COMPLIANCE AND FORCE CONTROL FOR COMPUTER CONTROLLED MANIPULATORS

by

MATTHEW THOMAS MASON
Massachusetts Institute of Technology
April, 1979

Revised version of a thesis submitted to the Department of Electrical Engineering and Computer Science on May 19, 1978 in partial fulfillment of the requirements of the degree of Master of Science.

ABSTRACT

Compliant motion occurs when the manipulator position is constrained by the task geometry. Compliant motion may be produced either by a passive mechanical compliance built in to the manipulator, or by an active compliance implemented in the control servo loop. The second method, called *force control*, is the subject of this report. In particular, this report presents a theory of force control based on formal models of the manipulator and the task geometry. The *ideal effector* is used to model the manipulator, and the task geometry is modeled by the *ideal surface*, which is the locus of all positions accessible to the ideal effector. Models are also defined for the goal trajectory, position control, and force control.

These models are useful in two respects. First, the model of force control provides a precise semantics for force control primitives in manipulator programming languages. The model defines a simple interface between the manipulator and the programmer, isolating the programmer from the fundamental complexity of low-level manipulator control.

Second, the formalism provides a method of synthesizing force control programs for compliant motion. A force control program is modeled as a set of equations in the components of manipulator velocity (angular velocity) and force (torque). The task geometry constraints can be modeled similarly. The control program synthesis problem is to construct a program whose equations, when combined with the task geometry equations, have a single unique solution equal to the goal trajectory. We show that to obtain good performance in the presence of planning model error, it is also desirable that these two sets of equations be orthogonal and non-redundant.

Thesis Supervisor:........................... Berthold K. P. Horn, Associate Professor

ACKNOWLEDGEMENTS

This work would not have been accomplished without the help of Tomas Lozano-Perez, who gave freely of his time to read drafts and discuss the problems with me. Marc Raibert was instrumental in encouraging my interest in manipulation. Discussions with Berthold Horn, John Hollerbach, John Purbrick, and Patrick Winston were also very helpful. Special thanks to Mary Mason for financial and moral support, and for drawing the figures.

CONTENTS

1. Introduction

1.1 Overview and Motivation

A computer-controlled mechanical manipulator is a tool of fantastic potential, but so far the applications have been disappointing both in number and in sophistication. Current applications of industrial manipulators are for the most part limited to large-motion tasks such as material transfer, spray-painting, and spot-welding. These operations are accomplished by moving along a pre-defined sequence of points without any variation from one cycle to the next. This method is effective in many cases, but a variety of potential applications, notably automatic assembly, require more sophisticated use of the manipulator. Successful development of such applications awaits progress in several aspects of manipulation including compliant motion, manipulator programming languages, and automated task planning. This report will focus on the first of these, compliant motion.

"Compliant motion" occurs when the position of the manipulator is constrained by the task. When sliding along a tabletop, for instance, downward motion of the manipulator is prevented. Another example is the opening of a drawer, where no rotation is possible and translation is possible only along the drawer's axis. Compliant motion is an important part of a mechanical assembly system, since fitting parts together generally requires motion between objects in contact. Ideally these operations should be performed quickly but without producing excessive forces at points of contact.

There are two primary methods for producing compliant motion: a passive mechanical compliance built in to the manipulator, or an active compliance implemented in the software control loop, <u>force control</u>. Passive compliance offers some performance advantages, but the force control method offers the advantage of programmability. This allows the manipulator to choose the particular form of compliance necessary for the particular application. Using force control, a single manipulator can both open a drawer and slide on a tabletop without pausing to change hands.

Various force control systems have been implemented, but development of an underlying theory of these systems has been neglected. This report approaches such a theory by constructing models of force control and of manipulator task geometry. These models can be useful in two ways. First, a force control system can be evaluated according to how well it agrees with the formal model. This is common in the case of position control, where a control system is judged by its "error" — the difference between the desired and actual trajectories. Second, a formal model greatly simplifies manipulator programming. Effective programming occurs only when the programmer has a thorough understanding of the programming language primitives. Hence, the relevant primitives should be well-defined, which in the case of compliant motion primitives requires a model of force control.

