible. This is not immediately obvious. It could be, for instance, that the robot can only reconstruct one side of the shape and that it is impossible to reach the other side. To answer the question of global observability it will be useful to consider the motion of the center of mass and the center of rotation relative to each other. For a pose to be stable the object needs to be at a local minimum of the potential energy function. Recall from section 3.2 the two conditions that



(a) The stable pose surface in configuration space



(b) The generating object shape

Figure 3.13: Stable poses for a particular shape. Note that only one 2π period of ϕ_0 is shown in (a) and the surface extends from $-\infty$ to ∞ in the ϕ_0 direction, i.e., the inside and outside boundary do not meet.

have to hold for a stable pose:

$$(\boldsymbol{c}_m - \boldsymbol{c}_r) \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = 0, \tag{3.40}$$

$$\frac{\partial}{\partial \phi_0}\left((\boldsymbol{c}_m - \boldsymbol{c}_r) \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix}\right) = 0. \tag{3.41}$$

The stable poses induce a two-dimensional subset of the (ϕ_1, ϕ_2, ϕ_0)-configuration space. Figure 3.13 shows all the stable poses for a given shape. These stable configurations form a spiraling surface. From figure 3.13 it follows that for this particular shape it is indeed possible to reconstruct the entire shape, because there exists a path on the surface of stable configurations between any two stable configurations. We suspect that this is true in general. Although we do not have a complete proof for it, we will give some intuition for the following conjecture:

> For any smooth convex shape there exists a surface of stable configurations such that we can bring the object from any orientation to any other orientation by moving along a curve embedded in this surface.

We argue in favor of this conjecture by considering the boundaries of the stable configuration surface. Let us define the *inside boundary* as those configurations where both the first and second derivative with respect to ϕ_0 of the potential energy function vanish. Using expressions 3.22–3.27 we can write these two constraints as:

$$(\boldsymbol{c}_m - \boldsymbol{c}_r) \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix} = -\tilde{d}_2 \sin\phi_1 / \sin\phi_2 - \begin{pmatrix} \cos\phi_0 \\ -\sin\phi_0 \end{pmatrix} \cdot \boldsymbol{x}_1 = 0, \tag{3.42}$$

$$\frac{\partial}{\partial\phi_0}\left((\boldsymbol{c}_m - \boldsymbol{c}_r) \cdot \begin{pmatrix} 1 \\ 0 \end{pmatrix}\right) = \frac{v_1 \sin(\phi_1 + \phi_2) + (v_2 + \tilde{r}_2)\sin\phi_1}{\sin\phi_2} + \begin{pmatrix} \sin\phi_0 \\ \cos\phi_0 \end{pmatrix} \cdot \boldsymbol{x}_1 = 0. \tag{3.43}$$

The *outside boundary* of the stable configuration surface is determined by limits on the palm angles: $\phi_1, \phi_2 > 0$ and $\phi_1 + \phi_2 < \pi$. These limits can be geometrically interpreted as follows:

$\boldsymbol{\phi_1 = 0^+}$, $\boldsymbol{0 < \phi_2 < \pi}$: When $\phi_2 = \pi^-$, both palms are nearly horizontal, pointing in nearly opposite directions. In the limit, as ϕ_2 approaches π, $s_1 = s_2 = 0$, and the contact point is an extremum of the radius function (since the center of mass is right above the contact point and therefore $r'(\theta) = -d(\theta) = 0$). As ϕ_2 decreases, contact point 2 covers nearly half the shape. As ϕ_2 approaches 0, the palms form an antipodal grasp. The contact points are then at a minimum of the diameter function $D(\theta) \stackrel{\text{def}}{\equiv} r(\theta) + r(\theta - \pi)$.

$\boldsymbol{\phi_2 = 0^+}$, $\boldsymbol{0 < \phi_1 < \pi}$: When $\phi_1 = 0^+$, this boundary *usually* connects to the previous one. As ϕ_1 increases, the palms maintain an antipodal grasp, so the contact points do not change. As ϕ_1 approaches π, palm 1 and 2 both point to the left. Below we will describe when this boundary and the previous one do *not* connect. This case is where our argument breaks down.

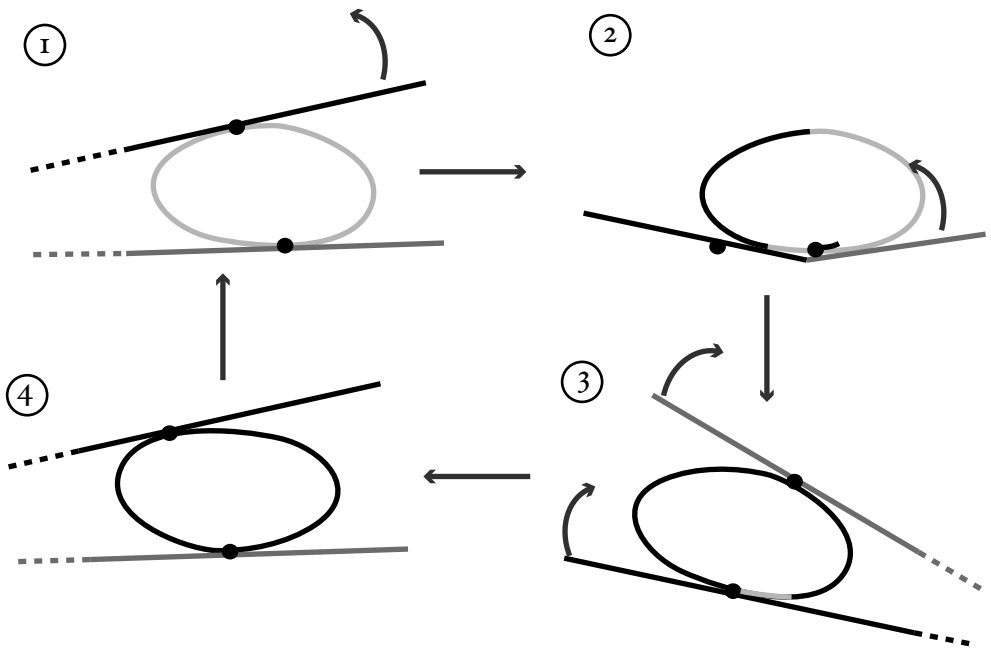


Figure 3.14: Plan for observing the entire shape of an unknown object.

$0 < \phi_1 < \pi$, $\phi_2 = \pi^- - \phi_1$: This case is symmetrical to the first one. Now contact point 1 covers nearly half the shape.

From this geometric interpretation it is clear that we can bring the object to any orientation by moving along these outside boundaries. More importantly, by following these boundaries we can reconstruct the entire shape. Figure 3.14 shows how a shape is reconstructed by palms that trace these boundaries. First palm 2 sweeps through its entire range of motion while palm 1 is nearly horizontal. Then palm 1 sweeps through its range. Finally both palms move simultaneously back to their original position. The object is now rotated roughly 180°. The part of the shape that has been reconstructed is drawn in black. We can repeat this plan to refine the estimate of the shape. However, these boundaries are themselves not part of the surface, so the question is whether there always exist stable configurations arbitrarily close to the outside boundary. The answer is "yes", *provided the inside and outside boundary do not meet*. Let a *generic smooth convex object* be defined as a smooth convex object in general position such that none of the singular cases described below apply. Below we will argue that for a generic smooth convex shape the inside boundary *never* (with probability 1) meets the outside boundary and, hence, there exists a path on the surface connecting any two orientations of the object. For each outside boundary condition we can

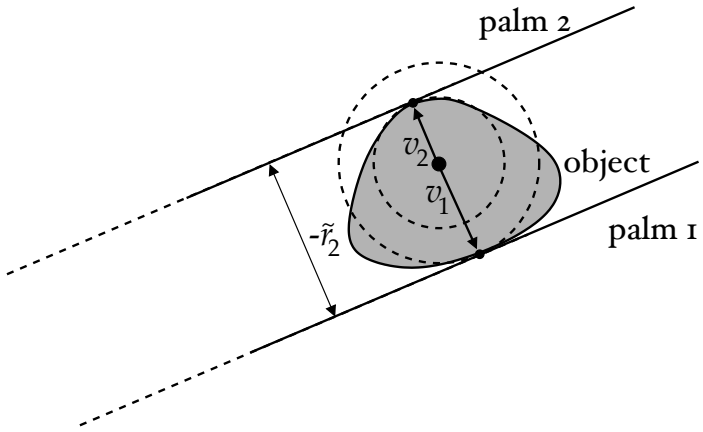


Figure 3.15: An antipodal grasp. For $\phi_2 = 0$ the inside and outside boundary meet if the sum of the radii of curvature of two antipodal points is equal to the distance between these points.

analyze what conditions must hold for the inside boundary to meet the outside boundary:

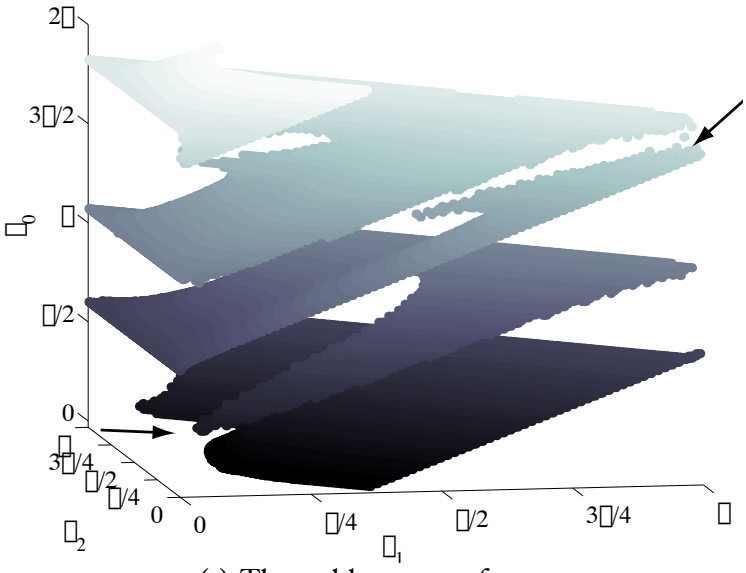
$\phi_1 = 0^+$: Without loss of generality, we can assume that $\phi_0 = 0$, in this case and below. If $\phi_1 = 0^+$, then equations 3.42 and 3.43 simplify to $x_1 \cdot \binom{1}{0} = 0$ and $x_1 \cdot \binom{0}{1} = -v_1$. In other words, contact point 1 is right below the center of mass. Furthermore, if we draw a circle with radius equal to v_1 and tangent to the contact point, its center coincides with the center of mass. For each point on the shape we can determine the center of the circle with radius v_1 and tangent to that point. The locus of these circle centers forms a curve. For a generic smooth object the center of mass is not on this curve.

$\phi_2 = 0^+$: Since the palms make an antipodal grasp, the possible contact points on the object are restricted to a finite set. Now for the inside boundary to meet the outside boundary, the limit of equation 3.43 as ϕ_2 goes to zero has to be equal to zero. So we have the following condition:

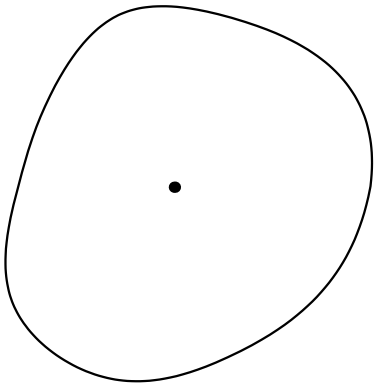
$$\lim_{\phi_2 \downarrow 0} \left((v_1 \sin(\phi_1 + \phi_2) + (v_2 + \tilde{r}_2) \sin \phi_1) / \sin \phi_2 + x_1 \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = 0. \tag{3.44}$$

If $\phi_1 > 0$ this limit only converges if

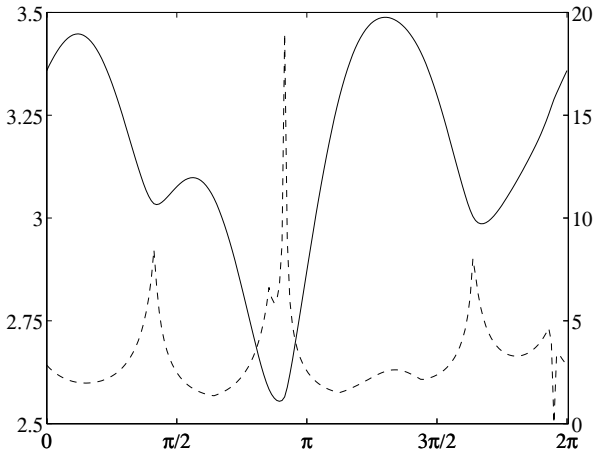
$$\lim_{\phi_2 \downarrow 0} \left(v_1 \sin(\phi_1 + \phi_2) + (v_2 + \tilde{r}_2) \sin \phi_1 \right) = 0. \tag{3.45}$$



(a) The stable pose surface.



(b) The generating shape.



(c) The radius function (solid line, left scale) and the curvature (dashed line, right scale)

Figure 3.16: A more complicated stable pose surface.

$-\tilde{r}_2$ will converge to the distance between the contact points. So expression 3.45 converges if the sum of the radii of curvature at the contact points is equal to the distance between the contact points. A geometric interpretation of this constraint is shown in figure 3.15. A generic smooth object does not have such a pair of antipodal points.

Now consider the case where both ϕ_1 and ϕ_2 converge to 0. (The case $\phi_1 \uparrow \pi$ and $\phi_2 \downarrow 0$ is analogous.) Let us write $\phi_1 = c\phi_2$, $c > 0$. Then equation 3.44 becomes

$$\lim_{\phi_2 \downarrow 0} \left((v_1 \sin((c+1)\phi_2) + (v_2 + \tilde{r}_2)\sin(c\phi_2)) / \sin\phi_2 + \boldsymbol{x}_1 \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right) = 0. \tag{3.46}$$

Using L'Hôpital's rule we can rewrite this as

$$v_1(c+1) + (v_2 + \tilde{r}_2)c + \boldsymbol{x}_1 \cdot \begin{pmatrix} 0 \\ 1 \end{pmatrix} = 0. \tag{3.47}$$

It is possible that this equation has a solution for $c > 0$ such that equation 3.42 is also satisfied. Figure 3.16 shows an example of this case. There are two places where the inside and outside boundary meet as indicated by the two arrows. Although this case seems to invalidate our argument, the surface is still connected and roughly has the same spiraling shape as before. This has been the case with all the shapes we generated.

$\phi_1 + \phi_2 = \pi$: This case is, as before, symmetrical to the first one.

Practical concerns For practical reasons it is undesirable to plan paths on the surface that are close to the boundary. First, we would need palms of infinite length for an antipodal grasp. We can get around this by removing the joint between the palms, thereby allowing the palms to move freely. For the analysis it is not essential that the palms are connected; the analysis just depends on relative orientations of the palms. The second reason that moving on the boundary of the stable configuration surface is undesirable is that for a wide angle between the palms, we are relying heavily on the assumption that there is no friction. Even the slightest amount of friction will throw off our estimate of the X-coordinate of the center of mass.

Multiple stable poses Figure 3.13 also shows that for almost all combinations or ϕ_1 and ϕ_2 there exist exactly two stable poses. However, it is possible that for a given ϕ_1 and ϕ_2 there are many stable poses. We can construct a shape with an arbitrary number of stable poses for a given palm configuration in the following way. Consider the arrangement of lines through the contact normals and the line through the center of mass (along the direction of gravity). We can rotate this arrangement around the center of mass and then translate along the line through the center of mass to create a new stable configuration. We pick

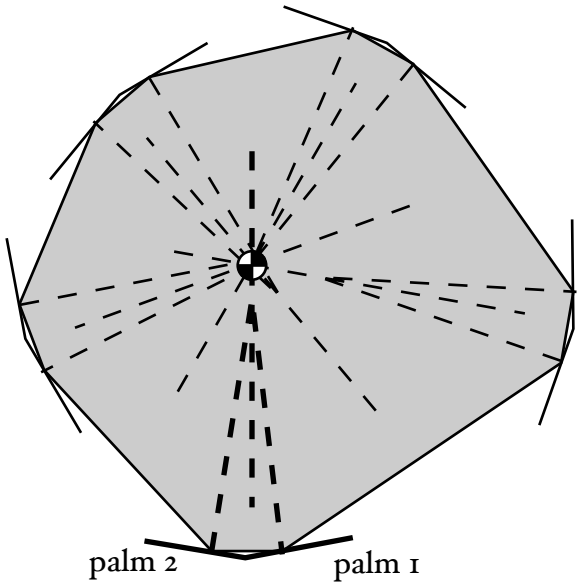


Figure 3.17: Many stable poses are possible for a given palm configuration that produce the same sensor readings.

the new contact points to be at the same distance from the intersection of the lines as in the original arrangement. This means that this new configuration produces the same sensor readings as well. We can repeat this process, picking a different amount of translation at each step to create an arbitrary number of stable configurations. We can create a smooth convex shape that has these stable poses in the following way. Consider the convex polygon that has the contact points of all these stable poses as vertices. If such a polygon does not exist we remove the poses that cause the concavities. The arrangement of lines described above corresponds to critical points of the potential energy function. To make sure that all the critical points are local minima we need to consider the second derivative of the potential energy function (see equation 3.43). For each contact point we can pick the radius of curvature to be arbitrarily large such that the second derivative is greater than 0. We can locally deform the polygon around each vertex such that at the contact point the radius of curvature is as desired and the shape remains convex. Figure 3.17 illustrates this geometric construction. Since the polygon induced by the contact points is convex, there exists a smooth convex shape with these stable poses.

3.7 Segmentation of Shape Reconstruction

It is possible that the object undergoes a discrete change in orientation as the palms move. This happens when the system reaches a boundary of the stable pose surface in the (ϕ_1, ϕ_2, ϕ_0)-configuration space. In terms of figure 3.13, the object "falls off" the stable pose surface. When this happens we need to restart our shape reconstruction process. Sudden changes in orientation give rise to two problems:

- How do we decide that the object is undergoing a discrete change in orientation? Since the sensor values are reported at a certain frequency, *every* measurement corresponds to a discrete change in orientation. More importantly, the sensor readings are likely to be noisy, so one spike in the measurements doesn't necessarily mean that a jump in the orientation occurred.

- Given a set of reconstructed shape segments, how do we piece them together? We have to find a method for determining the relative positions of the different segments. Also, if some segments overlap, we need to consolidate the possibly conflicting values for the radius function.

For the first problem we can use the following approach. If the object undergoes a discrete change in orientation, numerical integration of $\dot{\phi}_0$ will be very unstable. The predictor-corrector method will generally not be able to recover and $\dot{\phi}_0$ shoots off to infinity. So we can trigger a restart of the observer if $\dot{\phi}_0$ exceeds a certain threshold. This means that the observer is also restarted if there is a large error in the sensor data. Often there will be a sequence of spikes if the object's motion approaches a singularity. To prevent the creation of many tiny segments, we therefore also set a limit on the minimum segment length. The search for the initial conditions at the start of each segment is seeded with the endpoint of the previous segment.

For the initial segment we only have to search for the vertical offset of the center of mass with respect to the center of rotation at the start of the segment, since we are free to choose a particular orientation. But for subsequent segments the relative orientation with respect to this initial segment matters. So now we have to search for *two* unknowns for each segment: the relative orientation and the offset at the start of each segment.

Implicit in these segments there is still a temporal dependence: each segment consists of a temporal sequence of measurements. It will be useful for the shape reconstruction to eliminate this temporal dependence. We therefore introduce the notion of a *patch*. A *patch* is defined as an approximation of the radius

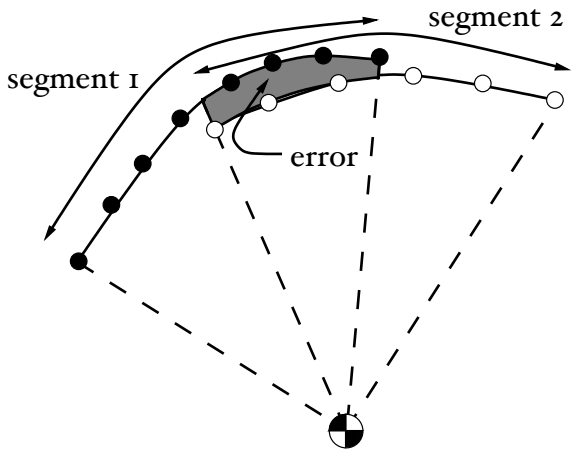


Figure 3.18: The error of two (radially) overlapping segments is equal to the area between them.

function on a domain $[\theta_1, \theta_2]$. We map a segment to a set of patches in the following way. Suppose we have a segment S consisting of n measurements. Each measurement is of the form (t_i, θ_i, r_i), $i = 1, \ldots, n$. We divide the segment S into smaller segments $S_1, \ldots, S_m$ at the m extrema of the (t_i, θ_i) sequence. If there are extrema, then the same part of the shape is covered more than once. Due to numerical errors and noise each time a part of the shape is covered the approximation will look slightly different. We therefore create a patch for each time a particular part of the shape is observed and consolidate patches with overlapping domains at a later stage. For each S_j, $j = 1, \ldots, m$, we create a patch P_j. The domain of P_j is defined as the closed interval ranging from the minimum to the maximum θ value in S_j's measurements. The approximation of the radius function is simply defined as the spline interpolation through P_j's observed values of the radius function.

In addition to the locally observable errors, we can also define a *global* error term as a function of a particular arrangement of segments: we can compute the area between conflicting sections of the segments and take that to be the error of the arrangement (see figure 3.18). For a given arrangement of segments we can compute all the patches as described above. If none of the patches overlap (radially), the error is equal to 0. Now suppose some of them do overlap. Let $[\theta_1, \theta_2]$ be a domain for which k patches $P_1, \ldots, P_k$ overlap. Let $\bar{P}$ be the average of the k patches. Then we can define the area error on $[\theta_1, \theta_2]$ to be equal to the

sum of the areas between each of the patches and $\bar{P}$:

$$A([\theta_1, \theta_2]) = \frac{1}{2} \sum_{i=1}^{k} \int_{\theta_1}^{\theta_2} |\bar{P}(\theta)^2 - P_i(\theta)^2| d\theta. \tag{3.48}$$

Recall that the area under a curve in polar coordinates $(\theta, f(\theta))$ is equal to $\int_\theta \int_0^{f(\theta)} r dr d\theta = \frac{1}{2} \int_\theta f(\theta)^2 d\theta$. We define $A(\cdot)$ to be equal to zero on intervals where no patch is defined. We can consider $\bar{P}$ to be the consolidated approximation of the shape. We can define the following error measure E for a given arrangement of reconstructed segments:

$$E = \big(1 + \int e_f(t)^2 dt\big)\big(1 + \int \|\boldsymbol{e}_c(t)\|^2 dt\big)\big(1 + A([0, 2\pi])\big) - 1 \tag{3.49}$$

Here e_f and $\boldsymbol{e}_c$ are the errors in the force/torque balance constraint and the two-point contact constraint, respectively, as defined before. These errors are integrated over the total duration of a simulation or an experiment. For large errors, E is dominated by the product of the three error terms, and for small errors, E is dominated by the sum of the three error terms. So the cost of increasing one term as we decrease another increases as E gets smaller. The shape reconstruction problem can now be rephrased as follows: given a series of measurements at time t_i, $i = 1, \ldots, n$, find a segmentation of this series and the initial conditions at the start of each segment such that E is minimized. Generally, E will have many local minima.

Finding a minimum of E is computationally very expensive. In most cases the search for the initial conditions at the start of each segment will dominate the running time. It is therefore important to set the thresholds such that the number of segments is small. One problem with the error function E is that it ignores the temporal dependencies between segments. One would expect that subsequent segments are also close in space, but there is no constraint to enforce this. We could try to simultaneously minimize the distances between segments and the error of the corresponding arrangement. The relative importance of distance versus error would add yet another parameter to the optimization problem. We can pick "reasonable" values for all these parameters, but slightly different parameters might result in radically different shape reconstructions. How to pick these parameters in an optimal way and how to compute a minimum of E are two important open problems. In our simulations the errors did not accumulate significantly and the numerical integration was robust enough to recover from singularities without having to start a new segment, but for more complicated shapes and manipulation strategies the segmentation of data is still an important problem.

Chapter 4

Dynamic Shape Reconstruction

In this chapter we drop the assumption that force/torque balance is maintained at all time. It turns out that it is still possible to observe the shape, but now we will need to consider second-order effects. We will need to model the accelerations, forces, and torques that occur in our system. Our approach is to construct an *observer* for our system. The notion of an observer will be explained later on in more detail, but for now one can think of an observer as an (on-line) state estimator that given an estimate quickly converges to the true state as time progresses. The construction of an observer requires several steps. The first step is to write our system in the following form:

$$\dot{q} = f(q) + \tau_1 g_1(q) + \tau_2 g_2(q), \tag{4.1}$$

$$y = h(q) \tag{4.2}$$

where q is a state vector, f, g_1 and g_2 are vector fields, and h is called the *output function*. In our case, the state is a vector of sensor readings and the configuration of the system. The configuration is defined by the orientations of the palms. The output function returns (a function of) the sensor readings. The vector fields g_1 and g_2 are called the *input vector fields* and describe the rate of change of our system as torques are being applied on palm 1 and palm 2, respectively, at their point of intersection. The vector field f is called the *drift vector field*. It includes the effects of gravity.

