Mathematical Questions in Robotics*

John Baillieul
Roger Brockett
Bruce Donald
Richard Murray
Madhusudan Raghavan
S. Shankar Sastry

TR 90-1097
February 1990

Department of Computer Science
Cornell University
Ithaca, NY 4853-7501

*This technical report represents a working draft of a book to be published is the "Symposia in Applied Mathematics" series of the American Mathematical Society. It represents the lectures given at a short course on January 16-17, 1990, in Louisville, Kentucky, at the Joint Meeting of the AMS and the MAA. The short course was entitled "Mathematical Questions in Robotics."

The book will also be called "Mathematical Questions in Robotics" and is by John Baillieul, Roger Brockett, Bruce Donald, Richard Murray, Madhusudan Raghavan, and S. Shankar Sastry. This work was supported by grants too numerous to list on this page, but inside, the authors acknowledge them individually.

AMERICAN MATHEMATICAL SOCIETY

Short Course Series

Mathematical Questions in Robotics
LECTURE NOTES

January 16–17, 1990
Louisville, Kentucky

CONTENTS:

American Mathematical Society Short Course Series

Introductory Survey Lectures on

Mathematical Questions in Robotics

January 16–17, 1990 *Louisville, Kentucky*

The program for this short course is being coordinated by Roger W. Brockett of Harvard University. It will include lectures by John B. Baillieul (Boston University), Bruce R. Donald (Cornell University), Madhusudan Raghavan (General Motors Research Laboratories), and Shankar Sastry (University of California, Berkeley). The course is one of a series given by the Society on the recommendation of the AMS-MAA Committee on Employment and Educational Policy (CEEP) whose members are Donna L. Beers, Morton Brown, Stefan A. Burr, Edward A. Connors (chair), Philip C. Curtis, Jr., David J. Lutzer, and James J. Tattersall. The short course series is under the direction of the Short Course Subcommittee, whose members are Stefan A. Burr (chair), R. Peter DeLong, Lisl Novak Gaal, Robert P. Kurshan, Barbara L. Osofsky, Marjorie L. Stein, and James J. Tattersall.

Tuesday, January 16, 1990

8:00 a.m.	– 2:30 p.m.	Registration
9:00 a.m.	– 10:15 a.m.	*Introduction* Roger W. Brockett
10:45 a.m.	– 12 noon	*Manipulator Kinematics* Madhusudan Raghavan
2:00 p.m.	– 3:15 p.m.	*Resolution of Kinematic Redundancy* John B. Baillieul
3:45 p.m.	– 5:15 p.m.	*Mathematical Problems in Grasping and Manipulation by Multifingered Robot Hands* Shankar Sastry
5:15 p.m.	– 5:45 p.m.	Discussion

Wednesday, January 17, 1990

2:15 p.m.	– 3:30 p.m.	*Planning and Executing Robot Assembly Strategies in the Presence of Uncertainty* Bruce R. Donald
4:00 p.m.	– 5:15 p.m.	*Symbolic Description of Movement* Roger W. Brockett

Synopses of the talks and lecture notes prepared by the individual speakers follow. The synopses were published in the October 1989 issue of the *Notices of the American Mathematical Society* on pages 1080–1082, and the November 1989 issue, pages 1236–1237 and are reprinted here.

Reprinted from
Notices of the American Mathematical Society
Volume 36, Number 8, October 1989, and Volume 36, Number 9, November, 1989

Synopses and Reading Lists

Introduction (Roger W. Brockett). The introduction will be devoted to establishing a framework for the study of robot kinematics, compliance and task description. We will select a few highlights from the literature and give enough background so as to make it possible to appreciate their mathematical context and their practical significance. Examples will include a physical and mathematical formulation of the idea of a kinematic chain, a discussion of compliance, the treatment of a basic result on grasping and a result on optimal kinematic configurations.

1. Richard Paul, *Robotic Manipulators: Mathematics, Programming and Control*, MIT Press, Cambridge, Mass., 1981.
2. R. W. Brockett, *Robotic Manipulations and the Product of Exponentials Formula*, in Lecture Notes in Control and Information Sciences, Proceedings of the International Symposium on Mathematical Theory of Networks and Systems (P. A. Fuhrman, ed.), Springer-Verlag, Berlin, 1984, pp. 120–127.
3. K. H. Hunt, *Kinematic Geometry of Mechanisms*, Oxford University Press, Oxford, 1978.
4. R. W. Brockett and Josip Loncaric, *The Geometry of Compliance Programming*, in Theory and Application of Nonlinear Control Systems (C. I. Byrnes and A. Lindquist, eds.), North Holland, Amsterdam, 1986, pp. 35–42.

Kinematics of Manipulators (M. Raghavan). The two classical problems in manipulator kinematics are: (a) *the forward kinematics problem*: given the manipulator geometry (i.e. base location, link lengths, twist angles, offsets) and values of the joint variables, find the position and orientation of the hand, (b) *the inverse kinematics problem*: given the manipulator geometry and the desired position and orientation of the hand, find the values of the joint variables which will place the hand in that desired configuration.

For manipulators comprised of rigid bodies, the above problems may be formulated as principal systems of multivariate polynomials over the reals. Algebraic methods for solving such systems usually proceed by eliminating variables and reducing the problem to one involving several univariate polynomials over the reals. Manipulator kinematics therefore relies heavily on elimination algorithms. In this lecture, we describe *Sylvester's dyalitic method*, an elimination procedure which exploits the sparsity and structure evident in polynomials arising in manipulator kinematics.

The equations describing a manipulator exhibit some interesting properties. Let $\bar{\mathbf{p}}$ be the position vector of a point on the hand of a n-revolute jointed spatial series manipulator. When $\bar{\mathbf{p}}$ is expressed as a function of the manipulator parameters, one may show that it is linear in $sin\theta_i$ and $cos\theta_i$, where $\theta_i, i = 1, \ldots, n$ are the joint variables. Interestingly enough, the function $\bar{\mathbf{p}}^T\bar{\mathbf{p}}$ is also linear in $sin\theta_i$ and $cos\theta_i$. We present a list of such functions in the ideal associated with revolute jointed manipulators. We then show how they may be used with the dyalitic elimination procedure to solve the inverse kinematics problem for the six-revolute jointed series manipulator of general geometry. This was once considered the most difficult problem in manipulator kinematics.

1. G. Salmon, *Lessons introductory to the modern higher algebra*, Chelsea Publishing Co., 1885, pages 79–90.
2. J. F. Canny, *The complexity of robot motion planning*, MIT Press, 1988, chapter 3.
3. F. S. Macaulay, *The algebraic theory of modular systems*, Stechert-Hafner Service Agency, 1964, pages 1–17.
4. J. J. Craig, *Introduction to robotics, mechanics and control*, Addison-Wesley Publishing Co., 1986, chapters 1–4.
5. L.-W. Tsai and A. Morgan, *Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods*, Transactions of the ASME, Journal of Mechanisms Transmissions and Automation in Design, Vol. 107, June 1985, pages 189–200.
6. H.-Y. Lee and C.-G. Liang, *A new vector theory for the analysis of spatial mechanisms*, Mechanisms and Machine Theory, Vol. 23, No. 3, 1988, pages 209–217.
7. H.-Y. Lee and C.-G. Liang, *Displacement analysis of the general spatial 7-link 7R mechanism*, ibid, pages 219–226.

Resolution of Kinematic Redundancy (J. Baillieul). Kinematic redundancy can be a design feature or an artifact of a particular mode of operation in robotics. In either case, motion programming of devices such as robotic hands, manipulator arms, etc. presents both challenge and opportunity since the task does not uniquely determine the trajectories of all degrees of freedom. For a variety of practical reasons, it is desirable to resolve kinematic redundancy in a rational fashion, and there is a growing contemporary body of literature on various aspects of such resolution.

A very basic example of the problems which are studied involves a robot arm which has more joints than necessary to move the end effector or hand into an arbitrary configuration (=position and orientation). A simple abstraction of the problem involves an analytic function $f : \Theta \to \mathcal{X}$ associating joint configurations $\theta \in \Theta$ to end effector configurations $x \in \mathcal{X}$. $\mathcal{X}$ will typically be (a subset of) the group of proper rigid motions of 3-space, $SE(3, \mathbf{R})$, and if, say, the robot arm has only revolute joints, Θ is (a subset of) an n-torus. Kinematic redundancy is typically an issue if $\dim\Theta > \dim\mathcal{X}$ in which case we face the problem of the equation $f(\theta) = x$ being underdetermined. One rational approach to associating a joint configuration to a given end effector configuration x is to seek that value θ which optimizes some criterion function $g(\theta)$ subject to satisfying the constraint $f(\theta) = x$. Under reasonable assumptions, the mathematics underlying such pointwise optimization needn't involve more than multivariable calculus. If as is typical in robotic applications, however, there is given a path of configurations $x(t)$, it is not always straightforward to generate a corresponding path $\theta(t)$ by optimizing some function along the path. Generally, one wants the algorithms for generating $\theta(t)$ to be *cyclic* (closed x-paths lift to closed θ-paths), and it has been shown that in many robotic applications there are topological obstructions to globally valid implementations of cyclic algorithms. These obstructions (called *algorithmic singularities* in the robotics literature) manifest themselves physically in nonsmooth motions and the robot's being unable to move in certain directions to first order.

The lecture will consist of a historical introduction to redundancy in robots and a look at the current state of algorithms which resolve kinematic redundancy by instantaneously optimizing a function of the joint configurations. We shall also discuss resolution of kinematic redundancy by means of optimizing nonholonomic and path-integral criteria. The latter approach has practical advantages in certain applications, but it leads to a class of two point boundary value problems which, if solved by standard numerical procedures, are substantially more complex than the previously mentioned resolution techniques based on optimizing instantaneous functions of the joint configurations. Recent research has shown that, using the theory of exponential dichotomies, we may develop continuation methods which render these problems more manageable while at the same time relating their solution to well understood instantaneous methods.

1. J. Baillieul, 1985. *Kinematic Programming Alternatives for Redundant Manipulators*, IEEE International Conference on Robotics and Automation, St. Louis MO, March 25–28, pp. 722–728.
2. D. R. Baker and C. W. Wampler, 1988. *On the Inverse Kinematics of Redundant Manipulators*, International Journal of Robotics Research, vol. 7, no. 2, pp. 3–21.
3. D. P. Martin, J. Baillieul, and J. M. Hollerbach, 1989. *Resolution*, IEEE Transactions on Robotics and Automation, v. 5, no. 4, pp. 529–533.

Symbolic Description of Movement (Roger W. Brockett). It can be very tedious to specify the instructions which will enable a robot to carry out a useful task. In order to make this process as efficient as possible it is desirable to have a precise, yet flexible, language for man-machine interaction. Since movement is central to what a robot does, this means that one needs a convenient language for describing motion in a 3-dimensional world populated by objects. In short, one needs a motion description language. In this talk we will give examples of current practice, drawing on work in various areas outside and within robotics. We will discuss the role of feedback (force and vision) and investigate robustness and stability based on idealized models. We will study in some detail a formal language (in the sense that this term is used in computational theory) for motion description and illustrate its use with a video.

1. H. R. Lewis and C. H. Papadimitriou, *Elements of the Theory of Computation*, Prentice-Hall, Englewood Cliffs, NJ, 1981.
2. R. W. Brockett, *On the Computer Control of Movement*, Proceedings of the 1988 IEEE Conference on Robotics and Automation, April 1988.

Planning and Executing Robot Assembly Strategies in the Presence of Uncertainty (Bruce R. Donald). Research in robot programming attempts to build robot planning systems that can function at the task-level. A task-level specification of a robot plan might have the form, *Put together this disk rotor assembly.* The planner is given geometric models of the parts, and geometric or analytic models of the robot dynamics. Beyond this, the specification, or input to the planner does not mention the specific kinematic and dynamic constraints that the robot must obey; these are determined by the planner using geometrical computation. The goal of a task-level planner is to take a task-level specification and to produce a runnable robot program—one which is fully specified in terms of force-control, kinematics, and dynamics—that can accomplish the task.

Major advances in task-level planning can enable robotics to achieve its full potential in the assembly domain. Today, even existing robots cannot be exploited to their full capacity. For example, assembly tasks require compliant motion; however, compliance requires force-control, and such force-control motion strategies are quite difficult for humans to specify. Furthermore, robot assembly programs are very sensitive to the details of geometry. Finally, reprogramming a general purpose robot for a new assembly task can take time on the order of man-months. For these reasons, we have been working on the automatic synthesis of motion strategies for robots.

Research in task-level planning is often characterized as *theoretical robotics*. There are several reasons for this; the first is that much of the work has been concerned with constructing a *theory* of planning. In other words, the computational problem "task-level planning" is not well-specified. Much of our work lies in specifying the computation precisely. Second, given some sort of decomposition of task-level planning into "planning problems", one is immediately driven to ask, *What are the algorithms for these problems? Can plans, in general, be computed? How efficiently can planning algorithms run?* Historically, the nature of these questions has led researchers to apply tools from theoretical computer science, computational geometry, and algebra.

Recently, a great deal of attention has been focused on a particular robotics problem, called the find-path, or generalized movers' problem. In this problem, we ask the purely kinematic question, can a robot system be moved from one configuration to another, without colliding with obstacles? This is a nicely-defined mathematical problem, and, after much research, at this point its computational complexity is precisely known.

In fact, the neatness of this problem is deceptive, so much so that this formal problem has even been called "the" motion planning problem. From a task-level viewpoint, there is much hidden in the statement "Can the robot system be moved...?" Specifically, the find-path problem assumes that the robot has a perfect control system that can exactly execute the plan, and that the geometric and analytic models of the robot and obstacles are exact.

In reality, of course, robot control systems are subject to significant uncertainty and error. Typical robots are also equipped with sensors—force sensors, kinesthetic position sensors, tactile sensors, vision, and so forth. However, these sensors are also subject to significant uncertainty. Finally, the geometrical models of the robot and the environment (parts, obstacles, etc.) cannot be exact—they are accurate only to manufacturing tolerances, or to the accuracy of the sensors used to acquire the models. Uncertainty is not a mere engineering detail; in particular, it is characteristically impossible to "patch" these perfect plans in such a way that they will function once uncertainty comes into play. Uncertainty is an absolutely fundamental problem in robotics, and plans produced under the assumption of no uncertainty are meaningless. What is needed is a principled theory of planning in the presence of uncertainty. Such a theory must not only be computational, but must also take uncertainty into account *a priori*. The overlap with exact motion planning algorithms can be stated roughly as follows: exact kinematic planning algorithms provide a computational-geometric theory of holonomic constraints. In motion planning with uncertainty, we exploit compliant motion—sliding on surfaces—in order to effect a "structural" reduction in uncertainty. Such compliant motion plans can be synthesized from a computational analysis of the geometry of the holonomic constraints.

We will present a precise framework for motion planning with uncertainty. In particular, given geometric bounds on the uncertainty in sensing and control, we develop algorithms for generating and verifying compliant motion strategies that are guaranteed to succeed as long as the sensing and control uncertainties lie within the specified bounds.

1. B. R. Donald, *Error Detection and Recovery in Robotics*, Springer-Verlag, Lecture Notes in Computer Science, (1989).

Control and Programming of Multifingered Robot Hands (Shankar Sastry). In this talk, I will discuss the dynamics, control, planning of motions and design of multifingered hands. The talk will be in two parts with extensive videotape footage (to be shown after the talk) illustrating both animated simulations and experiments. The talk is a blend of recent advances in our understanding of classical mechanics, computer graphics as well as some neurophysiology.

In the first part I will discuss the modelling of the low-level control problem for a multifingered hand manipulating an object under a variety of contact types: fixed, sliding, rolling and soft fingered or thealogical. The nonholonomies associated with rolling make it particularly interesting. We discuss the use of control laws for explicit linearization of these control systems under state feedback. We show simulations and discuss implementation considerations.

I will discuss in the second part of the talk how results from the (differential) geometric control theory literature can be brought to bear on the problem of changing grasps on an object. Some connections with Berry's phase formula will also be given. Finally, I will discuss a neurophysiologically motivated environment which we are developing for the specification, control and programming of multifingered hands.

The talk is based on collaborative work with my students: Li Zexiang (NYU), Ping Hsu (University of Illinois), Arlene Cole (AT&T Bell Labs), John Hauser (USC), Richard Murray, Curt Deno and Kris Pister.

1. Z. Li and S. S. Sastry, *Task Oriented Optimal Grasping by Multifingered Robot Hands*, IEEE Journal of Robotics and Automation, Vol. 4, No. 1, 1988, pp. 32-44.

2. D. J. Montana, *Tactile Sensing and the Kinematics of Contact*, Ph.D. dissertation, Div. of App. Sciences, Harvard Univ., Cambridge, Mass., 1986.

3. A. B. A. Cole and J. E. Hauser, *Kinematics and Control of Multifingered Hands with Rolling Contact*, IEEE Transactions on Automatic Control, Vol. 34, No. 4, 1989, pp. 398-404.

4. Z. Li, P. Hsu and S. S. Sastry, *Grasping and Coordinated Manipulation by a Multifingered Robot Hand*, International Journal of Robotics Research, Vol. 8, No. 4, August 1989, pp. 33-50.

Some Mathematical Aspects of Robotics

R. W. Brockett

Abstract

In these notes we discuss a few basic problems in robotics which have a clear mathematical meaning and which are partly, but not completely, solved.

1. Introduction

The field of robotics is concerned with the generation of motion which will cause objects in the world to move in some desired way. We may want to generate a motion which will cause a pencil to be sharpened or a motion which will move a silicon wafer from point A to point B. The robot itself is ordinarily just a means to an end but even so one often views the motion of the robot as being the main object of study.

In order to introduce the reader to the study of these matters we indicate how the subject of motion control can be divided into subproblems and then give an example of work in each area. Among the principal subareas we have:

i) The description of kinematic chains.
ii) The description of compliance.
iii) Motion planning and obstacle avoidance.
iv) The problem of grasping.

In this introduction I will discuss some vocabulary and a few problems relating to kinematic chains and compliance control.

2. The Problem of Inverse Kinematics

As is discussed in detail in the notes of M. Raghavan, the systematic description of robotic kinematics leads to the study of kinematic chains. To a first approximation, robots are used to move rigid objects in ordinary three dimensional euclidean space. For this reason it is not at all surprising that the six parameter family of rigid motion in three

space will continue to appear, and in particular that kinematic chains and rigid motion are closely related. In fact if we describe a particular rigid motion by the statement that it takes a point with coordinate vector x relative to some fixed coordinate system into a point with coordinate vector $Ax+b$ relative to the same system, then the effect of composing two such transformations is to send x into $A_2(A_1x+b_1)+b_2 = A_2A_1x+A_2b_1+b_2$. In view of the formula

$$\begin{bmatrix} A_2 & b_2 \\ 0 & 1 \end{bmatrix}\begin{bmatrix} A_1 & b_1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} A_2A_1 & A_2b_1+b_2 \\ 0 & 1 \end{bmatrix}$$

one sees that composition of rigid transformation is captured by a matrix multiplication in the order indicated.

We use the notation $\mathbf{E}^3$ to denote ordinary cartesian 3-space with the standard inner product $(x,y) = \Sigma x_iy_i$ and refer to it as euclidean 3 space. The motion of a rigid body can be described by a curve in the group of rigid motions in three dimensional euclidean space. This means that in a mathematical discussion of manipulation the Lie Group $\mathbf{E}(3)$ consisting of all euclidean motions, is bound to pay a role. An affine transformation

$$y = Ax + b$$

is said to define a euclidean transformation if A is an orthogonal matrix. Thought of as a group under composition, the rigid motions and the set of matrices of the form

$$M = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} \quad ; \quad A = \text{orthogonal}$$

thought of as a group under multiplication, are isomorphic. We refer to the above representation as the standard matrix representation.

Of course, the set of 3 by 3 orthogonal matrices is a three dimensional group and so $\mathbf{E}(3)$ is 6-dimensional. Its composition law is continuous in the obvious sense, and so $\mathbf{E}(3)$ can be regarded as a Lie group. Its Lie algebra has the representation

$$\mathcal{L} = \left\{ L \mid L = \begin{bmatrix} S & v \\ 0 & 0 \end{bmatrix} \quad ; \quad S = -S^T \right\}$$

The Lie group $\mathbf{E}(3)$ is neither simple nor solvable, but is the semidirect product of the simple Lie group of orthogonal transformations and the abelian Lie group consisting of all translations.

There is a natural Riemannian structure on every semisimple compact Lie group. This comes from the Killing form. In the case of the orthogonal group in three space we can express the matter in terms of the Euler angles. For the parametrization

$$\Theta = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & 0 & \sin\phi \\ 0 & 1 & 0 \\ -\sin\phi & 0 & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

the Riemannian metric is

$$(ds)^2 = \frac{1}{4}(d\theta)^2 + \frac{1}{4}(d\phi)^2 + \frac{1}{2}\sin\phi(d\psi)(d\theta) + \frac{1}{4}(d\psi)^2$$

There is no canonical choice of Riemannian structure on $\mathbf{E}(3)$, however, and not even a natural volume measure on $\mathbf{E}(3)$. This comes about because rotation and translation are not related by any natural scale. On the other hand, if we select a scale, which amounts to choosing a characteristic length, then we get a Riemannian metric on $\mathbf{E}(3)$. Thus we may say that there is a natural one parameter family of metrics on $\mathbf{E}(3)$, with the parameter being a length scale.

Given any matrix of the form

$$M = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} \quad ; \quad A^T A = I$$

there exists

$$N = \begin{bmatrix} S & x \\ 0 & 0 \end{bmatrix} \quad ; \quad S = -S^T$$

such that $\exp N = M$. That is, the exponential map is onto $\mathbf{E}(3)$. Since we can obviously write

$$\begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I & b \\ 0 & 1 \end{bmatrix} \exp \begin{bmatrix} S & 0 \\ 0 & 0 \end{bmatrix}$$

we obtain Euler's theorem stating that any rigid motion can be thought of as a translation followed by a rotation about a line passing through a preassigned fixed point. Chasles

showed that any rigid displacement can be achieved by rotation and translation which commute. A rotation about the origin

$$R = \begin{bmatrix} A & 0 \\ 0 & 1 \end{bmatrix}$$

and a translation

$$T = \begin{bmatrix} I & b \\ 0 & 1 \end{bmatrix}$$

commute if, and only if, $Ab = b$. Thus, Chasles theorem says that we can express any euclidean transformation as a shift of origin followed by a commuting rotation and translation, i.e.,

$$\begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I & c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} A & d \\ 0 & 1 \end{bmatrix}$$

with $Ad = d$ and $\langle c, d \rangle = 0$. (This last condition serves to make c and d unique unless $A = I$.) Since the range space and null space of a skew symmetric matrix are orthogonal, the identity

$$\begin{bmatrix} I & c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} S & d \\ 0 & 0 \end{bmatrix} \begin{bmatrix} I & -c \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} S & d - Sc \\ 0 & 0 \end{bmatrix}$$

implies that every motion of the form $\exp \begin{bmatrix} S & d \\ 0 & 0 \end{bmatrix} \theta$ can be thought of as a screw motion with respect to some choice of origin [7].

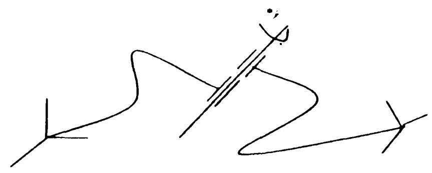


Figure 1.

Consider now robotic manipulators which consist of rigid bodies jointed at single degree of freedom joints (see figure 1). If we fix a right-handed triad of orthogonal vectors at the tip of each member of the chain, it is not too difficult to see that the euclidean transformation, which describes the position and orientation of the $(i+1)^{st}$ triad in terms of the i^{th}, is

$$\begin{bmatrix} M_i & b_i \\ 0 & 1 \end{bmatrix} \exp \begin{bmatrix} S_i & 0 \\ 0 & 0 \end{bmatrix} \theta_i$$

Thus, the triad fixed at the free end is related to that at the base by the product

$$T(\theta_1, \theta_2, \ldots, \theta_r) = M_1 \left(\exp \theta_1 \begin{bmatrix} S_r & 0 \\ 0 & 0 \end{bmatrix} \right) M_2 \left(\exp \theta_2 \begin{bmatrix} S_2 & 0 \\ 0 & 0 \end{bmatrix} \right) \ldots M_r \left(\exp \theta_r \begin{bmatrix} S_1 & 0 \\ 0 & 0 \end{bmatrix} \right)$$

where the M_i is the element of the euclidean group which maps the coordinate system at the end where it joins the $(i-1)^{st}$ element to the coordinate system where it joins the $(i+1)^{st}$.

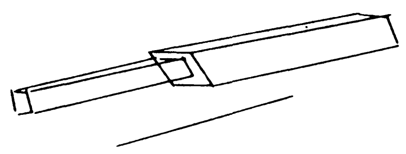


Figure 2

It is easy to see that $P(\exp M)^{-1}P^{-1} = \exp(PMP^{-1})$. Thus we can use the identity $M \exp N = (\exp MNM^{-1})M$ repeatedly to write

$$T(\theta_1, \theta_2, \ldots, \theta_r) = M e^{H_1\theta_1} e^{H_2\theta_2} \ldots e^{H_r\theta_r}$$

This product of exponentials formula not only applies to chains of the form shown in figure 1, but also applies to mechanisms containing prismatic joints (see figure 2) if we allow the product to include generators of the translation elements, i.e., factors of the form

$$\exp \begin{bmatrix} 0 & b_i \\ 0 & 0 \end{bmatrix} \theta_i$$

It is to be emphasized that any kinematic chain defines an M and the $H_1, H_2, \ldots, H_r$ with the H_i being elements of the Lie algebra of $\mathbf{E}(3)$ and the M and element of the Lie group $\mathbf{E}(3)$. Once a choice for the coordinate systems at the beginning and end of the chain

is made and a definition of which angles correspond to $\theta_i = 0$ is selected, those matrices characterize, and are characterized by, the chain.

An easy calculation shows that

$$\exp \begin{bmatrix} 0 & \theta & 0 & 0 \\ -\theta & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha\theta \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & \alpha\theta \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Because the one parameter family $(\cos\theta, \sin\theta, \alpha\theta)$ in $\mathbf{R}^3$ is not the zero locus of a set of algebraic equations unless $\alpha = 0$, it is not possible to eliminate transcendental functions from the study of screw motions. However, if the elements of the kinematic chain are all pure rotation or pure translation, it is possible to reexpress matters in such a way as to eliminate transcendental functions. Thus, we can algebraically parameterize the rotational one parameter subgroup as

$$\exp \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \theta = \begin{bmatrix} x & y & 0 & 0 \\ -y & x & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad ; \quad x^2 + y^2 = 1$$

and obviously we may describe the one parameter subgroups associated with translation in purely algebraic terms.

A basic problem in robotics is that of solving the equation in $\mathbf{E}(3)$

$$T = M \exp H_1\theta_1 \ldots \exp H_6\theta_6$$

for $\theta_1, \theta_2, \ldots, \theta_6$. This is called the inverse kinematics problem. M. Raghavan's notes discusses it in some detailed. From a mathematical point of view it is a special case of the following more general problem.

Problem A: Suppose that we have a representation of a k-dimensional Lie group G with algebra L and choose an ordered basis $L_1, L_2, \ldots, L_k$ for the algebra. One says that

$$T_1(v) = e^{L_1 v_1 + L_2 v_2 + \cdots + L_k v_k}$$

and

$$T_2(\theta) = e^{L_1\theta_1} e^{L_2\theta_2} \ldots e^{L_k\theta_k}$$

define coordinates of the first and second kind, respectively. If we assume that the G is an algebraic group in the sense that as a subset of $Gl(n)$ it can be characterized as the locus of solutions of a set of polynomial equations and if we assume that for each i $e^{L_i\theta_i}$ defines a set in G which is the locus of the solutions of a set of algebraic equations, then $T_2(\theta) = G$ is equivalent to a set of algebraic equations. A question of general interest is that of determining how the degree and Galois group of these equations depend on G and the ordered basis $L_1, L_2, \ldots, L_k$.

Example: If the group is the group of 3 by 3 orthogonal matrices and if

$$L_1 = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad ; \quad L_2 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad ; \quad L_3 = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

then the equations take the form

$$\begin{bmatrix} a & b & 0 \\ -b & a & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c & 0 & d \\ 0 & 1 & 0 \\ -d & 0 & c \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & e & f \\ 0 & -f & e \end{bmatrix} = \begin{bmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix}$$

$$a^2 + b^2 = c^2 + d^2 = e^2 + f^2 = 1$$

$$G^T G = I$$

In this case the degree is 8 and the Galois group is $\mathbf{Z}_2 \times \mathbf{Z}_2 \times \mathbf{Z}_2$. There is an explicit solution for the canonical coordinates of the second kind involving square roots.

3. Forces and Torques

We have emphasized the group of rigid motions and its associated Lie algebra in the previous section. This leads to the use of a certain matrix representation for this group and, by this choice, a certain matrix representation for the Lie algebra of velocities. Everyday experience, however, makes it believable that it is a combination of velocities and forces, or torques, which make things happen, and therefore that we need to see how forces and torques can best be thought of in this framework.

The familiar idea that work is the time integral of force times velocity, or torque times angular velocity, suggests that force/torque must belong to a space which is dual to the

velocity space. Since we are representing velocities via the equation

$$\frac{d}{dt}\begin{bmatrix} \Theta & x \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \Omega & v \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \Theta & x \\ 0 & 1 \end{bmatrix}$$

We should expect to see

$$\text{work} = \int_0^{t_1} \operatorname{tr} \begin{bmatrix} T & 0 \\ f & 0 \end{bmatrix} \begin{bmatrix} \Omega & v \\ 0 & 0 \end{bmatrix} dt$$

$$= \int_0^{t_1} \operatorname{tr}(T\Omega) + \langle f, v \rangle \, dt$$

Recall that if X and Y are n by n matrices and if P is invertible, then $\operatorname{tr} YX = \operatorname{tr}(P^{-1}YXP)$. As a consequence, when we change coordinates in $\mathbf{E}(3)$ according to

$$\begin{bmatrix} \Omega' & v' \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} \Omega & v \\ 0 & 0 \end{bmatrix} \begin{bmatrix} P & p \\ 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \Omega P & \Omega p + v \\ 0 & 0 \end{bmatrix}$$

we must change coordinates in the dual space of forces and torques according to the rule

$$\begin{bmatrix} P & p \\ 0 & 1 \end{bmatrix}^{-1} \begin{bmatrix} T & 0 \\ f & 0 \end{bmatrix} = \begin{bmatrix} T' & 0 \\ f' & 0 \end{bmatrix}$$

Working this out gives

$$\begin{bmatrix} P^{-1} & -P^{-1}p \\ 0 & 1 \end{bmatrix} \begin{bmatrix} T & 0 \\ f & 0 \end{bmatrix} = \begin{bmatrix} P^{-1}T - fP^{-1}p & 0 \\ f & 0 \end{bmatrix}$$

Of course if the transformation was an element of $\mathbf{E}(3)$, then P is orthogonal and $P^{-1} = P^T$.

The reference to "free" versus "bound" vectors which one sees in elementary statics books refers to the triangular nature of these transformations. Since Ω' does not depend on p, the angular velocity Ω is said to be a free vector; likewise, f' does not depend on p so force is also said to be a free vector. Linear velocity and torque are called bound vectors.

In one dimension the relationship between force and displacement is often quantified by a spring constant as in the familiar equation $f = kx$. Controlling the "spring constant" associated with various motions such as shaking hands and assembling a snap-together object is a critical part of motion control. In the robotics literature this is called compliance control and it is one of the more deeply studied aspects of the subject. [10-15]

From the above remarks we can see that for rigid motions in $\mathbf{E}^3$ the linear maps from velocities (linear and rotational) to forces/torques are maps from a six dimensional Lie algebra into its dual. If we assume, and this is certainly an important case, that the spring forces are derivable from a potential energy, then after a choice of coordinates the linearization of this map takes the form

$$\begin{bmatrix} f \\ T \end{bmatrix} = \begin{bmatrix} S & R \\ R^t & Q \end{bmatrix} \begin{bmatrix} \delta x \\ \delta\theta \end{bmatrix}$$

with S and Q being 3 by 3 symmetric matrices. It is this data which specifies the (linearized) compliance.

4. Uniform Coverage and Harmonic Maps

From section two we see that joint angles can be thought of as a specific choice of canonical coordinates of the second kind for the Lie group of rigid motions. From section three we see that forces/torques can be thought of as a linear functional on the tangent space of the group of rigid motions. If we restrict our attention to the case where we have a six degree of freedom kinematic claim with all joints being revolute, then we have a map of the six torus T^6 into $\mathbf{E}(3)$ given by

$$f : (\theta_1, \theta_2, \ldots, \theta_6) \mapsto M e^{A_1\theta_1} e^{A_2\theta_2} \ldots e^{A_6\theta_6}$$

This map has a linearization df. Dual to the $\dot\theta$'s are a set of torques $T_1, T_2, \ldots, T_6$ and dual to the velocity in $\mathbf{E}(3)$ we have a set of force-torques. These ideas are connected in the following important but elementary way. We can compute the rate of work done by the forces and torques either as $T_1\dot\theta_1 + \cdots + T_6\dot\theta_6$ or, as in the previous section, in terms of velocities and force/torques in the tangent space of the group of rigid motions. Because we must get the same answer in either case we see that if the velocities $\dot\theta_1, \dot\theta_2, \ldots, \dot\theta_6$ are mapped into velocities in $\mathbf{E}(3)$ via the Jacobian df, then the force/torques in $\mathbf{E}(3)$ must be mapped into torques on the cotangent space of T^6 by the adjoint of df. *This is just the law of levers applied to our situation.* It has nothing to do with the fact that f is expressible as the product of exponentials.

What makes one kinematic chain better than another? There are, of course, many ways one could attempt to answer this question and, judging from the diversity of kinematic chains which one sees in the animal world, it seems unlikely that there is a universal answer. Nonetheless, since one can think of (df) as representing the "velocity gain" which corresponds to the map from joint coordinates to end effector coordinates, it follows that (when it exists) $((df)^*)^{-1}$ is the "torque gain" for the same map. Since we expect the robot to be able to move quickly and apply reasonable force at all points of the work space, one might begin with the idea that df and $((df)^*)^{-1}$ should be as close as possible to being isometries, i.e., the chain should come as close as possible to having the same gain everywhere in the workspace. We discuss a way to make this idea precise in the next paragraph.

The map corresponding to the kinematic chain sends a torus into a subset of $\mathbf{E}(3)$. In general this map will have singularities and the image will not be of the same topological type as the domain. Nonetheless we can attempt to formulate an integral criterion

$$\eta = \int \ldots \int L(\theta_1, \theta_2, \ldots, \theta_6)\, d\theta_1 \ldots \theta_6$$

whose minimization would result in making the velocity gain or force gain as uniform as possible. One way to do this which is quite natural begins by noticing that as far as the θ's are concerned, there is an obvious distance measure on the flat torus, given by

$$(ds)^2 = \sum_i (d\theta_i)^2$$

(If the servomotors which drive the revolute joints are of different sizes, this might be a weighted sum but we ignore this generalization here.) On the other hand, in $\mathbf{E}(3)$ there is a natural one parameter family of Riemannian structures as discussed above. If we pick one of these and let its matrix representation be denoted by G, then minimizing

$$\eta = \int \ldots \int \mathrm{tr}[(df)^T G(df)]\, d\theta_1 \ldots \theta_k$$

is a natural way to enforce a condition that df is never too large. It can't be small everywhere and still be onto. Thus η may be taken to be a measure of the uniformity of the gain of a kinematic chain. This is the functional which plays the critical role in the definition of harmonic maps [13].

Example: (See [11].) A planar mechanism with three revolute joints is shown in figure 3. The x, y coordinates of the end effector are given by

$$
\begin{aligned}
x &= l_1 \cos\theta_1 + l_2 \cos(\theta_1 + \theta_2) + l_3 \cos(\theta_1 + \theta_2 + \theta_3) \\
y &= l_1 \sin\theta_1 + l_2 \sin(\theta_1 + \theta_2) + l_3 \sin(\theta_1 + \theta_2 + \theta_3)
\end{aligned}
$$

This map takes T^3 into the disk of radius $l_1 + l_2 + l_3$ provided the lengths do not satisfy any inequality of the form

$$l_i + l_j < l_k$$

If we use the metric $(ds)^2 = (dx)^2 + (dy)^2$ in the disk, it may be shown that the choice of l_1, l_2, l_3 which minimizes η subject to $l_1 + l_2 + l_3 = l$ is $l_1 = 6l/11$ $l_2 = 3l/11$ $l_3 = 2l/11$. This can be compared with the lengths of the three segments of the human index finger.

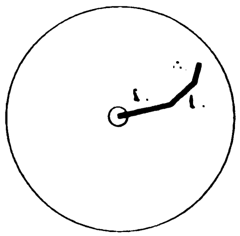


Figure 3: A three joint planar mechanism.

References

[1] Richard Paul, *Robot Manipulators: Mathematics, Programming and Control*, Cambridge, MA, MIT Press, 1981.

[2] R. W. Brockett, "Robotic Manipulations and the Product Exponentials Formulae," in *Lecture Notes in Control and Information Sciences*, *Proceedings of the International Symposium on Mathematical Theory of Networks and Systems*, Berlin, Springer-Verlag, 1984, pp. 120-127.

[3] K. H. Hunt, *Kinematic Geometry of Mechanism*, Oxford, England, Oxford University Press, 1978.

[4] J. M. Hervé, "Analyse structurelle des mécanismes par groupes de désplacements," *Mechanisms and Machine Theory*, Vol. 13, 1978, pp. 437-450.

[5] D. L. Pieper, *The Kinematics of Manipulation Under Computer Control*, Stanford Artificial Intelligence Laboratory, Stanford University, AIM 72, 1968.

[6] D. E. Whitney, "Quasi-static Assembly of Compliantly Supported Rigid Parts," *Journal of Dynamic Systems, Measurement and Control*, 1982, pp. 65-77.

[7] M. T. Mason, "Compliant Motion," in *Robot Motion Planning and Control*, (Brady *et al.*, eds.), Cambridge, MA, MIT Press, 1983.

[8] J. Loncaric, "Geometrical Analysis of Compliant Mechanisms in Robotics," Ph.D. Thesis, Division of Applied Sciences, Harvard University, 1985.

[9] T. Yoshikawa, "Manipulability of Robot Mechanism," *International Journal of Robotics Research*, Vol. 4-2, 1985.

[10] R. Vijaykumar, K. J. Waldron, and M. J. Tsai, "Geometric Optimization of Serial Chain Manipulator Structures for Working Volume and Dexterity," *International Journal of Robotics Research*, Vol. 5-2, 1986.

[11] Frank C. Park and R. W. Brockett, "Harmonic Maps and the Optimal Design of Mechanism," *Proceedings of the IEEE Conference on Decision and Control*, December 1989, pp.

[12] B. Paden and S. Sastry, "Optimal Kinematic Design of GR Manipulators," *Robotics Research*, Vol. 7, No. 2, 1988, pp. 43-61.

[13] J. Eells and J. H. Sampson, "Harmonic Mappings of Riemannian Manifolds," *American Journal of Mathematics*, Vol. 86, 1964, pp. 109-160.

A Formal Language for Motion Description

1. Review of Formal Models

There are a number of formal models in common use defining what is meant by computable. Perhaps the simplest of these is that of a finite state machine.

Let U be a finite set. We define U^* to be the set of all finite strings of elements of U. We think of U^* as being a monoid with concatenation being the binary operation. Since we impose no bound on the length of the strings in U^*, this set is infinite. Let Y be a second finite set. We consider a function $f : U^* \to Y$ and ask about the difficulty of computing (evaluating) f. A sample question, vaguely phrased, is this: let $U = Y = \{0,1\}$ and let f_1 be the function which assigns to an element of U^* the mod 2 sum of its entries; let f_2 be the function which assigns the value 1 to a string consisting of an arbitrary number of ones followed by exactly the same number of zeros and assigns the value zero to all other strings. Which of these two functions is "easier" to compute?

By a finite automaton one understands a five tuple $(U, X, Y, \delta, \gamma)$ where U, X and Y are finite sets, $\delta : X \times U \to X$, $\gamma : X \to Y$. The operation of the automaton is defined by

$$x(k+1) = \delta[x(k), u(k)]; \ \gamma[x(k)] = y(k)$$

which together with a specification of an initial state defines a mapping from U^* into Y. An automaton with a specified initial state is called an *initialized automaton*.

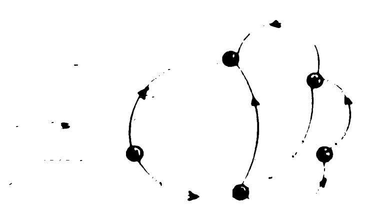


Figure 1. A five state finite automaton with some transitions missing.

We visualize the action of the automaton with the help of a diagram. The elements of X are represented by nodes. The branches connecting the nodes indicate the action of the

automaton on receipt of a particular input from U. The idea can be seen in figure 1. This together with a definition of γ defines the automaton.

Suppose that E and F are subsets of the free monoid U^*. By $E \cup F$ and $E \cdot F$ we understand the subsets

$$E \cup F = \{u | u \in E \text{ or } u \in F\}$$

$$E \cdot F = \{u | u = ef \text{ with } e \in E,\ f \in F\}$$

we also define E^* to be the set of all finite strings of elements of E

$$E^* = \{u | u = e_1, e_2, \ldots, e_n ;\ e_i \in E\}$$

In other words E^* is the smallest submonoid of U^* which contains E.

One says that a subset L of U^* is *regular* if it is finite or if it can be generated from a finite subset of U^* by a finite number of applications of the operations $\cup$, $\cdot$, and $*$.

A subset L of U^* is a *finite state language* if there exists an initialized finite state machine $(U, X, Y, \delta, \gamma)$ with $Y = \{0, 1\}$ such that for any input string in L the output is 1 and for any input string not in L the output is 0.

2. Kinematic Machines and Motion Languages

In the first section, we saw that a finite automaton is a simple model of computation, suitable for the recognition of some languages. Our goal now is to structure our notion of motion accordingly—in particular, to identify formal languages that describe certain classes of motion and to construct machines that model the computation, or generation, of such classes of motion. Given such specially tailored linguistic and computational devices, we should be better able to evaluate motion programming languages with respect to their expressiveness, efficiency, complexity, etc.

The first task at hand is to explain what we mean by a *formal motion language*. A motion language, simply stated, is a set of symbolic strings that represent idealized motions. Stated with more precision, a motion language L is a subset of a free monoid on a given motion alphabet Σ, that is, $L \subseteq \Sigma^*$. Symbols of alphabet Σ represent segments of motion, having specific durations and various other characteristics. Although this portrayal of a motion language appears to go well with the traditional perspective of a formal language, this is not quite the usual sort of formal language. While one normally does not think of the symbols of an alphabet in formal language theory in terms of their effect on a particular machine, one must do so in the case of motion languages. Consequently we define motion languages only in the context of an associated physical model, which we call a *kinematic machine*.

In a very broad sense, a kinematic machine can be thought of as the continuous analog of a finite automaton. Whereas finite automata are governed by difference equations of the form

$$x_{k+1} = \delta(x_k, u_k); \;\; y_k = \gamma(x_k)$$

kinematic machines are governed by differential equations of the form

$$\dot{x} = \Sigma g_i(x) u_i; \;\; y = h(x)$$

Recall that one often defines a finite automaton as a quintuple $(U, X, Y, \delta, \gamma)$, where U is a finite input alphabet, X is a finite set (the machine states), Y is a finite output alphabet, $\delta : X \times U \to X$ is a transition function that maps current states and inputs into next states, and $\gamma : X \to Y$ is a labelling that associates each machine state with an output symbol. In a like manner, we may denote a kinematic machine as a quintuple (U, X, Y, G, h), where U is an input (control) space, X is an actuator coordinate space (state space), Y is an output space, G is a matrix whose entries may depend on x, and $h : X \to Y$ transforms actuator coordinates to output coordinates. These characterizations of machine behavior mark their essential similarities and differences. Chief points to note are (1) the state space for a finite automaton is a discrete, finite set whereas the state space for a kinematic machine is a continuum, (2) both finite automata and kinematic machines prescribe a transformation from the state space of the machine state X to the output space Y, and (3) an initial state condition (x_0 or $x(0)$) and an input sequence $u_0 \ldots u_n \in U^*$ determine the state evolution for both finite automata and kinematic machines.

Henceforth, we consider kinematic machines to be models of the form

$$\dot{x} = G(x)u; \;\; y = h(x)$$

where u, x, and y are functions of time. The input u is subject to some regularity condition say u belongs to the set of all piecewise continuous functions. x has its range in n-dimensional actuator coordinate space X. y has its range in p-dimensional output space Y (Y typically has position coordinates and force coordinates). G is an n by m matrix that depends only on x. $h : X \to Y$ is an actuator-to-output coordinate transformation.

The atoms of our motion language include triples of the form (u, k, T). If at time t_0 the machine receives an input (u, k, T), the state x evolves according to

$$\dot{x} = G(x)(u + k(y)); \;\; y = h(x); \;\; 0 \leq t \leq T.$$

If at time t_0 the machine receives an input string $(u_1, k_1, T_1) \ldots (u_i, k_i, T_i)$, then x evolves according to

$$\begin{array}{ll} \dot{x} = G(x)(u_1 + k_1(y)); & t_0 \leq t \leq t_0 + T_1 \\ \vdots & \vdots \\ \dot{x} = G(x)(u_i + k_i(y)); & t_0 + T_1 + \ldots + T_{i-1} \leq t \leq t_0 + T_1 + \ldots + T_i \end{array}$$

where $k : Y \to U$ is a feedback function belonging to a function space K. With this encapsulation, we see that provisions have been made for the generation of motion (open-loop and closed-loop motion are both special cases). See figure 2.

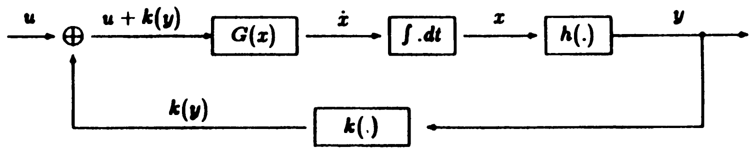


Figure2: Feedback loop associated with a kinematic machine.

For the sake of simplicity, and to give one concrete example of a motion language, we define the language $M1$ to be the set of strings of triples (u_i, k_i, T_i), where u_i are constant, k_i are linear functions from Y to U (i.e., identifiable with matrices), and T_i are positive real numbers.

Example: The following is a depiction of a specific kinematic machine which has been built to test these theories. In figure 3, a robotc device is shown with two 2-link fingers and pressure sensors mounted at the tips of each. Here actuator coordinates are θ_1, θ_2, θ_3, and θ_4, and output coordinates are y_1 through y_6. Then the kinematic machine is governed by these equations.

$$\begin{bmatrix} \dot\theta_1 \\ \dot\theta_2 \\ \dot\theta_3 \\ \dot\theta_4 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix}; \quad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = h\left(\begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix}\right) = \begin{bmatrix} 5\cos\theta_1 + 3\cos(\theta_1 - \theta_3) \\ 5\sin\theta_1 + 3\sin(\theta_1 - \theta_3) \\ 7 - 5\cos\theta_2 + 3\cos(\theta_4 - \theta_2) \\ 5\sin\theta_2 + 3\sin(\theta_4 - \theta_2) \\ \text{pressure}_1(\theta_1, \theta_2, \theta_3, \theta_4) \\ \text{pressure}_2(\theta_1, \theta_2, \theta_3, \theta_4) \end{bmatrix}.$$

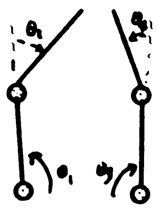


Figure 3. A four dgree of freedom kinematic machine.

3. World Models

We want a model of the world which will serve to describe the force/displacement relationships associated with robots interacting with objects. We do not want a detailed model but we need one which is qualitatively correct. We begin with a few points.

(i) It is to be expected that both the robot and the objects it manipulates are compliant to some extent.

(ii) Forces and torques are vector quantities. In many situations these quantities can be thought of as being derived from a single function, a potential, by way of

$$f = -\nabla V$$

(iii) In some situations one achieves an economy of thought by transferring properties of a moving object to the fixed objects which populate the workspace – in effect giving them a shadow and shrinking the moving object to a point. We will use this kind of thinking to model compliance.

(iv) We must make a distinction between world coordinates and robot coordinates. We must not assume that the robot has a perfect description of the world.

Our model of the world is that it is free space populated with a number of rigid objects B_i which are, to begin with, fixed in space. We assume that we have chosen a euclidean coordinate system for the world with coordinates (w_1, w_2, w_3). Neither the locations of the objects nor their shapes are known with precision. Associated with each ***fixed object*** is a potential function $V_i : \mathbf{E}^3 - B_i \to \mathbf{R}$ which maps the exterior of the i^{th} object into the real line according to the formula

$$V_i(w) = \begin{cases} \frac{1}{d} - \frac{d}{\epsilon^2} + \frac{2}{\epsilon^2} & \text{if } w \text{ is of distance} < d \text{ to } B_i \\ 0 & \text{if } w \text{ is of distance } d > 0 \text{ from } B_i \end{cases}$$

(Although V_i is not defined on the interior of B_i, one might think informally that it takes on the value $+\infty$ there.)

When the robot is in position y, it feels a force due to object B_i which is

$$f_i = -\nabla V_i|_y$$

We also will want to consider world models with ***movable rigid objects.*** Again, in this case we must be careful not to assume that the location or the shape of the object is known exactly. In this case we can associate with each movable object B_i a potential as above, and also a convex subset K of the space of forces and torques such that if a net applied force/torque belongs to this set, support and frictional forces will keep B_i fixed. Thus the description of a movable rigid object is more complex than that of a fixed rigid object by virtue of the need for K.

Example: Consider a world model which consists of one movable object in $\mathbf{E}^2$. The object is a disk and of radius $2 \pm .1$. The associated K is, forces related to the center,

$$-1 \le f_x \le 1$$
$$-1 \le f_y \le 1$$
$$-1/2 \le T \le 1/2$$

Exercise: Consider a disk of radius one centered at $w_1 = w_2 = 0$ and its associated potential $V(w_1, w_2)$ given by the formula of this section. Compute the associated Hessian, $H(w_1, w_2)$. Establish a tolerance μ such that if $(w_1 - z_1)^2 + (w_2 - z_2)^2 < \mu$, then the eigenvalues of

$$M(w_1, w_2, z_1, z_2) = -H(z_1, z_2) \cdot H(w_1, w_2)$$

have negative real parts.

4. Stability Issues; The Potential Method

We begin by defining three machines. These examples are point machines in that they occupy no volume. First consider a kinematic machine which operates in one dimension and senses its position and force

$$\dot{x} = u \; ; \; y_1 = x \; ; \; y_2 = f \qquad \text{(MACHINE I)}$$

with f being the force felt by the machine. Secondly, consider a machine which operates in the plane and senses position and force

$$\begin{aligned} \dot{x}_1 &= u_1 \; ; \; y_1 = x_1 \; ; \; y_3 = f_1 \\ \dot{x}_2 &= u_2 \; ; \; y_2 = x_2 \; ; \; y_4 = f_2 \end{aligned} \qquad \text{(MACHINE II)}$$

Here f_i is the force in the x_i direction. Finally, we consider a three degree of freedom machine which operates in the plane but which admits rotation as well as translation. It senses position and angle as well as force and torque

$$\begin{aligned} \dot{x}_1 &= u_1 \; ; \; y_1 = x_1 \; ; \; y_4 = f_1 \\ \dot{x}_2 &= u_2 \; ; \; y_2 = x_2 \; ; \; y_5 = f_2 \\ \dot{x}_3 &= u_3 \; ; \; y_3 = x_3 \; ; \; y_6 = \tau \end{aligned} \qquad \text{(MACHINE III)}$$

Now f_i is the force in the x_i direction and τ is the torque felt by the machine.