1.2 Position and Force Control

The development of a model for general force control is conveniently approached by discussing models for simpler control modes, <u>pure position control</u> and <u>pure force control</u>. The model for a general force control system will be intermediate between these two.

Pure position control is a transducer whose input is in symbolic form and whose output is encoded as the position and orientation[1] of the end-effector in some pre-determined coordinate system. Input and output are just different encodings of the same six-dimensional vector function of time **p**(t). In pure position control the user is allowed to completely specify the effector position trajectory. Pure force control can be defined in an analogous way — the user provides a vector function **f**(t) which specifies the forces to be exerted by the end effector.

Each of these models is appropriate only under particular conditions, which can be deduced from simple mechanical considerations. For example, if pure position control is applied to the problem of sliding a drawer, then the position of the end-effector is subject to constraints from the task configuration and from the control system simultaneously. Due to the inevitable errors in manipulator planning and control, it is unlikely that these simultaneous constraints will be consistent. Our model of pure position control will not be satisfied in such a case, for the manipulator will not be free to follow the desired trajectory.

This situation is illustrated by considering the conceptual extremes. If the manipulator tip is buried in an immobile stiff solid substance, then there is no positional freedom at all, and pure position control is meaningless. However, the manipulator will have complete force freedom, since any force it wishes to exert will be "accepted" by the solid. In this instance, pure force control of the manipulator is appropriate. Now consider the opposite extreme, with the manipulator in free space. The situation is completely reversed. There is no force freedom, because there is no possible source of the required reactive force. Since there is no constraint on manipulator position, pure position control is now

1. From now on the term "position" may be used to refer to both position and orientation. Likewise "force" may refer to both force and torque.

indicated. It appears that pure position and pure force control are in some sense dual concepts, and that the historical emphasis on position control is the natural result of applications which involve very little physical contact.

Intermediate between the extremes of solid space and free space are surfaces. Loosely speaking, a surface is a task configuration which allows only *partial* positional freedom. Freedom of motion occurs along surface tangents, while freedom of force occurs along surface normals. Neither pure position nor pure force control is appropriate in this case, but rather a hybrid mode of control, which gives control of effector force along the surface normal and control of effector position along the surface tangent.

To relate these issues to real-world manipulator control systems, consider the task of manipulating a round peg in a hole (figure 1.1). The peg's motion is constrained by the hole. Assuming a snug fit, the peg can rotate about its own axis or translate up or down along the axis. If a position control system is in use, the motion of the peg and the force exerted between the peg and hole are unpredictable. Any motion commands are very unlikely to satisfy the physical constraints, for the program's model of the task will inevitably contain some error, however slight. The effect depends critically on the details of the control servo and on the presence of mechanical compliance in the manipulator. In some cases the inconsistency between the program constraints and the task constraints will produce large forces between the peg and hole, ultimately disturbing the task configuration or even damaging the manipulator.

If the program may use force and position control simultaneously, the insertion becomes feasible. The program may specify that the forces and torques normal to the surface be zero during the motion. The control system can satisfy such a command only by producing small corrective motions that compensate for