The second step is to find out whether the system described by equations 4.1 and 4.2 is *observable*. Informally, this notion can be defined as: a system is observable if for any two different states the output function will return a different value after some time.

The final step is then to construct the actual observer, which is basically a control law. We can estimate the initial state and if our estimate is not too far from the true initial state, the observer will rapidly converge to the actual state.

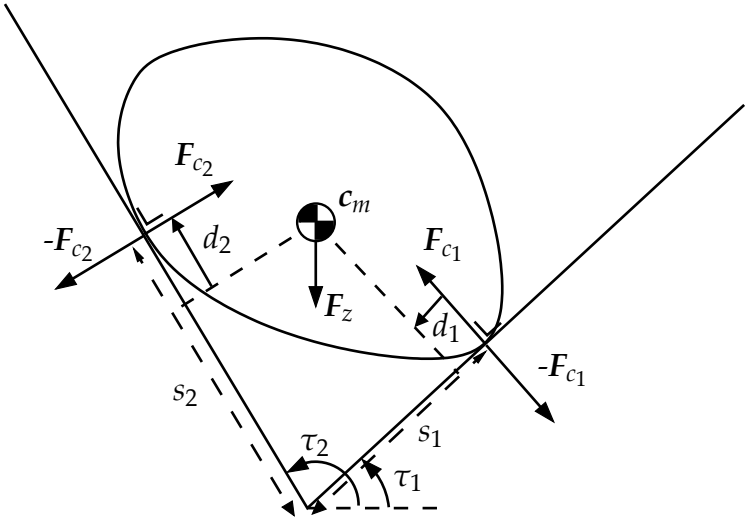


Figure 4.1: Forces acting on the palms and the object.

Interestingly, from the proof of observability we can not immediately derive an observer. The construction of observers for nonlinear systems is still an active area of research. For more on nonlinear control and nonlinear observers see, e.g., (Isidori, 1995) and (Nijmeijer and Van der Schaft, 1990).

4.1 Equations of Motion

In the previous chapter the quasistatic dynamics assumption provided a constraint that allowed us to solve for the velocity of the object. In this chapter we will model the forces exerted by the palms and gravity on the object. We will see that we can then solve for the acceleration of the object. By integration we can obtain the velocity. The dynamics of our simple model are very straightforward. We assume the effect of gravity on the palms is negligible. As in the previous chapter we assume there is no friction. The contact forces exert a pure torque on the palms about the origin. Let $\boldsymbol{F}_{c_1} = f_{c_1}\bar{\boldsymbol{n}}_1$ and $\boldsymbol{F}_{c_2} = f_{c_2}\bar{\boldsymbol{n}}_2$ be equal to the contact forces acting on the object. The torques generated by the two contact forces on the object are then

$$\tau_{c_1} = (R\boldsymbol{x}_1) \times \boldsymbol{F}_{c_1} = -f_{c_1} d_1 \tag{4.3}$$
$$\tau_{c_2} = (R\boldsymbol{x}_2) \times \boldsymbol{F}_{c_2} = -f_{c_2} d_2 \tag{4.4}$$

The dynamics of the system are described by the following equations (see also figure 4.1):

$$m\boldsymbol{a}_0 = \boldsymbol{F}_z + \boldsymbol{F}_{c_1} + \boldsymbol{F}_{c_2} \tag{4.5}$$

$$I_0\alpha_0 = \tau_{c_1} + \tau_{c_2} = -f_{c_1}d_1 - f_{c_2}d_2, \qquad f_{c_1}, f_{c_2} \geq 0 \tag{4.6}$$

$$I_1\alpha_1 = \tau_1 - f_{c_1}s_1 \tag{4.7}$$

$$I_2(\alpha_1 + \alpha_2) = \tau_2 + f_{c_2}s_2 \tag{4.8}$$

Here the subscript i, $(i = 0, 1, 2)$ refers to the object, palm 1 and palm 2, respectively. $\boldsymbol{F}_z = m\boldsymbol{g}$ is the gravitational force on the object. Note that α_2 is the angular acceleration of palm 2 measured with respect to palm 1, so that $\ddot{\phi}_2 = \alpha_2$. Solving for $\boldsymbol{a}_0$ and α_0, we get

$$\boldsymbol{a}_0 = -\frac{I_1\alpha_1 - \tau_1}{ms_1}\bar{\boldsymbol{n}}_1 + \frac{I_2(\alpha_1 + \alpha_2) - \tau_2}{ms_2}\bar{\boldsymbol{n}}_2 + \boldsymbol{g} \tag{4.9}$$

$$\alpha_0 = \frac{I_1\alpha_1 - \tau_1}{m\rho^2 s_1}d_1 - \frac{I_2(\alpha_1 + \alpha_2) - \tau_2}{m\rho^2 s_2}d_2 \tag{4.10}$$

where $\rho = \sqrt{I_0/m}$ is the radius of gyration of the object.

We can measure the mass m by letting the object come to rest. In that case $\boldsymbol{a}_0 = \boldsymbol{0}$ and we can solve for m by using $m = -(\boldsymbol{F}_{c_1} + \boldsymbol{F}_{c_2})/g$. We have to solve for the radius of gyration by other means, shown in the next section. The mass properties of the palms are assumed to be known.

4.2 General Case

We will now rewrite the constraints on the shape and motion of the object in the form of equation 4.1. We will introduce the state variables ω_0, ω_1 and ω_2 to denote $\dot{\phi}_0$, $\dot{\phi}_1$ and $\dot{\phi}_2$, respectively. The other state variables are: s_1, s_2, r_1, r_2, and ρ. So the shape at the contact points is represented by the values of the radius function at the contact points. Note that r_1 and r_2 are reparametrizations of the radius function as a function of time instead of θ. So even though shape is thought of as being time invariant, our representation describes shape as the rate of change in the radius function values at the contact points. The inputs to our system are the torques produced by the motors of the palms. The outputs are the sensor values. As we will see below, obtaining expressions for $\dot{s}_1$ and $\dot{s}_2$ is quite involved, while obtaining expressions for the derivatives of the other state variables is straightforward. Recall the position constraint on contact point 1 (equation 3.8):

$$s_1\bar{\boldsymbol{t}}_1 = \boldsymbol{c}_m + R\boldsymbol{x}_1 \tag{4.11}$$

We can differentiate this constraint twice to get a constraint on the acceleration of contact point 1. The right-hand side will contain a term with the curvature at contact point 1, and the acceleration constraint can be rewritten as a constraint on the curvature at contact point 1. But we already had an expression for curvature at contact point 1 that followed from differentiating the contact support function (see equation 3.29). By equating these two expressions and rearranging terms, we can obtain an expression for $\dot{s}_1$ that only depends on other state variables and the control inputs.

By differentiating the position constraint on contact point 1 we get a constraint on the velocity of contact point 1:

$$\dot{s}_1\bar{\boldsymbol{t}}_1 + s_1\omega_1\bar{\boldsymbol{n}}_1 = \dot{\boldsymbol{c}}_m + \omega_0 \times (R\boldsymbol{x}_1) + R\dot{\boldsymbol{x}}_1 \tag{4.12}$$

$$= \dot{\boldsymbol{c}}_m + \omega_0 \times (R\boldsymbol{x}_1) + (\omega_1 - \omega_0)v_1\bar{\boldsymbol{t}}_1 \tag{4.13}$$

This follows from $\theta = \phi_1 - \phi_0 - \frac{\pi}{2}$ and from our parametrization of the shape of the object. Differentiating again results in the following constraint on the acceleration:

$$\begin{aligned}
&\ddot{s}_1\bar{\boldsymbol{t}}_1 + 2\dot{s}_1\omega_1\bar{\boldsymbol{n}}_1 + s_1\alpha_1\bar{\boldsymbol{n}}_1 - s_1\omega_1^2\bar{\boldsymbol{t}}_1 = \\
&\quad = \boldsymbol{a}_0 + \alpha_0 \times (R\boldsymbol{x}_1) + \omega_0 \times (\omega_0 \times (R\boldsymbol{x}_1) + (\omega_1 - \omega_0)v_1\bar{\boldsymbol{t}}_1) \\
&\qquad + (\alpha_1 - \alpha_0)v_1\bar{\boldsymbol{t}}_1 + (\omega_1 - \omega_0)(\dot{v}_1\bar{\boldsymbol{t}}_1 + \omega_1 v_1\bar{\boldsymbol{n}}_1) && (4.14) \\
&\quad = \boldsymbol{a}_0 + \alpha_0 \times (R\boldsymbol{x}_1) - \omega_0^2 R\boldsymbol{x}_1 + (\omega_1^2 - \omega_0^2)v_1\bar{\boldsymbol{n}}_1 + (\alpha_1 - \alpha_0)v_1\bar{\boldsymbol{t}}_1 \\
&\qquad + (\omega_1 - \omega_0)\dot{v}_1\bar{\boldsymbol{t}}_1 && (4.15)
\end{aligned}$$

The acceleration constraint in the $\bar{\boldsymbol{n}}_1$ direction is therefore:

$$2\dot{s}_1\omega_1 + s_1\alpha_1 = \boldsymbol{a}_0 \cdot \bar{\boldsymbol{n}}_1 - \alpha_0 d_1 + \omega_0^2 r_1 + (\omega_1^2 - \omega_0^2)v_1. \tag{4.16}$$

We can solve this constraint for v_1:

$$v_1 = \frac{2\dot{s}_1\omega_1 + s_1\alpha_1 - \omega_0^2 r_1 - \boldsymbol{a}_0 \cdot \bar{\boldsymbol{n}}_1 + \alpha_0 d_1}{\omega_1^2 - \omega_0^2} \tag{4.17}$$

From before (equations 3.7 and 3.29) we had:

$$v_1 = -\frac{\dot{\tilde{r}}_2 + (\dot{\theta} + \dot{\phi}_2)\tilde{d}_2}{\dot{\theta}\sin\phi_2} = -\frac{(-\dot{s}_1\sin\phi_2 - s_1\omega_2\cos\phi_2) + (\omega_{12} - \omega_0)(s_1\cos\phi_2 - s_2)}{(\omega_1 - \omega_0)\sin\phi_2} \tag{4.18}$$

We can equate these two expressions for v_1 and solve for $\dot{s}_1$:

$$\dot{s}_1 = -\frac{s_1\alpha_1 - \omega_0^2 r_1 - \boldsymbol{a}_0 \cdot \bar{\boldsymbol{n}}_1 + \alpha_0 d_1}{\omega_1 - \omega_0} - \frac{\omega_1 + \omega_0}{\tan\phi_2}s_1 + \frac{(\omega_1 + \omega_0)(\omega_{12} - \omega_0)}{(\omega_1 - \omega_0)\sin\phi_2}s_2$$

Similarly we can derive an expression for $\dot{s}_2$. The differential equations describing our system can be summarized as follows:

$$\dot{r}_1 = -d_1(\omega_1 - \omega_0) \tag{4.19}$$
$$\dot{r}_2 = -d_2(\omega_{12} - \omega_0) \tag{4.20}$$
$$\dot{\omega}_0 = \alpha_0 \tag{4.21}$$
$$\dot{s}_1 = -\frac{s_1\alpha_1 - \omega_0^2 r_1 - \boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_1 + \alpha_0 d_1}{\omega_1 - \omega_0} - \frac{\omega_1+\omega_0}{\tan\phi_2}s_1 + \frac{(\omega_1+\omega_0)(\omega_{12}-\omega_0)}{(\omega_1-\omega_0)\sin\phi_2}s_2 \tag{4.22}$$
$$\dot{s}_2 = \frac{-s_2\alpha_{12} - \omega_0^2 r_2 - \boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_2 + \alpha_0 d_2}{\omega_{12} - \omega_0} + \frac{\omega_{12}+\omega_0}{\tan\phi_2}s_2 - \frac{(\omega_{12}+\omega_0)(\omega_1-\omega_0)}{(\omega_{12}-\omega_0)\sin\phi_2}s_1 \tag{4.23}$$
$$\dot{\phi}_1 = \omega_1 \tag{4.24}$$
$$\dot{\omega}_1 = \alpha_1 \tag{4.25}$$
$$\dot{\phi}_2 = \omega_2 \tag{4.26}$$
$$\dot{\omega}_2 = \alpha_2 \tag{4.27}$$
$$\dot{\rho} = 0 \tag{4.28}$$

Equation 4.19 and 4.20 follow from the properties of the radius function. Recall from section 3.1 that d_1 and d_2 can be written in terms of s_1, s_2, r_1, r_2 and ϕ_2. Therefore d_1 and d_2 do not need to be part of the state of our system. Leaving redundancies in the state would also make it hard, if not impossible, to prove observability of the system. Note also that the control inputs τ_1 and τ_2 are 'hidden' inside $\boldsymbol{a}_0$ and α_0. The expressions $-\boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_1 + \alpha_0 d_1$ and $-\boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_2 + \alpha_0 d_2$ can be rewritten using equations 4.9 and 4.10 as

$$-\boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_1 + \alpha_0 d_1 = \frac{(I_1\alpha_1 - \tau_1)(\rho^2 + d_1^2)}{m\rho^2 s_1} + \frac{(I_2\alpha_{12} - \tau_2)(\rho^2\cos\phi_2 - d_1 d_2)}{m\rho^2 s_2} - g\cos\phi_1, \tag{4.29}$$
$$-\boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_2 + \alpha_0 d_2 = -\frac{(I_1\alpha_1 - \tau_1)(\rho^2\cos\phi_2 - d_1 d_2)}{m\rho^2 s_1} - \frac{(I_2\alpha_{12} - \tau_2)(\rho^2 + d_2^2)}{m\rho^2 s_2} + g\cos\phi_{12}, \tag{4.30}$$

where $\alpha_{12} = \alpha_1 + \alpha_2$ and $\phi_{12} = \phi_1 + \phi_2$.

Let $\boldsymbol{q} = (r_1, r_2, \omega_0, s_1, s_2, \phi_1, \omega_1, \phi_2, \omega_2, \rho)^T$ be our state vector. Since τ_1 and τ_2 appear linearly in equations 4.19–4.28, our system fits the format of equa-

tion 4.1. The drift vector field is

$$\boldsymbol{f}(\boldsymbol{q}) = \begin{pmatrix} -d_1(\omega_1 - \omega_0) \\ -d_2(\omega_{12} - \omega_0) \\ \frac{I_1\alpha_1 d_1}{m\rho^2 s_1} - \frac{I_2\alpha_{12}d_2}{m\rho^2 s_2} \\ \frac{\omega_0^2 r_1 + g\cos\phi_1}{\omega_1 - \omega_0} - \frac{\omega_1+\omega_0}{\tan\phi_2}s_1 + \frac{(\omega_1+\omega_0)(\omega_{12}-\omega_0)}{(\omega_1-\omega_0)\sin\phi_2}s_2 + a_1 \\ \frac{-\omega_0^2 r_2 + g\cos\phi_{12}}{\omega_{12}-\omega_0} + \frac{\omega_{12}+\omega_0}{\tan\phi_2}s_2 - \frac{(\omega_{12}+\omega_0)(\omega_1-\omega_0)}{(\omega_{12}-\omega_0)\sin\phi_2}s_1 + a_2 \\ \omega_1 \\ \alpha_1 \\ \omega_2 \\ \alpha_2 \\ 0 \end{pmatrix}, \tag{4.31}$$

where a_1 and a_2 are terms that depend on the angular accelerations of the palms and are equal to

$$a_1 = -\frac{s_1\alpha_1}{\omega_1 - \omega_0} - \frac{I_1\alpha_1(\rho^2 + d_1^2)}{m\rho^2 s_1(\omega_1 - \omega_0)} - \frac{I_2\alpha_{12}(\rho^2\cos\phi_2 - d_1 d_2)}{m\rho^2 s_2(\omega_1 - \omega_0)} \tag{4.32}$$

$$a_2 = -\frac{s_2\alpha_{12}}{\omega_{12} - \omega_0} - \frac{I_1\alpha_1(\rho^2\cos\phi_2 - d_1 d_2)}{m\rho^2 s_1(\omega_{12} - \omega_0)} - \frac{I_2\alpha_{12}(\rho^2 + d_2^2)}{m\rho^2 s_2(\omega_{12} - \omega_0)} \tag{4.33}$$

The input vector fields are

$$\boldsymbol{g}_1(\boldsymbol{q}) = \begin{pmatrix} 0 \\ 0 \\ -\frac{d_1}{m\rho^2 s_1} \\ \frac{\rho^2 + d_1^2}{m\rho^2 s_1(\omega_1 - \omega_0)} \\ \frac{\rho^2\cos\phi_2 - d_1 d_2}{m\rho^2 s_1(\omega_{12} - \omega_0)} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \text{and} \quad \boldsymbol{g}_2(\boldsymbol{q}) = \begin{pmatrix} 0 \\ 0 \\ \frac{d_2}{m\rho^2 s_2} \\ \frac{\rho^2\cos\phi_2 - d_1 d_2}{m\rho^2 s_2(\omega_1 - \omega_0)} \\ \frac{\rho^2 + d_2^2}{m\rho^2 s_2(\omega_{12} - \omega_0)} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{4.34}$$

Finally, our output function $\boldsymbol{h}(\boldsymbol{q}) = \big(h_1(\boldsymbol{q}), \ldots, h_k(\boldsymbol{q})\big)^T$ is

$$\boldsymbol{h}(\boldsymbol{q}) = (s_1, s_2, \phi_1, \omega_1, \phi_2, \omega_2)^T. \tag{4.35}$$

Before we can determine the observability of this system we need to introduce some more notation. We define the *differential* $d\phi$ of a function ϕ defined on a subset of $\mathbb{R}^n$ as

$$d\phi(\boldsymbol{x}) = \left(\frac{\partial\phi}{\partial x_1}, \ldots, \frac{\partial\phi}{\partial x_n}\right)$$

The Lie derivative of a function ϕ along a vector field X, denoted $L_X\phi$, is defined as

$$L_X\phi = d\phi \cdot X$$

To determine whether the system above is observable we have to consider the *observation space* $\mathcal{O}$. The observation space is defined as the linear space of functions that includes $h_1, \ldots, h_k$, and all repeated Lie derivatives

$$L_{X_1}L_{X_2}\cdots L_{X_l}h_j, \qquad j = 1, \ldots, k, \quad l = 1, 2, \ldots \tag{4.36}$$

where $X_i \in \{\boldsymbol{f}, \boldsymbol{g}_1, \boldsymbol{g}_2\}$, $1 \leq i \leq l$. Let the *observability codistribution* at a state $\boldsymbol{q}$ be defined as

$$d\mathcal{O}(\boldsymbol{q}) = \text{span}\{dH(\boldsymbol{q}) | H \in \mathcal{O}\}. \tag{4.37}$$

Then the system described by equation 4.1 is locally observable at state $\boldsymbol{q}$ if $\dim d\mathcal{O}(\boldsymbol{q}) = n$, where n is the dimensionality of the state space (Hermann and Krener, 1977). For the system described by equations 4.19–4.28 this condition is too complicated to verify analytically, but one can still do this numerically.

The differentials of the components of the output function are

$$ds_1 = (0,0,0,1,0,0,0,0,0,0) \tag{4.38}$$
$$ds_2 = (0,0,0,0,1,0,0,0,0,0) \tag{4.39}$$
$$d\phi_1 = (0,0,0,0,0,1,0,0,0,0) \tag{4.40}$$
$$d\omega_1 = (0,0,0,0,0,0,1,0,0,0) \tag{4.41}$$
$$d\phi_2 = (0,0,0,0,0,0,0,1,0,0) \tag{4.42}$$
$$d\omega_2 = (0,0,0,0,0,0,0,0,1,0). \tag{4.43}$$

These differentials span six dimensions of the observability codistribution. The state space has ten dimensions, so to determine whether the system is observable we need to compute (numerically) the differentials of at least four Lie derivatives. In general $dL_{g_1}s_1$, $dL_{g_2}s_2$, $dL_{g_1}L_{g_1}s_1$ and $dL_{g_2}L_{g_2}s_2$ and the differentials above will span the observability codistribution $d\mathcal{O}$ (see appendix A.2). So the response of the system to the control inputs gives us information about the unknown part

of the state. We can use this information to improve the estimate for the values of the radius function at the contact points, the rotational velocity of the object and its radius of gyration. Note that we can not use the vector field f, because we do not have expressions for α_1 and α_2 in terms of the state variables and, hence, we can not compute the differentials of Lie derivatives along f.

The results above show that in general we will be able to observe the shape of an unkown object. Moreover, the output function contains enough information to recover a constant like the radius of gyration. This leads us to suspect that it may be possible to recover another constant as well: the coefficient of friction. Currently friction is not modeled, but we plan to address this in future work.

4.3 Moving the Palms at a Constant Rate

Although we have shown that the system in the previous section is observable, this does not directly translate to an actual observer. The observability tells us that an observer exists, but constructing a well-behaved observer for a nonlinear system is nontrivial and is still an active area of research. Many observers (such as those proposed by Gauthier et al. (1992) and Zimmer (1994)) rely on Lie derivatives of the drift field. If we want to use such an observer we have to constrain the motion of the palms by restricting them to move at a constant rate, i.e., $\alpha_1 = \alpha_2 = 0$. With this constraint the angular acceleration of the palms vanishes and we do not have to compute derivatives of the accelerations with respect to the state variables. Provided the palms are sufficiently stiff compared to the object, we can easily realize this. Note that this is an *assumption* and that in general a torque-based control system does not automatically translate to a velocity-based or position-based control system. For simplicity we will also assume that we already have recovered the radius of gyration.

Moving both palms at the same rate Suppose we move palm 1 and 2 at the same rate. Then $\omega_2 = 0$, since it measures the relative rate of palm 2 to palm 1. Our state vector then reduces to $\boldsymbol{q} = (r_1, r_2, \omega_0, s_1, s_2, \phi_1)^T$. The angle between the palms is constant, so ϕ_2 does not need to be part of the state. For the same reason we can omit ω_1 and ω_2 from the state vector. The output function is now

$\boldsymbol{h}(\boldsymbol{q}) = (s_1, s_2, \phi_1)^T$ and the drift vector field simplifies to

$$\boldsymbol{f}(\boldsymbol{q}) = \begin{pmatrix} -d_1(\omega_1 - \omega_0) \\ -d_2(\omega_1 - \omega_0) \\ 0 \\ \frac{\omega_0^2 r_1 + g\cos\phi_1}{\omega_1 - \omega_0} - (\omega_1 + \omega_0)\tilde{d}_2 / \sin\phi_2 \\ \frac{-\omega_0^2 r_2 + g\cos\phi_{12}}{\omega_1 - \omega_0} + (\omega_1 + \omega_0)\tilde{d}_1 / \sin\phi_2 \\ \omega_1 \end{pmatrix}. \tag{4.44}$$

Recall from chapter 3, equation 3.7 that $\tilde{d}_1 = s_2\cos\phi_2 - s_1$ and $\tilde{d}_2 = s_1\cos\phi_2 - s_2$. We can compute the differentials and Lie derivatives that are necessary to prove the observability:

$$ds_1 = (0,0,0,1,0,0) \tag{4.45}$$

$$ds_2 = (0,0,0,0,1,0) \tag{4.46}$$

$$d\phi_1 = (0,0,0,0,0,1) \tag{4.47}$$

$$L_f s_1 = ds_1 \cdot \boldsymbol{f} = \tfrac{\omega_0^2 r_1 + g\cos\phi_1}{\omega_1 - \omega_0} + (\omega_1 + \omega_0)(s_2 - s_1\cos\phi_2)/\sin\phi_2 \tag{4.48}$$

$$dL_f s_1 = \big(\tfrac{\omega_0^2}{\omega_1-\omega_0},\ 0,\ \tfrac{r_1\omega_0(2\omega_1-\omega_0)+g\cos\phi_1}{(\omega_1-\omega_0)^2} + \tfrac{s_2 - s_1\cos\phi_2}{\sin\phi_2},\ -\tfrac{\omega_1+\omega_0}{\tan\phi_2},\ \tfrac{\omega_1+\omega_0}{\sin\phi_2},\ -\tfrac{g\sin\phi_1}{\omega_1-\omega_0}\big) \tag{4.49}$$

$$L_f s_2 = \frac{-\omega_0^2 r_2 + g\cos\phi_{12}}{\omega_1 - \omega_0} + (\omega_1 + \omega_0)(s_2\cos\phi_2 - s_1)/\sin\phi_2 \tag{4.50}$$

$$dL_f s_2 = \big(0,\ \tfrac{-\omega_0^2}{\omega_1-\omega_0},\ \tfrac{-r_2\omega_0(2\omega_1-\omega_0)+g\cos\phi_{12}}{(\omega_1-\omega_0)^2} + \tfrac{s_2\cos\phi_2 - s_1}{\sin\phi_2},\ -\tfrac{\omega_1+\omega_0}{\sin\phi_2},\ \tfrac{\omega_1+\omega_0}{\tan\phi_2},\ -\tfrac{g\sin\phi_{12}}{\omega_1-\omega_0}\big) \tag{4.51}$$

$$L_f L_f s_1 = -\tfrac{2\omega_0^2\omega_1(r_1\cos\phi_2 + r_2)}{(\omega_1-\omega_0)\sin\phi_2} - s_1\omega_1(2\omega_0 + \omega_1) - \tfrac{g\sin\phi_1(\omega_0+2\omega_1)}{\omega_1-\omega_0} \tag{4.52}$$

$$dL_f L_f s_1 = \big(\tfrac{-2\omega_0^2\omega_1}{(\omega_1-\omega_0)\tan\phi_2},\ \tfrac{-2\omega_0^2\omega_1}{(\omega_1-\omega_0)\sin\phi_2},\ \tfrac{-2\omega_0\omega_1(2\omega_1-\omega_0)(r_1\cos\phi_2+r_2)}{(\omega_1-\omega_0)^2\sin\phi_2} - 2s_1\omega_1 - \tfrac{3\omega_1 g\sin\phi_1}{(\omega_1-\omega_0)^2},\ -\omega_1(2\omega_0+\omega_1),\ 0,\ \tfrac{-g\cos\phi_1(\omega_0+2\omega_1)}{\omega_1-\omega_0}\big) \tag{4.53}$$

It can be shown that the determinant of the matrix that has colums ds_1, ds_2, $d\phi_1$, $dL_f s_1$, $dL_f s_2$ and $dL_f L_f s_1$ is equal to

$$\frac{g\omega_0^4\omega_1\sin\phi_1}{(\omega_1 - \omega_0)^4} \tag{4.54}$$

In other words, the system is observable as long as ω_0 and ω_1 are nonzero and not equal to each other. Recall from equation 3.28 that $\dot{\theta}$ is equal to $\omega_1 - \omega_0$. So if ω_0

and ω_1 are equal to each other, $\dot{\theta} = 0$ and we cannot observe the curvature at the contact points (cf. equations 3.29 and 3.30). Note that the system is observable solely in terms of the drift vector field.