Suppose that we have a world model of the form discussed in the previous section. Consider the response of Machine I to the language atom (u, k, T) $(1,[0,2],10)$. According to our conventions, this gives an equation of motion

$$\dot{x} = 1 - 2\nabla V \qquad 0 \le t \le 10$$

We cannot, of course, say what the trajectory will be until we specify a world model. Say that the world model has an object of length 3 centered at $w = 9$, say that ϵ is .5 and say that $x(0) = -1$. What will happen on $t \in [0, 10)$ is sketched in figure ?. Without making any difficult calculations we can assert that $x(10)$ is between 8.0 and 9.5, that it is in contact with the object and that neither of these predictions is altered if the position of the center of the interval is changed from 9 to $9 + \alpha$ as long as $|\alpha| < 2$. Likewise T can

be changed to any number greater than 8.5 and the conclusions are not altered. *This is an extremely robust situation.*

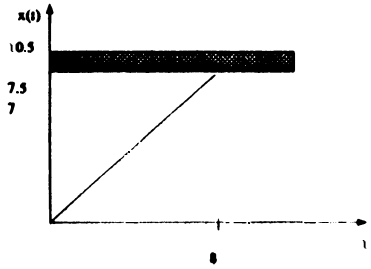


Figure 4: The motion which results from (u,k,T).

It is likewise true that the behavior is not strongly dependent on the world model. The $1/d - 1/\epsilon$ potential could be altered to be any differentiable strictly monotonic function which goes from zero at .5 units from the edge to infinity at the edge. We leave the proof of this as an exercise.

In the two dimensional situation matters become more delicate. Consider figure 5.

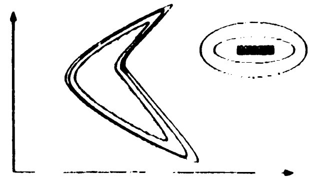


Figure 5: Object and Equipotential lines in E^2

If we turn Machine II loose in this environment, what will happen? In order to analyze this situation it is useful to associate a potential with u and to analyze the sum of $V(B_i)$ and the u-dependent potential. This works as follows. Notice that if we think of V as being expressed as a function of x then

$$\dot{x} = -\nabla V(B) + u = -\nabla(V(B) - x^T u)$$

Thus we can think of this rule as bringing x to the bottom of the valley of the function $V(B) - x^T u$).

We still have k at our disposal, however, and it can be used to advantage. What we see is that in the case of a convex object such as (b) above, the function $V(B) - x^T u$ has a saddle point and not a local minimum near the point of contact. Notice that near this saddle point

$$V_u = V(B) - x^T u \sim q + \frac{1}{2}(\delta x)' H (\delta x) \quad ; \quad \delta x = (x - x_0)$$

with $H = H^T$ and H having one positive and one negative eigenvalue. Near this saddle point, for a given $k(y) = Ky$ we have

$$\delta \dot{x} = -KH\delta x$$

Notice that if we would choose K to be H, then

$$\delta \dot{x} = -H^2 \delta x$$

and regardless of the eigenvalues of H, those of H^2 are nonnegative (zero only if H has a zero eigenvalue). Thus this equation is asymptotically stable.

We do not know the location of B exactly so we do not know V_u exactly so we do not know H exactly. Even so, except for the degenerate case (i.e., the case where H has a zero eigenvalue) we can say that the solutions of

$$\dot{x} = -(H + \delta H)\nabla V_u$$

will approach an equilibrium point.

Definition: We will call the choice (u, k, T) where $k(y) = Hy$ *descent programming*.

Although we will not go into the three dimensional version of these problems here, it should be clear that descent programming will work in three dimensions as well.

5. Changes of Coordinates

The essence of graceful motion is the smooth coordination of the available degrees of freedom. From one point of view the basic idea of a kinematic machine requires nothing further; better coordination is just a matter of choosing better sequences of (u_i, k_i, T_i). On the other hand, computational considerations and programming efficiency suggest that one should incorporate into the language some tools involving coordinate change and trajectory smoothing to ease the programmers task.

In the basic definition of a kinematic machine there is a confounding of the effects of k and u in the sense that a given $\dot{x}$ can be the result of a large k and small u or a large u and small k. To disambiguate this we need to reflect on the different mechanisms which are to be used to generate kx and u and to consider the different response speeds.

In this chapter, in fact, let us write the differential equations which result from applying $(u, [k_1, k_2], T)$ to Machine II as follows

$$\dot{x} = k_1(x - x_0) + k_2\nabla V + (u + k_1 x_0)$$

We may think of the equation as follows:

$$\text{Velocity} = \text{posture term} + \text{navigation term} + \text{planning term}$$

In some cases it may be useful to split x into two terms according to

$$x = \begin{bmatrix} x_{pose} \\ x_{plan} \end{bmatrix} \quad ; \quad x_{plan} \in Ker\ k_1 \quad ; \quad x_{pose} \in Range\ k_1$$

In the language of psychology the term involving x_{pose} is proprioceptive feedback and k_1 would be said to define a conditioned reflex. It is to be observed that kinematic machines must have a certain level of complexity if posture is to be a worthwhile idea.

Because the u and k elements in (u, k, T) are taken to be constants the equations of motion for the machine

$$\dot{x} = k_1 x + k_2 \nabla V + u$$

generate trajectories of a particular type. For example, if we let k_1 and k_2 be zero then $x(t)$ will take the form $x(t) = x_0 + u \cdot t$, a straight line in x-space. On the other hand, if we change coordinates, letting x be replaced by $z = g(x)$, then

$$\dot{z} = (\partial g/\partial x)\,\dot{x} = (\partial g/\partial x)u$$

If we were to trade the input u for $v = G(g^{-1}(z))u$, then we get

$$\dot{z} = v$$

for the kinematics of the machine in the new coordinate system. In this case a constant input gives rise to a straight line in z-coordinates. It is desirable, at the same time, to change the definition of the force in such a way as to maintain the energy-theoretic meaning of the inner product between input and force output. This means letting $y = \nabla V$ with the gradient being with respect to x. Interpreting triples (v, k, T) as inputs to this machine effectively adapts the programming problem to x-coordinates.

More formally, if g is a differentiable one to one mapping with a differentiable inverse, we will allow it to be an input to our machine, interpreting it as an instruction to change coordinates before executing the next triple (u, k, T). Thus the admissible input strings now include sequences including subsequences of the form

$$\ldots (u_1, k_1, T_1) g_1 (u_2, k_2, T_2)(u_3, k_3, T_3) g_2 \ldots$$

Of course it takes additional computation to support such an addition to the language. The essential additional computation is the evaluation of the Jacobian G of g and the mapping of u by G.

6. Recording and Mapping

In this section we are going to assume that all machines measure their position in addition to whatever else they measure. Consider a kinematic machine

$$\dot{x} = u \ ; \ y = f$$

So far we have in the motion language elements (u, k, T) and change of coordinate transformations g. We now add the final ingredient, R, T, which asks the computational structure which supports the machine to store the measurement record for the past T units of time and a variance matrix Σ which provides some indication as to how reliable x can be expected to be.

The effect of reaching the i^{th} occurrence of an R, T will be to record the measurement record as the i^{th} entry in a file.

We now turn to a discussion of how the record function might be used in map making. Consider the results of a robot which has made measurements about the world. After a certain period, the map file will contain some locations and some variances associated with those locations. As distance information is recorded, we create a directed graph. Our notation for this is $G = (V, E)$, where V is the set of vertices of G and E is the set of edges

of G. Vertices of G are labelled with locations, and edges of G are labelled with means and variances of distances between locations. If we are making a two-dimensional map, these means are 3-vectors.

Initially, the graph G is just a root vertex v_0; it has no edges. The root vertex is labelled with an initial location l_0. The graph G is then updated incrementally. Each time a measurement is taken, the current distance is computed from the most recent pair of location measurements. Figure 6 presents a typical picture. Associated with the entire set of measurements is a Gaussian density with a certain mean, d, and certain variance, Σ_d. By introducing an orientation on the edges we can associate a vector distance with each edge. If the map is planar, we are modeling the d's as two component vectors. The first component being the x component of the distance, the second being the y component.

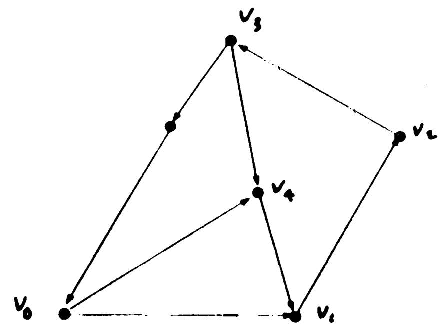


Figure 6. A graph representing the contents of a data file.

The measurements $d_1, \ldots, d_n$ are not, however, independent since geometry imposes some constraints. These constraints can be expressed easily by selecting a *maximal tree* in the graph; i.e., a subgraph which contains no loops and which cannot be enlarged without losing this property. (The maximal tree may not be unique.) We denote by d_t the vector of these distances along the branches of the maximal tree. Using the fact that the sum of the d_i's around any closed loop is zero, we can find an incidence matrix A such that

$$Ad_t = d$$

Notice that the entries of A need only take on the values $(+1, 0, -1)$. If there are k branches in the maximal tree and n branches altogether, then A is n by k and of rank k.

At a certain point in time, we may wish to analyze the map. For this purpose, we identify a spanning tree (or maximal tree) T. Any one of the remaining edges forms a cycle when added to the tree. The a priori Gaussian density on the edge measurement is, say

$$p(d) = N e^{-\frac{1}{2}(d-d)^T \Sigma^{-1}(d-d)}$$

However, when we impose the constraint that d can be expressed as Ad_t the density becomes

$$p(d) = N_1 e^{-\frac{1}{2}(Ad_t - d)^T \Sigma^{-1}(Ad_t - d)}$$

By maximizing this we see that the most likely tree-distance vector d_t is given by

$$d_t = (A^T\Sigma^{-1}A)^{-1}A^T\Sigma^{-1}d$$

This formula can be interpreted as giving a *rationalization* of the raw measurements represented by d. The effect of this process is to shift the "best guess" of d from d to

$$\hat{d} = A(A^T\Sigma^{-1}A)^{-1}A^T\Sigma^{-1}d$$

The correction being

$$(\hat{d} - d) = (A(A^T\Sigma^{-1}a)^{-1}A^T\Sigma^{-1} - I)d$$

Now suppose that we have a graph G which contains information about means and variances of distances. For any k, the expected value for the observed distance at the kth edge in G is d_k

$$d_k = d^t d_T$$

and the variance associated with this estimate of the distance along the kth edge is

$$\sigma_k = c^t\Sigma_T c = c^t(A^t\Sigma^{-1}A)^{-1}c.$$

Assume that we measure the length of the kth edge with an associated variance σ_m. The posterior estimate for the distance measurement at the kth edge of G is a linear combination of the prior estimate d_k and the actual distance measurement d_m.

$$\hat{d}_k = \frac{\sigma_m}{\sigma_m + \sigma_k}d_k + \frac{\sigma_k}{\sigma_m + \sigma_k}d_m$$

This equation shows how the kth edge gets updated. If the kth edge was not in the tree, then there are cycles in G that depend on this edge, and the edges of these cycles should

reflect the information gained from the measurement d_m. This is done by updating the edges of T, as the edges of G depend linearly on those of T.

If the observation d_m was made on a new edge of G, then d and Σ get augmented.

$$d = \begin{bmatrix} d \\ d_m \end{bmatrix} = \begin{bmatrix} A \\ c^t \end{bmatrix} d_T$$

Because d_m is independent of past measurements, covariances are zero in the new Σ.

$$\Sigma = \begin{bmatrix} \Sigma & 0 \\ 0 & \sigma_m \end{bmatrix}$$

The new tree variance Σ_T retains the same dimensions as before.

$$\Sigma_T = \left(\begin{bmatrix} A^t & c \end{bmatrix} \begin{bmatrix} \Sigma & 0 \\ 0 & \sigma_m \end{bmatrix}^{-1} \begin{bmatrix} A \\ c^t \end{bmatrix}\right)^{-1} = \left(\Sigma_T^{-1} + c\sigma_m^{-1}c^t\right)^{-1}$$

By the matrix inversion lemma, this equation can be expressed as a formula which requires only one inverse.

$$\Sigma_T = \Sigma_T - \Sigma_T c(\sigma_m + \sigma_k)^{-1}c^t\Sigma_T$$

Finally, the new estimate for tree distances is a linear combination of the old estimate and the measurement.

$$\hat{d}_T = \left(\Sigma_T^{-1} + c(\sigma_m - c^t\Sigma_T c)^{-1}c^t\right)^{-1}\left(\Sigma_T^{-1}d_T + c(\sigma_m - c^t\Sigma_T c)^{-1}c^t d_m\right)$$

This is supposed to be the conditional expectation of d_T given d_m. It can be simplified.

$$\hat{d}_T = (\Sigma_T - \Sigma_T\sigma_k\Sigma_T)(\Sigma_T^{-1}d_T + c(\sigma_m - c^t\Sigma_T c)^{-1}c^t d_m)$$

Reference

R. W. Brockett, "On the Computer Control of Movement," *Proceedings of the IEEE Conference on Robotics and Automation*, 1988.

Manipulator Kinematics

Madhusudan Raghavan

Abstract. Manipulator kinematics is the branch of mechanics which provides the analytical tools for describing a manipulator's geometry and motion. We present the two fundamental problems in manipulator kinematics viz. the forward kinematics problem and the inverse kinematics problem. We show how these problems may be formulated as principal systems of multivariate polynomials. We describe Sylvester's dyalitical elimination, an algebraic method for solving systems of multivariate polynomials. We then present some structure theorems on the equations describing series revolute-jointed manipulators. Finally we show how these theorems may be used along with the dyalitical elimination method to solve the inverse kinematics problem for the six-revolute-jointed series manipulator of general geometry.

1. Introduction. Manipulator kinematics is the branch of robotics which provides the analytical tools for describing a manipulator's geometry and motions. These tools are essential for robot motion planning and control. Formally kinematics is the branch of mechanics which treats the phenomenon of motion without regard to the cause of the motion. In kinematics there is no reference

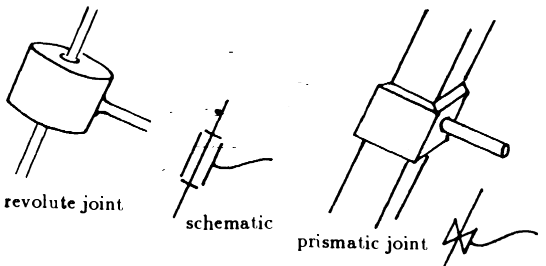


Figure 1: Joints

to mass or force. The concern is only with relative positions and their changes. A manipulator is comprised of bodies or members which are connected to one another by joints. Most manipulators have revolute and prismatic joints. A revolute joint is similar to a hinge. It permits rotation of one body relative to another about an axis fixed in both bodies. No relative translation is permitted. A prismatic joint permits relative translation but no rotation. Figure 1 shows revolute and prismatic joints along with schematic or simplified diagrams. A manipulator usually has a special member which interacts with the environment and does useful work. This member is called the hand or end-effector. A manipulator consisting of a sequential chain of links and joints (like the human arm) is called a series manipulator. A manipulator in which two or more series chains of links and joints connect the hand to the ground is referred to as an in-parallel manipulator. A hybrid manipulator consists of combinations of series and in-parallel subsystems. Figure 2 shows schematic drawings of series, in-parallel,

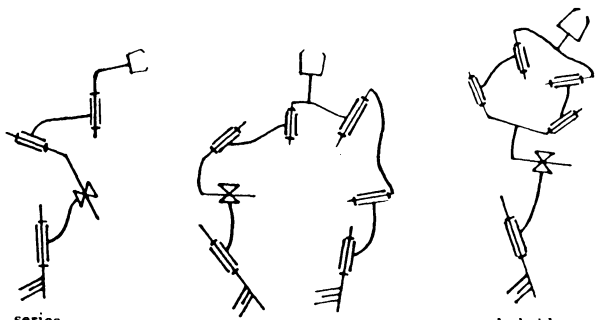


Figure 2: Examples

and hybrid manipulators. We restrict the discussion to series manipulators with rigid members and revolute joints. We present the two fundamental problems in manipulator kinematics, viz. the forward kinematics problem and the inverse kinematics problem. We show how the latter may be formulated as a system of n independent polynomials in n variables with the coefficients being real numbers. Such multivariate polynomial systems may be solved by algebraic elimination tools such as Gröbner Bases (BUCHBERGER[1985]), the multivariate resultant (CANNY[1988]) and Sylvester's dyalitical method (SALMON[1885].) The most difficult problem in series manipulator inverse kinematics is that of the spatial six-revolute-jointed manipulator. We present a solution that uses dyalitical elimination and exploits the structure of the polynomials describing the manipulator. To prepare the reader for this solution, we present a brief description of dyalitical elimination. We then present some results on the structure of the polynomials describing a general spatial n-revolute-jointed series manipulator. Finally we show how these two sets of ideas may be combined into an elegant solution.

2. The forward and inverse kinematics problems. Although manipulators are in general spatial or three dimensional devices, we describe the forward and inverse kinematics problems with the help of a planar manipulator, i.e. a manipulator in which each link moves in a plane parallel to some reference plane fixed to the ground. This facilitates the presentation and there is no loss of generality since the fundamental concepts extend quite naturally to manipulators in which the links undergo three dimensional motion. Figure 3 is a schematic drawing of a planar manipulator with three revolute joints. The link with the

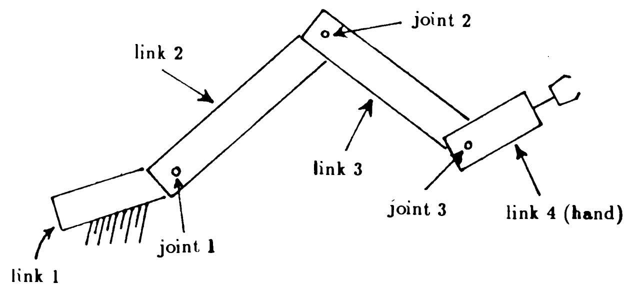


Figure 3: Planar Revolute-Jointed Manipulator

hatch marks is fixed to the ground and therefore remains stationary. Let us refer to it as link 1. Link 1 is connected to link 2 by means of a revolute joint (joint 1), the axis of which is normal to the plane of the paper. This joint permits link 2 to rotate relative to link 1. Similarly links 3 and 4 are connected by revolute joints 2 and 3 to links 2 and 3 respectively. The axes of joints 2 and 3 are also normal to the plane of the paper. Figure 4 is a side view of the planar manipulator showing the planes of motion of links 2, 3, and 4. These planes are parallel to each other. In order to develop a mathematical description of the manipulator we affix Cartesian coordinate systems to each link. These systems may be located and oriented arbitrarily on the links. We proceed with the arrangement

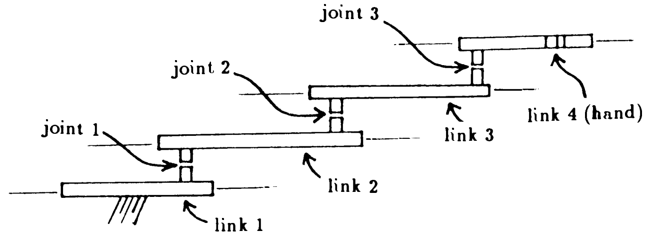


Figure 4: Side View of Planar Manipulator

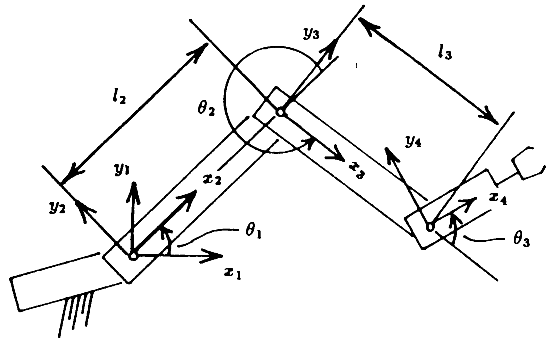


Figure 5: Planar Manipulator with Coordinate Systems

shown in Figure 5. Coordinate system x_i-y_i is located on link i, $i = 1, 2, 3, 4$. The origins of coordinate systems 1 and 2 are coincident. The position vector of the origin of coordinate system 3 in coordinate system 2 is $\begin{pmatrix} l_2 \\ 0 \end{pmatrix}$ and that of coordinate system 4 in system 3 is $\begin{pmatrix} l_3 \\ 0 \end{pmatrix}$. The forward and inverse kinematics problems are formulated as questions regarding the relative motions of these coordinate systems. The rotation of link $i+1$ relative to link i is measured by the angle θ_i. Let P be a point on link 4 with position vector $\mathbf{p}_4$ in coordinate system 4. Then $\mathbf{p}_3$ the position vector of P in coordinate system 3 is related to $\mathbf{p}_4$ by the following relation:

$$\mathbf{p}_3 = \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 \\ \sin\theta_3 & \cos\theta_3 \end{pmatrix} \mathbf{p}_4 + \begin{pmatrix} l_3 \\ 0 \end{pmatrix}. \tag{1}$$

The reader may verify this by examining the vectors shown in Figure 6. (1) may be rewritten as

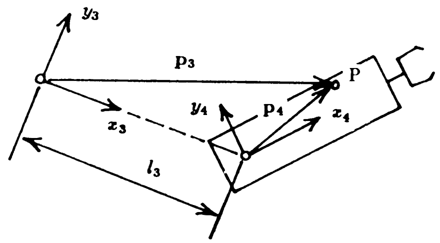


Figure 6: Position Vectors of P in Systems 3 and 4

$$\begin{pmatrix} \mathbf{p}_3 \\ 1 \end{pmatrix} = \begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & l_3 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{p}_4 \\ 1 \end{pmatrix}. \tag{2}$$

The matrix $\begin{pmatrix} \cos\theta_3 & -\sin\theta_3 & l_3 \\ \sin\theta_3 & \cos\theta_3 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ permits a change of coordinates from system 4 to system 3. Let us refer to it as ${}^4_3\mathrm{T}$. By similar arguments ${}^3_2\mathrm{T}$, the coordinate transformation matrix from system 3 to system 2 is $\begin{pmatrix} \cos\theta_2 & -\sin\theta_2 & l_2 \\ \sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{pmatrix}$,

and ${}^2_1\mathrm{T}$ is equal to $\begin{pmatrix} \cos\theta_1 & -\sin\theta_1 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$.

The matrices ${}^{i+1}_i\mathrm{T}, i = 1, 2, 3$, may be used to change coordinates from system 4 (the hand) to system 1 (the ground) by means of the following equation:

$${}^2_1\mathrm{T}\,{}^3_2\mathrm{T}\,{}^4_3\mathrm{T} = {}^4_1\mathrm{T}. \tag{3}$$

It governs the motion of the hand relative to the ground. The forward and inverse kinematics problems may now be defined with respect to (3). The forward problem is: given the lengths l_2, l_3, and the joints angles $\theta_1, \theta_2, \theta_3$, compute ${}^4_1\mathrm{T}$, the position and orientation of the hand relative to the ground. Clearly the forward kinematics problem is straightforward. It requires the evaluation of ${}^{i+1}_i\mathrm{T}, i = 1, 2, 3$, and the evaluation of the matrix product ${}^2_1\mathrm{T}{}^3_2\mathrm{T}{}^4_3\mathrm{T}$. On the other hand the inverse kinematics problem is fairly complicated. The problem statement is: given l_2, l_3, and ${}^4_1\mathrm{T}$, the hand position and orientation relative to the ground, find the corresponding values of the joint angles $\theta_1, \theta_2, \theta_3$. Equating the entries on the left and right-hand sides of rows 1 and 2 in (3) yields six equations in the three variables $\theta_1, \theta_2, \theta_3$. Only three of these equations are independent. This is because the (1,1), (1,2), (2,1), (2,2) submatrix on either side of (3) is an element of $SO(2)$, the group of orthogonal 2×2 matrices with determinant equal to $+1$.[1] Therefore only one of the four entries in this submatrix is independent. Equating the left and right-hand sides of this submatrix in (3) yields only one independent equation instead of four. Any one of the four equations may be treated as the independent one. Let us arbitrarily pick the (2,2) equation. This equation along with the (1,3) and (2,3) equations give us a system of three independent equations in the three variables $\theta_1, \theta_2, \theta_3$. These three equations are

$$l_2\cos\theta_1 + l_3\cos(\theta_1+\theta_2) = {}^4_1t_{13}, \tag{4}$$

$$l_2\sin\theta_1 + l_3\sin(\theta_1+\theta_2) = {}^4_1t_{23}, \tag{5}$$

$$\cos(\theta_1+\theta_2+\theta_3) = {}^4_1t_{22}, \tag{6}$$

where ${}^4_1t_{ij}$ is the $(i,j)^{th}$ entry of ${}^4_1\mathrm{T}$. The inverse kinematics problem requires that we solve these three equations for θ_1, θ_2, and θ_3. (4), (5), and (6) may be rewritten as polynomials by making the substitutions $\sin\theta_i = x_i$, $\cos\theta_i = y_i$, with the additional relation $x_i^2 + y_i^2 = 1$. The resulting system of six polynomials in the six variables $x_1, y_1, x_2, y_2, x_3, y_3$, is

$$l_2y_1 + l_3(y_1y_2 - x_1x_2) = {}^4_1t_{13}, \tag{7}$$

$$l_2x_1 + l_3(y_1x_2 + x_1y_2) = {}^4_1t_{23}, \tag{8}$$

$$-x_1(y_2x_3 + x_2y_3) - y_1(x_2x_3 - y_2y_3) = {}^4_1t_{22}, \tag{9}$$

[1] The reader should verify that the (1,1), (1,2), (2,1), (2,2) submatrix in ${}^{i+1}_i\mathrm{T}, i = 1,2,3$, is an element of $SO(2)$. The reader should also verify that the matrix obtained by multiplying any two of these transformation matrices also exhibits the same property.

$$x_1^2 + y_1^2 = 1, \tag{10}$$

$$x_2^2 + y_2^2 = 1, \tag{11}$$

$$x_3^2 + y_3^2 = 1. \tag{12}$$

An alternate substitution is $\sin\theta_i = \frac{2x_i}{1+x_i^2}$, $\cos\theta_i = \frac{1-x_i^2}{1+x_i^2}$, where $x_i = \tan(\frac{\theta_i}{2})$. This would give a system of three rational functions in the three variables x_1, x_2, x_3. After clearing denominators this system could be treated as a set of three polynomials in x_1, x_2, x_3. Systems of nonlinear multivariate polynomials such as these typically have several common zeros. For our inverse kinematics problem the physical significance of several $(\theta_1, \theta_2, \theta_3)$ tuples satisfying (4), (5), and (6) is that the manipulator can place its hand in the position and orientation specified by ${}^4_1T$, in more ways than one. Figure 7 shows two ways (or configurations) in which a planar three-revolute-jointed manipulator places its hand in a specified position and orientation. A knowledge of all solutions to an inverse kinematics problem is generally desirable for motion planning because the presence of obstacles could rule out the use of some of the solutions (see Figure 8.) In summary, the important issues in any manipulator inverse kinematics

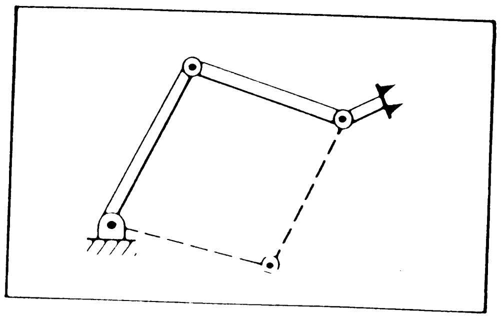
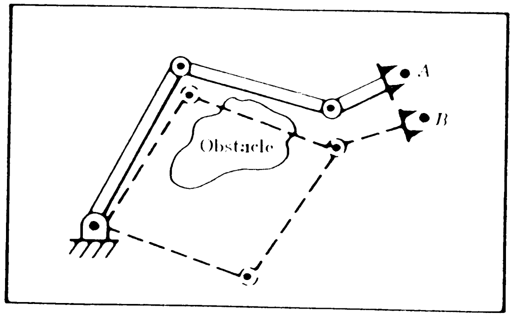


Figure 7: Two Inverse Kinematic Solutions

problem are: (i) how many different configurations of the manipulator permit its hand to occupy a desired position and orientation ? (ii) how does one compute these configurations ? The procedure to be followed in any inverse kinematics problem is the same as the one used for our planar three-revolute-jointed manipulator. The first step is to affix Cartesian coordinate systems on each link of the manipulator. The next step requires writing down the coordinate transformation



One of the two possible solutions to reach point B from point A causes a collision.

Figure 8: Obstacle Avoidance

matrices relating adjacent Cartesian coordinate systems. These matrices may then be used in a matrix equation of the form

$${}^2_1T\,{}^3_2T \cdots {}^{hand}_{n}T = {}^{hand}_{1}T, \tag{13}$$

relating the coordinate sytem on the hand to the coordinate system on the ground. The numerical value of ${}^{hand}_{1}T$, the matrix describing the *desired* position and orientation of the hand, would be specified in an inverse kinematics problem. Numerical values of the invariant parameters of the manipulator such as the lengths of its links, would also be specified. The above matrix equation must then be rewritten as a system of multivariate polynomials to be solved for the joint variables. Algebraic methods for solving systems of multivariate polynomials proceed by eliminating variables and reducing the problem to the solution of a univariate polynomial. The elimination of variables may be effected by Gröbner Bases, resultants and by dyalitical elimination. Elimination by Gröbner Bases and resultants generally requires a large number of computations for even fairly small problems. Dyalitical elimination on the other hand is quite efficient for polynomial systems exhibiting sparsity and structure, though in the worst case it could be as tedious as elimination by resultants. We now present a brief description of the dyalitical elimination procedure.

3. Dyalitical elimination. We describe this elimination procedure with the help of the following system of three non-homogeneous polynomials f_1, f_2, f_3, in the three variables x_1, x_2, x_3:

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & c_1 & d_1 & e_1 & g_1 \\ a_2 & b_2 & c_2 & d_2 & e_2 & g_2 \\ a_3 & b_3 & c_3 & d_3 & e_3 & g_3 \end{pmatrix} \begin{pmatrix} z_1^2 \\ z_1 z_2 \\ z_1 z_3 \\ z_2^2 \\ z_3 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \qquad (14)$$

The coefficients $a_i, \dots, g_i, i = 1, 2, 3$, may be arbitrary complex numbers. These polynomials are to be solved for their common zeros and we do so by eliminating z_2 and z_3 so as to obtain a polynomial in z_1 which vanishes at the common zeros of f_1, f_2, f_3.

We rewrite (14) as a system of three non-homogeneous polynomials in z_2, z_3, over the ring $C[z_1]$ as follows:

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} = \begin{pmatrix} d_1 & b_1 z_1 & e_1 + c_1 z_1 & a_1 z_1^2 + g_1 \\ d_2 & b_2 z_1 & e_2 + c_2 z_1 & a_2 z_1^2 + g_2 \\ d_3 & b_3 z_1 & e_3 + c_3 z_1 & a_3 z_1^2 + g_3 \end{pmatrix} \begin{pmatrix} z_2^2 \\ z_2 \\ z_3 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \qquad (15)$$

Multiply f_1, f_2, f_3, by powers of z_2 and z_3 so as to generate new elements in the ideal of f_1, f_2, f_3, which are linearly independent of f_1, f_2, f_3.[2] Generate as many elements as are required to form a square coefficient array. For example, multiplying f_1, f_2, f_3, by z_2 yields the following three elements in the ideal of f_1, f_2, f_3:

$$\begin{pmatrix} z_2 f_1 \\ z_2 f_2 \\ z_2 f_3 \end{pmatrix} = \begin{pmatrix} d_1 & b_1 z_1 & e_1 + c_1 z_1 & a_1 z_1^2 + g_1 \\ d_2 & b_2 z_1 & e_2 + c_2 z_1 & a_2 z_1^2 + g_2 \\ d_3 & b_3 z_1 & e_3 + c_3 z_1 & a_3 z_1^2 + g_3 \end{pmatrix} \begin{pmatrix} z_2^3 \\ z_2^2 \\ z_2 z_3 \\ z_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \qquad (16)$$

These elements are linearly independent of f_1, f_2, f_3, because clearly they cannot be expressed in the form $af_1 + bf_2 + cf_3$, where a, b, c, are elements of $C[z_1]$. (15) and (16) taken together may be written as:

$$\begin{pmatrix} f_1 \\ f_2 \\ f_3 \\ z_2 f_1 \\ z_2 f_2 \\ z_2 f_3 \end{pmatrix} =$$

[2] By the ideal of f_1, f_2, f_3, we mean the set of all polynomials of the form $q_1 f_1 + q_2 f_2 + q_3 f_3$, where q_1, q_2, q_3, are arbitrary elements of the set of polynomials in z_2, z_3, over the ring $C[z_1]$.

$$\begin{pmatrix} 0 & d_1 & 0 & b_1 z_1 & e_1 + c_1 z_1 & a_1 z_1^2 + g_1 \\ 0 & d_2 & 0 & b_2 z_1 & e_2 + c_2 z_1 & a_2 z_1^2 + g_2 \\ 0 & d_3 & 0 & b_3 z_1 & e_3 + c_3 z_1 & a_3 z_1^2 + g_3 \\ d_1 & b_1 z_1 & e_1 + c_1 z_1 & a_1 z_1^2 + g_1 & 0 & 0 \\ d_2 & b_2 z_1 & e_2 + c_2 z_1 & a_2 z_1^2 + g_2 & 0 & 0 \\ d_3 & b_3 z_1 & e_3 + c_3 z_1 & a_3 z_1^2 + g_3 & 0 & 0 \end{pmatrix} \begin{pmatrix} z_2^3 \\ z_2^2 \\ z_2 z_3 \\ z_2 \\ z_3 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \qquad (17)$$

The coefficient matrix is square and therefore we need not generate any more elements in the ideal of f_1, f_2, f_3. The determinant of the coefficient matrix in (17) must vanish at each root of f_1, f_2, f_3. If it does not vanish, the coefficient matrix is invertible at that value of z_1. Multiplying both sides of (17) by the inverse of the coefficient matrix yields:

$$\begin{pmatrix} z_2^3 \\ z_2^2 \\ z_2 z_3 \\ z_2 \\ z_3 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \qquad (18)$$

The last row of (18) is a contradiction. Therefore the determinant of the coefficient matrix in (17) is the desired univariate polynomial in z_1. After computing the roots of this polynomial we may compute the values of z_2 and z_3 corresponding to each value of z_1 by substituting for z_1 in the coefficient matrix of (17) to obtain

$$A \begin{pmatrix} z_2^3 \\ z_2^2 \\ z_2 z_3 \\ z_2 \\ z_3 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \qquad (19)$$

where A is a 6×6 array of complex numbers. We may pick any five independent equations in (19), treat them as a linear system in five variables, and solve for $z_2^3, z_2^2, z_2 z_3, z_2, z_3$. The drawback of the dyalitical elimination method is that the univariate polynomial in z_1 may be of degree higher than the number of common zeros of f_1, f_2, f_3. This procedure may therefore yield extraneous roots along with the true ones. This may be remedied by substituting each candidate root back into f_1, f_2, f_3 to see whether they vanish. Additional details of this elimination procedure may be found in SALMON[1885], pages 79-90. James Joseph Sylvester introduced the name "dyalitical" because the process dissolves the relations which connect the different combinations of powers of the variables and treats them as simple independent quantities.

4. Properties of the *n*-revolute-jointed manipulator. The position and orientation of a body in three dimensional space ($\Re^3$) is determined by six parameters; three for position and three for orientation. A manipulator capable of positioning and orienting a body (its hand) arbitrarily in three space must therefore possess at least six degrees of freedom. The number of degrees of freedom of a manipulator is the number of independent parameters which would have to be specified in order to locate all parts of the manipulator. In a series manipulator with revolute and prismatic joints each joint contributes one independent parameter viz. the joint rotation angle or the joint displacement, depending on whether the joint is revolute or prismatic. Therefore a six-degrees-of-freedom series manipulator with revolute and prismatic joints must have a total of six joints. The inverse kinematics of all six-degrees-of-freedom series manipulators with at least one prismatic joint has been solved over the past twenty years. However the inverse kinematics of the series manipulator with six revolute joints remained unsolved until 1988 (see LEE and LIANG[1988a and 1988b].) The reason for this is the excessive complexity of the variable elimination process for this manipulator. This problem was appropriately named "the Mount Everest of series manipulator inverse kinematics," by FREUDENSTEIN[1973]. Recently we derived a new solution to this problem. In the present section we develop the tools for our solution. We then show how they may be combined with the dyalitical elimination procedure to solve this problem.

4.1 Notation for spatial manipulators. There is a natural extension of the ideas in Section 2 to spatial or three dimensional manipulators. For such manipulators the Cartesian coordinate system attached to each link is three dimensional. The coordinate transformation matrices ${}_{i}^{i+1}\mathbf{T}$ are of size 4×4. Though the coordinate systems may be located and oriented arbitrarily on each link, the research community has agreed upon a set of rules with a view to standardizing notation. These rules are explained in detail in TSAI and MORGAN[1985] and the essential features are as follows:

1. The links are numbered in ascending order outwards with the fixed link being number 1. Joint i connects links i and $i+1$.
2. Coordinate system i is located on link i with its z axis (z_i) along the axis of joint i. The x_i axis is along the common normal to the axes of joints i and $i-1$. The y_i axis is normal to the x_i and z_i axes.
3. The first coordinate system is fixed to the ground with the x_1 axis directed arbitrarily in the plane normal to the z_1 axis.
4. The coordinate sytem on the hand may be located arbitrarily. The z axis direction may also be selected arbitrarily. The x axis is defined by the common normal to the z axis on the last link and the z axis on the preceding coordinate system. The y axis is normal to the x and z axes.

For such an arrangement the coordinate transformation matrix ${}_{i}^{i+1}\mathbf{T}$ is

$$\begin{pmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

where θ, α, a, and d are shown in Figure 9. A formal derivation of ${}_{i}^{i+1}\mathbf{T}$ may be

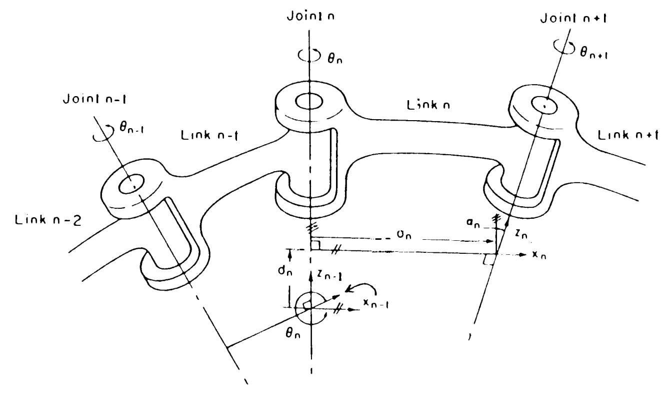


Figure 9: Kinematic Parameters

found in PAUL[1981], Chapter 2, page 53. For a revolute-jointed manipulator, θ_i is the angle of rotation of link $i+1$ relative to link i, about joint i. Unlike θ_i, the parameters α_i, a_i and d_i are constants. The transformation ${}_{1}^{n+1}\mathbf{T}$ for a n-revolute-jointed spatial series manipulator is related to the individual coordinate transformation matrices ${}_{i}^{i+1}\mathbf{T}$, $i = 1, \ldots, n$, by the following equation:

$$ {}_{1}^{2}\mathbf{T}{}_{2}^{3}\mathbf{T}\ldots{}_{n}^{n+1}\mathbf{T} = {}_{1}^{n+1}\mathbf{T}. \tag{20}$$

Let ${}_{j}^{i}\mathbf{T}$, (i, j, integers) be written as $\begin{pmatrix} {}_{j}^{i}\mathbf{R} & {}_{j}^{i}\mathbf{P} \\ 0 & 1 \end{pmatrix}$ where ${}_{j}^{i}\mathbf{R}$ is a 3×3 matrix comprised of the first three rows and columns of ${}_{j}^{i}\mathbf{T}$, and $\begin{pmatrix} {}_{j}^{i}\mathbf{P} \\ 1 \end{pmatrix}$ is column 4 of ${}_{j}^{i}\mathbf{T}$. The reader should verify that the relation ${}_{j}^{i}\mathbf{T} = {}_{j}^{j+1}\mathbf{T}{}_{j+1}^{j+2}\mathbf{T}\ldots{}_{i-1}^{i}\mathbf{T}$, implies that ${}_{j}^{i}\mathbf{R} = {}_{j}^{j+1}\mathbf{R}{}_{j+1}^{j+2}\mathbf{R}\ldots{}_{i-1}^{i}\mathbf{R}$. By definition, ${}_{j}^{j+1}\mathbf{R}$ is an element of $\mathcal{SO}(3)$ [3] for

[3] $\mathcal{SO}(3)$ is the group of orthogonal 3×3 matrices with determinant equal to $+1$.

all j. Let ${}^{i}_{j}\mathbf{R}$ be written as $\begin{pmatrix} {}^{i}_{j}\mathbf{l} & {}^{i}_{j}\mathbf{m} & {}^{i}_{j}\mathbf{n} \end{pmatrix}$ where ${}^{i}_{j}\mathbf{l}$, ${}^{i}_{j}\mathbf{m}$, ${}^{i}_{j}\mathbf{n}$, are respectively columns 1, 2, and 3 of ${}^{i}_{j}\mathbf{R}$. (20) may be rewritten as:

$$ {}^{n}_{1}\mathbf{T}\,{}^{n+1}_{n}\mathbf{T} = {}^{n+1}_{1}\mathbf{T}. \tag{21}$$

Making the following substitutions in (21)

$$ {}^{n}_{1}\mathbf{T} \leftarrow \begin{pmatrix} {}^{n}_{1}\mathbf{R} & {}^{n}_{1}\mathbf{p} \\ 0 & 1 \end{pmatrix},$$

$$ {}^{n+1}_{n}\mathbf{T} \leftarrow \begin{pmatrix} {}^{n+1}_{n}\mathbf{l} & {}^{n+1}_{n}\mathbf{m} & {}^{n+1}_{n}\mathbf{n} & {}^{n+1}_{n}\mathbf{p} \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$ {}^{n+1}_{1}\mathbf{T} \leftarrow \begin{pmatrix} {}^{n+1}_{1}\mathbf{l} & {}^{n+1}_{1}\mathbf{m} & {}^{n+1}_{1}\mathbf{n} & {}^{n+1}_{1}\mathbf{p} \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

we get

$$ \begin{pmatrix} {}^{n}_{1}\mathbf{R} & {}^{n}_{1}\mathbf{p} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} {}^{n+1}_{n}\mathbf{l} & {}^{n+1}_{n}\mathbf{m} & {}^{n+1}_{n}\mathbf{n} & {}^{n+1}_{n}\mathbf{p} \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} {}^{n+1}_{1}\mathbf{l} & {}^{n+1}_{1}\mathbf{m} & {}^{n+1}_{1}\mathbf{n} & {}^{n+1}_{1}\mathbf{p} \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{22}$$

Equating columns 1, 2, 3, and 4 on either side of (22) we get

$$ {}^{n}_{1}\mathbf{R}\,{}^{n+1}_{n}\mathbf{l} = {}^{n+1}_{1}\mathbf{l}, \tag{23}$$

$$ {}^{n}_{1}\mathbf{R}\,{}^{n+1}_{n}\mathbf{m} = {}^{n+1}_{1}\mathbf{m}, \tag{24}$$

$$ {}^{n}_{1}\mathbf{R}\,{}^{n+1}_{n}\mathbf{n} = {}^{n+1}_{1}\mathbf{n}, \tag{25}$$

$$ {}^{n}_{1}\mathbf{R}\,{}^{n+1}_{n}\mathbf{p} + {}^{n}_{1}\mathbf{p} = {}^{n+1}_{1}\mathbf{p}. \tag{26}$$

We now present some results on the structure of (20)-(26). The formal proofs supporting these results are available in Appendix 1.

4.2 Structure theorems.

LEMMA. *The equations obtained by equating entries from the left and right-hand sides of (20) are of degree 1 or less in* $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

This means that for any given integer i, the highest power of $\sin\theta_i, \cos\theta_i$, appearing in these equations is less than or equal to 1.

PROPERTY 1. $({}^{n+1}_{1}\mathbf{p}) \cdot ({}^{n+1}_{1}\mathbf{p})$ *is of degree 1 or less in* $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

PROPERTY 2. *Each of the functions* $({}^{n+1}_{1}\mathbf{p})\cdot({}^{n+1}_{1}\mathbf{l})$, $({}^{n+1}_{1}\mathbf{p})\cdot({}^{n+1}_{1}\mathbf{m})$, $({}^{n+1}_{1}\mathbf{p})\cdot({}^{n+1}_{1}\mathbf{n})$, *is of degree 1 or less in* $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

PROPERTY 3. *The entries of the vectors* ${}^{n+1}_{1}\mathbf{p}\times{}^{n+1}_{1}\mathbf{l}$, ${}^{n+1}_{1}\mathbf{p}\times{}^{n+1}_{1}\mathbf{m}$, ${}^{n+1}_{1}\mathbf{p}\times{}^{n+1}_{1}\mathbf{n}$ *are of degree 1 or less in* $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

PROPERTY 4. *The functions* $({}^{n+1}_{1}\mathbf{p}\cdot{}^{n+1}_{1}\mathbf{p}){}^{n+1}_{1}\mathbf{l} - (2\,{}^{n+1}_{1}\mathbf{p}\cdot{}^{n+1}_{1}\mathbf{l}){}^{n+1}_{1}\mathbf{p}$, $({}^{n+1}_{1}\mathbf{p}\cdot{}^{n+1}_{1}\mathbf{p}){}^{n+1}_{1}\mathbf{m} - (2\,{}^{n+1}_{1}\mathbf{p}\cdot{}^{n+1}_{1}\mathbf{m}){}^{n+1}_{1}\mathbf{p}$, *and* $({}^{n+1}_{1}\mathbf{p}\cdot{}^{n+1}_{1}\mathbf{p}){}^{n+1}_{1}\mathbf{n} - (2\,{}^{n+1}_{1}\mathbf{p}\cdot{}^{n+1}_{1}\mathbf{n}){}^{n+1}_{1}\mathbf{p}$ *are of degree 1 or less in* $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

5. Six-revolute-jointed manipulator inverse kinematics. We now show how the properties presented in the previous section may be used along with the dyalitical elimination procedure to solve the inverse kinematics problem for the spatial six-revolute-jointed manipulator (6R) of general geometry. (20), with n set equal to 6 is to be solved for the six variables $\theta_1, \ldots, \theta_6$. Numerical values for $a_i, \alpha_i, d_i, i = 1, \ldots, 6$, and ${}^{7}_{1}\mathbf{T}$ are provided and are arbitrarily real numbers. Let ${}^{7}_{1}\mathbf{T}$ be equal to $\begin{pmatrix} l_x & m_x & n_x & \rho_x \\ l_y & m_y & n_y & \rho_y \\ l_z & m_z & n_z & \rho_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$. (20) may be rewritten as

$$ {}^{4}_{3}\mathbf{T}\,{}^{5}_{4}\mathbf{T}\,{}^{6}_{5}\mathbf{T} = {}^{3}_{2}\mathbf{T}^{-1}\,{}^{2}_{1}\mathbf{T}^{-1}\,{}^{hand}_{1}\mathbf{T}\,{}^{7}_{6}\mathbf{T}^{-1}. \tag{27}$$

We do this so as to move θ_1, θ_2, and θ_6 to the right-hand side. This lowers the degrees of the equations and also reduces their complexity.

$$\begin{pmatrix} (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) \\ (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) & (\theta_3,\theta_4,\theta_5) \\ (\theta_4,\theta_5) & (\theta_4,\theta_5) & (\theta_4,\theta_5) & (\theta_4,\theta_5) \\ 0 & 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2) & (\theta_1,\theta_2) \\ (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2) & (\theta_1,\theta_2) \\ (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2,\theta_6) & (\theta_1,\theta_2) & (\theta_1,\theta_2) \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{28}$$

(28) shows the variables appearing in the individual entries in (27). The six scalar equations obtained from columns 4 and 3 of (27) are devoid of θ_6. These equations though linearly independent are governed by the constraint that the magnitude of the column 3 vector is unity. We work with these six equations with the goal of eliminating four of the five variables so as to obtain a univariate polynomial which will vanish at their common zeros. The six equations are (note: $\sin\theta_i$ is denoted by s_i, $\cos\theta_i$ by c_i, $\sin\alpha_i$ by μ_i, and $\cos\alpha_i$ by λ_i.):

$$ c_3 f_1 + s_3 f_2 = c_2 h_1 + s_2 h_2 - a_2, \tag{29}$$

$$ s_3 f_1 - c_3 f_2 = -\lambda_2(s_2 h_1 - c_2 h_2) + \mu_2(h_3 - d_2), \tag{30}$$

$$ f_3 = \mu_2(s_2 h_1 - c_2 h_2) + \lambda_2(h_3 - d_2), \tag{31}$$

$$ c_3 r_1 + s_3 r_2 = c_2 b_1 + s_2 b_2, \tag{32}$$

$$s_3 r_1 - c_3 r_2 = -\lambda_2(s_2 b_1 - c_2 b_2) + \mu_2 b_3, \tag{33}$$

$$r_3 = \mu_2(s_2 b_1 - c_2 b_2) + \lambda_2 b_3, \tag{34}$$

where

$$\begin{aligned} f_1 &= c_4 g_1 + s_4 g_2 + a_3, \\ f_2 &= -\lambda_3(s_4 g_1 - c_4 g_2) + \mu_3 g_3, \\ f_3 &= \mu_3(s_4 g_1 - c_4 g_2) + \lambda_3 g_3 + d_3, \end{aligned}$$

$$\begin{aligned} r_1 &= c_4 m_1 + s_4 m_2, \\ r_2 &= -\lambda_3(s_4 m_1 - c_4 m_2) + \mu_3 m_3, \\ r_3 &= \mu_3(s_4 m_1 - c_4 m_2) + \lambda_3 m_3, \end{aligned}$$

$$\begin{aligned} g_1 &= c_5 a_5 + a_4, \\ g_2 &= -s_5 \lambda_4 a_5 + \mu_4 d_5, \\ g_3 &= s_5 \mu_4 a_5 + \lambda_4 d_5 + d_4, \end{aligned}$$

$$\begin{aligned} m_1 &= s_5 \mu_5, \\ m_2 &= c_5 \lambda_4 \mu_5 + \mu_4 \lambda_5, \\ m_3 &= -c_5 \mu_4 \mu_5 + \lambda_4 \lambda_5, \end{aligned}$$

$$\begin{aligned} h_1 &= c_1 p + s_1 q - a_1, \\ h_2 &= -\lambda_1(s_1 p - c_1 q) + \mu_1(r - d_1), \\ h_3 &= \mu_1(s_1 p - c_1 q) + \lambda_1(r - d_1), \end{aligned}$$

$$\begin{aligned} b_1 &= c_1 u + s_1 v, \\ b_2 &= -\lambda_1(s_1 u - c_1 v) + \mu_1 w, \\ b_3 &= \mu_1(s_1 u - c_1 v) + \lambda_1 w, \end{aligned}$$

$$\begin{aligned} p &= -l_x a_6 - (m_x \mu_6 + n_x \lambda_6) d_6 + \rho_x, \\ q &= -l_y a_6 - (m_y \mu_6 + n_y \lambda_6) d_6 + \rho_y, \\ r &= -l_z a_6 - (m_z \mu_6 + n_z \lambda_6) d_6 + \rho_z, \end{aligned}$$

$$\begin{aligned} u &= m_x \mu_6 + n_x \lambda_6, \\ v &= m_y \mu_6 + n_y \lambda_6, \\ w &= m_z \mu_6 + n_z \lambda_6. \end{aligned}$$

(29)-(34) may be written in matrix form as follows:

$$\begin{pmatrix} c_3 & s_3 & 0 \\ s_3 & -c_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -\lambda_2 & \mu_2 \\ 0 & \mu_2 & \lambda_2 \end{pmatrix} \left\{ \begin{pmatrix} c_2 & s_2 & 0 \\ s_2 & -c_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix} + \begin{pmatrix} -a_2 \\ 0 \\ -d_2 \end{pmatrix} \right\}, \tag{35}$$

$$\begin{pmatrix} c_3 & s_3 & 0 \\ s_3 & -c_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -\lambda_2 & \mu_2 \\ 0 & \mu_2 & \lambda_2 \end{pmatrix} \begin{pmatrix} c_2 & s_2 & 0 \\ s_2 & -c_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}. \tag{36}$$

Premultiplying (35) by $\begin{pmatrix} 1 & 0 & 0 \\ 0 & -\lambda_2 & \mu_2 \\ 0 & \mu_2 & \lambda_2 \end{pmatrix}$ and adding $\begin{pmatrix} a_2 \\ 0 \\ d_2 \end{pmatrix}$ we get

$$\begin{pmatrix} c_2 & s_2 & 0 \\ s_2 & -c_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{h}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -\lambda_2 & \mu_2 \\ 0 & \mu_2 & \lambda_2 \end{pmatrix} \begin{pmatrix} c_3 & s_3 & 0 \\ s_3 & -c_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{f}} + \begin{pmatrix} a_2 \\ 0 \\ d_2 \end{pmatrix}, \tag{37}$$

where $\bar{\mathbf{h}} = \begin{pmatrix} h_1 \\ h_2 \\ h_3 \end{pmatrix}$ and $\bar{\mathbf{f}} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}$. Similarly (36) may be rewritten as:

$$\begin{pmatrix} c_2 & s_2 & 0 \\ s_2 & -c_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{b}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -\lambda_2 & \mu_2 \\ 0 & \mu_2 & \lambda_2 \end{pmatrix} \begin{pmatrix} c_3 & s_3 & 0 \\ s_3 & -c_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{r}}, \tag{38}$$

where $\bar{\mathbf{b}} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}$ and $\bar{\mathbf{r}} = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \end{pmatrix}$.