Keeping one palm fixed Let us now consider the case $\omega_1 = 0$: palm 1 is fixed and palm 2 is moving at a constant rate. The state vector then reduces to $\boldsymbol{q} = (r_1, r_2, \omega_0, s_1, s_2, \phi_2)^T$. The output function is now $\boldsymbol{h}(\boldsymbol{q}) = (s_1, s_2, \phi_2)^T$ and the drift vector field simplifies to

$$\boldsymbol{f}(\boldsymbol{q}) = \begin{pmatrix} d_1\omega_0 \\ -d_2(\omega_2 - \omega_0) \\ 0 \\ -\frac{\omega_0^2 r_1 + g\cos\phi_1}{\omega_0} - \frac{\omega_0 s_1}{\tan\phi_2} - \frac{(\omega_2-\omega_0)s_2}{\sin\phi_2} \\ \frac{-\omega_0^2 r_2 + g\cos\phi_{12}}{\omega_2-\omega_0} + \frac{(\omega_2+\omega_0)s_2}{\tan\phi_2} + \frac{(\omega_2+\omega_0)\omega_0 s_1}{(\omega_2-\omega_0)\sin\phi_2} \\ \omega_2 \end{pmatrix} \tag{4.55}$$

As before, we need to compute the differentials and Lie derivatives to determine observability:

$$ds_1 = (0,0,0,1,0,0) \tag{4.56}$$

$$ds_2 = (0,0,0,0,1,0) \tag{4.57}$$

$$d\phi_2 = (0,0,0,0,0,1) \tag{4.58}$$

$$L_f s_2 = -\frac{\omega_0^2 r_1 + g\cos\phi_1}{\omega_0} - \frac{\omega_0 s_1}{\tan\phi_2} - \frac{(\omega_2-\omega_0)s_2}{\sin\phi_2} \tag{4.59}$$

$$dL_f s_1 = \Big(-\omega_0,\ 0,\ -r_1 + \frac{g\cos\phi_1}{\omega_0^2} - \frac{s_1}{\tan\phi_2} + \frac{s_2}{\sin\phi_2},\ -\frac{\omega_0}{\tan\phi_2},\ -\frac{\omega_2-\omega_0}{\sin\phi_2},\ \frac{\omega_0 s_1 + (\omega_2-\omega_0)s_2\cos\phi_2}{\sin^2\phi_2}\Big) \tag{4.60}$$

$$L_f s_2 = \frac{-\omega_0^2 r_2 + g\cos\phi_{12}}{\omega_2-\omega_0} + \frac{(\omega_2+\omega_0)s_2}{\tan\phi_2} + \frac{(\omega_2+\omega_0)\omega_0 s_1}{(\omega_2-\omega_0)\sin\phi_2} \tag{4.61}$$

$$dL_f s_2 = \Big(0,\ \frac{-\omega_0^2}{\omega_2-\omega_0},\ \frac{-r_2\omega_0(2\omega_2-\omega_0)+g\cos\phi_{12}}{(\omega_2-\omega_0)^2} + \frac{s_2}{\tan\phi_2} - \frac{s_1(\omega_2^2+2\omega_0\omega_2-\omega_0^2)}{(\omega_2-\omega_0)^2\sin\phi_2},\ \frac{(\omega_2+\omega_0)\omega_0}{(\omega_2-\omega_0)\sin\phi_2},\ \frac{\omega_2+\omega_0}{\tan\phi_2},\ \frac{-g\sin\phi_{12}}{\omega_2-\omega_0} - \frac{s_2(\omega_2+\omega_0)}{\sin^2\phi_2} - \frac{(\omega_2+\omega_0)\omega_0 s_1\cos\phi_2}{(\omega_2-\omega_0)\sin^2\phi_2}\Big) \tag{4.62}$$

$$L_f L_f s_2 = -\frac{2\omega_0^2\omega_2(r_1 + r_2\cos\phi_2)}{(\omega_2-\omega_0)\sin\phi_2} - s_2\omega_2(2\omega_0+\omega_2) - \frac{g(\omega_0+2\omega_2)\sin\phi_{12}}{\omega_2-\omega_0} \tag{4.63}$$

$$dL_f L_f s_2 = \Big(\frac{-2\omega_0^2\omega_2}{(\omega_2-\omega_0)\sin\phi_2},\ \frac{-2\omega_0^2\omega_2}{(\omega_2-\omega_0)\tan\phi_2},\ \frac{-2\omega_0\omega_2(2\omega_2-\omega_0)(r_1+r_2\cos\phi_2)}{(\omega_2-\omega_0)^2\sin\phi_2} - 2s_2\omega_2 - \frac{3\omega_2 g\sin\phi_{12}}{(\omega_2-\omega_0)^2},\ 0,\ -\omega_2(2\omega_0+\omega_2),\ \frac{2\omega_0^2\omega_2(r_1\cos\phi_2+r_2)}{(\omega_2-\omega_0)\sin^2\phi_2} - \frac{-g\cos\phi_{12}(\omega_0+2\omega_2)}{\omega_2-\omega_0}\Big) \tag{4.64}$$

It can be shown that the determinant of the matrix that has colums ds_1, ds_2, $d\phi_1$, $dL_f s_1$, $dL_f s_2$ and $dL_f L_f s_2$ is equal to

$$-\frac{2\omega_0^3\omega_2^2}{(\omega_2-\omega_0)^3\sin\phi_2}\left[\left(\frac{\cos\phi_1}{\omega_0}+\frac{\sin\phi_1\sin\phi_{12}}{2\omega_2}\right)g+\omega_0 r_1+\frac{s_1(\omega_2+\omega_0)}{\tan\phi_2}+\frac{s_2(\omega_2-\omega_0)}{\sin\phi_2}\right]. \quad (4.65)$$

This determinant is generally nonzero if ω_0 and ω_2 are nonzero and not equal to each other. By the same argument as in the previous case, if $\omega_0 = 0$ or $\omega_2 - \omega_0 = 0$ we cannot observe the curvature at contact point 1 or contact point 2, respectively. So even if we move only palm 2 at a constant rate and hold palm 1 fixed, the system is still locally observable.

4.4 Fixed Palms

For illustrative purposes we will now consider one more special case, where it is possible to analytically determine observability. Suppose our control strategy consists of keeping the palms in the same position. In other words, $\omega_1 = \omega_2 = \alpha_1 = \alpha_2 = 0$. Note that this is not the same as clamping the palms, since we still assume that the palms are actively controlled. We can then reduce our state even further to $\boldsymbol{q} = (r_1, r_2, \omega_0, s_1, s_2)^T$. The input and control vector fields simplify to

$$\boldsymbol{f}(\boldsymbol{q}) = \begin{pmatrix} d_1\omega_0 \\ d_2\omega_0 \\ 0 \\ -\frac{g\cos\phi_1}{\omega_0} - \omega_0\left(r_1 + \frac{s_1}{\tan\phi_2} - \frac{s_2}{\sin\phi_2}\right) \\ -\frac{g\cos\phi_{12}}{\omega_0} + \omega_0\left(r_2 + \frac{s_2}{\tan\phi_2} - \frac{s_1}{\sin\phi_2}\right) \end{pmatrix},$$

$$\boldsymbol{g}_1(\boldsymbol{q}) = \begin{pmatrix} 0 \\ 0 \\ -\frac{d_1}{m\rho^2 s_1} \\ -\frac{\rho^2+d_1^2}{m\rho^2 s_1\omega_0} \\ -\frac{\rho^2\cos\phi_2 - d_1 d_2}{m\rho^2 s_1\omega_0} \end{pmatrix}, \quad \text{and} \quad \boldsymbol{g}_2(\boldsymbol{q}) = \begin{pmatrix} 0 \\ 0 \\ \frac{d_2}{m\rho^2 s_2} \\ -\frac{\rho^2\cos\phi_2 - d_1 d_2}{m\rho^2 s_2\omega_0} \\ -\frac{\rho^2+d_2^2}{m\rho^2 s_2\omega_0} \end{pmatrix}. \quad (4.66)$$

The output function is now simply $\boldsymbol{h}(\boldsymbol{q}) = (s_1, s_2)^T$. Since the output function has two components and our state space is five-dimensional, we need to take at least three Lie derivatives. Consider the following differentials and Lie

derivatives:

$$ds_1 = (0,0,0,1,0) \tag{4.67}$$
$$ds_2 = (0,0,0,0,1) \tag{4.68}$$
$$L_f s_1 = -\tfrac{g\cos\phi_1}{\omega_0} - \omega_0\big(r_1 + \tfrac{s_1}{\tan\phi_2} - \tfrac{s_2}{\sin\phi_2}\big) \tag{4.69}$$
$$dL_f s_1 = \Big(-\omega_0, 0, \tfrac{g\cos\phi_1}{\omega_0^2} - \big(r_1 + \tfrac{s_1}{\tan\phi_2} - \tfrac{s_2}{\sin\phi_2}\big), -\tfrac{\omega_0}{\tan\phi_2}, \tfrac{\omega_0}{\sin\phi_2}\Big) \tag{4.70}$$
$$L_f s_2 = -\tfrac{g\cos\phi_{12}}{\omega_0} - \omega_0\big(r_2 + \tfrac{s_2}{\tan\phi_2} - \tfrac{s_1}{\sin\phi_2}\big) \tag{4.71}$$
$$dL_f s_2 = \Big(0, -\omega_0, -\tfrac{g\cos\phi_{12}}{\omega_0^2} - \big(r_2 + \tfrac{s_2}{\tan\phi_2} - \tfrac{s_1}{\sin\phi_2}\big), \tfrac{\omega_0}{\sin\phi_2}, -\tfrac{\omega_0}{\tan\phi_2}\Big) \tag{4.72}$$
$$L_f L_f s_1 = \omega_0^2(-d_1 + \tfrac{r_1}{\tan\phi_2} + \tfrac{r_2}{\sin\phi_2} - s_1) + g\sin\phi_1 = g\sin\phi_1 \tag{4.73}$$
$$dL_f L_f s_1 = (0,0,0,0,0) \tag{4.74}$$

The step in equation 4.73 follows from expression 3.18. Because of symmetry $dL_f L_f s_2$ is equal to the $\mathbf{0}$ vector as well. This means that with fixed palms the system is *not* observable in terms of *just* the drift field f. Now suppose we compute the Lie derivative of s_1 along g_1 and its differential:

$$L_{g_1} s_1 = -\frac{\rho^2 + d_1^2}{m\rho^2 s_1 \omega_0} \tag{4.75}$$
$$dL_{g_1} s_1 = \tfrac{1}{m\rho^2 s_1\omega_0}\big(\tfrac{-2d_1}{\tan\phi_2},\ \tfrac{-2d_1}{\sin\phi_2},\ \tfrac{\rho^2+d_1^2}{\omega_0},\ 2d_1 + \tfrac{\rho^2+d_1^2}{s_1},\ 0\big) \tag{4.76}$$

The differentials $ds_1, ds_2, dL_f s_1, dL_f s_2$ and $dL_{g_1} s_1$ generally span the observability codistribution and, hence, the system is observable. It is important to remember that the palms are actively controlled, i.e., the palms are not clamped. Otherwise we would not know the torques exerted by the palms. We need the torques to integrate (by using an observer) the differential equation 4.1. As mentioned before, the construction of an observer that relies on the control vector fields is nontrivial. Since the motion of the palms is so constrained, the system is likely to observe only a small fraction of an unknown shape. Therefore we suspect that if one were to construct an observer it would have very limited practical value.

4.5 An Observer Based on Newton's Method

The observer we implemented is based on Newton's method (Zimmer, 1994) for finding the roots (i.e., zeros) of a function. Before we describe the details of our observer, let us first quickly review Newton's method. Suppose we want to find a root of a function $f : \mathbb{R} \to \mathbb{R}$. Let x_0 be an initial guess close to a root $\bar{x}$ of f.

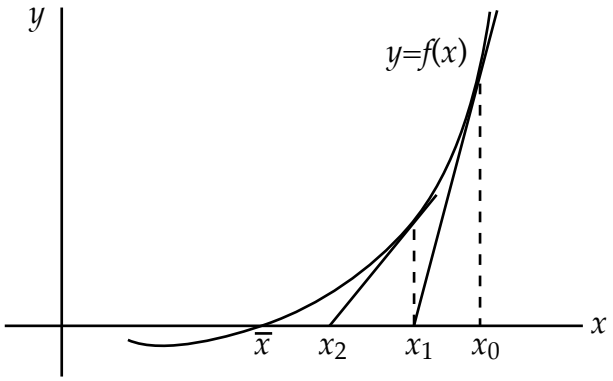


Figure 4.2: Newton's method.

Then the sequence $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$, $n \geq 0$, converges quadratically to $\bar{x}$. See figure 4.2 for an illustration of this method.

With an observer we are trying to minimize the difference between the state estimate and the true state. We cannot actually observe the true state, but we can consider the differences between the expected output based on the estimate and the actual output. Following Zimmer, we say that a system in the form of equations 4.1–4.2 is *strongly observable in* Q if and only if for any time interval $I = [0, T]$ and for any given initial states $\boldsymbol{q}_1(0), \boldsymbol{q}_2(0) \in Q$ and for any $t \in I$

- the states $\boldsymbol{q}_1(t)$ and $\boldsymbol{q}_2(t)$ remain in Q, and
- the system outputs $\boldsymbol{y}(\boldsymbol{q}_1(t))$ and $\boldsymbol{y}(\boldsymbol{q}_2(t))$ are identical in I only if $\boldsymbol{q}_1 = \boldsymbol{q}_2$.

A system is called *locally strongly observable in* Q if and only if there is an $\varepsilon > 0$ so that for every t in the time interval $I = [0, T]$ and for every $\boldsymbol{q} \in Q$ the system at time t is locally observable in $B_\varepsilon(\boldsymbol{q}) \cap Q$, where $B_\varepsilon(\boldsymbol{q})$ is a n-dimensional sphere with radius ε centered at $\boldsymbol{q}$.

Let us now define the distance function $k : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^+$:

$$k(\boldsymbol{q}_1(0), \boldsymbol{q}_2(0)) = \tfrac{1}{2} \int_0^T \|\boldsymbol{h}(\boldsymbol{q}_1(t)) - \boldsymbol{h}(\boldsymbol{q}_2(t))\|^2 dt \tag{4.77}$$

So the distance between two states $\boldsymbol{q}_1$ and $\boldsymbol{q}_2$ at $t = 0$ is defined as the difference between the outputs over time T obtained by integrating out equation 4.1 for each state and applying the output function to it. So we can transform the state estimation problem into a minimization problem. Let $\boldsymbol{q}^*$ be an estimate for $\boldsymbol{q}$ and let $k_q = k(\cdot, \boldsymbol{q})$ be the error function for a particular state. Clearly, if a system is strongly observable, then k_q has a unique minimum at $\boldsymbol{q}$, equal to 0. But $\boldsymbol{q}$ is also a root of the first derivative of k_q. We can apply Newton's method to improve

the state estimate q^* of q:

$$q^*_{\text{new}} = q^*_{\text{old}} - \left(\frac{\partial^2 k_q}{(\partial q)^2}\right)^{-1} \left(\frac{\partial k_q}{\partial q}\right)^T \tag{4.78}$$

(The 'T' denotes transpose here, not time.) Note that to evaluate $k_q(q^*)$ we do *not* need to know q. All we need is the *output* over an interval I. For this method to be successful a number of conditions need to be satisfied:

- The function k_q needs to be convex near q. This is equivalent to saying that the Hessian matrix $\frac{\partial^2 k_q}{(\partial q)^2}$ is positive definite. Zimmer showed that this is the case if and only if the linearized system around q is locally observable in I.
- This Hessian matrix also has to be invertible. Zimmer derived sufficient conditions to test if that is the case.

If these conditions are *not* satisfied (for instance, near or at singularities), we need to resort to other methods for shape recovery. The simplest method is to simply integrate the differential equations describing our system. Alternatively, we can ignore the sensor data while the above conditions are not satisfied and restart our observer when they are.

There are several ways we can use the update rule of equation 4.78 in an observer depending on the amount of computation we are willing to spend on improving the state estimate. If we are trying to reconstruct a shape in real-time, then we may not be able to take many Newton steps. In that case, we can update our state estimate every T seconds by taking one Newton step as given by equation 4.78. If we are reconstructing the shape offline, we can take an arbitrary number of Newton steps. We could even update the state estimate more frequently than every T seconds.

Although most observers are used for online state estimation, this Newton observer has more of an offline flavor. There is a minimal delay of T seconds between measuring the output of the system and being able to correct the state estimate of the system at that time, because the observer needs to know how an error in the state estimate at a given time affects the output in the T seconds after it.

4.6 Simulation Results

The observer described above has been implemented in Matlab. The results of one simulation are shown in figures 4.3 and 4.4. We simulated the motion of a

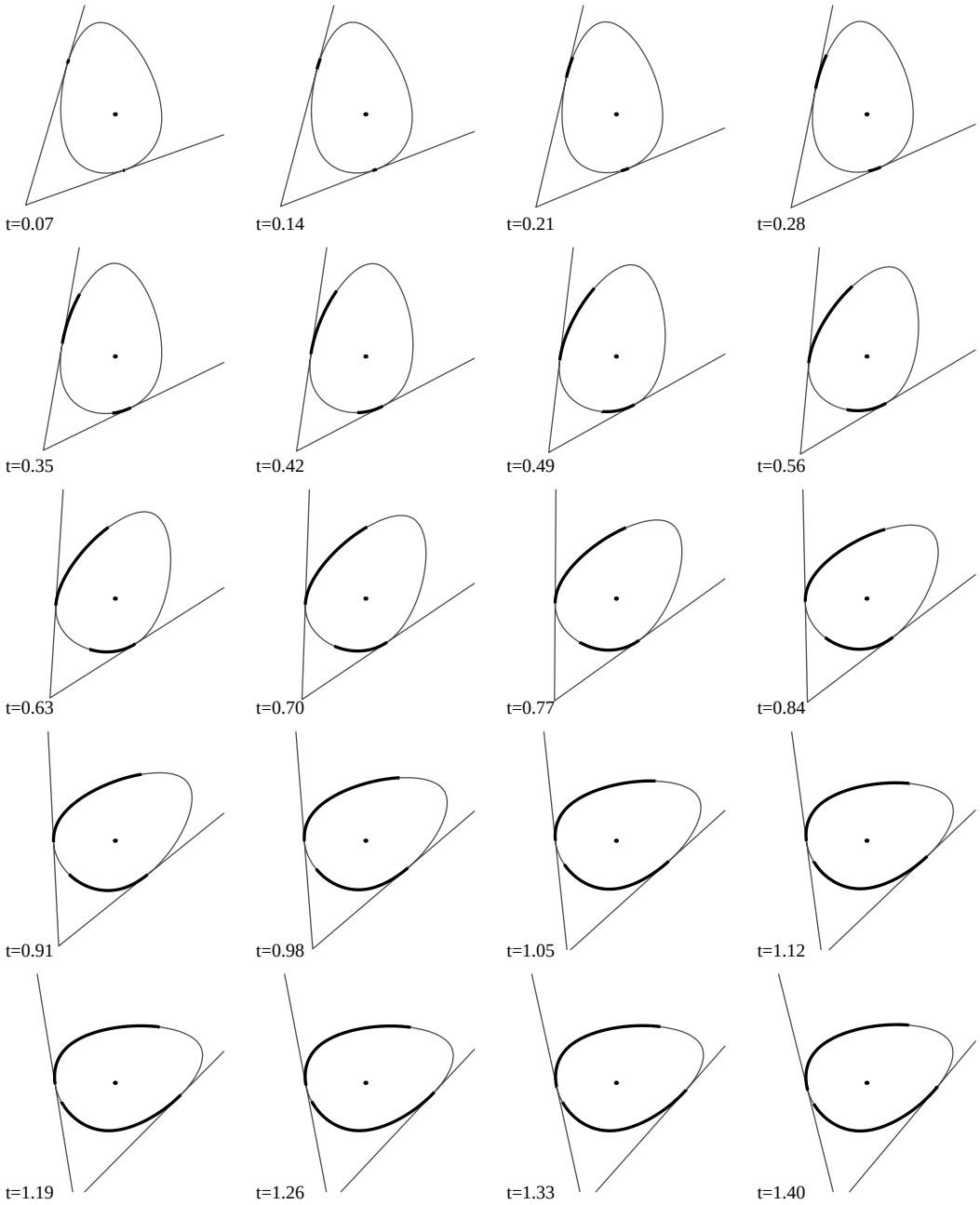
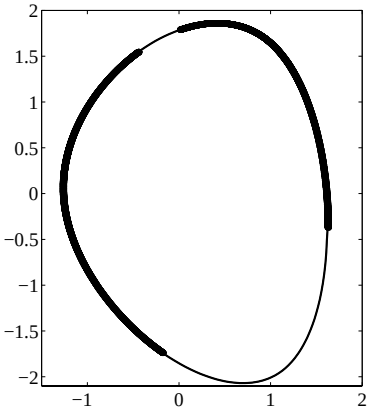
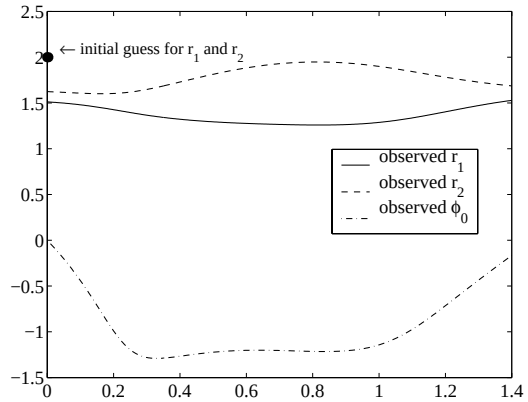


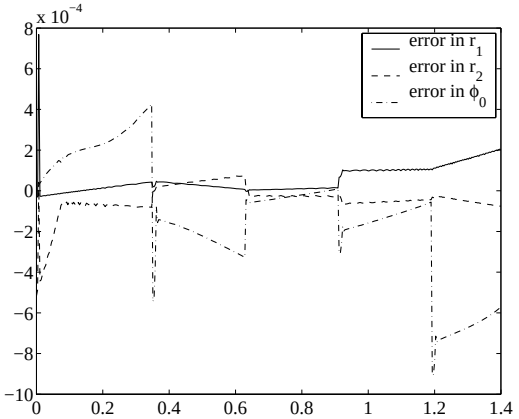
Figure 4.3: The frames show the reconstructed shape after 10, 20,...,400 measurements. The three large dots indicate the center of mass and the contact points at each time, the smaller dots show the part of the shape that has been reconstructed at that time.



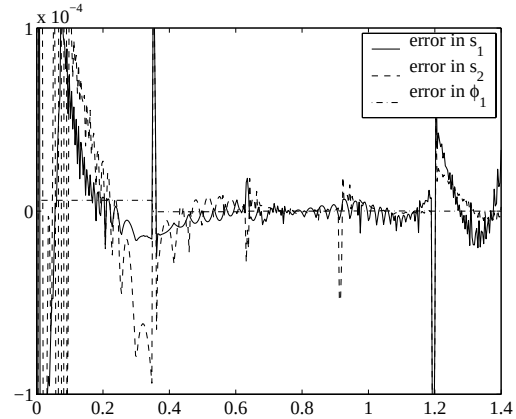
(a) The reconstructed shape



(b) The reconstructed radius function and orientation



(c) The absolute error in the radius function and the orientation



(d) The absolute error in the sensor values and palm angle 1

Figure 4.4: Shape reconstruction with an observer based on Newton's method.

random smooth shape over 1.4 seconds. We split the total time into 400 time steps and applied five Newton steps after every 80 time steps. At $t = 0$ we do something slightly different. We assume the object's initial velocity is equal to **0**. This simplifies the search for the initial conditions since we now only have to search for the initial values of the radius function at the contact points. The initial guess for the value of the radius function was set to 2 for both contact points. The initial value of ω_0 is 'clamped' to 0 at $t = 0$; the Newton observer cannot change this initial value. After five Newton steps the observer has found values for the radius function that are very close to the actual values. The recovered values of the radius function are very close to the true values. Figure 4.4(d) shows

the difference in the sensor values the observer expected to see and the actual sensor values.

4.7 Arbitrary Palm Shapes

In this chapter we wrote the shape and motion of the object as a function of the shape and motion of the palms. Instead of the shape and motion of the palms, we could also use the contact point positions and velocities in world coordinates. Let $\boldsymbol{c}_i$ be contact point i in world coordinates. Then there exist functions $\mathcal{C}_{\text{palms}}$ and $\mathcal{C}_{\text{object}}$ such that

$$\mathcal{C}_{\text{palms}}(\tau_1, \phi_1, \omega_1, s_1, \dot{s}_1, \tau_2, \phi_2, \omega_2, s_2, \dot{s}_2) = \begin{pmatrix} \boldsymbol{c}_1, & \dot{\boldsymbol{c}}_1, & \boldsymbol{c}_2, & \dot{\boldsymbol{c}}_2 \end{pmatrix}^T \tag{4.79}$$

$$\mathcal{C}_{\text{object}}(r_1, \dot{r}_1, r_2, \dot{r}_2, \phi_0, \omega_0, \alpha_0, \boldsymbol{a}_0) = \begin{pmatrix} \boldsymbol{c}_1, & \dot{\boldsymbol{c}}_1, & \boldsymbol{c}_2, & \dot{\boldsymbol{c}}_2 \end{pmatrix}^T \tag{4.80}$$

Effectively, we have used world coordinates to decouple the solution for the shape and motion of the unknown object from the shape and motion of the palm. This allows us to use differently shaped palms without changing the solution for the shape and motion of the unknown object. Suppose we were to use circular palms. (For different palm shapes the approach would be the same.) Let $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ be defined as

$$\boldsymbol{c}_1 = b_1 R_1 \begin{pmatrix} \sin(s_1/b_1) \\ 1 - \cos(s_1/b_1) \end{pmatrix} \tag{4.81}$$

$$\boldsymbol{c}_2 = b_2 R_2 \begin{pmatrix} \sin(s_2/b_2) \\ \cos(s_2/b_2) - 1 \end{pmatrix} \tag{4.82}$$

Here b_i is the radius of palm i and R_i a rotation matrix equal to

$$R_i = \begin{pmatrix} \cos\phi_i & -\sin\phi_i \\ \sin\phi_i & \cos\phi_i \end{pmatrix}.$$

Note that ϕ_2 is now the angle between the X-axis and palm 2, and not the angle between the palms. The sensor values correspond to arc length. See figure 4.5 for an illustration. Note that just as with flat palms, if $s_i = 0$, then $\boldsymbol{c}_i = \boldsymbol{0}$. In fact, in the limit, as b_1 and b_2 approach infinity, the circular palms are equivalent to the flat palms.

The local coordinate frames at the contact points are now

$$\bar{\boldsymbol{t}}_1 = R_1 \begin{pmatrix} \cos(s_1/b_1) \\ \sin(s_1/b_1) \end{pmatrix} \qquad \bar{\boldsymbol{t}}_2 = R_2 \begin{pmatrix} -\cos(s_2/b_2) \\ \sin(s_2/b_2) \end{pmatrix}$$

$$\bar{\boldsymbol{n}}_1 = R_1 \begin{pmatrix} -\sin(s_1/b_1) \\ \cos(s_1/b_1) \end{pmatrix} \qquad \bar{\boldsymbol{n}}_2 = -R_2 \begin{pmatrix} \sin(s_2/b_2) \\ \cos(s_2/b_2) \end{pmatrix}$$

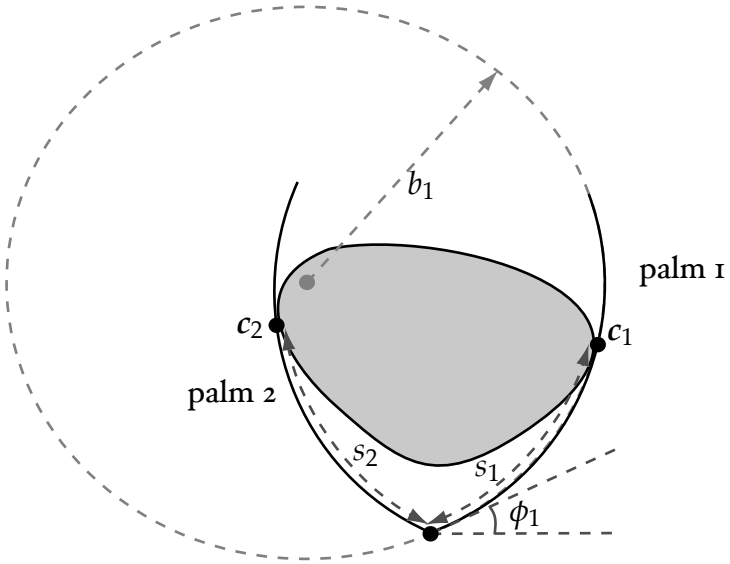


Figure 4.5: Circular palms.

By looking at projections of c_i onto these frames we can again obtain a system of differential equations similar to equations 4.19–4.28. Most of the differential equations remain the same. The ones that depend directly on the palm shape are the equations for $\dot{\omega}_0$, $\dot{s}_1$, and $\dot{s}_2$. Below we outline the procedure for proving observability for a system with circular palms.

The first and second derivative of contact point 1 are:

$$\dot{\boldsymbol{c}}_1 = \omega_1 \times \boldsymbol{c}_1 + \dot{s}_1 \bar{\boldsymbol{t}}_1 \tag{4.83}$$

$$\begin{aligned}\ddot{\boldsymbol{c}}_1 &= \alpha_1 \times \boldsymbol{c}_1 + \omega_1 \times (\omega_1 \times \boldsymbol{c}_1 + 2\dot{s}_1 \bar{\boldsymbol{t}}_1) + \ddot{s}_1 \bar{\boldsymbol{t}}_1 + (\dot{s}_1^2/b_1)\bar{\boldsymbol{n}}_1 \\ &= \alpha_1 \times \boldsymbol{c}_1 - \omega_1^2 \boldsymbol{c}_1 + 2\omega_1 \dot{s}_1 \bar{\boldsymbol{n}}_1 + \ddot{s}_1 \bar{\boldsymbol{t}}_1 + (\dot{s}_1^2/b_1)\bar{\boldsymbol{n}}_1 \end{aligned} \tag{4.84}$$

We can write $\boldsymbol{c}_1$ also as a function of the pose of the object: $\boldsymbol{c}_1 = \boldsymbol{c}_m + R_0 \boldsymbol{x}_1$. By differentiating this expression twice and equating it to expression 4.84, we obtain a constraint on the acceleration at contact point 1. In fact, we already computed the second derivative of this expression; it is given by the right-hand side of equation 4.15. The projection of the acceleration constraint onto the palm normal at contact point 1 is given by

$$\begin{aligned}\alpha_1(\boldsymbol{c}_1 \cdot \bar{\boldsymbol{t}}_1) - \omega_1^2(\boldsymbol{c}_1 \cdot \bar{\boldsymbol{n}}_1) + 2\omega_1 \dot{s}_1 + \dot{s}_1^2/b_1 \\ = \boldsymbol{a}_0 \cdot \bar{\boldsymbol{n}}_1 + \omega_0^2 r_1 - \alpha_0 d_1 + (\omega_1^2 - \omega_0^2) v_1. \end{aligned} \tag{4.85}$$

Solving for v_1 we get:

$$v_1 = \frac{\alpha_1(\boldsymbol{c}_1\cdot\bar{\boldsymbol{t}}_1)-\omega_1^2(\boldsymbol{c}_1\cdot\bar{\boldsymbol{n}}_1)+2\omega_1\dot{s}_1+\dot{s}_1^2/b_1-\omega_0^2 r_1-\boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_1+\alpha_0 d_1}{\omega_1^2-\omega_0^2} \tag{4.86}$$

$$= \frac{\alpha_1 b_1\sin(s_1/b_1)-\omega_1^2 b_1(\cos(s_1/b_1)-1)+2\omega_1\dot{s}_1+\dot{s}_1^2/b_1-\omega_0^2 r_1-\boldsymbol{a}_0\cdot\bar{\boldsymbol{n}}_1+\alpha_0 d_1}{\omega_1^2-\omega_0^2} \tag{4.87}$$

The expression we obtained for v_1 in chapter 3, equation 3.29, is still valid if we replace $\dot{\phi}_2$ with the time derivative of the angle between $\bar{\boldsymbol{t}}_1$ and $-\bar{\boldsymbol{t}}_2$. The generalized support functions can still be expressed as functions of the sensor values and palm angles, but these expressions will now be very different. We can equate equation 3.29 and 4.86, substitute the expressions for the generalized radius functions corresponding to spherical palms, and solve for $\dot{s}_1$. Since equation 3.29 is linear in $\dot{s}_1$ and equation 4.86 quadratic in $\dot{s}_1$, there exist *potentially* two solutions for $\dot{s}_1$. Since we assume the object is convex, we have the constraint $v_1 > 0$. This allows us to reject solutions for $\dot{s}_1$ that make v_1 negative. In a similar manner we can obtain a solution for $\dot{s}_2$.

The dynamics of a system with circular palms are almost the same as before. The only difference is that the moment arm for the moment of contact force i around the origin is no longer equal to s_i. The palm dynamics are now (cf. equations 4.7 and 4.8):

$$I_1\alpha_1 = \tau_1 - f_{c_1}(\boldsymbol{c}_1\cdot\bar{\boldsymbol{t}}_1) = \tau_1 - f_{c_1} b_1\sin(s_1/b_1) \tag{4.88}$$

$$I_2\alpha_{12} = \tau_2 - f_{c_2}(\boldsymbol{c}_2\cdot\bar{\boldsymbol{t}}_2) = \tau_2 + f_{c_2} b_2\sin(s_2/b_2) \tag{4.89}$$

The solutions for $\boldsymbol{a}_0$ and α_0 are therefore (cf. equations 4.9 and 4.10):

$$\boldsymbol{a}_0 = -\frac{I_1\alpha_1-\tau_1}{m b_1\sin(s_1/b_1)}\bar{\boldsymbol{n}}_1 + \frac{I_2\alpha_{12}-\tau_2}{m b_2\sin(s_2/b_2)}\bar{\boldsymbol{n}}_2 + \boldsymbol{g} \tag{4.90}$$

$$\alpha_0 = \frac{I_1\alpha_1-\tau_1}{m\rho^2 b_1\sin(s_1/b_1)}d_1 - \frac{I_2\alpha_{12}-\tau_2}{m\rho^2 b_2\sin(s_2/b_2)}d_2 \tag{4.91}$$

By definition, $\dot{\omega}_0 = \alpha_0$, but we also need these solutions for the solution of $\dot{s}_1$ and $\dot{s}_2$. Proving observability given these solutions is rather cumbersome, due to the nonlinearity of the palms. We plan to implement this model in future work. In this implementation we will check for observability numerically. It seems likely that the system is observable for most combinations of b_1 and b_2. So we may be able to choose the radii of the palms such that the system is observable.

Discussion Changing the palm shapes raises many new questions. Intuitively, it seems that circular palms would be 'better' than flat palms, but can we quantify

this? It also seems that with circular palms we would have to move the palms less to reach a point on the surface.

Suppose the palms are actually deformable, in a controlled way, by varying b_i. By reducing b_i from ∞ to 0 while maintaining the same tangent plane at the contact point we can probe the local curvature without having to move the shape. For this approach to work the palms need to be small enough to avoid collisions with parts of the shape that are not in a small neighborhood around the contact point. We think it will be interesting to find out how these enveloping grasps can be used if the palms are *not* small and if the contact points *do* change while the object is enveloped.

Chapter 5

Shape Reconstruction in Three Dimensions

Naturally we would like to extend the results from the previous chapters to three dimensions. We can manipulate an object with three planar palms as shown in figure 1.2. Although the same general approach can also be used for the 3D case, there are also some fundamental differences. First of all, the difference is not just one extra dimension, but three extra dimensions: a planar object has three degrees of freedom and a 3D object has six. Analogous to the planar case we can derive a constraint on the position of the center of mass if we assume quasistatic dynamics. This constraint gives us the X and Y coordinate of the center of mass (see appendix A.3). However, assuming quasistatic dynamics is not sufficient to solve for the motion of an unknown object: if the object stays in contact with the palms it has *three* degrees of freedom, but the quasistatic dynamics give us only *two* constraints. Another difference from the planar case is that in 3D we cannot *completely* recover an unknown shape in finite time, since the contact points trace out only curves on the surface of the shape.

5.1 Notation

In 3D the surface of an arbitrary smooth convex object can be parameterized with spherical coordinates $\boldsymbol{\theta} = (\theta_1, \theta_2) \in [0, 2\pi) \times [-\frac{\pi}{2}, \frac{\pi}{2}]$. For a given $\boldsymbol{\theta}$ we define the following right-handed coordinate frame

$$\boldsymbol{t}_1(\boldsymbol{\theta}) = \begin{pmatrix} -\sin\theta_1 \\ \cos\theta_1 \\ 0 \end{pmatrix} \tag{5.1}$$

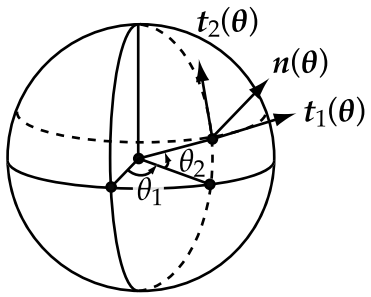


Figure 5.1: The coordinate frame defined using spherical coordinates

$$\boldsymbol{t}_2(\boldsymbol{\theta}) = \begin{pmatrix} -\cos\theta_1 \sin\theta_2 \\ -\sin\theta_1 \sin\theta_2 \\ \cos\theta_2 \end{pmatrix} \tag{5.2}$$

$$\boldsymbol{n}(\boldsymbol{\theta}) = \begin{pmatrix} \cos\theta_1 \cos\theta_2 \\ \sin\theta_1 \cos\theta_2 \\ \sin\theta_2 \end{pmatrix} \tag{5.3}$$

See figure 5.1. The function $\boldsymbol{x} : S^2 \to \mathbb{R}^3$ describing a smooth shape can then be defined as follows. The vector $\boldsymbol{x}(\boldsymbol{\theta})$ is defined as the vector from the center of mass to the point on the shape where the surface normal is equal to $\boldsymbol{n}(\boldsymbol{\theta})$.

The *contact support function* $(r(\boldsymbol{\theta}), d(\boldsymbol{\theta}), e(\boldsymbol{\theta}))$ is defined as

$$r(\boldsymbol{\theta}) = \boldsymbol{x}(\boldsymbol{\theta}) \cdot \boldsymbol{n}(\boldsymbol{\theta}), \qquad d(\boldsymbol{\theta}) = \boldsymbol{x}(\boldsymbol{\theta}) \cdot \boldsymbol{t}_1(\boldsymbol{\theta}), \qquad e(\boldsymbol{\theta}) = \boldsymbol{x}(\boldsymbol{\theta}) \cdot \boldsymbol{t}_2(\boldsymbol{\theta})$$

The function $r(\boldsymbol{\theta})$ is called the radius function. As in the planar case, the radius function completely describes the surface. The other two components of the contact support function appear in the derivatives of the radius function:

$$\begin{aligned} \frac{\partial r}{\partial \theta_1} &= \frac{\partial \boldsymbol{x}}{\partial \theta_1} \cdot \boldsymbol{n} + \boldsymbol{x} \cdot \frac{\partial \boldsymbol{n}}{\partial \theta_1} \\ &= 0 + (\boldsymbol{x} \cdot \boldsymbol{t}_1)\cos\theta_2 = d\cos\theta_2 \end{aligned} \tag{5.4}$$

$$\begin{aligned} \frac{\partial r}{\partial \theta_2} &= \frac{\partial \boldsymbol{x}}{\partial \theta_2} \cdot \boldsymbol{n} + \boldsymbol{x} \cdot \frac{\partial \boldsymbol{n}}{\partial \theta_2} \\ &= 0 + (\boldsymbol{x} \cdot \boldsymbol{t}_2) = e \end{aligned} \tag{5.5}$$

By definition, the partial derivatives of $\boldsymbol{x}$ lie in the tangent plane, so the dot product of the partials with the normal is equal to 0. Let $\boldsymbol{\theta}_i = (\theta_{i1}, \theta_{i2})^T$ denote the surface parameters for contact point i. Below we will drop the argument $\boldsymbol{\theta}_i$ and replace it with a subscript i where it does not lead to confusion. For

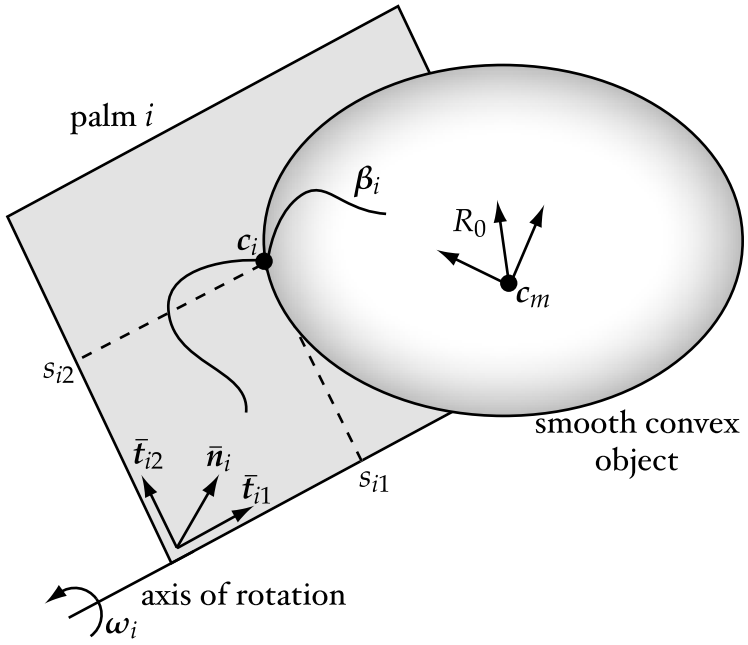


Figure 5.2: Illustration of the notation.

instance, we will write n_i for $n(\theta_i)$, the surface normal at contact point i in body coordinates.

The palms are modeled as three planes. The point of intersection of these planes is the origin of the world frame. Let us assume we can rotate each plane around the line of intersection with the horizontal plane through the origin. For each palm we can define a right-handed frame as follows. Let $\bar{n}_i$ be the normal to palm i (in world coordinates) pointing toward the object, let $\bar{t}_{i2}$ be the tangent normal to the axis of rotation and pointing in the positive Z direction and let $\bar{t}_{i1}$ be $\bar{t}_{i2} \times \bar{n}_i$. Then $R_i = [\bar{t}_{i1}, \bar{t}_{i2}, \bar{n}_i]$ is a right-handed frame for palm i. The configuration of the palms is completely described by the three rotation matrices R_1, R_2, and R_3. Let s_i denote the coordinates in palm frame i of contact point c_i, so that $R_i s_i = c_i$. Note that the third component of s_i is always zero; by definition the distance of the contact point along the normal is zero. See figure 5.2 for an illustration.

The position and orientation of the unknown object are described by c_m and R_0, respectively. The center of mass is located at c_m. The object coordinate frame defined by R_0 is chosen such that it coincides with the principal axes of

inertia. The inertia matrix $\mathcal{I}$ can then be written as

$$\mathcal{I} = m \begin{pmatrix} \rho_x^2 & 0 & 0 \\ 0 & \rho_y^2 & 0 \\ 0 & 0 & \rho_z^2 \end{pmatrix},$$

where m is the mass of the object, and the ρ's correspond to the radii of gyration. We will write $\boldsymbol{\beta}_i$ for the curve traced out on the surface of the object by contact point i. So $\boldsymbol{\beta}_i(t) = \boldsymbol{x}\big(\boldsymbol{\theta}_i(t)\big)$.

5.2 Local Shape

We can recover the local shape at the contact points by considering the distances between the contact points and the rates at which they change. We can write the constraint that the object maintains contact with each palm as

$$\boldsymbol{c}_i = \boldsymbol{c}_m + R_0 \boldsymbol{\beta}_i, \qquad i = 1, 2, 3 \tag{5.6}$$

The velocity of contact point i is therefore

$$\dot{\boldsymbol{c}}_i = \dot{\boldsymbol{c}}_m + \boldsymbol{\omega}_0 \times R_0 \boldsymbol{\beta}_i + R_0 \dot{\boldsymbol{\beta}}_i \tag{5.7}$$

The difference between two contact point velocities is

$$\dot{\boldsymbol{c}}_i - \dot{\boldsymbol{c}}_j = \boldsymbol{\omega}_0 \times R_0(\boldsymbol{\beta}_i - \boldsymbol{\beta}_j) + R_0(\dot{\boldsymbol{\beta}}_i - \dot{\boldsymbol{\beta}}_j) \tag{5.8}$$

$$= \boldsymbol{\omega}_0 \times (\boldsymbol{c}_i - \boldsymbol{c}_j) + R_0(\dot{\boldsymbol{\beta}}_i - \dot{\boldsymbol{\beta}}_j) \tag{5.9}$$

Since we assume the object is smooth, we have that $\boldsymbol{n}_i \cdot \dot{\boldsymbol{\beta}}_i = 0$. Furthermore, the palm normals and object are related by the object orientation matrix:

$$\bar{\boldsymbol{n}}_i = -R_0 \boldsymbol{n}_i \tag{5.10}$$

since $\bar{\boldsymbol{n}}_i$ is in world coordinates and $\boldsymbol{n}_i$ is in object coordinates. We can combine these constraints to solve for $\dot{\boldsymbol{\beta}}_i$:

$$\bar{\boldsymbol{n}}_i \cdot R_0 \dot{\boldsymbol{\beta}}_i = 0 \tag{5.11}$$

$$\bar{\boldsymbol{n}}_j \cdot R_0 \dot{\boldsymbol{\beta}}_i = \bar{\boldsymbol{n}}_j \cdot \big(\dot{\boldsymbol{c}}_i - \dot{\boldsymbol{c}}_j - \boldsymbol{\omega}_0 \times (\boldsymbol{c}_i - \boldsymbol{c}_j)\big) \tag{5.12}$$

$$\bar{\boldsymbol{n}}_k \cdot R_0 \dot{\boldsymbol{\beta}}_i = \bar{\boldsymbol{n}}_k \cdot \big(\dot{\boldsymbol{c}}_i - \dot{\boldsymbol{c}}_k - \boldsymbol{\omega}_0 \times (\boldsymbol{c}_i - \boldsymbol{c}_k)\big), \tag{5.13}$$

such that i, j, and k are distinct. Let Q be the defined as the 3×3 matrix with entries q_{ji}:

$$q_{ji} = \bar{\boldsymbol{n}}_j \cdot \big(\dot{\boldsymbol{c}}_i - \dot{\boldsymbol{c}}_j - \boldsymbol{\omega}_0 \times (\boldsymbol{c}_i - \boldsymbol{c}_j)\big) \tag{5.14}$$

Then we can write the solution for $\dot{\boldsymbol{\beta}}_1$, $\dot{\boldsymbol{\beta}}_2$, and $\dot{\boldsymbol{\beta}}_3$ more compactly as

$$\begin{pmatrix}\dot{\boldsymbol{\beta}}_1 & \dot{\boldsymbol{\beta}}_2 & \dot{\boldsymbol{\beta}}_3\end{pmatrix} = B^{-1}Q, \tag{5.15}$$

where $B = \begin{pmatrix}\bar{\boldsymbol{n}}_1 & \bar{\boldsymbol{n}}_2 & \bar{\boldsymbol{n}}_3\end{pmatrix}^T R_0$. As long as the palms are in general position, B will be invertible. Equation 5.15 describes the curves traced out by the contact points on the surface of the object (in body coordinates) as a function of the motion of the palms, the sensor values and the motion of the object. Note that these curves are not independent of each other. We know the configurations of the palms and the sensor values. If we also know one of the curves and the motion of the object, we can reconstruct the other curves. Below we will show that we can reconstruct all three curves by solving for the values of the radius function along the curves. Using equations 5.4 and 5.5, the derivative with respect to time of the radius function at contact point i is

$$\dot{r}_i = \frac{\partial r}{\partial \theta_{i1}}\dot{\theta}_{i1} + \frac{\partial r}{\partial \theta_{i2}}\dot{\theta}_{i2} = \begin{pmatrix} d_i \cos\theta_{i2} \\ e_i \end{pmatrix} \cdot \dot{\boldsymbol{\theta}}_i \tag{5.16}$$