Henceforth we will refer to (37) and (38) by the vectors $\bar{\mathbf{p}}$ and $\bar{\mathbf{n}}$ respectively. By this we mean

$$\bar{\mathbf{p}} = \begin{pmatrix} c_2 & s_2 & 0 \\ s_2 & -c_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{h}} =$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -\lambda_2 & \mu_2 \\ 0 & \mu_2 & \lambda_2 \end{pmatrix} \begin{pmatrix} c_3 & s_3 & 0 \\ s_3 & -c_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{f}} + \begin{pmatrix} a_2 \\ 0 \\ d_2 \end{pmatrix},$$

and

$$\bar{\mathbf{n}} = \begin{pmatrix} c_2 & s_2 & 0 \\ s_2 & -c_2 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{b}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -\lambda_2 & \mu_2 \\ 0 & \mu_2 & \lambda_2 \end{pmatrix} \begin{pmatrix} c_3 & s_3 & 0 \\ s_3 & -c_3 & 0 \\ 0 & 0 & 1 \end{pmatrix} \bar{\mathbf{r}}.$$

We now proceed to eliminate four of the five variables in $\bar{\mathbf{p}}$ and $\bar{\mathbf{n}}$ by exploiting the structure of the ideal generated by the component equations of $\bar{\mathbf{p}}$ and $\bar{\mathbf{n}}$. We know from Properties 1-4 of the previous section that $\bar{\mathbf{p}}\cdot\bar{\mathbf{p}}$, $\bar{\mathbf{p}}\cdot\bar{\mathbf{n}}$, $\bar{\mathbf{p}}\times\bar{\mathbf{n}}$, and $(\bar{\mathbf{p}}\cdot\bar{\mathbf{p}})\bar{\mathbf{n}}-(2\bar{\mathbf{p}}\cdot\bar{\mathbf{n}})\bar{\mathbf{p}}$, all have the same structure as $\bar{\mathbf{p}}$ and $\bar{\mathbf{n}}$ viz. they are all of degree 1 or less in $\sin\theta_i, \cos\theta_i$, $i = 1,\ldots,5$. Consequently we have the following set of fourteen equations all of which have the same set of power products:[4]

Vector	No. of Scalar Equations
$\bar{\mathbf{p}}$	3
$\bar{\mathbf{n}}$	3
$\bar{\mathbf{p}}\cdot\bar{\mathbf{p}}$	1
$\bar{\mathbf{p}}\cdot\bar{\mathbf{n}}$	1
$\bar{\mathbf{p}}\times\bar{\mathbf{n}}$	3
$(\bar{\mathbf{p}}\cdot\bar{\mathbf{p}})\bar{\mathbf{n}}-(2\bar{\mathbf{p}}\cdot\bar{\mathbf{n}})\bar{\mathbf{p}}$	3
	(Total) 14

These fourteen equations are linearly independent. They may be written in matrix form as:

$$(P)\begin{pmatrix} s_4s_5 \\ s_4c_5 \\ c_4s_5 \\ c_4c_5 \\ s_4 \\ c_4 \\ s_5 \\ c_5 \\ 1 \end{pmatrix} = (Q)\begin{pmatrix} s_1s_2 \\ s_1c_2 \\ c_1s_2 \\ c_1c_2 \\ s_1 \\ c_1 \\ s_2 \\ c_2 \end{pmatrix}, \tag{39}$$

where P is a 14×9 matrix, the entries of which are linear combinations of $s_3, c_3, 1$, and Q is a 14×8 matrix, the entries of which are constants. We use any eight of the fourteen equations in (39) to solve for the eight right-hand side terms containing θ_1 and θ_2 in terms of the left-hand side which is a function of θ_3, θ_4 and θ_5. We use these to eliminate terms containing θ_1 and θ_2 from the remaining six equations, which then take the form:

$$(\Sigma)\begin{pmatrix} s_4s_5 \\ s_4c_5 \\ c_4s_5 \\ c_4c_5 \\ s_4 \\ c_4 \\ s_5 \\ c_5 \\ 1 \end{pmatrix} = 0, \tag{40}$$

[4] By power products we mean "terms," e.g. the power products of the polynomial $5x^2y + 3xz + 9y^2 + 4z = 0$ are x^2y, xz, y^2 and z.

where Σ is a 6×9 matrix, the entries of which are linear combinations of $s_3, c_3, 1$. We make the following substitutions in (40):

$$s_4 \leftarrow \frac{2x_4}{1+x_4^2},\ c_4 \leftarrow \frac{1-x_4^2}{1+x_4^2},\ s_5 \leftarrow \frac{2x_5}{1+x_5^2},\ c_5 \leftarrow \frac{1-x_5^2}{1+x_5^2},$$

where $x_4 = tan(\frac{\theta_4}{2})$, $x_5 = tan(\frac{\theta_5}{2})$.

We then multiply each equation by $(1+x_4^2)$ and $(1+x_5^2)$ to clear denominators. (40) then takes the form

$$(\Sigma')\begin{pmatrix} x_4^2x_5^2 \\ x_4^2x_5 \\ x_4^2 \\ x_4x_5^2 \\ x_4x_5 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{pmatrix} = 0, \tag{41}$$

where Σ' is a 6×9 matrix, the entries of which are linear combinations of $s_3, c_3, 1$. We make the following substitutions in (41):

$$s_3 \leftarrow \frac{2x_3}{1+x_3^2},\ c_3 \leftarrow \frac{1-x_3^2}{1+x_3^2}.$$

We multiply the first four scalar equations in (41) by $(1+x_3^2)$ to clear denominators. The resulting equation is of the form

$$(\Sigma'')\begin{pmatrix} x_4^2x_5^2 \\ x_4^2x_5 \\ x_4^2 \\ x_4x_5^2 \\ x_4x_5 \\ x_4 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{pmatrix} = 0, \tag{42}$$

where Σ'' is a 6×9 matrix. The entries in the first four rows of Σ'' are quadratic polynomials in x_3. The entries in the last two rows are rational functions of x_3, the numerators being quadratic polynomials in x_3, the denominators being $(1+x_3^2)$. It is noteworthy that the determinant of the 6×6 array comprised of any set of six columns of Σ'' is always an 8^{th} degree polynomial *and not a rational function*. This fact is proved rigorously in Appendix 2.

We now eliminate x_4 and x_5 dyalitically as follows : multiplying (42) by x_4, we get the following equation:

$$(\Sigma'')\begin{pmatrix} x_4^3x_5^2 \\ x_4^3x_5 \\ x_4^3 \\ x_4^2x_5^2 \\ x_4^2x_5 \\ x_4^2 \\ x_4x_5^2 \\ x_4x_5 \\ x_4 \end{pmatrix} = 0. \tag{43}$$

(42) and (43) taken together may be written in matrix form as:

$$\begin{pmatrix} \Sigma'' & 0 \\ 0 & \Sigma'' \end{pmatrix}\begin{pmatrix} x_4^3x_5^2 \\ x_4^3x_5 \\ x_4^3 \\ x_4^2x_5^2 \\ x_4^2x_5 \\ x_4^2 \\ x_4x_5^2 \\ x_4x_5 \\ x_4 \\ x_5^2 \\ x_5 \\ 1 \end{pmatrix} = 0. \tag{44}$$

The determinant of the coefficient matrix is a 16^{th} degree polynomial in x_3. The roots of this polynomial give the values of x_3 corresponding to the sixteen solutions of the inverse kinematics problem. For each value of x_3 thus obtained, the corresponding value of θ_3 may be computed using the formula $\theta_3 = 2tan^{-1}x_3$. It so happens that a 6R manipulator can place its hand in a specified position and orientation in sixteen different ways. Therefore the above procedure gives no extraneous roots.

For each value of θ_3, we may compute the remaining joint variables as follows : we substitute the numerical value of θ_3 in the coefficient matrix of (44). We then use eleven independent members of (44) to solve for the eleven terms $x_4^3x_5^2$, $x_4^3x_5$, x_4^3, $x_4^2x_5^2$, $x_4^2x_5$, x_4^2, $x_4x_5^2$, x_4x_5, x_4, x_5^2, x_5. The numerical values of x_4 and x_5 may be used to compute θ_4 and θ_5. We then substitute numerical values for θ_3, θ_4, and θ_5 in (39). We use eight linearly independent members of the resulting equation to solve for s_1s_2, s_1c_2, c_1s_2, c_1c_2, s_1, c_1, s_2 and c_2. We then use the numerical values of s_1 and c_1 to obtain a unique value for θ_1. Similarly, θ_2 may be computed using the numerical values of s_2 and c_2. Finally, substituting values for θ_1, θ_2, θ_3, θ_4, and θ_5 in the (1,1) and (2,1) elements of the following equation

$${}^7_6T = {}^6_5T^{-1}\,{}^5_4T^{-1}\,{}^4_3T^{-1}\,{}^3_2T^{-1}\,{}^2_1T^{-1}\,{}^{hand}_1T, \tag{45}$$

yields two linear equations in s_6 and c_6. After solving for s_6 and c_6 we may use their values to determine a unique value for θ_6.

6. Brief history of manipulator kinematics. The use of coordinate systems and transformation matrices to model linkages was introduced by DENAVIT and HARTENBERG[1955]. The first systematic study of the inverse kinematics of spatial series manipulators with revolute and prismatic joints was performed by PIEPER[1968]. He presented a classification of manipulators based on the degree of difficulty of their inverse kinematics. He worked out closed-form solutions for various manipulators. Pieper recognized that an algebraic solution for the 6R would be quite difficult. Therefore he suggested a numerical solution. In the years following the appearance of Pieper's thesis, algebraic solutions to the inverse kinematics problem for most series manipulators with six or fewer degrees of freedom were presented. However the 6R remained unsolved. ROTH, RASTEGAR and SCHEINMAN[1973] showed that in general this problem has at most thirty-two solutions, i.e. there are at most thirty-two distinct configurations of the manipulator, in which its hand is at the desired goal position. Their proof was based on arguments from synthetic geometry and was non-constructive. ALBALA and ANGELES[1979], using indicial notation, presented the first constructive solution to this problem. Their solution was in the form of a 12×12 determinant the entries of which were quartic polynomials in the tangent of the half-angle of one of the joint variables. DUFFY and CRANE[1980] later provided a different solution by combining spherical trigonometry and dyalitical elimination to obtain a 32^{nd} degree polynomial in the tangent of the half-angle of one of the joint variables. Their 32^{nd} degree polynomial always yielded extraneous or false roots in addition to the desired values. TSAI and MORGAN[1985] used homotopy continuation to solve the inverse kinematics problem and found only sixteen solutions for various 6R manipulators of different geometries. They therefore conjectured that this problem has at most sixteen solutions. Recently, LEE and LIANG[1988a and 1988b], used extensions of Duffy's method to obtain by dyalitical elimination a 16^{th} degree polynomial in the tangent of the half-angle of one of the joint variables, thus confirming the conjecture of TSAI and MORGAN[1985] and also presenting the first correct and complete solution to this problem.

A1. Appendix 1.

A1.1 Lemma. *The equations obtained by equating entries from the left and right-hand sides of (20) are of degree 1 or less in* $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

By this we mean that for any given integer i, the highest power of $\sin\theta_i, \cos\theta_i$ appearing in these equations is less than or equal to 1.

A1.2 Proof. The entries of ${}^{i+1}_{i}T$ are of degree 1 or less in $\sin\theta_i, \cos\theta_i$. The lemma follows naturally from this fact.

Remark. By A1.1 (23), (24), (25), and (26) are of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

A1.3 **Lemma**. *The entries of* ${}^{n+1}_{1}\mathbf{T}^{-1}$ *are of degree 1 or less in* $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, n$.

A1.4 **Proof**. By definition

$$ {}^{n+1}_{1}\mathbf{T} = {}^{2}_{1}\mathbf{T}\,{}^{3}_{2}\mathbf{T}\cdots{}^{n}_{n-1}\mathbf{T}\,{}^{n+1}_{n}\mathbf{T}. \tag{46} $$

Therefore

$$ {}^{n+1}_{1}\mathbf{T}^{-1} = {}^{n+1}_{n}\mathbf{T}^{-1}\,{}^{n}_{n-1}\mathbf{T}^{-1}\,{}^{n-1}_{n-2}\mathbf{T}^{-1}\cdots{}^{2}_{1}\mathbf{T}^{-1}. \tag{47} $$

The reader may verify that for any i

$$ {}^{i+1}_{i}\mathbf{T}^{-1} = \begin{pmatrix} \cos\theta_i & \sin\theta_i & 0 & -a_i \\ -\sin\theta_i\cos\alpha_i & \cos\theta_i\cos\alpha_i & \sin\alpha_i & -d_i\sin\alpha_i \\ \sin\theta_i\sin\alpha_i & -\cos\theta_i\sin\alpha_i & \cos\alpha_i & -d_i\cos\alpha_i \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{48} $$

Since each entry of ${}^{i+1}_{i}\mathbf{T}^{-1}$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i$, each matrix on the right-hand side of (47) satisfies this property and the lemma follows.

A1.5 **Lemma**. *Each entry of the vector* ${}^{n}_{1}\mathbf{R}^{T}\,{}^{n}_{1}\mathbf{p}$ *is of degree 1 or less in* $\sin\theta_i$, $\cos\theta_i$, $i = 1, 2, \ldots, n-1$.

A1.6 **Proof**. Using the notation presented in Section 4 we may write ${}^{n}_{1}\mathbf{T}$ as $\begin{pmatrix} {}^{n}_{1}\mathbf{R} & {}^{n}_{1}\mathbf{p} \\ 0 & 1 \end{pmatrix}$. In this notation,

$$ {}^{n}_{1}\mathbf{T}^{-1} = \begin{pmatrix} {}^{n}_{1}\mathbf{R}^{T} & -{}^{n}_{1}\mathbf{R}^{T}\,{}^{n}_{1}\mathbf{p} \\ 0 & 1 \end{pmatrix}. \tag{49} $$

(49) may be verified very easily by seeing that ${}^{n}_{1}\mathbf{T}^{-1}\,{}^{n}_{1}\mathbf{T}$ is equal to I. The lemma follows from (49) and A1.3.

A1.7 **Lemma**. *Let* $\mathbf{v}_1, \mathbf{v}_2$ *be arbitrary vectors. Then* $({}^{i}_{j}\mathbf{R}\mathbf{v}_1) \times ({}^{i}_{j}\mathbf{R}\mathbf{v}_2) = {}^{i}_{j}\mathbf{R}(\mathbf{v}_1 \times \mathbf{v}_2)$, $\forall\, i, j$.

A1.8 **Proof**. Let $\mathbf{v}_1 = \begin{pmatrix} v_{11} & v_{12} & v_{13} \end{pmatrix}^T$, $\mathbf{v}_2 = \begin{pmatrix} v_{21} & v_{22} & v_{23} \end{pmatrix}^T$.

$$ {}^{i}_{j}\mathbf{R}\mathbf{v}_1 = \begin{pmatrix} {}^{i}_{j}\mathbf{l} & {}^{i}_{j}\mathbf{m} & {}^{i}_{j}\mathbf{n} \end{pmatrix}\mathbf{v}_1 = {}^{i}_{j}\mathbf{l}v_{11} + {}^{i}_{j}\mathbf{m}v_{12} + {}^{i}_{j}\mathbf{n}v_{13}. \tag{50} $$

Similarly

$$ {}^{i}_{j}\mathbf{R}\mathbf{v}_2 = {}^{i}_{j}\mathbf{l}v_{21} + {}^{i}_{j}\mathbf{m}v_{22} + {}^{i}_{j}\mathbf{n}v_{23}. \tag{51} $$

From (50) and (51), it follows that

$$ \begin{aligned} ({}^{i}_{j}\mathbf{R}\mathbf{v}_1) \times ({}^{i}_{j}\mathbf{R}\mathbf{v}_2) &= ({}^{i}_{j}\mathbf{l} \times {}^{i}_{j}\mathbf{l})v_{11}v_{21} + ({}^{i}_{j}\mathbf{l} \times {}^{i}_{j}\mathbf{m})v_{11}v_{22} + ({}^{i}_{j}\mathbf{l} \times {}^{i}_{j}\mathbf{n})v_{11}v_{23} \\ +({}^{i}_{j}\mathbf{m} \times {}^{i}_{j}\mathbf{l})v_{12}v_{21} &+ ({}^{i}_{j}\mathbf{m} \times {}^{i}_{j}\mathbf{m})v_{12}v_{22} + ({}^{i}_{j}\mathbf{m} \times {}^{i}_{j}\mathbf{n})v_{12}v_{23} \\ +({}^{i}_{j}\mathbf{n} \times {}^{i}_{j}\mathbf{l})v_{13}v_{21} &+ ({}^{i}_{j}\mathbf{n} \times {}^{i}_{j}\mathbf{m})v_{13}v_{22} + ({}^{i}_{j}\mathbf{n} \times {}^{i}_{j}\mathbf{n})v_{13}v_{23}. \end{aligned} \tag{52} $$

Since ${}^{i}_{j}\mathbf{R}$ is orthogonal, ${}^{i}_{j}\mathbf{l} \times {}^{i}_{j}\mathbf{m} = {}^{i}_{j}\mathbf{n}$, ${}^{i}_{j}\mathbf{m} \times {}^{i}_{j}\mathbf{n} = {}^{i}_{j}\mathbf{l}$, ${}^{i}_{j}\mathbf{n} \times {}^{i}_{j}\mathbf{l} = {}^{i}_{j}\mathbf{m}$. Using these relations (52) simplifies to

$$ \begin{aligned} ({}^{i}_{j}\mathbf{R}\mathbf{v}_1) \times ({}^{i}_{j}\mathbf{R}\mathbf{v}_2) &= {}^{i}_{j}\mathbf{l}(v_{12}v_{23} - v_{13}v_{22}) \\ &\quad + {}^{i}_{j}\mathbf{m}(v_{13}v_{21} - v_{11}v_{23}) \\ &\quad + {}^{i}_{j}\mathbf{n}(v_{11}v_{22} - v_{12}v_{21}), \\ &= {}^{i}_{j}\mathbf{R}(\mathbf{v}_1 \times \mathbf{v}_2). \end{aligned} \tag{53} $$

A1.9 **Property** 1. ${}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{p}$ *is of degree 1 or less in* $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, n$.

A1.10 **Proof**. For $n = 1$, the term ${}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{p} = {}^{2}_{1}\mathbf{p} \cdot {}^{2}_{1}\mathbf{p} = a_1^2 + d_1^2$, a constant. Therefore Property 1 is true for $n = 1$. Assume it is true for n equal to some integer $p - 1$, greater than 1. Then for n equal to p, the function ${}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{p}$ may be written as follows (using (26)):

$$ \begin{aligned} {}^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{p} &= \{{}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{p} + {}^{p}_{1}\mathbf{p}\} \cdot \{{}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{p} + {}^{p}_{1}\mathbf{p}\}, \\ &= {}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p} + ({}^{p+1}_{p}\mathbf{p}) \cdot ({}^{p+1}_{p}\mathbf{p}) + 2\,{}^{p+1}_{p}\mathbf{p}^{T}({}^{p}_{1}\mathbf{R}^{T}\,{}^{p}_{1}\mathbf{p}). \end{aligned} \tag{54} $$

The term ${}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{p}$ is equal to $a_p^2 + d_p^2$. By A1.5 the entries of ${}^{p}_{1}\mathbf{R}^{T}\,{}^{p}_{1}\mathbf{p}$ are of degree 1 or less in $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, p-1$. The vector ${}^{p+1}_{p}\mathbf{p}$ is by definition of degree 1 or less in $\sin\theta_p, \cos\theta_p$. Therefore the term $2\,{}^{p+1}_{1}\mathbf{p}^{T}({}^{p}_{1}\mathbf{R}^{T}\,{}^{p}_{1}\mathbf{p})$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, p$. Consequently (54) may be written as

$$ {}^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{p} = {}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p} + \textit{terms of degree 1 or less in } \sin\theta_i, \cos\theta_i,\ i = 1, 2, \ldots, p. \tag{55} $$

By induction on n, Property 1 is true for $n \geq 1$.

A1.11 **Property** 2. *Each of the functions* $({}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{l})$, $({}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{m})$, $({}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{n})$, *is of degree 1 or less in* $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, n$.

A1.12 **Proof**. Substituting from (23) and (26)

$$ \begin{aligned} {}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{l} &= ({}^{n}_{1}\mathbf{R}^{n+1}_{n}\mathbf{p} + {}^{n}_{1}\mathbf{p}) \cdot ({}^{n}_{1}\mathbf{R}^{n+1}_{n}\mathbf{l}), \\ &= ({}^{n+1}_{n}\mathbf{p}) \cdot ({}^{n+1}_{n}\mathbf{l}) + {}^{n+1}_{n}\mathbf{l}^{T}({}^{n}_{1}\mathbf{R}^{T}\,{}^{n}_{1}\mathbf{p}). \end{aligned} \tag{56} $$

The term ${}^{n+1}_{n}\mathbf{p} \cdot {}^{n+1}_{n}\mathbf{l}$ is equal to a_n, a constant. ${}^{n+1}_{n}\mathbf{l}$ is by definition of degree 1 or less in $\sin\theta_n$, $\cos\theta_n$. By A1.5, ${}^{n}_{1}\mathbf{R}^{T}\,{}^{n}_{1}\mathbf{p}$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, n-1$. Therefore the term ${}^{n+1}_{n}\mathbf{l}^{T}({}^{n}_{1}\mathbf{R}^{T}\,{}^{n}_{1}\mathbf{p})$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, n$. Consequently Property 2 is true for the function ${}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{l}$. Proofs for the functions ${}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{m}$ and ${}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{n}$ are along the same lines.

A1.13 **Property** 3. *The entries of the vectors* ${}^{n+1}_{1}\mathbf{p} \times {}^{n+1}_{1}\mathbf{l}$, ${}^{n+1}_{1}\mathbf{p} \times {}^{n+1}_{1}\mathbf{m}$, ${}^{n+1}_{1}\mathbf{p} \times {}^{n+1}_{1}\mathbf{n}$ *are of degree 1 or less in* $\sin\theta_i, \cos\theta_i$, $i = 1, 2, \ldots, n$.

A1.14 Proof. For $n = 1$, the above vectors are:

$$ {}^{2}_{1}\mathbf{p} \times {}^{2}_{1}\mathbf{l} = \begin{pmatrix} -d_1 \sin\theta_1 \\ d_1 \cos\theta_1 \\ 0 \end{pmatrix}, \tag{57}$$

$$ {}^{2}_{1}\mathbf{p} \times {}^{2}_{1}\mathbf{m} = \begin{pmatrix} a_1 \sin\theta_1 \sin\alpha_1 - d_1 \cos\theta_1 \cos\alpha_1 \\ -d_1 \sin\theta_1 \cos\alpha_1 - a_1 \cos\theta_1 \sin\alpha_1 \\ a_1 \cos\alpha_1 \end{pmatrix}, \tag{58}$$

$$ {}^{2}_{1}\mathbf{p} \times {}^{2}_{1}\mathbf{n} = \begin{pmatrix} a_1 \sin\theta_1 \cos\alpha_1 + d_1 \cos\theta_1 \sin\alpha_1 \\ d_1 \sin\theta_1 \sin\alpha_1 - a_1 \cos\theta_1 \cos\alpha_1 \\ -a_1 \sin\alpha_1 \end{pmatrix}. \tag{59}$$

Clearly Property 3 is true when n is equal to 1. Assume that it is true when n is equal to some integer $p-1$, greater than 1. For n equal to p, the vector ${}^{p+1}_{1}\mathbf{p} \times {}^{p+1}_{1}\mathbf{l}$ may be written as follows (using (23) and (26)):

$$\begin{aligned} {}^{p+1}_{1}\mathbf{p} \times {}^{p+1}_{1}\mathbf{l} &= ({}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{p} + {}^{p}_{1}\mathbf{p}) \times ({}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{l}), \\ &= {}^{p}_{1}\mathbf{R}({}^{p+1}_{p}\mathbf{p} \times {}^{p+1}_{p}\mathbf{l}) \\ &+ \left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right) {}^{p+1}_{p}\mathbf{l}. \end{aligned} \tag{60}$$

Similarly

$$ {}^{p+1}_{1}\mathbf{p} \times {}^{p+1}_{1}\mathbf{m} = {}^{p}_{1}\mathbf{R}({}^{p+1}_{p}\mathbf{p} \times {}^{p+1}_{p}\mathbf{m}) + \left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right) {}^{p+1}_{p}\mathbf{m}. \tag{61}$$

$$ {}^{p+1}_{1}\mathbf{p} \times {}^{p+1}_{1}\mathbf{n} = {}^{p}_{1}\mathbf{R}({}^{p+1}_{p}\mathbf{p} \times {}^{p+1}_{p}\mathbf{n}) + \left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right) {}^{p+1}_{p}\mathbf{n}. \tag{62}$$

Each entry of the matrix $\left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right)$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p-1$, because by assumption, Property 3 is true for n equal to $p-1$. Therefore the vectors $\left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right) {}^{p+1}_{p}\mathbf{l}$, $\left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right) {}^{p+1}_{p}\mathbf{m}$, and $\left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right) {}^{p+1}_{p}\mathbf{n}$, appearing in (60), (61), and (62) are of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p$. The reader may verify that the entries of the vectors ${}^{p+1}_{p}\mathbf{p} \times {}^{p+1}_{p}\mathbf{l}$, ${}^{p+1}_{p}\mathbf{p} \times {}^{p+1}_{p}\mathbf{m}$, and ${}^{p+1}_{p}\mathbf{p} \times {}^{p+1}_{p}\mathbf{n}$, are of degree 1 or less in $\sin\theta_p, \cos\theta_p$. Therefore it follows that the vectors on the right-hand side of (60), (61), and (62) are of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p$. By induction on n, Property 3 is true for $n \geq 1$. ∎

A1.15 Property 4. *The functions* $({}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{p})^{n+1}_{1}\mathbf{l} - (2^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{l})^{n+1}_{1}\mathbf{p}$, $({}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{p})^{n+1}_{1}\mathbf{m} - (2^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{m})^{n+1}_{1}\mathbf{p}$, *and* $({}^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{p})^{n+1}_{1}\mathbf{n} - (2^{n+1}_{1}\mathbf{p} \cdot {}^{n+1}_{1}\mathbf{n})^{n+1}_{1}\mathbf{p}$ *are of degree 1 or less in* $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, n$.

A1.16 Proof. Setting n equal to 1 in the above functions, we get

$$({}^{2}_{1}\mathbf{p} \cdot {}^{2}_{1}\mathbf{p})^{2}_{1}\mathbf{l} - (2^{2}_{1}\mathbf{p} \cdot {}^{2}_{1}\mathbf{l})^{2}_{1}\mathbf{p} = (a_1^2 + d_1^2)^{2}_{1}\mathbf{l} - 2a_1({}^{2}_{1}\mathbf{p}), \tag{63}$$

$$({}^{2}_{1}\mathbf{p} \cdot {}^{2}_{1}\mathbf{p})^{2}_{1}\mathbf{m} - (2^{2}_{1}\mathbf{p} \cdot {}^{2}_{1}\mathbf{m})^{2}_{1}\mathbf{p} = (a_1^2 + d_1^2)^{2}_{1}\mathbf{m} - 2d_1 \sin\alpha_1({}^{2}_{1}\mathbf{p}), \tag{64}$$

$$({}^{2}_{1}\mathbf{p} \cdot {}^{2}_{1}\mathbf{p})^{2}_{1}\mathbf{n} - (2^{2}_{1}\mathbf{p} \cdot {}^{2}_{1}\mathbf{n})^{2}_{1}\mathbf{p} = (a_1^2 + d_1^2)^{2}_{1}\mathbf{n} - 2d_1 \cos\alpha_1({}^{2}_{1}\mathbf{p}). \tag{65}$$

Clearly Property 4 is true when n is equal to 1. Let us assume that it is true when n is equal to some integer $p-1$, greater than 1.

For n equal to p, we get (using (23), (26), (54), and (56)):

$$\begin{aligned} &({}^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{p})^{p+1}_{1}\mathbf{l} - (2^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{l})^{p+1}_{1}\mathbf{p} = \\ &\{{}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{p} + {}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p} + 2({}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{p}) \cdot {}^{p}_{1}\mathbf{p}\}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{l} \\ &-2\{{}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{l} + {}^{p}_{1}\mathbf{p} \cdot ({}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{l})\}\{{}^{p}_{1}\mathbf{R}^{p+1}_{p}\mathbf{p} + {}^{p}_{1}\mathbf{p}\}. \end{aligned} \tag{66}$$

After some expansion and simplification, (66) may be written as

$$\begin{aligned} &({}^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{p})^{p+1}_{1}\mathbf{l} - (2^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{l})^{p+1}_{1}\mathbf{p} = \\ &{}^{p}_{1}\mathbf{R}\{({}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{p})^{p+1}_{p}\mathbf{l} - (2^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{l})^{p+1}_{p}\mathbf{p}\} \\ &+ \begin{pmatrix} \{({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{l} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{l})^{p}_{1}\mathbf{p}\}^T \\ \{({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{m} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{m})^{p}_{1}\mathbf{p}\}^T \\ \{({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{n} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{n})^{p}_{1}\mathbf{p}\}^T \end{pmatrix}^T {}^{p+1}_{p}\mathbf{l} \\ &+2^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{R}({}^{p+1}_{p}\mathbf{l} \times {}^{p+1}_{p}\mathbf{p}) - 2({}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{l})^{p}_{1}\mathbf{p}. \end{aligned} \tag{67}$$

The term $2^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{R}({}^{p+1}_{p}\mathbf{l} \times {}^{p+1}_{p}\mathbf{p})$ is equal to $2\left({}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{l} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{m} \quad {}^{p}_{1}\mathbf{p} \times {}^{p}_{1}\mathbf{n} \right)({}^{p+1}_{p}\mathbf{l} \times {}^{p+1}_{p}\mathbf{p})$ and is clearly of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p$, by A1.13 (Property 3.) The term $-2^{p}_{1}\mathbf{p}({}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{l})$ is also of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p$. The term $({}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{p})^{p+1}_{p}\mathbf{l} - (2^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{l})^{p+1}_{p}\mathbf{p}$ is equal to $(a_p^2 + d_p^2)^{p+1}_{p}\mathbf{l} - (2a_p)^{p+1}_{p}\mathbf{p}$, which is clearly of degree 1 or less in $\sin\theta_p, \cos\theta_p$. Therefore the term ${}^{p}_{1}\mathbf{R}(({}^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{p})^{p+1}_{p}\mathbf{l} - (2^{p+1}_{p}\mathbf{p} \cdot {}^{p+1}_{p}\mathbf{l})^{p+1}_{p}\mathbf{p})$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p$. By our assumption that Property 4 is true for n equal to $p-1$, the entries of the matrix $\begin{pmatrix} \{({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{l} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{l})^{p}_{1}\mathbf{p}\}^T \\ \{({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{m} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{m})^{p}_{1}\mathbf{p}\}^T \\ \{({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{n} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{n})^{p}_{1}\mathbf{p}\}^T \end{pmatrix}^T$ are of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p-1$. The vector ${}^{p+1}_{p}\mathbf{l}$ is by definition of degree 1 or less in $\sin\theta_p, \cos\theta_p$. Therefore the term $\begin{pmatrix} ({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{l} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{l})^{p}_{1}\mathbf{p}^T \\ ({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{m} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{m})^{p}_{1}\mathbf{p}^T \\ ({}^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{p})^{p}_{1}\mathbf{n} - (2^{p}_{1}\mathbf{p} \cdot {}^{p}_{1}\mathbf{n})^{p}_{1}\mathbf{p}^T \end{pmatrix}^T {}^{p+1}_{p}\mathbf{l}$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p$. As a result, $({}^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{p})^{p+1}_{1}\mathbf{l} - (2^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{l})^{p+1}_{1}\mathbf{p}$ is of degree 1 or less in $\sin\theta_i, \cos\theta_i, i = 1, 2, \ldots, p$. By similar arguments we may show that the other two functions $({}^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{p})^{p+1}_{1}\mathbf{m} - (2^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{m})^{p+1}_{1}\mathbf{p}$ and $({}^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{p})^{p+1}_{1}\mathbf{n} - (2^{p+1}_{1}\mathbf{p} \cdot {}^{p+1}_{1}\mathbf{n})^{p+1}_{1}\mathbf{p}$ also exhibit this property. By induction on n, Property 4 is true for $n \geq 1$.

A2. Appendix 2.

A2.1 Claim. *The determinant of every* 6×6 *submatrix of* Σ'' *in (42) is an* 8^{th} *degree polynomial in* x_3 *and not a rational function.*

A2.2 **Proof.** The entries in the first four rows of the matrix Σ'' of (42) are quadratic polynomials in x_3 and the entries in the last two rows are rational functions of the form $\frac{\text{quadratic polynomial in } x_3}{(1+x_3^2)}$. Therefore, the determinant of any 6×6 submatrix of Σ'' is of the form $\frac{12^{th} \text{ degree polynomial in } x_3}{(1+x_3^2)^2}$. We claim that the 12^{th} degree polynomial in the numerator contains $(1+x_3^2)^2$ as a factor. If this is true then, the determinant is indeed an 8^{th} degree polynomial. We proceed therefore to establish the claim that the above-mentioned 12^{th} degree numerator polynomial does contain $(1+x_3^2)^2$ as a factor.

Let us examine the left-hand side of (39). The following table provides a summary of the basis elements of the left-hand sides of the fourteen scalar equations in (39).

Equation	Basis Elements
$\bar{p}$	$c_3f_1 + s_3f_2,\ s_3f_1 - c_3f_2,\ f_3,\ 1$
$\bar{n}$	$c_3r_1 + s_3r_2,\ s_3r_1 - c_3r_2,\ r_3$
$\bar{p}\cdot\bar{p}$	$\bar{f}\cdot\bar{f},\ c_3f_1 + s_3f_2,\ s_3f_1 - c_3f_2,\ f_3,\ 1$
$\bar{p}\cdot\bar{n}$	$\bar{f}\cdot\bar{r},\ c_3r_1 + s_3r_2,\ s_3r_1 - c_3r_2,\ r_3$
$\bar{p}\times\bar{n}$	$s_3(f_3r_1 - f_1r_3) + c_3(f_2r_3 - f_3r_2)$, $c_3(f_3r_1 - f_1r_3) - s_3(f_2r_3 - f_3r_2)$, $f_1r_2 - f_2r_1,\ s_3r_1 - c_3r_2,\ c_3r_1 + s_3r_2,\ r_3$
$(\bar{p}\cdot\bar{p})\bar{n} - (2\bar{p}\cdot\bar{n})\bar{p}$	$(ffr_1)c_3 + (ffr_2)s_3,\ (ffr_1)s_3 - (ffr_2)c_3,\ (ffr_3)$, $s_3(f_3r_1 - f_1r_3) + c_3(f_2r_3 - f_3r_2)$, $c_3(f_3r_1 - f_1r_3) - s_3(f_2r_3 - f_3r_2)$, $f_1r_2 - f_2r_1,\ s_3r_1 - c_3r_2,\ c_3r_1 + s_3r_2,\ r_3$, $\bar{f}\cdot\bar{r},\ 1$

In the above table the symbol (ffr_i) has been used as an abbreviation for the term $((\bar{f}\cdot\bar{f})\bar{r} - (2\bar{f}\cdot\bar{r})\bar{f})_i$. The first row of the table tells us that the left-hand side of each component equation of $\bar{p}$ is a linear combination of the terms $c_3f_1 + s_3f_2,\ s_3f_1 - c_3f_2,\ f_3,\ 1$. The remaining rows of the table provide similar information about $\bar{n}$, $\bar{p}\cdot\bar{p}$, $\bar{p}\cdot\bar{n}$, $\bar{p}\times\bar{n}$, $(\bar{p}\cdot\bar{p})\bar{n} - (2\bar{p}\cdot\bar{n})\bar{p}$. Since (40) is a linear combination of the scalar equations in (39), the left-hand side of (40) contains all the terms appearing in the right-hand column of the above table, viz. $c_3f_1 + s_3f_2$, $s_3f_1 - c_3f_2$, f_3, $c_3r_1 + s_3r_2$, $s_3r_1 - c_3r_2$, r_3, $\bar{f}\cdot\bar{f}$, $\bar{f}\cdot\bar{r}$, $s_3(f_3r_1 - f_1r_3) + c_3(f_2r_3 - f_3r_2)$, $c_3(f_3r_1 - f_1r_3) - s_3(f_2r_3 - f_3r_2)$, $f_1r_2 - f_2r_1$, $(ffr_1)c_3 + (ffr_2)s_3$, $(ffr_1)s_3 - (ffr_2)c_3$, (ffr_3), 1. We make the substitution, $s_3 = \frac{2x_3}{(1+x_3^2)}$, $c_3 = \frac{(1-x_3^2)}{(1+x_3^2)}$, (where $x_3 = tan(\frac{\theta_3}{2})$) in (40) and multiply each scalar equation by $(1+x_3^2)$ to clear denominators. We then make the substitution $x_3 = i$. Each scalar equation in (40) then becomes a linear combination of the four terms: $f_1 + if_2$, $r_1 + ir_2$, $(f_3r_1 - f_1r_3) - i(f_2r_3 - f_3r_2)$, $(ffr_1) + i(ffr_2)$. Since there is a total of six equations, clearly two of them must be linearly dependent on the remaining four. We may write these equations in matrix form as:

$$(\bar{\Sigma})\begin{pmatrix} s_4s_5 \\ s_4c_5 \\ c_4s_5 \\ c_4c_5 \\ s_4 \\ c_4 \\ s_5 \\ c_5 \\ 1 \end{pmatrix} = 0, \tag{68}$$

where $\bar{\Sigma}$ is a 4×9 array with row rank equal to four.

Next consider the matrix $\bar{\bar{\Sigma}}$ obtained from Σ'' of (42) by multiplying the last two rows by $(1+x_3^2)$ in order to clear denominators. The row rank of $\bar{\bar{\Sigma}}$ is six because the six scalar equations in (42) are linearly independent. However, if we set $x_3 = i$ in $\bar{\bar{\Sigma}}$ then its row rank (and therefore column rank) becomes four, because $\bar{\bar{\Sigma}}|_{x_3=i}$ may be obtained from $\bar{\Sigma}$ by elementary column operations which cannot change the row rank. Therefore the determinant of every 6×6 submatrix of $\bar{\bar{\Sigma}}$ is a polynomial in x_3 which vanishes twice (rank deficiency $= 6 - 4 = 2$) at $x_3 = i$. By similar arguments, one may show that each such determinant also vanishes twice at $x_3 = -i$. Thus each 6×6 determinant of $\bar{\bar{\Sigma}}$ contains the factors $(x_3+i)^2(x_3-i)^2 = (1+x_3^2)^2$. This proves our claim.

References

ALBALA and ANGELES[1979]
H. Albala and J. Angeles, *Numerical solution to the input-output displacement equation of the general 7R spatial mechanism,* Proceedings of the Fifth World Congress on Theory of Machines and Mechanisms, pp. 1008-1011.

BUCHBERGER[1985]
B. Buchberger, *Gröbner Bases: An algorithmic method in polynomial ideal theory,* Chaper 6, in N.K. Bose (ed.), Multidimensional system theory, D. Reidel Publishing Co.

CANNY[1988]
J. Canny, *The complexity of robot motion planning,* MIT Press.

DENAVIT and HARTENBERG[1955]
J. Denavit and R. Hartenberg, *A kinematic notation for lower pair mechanisms based on matrices,* ASME Journal of Applied Mechanics, Vol. 77, pp. 215-221.

DUFFY and CRANE[1980]
J. Duffy and C. Crane, *A displacement analysis of the general spatial 7R mech-*

anism, Mechanisms and Machine Theory, Vol. 15, pp. 153-169.

FREUDENSTEIN[1973]
F. Freudenstein, *Kinematics: past, present and future*, Mechanisms and Machine Theory, Vol. 8, No. 2, pp. 151-161.

LEE and LIANG[1988a]
H. Lee and C. Liang, *A new vector theory for the analysis of spatial mechanisms*, Mechanisms and Machine Theory, Vol. 23, No. 3, pp. 209-217.

LEE and LIANG[1988b]
H. Lee and C. Liang, *Displacement analysis of the general spatial 7-link 7R mechanism*, Mechanisms and Machine Theory, Vol. 23, No. 3, pp. 219-226.

PAUL[1981]
R. Paul, *Robot manipulators: mathematics, programming, and control*, The MIT Press, Cambridge.

PIEPER[1968]
D. Pieper, *The kinematics of manipulators under computer control*, Ph.D. Thesis, Design Division, Stanford University.

ROTH, RASTEGAR and SCHEINMAN[1973]
B. Roth, J. Rastegar, and V. Scheinman, *On the design of computer controlled manipulators*, Proceedings of the First CISM-IFToMM Symposium, Vol. 1, pp. 93-113.

TSAI and MORGAN[1985]
L. Tsai and A. Morgan, *Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods*, Transactions of the ASME, Journal of Mechanisms, Transmissions, and Automation in Design, Vol. 107, pp. 189-200.

Power Systems Research Department, General Motors Research Laboratories, Warren, Michigan 48090

Resolution of Kinematic Redundancy

J. Baillieul *

Preliminary version

Abstract: Kinematically redundant robotic mechanisms have more than the minimal number of degrees of freedom required to carry out a given class of tasks. This paper deals with kinematic redundancy in manipulator arms and describes a body of recent research on motion planning for such devices. A short history of the inverse kinematics problem for redundant manipulators begins with a discussion of the limitations which have recently been found with pseudo-inverse methods. Specifically, it is shown that such methods do not avoid kinematic singularities and are not "repeatable". The *extended Jacobian technique* is defined and discussed, and it is noted that this is essentially the only repeatable local method for joint space path generation. It is also noted that any method (such as the extended Jacobian technique) which resolves kinematic redundancy through kinematic constraints on joint space motions inevitably involves *algorithmic singularities*. The implications of such singularities for local planning methods are discussed. Computational methods for pathwise optimal resolution of kinematic redundancy are also presented. Necessary conditions for pathwise optimality are given in terms of boundary value problems, and it is shown by example that the true optimal solution must be selected from a family of mutually nonhomotopic locally optimal solutions. It is shown how homotopy continuation may be used both to simplify calculations which would otherwise require substantial computational resources and also to single out globally optimal joint space trajectories from among (infinitely) many locally optimal candidates satisfying the basic necessary conditions.

*The author gratefully acknowledges the support of the U. S. Air Force through grant AFOSR-85-0144.

1 Introduction

Kinematic redundancy is encountered in robotics whenever a device or mechanism has more degrees of freedom than are necessary to accomplish a prescribed task. Thus kinematic redundancy will be a feature in multifingered robot hands, in manipulator arms mounted on mobile platforms, in systems involving multiple manipulator arms with shared workspace, and in manipulators for which the number of joint degrees of freedom exceeds the minimal number of degrees of freedom required to perform a given task or class of tasks. Most of the issues and results of this paper pertain equally to all these instances of kinematic redundancy. The precise statement of results requires attention to the qualitative mechanical differences in the systems, however, and to avoid overly complex and heavily bifurcated statements of results, we shall only explicitly treat kinematic redundancy in manipulator arms.

In robot kinematics we study the relationship between joint configurations and operational space configurations. In the present discussion, we shall restrict our attention to robot manipulator arms (=single strand kinematic chains) having *lower pair* joints. These are joints whose motions may be formally represented by certain distinguished subgroups of $SE(3,\mathbf{R})$, the group of rigid motions of 3-space. Following [15], we make the following definition.

Definition 1.1 *A subgroup G of $SE(3,\mathbf{R})$ will be called a* joint subgroup *if there is a neighborhood U of the identity in $SE(3,\mathbf{R})$ and a pair of rigid bodies in contact such that inside U the set of all relative configurations of these bodies is identical to G.*

Proposition 1.1 (Loncaric, [15]) *The only types of joint subgroups are $T(1)$ (=the set of all translations in a particular direction), $SO(2)$ (=the set of rotations about a given axis), $\bar{SO}(2)_p$ (=the group of screw motions with pitch p about a given axis), $SO(2)\times T(1)$, $SE(2,\mathbf{R})$, and $SO(3)$ (=the set of proper rotations of 3-space).*

The joint configuration space, M, (or simply joint space) for each of the manipulators discussed in this paper will be the product, $M = G_1 \times \cdots \times G_k$, of joint subgroups with the inherited differentiable structure. The operational space, X, of a robotic device will be a subset of a product of joint subgroups. To each joint configuration $\theta \in M$ there will correspond a unique operational space configuration $x \in X$, and we denote this functional relationship by writing

$$x = f(\theta). \tag{1}$$

At this level of abstraction, the inverse kinematics problem is to solve the equation (1) or more generally to find a right inverse for f. Because f is generally a many-to-one nonlinear function, solving (1) may be complicated and involve certain choices. The representation theory of

$SE(3,\mathbf{R})$, however, supports explicit computations. A standard approach to representation of robot kinematics which implicitly exploits this theory has evolved from the work of Denavit and Hartenberg ([9]). According to the D.-H. formalism, a coordinate frame is attached to each link in the kinematic chain. We may then represent the position and orientation of each of these coordinate frames with respect to its predecessor by means of a 4×4 ("homogeneous transformation") matrix, $\begin{pmatrix} Y & y \\ 0 & 1 \end{pmatrix}$. Here Y is a 3×3 orthogonal matrix prescribing the orientation and y is a 3-tuple giving the coordinates of the origin of the frame with respect to the preceding frame in the chain. For a more detailed explanation of this representation the reader is referred to any standard robotics text such as Craig, [8].

One rational approach to the inverse kinematics problem of associating a joint configuration θ to a given operational space configuration is to seek that value θ which optimizes some criterion function $g(\theta)$ subject to satisfying the constraint $f(\theta) = x$. Under reasonable assumptions, the mathematics underlying such pointwise optimization needn't involve more than multivariable calculus. If as is typical in robotic applications, however, there is given a path of configurations $x(t)$, it is not always straightforward to generate a corresponding path $\theta(t)$, and the attendant mathematical issues will be discussed in the remainder of this paper.

1.1 Examples of Kinematic Redundancy and Mechanisms with Task Symmetries

Since the operational space X of any robotic device is a subset of the product of joint subgroups of $SE(3,\mathbf{R})$, it is natural to study left and right actions on X by subgroups of $SE(3,\mathbf{R})$. If $X = G_1 \times \cdots \times G_k$ where each G_j is a subgroup of $SE(3,\mathbf{R})$, the natural *left action* $G \times X \to X$ is given by $g \circ (g_1, \ldots, g_k) = (gg_1, \ldots, gg_k)$, and the *right action* $X \times G \to X$ is given by $(g_1, \ldots, g_k) \circ g = (g_1 g, \ldots, g_k g)$. To simplify the discussion, we shall speak only about left actions, with the understanding that similar considerations apply to right actions.

Left actions on X are helpful in establishing a mathematical framework in which to discuss *relative configurations* in operational space. Let $G \subset SE(3,\mathbf{R})$ be a subgroup. We say there is a *G-equivalence* between operational space configurations $x_1, x_2 \in X$ if there exists a $g \in G$ such that $x_2 = gx_1$. In making this definition, it is not stipulated that G be a joint subgroup, and as illustrated in the following examples, it is at times interesting to study G-equivalence relative to finite subgroups of $SE(3,\mathbf{R})$.

Example 1.1 We take the joint space for the planar manipulator depicted in Figure 1 to be $\mathbf{T}^3 = S^1 \times S^1 \times S^1$, where each joint coordinate $\theta_i \in S^1$ denotes the relative angle between corresponding successive links. The set of operational space configurations is the set of all pos-

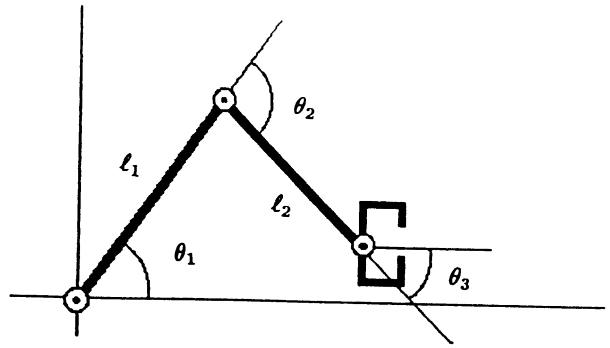


Figure 1: A planar manipulator with three revolute joints

sible positions and orientations of the end effector or hand with respect to some distinguished *base* coordinate frame. The operational space may be thought of as

$$X = \left\{ \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} : \begin{pmatrix} x \\ y \end{pmatrix} \in \mathbf{R}^2,\ 0 \le \theta < 2\pi,\ x^2 + y^2 \le \ell_1 + \ell_2 \right\}.$$

The functional relationship (1) between joint space and operational space takes the form

$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} = \begin{pmatrix} \ell_1 \cos\theta_1 + \ell_2 \cos(\theta_1 + \theta_2) \\ \ell_1 \sin\theta_1 + \ell_2 \sin(\theta_1 + \theta_2) \\ \theta_1 + \theta_2 + \theta_3 \end{pmatrix}. \tag{2}$$

To emphasize the representation of X as a joint subgroup, we may also write X in terms of the standard 3×3 (homogeneous transformation) matrix representation of $SE(2,\mathbf{R})$:

$$X = \left\{ \begin{pmatrix} \cos\theta & -\sin\theta & x \\ \sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{pmatrix} : x^2 + y^2 = 1,\ 0 \le \theta < 2\pi \right\}.$$

Apart from certain singular configurations (which we shall describe later), there are two distinct joint space configurations corresponding to each operational space configuration.

Example 1.2 If we add a joint and link to the manipulator described in Example 1.1, we obtain a mechanism with the same operational space and whose joint space is $M = \mathbf{T}^4 = S^1 \times S^1 \times S^1 \times S^1$. Note that in this case $\dim M > \dim X$.

Because there are more joint space degrees of freedom than the minimal number required (=three in this case) to position and orient the manipulator's end effector arbitrarily in its

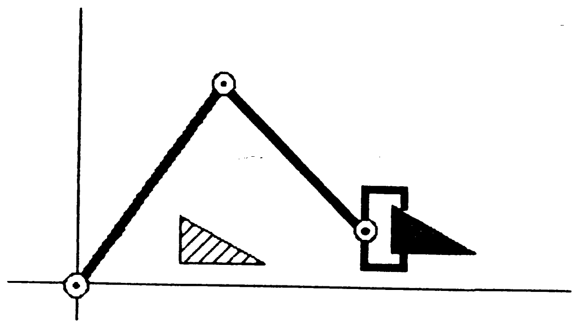


Figure 2: Planar manipulator inserting triangular part into a similarly shaped hole

work space, the mechanism described in Example 1.2 is said to be ***kinematically redundant***. The following examples will lead us to a somewhat more general definition of kinematic redundancy while at the same time pointing to the fact that many of the issues involved in motion planning for kinematically redundant mechanisms are also present in planning motions in the presence of task symmetries.