We will rewrite the right-hand side of this equation as a function of the motion of the palms and the object, and the values of the radius function at the contact points. Using the position constraints we can rewrite d_i and e_i as a function of the configuration of the palms and r_i. We can write $\boldsymbol{\beta}_i$ as

$$\boldsymbol{\beta}_i = r_i\boldsymbol{n}_i + d_i\boldsymbol{t}_{i1} + e\boldsymbol{t}_{i2} \tag{5.17}$$

The vector between contact point i and contact point j is then

$$\boldsymbol{c}_i - \boldsymbol{c}_j = R_0(\boldsymbol{\beta}_i - \boldsymbol{\beta}_j) = R_0(r_i\boldsymbol{n}_i + d_i\boldsymbol{t}_{i1} + e\boldsymbol{t}_{i2} - r_j\boldsymbol{n}_j + d_j\boldsymbol{t}_{j1} + e\boldsymbol{t}_{j2}) \tag{5.18}$$

By rearranging terms we can obtain the following solution for the d's and e's:

$$\begin{pmatrix} d_1 \\ e_1 \\ d_2 \\ e_2 \\ d_3 \\ e_3 \end{pmatrix} = \begin{pmatrix} & & & & 0 & 0 \\ \boldsymbol{t}_{11} & \boldsymbol{t}_{12} & -\boldsymbol{t}_{21} & -\boldsymbol{t}_{22} & 0 & 0 \\ & & & & 0 & 0 \\ & & 0 & 0 & & \\ \boldsymbol{t}_{11} & \boldsymbol{t}_{12} & 0 & 0 & -\boldsymbol{t}_{31} & -\boldsymbol{t}_{32} \\ & & 0 & 0 & & \end{pmatrix}^{-1} \begin{pmatrix} R_0^T(\boldsymbol{c}_1 - \boldsymbol{c}_2) - (r_1\boldsymbol{n}_1 - r_2\boldsymbol{n}_2) \\ R_0^T(\boldsymbol{c}_1 - \boldsymbol{c}_3) - (r_1\boldsymbol{n}_1 - r_3\boldsymbol{n}_3) \end{pmatrix} \tag{5.19}$$

'Hidden' in the tangent vectors are the $\boldsymbol{\theta}_i$'s. Using equation 5.3 we can write $\boldsymbol{\theta}_i$ as a function of the palm surface normal $\boldsymbol{n}_i$:

$$\boldsymbol{\theta}_i = \begin{pmatrix} \arctan(n_{i2}, n_{i1}) \\ \arcsin n_{i3} \end{pmatrix} \tag{5.20}$$

The relationship between the normal $\boldsymbol{n}_i$ and the orientations of the palms and object is given by equation 5.10. The expression for $\dot{r}_i$ also contains $\dot{\boldsymbol{\theta}}_i$. By considering the derivative of the normal $\boldsymbol{n}_i$ we can obtain simple expressions for $\dot{\boldsymbol{\theta}}_i$. On the one hand we have that

$$\dot{\boldsymbol{n}}_i = \frac{\partial \boldsymbol{n}}{\partial \theta_{i1}} \dot{\theta}_{i1} + \frac{\partial \boldsymbol{n}}{\partial \theta_{i2}} \dot{\theta}_{i2} = \dot{\theta}_{i1} \cos\theta_2 \boldsymbol{t}_{i1} + \dot{\theta}_{i2} \boldsymbol{t}_{i2}. \tag{5.21}$$

But we can also obtain $\dot{\boldsymbol{n}}_i$ by differentiating equation 5.10:

$$\boldsymbol{\omega}_i \times \bar{\boldsymbol{n}}_i = \dot{\bar{\boldsymbol{n}}}_i = -\boldsymbol{\omega}_0 \times R_0 \boldsymbol{n}_i - R_0 \dot{\boldsymbol{n}}_i \quad \Rightarrow \quad \dot{\boldsymbol{n}}_i = R_0^T \big((\boldsymbol{\omega}_0 - \boldsymbol{\omega}_i) \times \bar{\boldsymbol{n}}_i \big) \tag{5.22}$$

Here $\boldsymbol{\omega}_i$ is the rotational velocity of palm i. Combining these two expressions for $\dot{\boldsymbol{n}}_i$ we can write $\dot{\boldsymbol{\theta}}_i$ as

$$\dot{\boldsymbol{\theta}}_i = \begin{pmatrix} \boldsymbol{t}_{i1} / \cos\theta_{i2} & \boldsymbol{t}_{i2} \end{pmatrix}^T R_0^T \big((\boldsymbol{\omega}_0 - \boldsymbol{\omega}_i) \times \bar{\boldsymbol{n}}_i \big) \tag{5.23}$$

Let us now consider the constraints on the acceleration of the object induced by the three point contact assumption. This will provide us with an additional constraint on $\boldsymbol{\beta}_i$ and will give us some more insight into how the 3D case is fundamentally different from the planar case. By differentiating equation 5.7 we obtain the following constraint on the acceleration:

$$\ddot{\boldsymbol{c}}_i = \boldsymbol{a}_0 + \boldsymbol{\alpha}_0 \times R_0 \boldsymbol{\beta}_i + \boldsymbol{\omega}_0 \times (\boldsymbol{\omega}_0 \times R_0 \boldsymbol{\beta}_i + 2 R_0 \dot{\boldsymbol{\beta}}_i) + R_0 \ddot{\boldsymbol{\beta}}_i, \tag{5.24}$$

where $\boldsymbol{a}_0$ and $\boldsymbol{\alpha}_0$ are the acceleration and angular acceleration of the object. (We will solve for $\boldsymbol{a}_0$ and $\boldsymbol{\alpha}_0$ in the next section by analyzing the dynamics.) Observe that from differentiation of the smoothness constraint $\dot{\boldsymbol{\beta}}_i \cdot \boldsymbol{n}_i = 0$ it follows that $\ddot{\boldsymbol{\beta}}_i \cdot \boldsymbol{n}_i = -\dot{\boldsymbol{\beta}}_i \cdot \dot{\boldsymbol{n}}_i$. We can therefore rewrite the acceleration constraint in the normal direction as a constraint on $\dot{\boldsymbol{\beta}}_i$. First, we rewrite the terms containing $\dot{\boldsymbol{\beta}}_i$ and $\ddot{\boldsymbol{\beta}}_i$:

$$\begin{aligned} \bar{\boldsymbol{n}}_i \cdot (\boldsymbol{\omega}_0 \times 2 R_0 \dot{\boldsymbol{\beta}}_i) + \bar{\boldsymbol{n}}_i \cdot R_0 \ddot{\boldsymbol{\beta}}_i &= 2 (\bar{\boldsymbol{n}}_i \times \boldsymbol{\omega}_0) \cdot R_0 \dot{\boldsymbol{\beta}}_i + R_0 (\dot{\boldsymbol{n}}_i \cdot \dot{\boldsymbol{\beta}}_i) \\ &= 2 (\bar{\boldsymbol{n}}_i \times \boldsymbol{\omega}_0) \cdot R_0 \dot{\boldsymbol{\beta}}_i + ((\boldsymbol{\omega}_0 - \boldsymbol{\omega}_i) \times \bar{\boldsymbol{n}}_i) \cdot R_0 \dot{\boldsymbol{\beta}}_i \\ &= (\bar{\boldsymbol{n}}_i \times (\boldsymbol{\omega}_0 + \boldsymbol{\omega}_i)) \cdot R_0 \dot{\boldsymbol{\beta}}_i \end{aligned}$$

The constraint on $\dot{\boldsymbol{\beta}}_i$ is therefore

$$(\bar{\boldsymbol{n}}_i \times (\boldsymbol{\omega}_0 + \boldsymbol{\omega}_i)) \cdot R_0 \dot{\boldsymbol{\beta}}_i = \bar{\boldsymbol{n}}_i \cdot \big(\ddot{\boldsymbol{c}}_i - \boldsymbol{a}_0 - \boldsymbol{\alpha}_0 \times R_0 \boldsymbol{\beta}_i - \boldsymbol{\omega}_0 \times (\boldsymbol{\omega}_0 \times R_0 \boldsymbol{\beta}_i) \big) \tag{5.25}$$

Let us now consider what how the acceleration constraint describes certain time-independent shape properties of the contact curves. The *Darboux frame field* (Spivak, 1999b) can be used to describe curves on a surface. For the curve $\boldsymbol{\beta}_i$ the Darboux frame field is defined by the unit tangent T of $\boldsymbol{\beta}_i$, the surface normal U restricted to $\boldsymbol{\beta}_i$, and $V = U \times T$. The normal U coincides with $\boldsymbol{n}_i$. Note that the normal of the *curve* does not necessarily coincide with the normal of the *surface*. Similar to the Frenet frame field, the derivatives of T, V, and U can be expressed in terms of T, V, and U:

$$\dot{T} = v(\quad \kappa_g V + \kappa_n U) \tag{5.26}$$

$$\dot{V} = v(-\kappa_g T \quad + \tau_g U) \tag{5.27}$$

$$\dot{U} = v(-\kappa_n T - \tau_g V \quad) \tag{5.28}$$

Here $v = \|\dot{\boldsymbol{\beta}}_i\|$ is the velocity of the curve, κ_g the *geodesic curvature*, κ_n the *normal curvature*, and τ_g the *geodesic torsion*. The geodesic curvature at a point describes the 'bending' of the curve in the tangent plane of the surface at that point. The normal curvature at a point describes the 'bending' of the curve in the surface normal direction. Using this frame field we can write $\ddot{\boldsymbol{\beta}}_i$ as

$$\ddot{\boldsymbol{\beta}}_i = \dot{v}T + v\dot{T} = \dot{v}T + v^2(\kappa_g V + \kappa_n U) \tag{5.29}$$

So by taking the dot product with the normal on both sides of the acceleration constraint we can obtain a constraint on the normal curvature κ_n of the curve:

$$\kappa_n = \ddot{\boldsymbol{\beta}}_i \cdot U = \ddot{\boldsymbol{\beta}}_i \cdot \boldsymbol{n}_i = -\dot{\boldsymbol{\beta}}_i \cdot \dot{\boldsymbol{n}}_i \tag{5.30}$$

On the right-hand side we can substitute the solutions from equations 5.15 and 5.22 for $\dot{\boldsymbol{\beta}}_i$ and $\dot{\boldsymbol{n}}_i$, respectively. In the planar case the velocity of the curve is equal to the radius of curvature, and the acceleration constraint determines the (normal) curvature at the contact points. In the 3D case, the acceleration constraint puts a constraint on the normal curvature at the contact points. But now we have two extra curve shape parameters, κ_g and τ_g, which are equal to zero in the planar case. In other words, in 3D the contact point curves are less constrained than in 2D. Let the state of the system in 3D be defined as $\boldsymbol{q} = (r_1, r_2, r_3, R_0, \boldsymbol{\omega}_0, \boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, \phi_1, \phi_2, \phi_3)^T$, where ϕ_i is the angle between palm i and the X-Y plane. Then having fewer constraints in 3D means that we cannot write the behavior of the system in state-space form $\dot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \boldsymbol{u})$, where $\boldsymbol{u}$ is a vector of control inputs. More specifically, we can not write $\dot{\boldsymbol{s}}_i$ as a function of state variables and controls.

5.3 Dynamics

The solution for the shape of the object depends on the motion of the object. We can solve for the motion by writing out the dynamics equations for the system formed by the palms and the object. Let I_i be the moment of inertia of palm i around its axis of rotation, α_i the angular acceleration around that axis, τ_i the torque produced by palm i's motor at the axis of rotation, and f_i the magnitude of the contact force. Then the motion of palm i is described by

$$I_i\alpha_i = \tau_i - f_i s_{i2}, \tag{5.31}$$

where s_{i2} is the second component of $\boldsymbol{s}_i$. From the definition of the palm frame it follows that s_{i2} measures the distance to the axis of rotation (see also figure 5.2). The net force and net torque on the object are given by Newton's and Euler's equations:

$$\boldsymbol{F}_0 = m\boldsymbol{a}_0 = \boldsymbol{F}_g + \sum_{i=1}^{3} f_i \bar{\boldsymbol{n}}_i \tag{5.32}$$

$$\boldsymbol{\tau}_0 = \mathcal{I}'\boldsymbol{\alpha}_0 + \boldsymbol{\omega}_0 \times \mathcal{I}'\boldsymbol{\omega}_0 = \sum_{i=1}^{3} \boldsymbol{\tau}_{c_i} \tag{5.33}$$

where

$$\mathcal{I}' = R_0 \mathcal{I} R_0^T \tag{5.34}$$

$$\boldsymbol{\tau}_{c_i} = (R_0\boldsymbol{\beta}_i) \times (f_i\bar{\boldsymbol{n}}_i) = -f_i R_0(\boldsymbol{\beta}_i \times \boldsymbol{n}_i) \tag{5.35}$$

From these equations we can solve for the angular acceleration of the object, $\boldsymbol{\alpha}_0$:

$$\boldsymbol{\alpha}_0 = -\mathcal{I}'^{-1}\big(\boldsymbol{\omega}_0 \times \mathcal{I}'\boldsymbol{\omega}_0 + \sum_{i=1}^{3} \frac{\tau_i - I_i\alpha_i}{s_{i2}} R_0(\boldsymbol{\beta}_i \times \boldsymbol{n}_i)\big) \tag{5.36}$$

Let us assume we can control the palms to move at a constant rotational velocity. The angular acceleration terms α_i will then disappear. We can summarize the simultaneous solution for the shape and motion of an unknown smooth convex object manipulated by three flat palms with the following system of differential equations:

$$\dot{r}_i = (d_i \cos\theta_{i2},\, e_i)^T \cdot \dot{\boldsymbol{\theta}}_i, \qquad i = 1,2,3 \tag{5.37}$$

$$\dot{R}_0 = \hat{\boldsymbol{\omega}}_0 R_0 \tag{5.38}$$

$$\dot{\boldsymbol{\omega}}_0 = -\mathcal{I}'^{-1}\big(\boldsymbol{\omega}_0 \times \mathcal{I}'\boldsymbol{\omega}_0 + \sum_{i=1}^{3} \frac{\tau_i}{s_{i2}} R_0(\boldsymbol{\beta}_i \times \boldsymbol{n}_i)\big) \tag{5.39}$$

Here $\hat{\omega}_0$ is the matrix form of the cross product, i.e., the 3×3 matrix such that $\hat{\omega}_0 \boldsymbol{p} = \boldsymbol{\omega}_0 \times \boldsymbol{p}$ for any vector $\boldsymbol{p}$. Equation 5.37 describes the shape of the object at the contact points. Equations 5.38 and 5.39 describe the dynamics of the object. We can replace the variables d_i, e_i, θ_{i2}, $\dot{\boldsymbol{\theta}}_i$, and $\boldsymbol{\beta}_i$ with the solutions given in equations 5.19, 5.20, and 5.23 so that the system of differential equations only depends on the values of the radius function at the contact points, palm configurations and sensor values. This allows us to integrate the system of differential equations given the palm configurations and sensor values.

5.4 Integrating Rotations

Numerical integration of the above system of differential equations will result in very unstable behavior due to the highly redundant representation of orientations: a orientation matrix uses nine numbers to represent three degrees of freedom. There exist many three-variable parameterizations of orientations, but they all suffer from singularities. In 1843, the mathematician W.R. Hamilton invented quaternions: four-variable representations of orientation that do not suffer from singularities. For a brief introduction to quaternions, see (Chou, 1992). Although quaternions were originally formulated as a generalization of complex numbers, for our purposes it will be more convenient to think of them as consisting of a scalar part and a vector part. Let $\mathbf{p} = (p_0, \boldsymbol{p})$ and $\mathbf{q} = (q_0, \boldsymbol{q})$ be two quaternions. Then the *quaternion product* is defined as

$$\mathbf{pq} = (p_0 q_0 - \boldsymbol{p} \cdot \boldsymbol{q}, p_0 \boldsymbol{q} + q_0 \boldsymbol{p} + \boldsymbol{p} \times \boldsymbol{q}) \tag{5.40}$$

The *conjugate* $\mathbf{q}^*$ of a quaternion $\mathbf{q}$ is defined as $\mathbf{q}^* = (q_0, -\boldsymbol{q})$. The *norm* of a quaternion $\mathbf{q}$ is defined as $\|\mathbf{q}\| = \mathbf{q}\mathbf{q}^* = q_0^2 + \|\boldsymbol{q}\|^2$.

We will represent rotations with *unit quaternions* (i.e, $\|\mathbf{q}\| = 1$). Unit quaternions are also known as Euler parameters (not to be confused with Euler angles). Let $\mathbf{q}$ be a unit quaternion and let $\mathbf{v} = (0, \boldsymbol{v})$ be a vector quaternion (i.e., a quaternion with the scalar part equal to zero). We can write $\mathbf{q}$ as

$$\mathbf{q} = \big(\cos\frac{\theta}{2}, (\sin\frac{\theta}{2})\boldsymbol{u}\big), \tag{5.41}$$

where $\boldsymbol{u}$ is a unit vector. This quaternion represents the orientation that is rotated about $\boldsymbol{u}$ by θ with respect to the world frame. It can be shown that the vector part of $\mathbf{v}' = \mathbf{q}\mathbf{v}\mathbf{q}^*$ is equal to the vector $\boldsymbol{v}$ rotated around $\boldsymbol{u}$ by θ. From this expression it is clear that $\mathbf{q}$ and $-\mathbf{q}$ correspond to the same orientation. So there exists a two-to-one mapping from quaternion parameters to a rotation and a nonsingular

one-to-two mapping from rotations to unit quaternions. The rotation matrix R corresponding to $\mathbf{q}$ is equal to

$$R = (\cos\theta)I + (1 - \cos\theta)\boldsymbol{u}\boldsymbol{u}^T + (\sin\theta)\hat{\boldsymbol{u}},$$

where $\hat{\boldsymbol{u}}$ is the matrix representation of the cross product: $\hat{\boldsymbol{u}}\boldsymbol{p} = \boldsymbol{u} \times \boldsymbol{p}$, for any vector $\boldsymbol{p}$.

Let $\mathbf{q}$ be a unit quaternion representing the orientation of the object being manipulated by the three palms. The relationship between the rotational velocity $\boldsymbol{\omega}_0$ and $\dot{\mathbf{q}}$ is given by (Chou, 1992)

$$\boldsymbol{\omega}_0 = 2\dot{\mathbf{q}}\mathbf{q}^* \quad \text{or} \quad \dot{\mathbf{q}} = \tfrac{1}{2}\boldsymbol{\omega}_0\mathbf{q},$$

This is a slight abuse of notation: we do not distinguish between the three-vector $\boldsymbol{\omega}_0$ and the vector quaternion with $\boldsymbol{\omega}_0$ as vector component. The relationship between the angular acceleration $\boldsymbol{\alpha}_0$ and $\ddot{\mathbf{q}}$ is given by (Chou, 1992)

$$\boldsymbol{\alpha}_0 = 2(\ddot{\mathbf{q}}\mathbf{q}^* + \dot{\mathbf{q}}\dot{\mathbf{q}}^*) \quad \text{or} \quad \ddot{\mathbf{q}} = (-\|\dot{\mathbf{q}}\|^2, \tfrac{1}{2}\boldsymbol{\alpha}_0)\mathbf{q}$$

Using these relationship we can easily move back and forth between the four-parameter quaternion space and rotations.

During the integration of rotational motion using quaternions we need to maintain certain invariants that may be violated due to numerical inaccuracies. First, for $\mathbf{q}$ to represent a valid orientation it needs to have unit norm. So at each integration step we need to renormalize $\mathbf{q}$. Second, the unit-norm constraint also puts a constraint on the rotational velocity. By differentiating the constraint $\mathbf{q}\mathbf{q}^* = 1$ we obtain the constraint

$$q_0\dot{q}_0 + \boldsymbol{q} \cdot \dot{\boldsymbol{q}} = 0$$

At each timestep let δ be equal to $q_0\dot{q}_0 + \boldsymbol{q} \cdot \dot{\boldsymbol{q}}$. Then using the corrected quaternion derivative $\dot{\mathbf{q}}' = \dot{\mathbf{q}} - \delta\mathbf{q}$ significantly improves the numerical stability.

5.5 Simulation Results

We have written a program to simulate the motion of an arbitrary smooth convex object supported by three planes. To reconstruct the shape we need to integrate out the system of differential equations given by equations 5.37–5.39. There are several reasons why straightforward integration is not likely to produce good results. First, as we described in the previous section, we should use quaternions for orientations to improve numerical stability and avoid singularities of three-parameter representations of orientations. Second, we would like to enforce *all*

the constraints on the contact point curves given by equations 5.15 and 5.25. By integrating the values of the radius function at the contact points, we would not be able to use these constraints. So instead, we integrate the $\dot{\beta}_i$'s. At each time step we compute r_i from β_i. We then compute d_i and e_i from equation 5.19. From r_i, d_i and e_i we compute a corrected estimate for β_i using equation 5.17. We can compute $\dot{\beta}_i$ from equations 5.15 and 5.25. These equations provide 12 linear constraints for 9 unknowns. We compute a least squares solution to this system of equations.

We define the error in the state estimate to be the weighted sum of (1) the residual of the least squares solution for $\dot{\beta}_i$, and (2) the differences between the estimate for β_i and the corrected estimate. We search for the initial conditions of the system of differential equations describing the shape and motion by minimizing the integral of the error over a short time interval starting at time $t = 0$. The search is split up in two stages. First we sample uniformly random over all possible orientations and all possible values of the radius function within a certain range. We keep track of the sample that results in a minimal error. In evaluating a new sample we can stop integration once the error integral exceeds a previously found minimum. In our simulations we would only have to take a couple of integration steps for each sample after having taken a few thousand samples. In this manner we can quickly evaluate thousands and thousands of guesses for the initial conditions. In the second stage we perform several Nelder-Mead simplex searches for the minimum error near the minimum found in the first stage. With each simplex search we take the minimum found in the previous search and lengthen the time interval over which the error is integrated. The purpose of this approach is twofold. First, for the simplex search to return a minimum the function we are minimizing needs to be sufficiently smooth. If we pick the length of the time interval to be large, the neighborhood around the true initial values for which this is true is very small. A small error in the initial conditions will make the differential equations very unstable after a number of integration steps, resulting in chaotic behavior and large values for the error integral. Second, evaluating the error function over a large time interval is very expensive. So we try to get close to a minimum by first minimizing the error function over smaller intervals. It is possible that our error metric converges to a local minimum, but this has not been observed in our simulations. If shape and motion aliasing is possible in general remains an open problem.

Figure 5.3 shows the motion of an ellipsoid supported by three palms. The palms are at 60 degree angles with the horizontal plane. The rotational axes are at 60 degree angles with each other. The radii of the axes of the ellipsoid are 2.5, 2, and 1. One problem with simulating this system without friction is

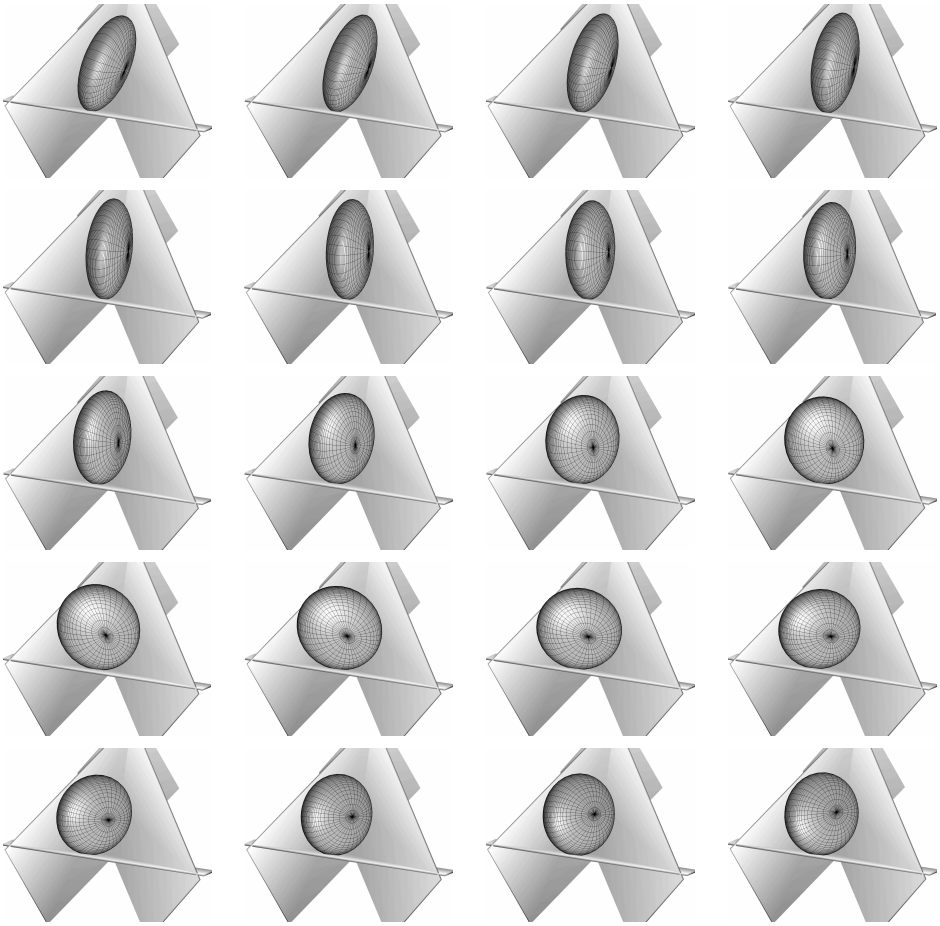
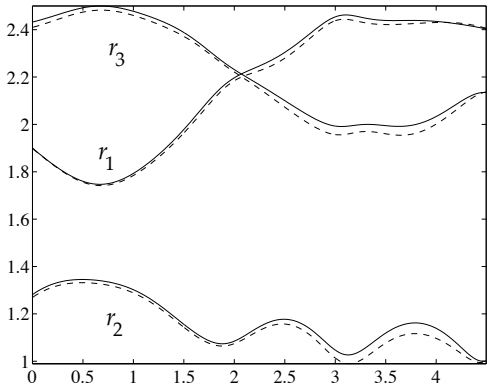


Figure 5.3: An object rolling and sliding on immobile palms with gravity and contact forces acting on it. The object is given some initial rotational velocity. The object is shown at $t = 0, 0.2, \ldots, 4.4$ (from left to right, top to bottom). The viewpoint is above the palms, looking down.

that if the palms are moving they continuously increase the kinetic energy of the object. In our simulations the ellipsoid breaks contact before a significant part of the object is recovered. To avoid this problem, we keep the palms in the same position and give the ellipsoid some initial velocity. In figure 5.3 the object has an initial rotational velocity of $(0, 0.3, 0.3)^T$. In the search for initial conditions we assume the initial rotational velocity is known. This is, of course, not very realistic. If we would model friction, then moving the palms would become feasible and we could start the system with the object at rest. In that case the rotational velocity would be zero. Our current program searches a six dimensional



(a) The radius function values at the contact points.