Example 1.3 Suppose the planar manipulator depicted in Figure 1 is given the task of inserting a $30° - 60°$ right triangular part into a hole with the same geometry. (See Figure 2.) The manipulator must match the part's position and orientation to that of the hole, and since there is only one way to do this, there are precisely two joint space configurations allowing the part to be brought into coincidence with the hole.

Example 1.4 Suppose next the planar manipulator is given the task of inserting a rectangular part or work piece into a similarly shaped hole. (See Figure 3). Assume the center of the rectangle has coordinates $\begin{pmatrix} a \\ b \end{pmatrix}$ with respect to the base frame of the manipulator. As depicted in the figure, if the end effector is positioned and oriented at

$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix},$$

the part will have been brought into coincidence with the hole. On the other hand, if the end effector is positioned and oriented at

$$\begin{pmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{pmatrix} = \begin{pmatrix} 2a - x \\ 2b - y \\ \theta + \pi \end{pmatrix},$$

the part will also be coincident with the hole.

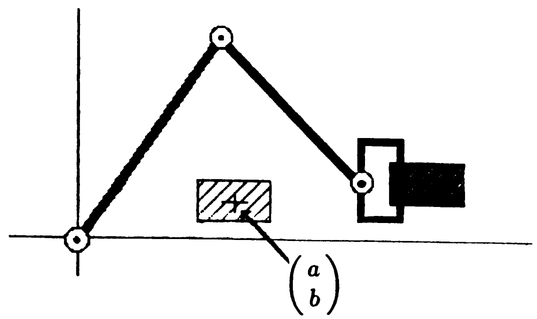


Figure 3: Planar manipulator inserting a rectangular part into a similarly shaped hole

Example 1.5 In the previous example, suppose the task is modified to insertion of a circular "peg" of radius r into a round hole also of radius r and centered at $\begin{pmatrix} a \\ b \end{pmatrix}$. The end effector is brought into coincidence with the hole for configurations of the end effector given by

$$\begin{pmatrix} x \\ y \\ \theta \end{pmatrix} = \begin{pmatrix} a - r\cos\phi \\ b - r\sin\phi \\ \phi \end{pmatrix}$$

for any ϕ.

The obvious symmetries in the two preceding examples may be formally described in terms of G-equivalence on operational space. This is most obvious when viewed from a local frame whose origin is located at $\begin{pmatrix} a \\ b \end{pmatrix}$ (=the center of the hole). With respect to this local frame, the task symmetry in Example 1.4 is described by a G-equivalence on operational space where G is a subgroup of $SE(2, \mathbf{R})$ which is isomorphic to $\mathbf{Z}_2$ and generated by the 180° rotation

$$\begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

The local frame G-equivalence of Example 1.5 is prescribed by

$$G = SO(2, \mathbf{R}) = \left\{ \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} : 0 \le \phi < 2\pi \right\}.$$

If operational space is described with respect to the base frame rather than the local frame, the symmetry groups of these examples are obtained by conjugation. For Example 1.4, G is generated by the element

$$\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{pmatrix}$$

while for Example 1.5, G is the $SO(2)$ cojugate

$$\begin{pmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ 0 & 0 & 1 \end{pmatrix}$$

where $0 \le \phi < 2\pi$. We note in both these examples that if

$$h = \begin{pmatrix} \cos\theta & -\sin\theta & x \\ \sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{pmatrix}$$

defines a position and orientation of the end effector for which the part is coincident with the hole, then any configuration gh with $g \in G$ also defines a part position/orientation which coincides with the hole.

Task symmetries of the type illustrated in these examples occur in many robotic applications. In welding for instance, any rotation of the welding torch about its longitudinal axis is irrelevant, and this task symmetry effectively reduces the dimension of the operational space. (The welding robots sold by several companies in fact have five rather than six degrees of freedom precisely in order to take advantage of this task symmetry.) More generally, we find that when a task admits a G-equivalence, the problem of positioning and orienting the end effector in the operational space X may be simplified if there is no need to distinguish between G-equivalent configurations. Distinctions among G-equivalent configurations will be important, for instance, in task applications where the manipulator's end effector is situated near the boundary of the work space, so that it is not possible to achieve all G-equivalent manipulator configurations. There will be other cases where such distinctions are not essential and where we may simplify the mathematical analysis by formally identifying G-equivalent configurations. An important case arises when the orbits of G comprise a regular foliation of X. Except for certain singularities, X will in a wide variety of interesting cases be foliated by G-orbits. When this is the case, $\tilde{X} = X/G$ has a natural differentiable structure, and the kinematic mapping $f : M \to X$ determines a differentiable mapping $\tilde{f} = \pi \circ f : M \to \tilde{X}$ where $\pi : X \to \tilde{X}$ is the natural projection. $\tilde{X}$ is called the *reduced* operational space. Note that $\dim\tilde{X} \le \dim X$, and the inequality is strict if $\dim G > 0$. We can now state the following formal definition.

Definition 1.2 *Let a robot manipulator be given with joint space M, (reduced) operational space X, and (reduced) kinematics mapping $f : M \to X$. If $\dim M > \dim X$ we say the manipulator is* kinematically redundant.

This definition enlarges the class of kinematically redundant manipulators to include mechanisms for which the redundancy is purely an artifact of symmetries in a designated task. When the task symmetry group has dimension 0, as it does in Example 1.4, the manipulator is not kinematically redundant in the formal sense. Nevertheless, the range of choices in solving the inverse kinematics problem is increased by any task symmetry, and as we shall note below, mechanisms with task symmetries exhibit some of the same path planning problems as kinematically redundant manipulators.

1.2 Problems in Motion Planning

The principal problem treated in this paper involves finding a joint space path $\theta(\cdot)$ which corresponds via (1) to a given operational space path $x(\cdot)$. The simple examples which we have given illustrate the key feature common to planning joint space motion for both redundant and nonredundant manipulators: there are (possibly infinitely) many joint space paths corresponding to each operational space path. Criteria for selecting among these include:

- A joint trajectory $\theta(t)$ should describe a closed curve whenever the corresponding end effector trajectory $x(t)$ describes a closed curve.
- Joint trajectories should be as smooth as possible (in an actual robot, rough and jerky motions could damage the mechanism), and the total motion of the joints should be no more than necessary to produce the prescribed end effector motion.
- Mechanical considerations limit the possible motions of any manipulator. Although we identify the motions of lower pair joints with joint subgroups, motions will typically be limited to some subset of the configurations specified by the group since the distance of travel in any joint will as a practical matter be limited, and of course the mechanism can never be configured such that solid members intersect one another. Joint limits should be avoided by the widest possible margin in any joint space trajectory.
- Joint space configurations at which the Jacobian $J = \frac{\partial f}{\partial \theta}$ is rank deficient are called ***kinematic singularities***. As we shall note below, when the manipulator configuration is close to a kinematic singularity, modest end effector velocities may unavoidably require very large joint velocities. Thus, we should try to specify joint trajectories which avoid singularities by a reasonable margin.
- Since there are choices to be made in specifying the motions of a kinematically redundant mechnism, these choices ought to be optimize some rational objective. Implicit in the preceding three bullets are kinematic optimality criteria. Other criteria, which we shall mention only briefly, may be defined in terms of dynamic considerations (minimum acceleration, minimum required power from actuators, etc.).

The remark regarding kinematic singularities warrants some amplification. By differentiating the relationship (1), we obtain the velocity equation

$$\dot{x} = J\dot{\theta} \tag{3}$$

(where J is the Jacobian of f). If J is rank deficient (i.e. the mechanism is in a kinematically singular configuration), there are values of $\dot{x}$ for which no $\dot{\theta}$ satisfies (3). As a practical matter, one finds that near a kinematically singular configuration moderate values of $\|\dot{x}\|$ will require very large values of $\|\dot{\theta}\|$ in order to satisfy (3). For nonredundant manipulators, kinematic singularities are in general unavoidable. Example 1.1 illustrates this point since any configuration for which $\sin\theta_2 = 0$ is singular. Thus, if $\ell_1 = \ell_2$, bringing the hand into coincidence with the base forces the mechanism into a singular configuration. Example 1.2 is also of interest, since a straightforward calculation shows that any configuration such that $\sin\theta_2 = \sin\theta_3 = 0$ is singular. This illustrates a significant point which will be discussed further in the next section: adding more degrees of freedom to a mechanism does not eliminate the possibility of singular configurations.

Whether or not a manipulator is kinematically redundant, we shall be guided by the objectives we have highlighted in choosing joint-space trajectories. In this regard, we shall examine two distinct approaches to resolving kinematic redundancy. ***Local methods*** solve (3) for $\dot{\theta}$ in terms of θ and x at each instant in time, and then generate an entire workspace path by integrating $\dot{\theta}$. ***Global methods***, on the other hand, seek to define the path $\theta(\cdot)$ in terms of the entire operational space path $x(\cdot)$. Because global methods utilize the entire operational space path, they are necessarily noncausal and thus not well suited for incorporation into real time algorithms. Local methods, by contrast only use current information to determine the next step in planning a joint motion. In principle, they have the capacity to adapt to unforseen changes in the operational space path, and they can be used in planning algorithms which utilize real time data from sensors. Moreover, as we shall note below, their implementation presents less daunting computational demands than global methods. The main disadvantage with local methods is the unavoidable occurrence of algorithmic singularities. (See Section 3.) These represent the ultimate performance penalty associated with an approach to planning which is in some reasonable sense optimal at each instant in time but which is incapable of incorporating information about future consequences of current motions. Local methods will be discussed in the next two sections, while global methods will be treated in Section 4.

Example 1.6 Consider the mechanism and work piece described in Example 1.4. Suppose we wish to move the work piece from position A to position B along the operational space path indicated in Figure 4 by grapsing it from above or below. If the work piece is grasped from below, there is considerably less overall motion required of the manipulator than if it is grasped from above. If a local method which respects the criteria for motion planning outlined above is applied from an initial grasp-from-above configuration, it is intuitively clear that the resulting motion will be far from optimal with respect to the same criteria applied from a more global perspective. This is precisely the deficiency to be expected with local methods.

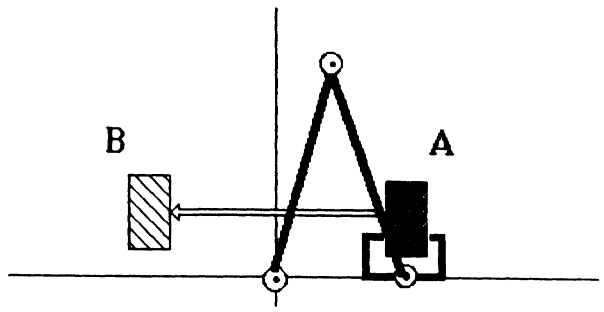


Figure 4: Manipulator task is to move the work piece from A to B along the indicated path

2 Historical Background

2.1 Singularities in Kinematically Redundant Manipulators

The practical appeal of kinematically redundant robot manipulators is their enhanced potential for working in a congested environment and for being able to avoid operating near joint limits or kinematically singular configurations. Recall from the last section that if $\ell_1 = \ell_2$ in Example 1.1, then placing its end effector at the manipulator's base forces the manipulator into a singular configuration. The kinematically redundant manipulator of Example 1.2 was also seen to have kinematic singularities, but apart from those corresponding to the end effector being located on the boundary of the work space, they are avoidable (in the sense that whenever the mechanism is in a singular configuration, it may be reconfigured via a "self motion" leaving the end effector unmoved but moving the joints into a non-singular configuration). Two questions suggested by these example are: (i) is it ever possible to add an extra joint space degree of freedom so as to create a singularity-free robot mechanism, and (ii) is there any useful mathematical characterization to distinguish between *avoidable* and *unavoidable* singularities?

A partial answer to the second question has recently been given by Shamir ([21]) who uses the qualitative theory of differential equations to classify the singularities of 3-joint redundant manipulators. The first question has been answered in the case of manipulators with one degree of freedom revolute joints by Hollerbach ([11]) using a physical argument and by Gottlieb ([10]) using a topological argument. The following theorem was discovered independently by R.W. Brockett.

Theorem 2.1 (Brockett, [6]) *Any robot manipulator whose joints are of the $SO(2)$ (=one* *degree of freedom revolute) type will have kinematically singular configurations.*

Proof: In the context of planar mechanisms, this theorem follows in a more or less straightforward manner from Example 1.2. For spatial mechanisms, the formula

$$Y = R(\vec{k}_1, \theta_1) R(\vec{k}_2, \theta_2) \cdots R(\vec{k}_n, \theta_n) \tag{4}$$

gives the standard Denavit-Hartenberg description of the orientation Y of the end effector in terms of the joint angles θ_j. Here $R(\vec{k}, \theta)$ designates the 3×3 proper orthogonal matrix corresponding to rotation θ in the counterclockwise sense about the axis defined by the unit vector $\vec{k}$.

We claim that for any three consecutive factors $R(\vec{k}_j, \theta_j) R(\vec{k}_{j+1}, \theta_{j+1}) R(\vec{k}_{j+2}, \theta_{j+2})$ in this expression there is a choice of θ_{j+1} such that $\vec{k}_j$ lies in the plane determined by $\vec{k}_{j+1}$ and $\vec{k}_{j+2}$. Suppose the coordinate frames have been chosen so that $\vec{k}_j$ is represented by $\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$ in the j-th coordinate frame in the chain, and $\vec{k}_{j+1}$ and $\vec{k}_{j+2}$ are represented respectively by $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$ in the $j+1$-st coordinate frame. It is also the case that $R(\vec{k}_{j+1}, \theta_{j+1})$ is represented by the matrix

$$\begin{pmatrix} \cos\theta_{j+1} & -\sin\theta_{j+1} & 0 \\ \sin\theta_{j+1} & \cos\theta_{j+1} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

with respect to the $j+1$-st frame, and the unit vector $\vec{k}_j$ is represented by

$$\begin{pmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \bar{x}_3 \end{pmatrix} = \begin{pmatrix} \cos\theta_{j+1} & \sin\theta_{j+1} & 0 \\ -\sin\theta_{j+1} & \cos\theta_{j+1} & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

with respect to this frame. It is clear that either $\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ or else there is some choice of θ_{j+1} such that $\begin{pmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \bar{x}_3 \end{pmatrix}$ lies in the plane determined by $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ and $\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix}$, in either case proving our claim.

Having established this claim, it then follows that for $j = 2, \ldots, n-1$, joint angles θ_j may be chosen such that all joint axes in the mechanism lie in the same plane. This is equivalent to saying that all columns in the $3 \times n$ Jacobian matrix associated with the mapping determined by the equation (4) will be orthogonal to a given unit vector (the suitably coordinatized normal to the plane in question). Hence the Jacobian will be rank deficient, proving the theorem.

2.2 Generalized Inverse Methods for Resolving Kinematic Redundancy

An early approach to resolution of kinematic redundancy in robot manipulators following a prescribed work space path $\dot{x}(t)$ was to solve (3) for the joint velocity $\dot{\theta}$ of minimum norm $\|\dot{\theta}\|$. This idea, when it was proposed by Whitney in 1969 ([22]), appeared to have two attractive features. First it seemed reasonable to hope that by minimizing $\|\dot{\theta}\|$ at each point in time, kinematic singularities would be avoided automatically, and second, because we can write the norm-minimizing solution explicitly as $\dot{\theta} = J^{\dagger}\dot{x}$, where $J^{\dagger} = J^T(JJ^T)^{-1}$ is the Moore-Penrose generalized inverse of the Jacobian, planning algorithms could be implemented recursively using standard numerical methods for ordinary differential equations. Unfortunately, closer analysis of the Moore-Penrose generalized inverse approach to motion planning for kinematically redundant manipulators has revealed several initially unanticipated problems. The follwoing theorem shows that the approach does not generate joint-space trajectories which avoid kinematic singularities in any practical sense.

Theorem 2.2 (Baillieul) *Consider a manipulator whose joint space and operational space configurations are related by (1). Let $\epsilon > 0$ be given. Let θ_s be a point at which $J = J(\theta)$ is rank deficient (i.e. θ_s defines a kinematically singular configuration), and let θ^* be a point such that $\|\theta^* - \theta_s\| < \epsilon$. Finally, let $x^* = f(\theta^*$. Then for any operational space configuration x_0 and any C^1 curve $x(\cdot)$ initiating at x_0 and terminating at x^*, there is a curve $\theta(\cdot)$ corresponding to $x(\cdot)$ via*

$$\dot{\theta} = J^{\dagger}\dot{x} \tag{5}$$

which terminates at θ^*.

Proof: Suppose ϵ, θ_s, θ^*, x_0, x^*, and the curve $x(\cdot)$ are as in the hypothesis of this theorem. By reparameterization if necessary, we may assume $x(0) = x_0$ and $x(1) = x^*$. Define a reverse trajectory in operational space by $y(t) = x(1-t)$ for $0 \le t \le 1$, and define a trajectory $\bar{\theta}(\cdot)$ in joint space by means of the differential equation

$$\dot{\bar{\theta}} = J^{\dagger}\dot{y}, \quad \bar{\theta}(0) = \theta^*.$$

Then one easily verifies that the curve $\theta(t) = \bar{\theta}(1-t)$ satisfies the conclusion of the theorem.

Another practical problem with the Moore-Penrose generalized inverse approach was pointed out by Klein and Huang ([14]), whose simulations showed that closed trajectories in operational space did not generally give rise to closed joint space trajectories. This feature was of concern to robotics engineers who felt uncomfortable not knowing how the robot's joints would configure themselves after several repeated executions of a task cycle. Topological issues involved in associating closed joint space paths to closed operational space paths were treated in [5]. In

[20] it was shown that a nonintegrable relationship between operational space paths and joint space paths could be interpreted in terms of the non-involutivity of a certain distribution of vector fields. This repeatability problem was solved in 1985 ([1]) by the *extended Jacobian method*, details of which will be given in the next section.

2.3 Pointwise Optimization Criteria for Resolving Kinematic Redundancy

We conclude this section by discussing the role optimization has played in various approaches to resolving kinematic redundancy. A great many kinematic optimality criteria have been proposed in the literature, including minimizing joint sace velocity, avoiding obstacles by the greatest possible amount with respect to various distance measures, avoiding joint limits, and avoiding kinematic singularities. Criteria closely related to the latter have included minimizing $g_1(\theta) = tr((JJ^T)^{-1})$ (Baillieul, [2]), maximizing $g_2(\theta) = \lambda_{\min}(JJ^T)$ (=smallest eigenvalue of JJ^T, Klein, [13]), and maximizing $g_3(\theta) = \sqrt{\det(JJ^T)}$, the *manipulatability index* (Yoshikawa, [23]). If we are given a kinematic function f, then according to the following theorem any optimality criterion g of this form defines an "optimality surface" which effectively resolves the redundancy.

Theorem 2.3 (Baillieul, [1]) *A necessary condition for θ_0 to maximize an objective function $g(\theta)$ subject to the positional constraint $f(\theta) = x$ is that*

$$G(\theta_0) = L_{\mathbf{n}} g\Big|_{\theta_0} = 0)$$

for all vectors n *in the null space of $J(\theta)$ where $L_{\mathbf{n}}g$ denotes the Lie derivative of the function g in the direction of the vector* **n**.

The importance of optimality criteria for functionally constraining the motions of redundant mechanisms (and thus resolving the redundancy) is not completely revealed by this theorem. The use of constraints which do not arise in conjunction with some form of optimization can inhibit motions of the mechanism in such a way as to result in significant effective reduction of work space volume. The reader may wish to construct examples of this phenomenon.

For a variety of reasons, it is desirable to understand how to define optimality criteria in a coordinate-free way. The need for coordinate-free definitions is well appreciated by geometers and may be understood quite simply in the present context as follows. Suppose the position of the end effector of a manipulator is prescribed in terms of its joint angle settings by

$$x = f(\theta)$$

where x is an m-vector, and θ is an $n > m$-vector. The redundancy may be resolved by constraining the joint variables to an m-dimensional surface. Suppose such a surface has a local parameterization of the form $\theta = \theta(u)$, where $u \in \mathbf{R}^m$. Then locally the above positional relationship is given by

$$x = G(u)$$

where $G(u) = f(\theta(u))$. It is thus tempting to define, say, the constrained analog of the above manipulatability index $g_3(\theta)$ by writing

$$g(u) = \left(\det\left[(J_u G)(J_u G)^T\right]\right)^{\frac{1}{2}},$$

but this is easily seen to depend on the chosen parameterization $\theta = \theta(u)$. The following theorem will be useful in removing this undesirable coordinate dependence.

Theorem 2.4 *The criteria given above are equivalent to the following:*

$$g_1(\theta) = \int_{S^{m-1}} \|J(\theta)^+ v\|^2 \, dv$$

$$g_2(\theta) = \frac{1}{\max_{v \in S^{m-1}} \|J(\theta)^+ v\|^2}$$

$$g_3(\theta) = \lim_{\epsilon \to 0} \sum_k \frac{(vol_m\, f(F_k(\theta,\epsilon)))^2}{(vol_m\, F_k(\theta,\epsilon))^2}$$

where $F_k(\theta,\epsilon)$ is the k-th m-dimensional slice (parallel to the k-th coordinate m-plane) through a standard n-cube with linear dimension ϵ, centered at θ, and vol_m denotes m-dimensional volume. The sum here is taken over the $\binom{n}{m}$ standard coordinate m-planes in $\mathbf{R}^n$. S^{m-1} denotes the unit m-1-sphere in $\mathbf{R}^m$.

The reader is encouraged to work out a proof of this theorem for herself. The case of g_3 is proved in [2]. An important observation made in [2] is that $\det(JJ^T)$ = *sum of squares of $m \times m$ minors of J*. I.e.

$$\det(JJ^T) = \sum_{i_1 < \ldots < i_m} \left|\frac{\partial(f_1 \cdots f_m)}{\partial(\theta_{i_1} \cdots \theta_{i_m})}\right|^2. \qquad (6)$$

Each term in the sum is the square of the Jacobian determinant of a certain mapping. Thus, as suggested by the theorem, it may be thought of as a certain element of m-dimensional volume or a ratio of volumes swept out in operational space to corresponding volumes in the joint space. This point of view turns out to be crucial in extending the definition of the manipulatability index to situations where the manipulator's joint configurations are constrained to a hyper-surface $G(\theta) = 0$. To make this extension, we must be able to compute the manipulatability index relative to surface coordinates.

Suppose that in a neighborhood of some point on the surface we may solve $G(\theta) = 0$ to write $\theta_n = \theta_n(\theta_1, \ldots, \theta_{n-1})$. In this neighborhood, the kinematics function f may be thought of as a function of $n-1$ variables: $\hat{f}(\theta_1, \ldots, \theta_{n-1}) = f(\theta_1, \ldots, \theta_{n-1}, \theta_n(\theta_1, \ldots, \theta_{n-1}))$. Each term in the sum (6) is not replaced by $|\frac{\partial(\hat{f}_1 \ldots \hat{f}_m)}{\partial(\theta_{i_1} \ldots \theta_{i_m})}|^2$ but rather by $|\frac{\partial(\hat{f}_1 \ldots \hat{f}_m)}{\partial(\theta_{i_1} \ldots \theta_{i_m})}|^2 / V_{i_1 \cdots i_m}$ where $V_{i_1 \cdots i_m}$ is the square of the element of m-dimensional volume in the surface: $V_{i_1 \cdots i_m} = (1 + (\frac{\partial \theta_n}{\partial \theta_{i_1}})^2 + \cdots + (\frac{\partial \theta_n}{\partial \theta_{i_m}})^2$. Now one may show that

$$\left|\frac{\partial(\hat{f}_1 \ldots \hat{f}_m)}{\partial(\theta_{i_1} \ldots \theta_{i_m})}\right| = \det \begin{pmatrix} \frac{\partial f}{\partial \theta_{i_1}} & \cdots & \frac{\partial f}{\partial \theta_{i_m}} & \frac{\partial f}{\partial \theta_n} \\ \frac{\partial G}{\partial \theta_{i_1}} & \cdots & \frac{\partial G}{\partial \theta_{i_m}} & \frac{\partial G}{\partial \theta_n} \end{pmatrix} / \frac{\partial G}{\partial \theta_n}.$$

Hence

$$\left|\frac{\partial(\hat{f}_1 \ldots \hat{f}_m)}{\partial(\theta_{i_1} \ldots \theta_{i_m})}\right|^2 / V_{i_1 \cdots i_m} = \frac{\left(\det \begin{pmatrix} \frac{\partial f}{\partial \theta_{i_1}} & \cdots & \frac{\partial f}{\partial \theta_{i_m}} & \frac{\partial f}{\partial \theta_n} \\ \frac{\partial G}{\partial \theta_{i_1}} & \cdots & \frac{\partial G}{\partial \theta_{i_m}} & \frac{\partial G}{\partial \theta_n} \end{pmatrix}\right)^2}{\frac{\partial G}{\partial \theta_{i_1}}^2 + \cdots + \frac{\partial G}{\partial \theta_{i_m}}^2 + \frac{\partial G}{\partial \theta_n}^2}.$$

Although this calculation suggests that we might define the constrained manipulatability index of the kinematics function f relative to the constraint function G by summing terms of this type, such an approach would clearly reflect the distinguished role the coordinate θ_n played in the above calculation. In generic situations, however, the above calculation may be repeated repeated for each coordinate θ_k, and the results may be averaged to produce the sought after invariant quantity. Hence we have the following.

Definition 2.1 *Consider a kinematically redundant manipulator having kinematics function f with joint space dimension n and reduced workspace dimension m. The* constrained manipulatability index *associated to f and the constraint $G(\theta) = 0$ is given by $m_G(\theta) =$*

$$\left(\sum_{i_1 < i_2 < \cdots < i_{m+1}} \frac{\left(\det \begin{pmatrix} \frac{\partial f}{\partial \theta_{i_1}} & \cdots & \frac{\partial f}{\partial \theta_{i_{m+1}}} \\ \frac{\partial G}{\partial \theta_{i_1}} & \cdots & \frac{\partial G}{\partial \theta_{i_{m+1}}} \end{pmatrix}\right)^2}{\frac{\partial G}{\partial \theta_{i_1}}^2 + \cdots + \frac{\partial G}{\partial \theta_{i_{m+1}}}^2} \right)^{\frac{1}{2}}.$$

A detailed exploration of the practical use of this index will not be pursued (see [2]), except that we mention our primary motivation in defining this quantity has been to study constraint hierarchies. In the following section, it will be noted that algorithmic singularities associated with the extended Jacobian technique lie on the zero locus of the constrained manipulatability index.

Remark 2.1 In the special case $n = m + 1$, the constrained manipulatability index is written as

$$\frac{\left|\det \begin{pmatrix} \frac{\partial f}{\partial \theta_1} & \cdots & \frac{\partial f}{\partial \theta_n} \\ \frac{\partial G}{\partial \theta_1} & \cdots & \frac{\partial G}{\partial \theta_n} \end{pmatrix}\right|}{\|\frac{\partial G}{\partial \theta}\|}.$$

In addition to kinematic optimality criteria of the type we have discussed, a number of authors have pursued optimization of dynamic and kinetic quantities to resolve the motion of kinematically redundant manipulators. Hollerbach and Suh ([12]) for instance, have studied the motions of trajectories designed to keep torque demands on all joint actuators within prescribed ranges. Unfortunately, simulations ([12]) have shown that such torque-based resolution of redundancy may produce unforseen and dramatically unstable motions. The problem, of course, is that all locally optimal methods for resolving redundancy ignore the future, and by optimizing a criterion at each instant, they may in fact prescribe motions which are disastrous from a global point of view. We have seen an example of local optimization resulting in poor global characteristics above where we showed that minimizing the velocity at each instant would not generate joint space trajectories which avoided kinematic singularities. Maciejewski ([16]) has discussed similar pathologies arising in motions based on local torque minimization.

3 Local Methods for Resolution of Kinematic Redundancy

Local methods for generating joint space trajectories $\theta(\cdot)$ corresponding to given operational space trajectories $x(\cdot)$ have the dual advantages of being easy to implement and having the capacity to adapt to unforseen changes in $x(\cdot)$. In this section we shall introduce the *extended Jacobian technique* which we originally proposed in [1]. This method meets the requirement of being repeatable, and it naturally incorporates kinematic optimality criteria of the type introduced in the last section. The recent work of Shamir and Yomdin ([20]) shows that this is essentially the only repeatable local method.

To incorporate an objective function g which is to be optimized in carrying out the motion, note that Theorem 2.3 implies there will be $n - m$ constraints $G_j(\theta) = 0$, $j = 1, \ldots, n-m$, where $G_j(\theta) = L_{n_j} g$, the Lie derivative of g in the j-th independent direction in the Jacobian null space. If the end effector traces a trajectory $x(t)$ along which the corresponding joint configuration $\theta(t)$ extremizes g at each point, then we have

$$\begin{pmatrix} f(\theta) \\ G_1(\theta) \\ \vdots \\ G_{n-m}(\theta) \end{pmatrix} = \begin{pmatrix} x(t) \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

Differentiating both sides provides a (forward) kinematic relationship between joint angle velocities and end effector velocity:

$$\begin{pmatrix} \frac{\partial f}{\partial \theta} \\ \cdots \\ \frac{\partial G}{\partial \theta} \end{pmatrix} \cdot \dot\theta(t) = \begin{pmatrix} \dot x(t) \\ 0 \end{pmatrix}. \tag{7}$$

The coefficient matrix

$$J_e = \begin{pmatrix} \frac{\partial f}{\partial \theta} \\ \cdots \\ \frac{\partial G}{\partial \theta} \end{pmatrix}$$

is square, and we shall call this the *extended Jacobian.*

Provided the extended Jacobian is nonsingular along a trajectory of interest, we may solve the inverse kinematics problem by writing

$$\dot\theta(t) = J_e^{-1} \begin{pmatrix} \dot x(t) \\ 0 \end{pmatrix}.$$

Obviously the extended Jacobian will be singular at any point where $J = \frac{\partial f}{\partial \theta}$ is singular. Other singular points of the extended Jacobian are characterized by the following.

Theorem 3.1 (Baillieul, [3]) *Suppose θ_0 is a joint space configuration at which J has full rank m. Then J_e is singular if and only if the linear operator defined by the $(n-m) \times n$ matrix $\frac{\partial G}{\partial \theta}$ restricted to the null space of J has rank $< n - m$.*

Proof: Suppose J_e is singular. If $\frac{\partial G}{\partial \theta}(\theta_0)$ has rank $< n - m$, it has rank $< n - m$ *a fortiori* when restricted to the subspace ker$J$. If $J_e(\theta)$ is singular but both J and $\frac{\partial G}{\partial \theta}$ have full rank, there is a row vector $v \neq 0$ such that $v \cdot J_e(\theta_0) = 0$, and such that v has nonzero components v_i with $i \le m$ and also nonzero components v_i with $i > m$. But this implies there is an $(n-m)$-tuple $\bar v \neq 0$ such that $\bar v \cdot \frac{\partial G}{\partial \theta}(\theta_0)$ lies in the row space of $J(\theta_0)$. Then $\bar v \cdot \frac{\partial G}{\partial \theta}(\theta_0) \cdot \hat n = 0$ for all $\hat n \in$ ker$J(\theta_0)$, and hence $\frac{\partial G}{\partial \theta}(\theta_0)$ has rank $< n - m$ on ker$J(\theta_0)$.

Suppose, on the other hand, that $\frac{\partial G}{\partial \theta}(\theta_0)$ has rank $< n-m$ on ker$J(\theta_0)$. Let N be an $(n-m)\times n$ matrix whose rows span ker$J(\theta_0)$. We may write

$$J_e \cdot (J^\dagger \vdots N^T) = \begin{pmatrix} I & 0 \\ A(\theta) & B(\theta) \end{pmatrix}, \tag{8}$$

and since $B(\theta_0) = \frac{\partial G}{\partial \theta}(\theta_0) N^T$ has rank $< n - m$ and *rank* $J_e = m +$ *rank* $B(\theta_0)$, it follows $J_e(\theta_0)$ is singular, proving the theorem.

It was shown in Baillieul *et al.* ([4]) that any C^1 joint space trajectory not passing through a singularity of J could be generated by an appropriate choice of $v(\cdot)$ in

$$\dot\theta = J^\dagger \dot x + N^T v(t). \tag{9}$$

The problem of specifying $v(\cdot)$ in some useful way can be approached via the *extended Jacobian technique*. Multiplying both sides of (8) by the composite vector $\begin{pmatrix} \dot x \\ v \end{pmatrix}$, we see from (8) and (9) that

$$0 = A(\theta)\dot x + B(\theta)v. \tag{10}$$

If J_e is invertible, we have seen that $B(\theta)$ has rank $n - m$. In this case, v can always be found. If J_e is not invertible, it may still be possible to find a v which solves (10), but the conditions are more delicate. This motivates the following.

Definition 3.1 *Points in the joint space where (10) cannot be solved for some value of $\dot x$ are called* algorithmic singularities *associated with the constraint(s)* $G_1(\theta) = 0, \ldots, G_{n-m}(\theta) = 0$.

It is important to emphasize that algorithmic singularities are *not* artifacts of the extended Jacobian technique. Unfortunately, they are inevitably introduced if functional constraints (or optimality criteria) are used to resolve the redundancy in almost any manipulator having revolute joints.

Example 3.1 Suppose the optimality criterion is the *manipulatability index* $g(\theta) = \sqrt{\det(JJ^T)}$ discussed in the previous section. Consider the (redundant) planar manipulator of Example 1.2. The position of the end effector with respect to the base is given by

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \ell_1 \cos\theta_1 + \ell_2 \cos(\theta_1+\theta_2) + \ell_3 \cos(\theta_1+\theta_2+\theta_3) \\ \ell_1 \sin\theta_1 + \ell_2 \sin(\theta_1+\theta_2) + \ell_3 \sin(\theta_1+\theta_2+\theta_3) \end{pmatrix}.$$

Maximizing the manipulatability index for this kinematics function is equivalent to maximizing $g(\theta) = \sin^2\theta_2 + \sin^2\theta_3$. This defines the constraint $G(\theta) = L_g \hat n = 0$ in terms of which the extended Jacobian is defined. (When $\ell_j = 1$ for $j = 1,2,3$, this constraint is simply $\theta_2 - \theta_3 = 0$.) As noted in [1], the algorithmic singularities for this criterion occur when the end effector touches the base, forming a closed kinematic chain allowing no motions in joint variables θ_2 and θ_3. These algorithmic singularities may also be interpreted in terms of the constrained manipulatability index, $m_G(\theta) = |\det(J_e)|/\|\frac{\partial G}{\partial \theta}\|$, of Section 2. If $\ell_j = 1$ for $j = 1,2,3$, we have $m_G(\theta) = \sqrt{2} \cdot |\sin\theta_2| \cdot |1 + 2\cos\theta_2|$. Setting this equal to zero, we find the singularities associated with the criterion occur when $\sin\theta_2 = 0$ (which are the singularities of J) and when $\theta_2 = \pm 2\pi/3$ which are the algorithmic singularities associated with the constraint.

4 Pathwise Resolution of Kinematic Redundancy

Recently, attention has turned to global optimization as a means of resolving kinematic redundancy. Nakamura and Hanafusa ([19]) describe this approach and present an algorithm for computing globally optimal joint space trajectories for redundant manipulators. In the present section, we present an alternative formulation of the necessary conditions, describe qualitative features of extremal trajectories, and investigate several new approaches to algorithm design for these computationally rigorous problems.

We begin with the velocity equation (3). Our approach is to formulate the problem of finding a joint-space trajectory as a constrained optimization problem where we choose to minimize the path integral

$$\int_0^T \left(\frac{1}{2}\dot{\theta}^T W^{-1}\dot{\theta} + g(\theta)\right) dt \tag{11}$$

subject to the kinematic constraint $x(t) = f(\theta(t))$. In this criterion, we assume the weighting matrix W is symmetric, and it is allowed to have an explicit dependence on t.

Theorem 4.1 *Joint space trajectories which optimize (11) satisfy*

$$\ddot{\theta} = J_W^{\dagger}(\ddot{x} - \dot{J}\dot{\theta}) + \mathbf{P}_W(\dot{W}W^{-1}\dot{\theta} + Wg_{\theta}) \tag{12}$$

where

$$\mathbf{P}_W = (I - J_W^{\dagger}J)$$

is the weighted null space projection operator of the Jacobian, J, and

$$J_W^{\dagger} = WJ^T(JWJ^T)^{-1}$$

and $J_W^{\dagger}$ is the weighted pseudo-inverse of J.

Proof: This theorem follows from a variational argument in which the joint space trajectory variations $\delta\theta$ are constrained to lie in the direction of the null space of J. The Euler-Lagrange operator defined by the functional (11) is

$$\frac{d}{dt}(W^{-1}\dot{\theta}) - \frac{\partial g}{\partial \theta},$$

and if $\theta(\cdot)$ minimizes (11) over all trajectories which vary with respect to $\theta(\cdot)$ in the direction $\ker J(\theta(t))$—i.e. with respect to all trajectories corresponding to the given operational space path $x(\cdot)$—, it follows that $\theta(\cdot)$ must satisfy

$$\mathbf{P}_W\{W[\frac{d}{dt}(W^{-1}\dot{\theta}) - \frac{\partial g}{\partial \theta}]\} = 0. \tag{13}$$

Note that $\mathbf{P}_W$ is the orthogonal projection relative to the inner product defined by the symmetric positive definite matrix W^{-1}. If we differentiate (1) twice with respect to t, we obtain

$$\ddot{x} = \dot{J}\dot{\theta} + J\ddot{\theta}$$

which may be equivalently written as

$$\ddot{\theta} = J_W^{\dagger}(\ddot{x} - \dot{J}\dot{\theta}) + \mathbf{P}_W\nu \tag{14}$$

for an appropriate choice of ν. Multiplying both sides of this equation by the projection operator $\mathbf{P}_W$ and noting that $\mathbf{P}_W J_W^{\dagger} = 0$, we obtain

$$\mathbf{P}_W\ddot{\theta} = \mathbf{P}_W\nu.$$

Using (13), we find that this implies

$$\mathbf{P}_W\nu = \dot{W}W^{-1}\dot{\theta} + W\frac{\partial g}{\partial \theta},$$

and substituting this into (14) proves the theorem.

In order to uniquely specify the optimal solution, one must consider boundary conditions as well as the necessary condition provided by (12). When we first began studying integral criteria for resolving kinematic redundancy, it had been hoped that by minimizing the integral criterion (11) we would find joint-space trajectories that met the kinematic constraint (1) while staying more than a minimum distance from kinematic singularities. Unfortunately, as has been noted in [18], solutions to (12) may include singular points if the boundary conditions are chosen in a sufficiently malicious way. In the present paper, we shall treat the case of periodic boundary conditions. This is natural since we are treating periodic operational space trajectories. Thus if x satisfies $x(0) = x(T)$, the objective is to find trajectories which statisfy (12) subject to $\theta(0) = \theta(T)$ and $\dot{\theta}(0) = \dot{\theta}(T)$.

Example 4.1 The Existence of Multiple Nonhomotopic Extremal Solutions This circle of ideas may be applied to path planning for the manipulator studied in Example 1.2. This geometry has been chosen for the purpose of illustration since it is especially simple to observe qualitative differences in comparing optimal and non-optimal trajectories. The necessary conditions expressed in terms of the system of differential equation (12) together with periodic boundary conditions will, in general, be sufficient to determine a family of trajectories that extremize the path integral (11). Unfortunately, not all these extremal trajectories are optimal, nor is the optimal trajectory among them always unique. These results were generated numerically using the necessary conditions ((12) and periodic boundary conditions) just presented. (Note we have chosen $g(\theta) \equiv 0$ and $W(t) \equiv I$.)

Aside from multiple solutions due to symmetry, other situations arise in which a trajectory satisfying equation (12) and periodic boundary conditions is locally optimal, but not globally

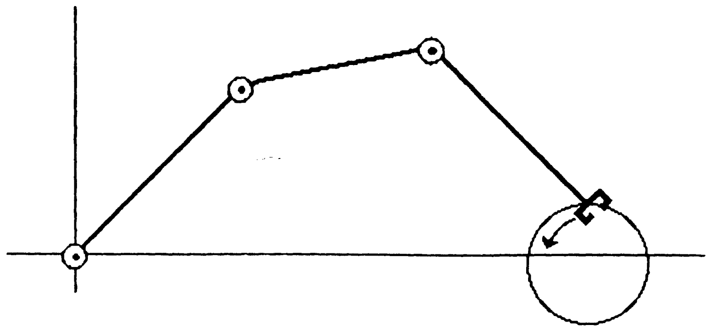


Figure 5: A three bar planar manipulator to trace a circle

optimal. These situations are related to the geometry of the solution space of the kinematic constraint equation (1). Figure 5 shows the geometry of the planar three bar manipulator, and the trajectory in the workspace that it is to trace. The endpoint of the manipulator is to describe the circle of unit radius, in unit time, in a counterclockwise direction.

Figures 6 and 7 show the joint angle time histories of two different trajectories that satisfy equation (12) and are periodic. The trajectory of Figure 6 is of much lower cost than that of Figure 7. In order to better understand this behavior, Figure 8 shows the relationship of these two trajectories when viewed as trajectories in the joint space. We see in this figure a three dimensional rendering of the set of joint angles that satisfy the kinematic constraint $f(\theta) = x(t)$ for some t. Since the manipulator has one degree of redundancy, the set of such joint angles is a set of dimension one greater than the dimension of the set $x(t)$. In this case, it is a surface equivalent to a deformed torus. For fixed t, the set of θ-values corresponding to $x(t)$ is a closed curve that circles the deformed torus about it's larger radius. As time increases from 0 to 1, the manipulator will trace out a curve that circles the torus about the small radius, in order to stay on the desired trajectory.

As the figure shows, the lower cost trajectory encircles the torus about the smaller radius at a point where the total distance (in the joint space) that must be travelled is small. The higher cost trajectory encircles the torus about the small radius once, but at the same time, encircles the torus once about the larger radius. Roughly speaking, the manipulator traverses the null space once while performing the motion. It is important to note that the longer trajectory cannot be deformed continuously into the shorter trajectory; we have thus arrived at solutions to the inverse kinematics problem that are in different homotopy classes. Both curves are locally optimal solutions within their homotopy classes.

This example shows that algorithms based on the necessary equations alone will not be able to

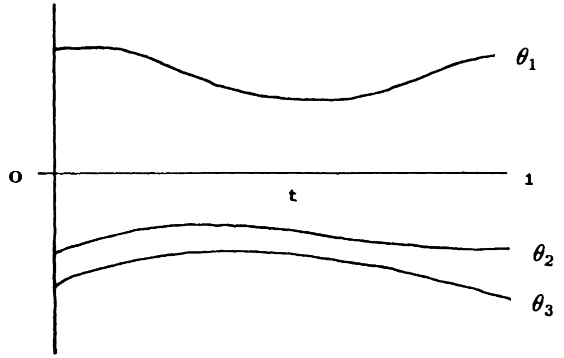


Figure 6: Extremal solution A to the necessary conditions

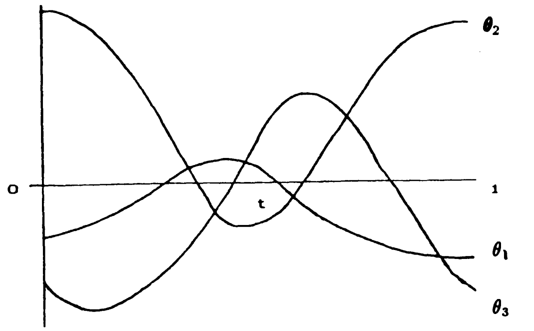


Figure 7: Extremal solution B to the necessary conditions

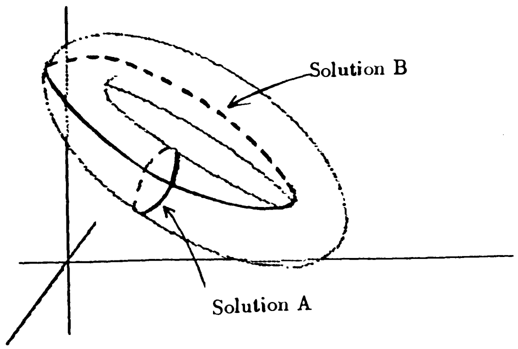


Figure 8: Solution A and solution B trajectories drawn on surface $\{\theta : f(\theta) = x(t), 0 \leq t \leq 1\}$

distinguish solutions in differing homotopy classes, without further refinement. In part to address this problem, and also in part to address the problem of finding computationally efficient ways of solving the boundary value problems associated with global path planning methods, we conclude this section with a report of recent results on applying homotopy continuation methods to the path planning problem.

The specific problem we shall address is that of minimizing the cost functional

$$\int_0^T \left(\frac{\epsilon}{2}\|\dot{\theta}\|^2 + (1-\epsilon)g(\theta)\right) dt \tag{15}$$

subject to the constraint (1) and the periodic boundary conditions

$$\theta(0) = \theta(T), \quad \dot{\theta}(0) = \dot{\theta}(T). \tag{16}$$

We assume the operational space trajectory $x(\cdot)$ is closed (i.e. $x(0) = x(T)$), and the function g appearing in the integrand is a differentiable function of the joint space configuration. (This might be the manipulatability index discussed in the last section, for example.) We also assume for simplicity that the degree of redundancy $n - m = 1$. Equation (12) now takes the form

$$\epsilon\hat{n} \cdot \ddot{\theta} = (1-\epsilon)G(\theta) \tag{17}$$

where

$$G(\theta) \equiv \frac{\partial g}{\partial \theta}(\theta) \cdot \hat{n}(\theta), \tag{18}$$

and $\hat{n}(\theta)$ is the normalized null space vector of the Jacobian $J(\theta)$.

The homotopy parameter ϵ varies between 0 and 1, and when $\epsilon = 0$, the problem reduces to that of extremizing the integral of $g(\theta)$ subject to the constraint. It is not difficult to show that this is equivalent to pointwise optimization of g subject to the constraint (1) at each point θ along the path. Since at $\epsilon = 0$ the motion planning problem is much easier to solve (using, say, the extended Jacobian technique) than the general pathwise optimal problem, we would like to understand whether solutions to the $\epsilon = 0$ problem can in fact be used to initiate a homotopy continuation. We note that this is a potentially troublesome question since the character of the optimization problem (15) changes dramatically as $\epsilon \to 0$. The necessary conditions change from the second order differential equation (17) to the algebraic equation $G(\theta) = 0$, so that the homotopy involves a *singular perturbation*.

A general theory of the qualitative behavior of the necessary conditions (17)-(16) in the vicinity of $\epsilon = 0$ is under development. In light of the above example, we do not expect that every extremal solution to the optimization problem prescribed by (17)-(16) can be continuously deformed into a solution of the given local optimization problem. To emphasize this point, we look once again at the manipulator of Example 1.2 together with the local criterion function $g(\theta) = \sin^2\theta_2 + \sin^2\theta_3$. In Example 3.1 we saw that in the case $\ell_j = 1$, $j = 1,2,3$, this gives rise to the constraint $G(\theta) = \theta_2 - \theta_3$. The extended Jacobian technique will generate a joint

space trajectory whose point set is a subset of the intersection of the surfaces $\{\theta : G(\theta) = 0\}$ and $\{\theta : f(\theta) = x(t) \text{ for } 0 \le t \le T\}$. For our example, the zero locus of G intersects the inverse operational space curve locus depicted in Figure 8 in curves which encircle the torus about two small equators. Note that of the two extremal trajectories which we have found for the global minimization problem in Example 4.1, only the trajectory we have called Solution A is homotopic to curves associated with the local optimization problem. Solution B, being in a different homotopy class, obviously cannot be continuously deformed into the solution prescribed by the extended Jacobian technique with this criterion function. A detailed study of the continuous dependence of solutions to the global optimization problem on ϵ appears in [17].

A standard approach in homotopy continuation is to study the homotopy $\theta_\epsilon(t)$ in terms of Davidenko's differential equation relating $\frac{d\theta_\epsilon}{d\epsilon}$ and θ_ϵ. The next theorem shows that this equation is hyperbolic.

Theorem 4.2 (Martin, [17]) *Let $\theta_\epsilon(t)$ be a parameterized family of solutions to the optimization problem (15) with constraint (1) and boundary conditions (16). Suppose $\theta_\epsilon(t)$ is jointly continuous in ϵ and t with $\lim_{\epsilon\to 0}\theta_\epsilon(t) = \theta_0(t)$ where $\theta_0(t)$ satisfies*

$$\theta_0(t) = \min_{f(\theta)=x(t)} g(\theta).$$

Then $\frac{d\theta_\epsilon(t)}{dt}$ satisfies

$$\epsilon\hat{n} \cdot \frac{d\ddot{\theta}_\epsilon}{d\epsilon} + \left[\epsilon\bar{\theta}_\epsilon \cdot \frac{\partial \hat{n}}{\partial \theta} - (1-\epsilon)\frac{\partial G}{\partial \theta}\right] \cdot \frac{d\theta_\epsilon(t)}{d\epsilon} + (\hat{n} \cdot \ddot{\theta}_\epsilon + G(\theta_\epsilon)) = 0 \tag{19}$$

together with periodic boundary conditions. For small ϵ, this equation restricted to $\{\theta : f(\theta) = x(t)\}$ is hyperbolic.

Proof: Equation (19) is obtained by differentiating (17) with respect to ϵ. To see that the equation when restricted to the iverse curve locus $\{\theta : f(\theta) = x(t)\}$ is hyperbolic, we first observe that the derivative $\frac{d\theta_\epsilon}{d\epsilon}$ must lie in the null space direction. This follows from differentiating the kinematics constraint (1) with respect to ϵ. This implies that we may write

$$\frac{d\theta_\epsilon(t)}{d\epsilon} = \hat{n}\alpha_\epsilon(t). \tag{20}$$

Now equation (19) may be rewritten in terms of α_ϵ:

$$\epsilon(\ddot{\alpha}_\epsilon - \dot{\hat{n}} \cdot \dot{\hat{n}}\alpha_\epsilon) + \epsilon\bar{\theta}_\epsilon \cdot \frac{\partial \hat{n}}{\partial \theta} \cdot \hat{n}\alpha_\epsilon - (1-\epsilon)\frac{\partial G}{\partial \theta} \cdot \hat{n}\alpha_\epsilon = c(t) \tag{21}$$

where $c(t) = -\hat{n} \cdot \ddot{\theta}_\epsilon - G(\theta_\epsilon)$. Here we have used the relations $\dot{\hat{n}} \cdot \hat{n} = 0$ and $\ddot{\hat{n}} \cdot \hat{n} = -\dot{\hat{n}} \cdot \dot{\hat{n}}$. If we define

$$y = \begin{pmatrix} \alpha_\epsilon \\ \dot{\alpha}_\epsilon \end{pmatrix},$$

(21) may be rewritten (for $\epsilon \neq 0$) as

$$\dot{y} = \begin{pmatrix} 0 & 1 \\ \frac{a(t)}{\epsilon} + b(t) & 0 \end{pmatrix} y + \begin{pmatrix} 0 \\ \frac{1}{\epsilon}c(t) \end{pmatrix} \tag{22}$$

where

$$a(t) = \frac{\partial G}{\partial \boldsymbol{\theta}} \cdot \hat{n}, \quad b(t) = \dot{\hat{n}} \cdot \dot{\hat{n}} - \dot{\boldsymbol{\theta}}_\epsilon \cdot \frac{\partial \hat{n}}{\partial \boldsymbol{\theta}} \cdot \dot{\hat{n}} - \frac{\partial G}{\partial \boldsymbol{\theta}} \cdot \hat{n}.$$

For $0 < \epsilon \ll 1$, these equations will be clearly hyperbolic if we can show that $a(t) > 0$ since the characteristic equation of the coefficient matrix in (22) has eigenvalues $\pm\sqrt{\frac{a(t)}{\epsilon}}$. We have assumed that $\boldsymbol{\theta}_0(t)$ is a strict minimum of g, subject to the constraint $f(\boldsymbol{\theta}) = x(t)$, which in turn implies that $G(\boldsymbol{\theta}_0(t)) = 0$ and $\frac{\partial G}{\partial \boldsymbol{\theta}} \cdot \hat{n}(\boldsymbol{\theta}_0(t)) > 0$. By continuity this inequality must hold for all $\epsilon > 0$ sufficiently close to $\epsilon = 0$, showing that for $\epsilon > 0$ sufficiently small $a(t) > 0$. This proves the theorem.

The hyperbolicity of Davidenko's equation (19) indicates that while it may be used in principle to determine $\boldsymbol{\theta}_\epsilon(t)$ for small $\epsilon > 0$, it will also be likely to suffer from the numerical instabilities commonly associated with stiff differential equations. In [17], it is shown that in fact (19) exhibits an *exponential dichotomy* (in the sense of Coppel, [7]), and stable numerical integration procedures may be developed on this basis. Recent numerical studies have also shown that computations may be carried out in parallel to propagate solutions $\boldsymbol{\theta}_\epsilon$ to (19) with respect to ϵ for an appropriate discretization with respect to t. This approach has proven to be extremely efficient compared with more traditional computational methods for treating these boundary value problems.

Many questions regarding pathwise optimal resolution of kinematic redundancy remain at this writing. While the approach we have described may be thought of as being "global" with respect to a given operational space path, it remains unknown how the results of the method may be expected to change as operational space paths are varied. The local methods of Section 3 were seen to inevitably involve algorithmic singularities, and it seems unlikely that the repeatable path otimal methods can be free of such pathology. Indeed, the existence of multiple, nonhomotopic, locally optimal liftings of operational space paths suggests that as problem parameters are varied, there will be bifurcations of such solutions. There may be much work remaining to understand such phenomena.

References

[1] Baillieul, J., 1985. "Kinematic Programming Alternatives for Redundant Manipulators," *IEEE International Conference on Robotics and Automation*, St. Louis MO, March 25-28, pp. 722-728.

[2] Baillieul, J., 1987. "A Constraint Oriented Approach to Inverse Problems for Kinematically Redundant Manipulators," *IEEE International Conference on Robotics and Automation*, Raliegh, NC, March 31-April 3, pp. 1827-1833.

[3] Baillieul, J., 1986. "Avoiding Obstacles and Resolving Kinematic Redundancy," *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, CA, April 7-10, pp. 1698-1704.

[4] Baillieul, J., Hollerbach, J.M., and Brockett, R.W., 1984. "Programming and Control of Kinematically Redundant Manipulators," *Proc. of 1984 IEEE Conference on Decision and Control*, Las Vegas, Nevada, pp. 768-774.

[5] Baker, D. R. and Wampler, C. W. 1988. "On the Inverse Kinematics of Redundant Manipulators," *International Journal of Robotics Research*, vol. 7, no. 2, pp. 3-21.

[6] Brockett, R.W., 1983. Private communication.

[7] Coppel, W.A., 1978. *Dichotomies in Stability Theory*, Springer-Verlag, Berlin.

[8] Craig,J.J., 1989. *Introduction to Robotics: Mechanics and Control, Second Ed.*, Addison-Wesley, Reading, MA.

[9] Denavit, J., and Hartenberg, R.S., 1955. "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," ASME: J. of Appl. Mech., Series E, pp. 215-221.

[10] Gottlieb, D.H., 1986. "Topology and Robots," *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, pp. 1689-1691.

[11] Hollerbach, J.M., 1985. "Optimum Kinematic Design for a Seven Degree of Freedom Manipulator," appearing in *Robotics Research—The 2nd International Symposium*, H. Hanafusa and H. Inoue, eds., MIT Press, Cambridge, MA, pp. 215-222.

[12] Hollerbach, J.M., and Suh, K.C., 1987. "Redundancy Resolution of Manipulators through Torque Optimization," ***IEEE Journal of Robotics and Automation***, vol. RA-3, no. 4, pp. 308-316.

[13] Klein, C.A., 1985. "Use of Redundancy in the Design of Robotic Systems," ***Robotics Research: The Second International Symposium***, Hanafusa and Inoue, eds., MIT Press, Cambridge, MA, pp. 207-214.

[14] Klein, C.A., and Huang, C.-H., 1983. "Review of Pseudoinverse Control for Use with Kinematically Redundant Manipulators," ***IEEE Trans. on Systems, Man, and Cybernetics***, SMC-13, pp. 245-250.

[15] Loncaric, J., 1985. ***Geometrical Analysis of Compliant Mechanisms in Robotics***, PhD Thesis, Division of Applied Sciences, Harvard University.

[16] Maciejewski, A.A., 1989. "Kinetic Limitations on the Use of Redundancy in Robotic Manipulators," ***Proc. IEEE Int. Conf. on Robotics and Automation***, Scottsdale, Arizona, pp. 113-118

[17] Martin, D.P., 1990. "Mathematical Methods for Problems of Kinematic Redundancy in Robotics," Technical Report. Department of Aerospace and Mechanical Engineering, Boston University.

[18] Martin, D.P., Baillieul, J., and Hollerbach, J.M. 1989. "Resolution of Kinematic Redundancy Using Optimization Techniques" ***IEEE Transactions on Robotics and Automation***, v. 5, no. 4, pp. 529-533.

[19] Nakamura, Y. and Hanafusa, H., 1987. "Optimal Redundancy Control of Robot Manipulators," ***International Journal of Robotics Research***, vol. 6, no. 1, pp. 32-42.

[20] Shamir, T., and Yomdin, Y., 1988. "Repeatability of Redundant Manipulators: Mathematical Solution of the Problem," ***IEEE Transactions on Automatic Control***, AC-33, No. 11, pp. 1004-1009.

[21] Shamir, T., 1989. "On Singularities of Redundant Robot Arms." To appear.

[22] Whitney, D.E. 1969. "Resolved Motion Rate Control of Manipulators and Human Prostheses," *IEEE Trans. on Man-Machine Systems*, MMS-10, pp. 47-53.

[23] Yoshikawa, T., 1984. "Manipulatability of Robotic Mechanisms," *Second International* ***Symposium on Robotics Research***, Kyoto, Japan, August 20-23.

DEPARTMENT OF AERSOPACE/MECHANICAL ENGINEERING, BOSTON UNIVERSITY, BOSTON, MA, 02215

Mathematical Problems in Grasping and Manipulation by Multifingered Robot Hands

Richard M. Murray* S. Shankar Sastry†

1 Introduction

In these notes we give the reader a feel for the mathematical problems involved in describing grasping and fine motion maipulation of objects with multifingered robot hands. Multifingered robot hands can be thought of as several robots (fingers) on a common base (palm) cooperatively manipulating an object. It is clear that positioning an object in space, namely specifying its position and orientation needs 6 degrees of freedom. However, dextrously manipulating objects requires far more degrees of freedom especially in the execution of tasks involving picking up an object, regrasping it and using the object. This is where the study of multifingered hands comes in. The study of multifingered hands has a long history not just in the context of robotics but also in the context of prosthesis.

In Chapter 2, we set down a brief discussion of the kinematics of a single rigid body, followed by a study of contacts and the kinematics of rolling. Rolling is an especially important way in which finger tips move over the surface of an object in order both to reposition and regrasp the object. In Section 2.4 we study the kinematics of a multifingered hand in terms of the kinematics of the individual fingers. Finally, we define grasp stability and the manipulability of grasps. The appendix contains a derivation of the contact equations in terms of the metric tensor and connection form of the surfaces in contact at the finger tip and object.

In Chapter 3, we develop the dynamcis of multifingered hands by aggregating the dynamics of individual fingers with the dynamics of the grasped object and the kinematic equations of contact. In Section 3.3 we describe a few different control techniques to follow a specified trajectory for the body and the grasp forces exerted on it.

In Chapter 4, we axiomatize the process of regrasping an object by rolling the finger tips on the surface of the object. We show how the problem of finding geodesics for singular or nonholonomic or Carnot-Caratheodory metrics is useful in steering the finger tips from one grasp to another. We conclude with some open problems.

The notation used throughout these notes is summarized after Chapter 4.

*Supported in part by an IBM Manufacturing Fellowship
†Supported in part by NSF-PYI grant DMC84-51129

The discussion of this paper is a summary of our own work and that of others notably those at Harvard in the last few years in this area. Detailed references to these appear in the body of the notes.

2 Kinematics and statics

This chapter provides a brief introduction to grasping and the notation used in this paper. We derive the basic velocity and force transformations for both fixed and rolling contacts. For a more complete discussion of the kinematics of grasping see Kerr [5] and Montana [12].

2.1 Rigid body kinematics

A rigid motion of an object is a motion which preserves distance and orientation. Every such rigid motion can be represented by a rotation followed by a translation. Letting $SO(3)$ represent the group of all proper 3×3 rotation matrices and $\mathbf{R}$ denote the real numbers, we can represent a rigid motion by the pair $(R, p) \in SO(3) \times \mathbf{R}^3$. We define $SE(3) = SO(3) \times \mathbf{R}^3$ to be the set of all rigid motions and note that $SE(3)$ is a manifold of dimension 6 as well as a group. It may be verified that $SE(3)$ is a Lie group.

The configuration of a rigid body with respect to some identity configuration is described by an element of $g \in SE(3)$. g acting on a point attached to the body defines the new location of the point relative to its identity configuration. If $q \in \mathbf{R}^3$ is a point on the body relative to some base (world) reference frame, then the location of q with respect to that basis after the body undergoes a rigid motion g is

$$g(q) = Rq + p \tag{1}$$

where R and p are represented in the same basis as q. This action is shown pictorially in Figure 1. We refer to the absolute coordinates as the *world* or *base coordinates* and the coordinates of a point on the object relative to the identity configuration as the *body coordinates*.

An object trajectory is described by a time parameterized curve, $g(t) \in SE(3)$. The velocity of an object is a tangent vector at g, so $\dot{g} \in T_g SE(3)$. $\dot{g}$ also acts on points in $\mathbf{R}^3$, giving a velocity vector $\dot{g}(q) \in \mathbf{R}^3$. Since $SE(3)$ is a Lie group, we can associate each element of $T_g SE(3)$ with the Lie algebra $se(3) \approx T_e SE(3)$ where e is the identity element. An element $\xi \in se(3)$ can be represented as a skew-symmetric matrix, $\mathcal{S} \in so(3)$ and a vector $v \in \mathbf{R}^3$. Furthermore, any skew symmetric matrix has the form:

$$\mathcal{S} = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{2}$$

and hence we will often write $\mathcal{S}(\omega) \in so(3)$ to be the skew symmetric matrix associated with $\omega \in \mathbf{R}^3$. Note that $\mathcal{S}(\omega)q = \omega \times q$.

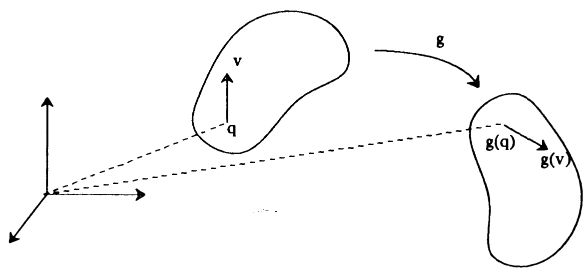


Figure 1: Rigid motion

There are two ways to map $T_gSE(3)$ to $T_eSE(3)$ — left and right translation. The usual method is to use left translation, $L_{g^{-1}}$, where $L_g h = g \circ h$. The tangent map of $L_{g^{-1}}$ maps $T_gSE(3)$ to $T_eSE(3)$ and when applied to $\dot g$, the resulting map, $T_{g^{-1}}(L_{g^{-1}})\dot g$, takes a point in body coordinates to the velocity in body coordinates. For our purposes it is more natural to use the velocity of the point in world coordinates. This can be accomplished by using right translation and the resulting map takes a point in world coordinates to a velocity in world coordinates. Formally, we define the generalized velocity, $\xi \in T_eSE(3)$, in terms of $\dot g \in T_gSE(3)$ as

$$\xi = \dot g g^{-1} \tag{3}$$

The generalized velocity ξ is also called a **twist**.

Elements of $SE(3)$ can be represented as 4×4 matrices, referred to as *homogeneous coordinates*. If $g \in SE(3)$ we write

$$g = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix} \tag{4}$$

A point $q \in \mathbf{R}^3$ can be represented as a vector in $\mathbf{R}^4$ by defining $\bar q = (q, 1) \in \mathbf{R}^3 \times \mathbf{R}$. Using this representation, $g(q)$ becomes matrix multiplication

$$g\bar q = \begin{bmatrix} R & p \\ 0 & 1 \end{bmatrix}\begin{pmatrix} q \\ 1 \end{pmatrix} = \begin{pmatrix} Rq + p \\ 1 \end{pmatrix} \tag{5}$$

To simplify notation we shall usually refer to $\bar q$ simply as q.

The generalized velocity of a motion, in world coordinates, is

$$\xi = \dot g g^{-1} = \begin{bmatrix} \dot R R^T & \dot p - \dot R R^T p \\ 0 & 0 \end{bmatrix} \tag{6}$$

which can be rewritten as

$$\xi = \begin{bmatrix} S(\omega) & v \\ 0 & 0 \end{bmatrix} \tag{7}$$

where $\omega \in \mathbf{R}^3$, $v \in \mathbf{R}^3$ and $S(\omega)$ is the skew symmetric matrix generated by ω. The vector

$$\dot\xi = \begin{pmatrix} \omega \\ v \end{pmatrix} \tag{8}$$

is referred to as the *twist coordinates* of ξ and represents the rotational and linear velocity of an object as viewed in world coordinates.

2.2 Fixed contact kinematics

Traditionally, a *fixed contact* between a finger and an object is described as a mapping between forces exerted by the finger at the point of contact and the resultant forces at some reference point on the object (e.g., the center of mass). We represent the force exerted at the i^{th} contact as $\bar F_{c_i} = (\bar f_{c_i}, \bar\tau_{c_i}) \in \mathbf{R}^6$ where $\bar f_{c_i}$ is the force exerted by contact and $\bar\tau_{c_i}$ is the moment. The relationship between contact force and object force has the form

$$F_o = \begin{pmatrix} f_o \\ \tau_o \end{pmatrix} = \begin{pmatrix} \bar f_{c_i} \\ \bar\tau_{c_i} + \boldsymbol{r}_{c_i} \times \bar f_{c_i} \end{pmatrix} = \begin{bmatrix} I & 0 \\ \boldsymbol{r}_{c_i}\times & I \end{bmatrix} \bar F_{c_i} \tag{9}$$

where $\boldsymbol{r}_{c_i} \in \mathbf{R}^3$ is the vector between the object reference point and the contact.

Typically, a finger will not be able to exert forces in every direction; several simple contact models are used to classify common contact configurations. A *point contact* is obtained when there is no friction between the fingertip and the object. In this case, forces can only be applied in the direction normal to the surface of the object and hence

$$\bar F_{c_i} = \begin{bmatrix} \boldsymbol{n}_{c_i}\boldsymbol{n}_{c_i}^T & 0 \end{bmatrix}\begin{pmatrix} f_{c_i} \\ \tau_{c_i} \end{pmatrix} \tag{10}$$

where $(f_{c_i}, \tau_{c_i}) \in \mathbf{R}^6$ is the force and moment applied by the finger and $\boldsymbol{n}_{c_i}$ is the unit vector normal to the object.

A *point contact with friction* model is used when friction exists between the fingertip and the object, in which case forces can be exerted in any direction that is within a cone of forces about the direction of the surface normal. This cone, called the *friction cone* is determined by the coefficient of friction. Figure 2b shows a point contact with friction and the resultant friction cone. This model assumes that moments cannot be applied (i.e., there is no torsional friction about the surface normal)

$$\bar F_{c_i} = \begin{bmatrix} I & 0 \end{bmatrix}\begin{pmatrix} f_{c_i} \\ \tau_{c_i} \end{pmatrix} \tag{11}$$

A more realistic contact model is the *soft finger* contact. Here we allow not only forces to be applied in a cone about the surface normal but also torques about that normal (see Figure 2c). These torques are limited by the torsional friction coefficient. Inside the relevant friction cones, this contact can be described as

$$\bar F_{c_i} = \begin{bmatrix} I & \boldsymbol{n}_{c_i}\boldsymbol{n}_{c_i}^T \end{bmatrix}\begin{pmatrix} f_{c_i} \\ \tau_{c_i} \end{pmatrix} \tag{12}$$

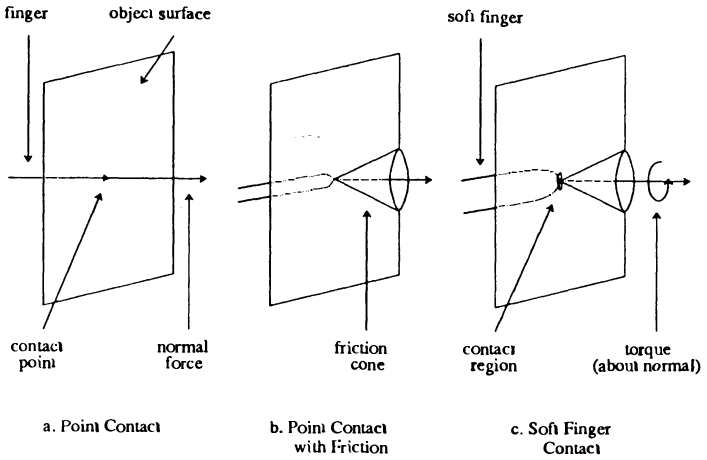


Figure 2: Contact types

Matrices mapping finger forces to contact force as in equations (10), (11) and (12) are referred to as *selection matrices* and we denote them by $B_i(x_o) \in \mathbf{R}^{6\times m_i}$, where m_i is the dimension of the range of forces and moments that can be applied for a given contact type. Note their dependence on the (fixed) contact point and the orientation of the object. Each of the contact types thus can be represented as a linear map $G_i(r_{c_i}, x_o)$: $F_{c_i} \in \mathbf{R}^{m_i} \mapsto F_o$

$$G_i(r_{c_i}, x_0) = \begin{bmatrix} I & 0 \\ r_{c_i}\times & I \end{bmatrix} B_i(x_0) \tag{13}$$

Since r_{c_i} is a function of the object orientation, we shall usually write $G_i(x_o) \in \mathbf{R}^{6\times m_i}$.

If we have several fingers contacting an object then the net force on the object is the sum of the forces due to each finger. The *grasp map*, $G: \mathbf{R}^m \to \mathbf{R}^6$, is the map between finger forces and the resultant total object force. Since each contact map is linear and forces can be superposed, we can add the individual contact maps to form G:

$$F_o = \begin{bmatrix} G_1 & \cdots & G_k \end{bmatrix} \begin{pmatrix} F_{c_1} \\ \vdots \\ F_{c_k} \end{pmatrix} = GF_c, \qquad \begin{array}{l} F_o \in \mathbf{R}^6 \\ F_c \in \mathbf{R}^{m_1} \times \mathbf{R}^{m_2} \times \ldots \times \mathbf{R}^{m_k} \end{array} \tag{14}$$

The null space of the grasp map corresponds to finger forces which cause no net force to be exerted on the object. We call the force on the object resulting from finger forces which lie in the null space of G, denoted $\mathcal{N}(G)$, *internal* or

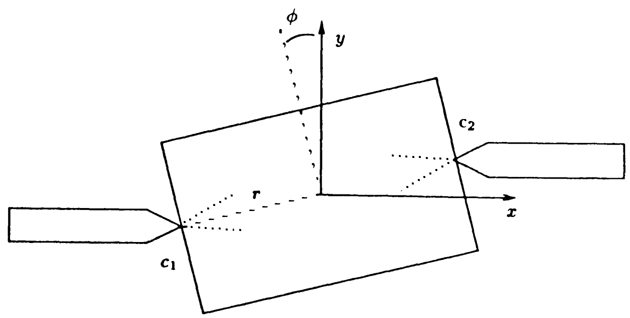


Figure 3: Planar two fingered grasp

null forces. It is in part these internal forces which allow us to grip or squeeze an object.

Dual to the representation of contacts as applied force and torque, one may also represent a contact as a constraint between the relative velocity of the object and the finger. Letting v_{c_i} and ω_{c_i} represent the linear and angular velocity of the contact point and v_o and ω_o represent the object velocity,

$$\begin{pmatrix} \tilde{v}_{c_i} \\ \tilde{\omega}_{c_i} \end{pmatrix} = \begin{bmatrix} I & r_{c_i}\times \\ 0 & I \end{bmatrix} \begin{pmatrix} v_o \\ \omega_o \end{pmatrix} \tag{15}$$

If we define v_c to be the velocities conjugate to f_c, the forces exerted by the fingers, it follows that

$$\begin{pmatrix} v_c \\ \omega_c \end{pmatrix} = G^T \begin{pmatrix} v_o \\ \omega_o \end{pmatrix} \tag{16}$$

This relationship between object velocity and finger velocities can also be derived in a more general setting using the principle of virtual work.

*

Example Consider a simple two-fingered planar hand as shown in Figure 3. Since we are in the plane, the grasp matrix maps finger forces into x and y forces, and a torque perpendicular to the xy plane. If we assume that the contacts are point contacts with friction,

$$G_i(x, y, \phi) = \begin{bmatrix} I & 0 \\ \begin{pmatrix} \pm r\cos\phi \\ \pm r\sin\phi \\ 0 \end{pmatrix}\times & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{17}$$

and the grasp map for Figure 3 is

$$G(x,y,\phi) = \underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ r\sin(\phi) & -r\cos(\phi) \end{bmatrix}}_{G_1}\underbrace{\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -r\sin(\phi) & r\cos(\phi) \end{bmatrix}}_{G_2} \tag{18}$$

where all forces are measured with respect to the xy coordinates shown in the figure.

Equation (18) shows that x and y forces from the fingers cause the same x and y forces to be exerted on the object as well as a torque that is dependent on the orientation of the object. The null space of this map is spanned by the vector

$$\begin{pmatrix} \cos\phi \\ \sin\phi \\ -\cos\phi \\ -\sin\phi \end{pmatrix} \tag{19}$$

which corresponds to forces applied along the line connecting the two fingertips. Finger forces applied along this line will cause no net force on the object.

2.3 Rolling contact kinematics

Most real world grasping situations involve moving rather than fixed contacts. Human fingers and many robotic fingers are actually surfaces and manipulation of an object by a set of fingers involves rolling of the fingers along the object surface. In this section we derive the kinematic equations for one object rolling against another.

Consider two objects, S_o and S_f in $\mathbf{R}^2$ which are touching at a point. We will restrict ourselves to the case where motion is contained in a single coordinate chart for each object. Let (c_o, U_o) and (c_f, U_f) be charts for the two surfaces and $\alpha_o = (u_o, v_o) \in U_o$ and $\alpha_f = (u_f, v_f) \in U_f$ be local coordinates. We will assume that c_o and c_f are orthogonal representations of the surface.[1] Furthermore, we let ψ represent the relative orientation of the tangent planes at the point of contact (see Figure 4). We call $\eta = (\alpha_o, \alpha_f, \psi)$ the *contact coordinates*.

Let $g \in SE(3)$ describe the relative position and orientation of S_f with respect to S_o. We wish to study the relationship between g and the local contact coordinates. To do so we assume that $g \in W \subset SE(3)$ where W is the set of all relative positions for which the two objects remain in contact.

We begin by writing the algebraic equations that η must satisfy. At any point of contact the location of the contact in space must agree for both objects

$$g \circ c_f(\alpha_f) = c_o(\alpha_o) \tag{20}$$

Furthermore, the tangent planes must coincide and hence the outward surface normals $\boldsymbol{n}_o : S_o \to S^2 \subset \mathbf{R}^3$ and $\boldsymbol{n}_f : S_f \to S^2 \subset \mathbf{R}^3$ must also agree. Letting

[1] A surface representation $c : (u,v) \to \mathbf{R}^3$ is orthogonal if $\frac{\partial c}{\partial u}$ and $\frac{\partial c}{\partial v}$ are orthogonal. Such a representation can always be constructed for a regular surface in a given coordinate chart.

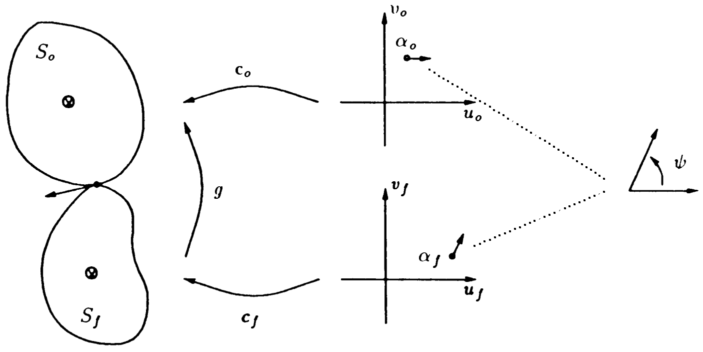


Figure 4: Parameterization of rolling contacts

$R \in SO(3)$ be the rotational component of g

$$R\boldsymbol{n}_f(\alpha_f) = -\boldsymbol{n}_o(\alpha_o) \tag{21}$$

Finally, we define the angle between the tangent planes as the unique angle $\psi \in [0, 2\pi)$ such that

$$R\frac{\partial c_f}{\partial \alpha_f} M_f^{-1} R_\psi = \frac{\partial c_o}{\partial \alpha_o} M_o^{-1} \tag{22}$$

where

$$M_i = \begin{bmatrix} \|\frac{\partial c_i}{\partial u_i}\| & 0 \\ 0 & \|\frac{\partial c_i}{\partial v_i}\| \end{bmatrix} \tag{23}$$

insures that the columns of $\frac{\partial c}{\partial \alpha}$ are unit length and

$$R_\psi = \begin{bmatrix} \cos\psi & -\sin\psi \\ -\sin\psi & -\cos\psi \end{bmatrix} \tag{24}$$

converts α_o coordinates to the equivalent α_f coordinates at the point of contact. Since the normals are in opposite directions, R_ψ acts by negating the y coordinate and rotating by an angle ψ. Note that $R_\psi = R_\psi^T = R_\psi^{-1}$.

Claim 2.1 *There is a smooth bijection between η and g.*

Proof Functionally, equations (20) through (22) are of the form $h(g,\eta) = 0$. It is therefore sufficient (and necessary) to show that $\frac{\partial h}{\partial \eta}$ spans the allowable velocity space, TW. Since ψ can be defined directly, we omit the ψ coordinate and consider the dependence on $\alpha = (\alpha_o, \alpha_f)$,

$$h(g,\alpha) = \begin{pmatrix} c_o(\alpha_o) - gc_f(\alpha_f) \\ n_o(\alpha_o) + Rn_f(\alpha_f) \end{pmatrix} \tag{25}$$

$$\frac{\partial h}{\partial \alpha}(g,\alpha) = \begin{pmatrix} \frac{\partial c_o(\alpha_o)}{\partial \alpha_o} & -R\frac{\partial c_f(\alpha_f)}{\partial \alpha_f} \\ \frac{\partial n_o(\alpha_o)}{\partial \alpha_o} & R\frac{\partial n_f(\alpha_f)}{\partial \alpha_f} \end{pmatrix} \tag{26}$$

First we show that the span of the rows of $\frac{\partial h}{\partial \alpha}$ does not contain either $(n_o, 0)$ or $(0, n_o)$, corresponding to translation and rotation about n_o. $(0, n_o)$ is spanned directly by $d\psi$ and $(n_o, 0)$ should not belong to the row span of $\frac{\partial h}{\partial \alpha}$ since motion in the n_o direction is not contained in TW. Since the range of $\frac{\partial c_o}{\partial \alpha_o}$ and $\frac{\partial c_f}{\partial \alpha_f}$ define the tangent plane and the n_i's have unit magnitude and using equation (21), we find

$$\begin{pmatrix} n_o \\ 0 \end{pmatrix}^T \frac{\partial h}{\partial \alpha} = \begin{bmatrix} n_o^T \frac{\partial c_o}{\partial \alpha_o} & n_f^T R^T R \frac{\partial c_f}{\partial \alpha_f} \end{bmatrix} = 0 \tag{27}$$

$$\begin{pmatrix} 0 \\ n_o \end{pmatrix}^T \frac{\partial h}{\partial \alpha} = \begin{bmatrix} n_o^T \frac{\partial n_o}{\partial \alpha_o} & n_f^T R^T R \frac{\partial n_f}{\partial \alpha_f} \end{bmatrix} = 0 \tag{28}$$

Next we examine the conditions under which $\frac{\partial h}{\partial \alpha}$ loses rank. Plugging equation (22) into equation (26), $\frac{\partial h}{\partial \alpha}$ can only lose rank when $\alpha_f = M_f^{-1} R_\psi M_o \alpha_o$, so $\frac{\partial h}{\partial \alpha}$ is full rank if and only if

$$\frac{\partial n_o}{\partial \alpha_o} M_o^{-1} + R \frac{\partial n_f}{\partial \alpha_f} M_f^{-1} R_\psi \tag{29}$$

is full rank. □

Proceeding along the lines of the proof given above, the differential relationship between η and g can be derived (see appendix at end of this chapter). It is convenient to make use of the *normalized gauss frame* defined on each surface

$$\begin{bmatrix} x_i & y_i & z_i \end{bmatrix} = \begin{bmatrix} \frac{\partial c_i}{\partial \alpha} M_i^{-1} & n_i \end{bmatrix} \tag{30}$$

If we do not allow the fingers to slide on the object (soft finger contacts) then the motion of the contacts, $\dot\eta$, as a function of the relative motion, (ω, v), is given by

$$\dot\alpha_o = M_o^{-1}(K_o + \tilde K_f)^{-1}\omega_t \tag{31}$$

$$\dot\alpha_f = M_f^{-1} R_\psi (K_o + \tilde K_f)^{-1}\omega_t \tag{32}$$

$$\dot\psi = T_o M_o^{-1}\dot\alpha_o + T_f M_f^{-1}\dot\alpha_f \tag{33}$$

where

$$\omega_t = \begin{bmatrix} x_o^T \\ y_o^T \end{bmatrix} n_o \times \omega \tag{34}$$

$$K_o = \begin{bmatrix} x_o^T \\ y_o^T \end{bmatrix} \frac{\partial n_o}{\partial \alpha_o} M_o^{-1} \tag{35}$$

$$\tilde K_f = R_\psi \begin{bmatrix} x_f^T \\ y_f^T \end{bmatrix} \frac{\partial n_f}{\partial \alpha_f} M_f^{-1} R_\psi \tag{36}$$

$$T_o = y_o^T \frac{\partial x_o}{\partial \alpha_o} M_o^{-1} \tag{37}$$

$$T_f = y_f^T \frac{\partial x_f}{\partial \alpha_f} M_f^{-1} \tag{38}$$

$(K_o + \tilde K_f)$ is called the *relative curvature*. From equations (22) and (29) we see that the relative curvature is invertible precisely when $\frac{\partial h}{\partial \eta}$ is onto TW.

We can now describe the kinematics for rolling contact—the relationship between the object velocities and a set of finger velocities. This situation is identical to that given for fixed contacts except that the vector r_{c_i} between the object reference frame and the i^{th} contact point is now a function of η as well as the object orientation. But η is a continuous function of $g = x_o^{-1} x_{f_i}$ so we have

$$F_o = G(x_o, x_f) F_c \tag{39}$$

where $x_f = (x_{f_1}, \cdots, x_{f_k})$ is the position and orientation of the fingers and $F_c \in \mathbf{R}^{n_f} \times \cdots \times \mathbf{R}^{n_k}$ is the force exerted by the fingers at the contact point. As before, G is composed of matrices of the form

$$G_i(x_o, \theta) = \begin{bmatrix} I & 0 \\ r_{c_i}\times & I \end{bmatrix} B_i(x_o, \theta) \tag{40}$$

The velocity relationship can again be derived from the principle of virtual work or algebraically to determine

$$\begin{pmatrix} v_c \\ \omega_c \end{pmatrix} = G^T(x_o, x_f) \begin{pmatrix} v_o \\ \omega_o \end{pmatrix} \tag{41}$$

2.4 Finger Kinematics

Up to this point we have assumed that our fingers are points or surfaces in space. In fact, we are more interested in considering fingers which are kinematic mechanisms. For each finger i we associate a forward kinematic map $K_{f_i} : \mathbf{R}^{n_i} \to SE(3)$ which takes joint position to end effector position and orientation. The Jacobian of the forward kinematic map relates joint velocities to the end effector velocities,

$$\begin{pmatrix} \omega_{f_i} \\ v_{f_i} \end{pmatrix} = \frac{\partial K_{f_i}}{\partial \theta_{f_i}} \frac{d\theta}{dt} K_{f_i}^{-1} = J_{f_i}(\theta_{f_i})\dot\theta_{f_i}, \qquad \theta_{f_i} \in \mathbf{R}^{n_i}, (\omega_{f_i}, v_{f_i}) \in \mathbf{R}^6 \tag{42}$$

Combining this with the velocity transformation between the object location and the contact location (a function of $x_o^{-1} x_f$) we write the *contact Jacobian* as

$$J_{c_i}(x_o, \theta_{f_i}) = \begin{pmatrix} I & r_{c_i}\times \\ 0 & I \end{pmatrix} J_{f_i} \qquad J_{c_i} : \dot\theta \mapsto \begin{pmatrix} \omega_{c_i} \\ v_{c_i} \end{pmatrix} \tag{43}$$

As with the fixed contacts, fingers are only allowed to exert forces in certain directions depending on the contact type. This is equivalent to saying that finger motions are only constrained in certain directions; these directions are given by the column span of $B_i^T(x_o, \theta_{f_i}): \mathbf{R}^{m_i} \to \mathbf{R}^6$ (where B_i is the selection matrix defined in section). Combining this with the grasp map for the i^{th} finger, we obtain the velocity constraint due to the i^{th} contact,

$$G_i^T(x_o, \theta_{f_i})\begin{pmatrix} v_o \\ \omega_o \end{pmatrix} = B_i^T(x_o, \theta_{f_i})J_{c_i}(x_o, \theta_{f_i})\dot{\theta}_{f_i} \tag{44}$$

We now stack these matrices and write the grasp constraint for the hand as

$$\begin{bmatrix} G_1 \\ \vdots \\ G_k \end{bmatrix}\begin{pmatrix} \omega_o \\ v_o \end{pmatrix} = \begin{bmatrix} B_1^T J_{c_1} & & 0 \\ & \ddots & \\ 0 & & B_k^T J_{c_k} \end{bmatrix}\dot{\theta} \tag{45}$$

$$G(x_o, \theta)\begin{pmatrix} \omega_o \\ v_o \end{pmatrix} = J(x_o, \theta)\dot{\theta} \tag{46}$$

2.5 Grasp stability and manipulability

One measure of the stability of a grasp is the range of forces that can be exerted by the hand on the object. The fingers can only apply unidirectional forces and all forces must lie within the friction cone for the contact. We say a grasp on an object satisfies *force closure* if we can exert, through a set of contacts, arbitrary forces and torques on the object [14]. More formally, we define the set FC as

$$FC = \left\{ f_c \in \mathbf{R}^n : \|f_{c_{ij}}^t\| \le \mu_{ij}\|f_{c_i}^n\|, \quad i = 1, \ldots, k, \quad j = 1, \ldots, n_i \right\} \tag{47}$$

where $f_{c_{ij}}^t$ is the tangent component of the j^{th} element of f_{c_i}, $f_{c_i}^n$ is the normal force for the i^{th} contact, and μ_{ij} is the coefficient of friction corresponding to $f_{c_{ij}}$. For soft finger contacts, the torques exerted by the fingers also satisfy equation 47 with $f_{c_{ij}}^t$ replaced by the torque (i.e., we do not want to apply a torque which is greater than the torsional friction coefficient multiplied by the magnitude of the normal force). The force closure condition can now be stated as

$$\text{force closure} \Leftrightarrow G(FC) = \mathbf{R}^6 \tag{48}$$

which says that the range of the grasp map over forces lying in the friction cone covers $\mathbf{R}^6$ (the space of forces and torques applied to the object). One property of a force closure grasp is that G must have full row rank. If this were not true then there would be some object force which could not be produced by the fingers (a contradiction).

A grasp is said to be *manipulable* if arbitrary motions of the objects can be accommodated by the fingers.

It is also useful to define the concept of prehensility. A grasp is *prehensile* if there exists a force contained in the null space of the grasp map which also lies in the *interior* of the friction cone. More formally,

$$\text{prehensility} \Leftrightarrow \mathcal{N}(G) \cap \mathring{FC} \neq \emptyset \tag{49}$$

where $\mathring{FC}$ is the set of forces lying completely within the friction cone (i.e., $\|f_{c_{ij}}^t\| < \mu_{ij}\|f_{c_i}^n\|$). We shall require this property in order to insure that our controllers can maintain a grip on an object while manipulating it.

2.6 Summary

In this chapter we have derived the kinematic equations for fixed and rolling contacts. The fundamental kinematic relationships are the grasp kinematics

$$G^T(x_o, \theta)\dot{x}_o = J(x_o, \theta)\dot{\theta} \tag{50}$$

and, for rolling contacts, the contact kinematics

$$\begin{aligned} \dot{\alpha}_o &= (K_o + \tilde{K}_f)^{-1}\omega_t \\ \dot{\alpha}_f &= R_\psi(K_o + \tilde{K}_f)^{-1}\omega_t \\ \dot{\psi} &= T_o\dot{\alpha}_o + T_f\dot{\alpha}_f \end{aligned} \tag{51}$$

Such a grasp is said to be stable when finger forces lying in the friction cone span the space of object forces

$$\mathbf{R}^6 \subseteq G(FC) \tag{52}$$

manipulable when arbitrary motions can be generated by the fingers

$$\mathcal{R}(G^T) \subseteq \mathcal{R}(J) \tag{53}$$

and prehensile when

$$\mathcal{N}(G) \cap \mathring{FC} \neq \emptyset \tag{54}$$

Appendix – Contact kinematics derivation

In this appendix we derive the kinematics of contact for two objects touching each other at a point. The notation is described more fully in section . A similar derivation can be found in a recent paper by Montana [12].

To derive the kinematics, we begin with constraint equations given by equating the points of contact, normals of contact and tangent planes at the contact points:

$$Rc_f(\alpha_f) + p = c_o(\alpha_o) \tag{55}$$

$$Rn_f(\alpha_f) = -n_o(\alpha_o) \tag{56}$$

$$R\frac{\partial c_f}{\partial \alpha_f}M_f^{-1}R_\psi = \frac{\partial c_o}{\partial \alpha_o}M_o^{-1} \tag{57}$$

Differentiate (55) and (56)

$$\dot{R}c_f + R\frac{\partial c_f}{\partial \alpha_f}\dot{\alpha}_f + \dot{p} = \frac{\partial c_o}{\partial \alpha_o}\dot{\alpha}_o \tag{58}$$

$$\dot{R}n_f + R\frac{\partial n_f}{\partial \alpha_f}\dot{\alpha}_f = -\frac{\partial n_o}{\partial \alpha_o}\dot{\alpha}_o \tag{59}$$

Multiply (58) by $\frac{\partial c_o}{\partial \alpha_o}^T$ and substitute $\dot\alpha_o$ into (59)

$$\dot{R} n_f + R\frac{\partial n_f}{\partial \alpha_f}\dot\alpha_f = -\frac{\partial n_o}{\partial \alpha_o} M_o^{-2} \frac{\partial c_o}{\partial \alpha_o}^T \left(\dot{R} c_f + R\frac{\partial c_f}{\partial \alpha_f}\dot\alpha_f + \dot p\right) \tag{60}$$

Using (57) in the last term of (60) and rearranging

$$\begin{aligned}\left(R\frac{\partial n_f}{\partial \alpha_f} + \frac{\partial n_o}{\partial \alpha_o} M_o^{-2}\left(\frac{\partial c_o}{\partial \alpha_o}^T \frac{\partial c_o}{\partial \alpha_o}\right) M_o^{-1} R_\psi M_f\right)\dot\alpha_f \\ = -\dot{R} n_f - \frac{\partial n_o}{\partial \alpha_o} M_o^{-2} \frac{\partial c_o}{\partial \alpha_o}^T (\dot{R} c_f + \dot p)\end{aligned} \tag{61}$$

Simplify the first term and multiply by $M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T$ on the left

$$\begin{aligned} & M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T \left(R\frac{\partial n_f}{\partial \alpha_f} + \frac{\partial n_o}{\partial \alpha_o} M_o^{-1} R_\psi M_f\right) & (62)\\ = {} & M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T \left(R\frac{\partial n_f}{\partial \alpha_f} M_f^{-1} R_\psi + \frac{\partial n_o}{\partial \alpha_o} M_o^{-1}\right) R_\psi M_f & (63)\\ = {} & \left(\underbrace{R_\psi M_f^{-T}\frac{\partial c_f}{\partial \alpha_f}\frac{\partial n_f}{\partial \alpha_f} M_f^{-1} R_\psi}_{\tilde K_f} + \underbrace{M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T \frac{\partial n_o}{\partial \alpha_o} M_o^{-1}}_{K_o}\right) R_\psi M_f & (64)\end{aligned}$$

Multiply both sides of (61) by $M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T$ and use the previous calculation

$$\begin{aligned}\dot\alpha_f &= M_f^{-1} R_\psi \left(\tilde K_f + K_o\right)^{-1} \cdot & (65)\\ & \quad M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T\left(-\dot R n_f - \frac{\partial n_o}{\partial \alpha_o} M_o^{-2}\frac{\partial c_o}{\partial \alpha_o}^T (\dot R c_f + \dot p)\right) & (66)\\ &= M_f^{-1} R_\psi \left(\tilde K_f + K_o\right)^{-1} \cdot & (67)\\ & \quad \left(-M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T \dot R n_f - K_o M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T (\dot R c_f + \dot p)\right) & (68)\end{aligned}$$

Let ω_t stand for the $\dot R n_f$ term and v_t represent the $\dot R c_f + \dot p$ term:

$$\dot\alpha_f = M_f^{-1} R_\psi \left(\tilde K_f + K_o\right)^{-1} (w_t - K_o v_t) \tag{69}$$

Now w_t and v_t can now be calculated in terms of the relative velocity given by $(S(\omega), v) = \dot g g^{-1}$. We use the fact that $S(\omega)a = \omega \times a$ and $\omega \times a = -a \times \omega$ to obtain

$$w_t = -M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T (\omega \times (R n_f)) = -M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T (n_o \times \omega) \tag{70}$$

$$\begin{aligned} v_t &= M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T (\omega \times (R c_f) + \omega \times p + v) & (71)\\ &= M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T (\omega \times (c_o - p) + \omega \times p + v) & (72)\\ &= M_o^{-T}\frac{\partial c_o}{\partial \alpha_o}^T (-c_o \times \omega + v) & (73)\end{aligned}$$

We see that ω_t is the relative rotational velocity projected onto the tangent plane at the contact. It includes only terms due to *rolling* since rotation normal to the surface is annihilated by taking the cross product with n_o. Likewise, v_t is the relative linear velocity between the contacts, projected onto the tangent plane, i.e., the *sliding* velocity.

A similar calculation yields

$$\dot\alpha_o = M_o^{-1}\left(\tilde K_f + K_o\right)^{-1}\left(w_t - \tilde K_f v_t\right) \tag{74}$$

Next we solve for ψ, the angle between the tangent planes of the finger and object. Combining (56) and (57) we can write

$$R\begin{bmatrix} \frac{\partial c_f}{\partial \alpha_f} M_f^{-1} & n_f \end{bmatrix}\begin{bmatrix} R_\psi & 0 \\ 0 & -1\end{bmatrix} = \begin{bmatrix} \frac{\partial c_o}{\partial \alpha_o} M_o^{-1} & n_o \end{bmatrix} \tag{75}$$

and using the normalized gaussian coordinates this can be rewritten

$$R[x_f\ y_f\ z_f]\bar R_\psi = [x_o\ y_o\ z_o] \tag{76}$$

Take the derivative of (76)

$$\dot R[x_f\, y_f\, z_f]\bar R_\psi + R[\dot x_f\, \dot y_f\, \dot z_f]\bar R_\psi + R[x_f\, y_f\, z_f]\begin{bmatrix}\dot R_\psi & 0\\ 0 & 0\end{bmatrix} = [\dot x_o\, \dot y_o\, \dot z_o] \tag{77}$$

Premultiply by $y_f^T R^T$

$$y_f^T R^T \dot R[x_f\, y_f\, z_f]\bar R_\psi + y_f^T[\dot x_f\, \dot y_f\, \dot z_f]\bar R_\psi + (0\,1\,0)\begin{bmatrix}\dot R_\psi & 0\\ 0 & 0\end{bmatrix} = [\dot x_o\, \dot y_o\, \dot z_o] \tag{78}$$

Postmultiply by $\bar R_\psi\begin{pmatrix}1\\0\\0\end{pmatrix}$ and note $\bar R_\psi \bar R_\psi = I$

$$y_f^T R^T \dot R x_f + y_f^T \dot x_f + (0\,1\,0)\begin{bmatrix}\dot R_\psi R_\psi & 0\\ 0 & 0\end{bmatrix}\begin{pmatrix}1\\0\\0\end{pmatrix} = y_f^T R^T[\dot x_o\, \dot y_o\, \dot z_o]\bar R_\psi\begin{pmatrix}1\\0\\0\end{pmatrix} \tag{79}$$

$$y_f^T R^T \dot R x_f + y_f^T \dot x_f + (0\,1)\begin{bmatrix}0 & \dot\psi\\ -\dot\psi & 0\end{bmatrix}\begin{pmatrix}1\\0\end{pmatrix} = y_f^T R^T[\dot x_o\, \dot y_o\, \dot z_o]\bar R_\psi\begin{pmatrix}1\\0\\0\end{pmatrix} \tag{80}$$

$$y_f^T R^T \dot R x_f + y_f^T \dot x_f - \dot\psi = y_f^T R^T[\dot x_o\, \dot y_o\, \dot z_o]\bar R_\psi\begin{pmatrix}1\\0\\0\end{pmatrix} \tag{81}$$

From (76) we see that $y_f^T R^T = (0\,1\,0)\bar{R}_\psi \begin{bmatrix} x_o^T \\ y_o^T \\ z_o^T \end{bmatrix} = (0\,1)R_\psi \begin{bmatrix} x_o^T \\ y_o^T \end{bmatrix}$ and so

$$\dot{\psi} = y_f^T R^T \dot{R} x_f + y_f \frac{\partial x_f}{\partial \alpha_f}\dot{\alpha}_f - (0\,1)R_\psi \begin{bmatrix} x_o^T \dot{x}_o & x_o^T \dot{y}_o \\ y_o^T \dot{x}_o & y_o^T \dot{y}_o \end{bmatrix} R_\psi \binom{1}{0} \tag{82}$$

Using the following identities

$$\begin{aligned} x_i^T y_i = 0 \quad &\Rightarrow \quad \dot{x}_i^T y_i = -x_i^T \dot{y}_i = y_i^T \dot{x}_i \\ x_i^T x_i = 1 \quad &\Rightarrow \quad \dot{x}_i^T x_i = 0 \end{aligned} \tag{83}$$

(82) can be written as

$$\dot{\psi} = y_f^T R^T \dot{R} x_f + y_o^T \frac{\partial x_o}{\partial \alpha_o}\dot{\alpha}_o + y_f^T \frac{\partial x_f}{\partial \alpha_f}\dot{\alpha}_f \tag{84}$$

$$= \omega_n + T_o M_o \dot{\alpha}_o + T_f M_f \dot{\alpha}_f \tag{85}$$

where

$$\omega_n = y_f^T R^T \dot{R} x_f = (R y_f)^T \omega \times (R x_f) \tag{86}$$

$$= (R z_f)^T \omega = z_o^T \omega \tag{87}$$

where the last equality follows from the vector formula $a \cdot b \times c = b \cdot c \times a$. This last equation shows that ω_n is just the relative rotational velocity projected onto the surface normal.

Collecting equations (69), (74) and (85) we have

$$\dot{\alpha}_o = (K_o + \tilde{K}_f)^{-1}\left(\omega_t - \tilde{K}_f v_t\right) \tag{88}$$

$$\dot{\alpha}_f = R_\psi (K_o + \tilde{K}_f)^{-1}(\omega_t - K_o v_t) \tag{89}$$

$$\dot{\psi} = \omega_n + T_o \dot{\alpha}_o + T_f \dot{\alpha}_f \tag{90}$$

The matrix $K_o + \tilde{K}_f$ is called the *relative curvature* by Montana[12].

3 Dynamics and control

In this section we review some basic results in dynamics of robot systems. The primary result which we present is that even for relatively complicated robot systems, the equations of motion for the system can be written in a standard form. This point of view has been used by Khatib in his operational space formulation [7] and in some recent extensions [6]. The results presented in this section are direct extensions of those works, although the approach is different.

3.1 Robot dynamics

We begin by deriving the robot dynamics for a manipulator in joint space. Let $\theta \in \mathbf{R}^n$ be the joint angles for the manipulator and $\tau \in \mathbf{R}^n$ be the corresponding joint torques. The Lagrangean for the system may be shown to be of the form

$$L = M(\theta)(\dot{\theta}, \dot{\theta}) + V(\theta) \tag{91}$$

where $M(\theta)$ is the inertia matrix for the manipulator and V is the potential energy due to gravity. Substituting into Lagrange's equations

$$\left(\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} - \tau\right)\delta\theta = 0 \tag{92}$$

and letting τ represent the actuator torques (and other non-conservative forces), we obtain

$$M(\theta)(\ddot{\theta}, \cdot) + DM(\theta)(\dot{\theta}, \cdot)(\dot{\theta}) - \frac{1}{2}DM(\theta)(\dot{\theta}, \dot{\theta})(\cdot) + DV(\theta)(\cdot) = \tau \tag{93}$$

To put this in a more conventional form we define the matrix $C(\theta, \dot{\theta})$ as

$$a^T C(\theta, \dot{\theta}) b = \frac{1}{2}DM(\theta)(\dot{\theta}, a)(b) + \frac{1}{2}DM(\theta)(b, a)(\dot{\theta}) - \frac{1}{2}DM(\theta)(\dot{\theta}, b)(a) \tag{94}$$

and write

$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + N(\theta, \dot{\theta}) = \tau \tag{95}$$

where $N(\theta, \dot{\theta})$ includes gravity terms and other forces (such as friction) which act at the joints.

For systems of this type, the inertia matrix is always symmetric and positive definite and it can be shown that $\dot{M} - 2C$ is skew symmetric (using this particular choice of C). It is both the form and the structure of this equation that we will attempt to maintain in more complicated systems.

3.2 Robot hand dynamics

We now examine the dynamics of a set of fingers actuated at each joint connected through a set of contacts to a rigid body. The finger dynamics can be written as

$$M_f(\theta)\ddot{\theta} + C_f(\theta, \dot{\theta})\dot{\theta} + N_f(\theta, \dot{\theta}) = \tau \tag{96}$$

where $\theta \in \mathbf{R}^{n_1} \times \cdots \times \mathbf{R}^{n_k}$ is now the set of joint angles for *all* of the robots and τ is the corresponding set of torques. The object dynamics are given by the Newton-Euler equations

$$\begin{bmatrix} \mathcal{I}_o & 0 \\ 0 & m_o I \end{bmatrix}\begin{pmatrix} \dot{\omega}_o \\ \dot{v}_o \end{pmatrix} + \begin{pmatrix} \omega_o \times \mathcal{I}_o \omega_o \\ 0 \end{pmatrix} + N_o(x_o, \dot{x}_o) = \begin{pmatrix} \tau_o \\ f_o \end{pmatrix} \tag{97}$$

where $\mathcal{I}_o = R\mathcal{I}R^T$ is the object inertia in world coordinates. In local coordinates this has the same basic form as the robot dynamics, lacking only the actuator torques:

$$M_o(x)\ddot{x} + C_o(x_0, \dot{x}_0)\dot{x} + N_o(x_0, \dot{x}) = 0 \tag{98}$$

where x is a local parameterization of $x_o \in SE(3)$. We attach these two systems with a set of constraints

$$G(x,\theta)\dot{x} = J(x,\theta)\dot{\theta} \tag{99}$$

which represents the grasp. We will assume that the grasp is both stable and manipulable. For the moment we will also require J to be injective.