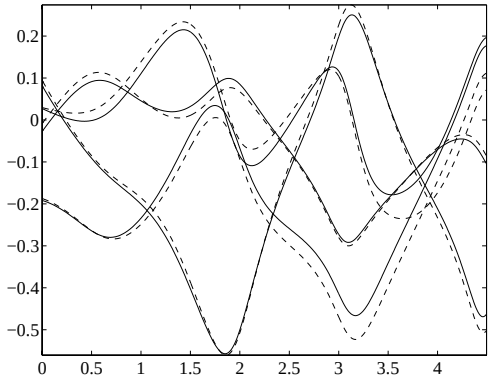

(b) The four quaternion components of the rotational velocity.

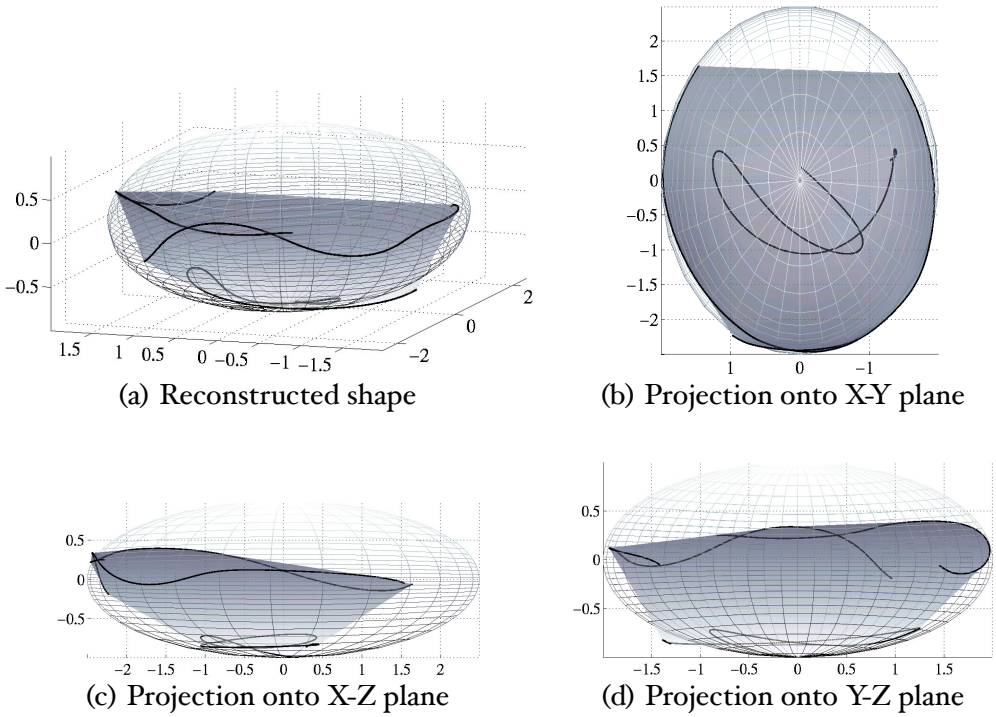
Figure 5.4: Differences between real and recovered shape and motion. Real values are plotted as solid lines, observed values as dashed lines.

space: three dimensions for orientation and three for the values of the radius function at the contact points. The reconstructed shape and motion are shown in figure 5.4. Our program found a different, but equally valid initial orientation for the ellipsoid. Because of the symmetries of the ellipsoid, rotations of 180 degrees around the principal axes result in orientations that are indistinguishable: the resulting rotational velocity and values for the radius function are identical (up to numerical error).

5.6 Shape Approximations

Given the recovered motion and contact curves we can also give a lower and an upper bound on the volume occupied by the entire shape. Since we assume the object is convex, the convex hull of the contact curves in the object frame is a lower bound on the shape. Figure 5.5 shows the lower bound on the shape obtained from the example in the previous section.

We can obtain an upper bound by observing that at each contact point the corresponding palm plane introduces a half-space constraint: the object has to lie entirely on one side of the plane. Clearly, the intersection of half-spaces along the contact curves in the object frame forms an upper bound on the shape. Computing the intersection of half-spaces can be reduced to a convex hull problem as follows. Suppose we have a set of n points on the contact curves and the unit normals at these points. Let $\boldsymbol{p}_i$, $i = 1, \ldots, n$ be one of the contact points and $\boldsymbol{n}_i$ its corresponding unit normal. The half-space constraint at point i



(a) Reconstructed shape (b) Projection onto X-Y plane

(c) Projection onto X-Z plane (d) Projection onto Y-Z plane

Figure 5.5: The convex hull of the contact curves gives a lower bound on the volume occupied by the object. The true shape of the object is shown as a wire-frame model.

is then

$$n_{ix}x + n_{iy}y + n_{iz}z - w_i < 0,$$

where $w_i = \boldsymbol{n}_i \cdot \boldsymbol{p}_i > 0$. We can dualize the tangent plane $n_{ix}x + n_{iy}y + n_{iz}z - w_i = 0$ to the point $\bar{\boldsymbol{p}}_i = (n_{ix}/w_i, n_{iy}/w_i, n_{iz}/w_i)^T$. The dual of the convex hull of the $\bar{\boldsymbol{p}}_i$'s is the desired intersection of the half-spaces. The dual of the convex hull can be computed by either computing the dual planes of all the vertices of the convex hull or by computing the dual points of the planes tangent to the faces of the convex hull. In the general case, when some w_i's are negative, this approach does not work and we need a more complex algorithm (that still runs in $O(n \log n)$ time, though). See (Preparata and Shamos, 1985) for details. Note that we can obtain slightly tighter upper and lower bounds if we are given bounds on the curvature of the object.

These bounds could form the basis for a manipulation strategy. Having bounds makes it possible to speculate about the outcomes of actions. Suppose we would

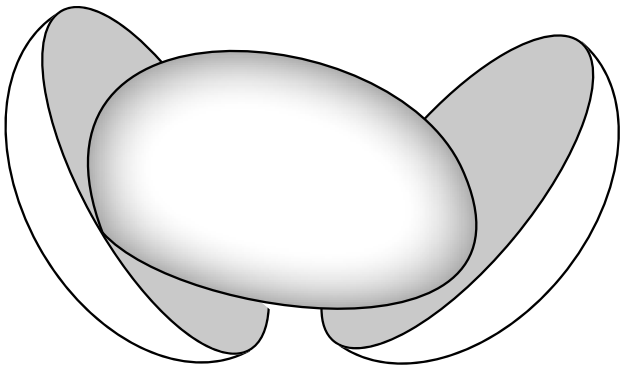


Figure 5.6: Two spherical palms holding an object.

like to estimate the entire shape. One objective for a planner could be to minimize the difference between the upper bound and lower bound. A simple 'greedy' planner would always try to move the contact points toward points on the lower bound with the largest distance to the upper bound. A smarter planner would also take into account the path length, such that the palms would minimize nearby differences between the bounds before trying to minimize faraway differences. This assumes we have some reasonable metric to measure distance between palm/object configurations. This manipulation strategy is reminiscent of some of the work in geometric probing, but unlike probing algorithms our manipulation strategy *continuously* 'probes' the object. The palms cannot instantaneously move from one configuration to another.

Let us now consider how these bounds change if we change the shape of the palms. We can use the shape of the palms to get tighter upper bounds on the shape of the unknown object. Suppose we have spherical palms. Figure 5.6 shows two spherical palms holding an object. Instead of taking intersections of half-spaces, we can now use intersections of spheres as an upper bound. The contact point on palm i can be written as

$$\boldsymbol{c}_i = b_i R_i \begin{pmatrix} \cos s_{i1} \cos s_{i2} \\ \sin s_{i1} \cos s_{i2} \\ \sin s_{i2} \end{pmatrix}, \tag{5.42}$$

where b_i is the radius of sphere/palm i, and (s_{i1}, s_{i2}) are the spherical sensor coordinates. We can obtain expressions for $\dot{\boldsymbol{c}}_i$ by differentiating the right-hand side of equation 5.42 with respect to R_i and $\boldsymbol{s}_i$. The solution for the dynamics of the object would change slightly with spherical palms, but the solution for the shape and motion of the object would remain structurally the same. Equations 5.14

and 5.15 show clearly how the solution for the shape of the object depends on the motion of the contact points.

If we allow some of the palms to be convex, we could actually recover concavities on the object provided the radius of curvature along the concavities is larger than the radius of the palm. Allowing the object to have concavities would invalidate the upper bound derived above. We can define a new upper bound as the intersection of half-spaces at those points on the contact curves that are also on the boundary of the convex hull of the curves. Assuming the palms are infinite rays this is a true upper bound even if there exists protrusions between different parts of the shape that have been recovered, because the protrusions cannot intersect the palms. We can also define other approximate representations of the shape of the object that may be closer to the true shape. Let us assume that instead of continuous contact curves we have a finite set of points along those curves. Based on this set of points we can define a family of α-shapes (Edelsbrunner and Mücke, 1994). One can think of α-shapes as a generalization of the convex hull. Suppose we compute a Delaunay triangulation of the set of points. Then informally, for a given value of α the corresponding α-shape is equal to the triangulation minus the triangles that a sphere with radius α can pass through without intersecting any of the vertices. The entire family of α-shapes for a set of n points can be computed in $O(n^2)$. This family oh α-shapes can be seen as a set of interpretations. Another way to represent the shape is to compute a smooth interpolated surface. There are many ways to smoothly interpolate a point set. We could use splines or iterative methods such as principal surfaces, an extension of principal curves (Hastie and Stuetzle, 1989; LeBlanc and Tibshirani, 1994). Splines and principal curves can also be used in the planar case, of course.

Chapter 6

Conclusion

6.1 Contributions

This thesis has shown how a robot can reconstruct the local shape of an unknown smooth convex object with strictly positive curvature along the surface. The object is manipulated by palms covered with tactile sensors. The object is not immobilized by the palms, but instead moves around as the contact forces and gravity act on it. This is a novel way to combine manipulation and tactile sensing that allows for a continuous interaction between manipulation and sensing.

In chapter 3 we presented the analysis, simulation results and experimental results for the quasistatic case. We derived expressions for the values of the radius function at the contact points and the rotational speed of the object. This completely describes the shape and motion of an unknown object as a system of differential equations. The simulation results showed that our approach works well. We showed that finding the initial conditions of this system of differential equations corresponds to a 1D search for the center of mass: the center of mass has to lie on the vertical line through the intersection of the lines of force at the contact points. Further research is needed on finding these initial conditions: we can find a minimum of an error measure along this line, but it is not known if this is a global minimum.

Our experimental results suggest that our approach could also work in practice. The reconstructed shape was close to the actual shape, despite significant errors in the reconstructed motion of our experimental object. These errors were caused in part by violations of the assumptions, the most important one being the no-friction assumption. Once we have modeled friction, we expect the performance of our shape reconstruction method to improve considerably.

It is possible that the motion of the object has discontinuities, even if the motion of the palms is continuous. This happens when a local minimum of the

potential energy of the object becomes a local maximum as the palms rotate. These discontinuities in the motion of the object result in different disconnected or overlapping pieces of the shape. We presented a method for piecing these different segments together. This method tries to minimize the errors in the constraints on the motion and shape of the object and at the same time it tries to minimize the area between segments that overlap. This method can be useful to consolidate large amounts of tactile data. With only a few small segments, this method will arrange the different segments to not overlap. In general, this may not be a correct interpretation of the sensor data.

The shape of unknown objects can be reconstructed up to symmetry in the radius function. Symmetries in the diameter function[1] do not pose a problem (like they do for parallel jaw grippers). It is possible, though, that there exist combinations of shapes and motions that result in the same sensor values. Since the mass properties are global properties it seems that this can only happen for very specific control inputs; changing the control inputs slightly should disambiguate different interpretations.

In chapter 4 we addressed the dynamic case, where force/torque balance is no longer assumed. We established that it is possible to reconstruct the shape locally in this case too. By using results from non-linear control theory, we showed that the system formed by the palms and the object is *observable*. This means that there exists an *observer* (or state estimator) that can correct small errors and filter out noise in the state estimate. We first proved observability of the general and case and then showed that certain special cases are also observable. Constructing an observer is nontrivial. We constructed an observer for the special case where the palms are moving at the same rate. This is based on Newton's method for root finding. This observer performed very well in simulation. If both palms are motionless, the local shape is still observable. However, the construction of an observer for this case is very nontrivial and of limited practical value.

In chapter 5 we showed that the results for planar shapes can be extended to three dimensions. One important difference from the planar case is that we can not expect a complete reconstruction of a 3D shape in finite time with point contacts. We presented various bounds and shape approximations that given the curves traced out by the contact points on the surface of the object give different interpretations of the global shape of the object. Another big difference with the planar case is that in 3D there are fewer constraints on the shape and motion of the object. As a result, the quasistatic approach from chapter 3 cannot be applied. We derived twelve constraints on the velocity of the three contact point curves.

[1] Reuleaux (1876) showed that there exist many different objects with the same *constant* diameter function.

	quasistatic	dynamic	constant speed	fixed palms
2D	I	OC	OD	OC
3D	X	I	I	I

Table 6.1: Summary of shape reconstruction results for different types of dynamics.

Since the coordinates of the curves at the contact points can be described with nine unknowns, we used the three additional constraints to improve the stability of the integration of the system dynamics. We do this by computing a least squares solution to the twelve constraints. Although we have extra constraints, we still do not have a large enough number of constraints to write the system dynamics in state-space form. As a result we cannot test for observability and the least-square solution described above is not guaranteed to be the best solution. A more abstract notion of observability is needed to describe what information the additional constraints give about the unknown state variables.

Table 6.1 summarizes the results of this thesis. In the quasistatic case force/torque balance is assumed. In the dynamic case we model all the forces and accelerations. The last two columns are special cases of the dynamic case. We do not assume any restrictions on the motion of the object, but put restrictions on the motions of the palms in these two cases. In the penultimate column the palms are assumed to move with constant speeds. The last case summarizes the case where the palms stay put. The meaning of the table entries is:

X: No solution exists.

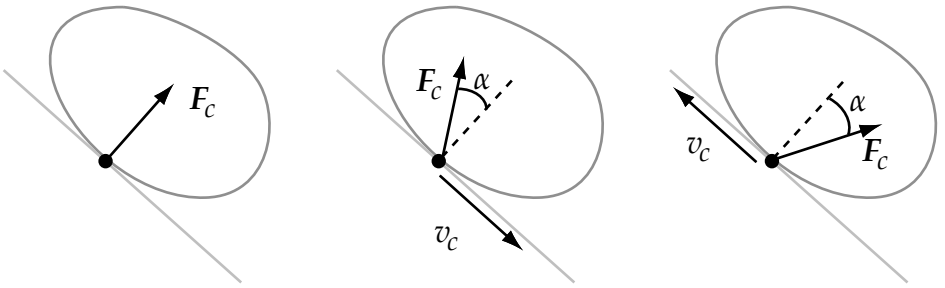
I: We can solve for the shape and motion by integration of a system of differential equations.

OC: The system is observable in terms of the control vector fields.

OD: The system is observable in terms of the drift vector field.

Systems that are observable can also be solved by integration. With pure integration we may only be able to *detect* an error in our state estimate. Using an observer is preferable to pure integration, because an observer may be able to *correct* small errors and filter out noise. In 2D two palms are sufficient, in 3D three palms.

Ultimately we would like to combine our approach with the work on manipulation by pure rolling by Bicchi and colleagues (Bicchi et al., 1999). We are working toward a framework that formally describes the shape and motion of an unknown object with changing contact modes as it is manipulated and sensed by a robot.



(a) No friction.

(b) With friction, object sliding right.

(c) With friction, object sliding left.

Figure 6.1: Direction of the contact force in the absence and presence of friction.

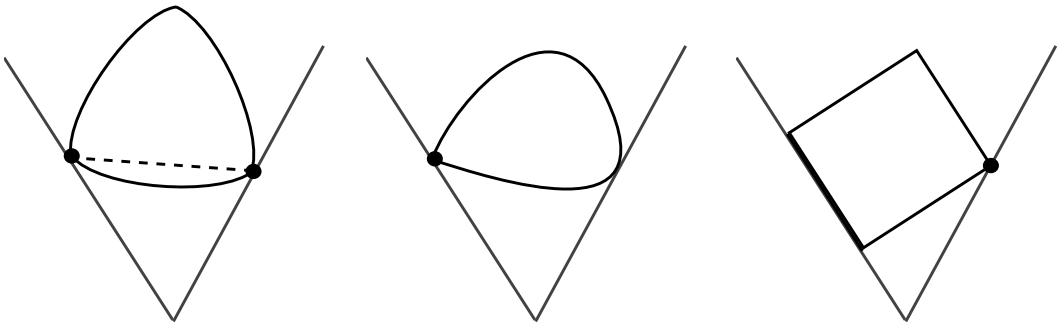
6.2 Future Directions

Removing Assumptions

In this thesis we assumed there was no friction and that the unknown object being manipulated by the palms was smooth and convex. In this section we will describe how these assumptions can be removed and how this changes the problem of reconstructing the shape and motion of the object.

Adding friction to the model does not fundamentally change the structure of the solutions for shape and motion. If a contact point is moving, the contact force lies on the edge of the friction cone, opposite the direction of the contact point velocity (see figure 6.1 for an illustration of the planar case). The tangential component is equal to the coefficient of friction, μ, times the normal component. In figure 6.1 $\tan\alpha$ is equal to μ. So if we would know μ we could solve for the acceleration and angular acceleration from the dynamics equations. The quasistatic approach from chapter 3 would no longer work: for a given palm configuration there is now an infinite number of stable poses. We plan to recover the value of the friction coefficient using a nonlinear observer by making μ part of the state vector (like the radius of gyration in chapter 4).

If one contact point is not moving (i.e., the contact force lies inside the friction cone), we can write the shape and motion of the other contact point relative to the fixed contact point. We only need position and velocity constraints for this (Erdmann, 1999). If neither contact point is moving, we can not observe anything about the shape unless we rotate the palms about the contact points. We can always force at least one of the contact points to change by varying the angles between the palms. Modeling friction means we have to distinguish all the



(a) Two edge-vertex contacts.

(b) One edge-vertex contact.

(c) One edge-edge contact, one edge-vertex contact.

Figure 6.2: Different types of contact.

different modes for both contact points: left sliding, right sliding, not moving. Instantaneously this may not be possible, but we hope that given enough sensor data we can find a shape such that its motion is consistent with all sensor data. We also need to determine whether the contact point is rolling or not.

If we remove the smoothness constraint, we also have to distinguish different cases. Figure 6.2 shows examples of non-smooth contact modes. Instantaneously we can not determine if a contact point is a corner of the object or not. However, we can write out the system behavior in state-space form for each of the different cases and construct an observer for each case. For a given state estimate we can run all observers in parallel and see which one minimizes the error in the output the most. We also need to detect when the contact mode changes. Note that the system can not instantaneously change from one contact mode to every other contact mode. This observation can be used to eliminate certain interpretations of the sensor data. Although we can model edge-edge contact, with our touchpads we can only measure the centroid of contact. With edge-edge contact we expect to see discontinuities in the sensor values, but we may not be able to distinguish these from the discontinuities due to discrete changes in orientation. Generally speaking, the non-smoothness provides additional constraints and features that we can use to eliminate uncertainty about the object's shape and motion. By assuming objects are smooth we are limiting ourselves to the hardest case; the solution for the shape and motion in all of the non-smooth cases is much simpler. What remains an open problem is whether we can effectively track the changes in contact modes.

Different Representations of Shape

The contact kinematics lead naturally to a description of the shape of an object in terms of the curvature at the contact points. One disadvantage of this approach is that the shape has an infinite number of degrees of freedom. Although this allows us to reconstruct *any* smooth convex shape, in many cases it may be sufficient to have a reasonable approximation. We could, for instance, pick n control points that are radially equally spaced. A closed spline interpolation through those n points is then one possible shape approximation. So now the shape has only n degrees of freedom. If we want to do something fancier, we could use Fourier and wavelet transforms. There has been a large amount of research in computer vision and pattern matching on representing shapes. With future research we would like to explore how results in those areas can be used for tactile shape reconstruction. In section 4.7 we mentioned alpha shapes and principal curves as approximate shape representations. We would also like to find ways to use these different representations to guide the manipulation.

Different Observers

Observer design for nonlinear systems is still an active area of research. The observer described in chapter 4 is relatively simple. It is easy to implement, but is computationally very expensive. In future research we would like to develop observers that do not rely as heavily on numerical methods. Many observers use the following approach. For a given state $\boldsymbol{q}$ we can define a transform $Z : \boldsymbol{q} \rightarrow \boldsymbol{z}$, such that the transformed system is in some canonical form. Often the transformed system looks like

$$\dot{\boldsymbol{z}} = A\boldsymbol{z} + \boldsymbol{f}(\boldsymbol{z}, \boldsymbol{u}) \tag{6.1}$$

$$\boldsymbol{y} = C\boldsymbol{z}. \tag{6.2}$$

That is, the system is almost linear, except for some nonlinearities that depend on the controls. To guarantee exponential convergence certain constraints on $\boldsymbol{f}$ need to be satisfied. In many cases the control inputs can be ignored, if we assume the control inputs are piecewise constant. Typically, the coordinate transform is valid only locally and needs to be recomputed at each time step. An example of this type of observer can be found in e.g. (Gauthier et al., 1992). Gauthier et al. assumed a single-output system without inputs, but Bornard and Hammouri (1991) showed that this observer can be generalized to multi-input-multi-output systems. If we assume the sensor values are reliable, we can construct a so-called reduced-order observer, which, as the name suggests, only tries to observe those state variables which are not part of the output (Besançon and Hammouri,

1995). The observers mentioned above all use a Lie-algebraic approach. Other common techniques include: Lyapunov methods (Thau, 1973; Kou et al., 1975), and linearization by nonlinear output injection (Krener and Isidori, 1983; Bestle and Zeitz, 1983). We would like to explore how these techniques can be applied to the tactile shape reconstruction problem.

Tactile SLAM

Simultaneous localization and mapping (SLAM) is an important problem in mobile robotics. The SLAM problem can be stated as: how can a mobile robot efficiently build a map of an unknown environment and at the same time keep track of where it is on the map? A common failure mode is that a robot returns to a place where it has been before, but classifies it as new place on the map. With the tactile shape reconstruction method proposed in this thesis we run into a similar problem. If the sensors cover the same part of the surface many times, there is no constraint that forces the contact point curves to lie on the same surface. So we need a strategy to consolidate all data into one surface (or a probability distribution of surfaces, if one prefers a probabilistic approach). Note that the presence of non-smooth features on the surface will make the tactile localization problem significantly easier (Okamura and Cutkosky, 2001). In section 3.7 we proposed a method to consolidate conflicting data by averaging, but we would like to do better. We expect that some results for the SLAM problem will transfer to tactile shape reconstruction.

In computer vision and pattern matching the localization problem is often called the registration (or correspondence) problem. From this area we may be able to borrow techniques to compare different shapes. For instance, we could use distance metrics that are invariant with respect to rotation, scale, or both (Arkin et al., 1991; Huttenlocher and Kedem, 1990).