This velocity constraint generates a constraint on the virtual displacements $\delta\theta$ and δx, namely $\delta\theta = J^{-1}(q)G^T(q)\delta x$ with $q = (x, \theta)$. Using this relationship, we can write Lagrange's equations as

$$\left(\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} - (\tau, 0)\right)\delta q = 0 \tag{100}$$

$$\begin{pmatrix} \frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} - \tau \\ \frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} \end{pmatrix} \begin{pmatrix} \delta\theta \\ \delta x \end{pmatrix} = 0 \tag{101}$$

$$\left(\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} - \tau\right)\delta\theta + \left(\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x}\right)\delta x = 0 \tag{102}$$

$$GJ^{-T}\left(\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta} - \tau\right)\delta x + \left(\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x}\right)\delta x = 0 \tag{103}$$

and since δx is arbitrary

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{x}} - \frac{\partial L}{\partial x} + GJ^{-T}\left(\frac{d}{dt}\frac{\partial L}{\partial \dot{\theta}} - \frac{\partial L}{\partial \theta}\right) = GJ^{-T}\tau \tag{104}$$

This equation together with the velocity constraint given in equation 99 describes the system completely. Note that equation 104 is a vector differential equation with $n - m$ rows and equation 99 is a vector equation with m rows.

It is tempting to derive equation 104 by using the velocity constraint directly in the kinetic energy equation (which is a function of $\dot{\theta}$ and $\dot{x}$) and then substituting this into Lagrange's equations. As noted in Rosenberg [15] this can only be done if the constraint is holonomic, i.e., θ can be written as a function of x.

Next we separate the kinetic energy into an object portion and a robot portion

$$T = \dot{\theta}^T M_f(\theta)\dot{\theta} + \dot{x}^T M_o(x)\dot{x} \tag{105}$$

Using equation 104 we find

$$\tilde{M}(q)\ddot{x} + \tilde{C}(q,\dot{q})\dot{x} = GJ^{-T}\tau \tag{106}$$

where

$$\begin{aligned} \tilde{M} &= M_o + GJ^{-T}M_fJ^{-1}G^T \\ \tilde{C} &= C_o + GJ^{-T}\left(C_fJ^{-1}G^T + M_f\frac{d}{dt}(J^{-1}G^T)\right) \end{aligned}$$

and C_f and C_o are obtained from equation 94 by replacing M with M_f and M_o respectively. It can be shown that the matrix $\dot{\tilde{M}} - 2\tilde{C}$ is still skew symmetric.

Note that in general G is not square and by examining the right side of the equations of motion (106) we note that if $J^{-T}\tau \in \mathcal{N}(G)$ then the net force in the object frame of reference is zero and hence forces of this form cause no net motion on the object. These forces are in fact the forces which act against the constraint and are generally termed *internal* or *constraint* forces. We can use these internal forces to satisfy other conditions, such as keeping the contact forces inside the friction code (to avoid slipping) or varying the load distribution of a set of manipulators rigidly grasping an object.

Thus we have an equation with form (and structure) similar to our "simple" robot. In the box frame of reference, $\tilde{M}$ is the effective mass of the box, and $\tilde{C}$ is the effective Coriolis and centrifugal matrix. These matrices include the dynamics of the fingers, which are being used to actually control the motion of the box. However the details of the finger kinematics and dynamics are effectively hidden in the definition of $\tilde{M}$ and $\tilde{C}$.

This simple result has some interesting consequences in control. Typically robot controllers are designed by placing a feedback loop around the joint positions (and velocities) of the robot. The controller generates torques which attempt to make the robot follow a prescribed joint trajectory. However, since the robot dynamics are of the same form in either joint or Cartesian space, we can just as easily write the control algorithm in Cartesian coordinates. In this case, we must take the output force from the controller and transform it back into joint torques, by premultiplying it by J^T. One advantage of this approach is that controller objectives are often specified in Cartesian space and hence it might be easier to perform the controller design and analysis in that space.

We note that even though we will write our controllers in terms of F, it is actually the joint torques which we are able to specify. Given the desired force in constrained coordinates, we can apply that force using an actuator force of $J^TG^+\tau$, where J and G are as defined previously and G^+ is a pseudo inverse for G.

Redundant manipulators

To study this motion, we first choose a basis for the redundant velocities. Let $K(\theta) \in \mathbf{R}^{(n-m)\times n}$ be a matrix that combines with J to make a matrix which is full rank for every θ[2]. Then we can write our constraint as

$$\begin{bmatrix} J \\ K \end{bmatrix}\dot{\theta} = \begin{bmatrix} G^T & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} \tag{107}$$

Since $\bar{J} = \begin{bmatrix} J \\ K \end{bmatrix}$ is invertible, we can write

$$T = (\dot{x}^T\,\dot{y}^T)\left(\bar{J}^{-T}M_f\bar{J}^{-1} + \begin{bmatrix} M(x) & 0 \\ 0 & 0 \end{bmatrix}\right)\begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \tag{108}$$

[2] This is easily done using the LU decomposition, see [18]

and it follows from our previous derivation that

$$\tilde{M}(\theta)\begin{pmatrix} \ddot{x}_o \\ \ddot{y} \end{pmatrix} + \tilde{C}(\theta,\dot{\theta})\begin{pmatrix} x \\ \dot{y} \end{pmatrix} = G\begin{bmatrix} J \\ K \end{bmatrix}^{-} T\tau \qquad (109)$$

3.3 Control

The grasping control problem can be broken into two parts

1. Tracking – the center of mass of the object should follow a specified trajectory.

2. Holding – the finger forces should lie within the friction cone at all times.

Condition 2 is important not only because we do not wish to lose our grip on the object, but also because we assumed in our derivation of the grasp dynamics that contact was maintained. Without this constraint we would have to specify the dynamics of contact.

If a grasp is prehensile it can be shown that given an arbitrary set of finger forces, F_c, we can find an internal force, $F_N \in \mathcal{N}(G)$, such that the combined force $F_c + F_N$ is inside the friction cone. Thus, given a force generated to solve the tracking problem, we can always add a force to this such that condition 2 is satisfied. Since internal forces cause no net motion of the hand or object, this additional force does not affect the net force exerted by the fingers on the object. We shall assume in the sequel that such an internal force is available at all times. The choice of this force is discussed in more detail below.

To illustrate the control of robot systems, we look at two controllers which have appeared in the robotics literature. We consider only grasp which are stable, manipulable and prehensile. We start by considering systems of the form

$$M(q)\ddot{x}_o + C(q,\dot{q})x + N(q,\dot{q}) = F \qquad (110)$$

where $M(q)$ is a positive definite inertia matrix and $C(q,\dot{q})x$ is the Coriolis and centrifugal force vector. The vector $N(q,\dot{q}) \in \mathbf{R}^n$ contains all friction and gravity terms and the vector $F \in \mathbf{R}^n$ represents generalized forces in the object coordinate frame.

Computed torque

Computed torque is an exactly linearizing control law (i.e., the dynamics are rendered linear by state feedback) that has been used extensively in robotics research. It has been used for joint level control [1], Cartesian control [11], and most recently, control of multi-fingered hands [10, 4]. Given a desired trajectory x_d we use the control

$$\begin{aligned} F &= M(q)\left(\ddot{x}_d + K_v\dot{e} + K_p e\right) + C(q,\dot{q})x + N(q,\dot{q}) && (111) \\ \tau &= J^T G^+ F + J^T F_N && (112) \end{aligned}$$

where error $e = x_d - x$ and K_v and K_p are constant gain matrices. The resulting dynamics equations are linear with exponential rate of convergence determined

by K_v and K_p. Since the system is linear, we can use linear control theory to choose the gains (K_v and K_p) such that they satisfy some set of design criteria.

The disadvantage of this control law is that it is not easy to specify the interaction with the environment. From the form of the error equation we might think that we could use K_p to model the stiffness of the system and exert forces by commanding trajectories which result in fixed errors. Unfortunately this is not uniformly applicable as can be seen by examining the force due to a quasi-static displacement Δx:

$$\Delta F = M(q)K_p\Delta x \qquad (113)$$

Since K_p must be constant in order to prove stability, the resultant stiffness will vary with configuration. Additionally, given a desired stiffness matrix it may not be possible to find a positive definite K_p that achieves that stiffness.

'PD' control

PD controllers differ from computed torque controllers in that the desired stiffness (and potentially damping) of the end effector is specified, rather than its position tracking characteristics. Typically, control laws of this form rely on the skew-symmetric property of robot dynamics, namely $\alpha^T\left(\dot{M} - 2C\right)\alpha = 0$ for all $\alpha \in \mathbf{R}^n$. Consider the control law

$$\begin{aligned} F &= M(q)\ddot{x}_{o,d} + C(q,\dot{q})x_{o,d} + N(q,\dot{q}) + K_v\dot{e} + K_p e && (114) \\ \tau &= J^T G^+ F + J^T F_N && (115) \end{aligned}$$

where K_v and K_p are symmetric positive definite. Using a Liapunov stability argument, it can be shown that the actual trajectory of the robot converges to the desired trajectory asymptotically [8]. Extensions to the control law result in exponential rate of convergence [17, 16].

This PD control law has the advantage that for a quasi-static change in position Δx the resulting force is

$$\Delta F = K_p\Delta x \qquad (116)$$

and thus we can achieve an arbitrary symmetric stiffness. Experimental results indicate that the trajectory tracking performance of this control law does not always compare favorably with the computed torque control law [13]. Additionally there is no simple design criteria for choosing K_v and K_p to achieve good tracking performance. While the stability results give necessary conditions for stability they do not provide a method for choosing the gains. Nonetheless, PD control has been used effectively in many robot controllers and has some computational features which make it an attractive alternative.

Internal forces

All of the controllers have relied on the choice of a grasping force, $F_N \in \mathcal{N}(G)$ which maintains contact between the fingertips and the object by insuring that the finger forces lie in the friction cone. There are several possible methods for calculating this term. Since F_N does not affect the motion of the object, its choice does not affect object tracking. We begin by showing that given any desired object force, there exists a set of ifnger forces lying in the friction cone which acheives it.

Claim 3.1 *If a grasp is prehensile then given any $F_c \in \mathbf{R}^m$, there exists a null force, F_N, such that the total finger force, $F_c + F_N$, is inside the friction cone.*

Proof By the definition of a prehensile grasp there exists $F_N \in \mathcal{N}(G) \cap \overset{\circ}{FC}$ such that

$$\|F^t_{N_{i,j}}\| < \mu_i \|F^n_{N_i}\| \tag{117}$$

where $F^t_{N_{i,j}}$ is the tangent component of F_N projected onto the j^{th} force direction of the i^{th} contact and F_{N_i} is the normal component of F_N at the i^{th} contact point. F_{N_i} is nonzero for each i and therefore by increasing F_N, we always increase the normal component of the force exerted at each contact with respect to the tangential forces. Since $\overset{\circ}{FC}$ is defined as the cartesian product of the n friction cones in equation 117, $F_N \in \mathcal{N}(G) \cap \overset{\circ}{FC}$ implies $\alpha F_N \in \mathcal{N}(G) \cap \overset{\circ}{FC}$ for all $\alpha \in \mathbf{R}^+$. Now we can look at the unit vector in the $fc + \alpha F_N$ direction as $\alpha \to \infty$:

$$\lim_{\alpha \to \infty} \frac{F_c + \alpha F_N}{\|F_c + \alpha F_N\|} = F_N \tag{118}$$

Since $F_N \in \overset{\circ}{FC}$ it follows that for sufficiently high α, $\frac{F_c + \alpha F_N}{\|F_c + \alpha F_N\|}$ is also in $\overset{\circ}{FC}$ and hence $F_c + \alpha F_N$ is in the interior of the FC. Now from the definition of FC, the individual contact forces must all lie within their respective friction cones simultaneously. □

The simplest F_N is a constant F_N. It must be large enough so that finger forces never leave the friction cone over the entire trajectory of the object. Generally this requires a knowledge of the bounds on the external forces that can be exerted on the object. The advantage of this approach is that $J_h^T F_N$ can be calculated at the same rate as J_h—saving computation time.

A more robust F_N could be calculated by looking at the finger forces (these can be derived from the joint torques, τ, using $F_c = (J_h^T)^{-1}\tau$) and finding a null force which causes $F_c + F_N$ to lie in FC. If the grasp map has a simple form, such as the one given in the example in section , a basis for the null space can be used to construct the set of all valid F_N. This calculation takes more time but may be necessary in the case of large uncertainties.

Other grasp force calculations are discussed in [10] but all of these share some fundamental problems. One difficulty is that in a real-world hand the maximum motor torques that can be generated are finite. Thus, we are not guaranteed that we can apply an F_N which satisfies $F_c + F_N \in FC$ without saturating the motors. Another issue is the effect of the null force term in the presence of errors. If a large internal force term is used and, due to sensor or actuator errors, it does not actually lie in the null space of the grasp matrix, the resulting force can cause positioning errors and in the extreme case, instability.

4 Planning

Chapter 3 was dedicated to establishing control laws under which a grasped object moved along a specified trajectory denoted $x_d(t)$. This is useful in the instance that the task involved does not necessitate a change of grasp. This is not to say that the model and control laws do not allow for fingers to roll on the surface of the object. Indeed, in this instance the motion of the finger tips described by the equations (51) will determine where the grasp points go and how the grasp map changes during the course of the manipulation. However, there is no explicit control of the locations of the fingertips on the surface of the object. There are however a number of applications in which an object needs to be moved in while the fingers are being repositioned in some controlled fashion on the surface of the object: for instance, twirling a baton or regrasping an object for greater stability or manipulability. In this chapter we will discuss the planning of individual finger motions on the surface of an object.

4.1 Dynamic finger repositioning

In Chapter 2, we derived the kinematic equations of contact for a single finger rolling on a body. We will aggregate these into a composite equation for all of the fingers. To review the notation of Chapter 2, we recall that $g_i = x_o x_{f_i}^{-1}$ stands for the position and orientation of the i^{th} finger ($x_{f_i} \in SE(3)$) relative to the body ($x_o \in SE(3)$). Also $\eta_i = (\alpha_{o_i}^T, \alpha_{f_i}^T, \psi_i)^T$ is the vector of the i^{th} contact coordinates with $\alpha_{o_i} \in \mathbf{R}^2$ standing for the surface representation of the object, $\alpha_{f_i} \in \mathbf{R}^2$ the surface representation of the i^{th} finger and ψ_i, the angle of contact (angle between the two orthogonal surface frames). The equations (51) can then be written as

$$\dot\eta_i = B_i(x_o, \eta_i) \begin{pmatrix} \omega_t \\ v_t \end{pmatrix} \tag{119}$$

where

$$\begin{pmatrix} \omega_t \\ v_t \end{pmatrix} = \begin{bmatrix} M_o^{-T} \frac{\partial c_o}{\partial \alpha_o}^T & 0 \\ 0 & M_o^{-T} \frac{\partial c_o}{\partial \alpha_o}^T \end{bmatrix} \begin{bmatrix} n_o \times & 0 \\ c_o \times & I \end{bmatrix} \begin{pmatrix} \omega \\ v \end{pmatrix} \tag{120}$$

and

$$\begin{pmatrix} \omega \\ v \end{pmatrix} = \dot g_i g_i^{-1} \tag{121}$$

In turn, ω and v are linear functions of $\omega_o, v_o, \omega_{f_i}, v_{f_i}$ so that (119) may be rewritten as

$$\dot\eta_i = \tilde B_i(x_o, \eta_i) \begin{pmatrix} \omega_o \\ v_o \\ v_{f_i} \\ \omega_{f_i} \end{pmatrix} \tag{122}$$

Aggregating equations (122) for $i = 1, \ldots, k$ and using the Jacobian of the finger kinematics (42) we have

$$\dot\eta = \tilde B(x_o, \eta) \begin{pmatrix} \omega_o \\ v_o \\ \dot\theta \end{pmatrix} \tag{123}$$

where

$$\bar{B}(x_o,\eta) = \bar{B}(x_o,\eta)\begin{bmatrix} I & 0 \\ 0 & J_f(\theta) \end{bmatrix} \tag{124}$$

Also, the grasping constraint (cf. equation (46) is given by

$$G^T(x_o,\theta)\begin{pmatrix} \omega_o \\ v_o \end{pmatrix} = J(x_o,\theta)\dot{\theta} \tag{125}$$

If the grasp map G is onto (the grasp map is stable) we can uniquely solve for (ω_o, v_o) in (125) as

$$\begin{pmatrix} \omega_o \\ v_o \end{pmatrix} = G^\dagger(x_o,\theta)J(\theta)\dot{\theta} \tag{126}$$

Using (126) in (123) we have

$$\dot{\eta} = \bar{B}(x_o,\eta)\begin{bmatrix} G^\dagger(x_o,\theta)J(\theta) \\ I \end{bmatrix}\dot{\theta} \tag{127}$$

We have determined that η is a smooth bijection of $x_o^{-1}x_f$. Further provided that the fingers do not have more than 6 degrees of freedom, x_f (locally) uniquely determines θ. Consequently (127) can be rewritten as

$$\dot{\eta} = \hat{B}(x_o,\eta)\dot{\theta} \tag{128}$$

Noting that the left hand side of (126) determines x_o, we now combine (126) and (128) as follows: define

$$\dot{\theta} = \dot{\theta}_1 + \dot{\theta}_2 \tag{129}$$

where $\dot{\theta}_1 \in \mathcal{R}(J^T(\theta))$ and $\dot{\theta}_2 \in \mathcal{N}(J(\theta))$. Further, define

$$u_1 = G^\dagger(x_o,\theta)J(\theta)\dot{\theta}_1 x_o \tag{130}$$

Note that the map between u_1 and $\dot{\theta}_1$ is a bijection and let

$$\dot{\theta}_2 = K(\theta)u_2 \tag{131}$$

where the columns of $K(\theta)$ span the null space of $J(\theta)$. Using these two definitions it may be seen that (126) and (127) can be written as

$$\begin{aligned} \dot{x}_o &= u_1 \\ \dot{\eta} &= B_1(x_o,\eta)u_1 + B_2(x_o,\eta)u_2 \end{aligned} \tag{132}$$

Thus the problem of finger repositioning and body manipulation can be reformulated as the problem of steering the states (x_o,η) of the control system (132). In the previous chapter we neglected the u_2 and considered the problem of steering x_o (u_1 was referred to as $\dot{x}_d$). There are as many u_2 as the dimension of the null space of $J(\theta)$. Away from kinematic singularities this dimension is

$$\sum_{i=1}^{k} \max(n_i - m_i, 0) \tag{133}$$

Recall that n_i is the number of joints in the i^{th} finger and m_i is determined by the contact type of the i^{th} finger. The formula above represents the number of extra finger degrees of freedom available to reposition the fingers. Note also that even if $u_2 = 0$ (i.e. no extra finger degrees of freedom) it may still be possible to steer both x_o, η using the u_1 alone.

With this discussion by way of preamble, we begin a detailed of steering systems of the form

$$\dot{x} = B(x)u \tag{134}$$

with $x \in \mathbf{R}^n$, $u \in \mathbf{R}^m$. Note that our steering problem really is a steering problem on a nontrivial manifold $SE(3) \times \mathbf{R}^{5k}$ ((x_o,η) space) rather than $\mathbf{R}^n$ but we will content ourselves with a local discussion, namely on a coordinate chart of $SE(3) \times \mathbf{R}^{5k}$.

4.2 Review of Optimal Control

Following Brockett [2], we will review some results from optimal control. Consider control systems of the form

$$\dot{x} = B(x)u \tag{135}$$

Here $x \in \mathbf{R}^n$, $u \in \mathbf{R}^m$ and $B(x) \in \mathbf{R}^{n\times m}$. The optimal control problem is to minimize

$$\frac{1}{2}\int_0^1 |u|^2 dt \tag{136}$$

subject to the conditions $x(0) = x^i$, $x(1) = x^f$. When $m < n$, this problem is a geodesic problem with a singular Riemannian metric specified by the equations (135) and (136). From Chow's theorem, it follows that this problem has a solution for arbitrary x^i, x^f if and only if the involutive closure of the vector fields described by the columns of $B(x)$ is all of $\mathbf{R}^n$.

It is instructive to analyze some sample solutions to this problem which are in some sense canonic. We start with $n = 3$ and $m = 2$:

$$\begin{aligned} \dot{x}_1 &= u_1 \\ \dot{x}_2 &= u_2 \\ \dot{x}_3 &= x_1u_2 - x_2u_1 \end{aligned} \tag{137}$$

with $x(0) = (0,0,0)$ and $x(1) = (0,0,a)$. The cost function may be written as

$$\min \frac{1}{2}\int_0^1 (\dot{x}_1^2 + \dot{x}_2^2)dt \tag{138}$$

with the control system (137) written as a constraint

$$\dot{x}_3 = x_2\dot{x}_1 - x_1\dot{x}_2 \tag{139}$$

From standard calculus of variations we may write the Euler-Lagrange equations with $\lambda(t)$ denoting the Lagrange multiplier as

$$\begin{aligned} \ddot{x}_1 - \lambda\dot{x}_2 &= 0 \\ \ddot{x}_2 + \lambda\dot{x}_1 &= 0 \\ \dot{\lambda} &= 0 \end{aligned} \tag{140}$$

Equation (140) establishes that $\lambda(t)$ is constant and using equation (137) we see that

$$\begin{bmatrix} \dot{u}_1 \\ \dot{u}_2 \end{bmatrix} = \begin{bmatrix} 0 & \lambda \\ -\lambda & 0 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} = \Lambda \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{141}$$

so that

$$\begin{pmatrix} u_1(t) \\ u_2(t) \end{pmatrix} = e^{\Lambda t} u(0) \tag{142}$$

Consequently, since $x_1(0) = x_2(0) = 0$

$$\begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} = (e^{\Lambda t} - I)\, b; \qquad b = \Lambda^{-1} u(0) \tag{143}$$

Further, because $x_1(1) = x_2(1) = 0$, we have that $\lambda = 2n\pi$ so as to make $e^{\Lambda} = I$. Using this fact and noticing $\dot{x}_3 = \lambda/2(x_1^2 + x_2^2)$ it follows that

$$x_3(1) = x_3(0) + \lambda |b|^2 = a \tag{144}$$

Also since the total cost is

$$\int_0^1 |u|^2 dt = \lambda^2 |b|^2 \tag{145}$$

we see that the optimal choice of $\lambda = 2\pi$ and $|b|^2 = a/2\pi$ but otherwise is arbitrary. The structure of the optimal control for steering between the conjugate points $(0,0,0)$ and $(0,0,a)$ is interesting—it is sums of sines and cosines at a frequency of 2π. The frequency of the signal is dictated by the time interval. Note that the optimal input is still a combination of sines and cosines even if $x_1(1) \neq 0$ or $x_2(1) \neq 0$.

The generalization of this example to the case that $m > 2$ is as follows: consider the situation that the $\{b_i(x) \colon i = 1, \cdots, m\}$ are linearly independent for all x and also all the $m(m-1)$ first-order (first-*etage*) brackets $\{[b_i(x), b_j(x)] \colon i, j = 1, \cdots, m\}$ are linearly independent of the $b_i(x)$. The minimum dimension of the state space to allow for this possibility is $n = m + m(m-1)/2$. The canonic example of this situation is

$$\begin{aligned} \dot{x}_i &= u_i & i &= 1, \cdots, m \\ \dot{x}_{ij} &= x_i u_j - x_j u_i & i &= 1, \cdots, m \end{aligned} \tag{146}$$

A slightly more pleasing representation of equation (146) is obtained by forming the skew symmetric matrix $Y \in \mathbf{R}^{m \times m}$ with the x_{ij} as the bottom lower half (below the diagonal).

$$\dot{x} = u \tag{147}$$

$$\dot{Y} = x u^T - u x^T \tag{148}$$

The Euler-Lagrange equations for (146) are an extension of (140):

$$\ddot{x} = \Lambda \dot{x} \tag{149}$$

$$\dot{\Lambda} = 0 \tag{150}$$

where Λ is the skew symmetric $m \times m$ matrix of Lagrange multipliers associated with Y. Thus, as before, the optimal input u satisfies the equation

$$\dot{u} = \Lambda u \tag{151}$$

with $\Lambda \in \mathbf{R}^{m \times m}$ begin a constant, skew symmetric matrix. Thus $u(t)$ is a linear combination of sinusoids. The exact eigenvalues of Λ are determined by the initial and final state. In fact, if $x(0) = x(1) = 0$, $Y(0) = 0$ and $Y(1)$ a nonsingular $m \times m$ skew symmetric matrix (this requires that m is even), then it can be shown that Λ has $m/2$ sinusoids at frequencies $2\pi, 2 \cdot 2\pi, \cdots, m/2 \cdot 2\pi$. If m is odd and $Y(1)$ has only one zero eigenvalue, Λ has one zero eigenvalue and $(m-1)/2$ sinusoids at frequencies $2\pi, 2 \cdot 2\pi, \cdots, (m-1)/2 \cdot 2\pi$.

4.3 Steering of controllable systems

The results of the previous section derive the geodesics for first etage, linear systems of the form in equation (135). There are several ways of generalizing this further:

1. build the model control systems where the vector fields

$$\begin{array}{ll} b_i & i = 1, \cdots, m \\ [b_i, b_j] & i, j = 1, \ldots, m \\ [[b_i, b_j], b_k] & i, j, k = 1, \cdots, m \end{array}$$

   are linearly independent and $n = m + m(m-1)/2 + m(m-1)(m+1)/3$.

2. understand conditions under which more general control systems can be transformed into canonical systems studied above

3. use the results summarized thus far as inspiration to propose sinusoidal inputs at multiple frequencies which are integrally related to provide (sub-optimal) control laws to steer the systems between arbitrary initial and final conditions.

In this section we shall explore the latter possibility.

First Etage Controllable Systems

By *first etage controllable systems* we mean systems of the form (135) with $m < n$ with the columns of $B(x)$ linearly independent and with

$$\operatorname{span}\{b_i(x),\ [b_i(x), b_j(x)] : i, j = 1, \ldots, m\} = \mathbf{R}^n \tag{152}$$

Namely, one level of Lie brackets is adequate to achieve the full tangent space for all x. We discussed the optimal control of canonical systems satisfying (152) in Section 4.1. Here we illustrate the application of this problem to the steering of a unicycle as shown in Figure 4.1. If u_1 denotes the driving velocity and u_2

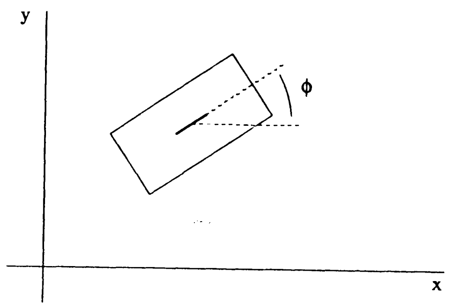


Figure 5: Steerable unicycle. The unicycle has two independent inputs: the *steering* input controls the angle of the wheel, ϕ; the *driving* input controls the velocity of the cart in the direction of the wheel. The configuration of the cart is its Cartesian location and the wheel angle.

---

the steering velocity the functional form of the state equations for this system is

$$\begin{aligned} \dot{x} &= \cos\phi u_1 \\ \dot{y} &= \sin\phi u_1 \\ \dot{\phi} &= u_2 \end{aligned} \tag{153}$$

An approximation to this system is obtained by setting $\cos\phi \simeq 1, \sin\phi \simeq \phi$ and relabelling x as x_1, y as x_3 and ϕ as x_2 to get

$$\begin{aligned} \dot{x}_1 &= u_1 \\ \dot{x}_2 &= u_2 \\ \dot{x}_3 &= x_2 u_1 \end{aligned} \tag{154}$$

To steer this system, we first use u_1, u_2 to steer x_1, x_2 to their desired locations; this may cause x_3 to drift. Now use $u_1 = \alpha\sin(\omega t), u_2 = \beta\cos(\omega t)$ and note that after $2\pi/\omega$ seconds, x_1 and x_2 complete a periodic trajectory and the x_3 coordinate advances by an amount equal to

$$\frac{\pi\alpha\beta}{\omega^2}$$

α, β, ω can now be chosen appropriately. In order to apply this strategy to the unapproximated system (153) we modify the input to

$$\begin{aligned} v_1 &= \cos(\phi)u_1 \\ v_2 &= u_2 \end{aligned}$$

and relabel the states to get

$$\begin{aligned} \dot{x}_1 &= v_1 \\ \dot{x}_2 &= v_2 \\ \dot{x}_3 &= (\tan x_2)v_1 \end{aligned} \tag{155}$$

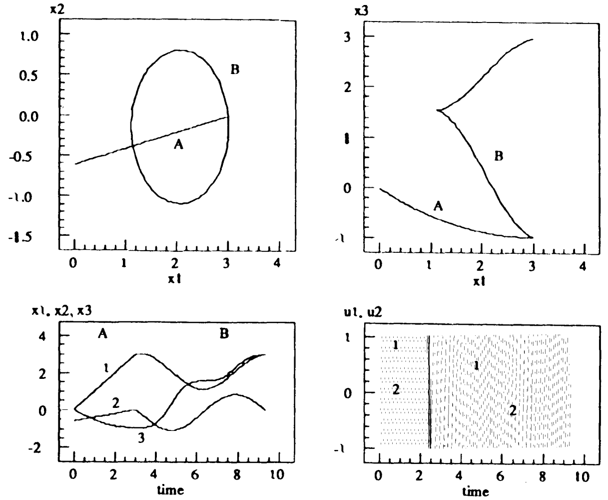


Figure 6: Sample Trajectories for the unicycle. The trajectory shown is a two stage path which moves the unicycle from (0, -.6, 0) to (3, 0, 3). The first portion of the path, labeled A, drives the x_1 and x_2 states to their desired values using a constant input. The second portion, labeled B, uses a periodic input to drive x_3 while bringing the other two states back to their desired values. The top two figures show the states versus x_1; the bottom figures show the states and inputs as functions of time.

As before, we steer x_1 and x_2 using v_1, v_2. To steer the third variable, we use $v_1 = \alpha\sin(\omega t), v_2 = \beta\cos(\omega t)$ as before. Then

$$\dot{x}_3 = \tan(\beta/\omega\sin\omega t) * \alpha\sin(\omega t) \tag{156}$$

The value of x_3 after $2\pi/\omega$ seconds is determined by the constant part of the right hand side of (156). The constant coefficient is given by

$$\frac{1}{2}\cdot\frac{1}{\pi}\int_{-\pi}^{\pi}\tan(\beta/\omega\sin\theta)\alpha\sin(\theta)d\theta$$

Sample trajectories for this scenario are shown in Figure 4.2

Second and Higher Etage Controllable Systems

Systems (135) are said to be *second etage controllable* if

$$\operatorname{span}\{b_i(x),\ [b_i(x), b_j(x)],\ [b_i, [b_j, b_k]] : i, j, k = 1, \ldots, m\} = \mathbf{R}^n$$

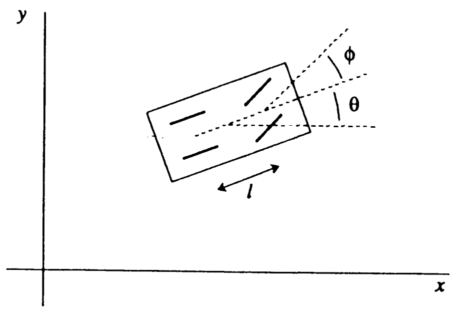


Figure 7: Front wheel drive cart. The configuration of the cart is determined by its Cartesian location, the angle the car makes with the horizontal and the steering wheel angle relative to the car body. The two inputs are the velocity of the front wheels (in the direction the wheels are pointing) and the steering velocity. The rear wheels of the cart are always aligned with the cart body and are constrained to move along the line in which they point or rotate about their center.

An example of such a system is a front wheel drive cart of the form shown in Figure 4.3. As in the case of the previous example u_1 is the driving velocity and u_2 the steering velocity. The equations of this cart are

$$\begin{aligned} \dot{x} &= \cos\theta\cos\phi u_1 \\ \dot{y} &= \sin\theta\cos\phi u_1 \\ \dot{\phi} &= u_2 \\ \dot{\theta} &= \tfrac{1}{l}\sin\phi u_1 \end{aligned} \tag{157}$$

The form of the equations shows that when $\phi = \pi/2$, the cart cannot be driven forward. As in the previous section an approximation to this system is instructive. Relabelling the variables x, y, ϕ, θ as x_1, x_4, x_2, x_3 and approximating sines and cosines as before yields

$$\begin{aligned} \dot{x}_1 &= u_1 \\ \dot{x}_2 &= u_2 \\ \dot{x}_3 &= \tfrac{1}{l}x_2u_1 \\ \dot{x}_4 &= x_3u_1 \end{aligned} \tag{158}$$

Note that $\mathrm{span}\{b_1, b_2, [b_1, b_2], [b_1, [b_1, b_2]]\} = \mathbf{R}^n$ so that system is a second etage system. It is also easy to verify that this condition also holds for the original system. Steering the states x_1, x_2, x_3 of (158) is immediate from the previous section. To steer x_4 note that if

$$u_1 = \alpha\cos\omega t \ , \ u_2 = \beta\cos 2\omega t$$

then x_1, x_2 and x_3 are all periodic and return to their initial values after $2\pi/\omega$ seconds. Also

$$x_3 = -\frac{\alpha\beta}{4\omega^2}\cos\omega t \ - \ \frac{\alpha\beta}{12\omega^2}\cos 3\omega t$$

so that it too is periodic. Finally the increment in x_4 is given by

$$-\frac{\pi\alpha^2\beta}{4\omega^2}$$

To carry this development through for the unapproximated system define $v_1 = u_1\cos\theta\cos\phi$ and $v_2 = u_2$. Then with the same relabelling as before, the equations become

$$\begin{aligned} \dot{x}_1 &= v_1 \\ \dot{x}_2 &= v_2 \\ \dot{x}_3 &= \tfrac{1}{l}\tfrac{\tan(x_2)}{\cos(x_3)}v_1 \\ \dot{x}_4 &= \tan x_3 v_1 \end{aligned} \tag{159}$$

We refer to such systems as *triangular* but not *strictly triangular* since $\dot{x}_3$ depends on x_3. By approximating $\cos x_3$ by 1, the equations become strictly triangular; using $v_1 = \alpha\cos\omega t$, $v_2 = \beta\cos 2\omega t$ we can solve for the Fourier series coefficients of x_1, x_2, x_3 and x_4. Note that only the Fourier coefficient corresponding to the zero frequency is needed to get the change in x_4 after one time period.

To summarize, it is easy to see that for higher etage (than 2) controllable systems one can use simple Fourier series techniques to steer the systems using as inputs integrally related sinusoids provided that they are strictly triangular in the sense discussed above. To steer the variable corresponding to the k^{th} etage it is possible to use frequencies ω and $k\omega$ in the two inputs. The Lissajous figures that are obtained from the phase portraits of the different variables are quite instructive. Consider the Figure 4.4, which is the system of (159) with $\cos x_3$ replaced by 1 and the inputs $v_1 = \alpha\cos\omega t$, $v_2 = \beta\cos 2\omega t$. The upper left plot is the Lissajous figure for x_1, x_2 (two loops): The lower left plot is the corresponding figure for x_3, x_1 (one loop) and the open curve in x_4,x_1 shows the increment in the x_4 variable. The very powerful implication here is that the *Lie bracket directions correspond to rectification of harmonic periodic motions of the driving vector fields and the harmonic relations are determined by the etage of controllability desired.* This point has also been made rather elegantly by Brockett [3] in the context of the rectification of mechanical motion.

Open Problems and Nontriangular Higher Etage Systems

Consider the kinematic equations for a front wheel drive cart with a trailer as shown in Figure 4.5.

$$\begin{aligned} \dot{x} &= \cos\theta\cos\phi u_1 \\ \dot{y} &= \sin\theta\cos\phi u_1 \\ \dot{\theta} &= \tfrac{1}{l}\sin\phi u_1 \\ \dot{\phi} &= u_2 \\ \dot{\psi} &= \tfrac{1}{d}\sin(\psi - \theta)\cos\phi u_1 \end{aligned} \tag{160}$$

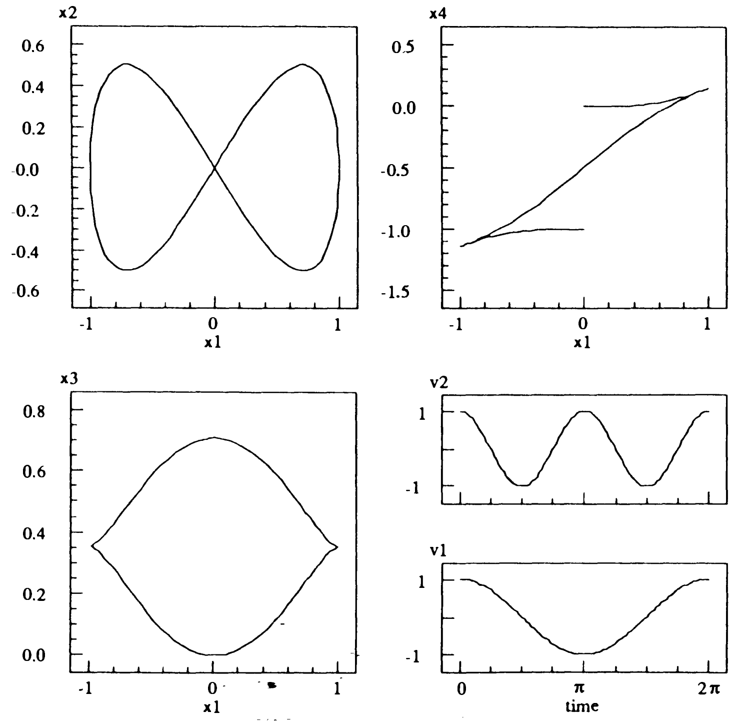


Figure 8: Trajectories for a simple cart. The trajectory shown illustrates motion in the x_4 direction. Periodic inputs are used to generate periodic trajectories in the first three states while giving an open trajectory in the last state.

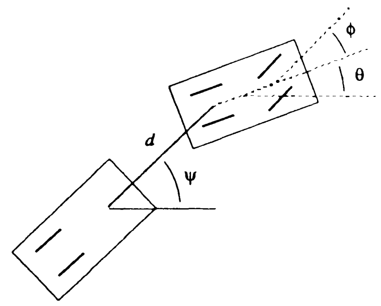


Figure 9: Front wheel drive cart with trailer. The trailer configuration is described the the angle the trailer makes with the horizontal, ψ. The rear wheels of the trailer are fixed and constrained to move along the line in which they point or rotate about their center. The inputs to the system are the inputs to the cart: the driving velocity (of the front wheels) and the steering velocity. This system is an example of a third etage system; higher etage systems can be generated by adding extra trailers.

It may be verified that

$$\text{span}\{b_1,\ b_2,\ [b_1, b_2],\ [b_1, [b_1, b_2]],\ [b_1, [b_1, [b_1, b_2]]]\} = \mathbf{R}^5$$

It is also not difficult to see that with k trailers we need Lie brackets up to the $k+2$ etage to guarantee controllability. Also, it may be seen that after redefining the inputs the system is only triangular rather than strictly triangular so that the harmonic analysis techniques of the previous subsection cannot be applied even though numerical simulation suggests that sinusoids of integer multiples are useful to steer along the direction of the jth Lie bracket. The full theory for these systems is as yet incomplete.

By way of concluding remarks for this section we would like to mention that the program begun in Section 4.1 has obvious implications for the problem of geodesics for nonholonomic or singular or Carnot-Caratheodory Riemannian metrics. Also, we would like to mention that the application that motivated the discussion of this chapter really involves steering on $SE(3) \times \mathbf{R}^{5k}$ rather than $\mathbf{R}^n$ so that the corresponding optimal control theory is more involved. The problem of controlling a finger rolling on the surface of an object in terms of the contact curvatures of the object and the finger tip (i.e. one of the $\mathbf{R}^5$ of the space mentioned above) is discussed in some detail in Li [9]. In that paper a solution for a spherical finger rolling on a plane was given; more general cases remain unsolved.

Acknowledgements

We would like to thank Roger Brockett for introducing us to his work on singular Riemannian metrics (chapter 4), Li Zexiang for making the connections between regrasping and steering, and John Canny, Jean Paul Laumond and the denizens of the robotics lab for several useful discussions.

Notation

G	grasp map; maps contact forces to object forces
J	hand Jacobian; maps joint velocities to contact velocities
$\mathcal{N}(A)$	null space of the matrix A
$\mathcal{R}(A)$	range space of the matrix A
$\mathcal{S}(\omega)$	skew symmetric matrix associated with ω; defined by $\mathcal{S}(\omega)a = \omega \times a$ for all $a \in \mathbf{R}$; $\mathcal{S}(\omega) \in se(3)$
$SE(3)$	special Euclidean group; rigid motions
$se(3)$	$T_eSE(3)$; Lie algebra of $SE(3)$; generalized velocity
$SO(3)$	special orthogonal group; rotation matrices
$so(3)$	$T_eSO(3)$; Lie algebra of $SO(3)$; rotational velocity
$[f,g]$	Lie bracket between two vector fields; in coordinates $[f,g] = \frac{\partial g}{\partial x} f - \frac{\partial f}{\partial x} g$

The following dimensions are used throughout the paper

k	number of fingers
m_i	number of forces exerted by the i^{th} contact
m	total number of grasping constraints
n_i	number of degrees of freedom of the i^{th} finger
n	total number of degrees of freedom for the hand

References

[1] A. K. Bejczy. *Robot Arm Dynamics and Control*. Technical Report 33-699, Jet Propulsion Laboratory, 1974.

[2] R. W. Brockett. Control theory and singular riemannian geometry. In *New Directions in Applied Mathematics*, pages 11–27, Springer-Verlag, New York, 1981.

[3] R. W. Brockett. On the rectification of vibratory motion. *Sensors and Actuators*, 1989. (to appear).

[4] A. Cole, J. Hauser, and S. Sastry. Kinematics and control of multifingered hands with rolling contact. In *IEEE Conference on Robotics and Automation*, pages 228–233, 1988.

[5] J. Kerr. *An Analysis of Multi-fingered Hands*. PhD thesis, Stanford University, Department of Mechanical Engineering, 1984.

[6] O. Khatib. Augmented object and reduced effective inertia in robot systems. In *Automatic Control Conference*, 1988.

[7] O. Khatib. A unified approach for motion and force control of robot manipulators: the opertional space formulation. *IEEE Journal on Robotics and Automation*, RA-3(1):43–53, February 1987.

[8] D. Koditschek. Natural motion for robot arms. In *IEEE Control and Decision Conference*, pages 733–735, 1984.

[9] Z. Li. *Kinematics, Planning and Control of Dextrous Robot Hands*. PhD thesis, Department of Electrical Engineering and Computer Science, University of California, Berkeley, 1989.

[10] Z. Li, P. Hsu, and S. Sastry. On kinematics and control of multifingered hands. In *IEEE Conference on Robotics and Automation*, pages 384–389, 1988.

[11] J. Y. S. Luh, M. W. Walker, and R. P. Paul. Resolved acceleration control of mechanical manipulators. *IEEE Transactions on Automatic Control*, AC-25, 1980.

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

[13] R. Murray and S. Sastry. Control experiments in planar manipulation and grasping. In *IEEE Conference on Robotics and Automation*, pages 624–631, 1989.

[14] V. Nguyen. *The Synthesis of Stable Force-Closure Grasp*. Master's thesis, Massachusetts Institute of Technology, 1986.

[15] R. M. Rosenberg. *Analytical Dynamics of Discrete Systems*. Plenum Press, New York, 1977.

[16] N. Sadegh. *Adaptive Control of Mechanical Manipulators: Stability and Robustness Analysis*. PhD thesis, Department of Mechanical Engineering, University of California, Berkeley, California, 1987.

[17] J. E. Slotine and W. Li. On the adaptive control of robot manipulators. *International Journal of Robotics Research*, 6:49–59, 1987.

[18] G. Strang. *Linear Algebra and its Applications*. Harcourt Brace Jovanovich, San Diego, third edition, 1988.

Deparment of Electrical Engineering and Computer Science,
University of California, Berkeley CA 94720

Planning and Executing Robot Assembly Strategies in the Presence of Uncertainty

Bruce R. Donald[1]

Abstract

Robot control systems are subject to significant uncertainty and error. Typical robots are also equipped with sensors—force sensors, kinesthetic position sensors, tactile sensors, vision, and so forth. However, these sensors are also subject to significant uncertainty. Finally, the geometrical models of the robot and the environment (parts, obstacles, *etc.*) cannot be exact—they are accurate only to manufacturing tolerances, or to the accuracy of the sensors used to acquire the models. Uncertainty is an absolutely fundamental problem in robotics, and plans produced under the assumption of no uncertainty are meaningless. What is needed is a principled theory of planning in the presence of uncertainty. Such a theory must not only be computational, but must also take uncertainty into account *a priori*. In motion planning with uncertainty, we exploit compliant motion—sliding on surfaces—in order to effect a "structural" reduction in uncertainty. Such compliant motion plans can be synthesized from a computational analysis of the geometry of the holonomic constraints.

We will present a precise framework for motion planning with uncertainty. In particular, given geometric bounds on the uncertainty in sensing and control, we develop algorithms for generating and verifying compliant motion strategies that are guaranteed to succeed as long as the sensing and control uncertainties lie within the specified bounds. The first results in this theory begin with Lozano-Pérez, Mason, and Taylor [LMT], with subsequent contributions by Mason [Ma2], Erdmann [E], Donald [D], and others. This research has led to a theoretical computational framework for motion planning with uncertainty, which we explore in this focused survey paper.

[1]This paper describes research done in the Computer Science Robotics Laboratory at Cornell University. Support for our robotics research is provided in part by the National Science Foundation under grant No. IRI-8802390 and by a Presidential Young Investigator award, and in part by the Mathematical Sciences Institute.

Contents

1 A Geometric Approach to Robot Planning

1.1 Introduction

For the past seven years, I have been interested in building robot planning systems that can function at the task-level. A task-level specification of a robot plan might have the form, *Put together this disk rotor assembly.* The planner is given geometric models of the parts, and geometric or analytic models of the robot dynamics. Beyond this, the specification, or input to the planner does not mention the specific kinematic and dynamic constraints that the robot must obey; these are determined by the planner using geometrical computation. Typically, there are additional constraints that the planner must also obey, for example, *Construct a robot plan that is robust in the face of uncertainty and error. Use compliant motion where appropriate to reduce uncertainty.* The goal of a task-level planner is to take a task-level specification and to produce a runnable robot program — one which is fully specified in terms of force-control, kinematics, and dynamics — that can accomplish the task.

Major advances in task-level planning can enable robotics to achieve its full potential in the assembly domain. Today, even existing robots cannot be exploited to their full capacity. For example, assembly tasks require compliant motion; however, compliance requires force-control, and such force-control motion strategies are quite difficult for humans to specify. Furthermore, robot assembly programs are very sensitive to the details of geometry. Finally, reprogramming a general purpose robot for a new assembly task can take time on the order of man-months. For these reasons, we have been working on the automatic synthesis of motion strategies for robots. Much work is required to reduce such current task-level technologies to practice.

Research in task-level planning is often characterized as *theoretical robotics*. There are several reasons for this; the first is that much of the work has been concerned with constructing a *theory* of planning. In other words, the computational problem "task-level planning" is not well-specified. Much of our work lies in specifying the computation precisely. Second, given some sort of decomposition of task-level planning into "planning problems", one is immediately driven to ask, *What are the algorithms for these problems? Can plans, in general, be computed? How efficiently can planning algorithms run?* Historically, the nature of these questions has led researchers to apply tools from theoretical computer science, computational geometry, and algebra.

Recently, a great deal of attention has been focused on a particular robotics problem, called the find-path, or generalized movers' problem. In this problem, we ask the purely kinematic question, can a robot system be moved from one configuration to another, without colliding with obstacles? (See, See figs. 1, 2). This is a nicely-defined mathematical problem, and, after much research, at this point its computational complexity is precisely known.

In fact, the neatness of this problem is deceptive, so much so that this formal problem has even been called "the" motion planning problem. From a task-level viewpoint, there

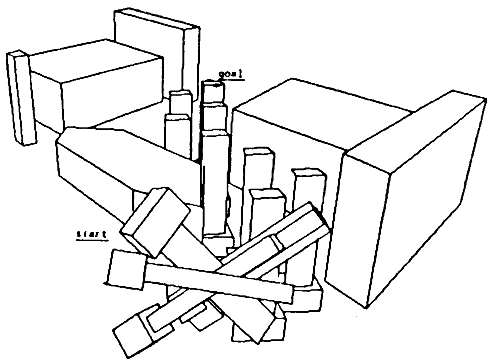


Figure 1: An example of a solution path for the classical find-path, or Movers' problem. This illustration is a "time-lapse" picture of a path found for a hammer-shaped object through a maze of polyhedral obstacles. From [Donald, 87a].

is much hidden in the statement "Can the robot system be moved...?" Specifically, the find-path problem assumes that the robot has a perfect control system that can exactly execute the plan, and that the geometric and analytic models of the robot and obstacles are exact.

In reality, of course, robot control systems are subject to significant uncertainty and error. Typical robots are also equipped with sensors—force sensors, kinesthetic position sensors, tactile sensors, vision, and so forth. However, these sensors are also subject to significant uncertainty. Finally, the geometrical models of the robot and the environment (parts, obstacles, etc.) cannot be exact—they are accurate only to manufacturing tolerances, or to the accuracy of the sensors used to acquire the models. Uncertainty is not a mere engineering detail; in particular, it is characteristically impossible to "patch" these perfect plans in such a way that they will function once uncertainty comes into play. Uncertainty is an absolutely fundamental problem in robotics, and plans produced under the assumption of no uncertainty are meaningless. What is needed is a principled theory of planning in the presence of uncertainty. Such a theory must not only be computational, but must also take uncertainty into account *a priori*. The overlap with exact motion planning algorithms can be stated roughly as follows: exact kinematic planning algorithms provide a computational-geometric theory of holonomic constraints. In motion planning with uncertainty, we exploit compliant motion—sliding on surfaces—in order to effect a "structural" reduction in uncertainty. Such compliant motion plans can be synthesized from a computational analysis of the geometry of the holonomic constraints.

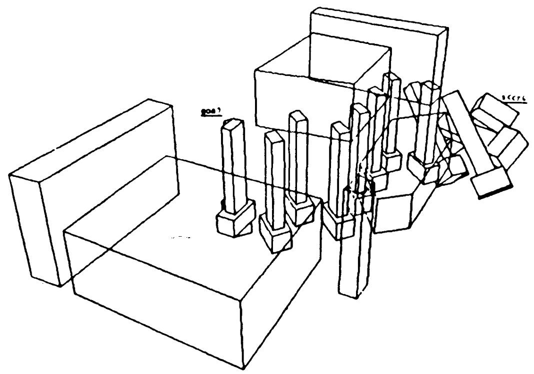


Figure 2: A different view of the solution path for the Movers' problem in fig. 1, with the obstacles "transparent" to allow us to view the rotations better.

1.2 Uncertainty and Compliant Motion

Robots are subject to the following kinds of uncertainty:

1. Inaccuracy and errors in sensing,
2. Inaccuracy and errors in control,
3. Uncertainty about the geometry of the environment.

The last (3) is called "model error", and has received little previous attention. Model error arises because, in general, a robot can have only approximate knowledge of the shape and position of objects in the environment.

We now ask the question:

- How can robots plan and execute tasks (for example, a mechanical assembly using compliant motion) in the presence of these three kinds of uncertainty?

This is perhaps the most fundamental problem in robotics today. We call it the problem of *motion planning with uncertainty*.

In motion planning with uncertainty, the objective is to find a plan which is guaranteed to succeed even when the robot cannot execute it perfectly due to control and sensing uncertainty. With control uncertainty, it is impossible to perform assembly tasks

which involve sliding motions using position control alone. To successfully perform assembly tasks, uncertainty must be taken into account, and other types of control must be employed which allow *compliant motion*.

Compliant motion occurs when a robot is commanded to move into an obstacle, but rather than stubbornly obeying its motion command, it complies to the surface of the obstacle. Work on compliant motion[2] attempts to utilize the task geometry to plan motions that reduce the uncertainty in position by maintaining sliding contact with a surface. Plans consisting of such motions can be designed to exploit the geometry of surfaces around the goal to guide the robot. By computing "preimages"[3] of a geometrical goal in configuration space, guaranteed strategies can be synthesized geometrically: We call this a *geometrical theory of planning*. The first results in this theory begin with Lozano-Pérez, Mason, and Taylor (or [LMT]), with subsequent contributions by Mason [Ma2], Erdmann [Erdmann] and Donald [D]. This research has led to a theoretical computational framework for motion planning with uncertainty, which we denote [LMT,E,D]. See [Buc, EM, Bro, CR, Can89a, FHS, LLS] for other allied work.

The [LMT,E,D] framework begins by observing that the use of active compliance enables robots to carry out tasks in the presence of significant sensing and control errors. Compliant motion meets external constraints by specifying how the robot's motion should be modified in response to the forces generated when the constraints are violated. For example, contact with a surface can be guaranteed by maintaining a small force normal to the surface. The remaining degrees of freedom (DOF)—the orthogonal complement of the normal-space—can then be position-controlled. Using this technique, the robot can achieve and retain contact with a surface that may vary significantly in shape and orientation from the programmer's expectations. Generalizations of this principle can be used to accomplish a wide variety of tasks involving constrained motion, e.g., inserting a peg in a hole, or following a weld seam. The specification of particular compliant motions to achieve a task requires knowledge of the geometric constraints imposed by the task. Given a description of the constraints, choices can be made for the compliant motion parameters, e.g., the motion freedoms to be force controlled and those to be position controlled. It is common, however, for position uncertainty to be large enough so that the programmer cannot unambiguously determine which geometric constraints hold at any instant in time. For example, the possible initial configurations for a peg in hole strategy (see fig. 3) may be "topologically" very different, in that different surfaces of the peg and hole are in contact. Under these circumstances, the programmer must employ a combined strategy of force and position control that guarantees reaching the desired final configuration from all the likely initial configurations. We call such a strategy a *motion strategy*.

Motion strategies are quite difficult for humans to specify. Furthermore, robot programs are very sensitive to the details of geometry. For this reason, we have been working

[2] See [Ma] for an introduction and survey.

[3] The *preimage* of a goal [LMT] is the set of configurations from which a particular commanded compliant motion is guaranteed to succeed.

on the automatic synthesis of motion strategies for robots.

Note that compliant motion planning with uncertainty is significantly different from motion planning with perfect sensing and control along completely-known configuration space obstacle boundaries [Kou, HW, BK]. The first difference is physical:

- From a practical point of view, the motion-in-contact plans generated under the assumption of perfect control cannot ever be executed by a physical robot using position control alone.

The second difference is combinatorial:

- The planning of motions in contact with perfect control has the same time-complexity as planning free-space motions; that is, it can be done in time $O(n^r \log n)$ for r degrees of freedom and n faces or surfaces in the environment [C1]; the exponent is worst-case optimal. However, prior to [Donald 87b,88a], there are no upper bounds for planning compliant motions *with* uncertainty. However, for r fixed at 3, the problem is hard for non-deterministic exponential time [CR]. We showed in [Donald 87b,88a] that the planar case, somewhat surprisingly, turns out to be tractable.

The physical difficulty in fact motivates our work. This experimental viewpoint is manifest in [Donald 87b,88b; Jennings, Donald and Campbell]. It is also possible to adopt a more theoretical perspective; in [Donald 88a; Briggs 89] we concentrate on the geometric and combinatorial aspects of the problem.

1.3 Example: Synthesizing Guaranteed Plans

Error Detection and Recovery (EDR) strategies arose as a response to certain inadequacies in the "guaranteed success" planning model. Hence, in order to study EDR, it is first necessary to understand the structure of guaranteed compliant motion strategies, and how they may be synthesized. To this end, in these notes, we wish to give a flavor for the kinds of issues that arise in planning guaranteed motion strategies under uncertainty. We examine a very special case—the planar polygonal case. (That is, the workspace environment ("parts" and "obstacles") are polygonal). First, we will briefly develop a simple dynamic model that is adequate for this situation. Next, we carefully define the computational problem of synthesizing planar guaranteed strategies under uncertainty in sensing and control, and generalize our definition to include uncertainty in the shape of the parts and obstacles. Finally, we hint at the types of computational techniques employed.

For this very simple planning problem, we find that motion plans with a simple structure suffice. This permits us to illustrate by a specific example the situation that we investigated in [Donald 87b, 88a, 88b, 89] for fairly arbitrary plans. Together with the problem of constructing guaranteed compliant motion strategies under uncertainty, we will consider a rather restricted class of strategies, namely those that terminate by sticking on a surface.

1.4 Dynamic Model

Compliant motion is only possible with certain dynamic models. We will employ the generalized damper model [W, Ma]. We assume that the environment is polyhedral, and that it describes the configuration space of the robot, so that the robot is always a point. The planned path consists of r successive motions in directions $v_1, \ldots, v_r$. Each motion terminates when it sticks, due to coulomb friction, on some surface in the environment. Because of control uncertainty, however, the robot cannot move with precisely velocity v_i on the i^{th} motion. Instead, it moves with velocity v_i^{free}, which lies in a cone of velocities $B_{ec}(v_i)$ about v_i. The boundaries of the cone form an angle of ϵ_c with v_i. ϵ_c is called the *control uncertainty*, and $B_{ec}(v_i)$ the *control uncertainty cone about* v_i. It is, in fact, equivalent to regard ϵ_c as specifying that v_i^{free} lies within a ball about v_i in velocity space.

For a compliant motion, the robot moves along an obstacle surface with a sliding velocity v_i^{slide} which is the projection onto the surface of the obstacle of some v_i^{free} in $B_{ec}(v_i)$. Under generalized damper dynamics, the motion of a polyhedral robot without rotations is completely specified by the motion of its reference point in configuration space. See fig 3.

The i^{th} motion terminates by sticking on a surface when the velocity v_i^{free} in $B_{ec}(v_i)$ points into the negative coulomb friction cone on a surface (see fig. 4). Thus sticking on a surface can be non-deterministic. We will assume that motion i can terminate on any reachable surface for which some velocity $v_i^{free} \in B_{ec}(v_i)$ is inside the negative friction cone. Sticking termination is motivated by the fact that a robot with a force-sensing wrist can easily recognize sticking and robustly terminate the motion.

To test whether sticking is possible on some set of (say, goal) edges, we simply perform a geometric cone intersection on each edge. Sticking is possible when the intersection of the cone of velocity uncertainty and the negative friction cone have a non-trivial intersection. Since determining the possibility (or necessity) of sticking reduces to a simple cone intersection, which may be done in constant time per edge, in this preface we will focus on the more difficult issue of computing reachability. Representing friction in our planar polygonal configuration space is easy; see fig. 4. However, the more general question of representing friction in configuration spaces with rotations is subtle: see [Erdmann, BRS].

While robust implementation of generalized damper dynamics is still a research issue, in our robotics laboratory we have recently implemented an experimental force-control system with this dynamic model to test our geometrical planning theories.

1.5 Definitions

We will regard the goal region G as a polyhedral region in configuration space. Since in general we cannot precisely know the initial configuration of the robot, we will also assume that the start region R is some polyhedral region in configuration space.

We now pose three problems (see fig. 6):

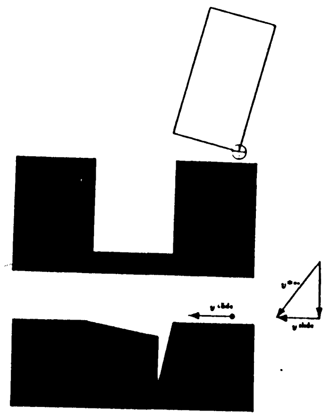


Figure 3: (a) Peg in hole environment. (b) Configuration space, showing the motion of the reference point during compliant motion.

Problem 1: One-Step Compliant Motion Planning with Uncertainty. Given a polyhedral start region R, a polyhedral environment $\mathcal{P}$ of n vertices, control uncertainty ϵ_c, coefficient of friction μ, and a polyhedral goal G, find one commanded motion direction v such that under v, all possible motions from R terminate by sticking in G.

Problem 2: One-Step Compliant Motion Verification. Given $(R, \mathcal{P}, \epsilon_c, \mu, G)$ and v, verify that under v, all possible motions from R terminate by sticking in G.

Problem 3: Compliant Motion Planning with Uncertainty Given $(R, \mathcal{P}, \epsilon_c, \mu, G)$, and an integer r, find a sequence of r motions such that each motion terminates in sticking, and the final motion terminates in the goal. Or, if no such r-step strategy exists, then say so.

1.6 Model Error

We now introduce model error into the picture. How can compliant motion strategies be synthesized in the presence of sensing, control, and geometric model error, such that the strategies are guaranteed to succeed so long as the errors lie within the specified bounds? As an example, consider a peg-in-hole assembly with sensing and control uncertainty, with toleranced parts. We wish to synthesize a compliant motion strategy that is guaranteed to succeed so long as the parts lie within the specified tolerances, and the sensing and control errors lie within the specified bounds.

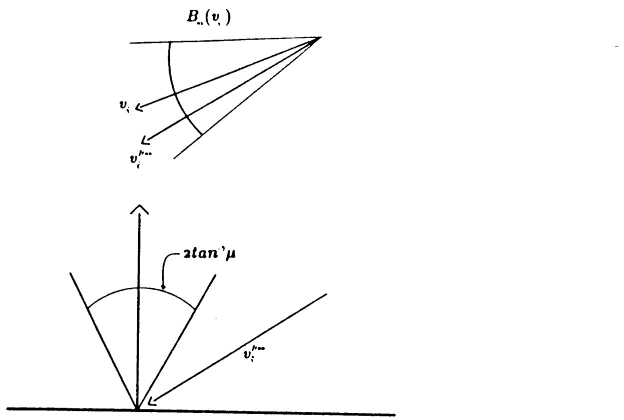


Figure 4: The control uncertainty cone B_{ec} about a nominal commanded velocity v_c. The coulomb friction cone edge makes an angle of $\tan^{-1}\mu$ with the normal. The velocity v_c^{free} will result in sliding on the surface.

We can now state our fourth problem. We parameterize our family of geometries by the parameter space $\alpha_1, \ldots, \alpha_k$, so that $\mathcal{P}(\alpha_1, \ldots, \alpha_k)$ denotes a particular geometry, and each α_i lies within some given interval. Note that this parameterization need not be "continuous". We wish to find a plan which will succeed for all geometries $\mathcal{P}(\alpha_1, \ldots, \alpha_k)$.

Problem 4: Compliant Motion Planning with Uncertainty and Geometric Model Error Given $(R, \mathcal{P}(\alpha_1, \ldots, \alpha_k), \epsilon_c, \mu, G)$, and an integer r, find a sequence of r motions such that, for all possible values of $\alpha_1, \ldots, \alpha_k$, each motion terminates in sticking, and the final motion terminates in the goal. Or, if no such r-step strategy exists, then say so.

We have begun an attack on this problem by introducing additional dimensions to the configuration space; each dimension represents a way in which the parts could parametrically vary. We termed the product space of the motion degrees of freedom and the geometric model variational dimensions "generalized configuration space" and showed how to compute "preimages" [LMT,E] of a geometrical goal in this generalized configuration space. The preimage of a goal is the set of (generalized) configurations from which a particular commanded compliant motion is guaranteed to succeed. Using this technique, we have developed a framework for computing motion strategies that are guaranteed to succeed in the presence of sensing, control, and geometric model uncertainty. The motion strategies comprise sensor-based gross motions, compliant motions, and simple pushing motions.

1.7 Backprojections

We now sketch, for a specific example, a computational approach to the compliant motion planning problem with uncertainty. In [Donald 87b, 88a;b, 89] we have mounted a more systematic attack using similar, albeit considerably generalized methods.

Erdmann [Erdmann] has shown that in the plane, when G is a single edge of the polygonal environment $\mathcal{P}$, then the one-step verification problem (2) can be done in time $O((n + c)\log n)$, where c is the number of intersections encountered by a planar arrangement algorithm.

Erdmann's algorithm makes use of *backprojections*, which he defined as a simplified case of the [LMT] notion of geometrical preimages. The question of goal reachability from a start region can be reduced to deciding the containment of the start region within the backprojection of the goal.

The backprojection $B_\theta(G)$ of a goal G (with respect to a commanded velocity v_θ^*) consists of those configurations guaranteed to enter the goal (under v_θ^*).[4] That is, the backprojection is the set of all positions from which all possible trajectories consistent with the control uncertainty are guaranteed to reach G. See fig. 5. The terms "preimage" and "backprojection" come from viewing motions as "mappings" between subsets of configuration space. Hence the backprojection of a goal is the set of configurations from which a particular commanded compliant motion is guaranteed to succeed. [LMT] envisioned a back-chaining planner that recursively computes preimages of a goal region. Successive subgoals are attained by motion strategies. Each motion terminates when all sensor interpretations indicate that the robot must be within the subgoal.

See fig. 6. Here is the key point about backprojections: Given $(R, \mathcal{P}, \epsilon_c, \mu, G, v_\theta^*)$, the one-step verification problem (2) reduces to testing set containment, i.e., that

$$R \subset B_\theta(G).$$

Erdmann showed that when G is a single edge of a planar environment $\mathcal{P}$, then $B_\theta(G)$ has size $O(n)$ and can be computed as follows (see fig. 7):

(a) Find all vertices in the environment where sticking is *possible* under v_θ^*.

(b) At each of these vertices, erect two rays, parallel to the two edges of the *inverted* velocity cone $-B_{ec}(v_\theta^*)$.

(c) Compute the arrangement from the environment plus these additional $O(n)$ constraints.

(d) Starting at the goal edge, trace out the backprojection region.

An excellent exposition of Erdmann's algorithm can be found in [E]. With John Canny, we implemented a plane-sweep algorithm for backprojections from general polygonal

[4] The star * denotes the ideal, or perfect control velocity. Henceforth, we will typically identify a commanded motion v_θ^* with its angular direction θ.

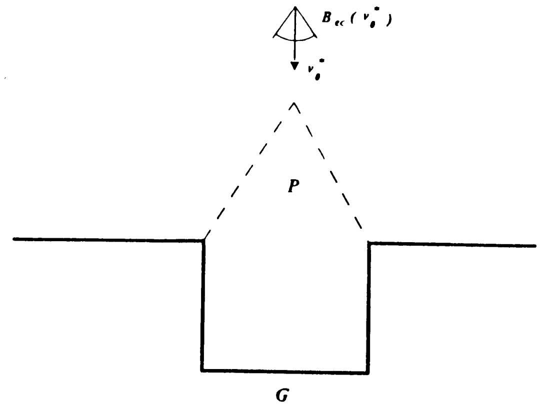


Figure 5: The goal is the region G. Sliding occurs on vertical surfaces, and sticking on horizontal ones. The commanded velocity is v_θ^*, and the control uncertainty is $B_{ec}(v_\theta^*)$. The *backprojection* of G with respect to θ is the region P.

goals. The idea is similar, and interested readers may find details in chapter V. While in general the complexity of these algorithms is $O((n+c)\log n)$, both methods take time $O(n \log n)$ and space $O(n)$ when the goal has $c = O(n)$ intersections with $\mathcal{P}$.

This sketch of the algorithm for verifying a compliant motion strategy gives some flavor for the kind of solution we desire. Consider the remaining problems. The trick to obtaining a computational solution to problem (1) lies in considering all possible backprojections (for all possible θ) simultaneously, and choosing θ that are suitable. In [Donald 87b,88a, Briggs 89] we show that it is possible to do this efficiently, and also provide exact algorithms for problems (3) and (4) as well.

Finally, we must note that there are many variants and facets of the guaranteed compliant motion planning problem; these we address in [Donald 87b, 88a;b, 89, 90]. For these problems, we also gave computational approaches.

1.8 Error Detection and Recovery

A task-level planning system requires a precise theory of Error Diagnosis and Recovery, and a method for generating sensor-based plans with built-in error detection and recovery. We call this the *Error Detection and Recovery (EDR) Theory*. I believe that the chief contribution of our work has been to define and develop a theory of EDR. Perhaps the simplest way to view EDR is as a generalization, or extension, of guaranteed geometrical planning theories. Of course, much work is required to make this notion precise, and

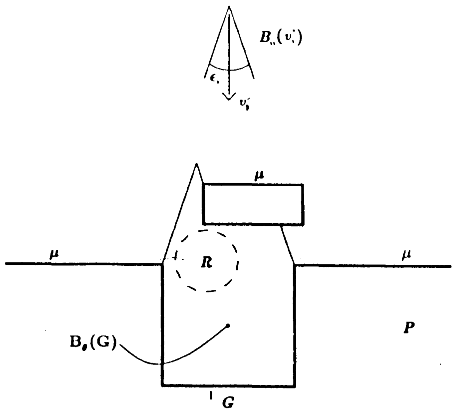


Figure 6: A compliant motion planning problem with goal G, start region R, control uncertainty ϵ_c, obstacles $\mathcal{P}$, and coefficient of friction μ. v_θ^* is a solution which is guaranteed to succeed.

useful: that is the bulk of the work in our monograph [Donald 89], in which in effect, we proposed a "new" theory of planning.

Our approach to EDR observes that there are certain inadequacies with the "guaranteed success" planning model. It is simply not always possible to find plans that are guaranteed to succeed. For example, if tolerancing errors render an assembly infeasible, the plan executor should stop and signal failure. In such cases the insistence on guaranteed success is too restrictive. For this reason we turn to EDR strategies. EDR plans will succeed or fail recognizably: in these more general strategies, there is no possibility that the plan will fail without the executor realizing it. The EDR framework fills a gap when guaranteed plans cannot be found or do not exist: it provides a technology for constructing plans that might work, but fail in a "reasonable" way when they cannot.

The key theoretical issue is: How can we relax the restriction that plans must be guaranteed to succeed, and still retain a theory of planning that is not completely *ad hoc*? We attempt to answer this by giving a constructive definition of EDR strategies. In particular, our approach provides a formal test for verifying whether a given strategy is an EDR strategy. The test is formulated as a decision problem about projection sets in a generalized configuration space which also encodes model error. Roughly speaking, the projection sets represent all possible outcomes of a motion (the *forward projection*), and weakest preconditions for attaining a subgoal (the *preimage*).

Given the formal test for "recognizing" an EDR strategy, we can then test the definition by building a generate-and-test planner. The generator is trivial; the recognizer is an algorithmic embodiment of the formal test. It lies at the heart of this research. A

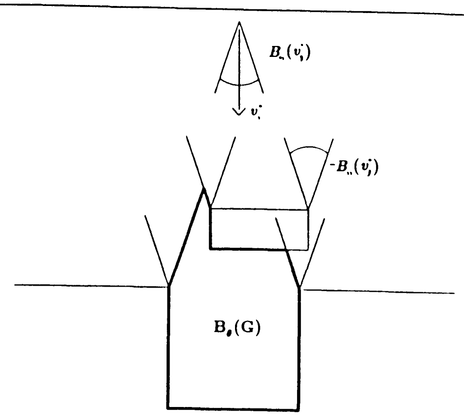


Figure 7: An illustration of Erdmann's algorithm for computing the backprojection $B_\theta(G)$ of a goal G in the plane.

second key component of the planner is a set of techniques for chaining together motions to synthesize multi-step strategies. The planner, called LIMITED, synthesizes robot control programs with built-in sensor-based EDR.

While EDR is largely motivated by the problems of uncertainty and model error, its applicability may be quite broad. EDR has been a persistent but ill-defined theme in both AI and robotics research. We give a constructive, geometric definition for EDR strategies and show how they can be computed. This theory represents what is perhaps the first systematic attack on the problem of error detection and recovery based on geometric and physical reasoning.

1.9 Experimental Results

Since 1987, my group at Cornell has implemented an experimental force-control system on our PUMA robot. In particular, we implemented an approximate generalized damper. Our PUMA 560 robot arm has a VAL-II controller, a Multibus-based 68000 computer system running NRTX (a real-time variant of UNIX) for real-time control, an Astek 6-axis force-sensing wrist, and a Monforte tool-changing gripper with finger positioning and sensing. We closed the servo loop outside of VAL-II to obtain a servo rate we estimate as approximately 20-30 hz. Probably due to the inherent passive compliance in the joints and wrist, the robot can maintain compliant contact between a (gripped) steel bolt and an aluminum surface at speeds of 1-3 inches per second.

This experimental system was sufficiently stable to perform experiments executing the plans generated by LIMITED. In particular, we executed the plans and we performed all the assemblies shown in [Donald 87b, 89], chapter I, figs. 2-13. Again, the interesting thing about these plans is that they were generated by a machine, from task-level descriptions. Thus we were able to build a small slice of a task-level system, right down to the physical robot [Donald 88c, 90; Jennings, Donald and Campbell].

The peg-in-hole plans worked very well, even with tight (1 mm) clearances. The gear-meshing plans also worked, although we found it somewhat tricky to implement the failure mode analysis "oracle" that determines when sticking or breaking contact occurs using force- and position-sensing ([Donald 87b, 88c, 89, 90; Jennings, Donald and Campbell]). For both types of plans, we found that there were unmodelled effects (impact, bouncing) that could have an effect on plan execution, and unmodelled constants (the damping constants, the amount of force to maintain for sliding on a surface, the force termination thresholds) that must be chosen by the runtime executive. A more advanced planner would take these effects and constants into account. A more sophisticated runtime force-control plan executor might adaptively adjust these parameters to a local optimum. In addition, from an experimental viewpoint, much research is needed on the implementation and choice of termination predicates.

2 Mathematical Model

2.1 Research Issues

2.1.1 Guaranteed Strategies Under Bounded Control And Sensing Uncertainty

The gross motion planning problem with no uncertainty has received a great deal of attention recently. In this problem, the state of the robot may be represented as a point in a configuration space. Thus moving from a start to a goal point may be viewed as finding an arc in free space connecting the two points. Since the robot is assumed to have perfect control and sensing, any such arc may be reliably executed once it is found. In particular, given a candidate arc, it may be tested. That is, motion along the arc may be simulated to see whether it is collision free. For example, an algebraic curve may be intersected with semi-algebraic sets defining the configuration space obstacles. In the presence of uncertainty, however, we cannot simply simulate a motion strategy to verify it. Instead, we need some technique for simulating *all* possible orbits, or evolutions of the robot system, under any possible choice of the uncertain parameters. With sensing and control uncertainty, the state of the robot must be viewed as a subset of the configuration space. Motions, then, can be viewed as mappings between these subsets. Of course there are many such subsets! From this perspective, it is clear that a chief contribution of [LMT] has been to identify and give a constructive definition for a privileged class of subsets, called *preimages*, and show that it is necessary and sufficient to search among this class. This framework appears very promising for planning guaranteed motion strategies under

sensing and control uncertainty.

2.1.2 Uncertain Geometry

The [LMT] framework assumes no geometric model error. Donald [Don89] reduced the problem of planning guaranteed strategies with sensing, control, *and geometric model* uncertainty[5] to the problem of computing preimages in a (higher dimensional) generalized configuration space $C \times J$, where C is a configuration space representing the motion degrees of freedom for the robot, and J is a parameter space of model error. See sec. 2.3 for details and an example.

2.1.3 Tolerating Failure: Error Detection and Recovery

There are certain inadequacies with the planning model. The insistence that strategies be guaranteed to succeed is too restrictive in practice. To see this, observe that guaranteed strategies do not always exist. For example, in a peg-in-hole problem with model error (fig. 3) there may be no guaranteed strategy for achieving the goal, since the hole may be too small for some model error values. For these values the goal in configuration space does not exist. Because tolerances may cause gross topological changes in configuration space, this problem is particularly prevalent in the presence of model error. More generally, there may be model error values for which the goal may still exist, but it may not be reachable. For example, in a variant of the problem in fig. 6, an obstacle could block the channel to the goal. Then the goal is non-empty, but also not reachable. Finally, and most generally, there may be model error values for which the goal is reachable but not *recognizably* reachable. In this case we still cannot guarantee plans, since a planner cannot know when they have succeeded.

These problems may occur even in the absence of model error. However, without model error a guaranteed plan is often obtainable by back-chaining and adding more steps to the plan. In the presence of model error this technique frequently fails: for example, in the peg-in-hole problem with model error, this technique will not work since no plan of any length can succeed when the hole closes up.

This is why we investigate *error detection and recovery (EDR) strategies*, as introduced in sec. 1.8. Since EDR strategies enjoy the ability to tolerate failure in a way that is robust and mathematically formal, they also provide a theoretical cornerstone for *randomized* or *probabilistic robotic strategies*, as explored by Erdmann [Erd 89] and Goldberg and Mason [GM 89].

[5] We use the terms model error and model uncertainty interchangeably.

2.2 Preimages

2.2.1 Preliminaries

We now elaborate on the definition of preimages subject to the dynamic model introduced in sec. 1.4. After the initial definitions suggested by [LMT,Ma84], this model was proposed by Erdmann [E]. Section 2.2 reviews Erdmann's definitions for the simplest termination predicates. Now, there can be much discussion as to which dynamic model is appropriate for compliant robot control, and what sensor models and termination predicates are more realistic. These are important issues. It is our goal here, to choose a *particular* model, and show how its information content can be formalized, and how, under this model, plans may be analyzed and computed. While in fact I believe that such a model is a reasonable and implementable approximation to actual robot assembly systems, I don't intend to argue that here. It is our purpose here to adopt a simple model, and then pursue the purely mathematical questions relating to reachability, sensor information, recognizability, plan computation, and complexity.

Whereas in section 1 we restricted our consideration to the configuration space $\Re^2$, we now assume our configuration space C is a Lie group, and view control velocities as generators in the Lie algebra. Sec. 1.4 may be formalized by saying that the behavior of our system is specified by a first order differential equation, called the *damper equation*, which linearly relates forces and velocities:

$$\boldsymbol{F} = \boldsymbol{B}(\boldsymbol{v} - \boldsymbol{v}_0) \tag{1}$$

Here, $\boldsymbol{F}$ is the generalized reaction force on the robot, $\boldsymbol{B}$ is the damping matrix of the form $b\boldsymbol{I}$, $\boldsymbol{v}_0$ is the control velocity, and $\boldsymbol{v}$ is the actual velocity the system.[6] One may view the vector $\boldsymbol{F}$ as the net force acting on the robot due to other objects in the robot's environment. The reaction forces due to these objects are specified by coulomb's law. $\boldsymbol{v}_0$ is related to the commanded velocity $\boldsymbol{v}_\theta^*$ as follows: $\boldsymbol{v}_0$ lies in a error ball (or cone) $B_{ec}(\boldsymbol{v}_\theta^*)$ about $\boldsymbol{v}_\theta^*$. Here is how the ball B_{ec} represents error. The possible motion directions that an unobstructed (point) robot[7] may follow when commanded to move in direction $\boldsymbol{v}_\theta^*$ lie within the cone $B_{ec}(\boldsymbol{v}_\theta^*)$. The effect of uncertainty on equation (1) is therefore to substitute for $\boldsymbol{v}_0$ any of the velocities in the error cone about $\boldsymbol{v}_\theta^*$.

As Erdmann [E] points out, to predict the outcome of a commanded motion $\boldsymbol{v}_\theta^*$, we must solve the generalized damper equation (1). Given that the commanded velocity in this equation (1) is subject to uncertainty

$$\boldsymbol{v}_0 \in B_{ec}(\boldsymbol{v}_\theta^*), \tag{2}$$

the solutions are actually *classes* of trajectories. Motion along configuration space surfaces corresponds exactly to motion subject to a holonomic constraint: the damper equation (1) must be solved with contact for the surface in question. See fig. 11.

[6] All forces and velocities are generalized.

[7] Note it suffices to consider point robots in configuration space; see fig. 3.

We will assume that

1. The control velocity $\boldsymbol{v}_0$ can vary arbitrarily within $B_{ec}(\boldsymbol{v}_\theta^*)$.
2. Trajectories must be the result of integrating the first order damper equation (1):

$$\boldsymbol{p}(t_0) = \boldsymbol{p}(0) + \int_0^{t_0} \boldsymbol{v}(t)dt. \tag{3}$$

Definition 2.1 *A trajectory that* satisfies the damper equation with uncertainty relative to a commanded velocity $\boldsymbol{v}_\theta^*$ *is a mapping*

$$\begin{array}{rcl} T : [0,\infty) & \longrightarrow & TC \\ t & \mapsto & (T_p(t), T_v(t)) \end{array} \tag{4}$$

such that at all times t there exists a $v_0(t)$ satisfying (2) with

$$\boldsymbol{F}(t) = \boldsymbol{B}(\boldsymbol{v}(t) - \boldsymbol{v}_0(t)), \tag{5}$$

and (3) relates T_p and T_v.

Definition 2.2 *Define $T(\boldsymbol{v}_\theta^*)$ to be the set of all trajectories that satisfy the damper equation with uncertainty relative to $\boldsymbol{v}_\theta^*$.*

2.2.2 Sensing Model

Position sensing is specified as follows. For all times t, the sensed position $\boldsymbol{p}^*$ must lie within a ball $B_{ep}(\boldsymbol{p})$ about the actual position $\boldsymbol{p}$.

Now, given the damper equation (1), velocity- and force-sensing are equivalent. Hence we assume that for all times t, the sensed velocity $\boldsymbol{v}^*$ must lie within a ball $B_{ev}(\boldsymbol{v})$ about the actual position $\boldsymbol{v}$.[8]

2.2.3 Termination Predicate and Forward Projections

When the motion is initiated, the termination predicate knows the commanded velocity $\boldsymbol{v}_\theta^*$, the error bounds B_{ec}, B_{ep} and B_{ev}, and the set of goals to be reached, $\{ G_\alpha \}$. In addition, it also knows a bound on the initial conditions; in other words, it knows an initial set R such that the resulting trajectory must originate in R. In this survey, we do not assume that the termination predicate can "remember" past sensor values, although this model has been considered by Mason [Ma2] and Erdmann [E].

We assume that our termination predicate continuously monitors sensing values and time, and that it terminates the motion when the trajectory is guaranteed to have entered a goal. At that point the termination predicate returns which goal $G \in \{ G_\alpha \}$ has been

[8] For technical reasons, we must also assume symmetry, i.e., that $\boldsymbol{p} \in B_{ep}(\boldsymbol{p}^*)$ and $\boldsymbol{v} \in B_{ev}(\boldsymbol{v}^*)$ as well.

achieved. Hence, we view the computational role of the termination predicate as follows: at all times, the termination predicate computes the set of effective interpretations of its sensors, and checks to see whether this set is entirely contained within some goal. When containment is decided, the predicate returns.

In order to model this process mathematically, it is useful to speak of a trajectory as being *consistent* with a sensor value—or the other way around. Hence:

Definition 2.3 *A trajectory T is* sensor consistent at time t with reading $(\mathbf{p}^*, \mathbf{v}^*)$ *when*

$$(T_p(t), T_v(t)) \in B_{ep}(\mathbf{p}^*) \times B_{ev}(\mathbf{v}^*). \tag{6}$$

Since the termination predicate knows the controls, and an initial set R in which the motion is guaranteed to originate, it can construct a bound on all possible reachable states. This bound is called the *forward projection.*

Definition 2.4 *The* forward projection at time t of a set of initial conditions R, under commanded velocity $\mathbf{v}_\theta^*$ subject to uncertainty (2), *is given by*

$$F_\theta(R, t) = \{\, T(t) \mid T \in \mathcal{T}(\mathbf{v}_\theta^*) \text{ and } T_p(0) \in R \,\}. \tag{7}$$

Correspondingly,

Definition 2.5 *The* timeless forward projection *is*

$$F_\theta(R) = \bigcup_{t \geq 0} F_\theta(R, t). \tag{8}$$

2.2.4 Application: Computing Forward Projections

We now consider the computation of forward projections in various simple configuration spaces. Note that forward projections lie in phase space. Now, for a configuration space C, let $\pi : TC \rightarrow C$ be the canonical projection map. Buckley [Buc] developed the first algorithms in $C = \Re^3$ for computing the position component of the forward projection $\pi F_\theta(R)$. By the results of Canny and Reif [CR], these sets can have exponential size, and the following decision problem is computationally intractable:

Lemma 2.1 [CR]. *In the configuration space $\Re^3$, given a polyhedral start region R, a polyhedral environment $\mathcal{P}$ of n vertices, control uncertainty ϵ_c, coefficient of friction μ, and a polyhedral goal G, and a commanded motion $\mathbf{v}_\theta^*$, deciding containment of a point $\mathbf{x}$ in the forward projection $\pi F_\theta(R)$ is $\mathcal{NP}$-hard.*

This immediately implies the following

Corollary 2.2 [CR]. *The One-Step Compliant Motion Verification Problem (problem 2 in sec. 1.5) $(R, \mathcal{P}, \epsilon_c, \mu, G)$ is $\mathcal{NP}$-hard in $\Re^3$.*

However, the planar case turns out to be tractable. By generalizing results of Erdmann [E], Donald and Canny developed forward projection algorithms for that are optimal $(O(n \log n))$ in the plane [Don 89].

We now consider some computational complications of introducing rotations. Note that our discussion generalizes straightforwardly to the configuration space $\Re^3 \times SO(3)$ of rigid body euclidean motions. For the remainder of section 2.2.4, it it convenient to drop the boldface notation for points and tangent vectors to C.

In section 1 we considered the configuration space $\Re^2$ to develop intuition. Suppose we continue to assume the obstacles and peg are polyhedral, but we allow the peg to rotate as well as translate. This lifts the problem into the three dimensional manifold corresponding to the special euclidean group acting on the plane, which we will represent as $C = \Re^2 \times S^1$. This configuration space may be given coordinates (x, y, θ), or, if we desire an "algebraic" configuration space, we may use the coordinates (x, y, u) where $u = \tan\frac{\theta}{2}$. Configuration space obstacles in this space are 3-dimensional semi-algebraic sets (see figs. 8-10. from Brost's work [Brost 89]).[9] These obstacles are generated by one triangular moving "peg" A, and one triangular stationary "obstacle" B. The surfaces bounding these obstacles are ruled, algebraic surfaces; they are simultaneously linear in x and y, and quadratic in u. Each configuration space surface S, or "C-surface" in figs. 8-10 corresponds exactly to a particular holonomic constraint. In particular, each C-surface is "generated" by a particular contact type. That is, each C-surface is generated by the constraint that a vertex of A touch an edge of B, or that an edge of A touch a vertex of B. Note that $C = \Re^2 \times S^1$ is made into a Riemannian manifold by the choice of inner product as follows: If v is a tangent vector at $x \in C$, then $\langle v, v \rangle_x = v\boldsymbol{I}_x v$ where $\boldsymbol{I}_x$ is the moment of inertia tensor at x. This means that $\langle \cdot, \cdot \rangle_x$ is a quadratic form that computes the energy of the system, and hence is a "natural" choice in the sense of Lagrangian dynamics. Each C-surface S hence inherits this inner product. We write the normal to S at x under this inner product as N_x. N_x corresponds to the direction of the pure constraint force for S satisfying the D'Alembert-Lagrange principle.

In free space (the complement of the obstacles), the set of all possible motions simply fans out in a 3D cone. The dynamics of sliding on surfaces, even under generalized damper dynamics, is more complicated. In particular, the dynamics changes from point to point as a s.a. function of orientation. See fig. 11, which shows a typical C-surface S. The dynamics form a differential constraint at each point x on S. The differential constraint is a projection of the 3D velocity uncertainty cone $B_{ec}(v_\theta^*)$ onto the tangent space to S at x. This projection forms a 2D cone; the differential constraint specifies that for a trajectory satisfying the damper equation with uncertainty relative to v_θ^*, and subject to constraint S, that the velocity $T_v(t)$ at $x = T_p(t)$ must lie inside the projected cone. The projection mapping, and hence the cone, varies from point to point. Now, this projection mapping is given by the mechanics of coulomb friction; since friction changes in a non-linear way

[9] These figures are very similar to those in [Brost 89] but were kindly provided to me by the author and are reprinted with permission here.

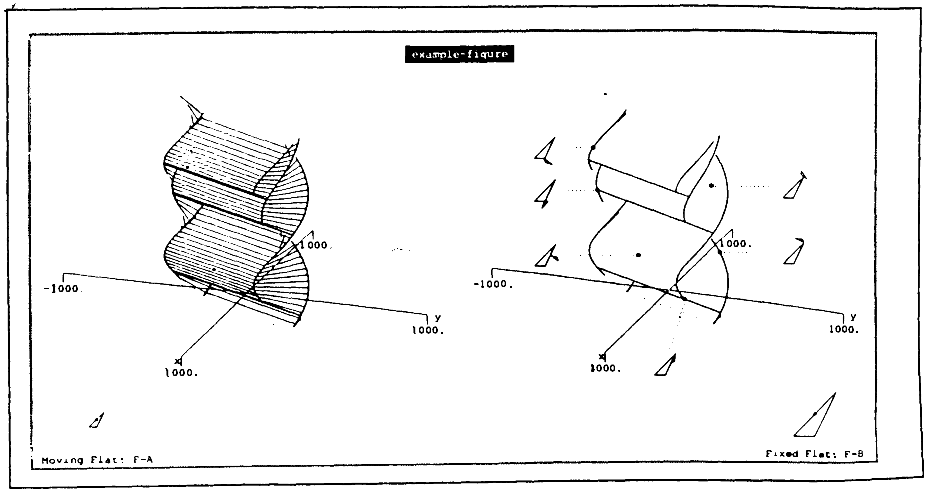


Figure 8: In the configuration space $\Re^2 \times S^1$, shows the configuration space obstacle for a small triangular "peg" A and a large triangular "obstacle" B. This figure shows how each C-surface bounding the obstacle is generated by a contact pair between a vertex (resp. edge) of A and an edge (resp. vertex) of B. Figs. 8-10 reprinted with permission from [Brost 89], and from new figures provided by R. Brost for this article.

over the surface,[10] the projection does also. Erdmann [E] gives the form of the projection map π_x, which is semi-algebraic (s.a.). We note that computing forward projections in $\Re^2 \times S^1$ has both a "dynamical systems" component (finding the outer envelope of all possible evolutions of the dynamical system π_x, subject to a holonomic constraint S), and a "combinatorial" component (there are a finite number of C-surfaces S that we can slide on).

Example: We derive the map π_x for the case of no friction. C is a parallelizable n-manifold, hence $TC \cong C \times \Re^n$. Let $\phi_x : T_eC \to T_xC$ smoothly identify each tangent space with the tangent space at the identity. Now, we may view $x \in S$ as a point in C, and hence identify T_xS with a subspace of T_xC, such that the normal N_x is orthogonal to T_xS. Now, define $p_x : T_xC \to T_xS$ to be the projection of T_xC onto T_xS along N_x. Then it is easily seen that $\pi_x = p_x \circ \phi_x$. With friction, the situation is somewhat more complicated, but the definition of π_x has the same spirit. See Erdmann [E].

[10] That is, the damper equation (1) during contact is non-linear.

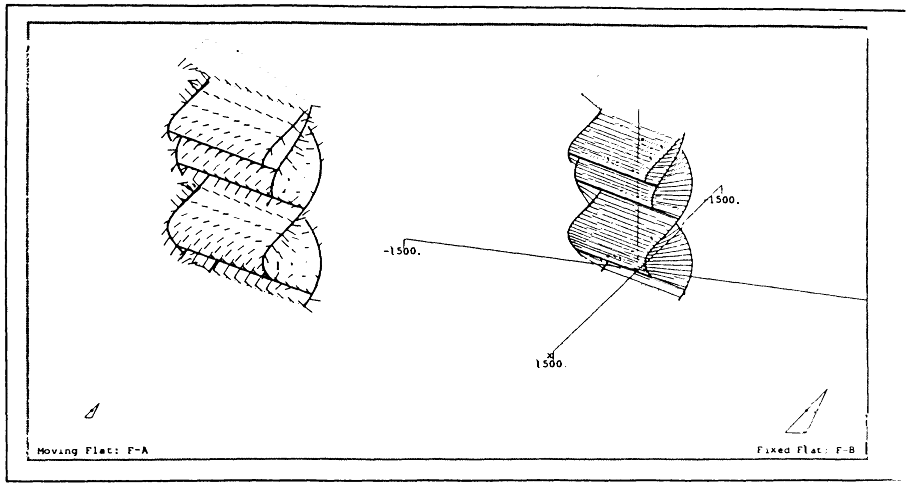


Figure 9: Normals to the C-surfaces. Each C-surface S is ruled and algebraic, and is simultaneously linear in x and y, and quadratic in $u = \tan \frac{\theta}{2}$.

Definition 2.6 *Let S be an algebraic surface in the configuration space C, and $R \subset S$ a set of initial conditions. Let $B_{ec}(v_\theta^*)$ be the control uncertainty cone in the tangent space T_eC to C at the identity. Let $\pi_x : T_eC \to T_xS$ be a projection map that varies with x and is semi-algebraic. Then the* forward projection $F_\theta(R)|_S$ of R under θ subject to S *is defined as follows:*

$F_\theta(R)|_S$ is the image of all trajectories T with $T_p(0) \in R$, such that at $T(t) = (x, v)$, with controls $v_0(t)$ satisfying (2), one of the following holds:

1. (Maintain contact): $\langle v_0(t), N_x \rangle_x \leq 0$ *and* $v \in \pi_x(B_{ec}(v_\theta^*))$, or
2. (Break Contact): $\langle v_0(t), N_x \rangle_x \geq 0$ *and* $v = v_0(t)$.

The forward projection with time subject to S is defined analogously. We can now state

Problem 5. Suppose S is a ruled, quadric C-surface in the configuration space $\Re^2 \times S^1$, as described above. Find combinatorially precise algorithms for computing $F_\theta(R)|_S$, and for deciding containment of a point in this forward projection.

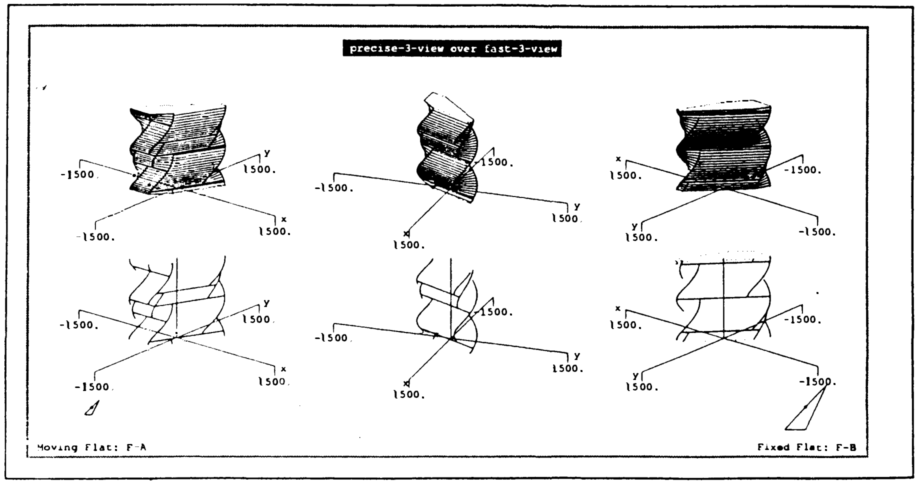


Figure 10: Three different "views" of the configuration space obstacles.

At this time, problem (5) is open, although Erdmann [E] and Brost [Brost 85] have given "numerical" approximation algorithms for backprojections, and Donald [Don 89] has given (similar) "numerical" algorithms for forward projections. In a recent breakthrough, Canny has developed a combinatorially precise algorithm for computing fully-general LMT multi-step strategies in the configuration space $\Re^3$, using the termination predicate in eq. (10). No such algorithms exist yet for $\Re^2 \times S^1$ or $\Re^3 \times SO(3)$, although there are approximation algorithms; some of these apply to restricted subproblems and are precise algorithms [Don88a, Briggs, FHS].

2.2.5 The Role of the Forward Projection in Preimages

Now, knowing where a motion started gives the termination predicate additional constraint on the set of effective sensor interpretations. For example, suppose that the position sensors read $\boldsymbol{p}^*$. *A priori*, this means that the actual position $\boldsymbol{p}$ can lie anywhere in the ball $B_{ep}(\boldsymbol{p}^*)$. However, suppose that the termination predicate has computed the forward projection $\pi F_\theta(R)$; this is an "upper bound" on all reachable positions. If part of the sensor ball $B_{ep}(\boldsymbol{p}^*)$ lies outside the forward projection, then those configurations are clearly excluded as effective sensor interpretations, since they are unreachable. More specifically, we have that

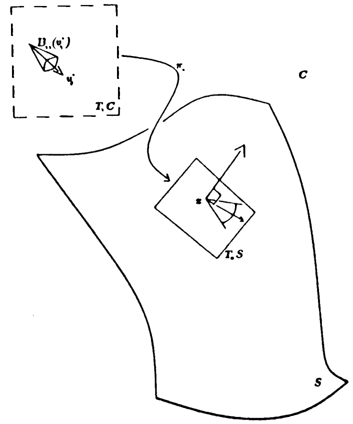


Figure 11: S is an algebraic surface in the lie group C. π_x projects control set B_r, from the Lie algebra of C onto the tangent space to S at x. π_x is a semi-algebraic map, (and s.a. parameterized by x). The projected velocity cone is a differential constraint that the velocity of admissible trajectories in TS must obey.

Proposition 2.3 *The effective interpretation of the sensor values $(\boldsymbol{p}^*, \boldsymbol{v}^*)$ at time t is*

$$F_\theta(R,t) \cap B_{ep}(\boldsymbol{p}^*) \times B_{ev}(\boldsymbol{v}^*). \tag{9}$$

2.2.6 Definition of Preimages

We can now define preimages. First, we define the "set of successful controls" $S(R, \{G_\alpha\})$:

Definition 2.7 *For a start region R and a set of goals $\{G_\alpha\}$, Define the set $S(R, \{G_\alpha\})$ as follows. $\boldsymbol{v}_\theta^* \in S(R, \{G_\alpha\})$ if and only if:*

For all trajectories $T \in \mathcal{T}(\boldsymbol{v}_\theta^*)$ consistent with the damper equation under uncertainty, there exists a time $t \geq 0$ such that for all sensor values $(\boldsymbol{p}^*, \boldsymbol{v}^*)$ consistent with $T(t)$, there exists some goal $G \in \{G_\alpha\}$, such that $\pi^{-1}G$ contains the set of effective sensor interpretations (9), that is,

$$F_\theta(R,t) \cap B_{ep}(\boldsymbol{p}^*) \times B_{ev}(\boldsymbol{v}^*) \subset \pi^{-1}G. \tag{10}$$

Then preimages are as follows:

Definition 2.8 *The* Preimage of a set of goals $\{G_\alpha\}$ with respect to a start region R, for a commanded velocity $\boldsymbol{v}_\theta^*$, *is defined as*

$$P_{R,\theta}(\{ G_\alpha \}) = \{ \boldsymbol{p} \in R \mid \boldsymbol{v}_\theta^* \in S(R, \{ G_\alpha \}) \}. \tag{11}$$

Lozano-Pérez, Mason, and Taylor [LMT] envisioned a back-chaining planner that would recursively compute preimages of a goal until the start region was contained in a preimage. For example, starting with a goal G_0, one first computes the preimage G_1 of G under θ_1. Then one computes the preimage G_2 of G_1 under θ_2, and so forth, until the start region R is contained in G_n. Then one executes the plan $(\theta_n, \ldots, \theta_1)$, and the termination predicate **(10)** is guaranteed to recognize entry into each successive subgoal G_i.[11]

One then asks, what characterizes preimage sets, and when are they suitable subgoals for the next level of backchaining? The first result is:

Theorem 2.4 *(Fixed-Point Theorem [LMT]). In order that $P_{R,\theta}(\{ G_\alpha \})$ be a suitable subgoal for the next level of backchaining, the equation*

$$R = P_{R,\theta}(\{ G_\alpha \}) \tag{12}$$

must hold.

Hence, we may view the problem of guaranteed motion planning with uncertainty as solving the preimage equation (12) for R. The preimage equation is a consequence of the following property of the preimage mapping:

Lemma 2.5 (Erdmann). *The mapping $P_{R,\theta}$ is idempotent, when viewed as a function of R, for a fixed collection of goals.*

Comment: (On distinguishable goal sets). Consider the following preimage equation:

$$P_{R,\theta}(\{ G, H \}) = R. \tag{13}$$

We say that the preimage (13) is taken *with respect to* R. (13) means that the preimage of the set of goals $\{ G, H \}$, with respect to commanded velocity v_θ^*, is all of R. Note that by def. 2.7, when we have a *set* of goals, the termination predicate must return *which* goal (G or H) has been achieved. This is different from $P_{\theta,R}(G \cup H)$, which means the termination predicate will halt saying "we've terminated in G or H, but I don't know which." The region R appears on both sides of (13) because the preimage depends on knowing where the motion started. Thus solving preimage equations like (13) for R is like finding the fixed point of a recursive equation.

[11] This example does not illustrate branching, or conditional plans.

2.2.7 Preimage Approximation Theory

Preimages exhibit several undesirable qualities that make for computational difficulties. First, in general, maximal preimages are not unique. Second, in some cases, maximal preimages do not even exist. This led Erdmann to pursue an agenda that we may term "preimage approximation theory", in which he attempted to find a preimage-like mathematical object that enjoyed the properties of maximality, uniqueness, and efficient computability. This object he termed the *backprojection.*

Example: Fig. 12 illustrates the difference between backprojections and preimages. Here the radius of position sensing uncertainty is greater than twice the diameter of the hole. Sliding occurs on all surfaces. Furthermore, we assume that the robot has no sense of time (i.e., no clock)—for example, it might be equipped with independent x and y contact sensors that only fire once each.

We formalize this sensor model by assuming that the robot has accurate force-sensing in the y-direction, and infinitely bad force-sensing in the x-direction, and that there is no friction. The start region R can be all of free space. The backprojection $B_\theta(G)$ strictly contains the preimage $P_{R,\theta}(G)$: while all points in the backprojection are guaranteed to reach G, the sensing inaccuracy is so large that the termination predicate cannot tell whether the goal or the left horizontal surface has been reached. Only from the preimage can entry into G be recognized.

We may define the *simple backprojection* $\overline{P_{R,\theta}}$ of a set of goals to be the preimage with perfect position- and velocity- sensing. We observe that every preimage is contained in a simple backprojection. Hence, backprojections only address goal *reachability*, whereas preimages capture reachability *and recognizability*. No notion of termination predicate is needed to formalize the notion of backprojection, although in our case, it is useful pedagogically to think of a termination predicate with "perfect" sensing. Erdmann [E] made precise the notion that "backprojections may be used to approximate preimages." The idea is as follows. First, Erdmann observed that whether or not a point is in a backprojection depends solely on the properties of the point, and not on the properties of neighboring points. That led him to define a "maximal" backprojection $\overline{M_\theta}$ as the union of "simple" backprojections:

$$\overline{M_\theta} = \bigcup_R \overline{P_{R,\theta}}(\{ G_\alpha \}) \tag{14}$$

Note that by idempotency, it suffices to consider only sets R such that $R = \overline{P_{R,\theta}}(\{ G_\alpha \})$. It is clear, therefore, that the union of simple backprojections is also a simple backprojection:

$$\overline{M_\theta} = \overline{P_{\overline{M_\theta},\theta}}(\{ G_\alpha \}). \tag{15}$$

Hence, the absence of termination predicates (or, equivalently, the assumption of perfect termination predicates) makes it possible to define a unique maximal set $\overline{M_\theta}$ such

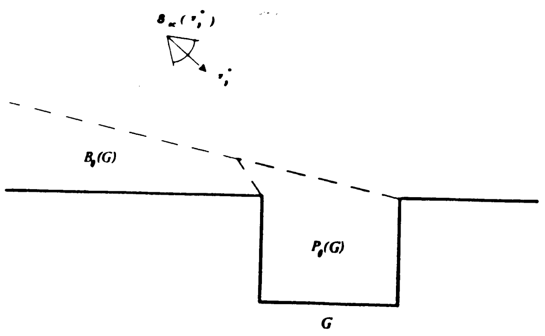


Figure 12: Here, the radius of the position sensing uncertainty ball is twice the width of the hole. Sliding occurs on all surfaces under the control velocities shown. Let the start region R be all of free space. The preimage of the goal under commanded velocity v_θ^* is $P_\theta(G) = P_{R,\theta}(G)$. The backprojection $B_\theta(G)$ strictly contains this preimage: while all points in the backprojection are guaranteed to reach G, the sensing inaccuracy is so large that the termination predicate cannot tell whether the goal or the left horizontal surface has been reached. Only from the preimage can entry into G be recognized.

that under controls v_θ^*, from $\overline{M_\theta}$ we are guaranteed to reach a goal $G \in \{ G_\alpha \}$. Erdmann called the mapping $\theta \times \{ G_\alpha \} \mapsto \overline{M_\theta}$ the *maximal backprojection*, denoted

$$B_\theta(\{ G_\alpha \}) = \overline{M_\theta}. \tag{16}$$

Hence we have

Lemma 2.6 *For all R, θ, and $\{ G_\alpha \}$.*

$$P_{R,\theta}(\{ G_\alpha \}) \subset \overline{P_{R,\theta}}(\{ G_\alpha \}) \subset B_\theta(\{ G_\alpha \}) = B_\theta\Big(\cup_\alpha\{ G_\alpha \}\Big). \tag{17}$$

2.2.8 Erdmann's Structure Equation

Now, Erdmann showed that every preimage $R = P_{R,\theta}(\{ G_\alpha \})$ may be "extended" to an "almost simple" preimage $A(R)$, where $A(R)$ contains R and has the general form

$$A(R) = \pi F_\theta(R) \cap B_\theta(E(R)), \tag{18}$$

where $E(R)$ is a subset of the union of the goals $\cup_\alpha\{ G_\alpha \}$. This says that given R, θ, and $\{ G_\alpha \}$, one may, in principle, "shrink" the goals to compute $E(R)$, and then compute $A(R)$ as in (18).