Other Sensing Strategies

Besides the model of planar palms, there are many other ways to reconstruct the shape of an unknown moving object with tactile sensors. One extension is to change the shape of the palms. In the previous chapters we described how the model can be extended to circular and spherical palms. An advantage of concave palms is that with less manipulation we can cover more of the shape, since a grasp with these palms comes closer to an enveloping grasp. Further research is needed to quantify this claim. The minimum radius of curvature along the palms should be chosen such that there is always just one contact point on each palm. However, if the palms can detect multiple contacts, it makes more sense

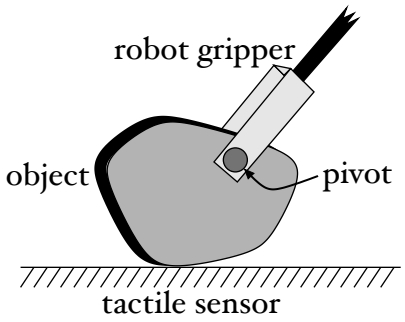


Figure 6.3: Dragging an object over a tactile sensor with a pivoting grasp.

to maximize the number of contact points (e.g., by using deformable palms).

Figure 6.3 shows an entirely different sensing strategy. An object is grasped by a robot arm using a pivoting gripper (Rao et al., 1994, 1995). With such a gripper the object is free to rotate around the line through the grasp contact points. The sensing strategy consists of dragging or pushing the object over a surface coated with tactile sensors. We think it would be interesting to determine whether this system is observable as well. Note that we need to be able to establish a stable grasp first.

It is also possible to combine techniques from pose and shape recognition with our approach. Jia and Erdmann (1999) describe how to recover the pose of a known object by pushing it with a finger covered with tactile sensors. With our palms we can do this by keeping some palms fixed and using one of the palms to push the objects. This could prove to be useful for reconstructing unknown objects as well. Suppose we have partially recovered an unknown shape. Then we can use Jia and Erdmann's approach to perform localization.

Another possibility is to turn the whole model (literally) upside-down: we can reconstruct a surface by driving on it while measuring the contact forces. It turns out that we can use the same approach we used for manipulating unknown objects for reconstructing unknown terrain as well. Suppose we have a planar mobile robot with two point-size wheels. Assume these wheels are in pure rolling contact (i.e., no slip) with a smooth unit-speed curve $\boldsymbol{x} : \mathbb{R} \to \mathbb{R}^2$. At each contact point i we define a coordinate frame $[\boldsymbol{t}_i, \boldsymbol{n}_i]$ using the local tangent and normal of the curve. Each frame can be characterized by the angle ϕ_i between the tangent and the horizontal axis. The only control of the robot is the torque of wheel 1. This torque produces a 'pulling force' along the tangential direction with magnitude f_0. See figure 6.4 for a sketch. The forces acting on the robot are gravity, normal forces and the force produced by the motor that is driving the front wheel. The net force on the robot is

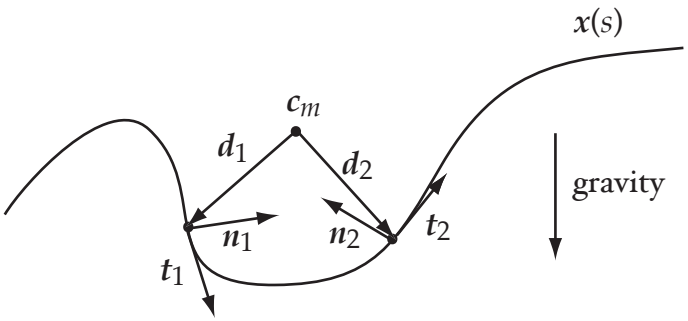


Figure 6.4: A mobile robot consisting of two mass-less rods $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ connected at the center of mass $\boldsymbol{c}_m$ in contact with a curve. The wheels are at the end points of $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$.

$$\boldsymbol{F} = f_0\boldsymbol{t}_1 + f_1\boldsymbol{n}_1 + f_2\boldsymbol{n}_2 + \boldsymbol{F}_g = m\boldsymbol{a}, \tag{6.3}$$

where f_0 is the magnitude of the force produced by the motor, f_1 and f_2 are the magnitudes of the normal forces and $\boldsymbol{F}_g$ is the gravitational force. The net torque on the robot is

$$\begin{aligned}\tau &= f_0\boldsymbol{d}_1 \times \boldsymbol{t}_1 + f_1\boldsymbol{d}_1 \times \boldsymbol{n}_1 + f_2\boldsymbol{d}_2 \times \boldsymbol{n}_2 \\ &= -f_0\boldsymbol{d}_1 \cdot \boldsymbol{n}_1 + f_1\boldsymbol{d}_1 \cdot \boldsymbol{t}_1 + f_2\boldsymbol{d}_2 \cdot \boldsymbol{t}_2 = m\rho^2\alpha,\end{aligned} \tag{6.4}$$

where $\boldsymbol{d}_i$, $i = 1, 2$, is the vector from the center of mass to wheel i. For simplicity, assume $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ are of equal length and make a right angle with each other.

We assume that the wheels always stay in contact with the curve. This imposes constraints on (derivatives of) the position of the robot. Let $\boldsymbol{c}_m$ denote the position of the center of mass, $\boldsymbol{x}_i$ the point on the curve in contact with wheel i, and v_i the velocity of contact point i. The position constraints for wheel i can be written as

$$\boldsymbol{c}_m + \boldsymbol{d}_i = \boldsymbol{x}_i \tag{6.5}$$

$$\boldsymbol{v} + \omega \times \boldsymbol{d}_i = v_i\boldsymbol{t}_i \tag{6.6}$$

$$\boldsymbol{a} + \alpha \times \boldsymbol{d}_i + \omega \times (\omega \times \boldsymbol{d}_i) = \dot{v}_i\boldsymbol{t}_i + \kappa_i v_i^2\boldsymbol{n}_i \tag{6.7}$$

The right-hand side of the last constraint follows from the Frenet formulas. We will assume that the control input f_0 is chosen such that the velocity at wheel 1 is constant. Let us the define the state $\boldsymbol{q}$ of the system as the vector $(\phi_0, \omega, \boldsymbol{x}_1, \boldsymbol{x}_2, v_2)^T$. The behavior of the system can then be described by the

following system of differential equations

$$\dot{\phi}_0 = \omega \tag{6.8}$$
$$\dot{\omega} = \alpha \tag{6.9}$$
$$\dot{\boldsymbol{x}}_1 = v_1 \boldsymbol{t}_1 \tag{6.10}$$
$$\dot{\boldsymbol{x}}_2 = v_2 \boldsymbol{t}_2 \tag{6.11}$$
$$\dot{v}_2 = (\boldsymbol{a} + \alpha \times \boldsymbol{d}_2 - \omega^2 \boldsymbol{d}_2) \cdot \boldsymbol{t}_2 \tag{6.12}$$

Note that since we drive wheel 1 at a constant speed, $\dot{v}_1$ is equal to zero and we do not need to make v_1 part of the state. It can be shown that if we can measure the magnitude of the one of the contact forces, we can solve for the acceleration and angular acceleration from the dynamics described above. We can measure the contact force with strain gauges. The above system is not in state-space form, since we do not have an expression for the rate of change in the contact force as a function of the state variables. Further research is needed to either reformulate this problem or construct another kind of state estimator with observer-like properties.

Appendix A

Derivations

A.1 Quasistatic Shape Reconstruction

Curvature at the Contact Points

The expressions for the curvature at the contact points can be found by differentiating the generalized contact support functions:

$$\begin{aligned}\tilde{r}_1' &= (\boldsymbol{x}_1' - \boldsymbol{x}_2') \cdot \boldsymbol{n}_1 + (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{n}_1' \\ &= v_2 \boldsymbol{t}_2 \cdot \boldsymbol{n}_1 - (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{t}_1 \\ &= -v_2 \sin\phi_2 - d_1 \end{aligned} \tag{A.1}$$

$$\begin{aligned}\dot{\tilde{r}}_1 &= (\dot{\boldsymbol{x}}_1 - \dot{\boldsymbol{x}}_2) \cdot \boldsymbol{n}_1 + (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \dot{\boldsymbol{n}}_1 \\ &= (\dot{\theta}\boldsymbol{x}_1' - (\dot{\theta} + \dot{\phi}_2)\boldsymbol{x}_2') \cdot \boldsymbol{n}_1 + (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot (\dot{\theta}\boldsymbol{n}_1') \\ &= -v_2(\dot{\theta} + \dot{\phi}_2)\sin\phi_2 - \dot{\theta} d_1 \end{aligned} \tag{A.2}$$

$$\begin{aligned}\tilde{r}_2' &= (\boldsymbol{x}_1' - \boldsymbol{x}_2') \cdot \boldsymbol{n}_2 + (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{n}_2' \\ &= -v_1 \boldsymbol{t}_1 \cdot \boldsymbol{n}_2 - (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \boldsymbol{t}_2 \\ &= -v_1 \sin\phi_2 - d_2 \end{aligned} \tag{A.3}$$

$$\begin{aligned}\dot{\tilde{r}}_2 &= (\dot{\boldsymbol{x}}_1 - \dot{\boldsymbol{x}}_2) \cdot \boldsymbol{n}_2 + (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot \dot{\boldsymbol{n}}_2 \\ &= (\dot{\theta}\boldsymbol{x}_1' - (\dot{\theta} + \dot{\phi}_2)\boldsymbol{x}_2') \cdot \boldsymbol{n}_2 + (\boldsymbol{x}_1 - \boldsymbol{x}_2) \cdot ((\dot{\theta} + \dot{\phi}_2)\boldsymbol{n}_2') \\ &= -v_1 \dot{\theta} \sin\phi_2 - (\dot{\theta} + \dot{\phi}_2) d_2 \end{aligned} \tag{A.4}$$

From these expressions it follows that the curvature at the contact points can be written as

$$v_1 = -\frac{\tilde{r}_2' + \tilde{d}_2}{\sin\phi_2} = -\frac{\dot{\tilde{r}}_2 + (\dot{\theta} + \dot{\phi}_2)\tilde{d}_2}{\dot{\theta}\sin\phi_2} \tag{A.5}$$

$$v_2 = -\frac{\tilde{r}_1' + \tilde{d}_1}{\sin\phi_2} = -\frac{\dot{\tilde{r}}_1 + \dot{\theta}\tilde{d}_1}{(\dot{\theta} + \dot{\phi}_2)\sin\phi_2} \tag{A.6}$$

Derivatives of Center of Mass and Center of Rotation

Recall the following expressions for the center of mass and the center of rotation:

$$\boldsymbol{c}_m(\phi_0, \phi_1, \phi_2) = -\tilde{r}_2\bar{\boldsymbol{t}}_1 / \sin\phi_2 - R_0\boldsymbol{x}_1 \tag{A.7}$$

$$\boldsymbol{c}_r(\phi_0, \phi_1, \phi_2) = -(\tilde{r}_2\bar{\boldsymbol{t}}_1 + \tilde{d}_2\bar{\boldsymbol{n}}_1) / \sin\phi_2 \tag{A.8}$$

We are interested in the partial derivatives of these expressions with respect to ϕ_0, because they tell us something about the stable poses of the object. In the previous section we computed derivatives with respect to curve parameters. The partial derivative of the curve parameter θ with respect to ϕ_0 is equal to -1. This follows from $\theta = \phi_1 - \phi_0 - \pi/2$ (equation 3.1, p. 19). The partial derivatives with respect to ϕ_0 of equations A.7 and A.8 are therefore

$$\frac{\partial \boldsymbol{c}_m}{\partial \phi_0} = \tilde{r}_2'\bar{\boldsymbol{t}}_1 / \sin\phi_2 - \left(\tfrac{\partial}{\partial\phi_0}R_0\right)\boldsymbol{x}_1 + v_1\bar{\boldsymbol{t}}_1 \tag{A.9}$$

$$= -(v_1\sin\phi_2 + d_2)\bar{\boldsymbol{t}}_1 / \sin\phi_2 - \left(\tfrac{\partial}{\partial\phi_0}R_0\right)\boldsymbol{x}_1 + v_1\bar{\boldsymbol{t}}_1 \tag{A.10}$$

$$= -\frac{\tilde{d}_2\bar{\boldsymbol{t}}_1}{\sin\phi_2} - \left(\tfrac{\partial}{\partial\phi_0}\boldsymbol{R}_0\right)\boldsymbol{x}_1, \tag{A.11}$$

$$\frac{\partial \boldsymbol{c}_r}{\partial \phi_0} = (\tilde{r}_2'\bar{\boldsymbol{t}}_1 + \tilde{d}_2'\bar{\boldsymbol{n}}_1) / \sin\phi_2 \tag{A.12}$$

$$= (-(v_1\sin\phi_2 + d_2)\bar{\boldsymbol{t}}_1 + (v_1\cos\phi_2 + v_2 + r_2)\bar{\boldsymbol{n}}_1) / \sin\phi_2 \tag{A.13}$$

$$= \left(-v_1\bar{\boldsymbol{n}}_2 + v_2\bar{\boldsymbol{n}}_1 + \tilde{r}_2\bar{\boldsymbol{n}}_1 - \tilde{d}_2\bar{\boldsymbol{t}}_1\right) / \sin\phi_2. \tag{A.14}$$

The derivative of $\tilde{d}_2$ can be obtained in a similar fashion as the derivatives of $\tilde{r}_1$ and $\tilde{r}_2$. Notice that equation A.11 is very similar to $\boldsymbol{c}_m - \boldsymbol{c}_r$:

$$\boldsymbol{c}_m - \boldsymbol{c}_r = \tilde{d}_2\bar{\boldsymbol{n}}_1 / \sin\phi_2 - R_0\boldsymbol{x}_1 \tag{A.15}$$

In fact, upon careful inspection we see that

$$\tfrac{\partial}{\partial\phi_0}R_0 = \begin{pmatrix} -\sin\phi_0 & -\cos\phi_0 \\ \cos\phi_0 & -\sin\phi_0 \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} R_0 \tag{A.16}$$

$$\bar{\boldsymbol{n}}_1 = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \bar{\boldsymbol{t}}_1 \tag{A.17}$$

So we can write the partial derivative of the center of mass as

$$\frac{\partial \boldsymbol{c}_m}{\partial\phi_0} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} (\boldsymbol{c}_m - \boldsymbol{c}_r)\,. \tag{A.18}$$

Rotational Velocity

In section 3.3 it was shown that the force/torque balance constraint can be written as

$$r_1 \sin\phi_1 - d_1 \cos\phi_1 = -\tilde{d}_2 \tfrac{\sin\phi_1}{\sin\phi_2}. \tag{A.19}$$

Differentiating the left-hand side of this equation we get:

$$\begin{aligned}
&\tfrac{d}{dt}(r_1 \sin\phi_1 - d_1 \cos\phi_1) = \\
&\quad = (\dot{r}_1 + d_1\dot{\phi}_1)\sin\phi_1 + (r_1\dot{\phi}_1 - \dot{d}_1)\cos\phi_1 && \text{(A.20)}\\
&\quad = d_1(\dot{\phi}_1 - \dot{\theta})\sin\phi_1 + r_1(\dot{\phi}_1 - \dot{\theta})\cos\phi_1 - \tfrac{\dot{\tilde{r}}_2 + (\dot{\theta}+\dot{\phi}_2)\tilde{d}_2}{\sin\phi_2}\cos\phi_1 && \text{(A.21)}\\
&\quad = d_1\dot{\phi}_0 \sin\phi_1 + r_1\dot{\phi}_0\cos\phi_1 - \tfrac{\dot{\tilde{r}}_2 + (\dot{\phi}_1+\dot{\phi}_2-\dot{\phi}_0)\tilde{d}_2}{\sin\phi_2}\cos\phi_1 && \text{(A.22)}\\
&\quad = \dot{\phi}_0\big(d_1\sin\phi_1 + r_1\cos\phi_1 + \tilde{d}_2\tfrac{\cos\phi_1}{\sin\phi_2}\big) - \tfrac{\dot{\tilde{r}}_2 + (\dot{\phi}_1+\dot{\phi}_2)\tilde{d}_2}{\sin\phi_2}\cos\phi_1 && \text{(A.23)}
\end{aligned}$$

The step in equation A.21 follows from properties of the contact support function: $r'(\theta) = -d(\theta)$ and $d'(\theta) = r(\theta) - v(\theta)$. The derivative of the right-hand side of equation A.19 can be written as

$$\tfrac{d}{dt}(-\tilde{d}_2\tfrac{\sin\phi_1}{\sin\phi_2}) = \big(-\dot{\tilde{d}}_2\sin\phi_1 - \tilde{d}_2\dot{\phi}_1\cos\phi_1 + \tilde{d}_2\dot{\phi}_2\sin\phi_1\cot\phi_2\big)/\sin\phi_2 \tag{A.24}$$

Equating expressions A.23 and A.24, substituting expression 3.18 for d_1, and solving for $\dot{\phi}_0$ we arrive at the following expression for $\dot{\phi}_0$:

$$\dot{\phi}_0 = \frac{\dot{\tilde{r}}_2\cos\phi_1 - \dot{\tilde{d}}_2\sin\phi_1 + \tilde{d}_2\dot{\phi}_2\tfrac{\sin\phi_{12}}{\sin\phi_2}}{r_1\sin\phi_{12} + (r_2+\tilde{r}_2)\sin\phi_1 + \tilde{d}_2\cos\phi_1}, \tag{A.25}$$

where $\phi_{12} = \phi_1 + \phi_2$.

A.2 Observability of the Planar Dynamic Case

In the general planar dynamic case the differentials $dL_{g_1}s_1$, $dL_{g_2}s_2$, $dL_{g_1}L_{g_1}s_1$, and $dL_{g_2}L_{g_2}s_2$ and the differentials of the components of the output function will span the observability codistribution $d\mathcal{O}$. To compute these differentials we used Maple V.7 (http://www.maplesoft.com), a symbolic mathematics program. To verify if these differentials span the observability codistribution, we consider the part of the state space that is not part of the output: r_1, r_2, ω_0, and ρ. In other words, the first three components and the last component of the state vector. Below we define a matrix A that has the differentials $dL_{g_1}s_1$, $dL_{g_2}s_2$, $dL_{g_1}L_{g_1}s_1$, and $dL_{g_2}L_{g_2}s_2$ as columns. From those columns we only need the first three components and the last component. If this matrix has full rank, or, equivalently, if the determinant of A is nonzero, then the system is observable. The symbolic expression for this determinant can be computed, but is very complicated. Below we substitute values for all the variables to check that the determinant does not evaluate to 0.

```
># include linear algebra functions
 with(linalg):
```

```
>n:=10: # define state vector and control vector fields
 q:=vector([r1,r2,omega0,s1,s2,phi1,omega1,phi2,omega2,rho]):
 g1:=[0,0,-d1/(m*rho^2*s1),(rho^2+d1^2)/(m*rho^2*s1*(omega1-omega0)),
      (rho^2*cos(phi2)-d1*d2)/(m*rho^2*s1*(omega1+omega2-omega0)),
      0,0,0,0,0]:
 g2:=[0,0,d2/(m*rho^2*s2),(rho^2*cos(phi2)-d1*d2)/(m*rho^2*s2*(omega1-
      omega0)),(rho^2+d2^2)/(m*rho^2*s2*(omega1+omega2-omega0)),
      0,0,0,0,0]:
```

```
># express d1 and d2 as functions of state variables
 d1:=(r1*cos(phi2)+r2)/sin(phi2)-s1:
 d2:=-(r2*cos(phi2)+r1)/sin(phi2)+s2:
```

```
># define what a differential is
 differential:= proc(vec,expr) map(proc(x) simplify(diff(expr,x)) end, vec): end:
```

```
># define what a Lie derivative is
 lie_derivative:= proc(vec,expr) innerprod(differential(q,expr),vec); end:
```

```
># Lvec returns an array of repeated Lie derivatives, an array of differentials of those
  Lie derivatives, and a matrix with those differentials as columns
  Lvec := proc(expr,vec,m)
          global q, f, n:
          local LF, dLF, LFm, i:
          LF:=array(1..n): dLF:=array(1..n):
          LF[1]:=expr:
          dLF[1]:=differential(q,LF[1]):
          for i from 2 to m do
            LF[i] := innerprod(vec,evalm(dLF[i-1])):
            dLF[i] := differential(q,LF[i]):
          od;
          LFm := array(map(proc(x) convert(x,'list') end, convert(dLF,'list'))):
          LF,dLF,LFm:
          end:
```

```
># compute repeated Lie derivatives and differentials
  (lg1,dlg1,lg1m):=Lvec(s1,g1,3): (lg2,dlg2,lg2m):=Lvec(s2,g2,3):
```

```
># form a matrix for that part of the state space that the differentials need to cover
  A:=array([[dlg1[2][1],dlg1[2][2],dlg1[2][3],dlg1[2][10]],
          [dlg1[3][1],dlg1[3][2],dlg1[3][3],dlg1[3][10]],
          [dlg2[2][1],dlg2[2][2],dlg2[2][3],dlg2[2][10]],
          [dlg2[3][1],dlg2[3][2],dlg2[3][3],dlg2[3][10]]]):
```

```
># compute the determinant of that matrix
  Adet:=det(A):
```

```
># 'simplify' this determinant
  simplify(Adet);
```

$$-18(-12r1^3\cos(\phi 2)^5\rho^4 r2\omega 1 - 4s1\sin(\phi 2)r1^2\cos(\phi 2)^6\rho^2 r2^3\omega 0 - 42r1^2\cos(\phi 2)^6 s1^2 r2^4\omega 2 - 2\rho^4\cos(\phi 2)^6 r2^4\omega 0 + 12r1^3\cos(\phi 2)^5\rho^4 r2\omega 0 + 12r1\cos(\phi 2)^3 r2^3\rho^4\omega 0 + 18s1^4\cos(\phi 2)^4 r1^2 s2^2\omega 2 - 6r2^4 s1^2\rho^2\cos(\phi 2)^2\omega 0 + 12s1\sin(\phi 2)r2\rho^4 r1^2\cos(\phi 2)^2\omega 0 - 12s1\sin(\phi 2)r2^2\rho^4\cos(\phi 2)^5 r1\omega 1 + \ldots$$

(several pages of output omitted)

```
># substitute random numbers for the variables and evaluate the determinant
  evalf(subs({m=1,rho=.5,phi2=2,s1=1,s2=2,r1=1.5,r2=1.5,omega0=1,omega1=2,
  omega2=3},Adet));
  25.67009841
```

```
># singularity 1: contact point 1 not moving, omega1-omega0=0
  evalf(subs({m=1,rho=.5,phi2=2,s1=1,s2=2,r1=1.5,r2=1.5,omega0=1,omega1=1,
  omega2=3},Adet));
  Error, numeric exception: division by zero
```

```
># singularity 2: contact point 2 not moving, omega1+omega2-omega0=0
evalf(subs({m=1,rho=.5,phi2=2,s1=1,s2=2,r1=1.5,r2=1.5,omega0=2,omega1=1,
omega2=1},Adet));
Error, numeric exception: division by zero
```

```
># palms move at same rate, omega2=0
evalf(subs({m=1,rho=.5,phi2=2,s1=1,s2=2,r1=1.5,r2=1.5,omega0=1,omega1=2,
omega2=0},Adet));
1647.940473
```

```
># palm 1 is not moving, omega1=0
evalf(subs({m=1,rho=.5,phi2=2,s1=1,s2=2,r1=1.5,r2=1.5,omega0=1,omega1=0,
omega2=2},Adet));
1634.446156
```

A.3 Force/Torque Balance in Three Dimensions

In two dimensions, force/torque balance of an object resting on two palms can be interpreted in the following geometric way: the object is in force/torque balance if and only if the lines through the normals at the contact points intersect the vertical line through the center of mass at one common point. In other words, we can find the X-coordinate of the center of mass if we know the contact points and the contact normals. For a three-dimensional object resting on three palms one can derive a similar constraint. The derivation of the geometric interpretation in two dimensions is analogous to the three-dimensional case.

Let $\bar{n}_i$ be the normal at contact point $c_i, i = 1, 2, 3$ and let f_i be the magnitude of the normal force at contact point c_i. Finally, let c_m be the position of the center of mass. Without loss of generality we assume that the magnitude of the gravitational force is 1. We can then write the force/torque balance constraint as:

$$\text{Force balance:} \qquad \sum f_i \bar{n}_i = \bar{n}_z \tag{A.26}$$

$$\text{Torque balance:} \qquad \sum f_i (c_i - c_m) \times \bar{n}_i = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{A.27}$$

Here, $\bar{n}_z = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ is the normal parallel to the Z-axis. We also require that all f_i's are greater than or equal to 0. The solution for equation A.26 is

$$f_1 = \frac{(\bar{n}_2 \times \bar{n}_3) \cdot \bar{n}_z}{(\bar{n}_1 \times \bar{n}_2) \cdot \bar{n}_3}, \qquad f_2 = \frac{(\bar{n}_3 \times \bar{n}_1) \cdot \bar{n}_z}{(\bar{n}_1 \times \bar{n}_2) \cdot \bar{n}_3}, \qquad f_3 = \frac{(\bar{n}_1 \times \bar{n}_2) \cdot \bar{n}_z}{(\bar{n}_1 \times \bar{n}_2) \cdot \bar{n}_3}$$

We can write the left-hand side of equation A.27 as

$$\begin{aligned}\sum f_i(\boldsymbol{c}_i - \boldsymbol{c}_m) \times \bar{\boldsymbol{n}}_i &= \sum f_i \boldsymbol{c}_i \times \bar{\boldsymbol{n}}_i - \sum f_i \boldsymbol{c}_m \times \bar{\boldsymbol{n}}_i \\ &= \sum f_i \boldsymbol{c}_i \times \bar{\boldsymbol{n}}_i - \boldsymbol{c}_m \times (\sum f_i \bar{\boldsymbol{n}}_i) \\ &= \sum f_i \boldsymbol{c}_i \times \bar{\boldsymbol{n}}_i - \boldsymbol{c}_m \times \bar{\boldsymbol{n}}_z \\ &= \sum f_i \boldsymbol{c}_i \times \bar{\boldsymbol{n}}_i - \begin{pmatrix} c_{m2} \\ -c_{m1} \\ 0 \end{pmatrix},\end{aligned}$$

where c_{m1} and c_{m2} are the X- and Y-coordinate of $\boldsymbol{c}_m$. In other words, the X- and Y-coordinate of $\boldsymbol{c}_m$ can be written as

$$\begin{aligned}c_{m1} &= -\left(\sum f_i \boldsymbol{c}_i \times \bar{\boldsymbol{n}}_i\right) \cdot \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \\ c_{m2} &= \left(\sum f_i \boldsymbol{c}_i \times \bar{\boldsymbol{n}}_i\right) \cdot \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix},\end{aligned}$$

where we substitute the solutions above for f_i. So in three dimensions we can compute the X- and Y-coordinate of the center of mass, if we know the positions of the contact points and the contact normals.