It remains to define $E(R)$, which Erdmann called the *First Entry Set*.

Definition 2.9 *For a trajectory T, let S_T be the set of successful termination times for R, θ, and $\{ G_\alpha \}$. That is, S_T is the set of all times $t \geq 0$ such that for all sensor values $(\mathbf{p}^*, \mathbf{v}^*)$ consistent with $T(t)$, there exists some goal $G \in \{ G_\alpha \}$, such that $\pi^{-1}G$ contains the set of effective sensor interpretations (eq. (9)).*

Definition 2.10 *Suppose $R = P_{R,\theta}(\{ G_\alpha \})$ is preimage. Define the* First Entry Set $E(R)$ *of R to be the set of all "first entry points" of trajectories that start in R. That is,*

$$E(R) = \{ T_p(t_0) \mid T \in \mathcal{T}(v_\theta^*), T_p(0) \in R, \text{ and } t_0 = \inf(S_T) \}. \tag{19}$$

Hence, we have that every suitable preimage R is a subset of an "almost simple" preimage" $A(R)$, and $A(R)$ is also a suitable preimage. We also have that $A(R)$, the extension of R, is "maximal" in the sense that A is idempotent. We call eq. (18) the *structure equation* for preimages and backprojections. On a practical note, while computing first entry sets may be in general, as hard as computing preimages, the utility of this definition is that many, easy-to-compute goal-shrinking schemes can quickly be proved correct. That is, a conservative goal-shrinking scheme is correct if it computes a subset G of $E(R)$. Then we may compute a preimage using the structure equation.

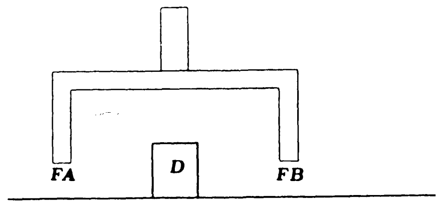


Figure 13: The grasp centering problem. The width of the block D on the table, and the position of the gripper are only known approximately.

2.3 Multi-Step Strategies

Having reduced motion planning under uncertainty to essentially "preimage-theoretic" equations, motion strategies can be synthesized by solving these preimage equations. We now present a worked-out example of a motion plan using preimages. The motion problem is grasp-centering for a robot gripper in the presence of model error. A guaranteed plan is found by solving the preimage equations. The example[12] illustrates the use of the preimage framework to derive a multi-step motion strategy in the presence of model error. The strategy employs time-sensing and force-sensing.

Consider the grasp-centering problem shown in fig. 13. The task is to center the robot gripper over the block D. The gripper can translate but not rotate in the plane. In its start position, the gripper is somewhere over D, such that the bottom of the fingers FA and FB are below the top of D. The width of D is unknown, but must be less than the distance between FA and FB. We assume D is fixed (it cannot be accidentally pushed).

Hence we can regard this as a planning problem with model error. C is taken to be the cartesian plane, and J, the space of model error, is a bounded interval of the positive reals. Our first question is, what does the generalized configuration space $C \times J$ look like? This is easily answered by considering the motion planning problem in fig. 14. The problem is to find a motion strategy for a point robot so that it can achieve a goal exactly

[12] This problem arose in discussions with Tomás Lozano-Pérez, John Canny, and Mike Erdmann. It appears in [Don89].

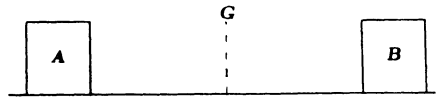


Figure 14: An equivalent problem. A point robot must be navigated halfway between the blocks A and B. The distance between A and B is not known. The robot has force sensing, and a clock. However, it has poor position sensing. We regard C as $\Re^2$ and J, the space of model error, as the bounded interval $(0, d]$ for d positive. The generalized configuration space for this problem is $\Re^2 \times J$, and is shown in fig. 18.

halfway between the blocks A and B. The distance α between A and B is unknown and positive. The point robot is known to start between A and B. Again, the point can translate in the plane. The distance α is the model error parameter. It is easy to see that the problems in figs. 13 and 14 are equivalent. The generalized configuration space for fig. 14 is, in fact, the same as that depicted in fig. 18, as we will see in sec. 2.4.

Next, we assume that the robot has perfect control, perfect velocity sensing, and a perfectly accurate sense of time. However, it has infinite position sensing error. [13]

Now, since the gripper starts over D with the bottom of the fingers below the top of D, and since the robot has perfect control, it suffices to consider the x axis of C. Since the y axis can be ignored, we develop our example in the plane, that is, in the generalized configuration space where C and J are both one-dimensional. This 2D generalized configuration space is shown in fig. 15, which is essentially an x-J cross-section of the full 3D generalized configuration space $\Re^2 \times J$ shown in fig. 18, holding y constant with α constrained to be positive. In fig. 15, L and R are left and right obstacle edge boundaries generated by A and B. The goal is the line in free-space bisecting L and R. The start region T is the triangular region in free-space between L and R. (T is the convex hull of L and R).

[13] This example is easily generalized to non-zero control, time-sensing, and force-sensing error, and finite position-sensing error. This requires giving the goal non-empty interior, however.

Now, since motion across J is not permitted, all motions are parallel to the x axis, that is to say, horizontal in fig. 15. There are only two kinds of motions the planner can command. Let $+$ denote a motion to the right, and $-$ a motion to the left. We assume the robot has perfect control over the magnitude as well as the direction of the commanded velocity.

See fig. 15. Now, if α is a point on the J axis, let E_α be the point on the left obstacle edge L with J coordinate α. We will denote the collection of all such points on L by $\{ E_\alpha \}$. Let S_α denote the maximal line segment within T containing E_α and parallel to G. Formally, if E_α has coordinates (x, α), then S_α is the line segment extending from E_α to (x, d) where d is an upper bound on the distance between A and B. We denote the collection of all lines S_α by $\{ S_\alpha \}$.

At this point we are prepared to derive a motion strategy for centering the grasp, that is, for attaining G from T. The strategy has three steps. The termination conditions for the motions involve time- and force-sensing. Here is the motion strategy in qualitative terms:

Strategy Guarantee-Center

1. *Command a motion to the right. Terminate on the right edge R based on force sensing.*

2. *Command a velocity of known magnitude to the left. Terminate when in contact with the left edge L, using force sensing. Measure the elapsed time of the motion. Compute the distance traversed. This gives exact knowledge of where the motion terminated on L. The effect of this step is to measure the distance α between the blocks.*

3. *Move distance $\frac{\alpha}{2}$ to the right, terminating in G based on time sensing.*

We now derive this strategy by solving the preimage equations for the motion planning problem.

First, note that if the run-time executive knows that the robot is inside a *particular* S_α, then G can be reliably achieved by commanding a motion to the right. Since the robot has perfect control and time sensing, the motion can be terminated after moving distance $\frac{\alpha}{2}$, that is, exactly when the line G is achieved. Using the preimage notation, we write this as

$$P_{+,\{S_\alpha\}}(G) = \{ S_\alpha \}. \tag{1}$$

Next, we take the collection $\{ S_\alpha \}$ as a set of subgoals, and try to find a motion that can recognizably attain this collection, and, furthermore, can distinguish which S_α the motion achieves. Consider a leftward motion starting from anywhere on the right edge R. The robot does not know where on R the motion starts, however. To recognizably achieve some S_α, such a motion should move leftward, and terminate when force-sensing

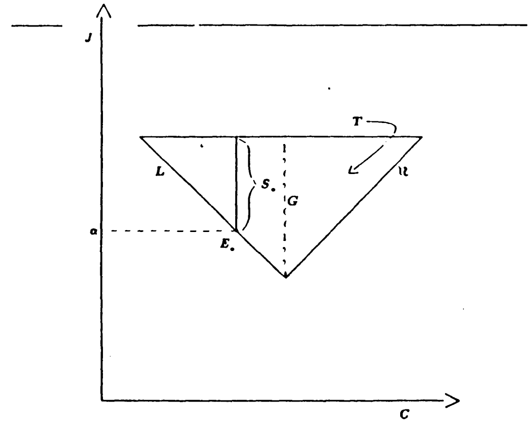


Figure 15: Assuming that the gripper fingers are initially lower than the top of the block D, the y dimension can effectively be ignored. This allows us to examine an x-J cross-section of the 3D generalized configuration space $\Re^2 \times J$ in fig. 18. We treat C as the x axis of motion freedom, yielding a 2D $C \times J$ planning space. L and R are obstacle boundaries in generalized configuration space . The goal is the bisector G between L and R in free-space. The start region T is the triangular region between L and R. E_α is a point on L. S_α is a line in T parallel to G and containing S_α.

indicates that L has been reached. If the termination predicate measures the elapsed time of the motion, and knows the magnitude of the commanded velocity, then it can recognize which point E_α has been reached, and hence which subgoal S_α has been achieved. Writing this down in preimage equations,

$$P_{-,R}(\{ S_\alpha \}) = P_{-,R}(\{ E_\alpha \}) = R. \tag{2}$$

Finally, the right edge R may be achieved from anywhere within the start region T by moving rightward, and terminating when force sensing indicates contact. This is simply

$$P_{+,T}(R) = T. \tag{3}$$

It is instructive to examine the termination conditions for motions (1)–(3). In developing the LMT framework for planning guaranteed strategies, [Erdmann] developed an elegant formalization of the question, "Using sensors and history, when can the termination predicate decide that a motion has recognizably entered a goal G_β?" The answer was as follows. Assume that G_β has been lifted into phase-space. Let R be the start region. The forward projection, $F_\theta(R)$ captures the notion of history: it is all positions and velocities that can be reached given that the motion started in R. At a particular instant t in time, let $B_{ep}(t)$ and $B_{ev}(t)$ be the sets of possible positions and velocities. These are the sensing uncertainty balls about a sensed position and velocity in phase space at time t. Thus sensing provides the information that the actual position and velocity must lie within the set $B_{ep}(t) \times B_{ev}(t)$. The forward projection further constrains the actual position and velocity to lie within $F_\theta(R)$. Thus the termination predicate can terminate the motion as having recognizably reached G_β when

$$F_\theta(R) \cap (B_{ep}(t) \times B_{ev}(t)) \subset G_\beta. \tag{$*$}$$

Now, let $F_\theta(R,t)$ denote the *time-indexed* or *instantaneous forward projection* of R under θ at time t. $F_\theta(R,t)$ denotes the set of positions and velocities that are possibly achievable at elapsed time t, under motion θ, given that the motion started in R. The termination predicate in this case monitors a clock, in addition to position and velocity sensors. In motion (1), only the time-indexed forward projection $F_+(S_\alpha, t)$ is relevant to deciding termination. The motion terminates when $F_+(S_\alpha, t) \subset G$. Motion (3) can be terminated using pure force sensing. It could also be terminated using time, since there exists some t for which $F_+(T,t) = R$. In motion (2), both force sensing and time are required to terminate within a distinguishable E_α. The general form of the termination condition for all three cases is as follows. The termination predicate has the form

$$F_\theta(U,t) \cap (B_{ep}(t) \times B_{ev}(t)) \subset G_\beta$$

for a goal G_β and a start region U. (Assume that all subgoals have been lifted into phase space). In our case, position sensing error is infinite, so $B_{ep}(t)$ is $C \times J$. Let us denote $(C \times J) \times B_{ev}(t)$ by the simpler expression $B_v(t)$. Then the termination conditions for motions (1)–(3) are as follows. For the first motion (3), to terminate, we must have

$$F_+(T,t) \cap B_v(t) \subset R. \tag{4}$$

For the second motion (2) to terminate, we must have

$$F_-(R,t) \cap B_v(t) \subset S_\alpha \tag{5}$$

for some S_α. We think of the termination predicate as "returning" this S_α. Finally, for termination of the last motion (1), we must have

$$F_+(S_\alpha) \cap B_v(t) \subset G, \tag{6}$$

where the S_α in (6) is the same as the one returned by the termination predicate after the second motion as the satisfying assignment for (5).

Finally, note that time is the source of some complexity in this example. This complexity might be removed by employing a distance sensor instead. The output of such a sensor could be modeled as position sensing in J. The sensing action in J would entail measuring the distance between A and B. This relaxes the assumption of no position sensing in the J dimensions, but such modification to the generalized configuration space framework is trivial. With this modification, B_{ep} is simply regarded as a product of a position sensing ball in C and a position sensing set in J.

This concludes the example. We have shown how to derive a multi-step guaranteed motion strategy in the presence of model error. The strategy was derived by solving the preimage equations in generalized configuration space for the motion plan. These preimage equations made the role of time- and force-sensing explicit in deriving conditions for distinguishable termination in a collection of subgoals.

2.4 Representing Model Error

The [LMT] framework assumes no geometric model error. Donald [Don89] reduced the problem of planning guaranteed strategies with sensing, control, *and geometric model* uncertainty to the problem of computing preimages in a (higher dimensional) generalized configuration space, which encodes model error as a kind of nonholonomic constraint. In fact, we used this method implicitly in the previous section 2.3. We now review the reduction by carefully examining some very simple planning problems with model error. These problems in fact motivate the definition of *Error Detection and Recovery (EDR) strategies*. Of course, this does not mean that EDR is limited to situations with model error.

2.4.1 A Simple Example: The Variable-Width Peg-In-Hole

Example 1: Consider fig. 16. There is position sensing uncertainty, so that the start position of the robot is only known to lie within some ball in the plane. The goal is to bring the robot in contact with the right vertical surface of A.

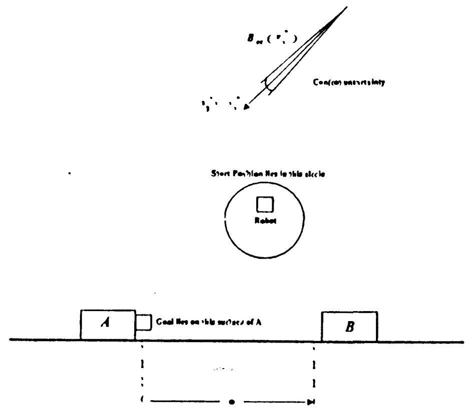


Figure 16: The goal is to bring the robot into contact with the right vertical surface of A. (For example, the "robot" could be a gripper finger). There is position sensing uncertainty, so in the start position the robot is only known to lie within some uncertainty ball. There is also control uncertainty in the commanded velocity to the robot. It is represented as a cone, as shown.

We will simplify the problem so that the computational task is in configuration space. This transformation reduces the planning task for a complicated moving object to navigating a point in configuration space. Consider fig. **17**. The configuration point starts out in the region R, which is the position sensing uncertainty ball B_{ep} about some initial sensed position. As usual, to model sliding behavior, we will assume Coulomb friction and generalized damper dynamics, which allows an identification of forces and velocities. Thus the commanded velocity v_0 is related to the effective velocity v by $f = B(v - v_0)$ where f is the effective force on the robot and B is a scalar. Given a nominal commanded velocity v_0^*, the control uncertainty is represented by a cone of velocities (B_{ec} in the figure). The actual commanded velocity v_0 must lie within this cone.

The goal in fig. **17** is to move to the region G. Now, with Coulomb friction, sticking occurs on a surface when the (actual) commanded velocity points into the friction cone. We assume the friction cones are such that sliding occurs (for all possible commanded velocities in B_{ec}) on all surfaces save G, where all velocities stick. We will assume that the planner can monitor position and velocity sensors to determine whether a motion has reached the goal. Velocity sensing is also subject to uncertainty: for an actual velocity v, the sensed velocity lies in some cone B_{ev} of velocities about v.

Now we introduce simple model error. The shape of A and B are known precisely, and the position of A is fixed. However, the position of B, relative to A is not known. B's position is characterized by the distance α. If $\alpha > 0$ the goal is reachable. But if

$\alpha = 0$, then the goal vanishes. No plan can be guaranteed to succeed if $\alpha = 0$ is possible. Suppose we allow α to be negative. In this case the blocks meet and fuse. Eventually, for sufficiently negative α, B will emerge on the other side of A. In this case, the goal "reappears," and may be reachable again.[14] Let us assume that α is bounded, and lies in the interval $[-d_0, d_0]$.

Our task is to find a plan that can attain G in the cases where it is recognizably reachable. Such a plan is called a *guaranteed strategy in the presence of model error*. But the plan cannot be guaranteed for the α where the goal vanishes. In these cases we want the plan to signal failure. Loosely speaking, a motion strategy which achieves the goal when it is recognizably reachable and signals failure when it is not is called an *Error Detection and Recovery (EDR) strategy*. Such strategies are more general than guaranteed strategies, in that they allow plans to fail.

To represent model error, we will choose a parameterization of the possible variation in the environment. The degrees of freedom of this parameterization are considered as additional degrees of freedom in the system. For example, in fig. **17**, we have the x and y degrees of freedom of the configuration space. In addition, we have the model error parameter α. A coordinate in this space has the form (x, y, α). The space itself is the cartesian product $\Re^2 \times [-d_0, d_0]$. Each *α-slice* of the space for a particular α is a configuration space with the obstacles A and B instantiated at distance α apart. Fig. **17** is such a slice.

More generally, suppose we have a configuration space C for the degrees of freedom of the moving object. Let J be an arbitrary index set which parameterizes the model error. (Above, J was $[-d_0, d_0]$). Then the *generalized configuration space* with model error is $C \times J$. One way to think of this construction is to imagine a collection of possible "universes", $\{ C_\alpha \}$ for α in J. Each C_α is a configuration space, containing configuration space obstacles. The ambient space for each C_α is some canonical C. $C \times J$ is simply the natural product representing the ambient space of their disjoint union. There is no constraint that J be finite or even countable.

In fig. **18** we show the generalized configuration space for example (**1**). Note that the goal in generalized configuration space becomes a 2-dimensional surface, and the obstacles are 3-dimensional polyhedra. Note that the goal surface vanishes where A and B meet.

Given a configuration space corresponding to a physical situation, it is well known how to represent motions, forces, velocities, and so forth in it (eg., see [Arnold]). The representations for classical mechanics exploit the geometry of differentiable manifolds. We must develop a similar representation to plan motions, forces, and velocities in generalized configuration space . We develop the following "axioms" for "physics" in $C \times J$.

1. At execution time, the robot finds itself in a particular slice of $C \times J$, (although it

[14] This model is adopted for the purposes of exposition, not for physical plausibility. It is not hard to model the case where the blocks meet but do not fuse.

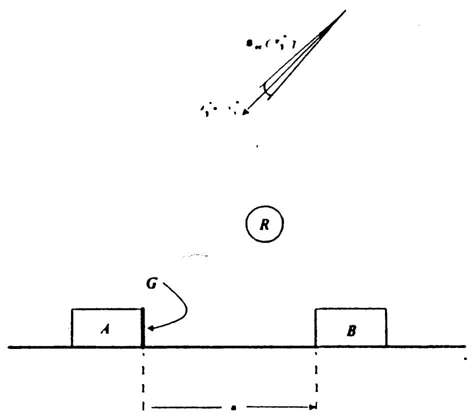


Figure 17: The equivalent problem in configuration space. The blocks A and B, the distance between the blocks α, and the commanded velocity $v_\theta = v_0^*$ with control error cone $B_{ec}(v_0^*)$. The position of A is fixed.

may not know which). Thus we say there is only one "real" universe, α_0 in[15] J. This α_0 is fixed. However, α_0 is not known *a priori*. Thus all motions are confined to a particular (unknown) α_0-slice, such as fig. 17. This is because motions cannot move between universes. In fig. 18, any legal motion in $C \times J$ is everywhere orthogonal to the J-axis and parallel to the x-y plane.

2. Suppose in any α-slice the position sensing uncertainty ball about a given sensed position is some set B_{ep}. The set R in fig. 17 is such a ball. We cannot sense across J: position sensing uncertainty is infinite in the J dimensions.[16] Thus the position sensing uncertainty in $C \times J$ is the cylinder $B_{ep} \times J$. In figs. 17,18, this simply says that x and y are known to some precision, while α is unknown. The initial position in fig. 17 is given by $R \times [-d_0, d_0]$. This cylinder is a 3-dimensional solid, orthogonal to the x-y plane and parallel to the J-axis in fig. 18.

3. Suppose in the configuration space C, the velocity control uncertainty about a given nominal commanded velocity is a cone of velocities B_{ec}. Such a cone is shown in fig. 17. This cone lies in the tangent bundle (or phase-space) TC of C. (Phase space is simply Position-space × Velocity-space. A point in phase space has the form (x, v), and denotes an instantaneous velocity of v at configuration x).

[15] α_0 is a point in the multi-dimensional space J.

[16] One generalization of the framework would permit and plan for sensing in J. In this case one would employ a bounded sensing uncertainty ball in the J dimensions.

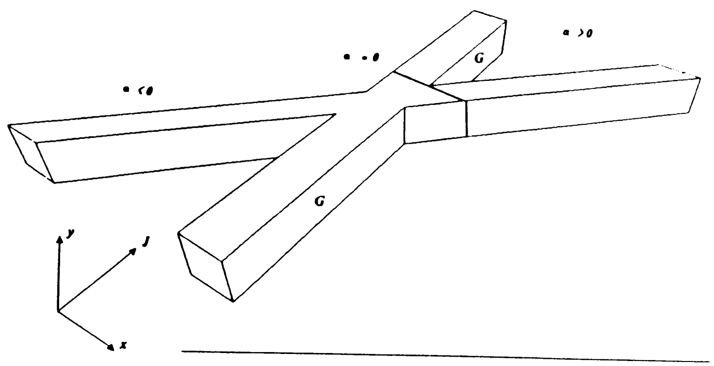


Figure 18: The generalized configuration space obstacles for example (1). The generalized configuration space is three dimensional, having x and y degrees of motion freedom, and an α degree of model error freedom. Legal motions are parallel to the x-y plane, and orthogonal to the J axis.

Phase space represents all possible velocities at all points in C. The phase space for $C \times J$ is obtained by indexing TC by J to obtain $TC \times J$. All velocities in generalized configuration space lie in $TC \times J$. For Ex. (1) $TC \times J$ is $\Re^1 \times [-d_0, d_0]$. The generalized velocity uncertainty cones are two-dimensional, parallel to the x-y plane, and orthogonal to the J axis.

4. Generalized damper dynamics extend straight-forwardly to $C \times J$, so motions satisfy $f = B(v - v_0)$ where f, v, and v_0 lie in $TC \times J$. Thus friction cones from configuration space (see [Erdmann]) naturally embed like generalized velocity cones in $TC \times J$.

These axioms give an intuitive description of the physics of $C \times J$. A formal axiomatization is given in [Don 89]. We have captured the physics of $C \times J$ using a set of *generalized uncertainties*, friction, and control characteristics (1-4). These axioms completely characterize the behavior of motions in $C \times J$.

2.4.2 Pushing

By relaxing axiom (1), above, we can consider a generalization of the model error framework, in which pushing motions are permitted, as well as compliant and gross motions. We relax the assumption that motion between universes is impossible, and permit certain motions across J. Consider example (3). Observe that a displacement in J corresponds to a displacement in the position of the block B. Thus a motion in J should correspond to a motion of B. Suppose the robot can change the position of B by pushing on it, that is, by exerting a force on the surface of B. The key point is that pushing operations may be modeled by observing that commanded forces to the robot may result in changes in the environment. That is, a commanded force to the robot can result in motion in C (sliding) as well as motion in J (pushing the block). Let us develop this notion further.

Our previous discussion assumed that motion across J was impossible. That is, all motion is confined to one α-slice of generalized configuration space. In example (3), this is equivalent to the axiom that B does not move or deform under an applied force. Such an axiom makes sense for applications where B is indeed immovable, for example, if A and B are machined tabs of a connected metal part. However, suppose that B is a block that can slide on the table. Then an applied force on the surface of the block can cause the block to slide. This corresponds to motion in J. In general, the effect of an applied force will be a motion which slides or sticks on the surface of B, and which causes B to slide or stick on the table. This corresponds to a coupled motion in both C and J, that is, a motion *across α-slices of generalized configuration space*. Such a motion is always tangent to a surface in generalized configuration space.

In [Don89], we generalize the description of the physics of $C \times J$ to permit a rigorous account of such motions. This model can then be employed by an automated planner. *Such a planner can construct motion strategies whose primitives are gross motions, compliant motions, and pushing motions.* This model of pushing is used in the gear-meshing example, where a model error parameter—the orientation of B—can be changed due to pushing.

2.4.3 Guaranteed Plans

Recall that a motion strategy is a commanded velocity v_θ^* together with a *termination predicate* which monitors the sensors and decides when the motion has achieved the goal. Given a goal G in configuration space, we can form its *preimage*. The preimage of G is the region in configuration space from which all motions are guaranteed to move into G in such a way that the entry is recognizable. That is, the preimage is the set of all positions from which all possible trajectories consistent with the control uncertainty are guaranteed to reach G recognizably.

Preimages provide a way to construct guaranteed plans for the situation with no model error. Can preimages and backprojections be generalized to situations with model error? The answer is yes. The generalized control and sensing uncertainties in $C \times J$ are given by the physics axioms above. These uncertainties completely determine how motions in generalized configuration space must behave. We form the backprojection of G under these uncertainties. The trick here is to view the motion planning problem with n degrees of motion freedom and k degrees of model error freedom as a planning problem in an $(n+k)$-dimensional generalized configuration space, endowed with the special physics described above. The physics is characterized precisely by axioms defining certain special

sensing and control uncertainties in $C \times J$. The definitions and results for pre-images and backprojections (sec. 2.2) in configuration space generalize *mutatis mutandis* to $C \times J$ endowed with this physics; this is proved in [Don89]. Thus our framework reduces the problem of constructing guaranteed motion strategies with model error to computing preimages in a somewhat more complicated, and higher-dimensional configuration space. For details, see [Don89].

2.5 Error Detection and Recovery

If we were exclusively interested in constructing guaranteed motion strategies in the presence of model error, we would be done defining the framework: having reduced the problem to computing preimages in $C \times J$, we could now turn to the important and difficult problems of computing and constructing $C \times J$, and computing preimages in generalized configuration space.

However, guaranteed strategies do not always exist. In example (1), (figs. 16–18) there is no guaranteed strategy for achieving the goal, since the goal may vanish for some values of α. Because tolerances may cause gross topological changes in configuration space, this problem is particularly prevalent in the presence of model error. In a peg-in-hole problem with model error (fig. 3) the goal may also vanish (the hole may close up) for certain regions in J. More generally, there may be values of α for which the goal may still exist, but it may not be reachable. For example, in a variant of the problem in fig. 6, an obstacle could block the channel to the goal. Then G is non-empty, but also not reachable. Finally, and most generally, there may be values of α for which the goal is reachable but not *recognizably* reachable. In this case we still cannot guarantee plans, since a planner cannot know when they have succeeded.

These problems may occur even in the absence of model error. However, without model error a guaranteed plan is often obtainable by back-chaining and adding more steps to the plan. In the presence of model error this technique frequently fails: in example (1), no chain of recursively-computed preimages can ever cover the start region $R \times J$. The failure is due to the peculiar sensing and control characteristics (1-4) in generalized configuration space .

In response, we will develop Error Detection and Recovery (EDR) strategies. These are characterized as follows:

- An EDR strategy should attain the goal when it is recognizably reachable, and signal failure when it is not.
- It should also permit serendipitous achievement of the goal.
- Furthermore, no motion guaranteed to terminate recognizably in the goal should ever be prematurely terminated as a failure.

- Finally, no motion should be terminated as a failure while there is any chance that it might serendipitously achieve the goal due to fortuitous sensing and control events.

These are called the "EDR Axioms", they will be our guiding principles. We now state how such EDR strategies may be constructed; for proofs and more detail, see [Don89].

For technical reasons (see [Don89]), in the EDR theory, we must construct preimages relative not to the initial set R, but rather to the forward projection $F_\theta(R)$. We will henceforth mercifully abbreviate $P_{F_\theta(R),\theta}$ by P_θ.

Suppose that a planning problem is given with two disjoint geometrical goals, G_1 and G_2. We may insist that the run-time executor be able to terminate the strategy and also be able to disambiguate *which* goal has been reached. In this case, we can construct the *preimage of the distinguishable union of* G_1 *and* G_2, which we write as

$$P_\theta(\{ G_1, G_2 \}).$$

If θ is executed starting in this preimage, then the motion can always be recognizably and distinguishably terminated in either G_1 or G_2.

We can characterize EDR strategies geometrically as follows. Suppose the geometric goal is G. The implicit "meaning" of G is: recognizably achieving G is equivalent to "success." We introduce an "additional" goal-like set H, which is disjoint and distinguishable from G, such that when H is recognizably achieved, then failure of the motion may be signalled. That is, we construct an H such that recognizably achieving H is equivalent to "failure." H is called the *EDR region*. Remarkably, H may be selected such that the EDR axioms are satisfied. In [Don89], we derive H as follows. Given a motion θ and a start region R, first we define H using reachability constructs only. Then we test whether H and G are distinguishable using sensors (this is the "formal test" alluded to in the prelude). If so, then by the construction of H, we have

$$R \subset P_\theta(\{ G, H \}).$$

Furthermore, using H, θ is a one-step EDR strategy satisfying the EDR axioms. Here is an idea of what H is like: In fig. 19 the EDR region H is shown (in position space only) for example (I). Consider H as a two-dimensional region in $C \times J$; just a slice of it is shown in fig. 19. Note that in this example, H only exists in the slices in which G vanishes. Here, given the sensing uncertainty bounds of example (I), the termination predicate can distinguish between G and H based on position sensing, velocity sensing, or elapsed time. Clearly, H satisfies the EDR axioms: the motion is guaranteed to terminate recognizably in G iff the motion began in a universe in which G does not vanish. Otherwise, the motion terminates recognizably in H. In the first case, the termination predicate signals success, in the latter, failure.

Here is how we construct H. Recall the *forward projection* $F_\theta(R)$ of a set R under θ is all positions and velocities that are possibly reachable from R under v_θ^* (subject to control uncertainty). Forward projections only address reachability: the termination predicate is

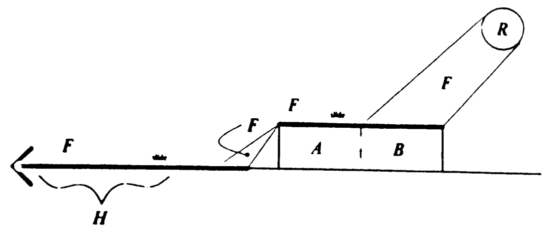


Figure 19: A typical α-slice of the EDR region H, for α small and negative. The goal vanishes in this slice; the dashed line indicates where the goal would be in other slices. Points in H lie within the forward projection (since they are reachable), yet outside the weak preimage (since the goal is unachievable).

ignored and only the control uncertainty bound and commanded velocity v_θ^* are needed to specify the forward projection.

So far the preimages we have considered are *strong* preimages, in that *all* possible motions are guaranteed to terminate recognizably in the goal. The *weak* preimage [LMT] (with respect to a commanded velocity) is the set of points which could *possibly* enter the goal recognizably, given fortuitous sensing and control events. See fig. 20. We will use the weak preimage to capture the notion of serendipity in the EDR axioms. The idea is that a motion may be terminated in failure as soon as egress from the weak preimage is recognized. The weak preimage is denoted $\hat{P}_\theta(G)$.

We define H_0 to be the set difference of the forward projection minus the weak preimage:

$$H_0 = \pi F_\theta(R) - \hat{P}_\theta(G). \tag{20}$$

Clearly, the motion θ can be terminated as a failure whenever H_0 has been reached, since H_0 is outside the weak preimage; hence the goal cannot be attained under θ from there.

We also define H_s to be all regions where sticking is possible in the weak minus strong preimage:

$$H_s = \{ x \in \hat{P}_\theta(G) - P_\theta(G) \mid \text{sticking is possible at } x \}.$$

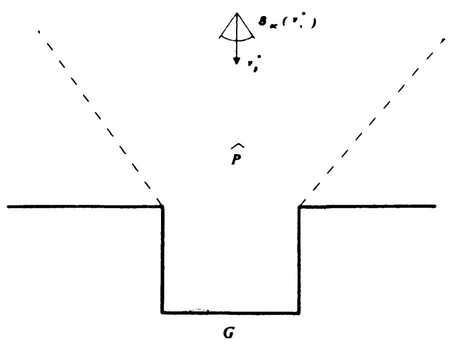


Figure 20: The *weak* preimage of the goal G under v_θ^*. Compare figs. 6 and 12.

See fig. 21. The motion θ should also be terminated as a failure if sticking occurs in H_s. We will decree that the robot has stuck in H_s if its velocity is zero for some duration, or "time-out" period.[17] More precisely, we define H to be a set in phase-space. First, note that H contains all phase-space points (x, v) where x is in H_0. Second, H contains points of the the form $(x, 0)$, where x is in H_s. Thus, viewing phase-space as the tangent bundle to generalized configuration space, H contains the "cylinder" $\pi^{-1}(H_0)$ of velocities over H_0, and the "zero section" $Z(H_s)$ of zero velocities over H_s:

$$H = \pi^{-1}(H_0) \cup Z(H_s).$$

This definition of H almost satisfies the EDR axioms—the only tricky point is that we cannot guarantee that after sticking in H_s for a long time, the robot cannot eventually slide into the goal. This may be handled in principle by introducing a time-out period by which the goal must be reached. That is, our definition of H satisfies the EDR axioms if the goal is specified in phase-space-*time* as the product of $\pi^{-1}(G)$ with a compact time interval.

2.5.1 The Preimage Structure of EDR Regions

Consider an equation such as (13) once more, viewing H as an EDR region:

[17] That is, the termination predicate halts the motion when the velocity is zero for some prespecified duration.

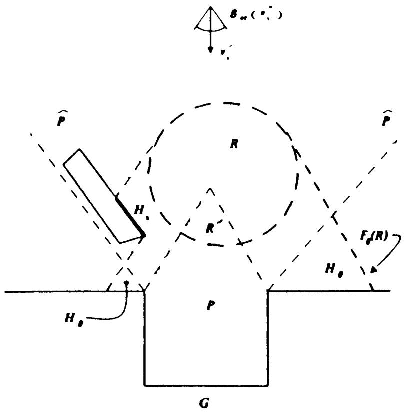


Figure 21: H_0 in eq. (20) is not the entire EDR region. Sticking may occur within the weak preimage in H_s. The EDR region must include H_0 for all possible velocities, and H_s for "sticking velocities."

$$R \subset P_{F_\theta(R),\theta}(\{G, H\}). \tag{21}$$

We remarked that in general, solving preimage equations like (13) and (21) for R is like finding the fixed point of a recursive equation. However, in the EDR application, we know R, H, and G, so (21) is a constraint which must be true, rather than an equation to solve. Presumably (21) is easier to check than to solve for R.

2.5.2 Application: Algorithms for EDR Strategies

In [Don89], we showed how to compute H in the domain of planar assemblies with model error. We also showed how to compute whether G and H are distinguishable. This is sufficient to generate one-step EDR strategies. These algorithms have been implemented in LIMITED. At a high level, the one-step algorithm is:

Algorithm Planar One-Step EDR

1. *Generate a commanded velocity v_θ^*.*
2. *Compute the EDR region H for v_θ^*.*
3. *Determine whether the EDR region H and the goal G are distinguishable using sensors. If so, then v_θ^* yields a one-step EDR strategy which recognizably terminates in G or H by monitoring position and force sensors.*

4. *Let* $\text{push}_\theta(G)$ *and* $\text{push}_\theta(H)$ *denote the* sticking push-forwards. *They are the set of obstacle edges within G and H, resp., on which sticking can occur under* v_θ^*. *Determine whether these regions are distinguishable using sensors. If so, then* v_θ^* *yields a one-step EDR strategy which recognizably terminates when sticking is detected.*

Here is how LIMITED decides the question, "Are G and H distinguishable using sensors?"

H and G are distinguishable using position sensing alone if their convolutions (Minkowski sums) by the position sensing error ball B_{ep} do not intersect.

Each obstacle edge of H and G has an associated configuration space friction cone. Two edges are distinguishable using force sensing if the convolutions of their friction cones by the force sensing uncertainty B_{ev} have a trivial intersection.

Similarly, the set of possible sensed reaction forces at an obstacle vertex w of G or H may be found by taking the direct sum of the friction cones of the edges cobounding w, and convolving by B_{ev}. Again, a vertex of H and a vertex (or edge) of G are distinguishable using force sensing if their associated cones of sensed reaction forces have a trivial intersection.

The procedure also works for determining the distinguishability of the push-forwards. Note that the procedure is correct for linear edges, where position- and force-sensing are separable, because the set of possible reaction forces is constant along an edge. For the general case, see [Don89].

2.6 Advanced Topics

Now, we have only addressed the planning of single-step EDR strategies. In order to plan multi-step EDR strategies, the techniques of sec. 2.3 may be employed. Multi-step strategy synthesis is complex, and the theory extends rather far beyond this paper.

We note that, in principle, having reduced both model error and EDR to essentially "preimage-theoretic" equations, multi-step strategies could be synthesized by solving these preimage equations. While this is proved or at least implicit in previous work [LMT,E,Ma2,D], it is far from obvious; furthermore, there are almost no published examples of such strategies. For this reason we presented a worked-out example of a motion plan using preimages in sec. 2.3. We could easily derive EDR as well as guaranteed strategies by solving the preimage equations [Don90].

Preimages are a key underlying tool for the geometric EDR theory, and the LMT framework is in some sense a "universal" method for synthesizing multi-step strategies. However, the technique of solving the preimage equations is not computational. For this reason, we have introduced a construction called the *push-forward* [Don89]. Roughly speaking, the push-forward is that subset of the forward projection where the motion can terminate. Since push-forwards address termination whereas forward projections do not, we may regard them as "dual" to preimages. That is, push-forwards are to forward projections as preimages are to backprojections. Second, the push-forward permits us to develop rather simple algorithms for planning multi-step strategies. These algorithms have been implemented in LIMITED. While the push-forward method for multi-step strategy synthesis is algorithmic, it is less general than the full preimage method (solving the preimage equations). We characterize the loss of power in push-forward algorithms in [Don89].

In [Don89], we presented two EDR plans generated by LIMITED. These were a peg-in-hole insertion strategy with model error, and a gear-meshing plan. Both were two-step plans. The peg-in-hole plan used push-forward techniques. The gear plan used a seemingly unrelated technique called *failure mode analysis*. However, there exists a view of multi-step strategies which essentially unifies all these techniques. This is called the "weak" EDR theory in [Don89]. The motivation behind this theory is that when a motion terminates ambiguously, a subsequent motion may be synthesized which disambiguates the success or failure of the first. Oddly enough, it is not necessary for either motion individually to satisfy the EDR axioms of sec. 2.5. However, when taken together, the two-motion plan can often be considered "equivalent" to a one-step EDR strategy.

The weak EDR theory effectively defines some laws of "composition" that permit two single-step plans to be concatenated into a two-step plan satisfying the EDR axioms. Hence it is often possible to construct multi-step plans that are EDR plans "globally" although not "locally". That is, considered as entire plans, they satisfy the EDR axioms; this is the "global" condition. However, "locally" they are not EDR plans, in that no single step is an EDR strategy. The key to pasting together non-EDR plans to make a global EDR strategy lies in defining certain local "niceness" conditions for how plans must mesh. These are called the *linking conditions.*

As the notation suggests, it is possible to formalize our view of "motions as mappings"—this notion is implicit in the term "preimage." To develop this viewpoint, one considers motions as a certain class of morphisms between distinguishable unions in the powerset of the tangent bundle to generalized configuration space . An EDR theory, then, is a covariant functor associated with a family of quotient maps of the form

$$\pi : TC \times J \to \{ P, \hat{P} - P, \tilde{F} - \hat{P} \}.$$

One focus for future work would be to push such a functorial viewpoint; any category-theoretic formulation of this flavor is still operational in nature and not yet fully abstract.

Acknowledgments: I am very grateful to Randy Brost for providing me with figures 8-10. Also, thanks to Tomás Lozano-Pérez, Matt Mason, Russ Taylor, Mike Erdmann, and John Canny for sharing with me the exciting research I survey in this paper.

3 References

Abraham, R. and Marsden, J. *Foundations of Mechanics*, Benjamin/Cummings, London (1978).

Arnold, V. I. *Mathematical Methods of Classical Mechanics*, Springer-Verlag, New York (1978).

[BK] Bajaj, C. and M. Kim, "Compliant Motion Planning with Geometric Models", *Proc. ACM Symposium on Computational Geometry*, Waterloo, 1987.

Brady, M. et. al. (eds). *Robot Motion: Planning and Control.*, Cambridge, Mass.: MIT Press. (1982).

Briggs, Amy *An Efficient Algorithm for One-Step Compliant Motion Planning with Uncertainty.* 5^{th} ACM Symposium on Computational Geometry, Saarbrucken, Germany. (1989).

Brost, R., "Automatic Grasp Planning in the Presence of Uncertainty", *IEEE International Conference on Robotics and Automation*, San Francisco, April, 1986.

Brost, R. *A State/Action Space Approach to Planning Robot Actions*, Forthcoming Ph.D. Thesis, Computer Science Dept., CMU (to appear).

Brost, R. *Computing Metric and Topological Properties of Confugurarion Space Obstacles*, IEEE Int. Conference on Robotics and Automation, Scottsdale, AZ (1989).

Buckley, S. J. *Planning and Teaching Compliant Motion Strategies*, Ph.D. Thesis. Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science. 1987. Also MIT-AI-TR-936 (1987).

Burridge, R., Rajan, V. T., and Schwartz, J. T. *The Peg-In-Hole Problem: Statics and Dynamics of Nearly Rigid Bodies in Frictional Contact*, IEEE ICRA, Raleigh, NC (1983).

[C1] **Canny, J.F.** *The Complexity of Robot Motion Planning*, Ph.D. Thesis, MIT Department of Electrical Engineering and Computer Science (1987).

[Can89a] **John Canny.** *On computability of fine motion plans.* , IEEE ICRA, Scottsdale, AZ, (1989)

[CD] Canny, J.F. and Donald, B. R. *Simplified Voronoi Diagrams*, Discrete and Computational Geometry **3** (3) pp. 219-236. (1988).

[CR] Canny, J., and J. Reif, "New Lower Bound Techniques for Robot Motion Planning Problems", *FOCS* (1987).

[D] Donald, B. R. *Robot Motion Planning with Uncertainty in the Geometric Models of the Robot and Environment: A Formal Framework for Error Detection and Recovery*, IEEE International Conference on Robotics and Automation, San Francisco, April (1986a).

Donald, B. R. *A Search Algorithm for Motion Planning with Six Degrees of Freedom.* Artificial Intelligence. 31 **(3)** (1987a).

Donald, B. R. *Error Detection and Recovery for Robot Motion Planning with Uncertainty.* Ph.D. Thesis, Dept. EECS, MIT Artificial Intelligence Laboratory, MIT-AI-TR 982 (1987b).

Donald, B. R. *The Complexity of Planar Compliant Motion Planning with Uncertainty.* Proc. ACM Symposium on Computational Geometry, Urbana, IL June, 1988. *To appear in* Algorithmica. (1988a).

[D] Donald, B. R. *A Geometric Approach to Error Detection and Recovery for Robot Motion Planning with Uncertainty.* Artificial Intelligence **37** (1-3), Dec. 1988. pp. 223-271. (1988b).

Donald, B. R. *Planning Multi-Step Error Detection and Recovery Strategies*, Proc. IEEE International Conference on Robotics and Automation, Philadelphia, PA April. 1988. (1988c).

Donald, B. R. *Error Detection and Recovery in Robotics*, Lecture Notes in Computer Science, Vol. 336, Springer-Verlag, New York (1989). (1989).

Donald, B. R. *Planning Multi-Step Error Detection and Recovery Strategies.* International Journal of Robotics Research, **3** (1), March, (In press). (1990).

Draper Laboratories, Fourth Annual Seminar on Robotics and Advanced Assembly Systems, Cambridge, Massachusetts. November, 1983.

Dufay, B., and J. Latombe, "An Approach to Automatic Robot Programming Based on Inductive Learning", in Brady, M., and R. Paul. *Robotics Research: The First International Symposium*, MIT Press, 1984.

Durrant-Whyte, H. *Concerning Uncertain Geometry in Robotics*. Intl. Workshop on Geometric Reasoning, Oxford, England, June (1986).

[E] Erdmann, M. *Using Backprojections for Fine Motion Planning with Uncertainty.* IJRR Vol. 5 no. 1 (1986).

Erdmann, M. *On Motion Planning With Uncertainty*. MIT AI Lab, MIT-AI-TR 810 (1984).

Erdmann, M. *On Probabilistic Robot Strategies*, Ph.D. thesis, MIT Department of EECS. MIT A.I. Lab, Cambridge (1989)..

Erdmann, M., and M. Mason, "An Exploration of Sensorless Manipulation", *IEEE International Conference on Robotics and Automation*. San Francisco. April, 1986.

[FHS] Friedman, J., J. Hershberger and J. Snoeyink *Compliant Motion in a Simple Polygon*, 5^{th} ACM Symposium on Computational Geometry. Saarbrucken, Germany. (1989).

[GM89] Goldberg, K., and Mason, M. *Bayesian Grasping*, Submitted to the IEEE Int'l. COnf. on Robotics and Automation, 1990 (1989).

Hopcroft, J. E., Schwartz, J. T., and Sharir, M. 1984 On the Complexity of Motion Planning for Multiple Independent Objects; *PSPACE*-Hardness of the "Warehouseman's Problem." *International Journal of Robotics Research.* 3(4):76–88.

Hopcroft J., and Wilfong G., "Motion of Objects in Contact," Int. Jour. Robotics Res. vol 4, no. 4, (1986).

Inoue, H., "Force Feedback in Precise Assembly Tasks", MIT Artificial Intelligence Laboratory, AIM-308, August, 1974.

J. Jennings, B. Donald, and D. Campbell *Towards Experimental Verification of an Automated Compliant Motion Planner based on a Geometric Theory of Error Detection and Recovery*, Proc. IEEE International Conference on Robotics and Automation, Scotsdale, Arizona (1989).

[LLS] Latombe, J.C., A. Lazanas, and S. Shekhar *Motion Planning with Uncertainty: Practical Computation of Non-Maximal Preimages*, IEEE/RSJ Intl. Workshop on Intelligent Robots and Systems, Sept. 4-6 (1989).

Laugier, C., "A Program for Automatic Grasping of Objects with a Robot Arm", *Eleventh Symposium of Industrial Robots*, Japan Society of Biomechanisms and Japan Industrial Robot Association, 1981.

Lieberman, L., and M. Wesley, "AUTOPASS: An Automatic Programming System for Computer Controlled Mechanical Assembly", *IBM Journal of Research Development*, Vol. 21, No. 4. 1977, pp. 321-333.

Lozano-Pérez, T., "The Design of a Mechanical Assembly System", S.M. dissertation, MIT Department of Electrical Engineering and Computer Science, also AI-TR-397, MIT Artificial Intelligence Laboratory, 1976.

Lozano–Pérez, T. *Automatic Planning of Manipulator Transfer Movements*, IEEE Trans. on Systems, Man and Cybernetics (SMC-11):681-698 (1981).

[LoP] Lozano–Pérez, T. *Spatial Planning: A Configuration Space Approach*, IEEE Trans. on Computers (C-32):108-120 (1983a).

Lozano-Pérez, T., "Robot Programming", *IEEE Proceedings*, 1983b.

Lozano-Pérez, "A Simple Motion Planning Algorithm for General Robot Manipulators," in Proceedings of Fifth National Conference for the American Association of Artificial Intelligence, Philadelphia, 1986, pp. 626-631.

[LMT] Lozano-Pérez, T., Mason, M. T., and Taylor, R. H. *Automatic Synthesis of Fine-Motion Strategies for Robots*. Int. J. of Robotics Research, Vol 3, no. 1 (1984).

Lozano-Pérez, T., and Wesley, M. A. *An algorithm for planning collision-free paths among polyhedral obstacles*. Communications of the ACM (22):560-570 (1979).

Lumelsky, V. J. *Continuous Motion Planning in Unknown Environment for a 3D Cartesian Robot Arm.* , *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pp. 1569-1574. (April 7-10, San Francisco, Calif.) (1986).

Mason, M.T. *Compliance and force control for computer controlled manipulators*. IEEE Trans. on Systems, Man and Cybernetics (SMC-11):418-432 (1981).

Mason, M. T. 1986. Mechanics and Planning of Manipulator Pushing Operations. *International Journal of Robotics Research* 5(3).

[Ma2] Mason, M. T. *Automatic Planning of Fine Motions: Correctness and Completeness*. 1984 IEEE International Conference on Robotics. Atlanta Ga. (1984).

Natarajan, B. K. 1986 (Oct. 27-29, Toronto, Ontario). An Algorithmic Approach to the Automated Design of Parts Orienters. *Proceedings of the 27th Annual IEEE Symposium on Foundations of Computer Science*, pp. 132-142.

Ohwovoriole, M., and B. Roth, "A Theory of Parts Mating For Assembly Automation", *Proceedings of the Robot and Man Symposium 81*, Warsaw, Poland, September 1981.

Peshkin, M., "Planning Robotic Manipulation Strategies for Sliding Objects", Ph.D. dissertation, Department of Physics, Carnegie-Mellon University, 1986.

Raibert, M., and J. Craig, "Hybrid Position/Force Control of Manipulators", *Journal of Dynamic Systems, Measurement, and Control*, No. 102, June, 1981, pp. 126-133.

Schwartz J., Hopcroft J., and Sharir M., "Planning, Geometry and Complexity of Robot Motion Planning", Albex Publishing Co., New Jersey, (1987).

Schwartz J. and Yap C. K., "Advances in Robotics," Lawrence Erlbaum associates, Hillside New Jersey, (1986).

Simunovic, S. N., "An Information Approach to Parts Mating", Ph.D. dissertation, Department of Electrical Engineering, Massachusetts Institute of Technology, 1979.

Taylor, R. H. *The Synthesis of Manipulator Control Programs from Task-level Specifications*, Stanford Artificial Intelligence Laboratory, AIM-282, July (1976).

Turk, M., "A Fine-Motion Planning Algorithm", *SPIE Conference on Intelligent Robots and Computer Vision*, Cambridge, Massachusetts, September, 1985.

Udupa S., "Collision Detection and Avoidance in Computer Controlled Manipulators". Proc. 5th Int. Joint. Conf. on Art. Intell., Mass. Inst. Tech. (1977) pp 737-748.

Valade, J., "Automatic Generation of Trajectories for Assembly Tasks", *Sixth European Conference on Artificial Intelligence*, Pisa, Italy, September, 1984.

[Whi76] Whitney, D., "Force Feedback Control of Manipulator Fine Motions", *Journal of Dynamic Systems, Measurement, and Control*, June, 1977, pp. 91-97.

[Whi82] **Whitney, D.**, "Quasi-Static Assembly of Compliantly Supported Rigid Parts", *Journal of Dynamic Systems, Measurement, and Control*, Vol. 104, March 1982.

Department of Computer Science, Cornell University, Ithaca, NY 14853.