References

Abell, T. and Erdmann, M. A. (1995). Stably supported rotations of a planar polygon with two frictionless contacts. In *Proceedings of the 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 411–418, Pittsburgh, Pennsylvania.

Akella, S., Huang, W. H., Lynch, K. M., and Mason, M. T. (2000). Parts feeding on a conveyor with a one joint robot. *Algorithmica*, 26:313–344.

Akella, S. and Mason, M. T. (1999). Using partial sensor information to orient parts. *International Journal of Robotics Research*, 18(10):963–997.

Allen, P. K. (1988). Integrating vision and touch for object recognition tasks. *International Journal of Robotics Research*, 7(6):15–33.

Allen, P. K. and Michelman, P. (1990). Acquisition and interpretation of 3-D sensor data from touch. *IEEE Transactions on Robotics and Automation*, 6(4):397–404.

Allen, P. K. and Roberts, K. S. (1989). Haptic object recognition using a multi-fingered dextrous hand. In *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, pages 342–347, Scottsdale, AZ.

Ando, S. and Shinoda, H. (1995). Ultrasonic emission tactile sensing. *IEEE Control Systems*, 15(1):61–69.

Arkin, E., Held, M., Mitchell, J., and Skiena, S. (1998). Recognizing polygonal parts from width measurements. *Computational Geometry: Theory and Applications*, 9(4):237–246.

Arkin, E. M., Chew, L. P., Huttenlocher, D. P., Kedem, K., and Mitchell, J. S. B. (1991). An efficiently computable metric for comparing polygonal shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):209–216.

Berretty, R.-P. (2000). *Geometric Design of Part Feeders*. PhD thesis, Utrecht University.

Berretty, R.-P., Goldberg, K., Overmars, M. H., and Van der Stappen, A. F. (1998). Computing fence designs for orienting parts. *Computational Geometry: Theory and Applications*, 10:249–262.

Besançon, G. and Hammouri, H. (1995). Reduced order observer for a class of non-uniformly observable systems. In *Proceedings of the 34th Conference on Decision & Control*, pages 121–125, New Orleans, LA.

Bestle, D. and Zeitz, M. (1983). Canonical form observer design for nonlinear time-variable systems. *International Journal of Control*, 38(2):419–431.

Bicchi, A., Marigo, A., and Prattichizzo, D. (1999). Dexterity through rolling: Manipulation of unknown objects. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pages 1583–1588, Detroit, Michigan.

Böhringer, K. F., Bhatt, V., Donald, B. R., and Goldberg, K. Y. (2000a). Algorithms for sensorless manipulation using a vibrating surface. *Algorithmica*, 26(3/4):389–429.

Böhringer, K.-F., Donald, B. R., Kavraki, L. E., and Lamiraux, F. (2000b). Part orientation with one or two stable equilibria using programmable force fields. *IEEE Transactions on Robotics and Automation*, 16(2):157–170.

Böhringer, K. F., Donald, B. R., and MacDonald, N. C. (1996). Upper and lower bounds for programmable vector fields with applications to MEMS and vibratory plate parts feeders. In Overmars, M. and Laumond, J.-P., editors, *Workshop on the Algorithmic Foundations of Robotics*, pages 255–276. A. K. Peters.

Böhringer, K. F., Donald, B. R., and MacDonald, N. C. (1999). Programmable vector fields for distributed manipulation, with applications to MEMS actuator arrays and vibratory parts feeders. *International Journal of Robotics Research*, 18(2):168–200.

Boissonnat, J. D. and Yvinec, M. (1992). Probing a scene of nonconvex polyhedra. *Algorithmica*, 8:321–342.

Bornard, G. and Hammouri, H. (1991). A high gain observer for a class of uniformly observable systems. In *Proceedings of the 30th Conference on Decision & Control*, pages 1494–1496, Brighton, England.

Cai, C. S. and Roth, B. (1986). On the planar motion of rigid bodies with point contact. *Mechanism and Machine Theory*, 21(6):453–466.

Cai, C. S. and Roth, B. (1987). On the spatial motion of a rigid body with point contact. In *Proceedings of the 1987 IEEE International Conference on Robotics and Automation*, pages 686–695.

Charlebois, M., Gupta, K., and Payandeh, S. (1996). Curvature based shape estimation using tactile sensing. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 3502–3507.

Charlebois, M., Gupta, K., and Payandeh, S. (1997). Shape description of general, curved surfaces using tactile sensing and surface normal information. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 2819–2824.

Chen, N., Rink, R., and Zhang, H. (1996). Local object shape from tactile sensing. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 3496–3501.

Chen, Y.-B. and Ierardi, D. (1995). The complexity of oblivious plans for orienting and distinguishing polygonal parts. *Algorithmica*, 14.

Choi, K. K., Jiang, S. L., and Li, Z. (1998). Multifingered robotic hands: Contact experiments using tactile sensors. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 2268–2273, Leuven, Belgium.

Chou, J. C. (1992). Quaternion kinematic and dynamic differential equations. *IEEE Transactions on Robotics and Automation*, 8(1):53–64.

Coelho Jr., J. A. and Grupen, R. A. (1996). Online grasp synthesis. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, MN.

Cole, R. and Yap, C. K. (1987). Shape from probing. *Journal of Algorithms*, 8(1):19–38.

Cutkosky, M. R. (1985). *Robotic Grasping and Fine Manipulation*. Kluwer, Dordrecht; Boston.

Cutkosky, M. R. and Hyde, J. M. (1993). Manipulation control with dynamic tactile sensing. In *Proc. Sixth International Symposium on Robotics Research*, Hidden Valley, Pennsylvania.

Dobkin, D., Edelsbrunner, H., and Yap, C. K. (1986). Probing convex polytopes. In *Proceedings of the Eighteenth Annual ACM Symposium on Theory of Computing*, pages 424–432, Berkeley, California.

Edelsbrunner, H. and Mücke, E. P. (1994). Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)*, 13(1):43–72.

Ellis, R. E. (1992). Planning tactile recognition in two and three dimensions. *International Journal of Robotics Research*, 11(2):87–111.

Eppstein, D. (1990). Reset sequences for monotonic automata. *SIAM J. Computing*, 19(3):500–510.

Erdmann, M. A. (1995). Understanding action and sensing by designing action-based sensors. *International Journal of Robotics Research*, 14(5):483–509.

Erdmann, M. A. (1998). An exploration of nonprehensile two-palm manipulation: Planning and execution. *International Journal of Robotics Research*, 17(5).

Erdmann, M. A. (1999). Shape recovery from passive locally dense tactile data. In Agarwal, P. K., Kavraki, L. E., and Mason, M. T., editors, *Robotics: The Algorithmic Perspective*. A.K. Peters.

Erdmann, M. A. and Mason, M. T. (1988). An exploration of sensorless manipulation. *IEEE Journal of Robotics and Automation*, 4(4):369–379.

Erdmann, M. A., Mason, M. T., and Vaněček, Jr., G. (1993). Mechanical parts orienting: The case of a polyhedron on a table. *Algorithmica*, 10:226–247.

Fearing, R. S. (1984). Simplified grasping and manipulation with dextrous robot hands. In *Proceedings of the 1984 American Control Conference*, pages 32–38, San Diego, California.

Fearing, R. S. (1990). Tactile sensing for shape interpretation. In Venkataraman, S. T. and Iberall, T., editors, *Dexterous Robot Hands*, chapter 10, pages 209–238. Springer Verlag, Berlin; Heidelberg; New York.

Fearing, R. S. and Binford, T. O. (1988). Using a cylindrical tactile sensor for determining curvature. In *Proceedings of the 1988 IEEE International Conference on Robotics and Automation*, pages 765–771.

Gaston, P. C. and Lozano-Pérez, T. (1984). Tactile recognition and localization using object models: The case of polyhedra on a plane. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(3):257–266.

Gauthier, J. P., Hammouri, H., and Othman, S. (1992). A simple observer for nonlinear systems applications to bioreactors. *IEEE Transactions on Automatic Control*, 37(6):875–880.

Goldberg, K. Y. (1993). Orienting polygonal parts without sensors. *Algorithmica*, 10(3):201–225.

Goldberg, K. Y. and Bajcsy, R. (1984). Active touch and robot perception. *Cognition and Brain Theory*, 7(2):199–214.

Golub, G. H. and Loan, C. F. V. (1996). *Matrix Computations*. Johns Hopkins University Press, third edition.

Grimson, W. E. L. and Lozano-Pérez, T. (1984). Model-based recognition and localization from sparse range or tactile data. *International Journal of Robotics Research*, 3(3):3–35.

Grupen, R. A. and Coelho Jr., J. A. (1993). Sensor-based contact geometry optimization for multifingered robot hands. In *Proc. of Intl. Workshop on Mechatronical Computer Systems for Perception and Action*, pages 147–156, Halmstad, Sweden.

Hastie, T. and Stuetzle, W. (1989). Principal curves. *Journal of the American Statistical Association*, 84(406):502–516.

Hermann, R. and Krener, A. J. (1977). Nonlinear controllability and observability. *IEEE Transactions on Automatic Control*, AC-22(5):728–740.

Hitakawa, H. (1988). Advanced parts orientation system has wide application. *Assembly Automation*, 8(3):147–150.

Hong, J., Lafferriere, G., Mishra, B., and Tan, X. (1990). Fine manipulation with multifinger hands. In *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, pages 1568–1573, Cincinnati, Ohio.

Howe, R. D. and Cutkosky, M. R. (1992). Touch sensing for robotic manipulation and recognition. In Khatib, O., Craig, J. J., and Lozano-Pérez, T., editors, *The Robotics Review 2*. MIT Press, Cambridge, MA.

Hristu, D., Ferrier, N., and Brockett, R. W. (2000). The performance of a deformable-membrane tactile sensor: Basic results on geometrically defined tasks. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pages 508–513, San Francisco, California.

Huttenlocher, D. P. and Kedem, K. (1990). Distance metrics for comparing shapes in the plane. In Donald, B., Kapur, D., and Mundy, J., editors, *AFOSR-NSF Saratoga Conference on the Integration of Numerical and Symbolic Methods*, chapter 8, pages 201–219. Saratoga, New York.

Isidori, A. (1995). *Nonlinear Control Systems*. Springer Verlag, Berlin; Heidelberg; New York, third edition.

Jia, Y.-B. (2000). Grasping curved objects through rolling. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, pages 377–382, San Francisco, California.

Jia, Y.-B. and Erdmann, M. A. (1996). Geometric sensing of known planar shapes. *International Journal of Robotics Research*, 15(4):365–392.

Jia, Y.-B. and Erdmann, M. A. (1999). Pose and motion from contact. *International Journal of Robotics Research*, 18(5).

Jockusch, J., Walter, J., and Ritter, H. (1997). A tactile sensor system for a three-fingered robot manipulator. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 3080–3086, Albuquerque, New Mexico.

Kaneko, M. and Tsuji, T. (2001). Pulling motion based tactile sensing. In Donald, B. R., Lynch, K. M., and Rus, D., editors, *Algorithmic and Computational Robotics: New Directions*, pages 157–167. A. K. Peters.

Kang, D. and Goldberg, K. (1995). Sorting parts by random grasping. *IEEE Transactions on Robotics and Automation*, 11(1):146–152.

Kao, I. and Cutkosky, M. R. (1992). Quasistatic manipulation with compliance and sliding. *International Journal of Robotics Research*, 11(1):20–40.

Kavraki, L. E. (1997). Part orientation with programmable vector fields: Two stable equilibria for most parts. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 2446–2451, Albuquerque, New Mexico.

Keren, D., Rivlin, E., Shimsoni, I., and Weiss, I. (1998). Recognizing surfaces using curve invariants and differential properties of curves and surfaces. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 3375–3381, Leuven, Belgium.

Kerr, J. and Roth, B. (1986). Analysis of multifingered hands. *International Journal of Robotics Research*, 4(4):3–17.

Kinoshita, G., Sugeno, Y., Oosumi, H., Umeda, K., and Muranaka, Y. (1999). High compliance sensing behaviour of a tactile sensor. In *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 826–831.

Klatzky, R. L., Lederman, S. J., and Metzger, V. A. (1985). Identifying objects by touch: An "expert system". *Perception and Psychophysics*, 37:299–302.

Kölzow, D., Kuba, A., and Volčič, A. (1989). An algorithm for reconstructing convex bodies from their projections. *Discrete and Computation Geometry*, 4:205–237.

Kou, S. R., Elliott, D. L., and Tarn, T. J. (1975). Exponential observers for nonlinear dynamic systems. *Information and Control*, 29:393–428.

Krener, A. J. and Isidori, A. (1983). Linearization by output injection and nonlinear observers. *Systems Control Letters*, 3:47–52.

LeBlanc, M. and Tibshirani, R. (1994). Adaptive principal surfaces. *Journal of the American Statistical Association*, 89(425):53–64.

Lederman, S. J. and Browse, R. A. (1988). The physiology and psychophysics of touch. In Dario, P., editor, *Sensors and Sensory Systems for Advanced Robots*, pages 71–91, Berlin; Heidelberg; New York. Springer Verlag.

Lee, M. H. (2000). Tactile sensing: New directions, new challenges. *International Journal of Robotics Research*, 19(7):636–643.

Li, S.-Y. R. (1988). Reconstruction of polygons from projections. *Information Processing Letters*, 28:235–240.

Lindenbaum, M. and Bruckstein, A. (1991). Parallel strategies for geometric probing. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 392–397.

Lindenbaum, M. and Bruckstein, A. M. (1994). Blind approximation of planar convex sets. *IEEE Transactions on Robotics and Automation*, 10(4):517–529.

Liu, F. and Hasegawa, T. (2001). Reconstruction of curved surfaces using active tactile sensing and surface normal information. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*, pages 4029–4034, Seoul, Korea.

Liu, H., Meusel, P., and Hirzinger, G. (1995). A tactile sensing system for the DLR three-finger robot hand. In *Fifth International Symposium on Measurement and Control in Robotics (ISMCR)*, pages 91–96.

Lozano-Pérez, T., Mason, M. T., and Taylor, R. H. (1984). Automatic synthesis of fine-motion strategies for robots. *International Journal of Robotics Research*, 3(1):3–24.

Lynch, K. M. (1997). *Nonprehensile Robotic Manipulation: Controllability and Planning*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Lynch, K. M., Shiroma, N., Arai, H., and Tanie, K. (1998). The roles of shape and motion in dynamic manipulation: The butterfly example. In *Proceedings of the 1998 IEEE International Conference on Robotics and Automation*, pages 1958–1963.

Maekawa, H., Tanie, K., and Komoriya, K. (1993). A finger-shaped tactile sensor using an optical waveguide. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, volume 5, pages 403–408.

Maekawa, H., Tanie, K., and Komoriya, K. (1997). Tactile feedback for multifingered dynamic grasping. *IEEE Control Systems Magazine*, 17(1):63–71.

Marigo, A., Chitour, Y., and Bicchi, A. (1997). Manipulation of polyhedral parts by rolling. In *Proceedings of the 1997 IEEE International Conference on Robotics and Automation*, pages 2992–2997.

Markenscoff, X., Ni, L., and Papadimitriou, C. (1990). The geometry of grasping. *International Journal of Robotics Research*, 9(1):61–74.

Mason, M. T. (1982). *Manipulator Grasping and Pushing Operations*. PhD thesis, AI-TR-690, Artificial Intelligence Laboratory, MIT.

Mason, M. T. (1985). The mechanics of manipulation. In *Proceedings of the 1985 IEEE International Conference on Robotics and Automation*, pages 544–548, St. Louis.

Mingrino, A., Bucci, A., Magni, R., and Dario, P. (1994). Slippage control in hand prostheses by sensing grasping forces and sliding motion. In *Proceedings of the 1994 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1803–1809.

Mishra, B., Schwartz, J. T., and Sharir, M. (1987). On the existence and synthesis of multifinger positive grips. *Algorithmica*, 2(4):541–558.

Moll, M. and Erdmann, M. A. (2002). Manipulation of pose distributions. *International Journal of Robotics Research*. To appear.

Moll, M., Goldberg, K., Erdmann, M. A., and Fearing, R. (2002). Aligning parts for micro assemblies. *Assembly Automation*, 22(1):46–54.

Montana, D. J. (1988). The kinematics of contact and grasp. *International Journal of Robotics Research*, 7(3):17–32.

Montana, D. J. (1995). The kinematics of multi-fingered manipulation. *IEEE Transactions on Robotics and Automation*, 11(4):491–503.

Nagata, K., Keino, T., and Omata, T. (1993). Acquisition of an object model by manipulation with a multifingered hand. In *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 1045–1051, Osaka, Japan.

Nguyen, V.-D. (1988). Constructing force-closure grasps. *International Journal of Robotics Research*, 7(3):3–16.

Nicholls, H. R. and Lee, M. H. (1989). A survey of robot tactile sensing. *International Journal of Robotics Research*, 8(3):3–30.

Nijmeijer, H. and Van der Schaft, A. (1990). *Nonlinear Dynamical Control Systems*. Springer Verlag, Berlin; Heidelberg; New York.

Okamura, A. M., Costa, M. A., Turner, M. L., Richard, C., and Cutkosky, M. R. (1999). Haptic surface exploration. In Corke, P. and Trevelyan, J., editors, *Experimental Robotics VI*, Sydney, Australia. Springer Verlag.

Okamura, A. M. and Cutkosky, M. R. (1999). Haptic exploration of fine surface features. In *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, pages 2930–2936, Detroit, Michigan.

Okamura, A. M. and Cutkosky, M. R. (2001). Feature detection for haptic exploration with robotic fingers. *International Journal of Robotics Research*, 20(12):925–938.

Paljug, E., Yun, X., and Kumar, V. (1994). Control of rolling contacts in multi-arm manipulation. *IEEE Transactions on Robotics and Automation*, 10(4):441–452.

Peshkin, M. A. and Sanderson, A. C. (1988). Planning robotic manipulation strategies for sliding objects. *IEEE Journal of Robotics and Automation*, 4(5).

Ponce, J., Sullivan, S., Sudsang, A., Boissonnat, J.-D., and Merlet, J.-P. (1997). On computing four-finger equilibrium and force-closure grasps of polyhedral objects. *International Journal of Robotics Research*, 16(1):11–35.

Preparata, F. P. and Shamos, M. I. (1985). *Computational Geometry — An Introduction*. Springer Verlag, Berlin; Heidelberg; New York.

Rao, A., Kriegman, D., and Goldberg, K. (1994). Complete algorithms for feeding polyhedral parts using pivot grasps. Technical Report UU-CS-1994-49, Utrecht University. ftp://ftp.cs.ruu.nl/pub/RUU/CS/techreps/CS-1994/1994-49.ps.gz.

Rao, A., Kriegman, D., and Goldberg, K. (1995). Complete algorithms for reorienting polyhedral parts using a pivoting gripper. In *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, pages 2242–2248.

Rao, A. S. and Goldberg, K. Y. (1994). Shape from diameter: Recognizing polygonal parts with a parallel-jaw gripper. *International Journal of Robotics Research*, 13(1):16–37.

Reuleaux, F. (1876). *The Kinematics of Machinery*. MacMillan. Reprinted by Dover, 1963.

Reznik, D., Moshkovich, E., and Canny, J. (1999). Building a universal part manipulator. In Böhringer, K. and Choset, H., editors, *Distributed Manipulation*. Kluwer.

Rizzi, A. A. and Koditschek, D. E. (1993). Further progress in robot juggling: The spatial two juggle. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, volume 3, pages 919–924, Atlanta, Georgia.

Roberts, K. S. (1990). Robot active touch exploration: Constraints and strategies. In *Proceedings of the 1990 IEEE International Conference on Robotics and Automation*, pages 980–985.

Romanik, K. (1995). Geometric probing and testing – a survey. Technical Report 95-42, DIMACS Center for Discrete Mathematics and Theoretical Computer Science, Rutgers University.

Russell, R. A. (1992). Using tactile whiskers to measure surface contours. In *Proceedings of the 1992 IEEE International Conference on Robotics and Automation*, volume 2, pages 1295–1299, Nice, France.

Salisbury, J. K. (1982). *Kinematic and Force Analysis of Articulated Hands*. PhD thesis, Department of Mechanical Engineering, Stanford University.

Salisbury, K. (1987). Whole arm manipulation. In *Proc. Fourth International Symposium on Robotics Research*, pages 183–189, Santa Cruz, California.

Santaló, L. A. (1976). *Integral Geometry and Geometric Probability*, volume 1 of *Encyclopedia of Mathematics and its Applications*. Addison-Wesley, Reading, MA.

Schneiter, J. L. and Sheridan, T. B. (1990). An automated tactile sensing strategy for planar object recognition and localization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(8):775–786.

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

Shinoda, H. and Oasa, H. (2000). Wireless tactile sensing element using stress-sensitive resonator. *IEEE/ASME Trans. on Mechatronics*, 5(3):258–265.

Siegel, D. M. (1991). Finding the pose of an object in a hand. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 406–411.

Skiena, S. S. (1989). Problems in geometric probing. *Algorithmica*, 4(4):599–605.

Speeter, T. H. (1990). A tactile sensing system for robotic manipulation. *International Journal of Robotics Research*, 9(6):25–36.

Spivak, M. (1999a). *A Comprehensive Introduction to Differential Geometry*, volume 2. Publish or Perish, Houston, Texas, third edition.

Spivak, M. (1999b). *A Comprehensive Introduction to Differential Geometry*, volume 3. Publish or Perish, Houston, Texas, third edition.

Sudsang, A. and Kavraki, L. (2001). A geometric approach to designing a programmable force field with a unique stable equilibrium for parts in the plane. In *Proceedings of the 2001 IEEE International Conference on Robotics and Automation*.

Sudsang, A., Ponce, J., and Srinivasa, N. (2000). In-hand manipulation: Geometry and algorithms. *Algorithmica*, 26(4):466–493.

Teichmann, M. and Mishra, B. (2000). Reactive robotics I: Reactive grasping with a modified gripper and multi-fingered hands. *International Journal of Robotics Research*, 19(7):697–708.

Teramoto, K. and Watanabe, K. (2000). Principal curvature estimation by acoustical tactile sensing system. In *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1522–1527.

Thau, F. E. (1973). Observing the state of nonlinear dynamic systems. *International Journal of Control*, 17:471–479.

Trinkle, J. C., Abel, J. M., and Paul, R. P. (1988). An investigation of frictionless enveloping grasping in the plane. *International Journal of Robotics Research*, 7(3):33–51.

Trinkle, J. C. and Hunter, J. J. (1991). A framework for planning dexterous manipulation. In *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, pages 1245–1251.

Trinkle, J. C. and Paul, R. P. (1990). Planning for dexterous manipulation with sliding contacts. *International Journal of Robotics Research*, 9(3):24–48.

Trinkle, J. C., Ram, R. C., Farahat, A. O., and Stiller, P. F. (1993). Dexterous manipulation planning and execution of an enveloped slippery workpiece. In *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, pages 442–448.

Van der Stappen, A. F., Goldberg, K., and Overmars, M. H. (2000). Geometric eccentricity and the complexity of manipulation plans. *Algorithmica*, 26(3):494–514.

Whitcomb, L. L., Rizzi, A. A., and Koditschek, D. E. (1993). Comparative experiments with a new adaptive controller for robot arms. *IEEE Transactions on Robotics and Automation*, 9(1):59–70.

Wiegley, J., Goldberg, K., Peshkin, M., and Brokowski, M. (1996). A complete algorithm for designing passive fences to orient parts. In *Proceedings of the 1996 IEEE International Conference on Robotics and Automation*, pages 1133–1139.

Yoshikawa, T., Yokokohji, Y., and Nagayama, A. (1993). Object handling by three-fingered hands using slip motion. In *Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 99–105, Yokohama, Japan.

Zimmer, G. (1994). State observation by on-line minimization. *International Journal of Control*, 60(4):595–606.

Zumel, N. B. (1997). *A Nonprehensile Method for Reliable Parts Orienting*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.