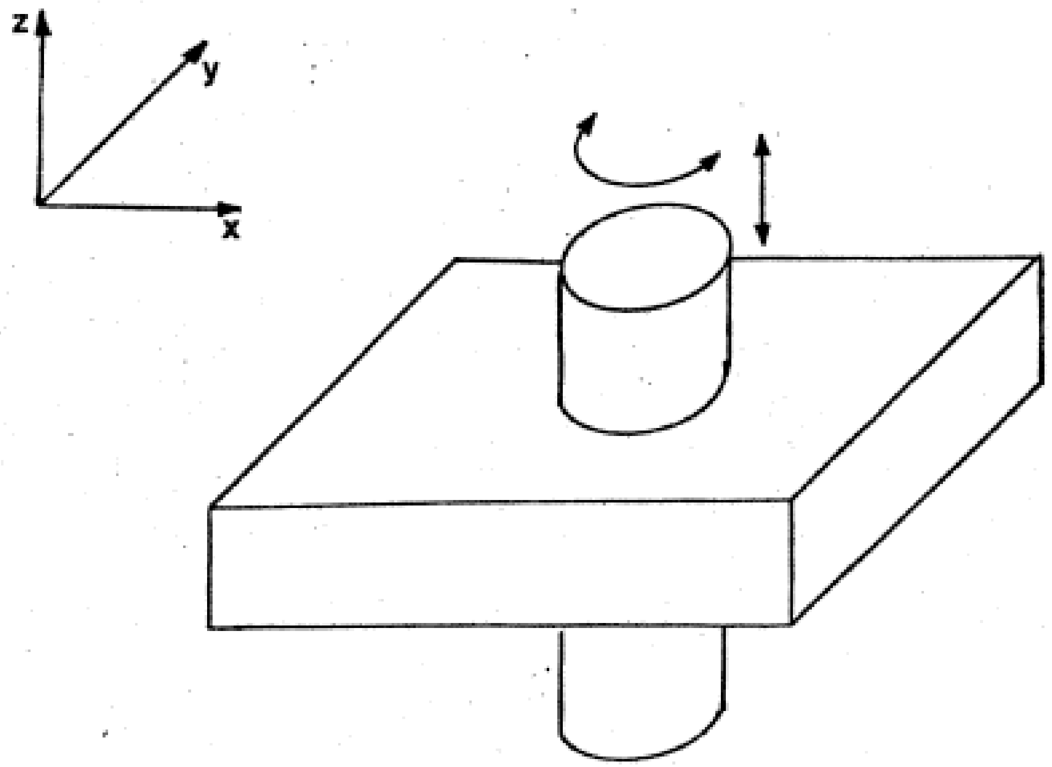

Figure 1.1. Insertion of a Peg in a Hole.

the inaccuracies of the position commands [Inoue74].

1.3 Control Strategies and Programming Language Primitives

Programming is often simplified by constructing the program hierarchically, with code at each level written using primitives defined at a lower level. For manipulator programming, the lowest level provides the interface with the manipulator sensors and actuators. Each primitive at this level constitutes a <u>control</u> <u>strategy</u>, i.e. a particular convention for combining input from higher level code and from sensors to produce signals for the actuators. With every control strategy is associated a set of assumptions on the task configuration, and a particular function, often formulated as some aspect of the manipulator's behavior to be regulated. We have already seen two examples of control strategies — pure position control and pure force control. The assumptions associated with these control strategies were the absence of physical contact and the presence of it, respectively.

Another control strategy in common use is the guarded move [Will75], which is used to approach and touch a surface without producing excessive force after contact is made. This is achieved by moving toward the surface slowly while closely monitoring a sensor which can detect contact. Guarded move control strategies and compliant motion control strategies are complementary, if the two classes are suitably generalized. In chapter two we will present a formal definition of surface which includes solid space or free space as well as the locus of possible positions of a drawer or of a peg in hole. Compliant motion occurs when the manipulator trajectory traverses such a surface, and a guarded move occurs when the manipulator trajectory crosses the junction between two such surfaces. Hence a manipulator trajectory can be decomposed into a sequence of compliant motions joined together by guarded moves.

The use of control strategies provides a convenient structure for the low-level control system without unduly compromising flexibility. In addition, the task state assumptions can be "compiled" into the control strategy to produce a very efficient program. Since each control strategy is used for a particular manipulator function and a particular type of task configuration, the control strategies must be switched in and out of the control loop by a higher-level process.

There are three primary criteria for a useful control strategy. First, of course, is that its function be relevant to the task. Second is that it execute quickly, since it is generally executed at least twenty times per second. Third, the resulting behavior of the manipulator must be conceptually simple. This is important from a planning and modeling viewpoint. In many respects the behavior of the manipulator itself will be the most difficult component of the task configuration to model, due to its great versatility. The role of the control strategy is to mask this complexity from higher levels of code. The control strategy model is the semantic component of a primitive of the manipulator programming language. Hence there are two characterizations of the problems addressed in this report: the synthesis of control strategies for compliant motion; and development of semantics of force control primitives for a manipulator programming language.

1.4 Example and Overview of Control Strategy Synthesis

Consider again the problem of manipulating a peg in a hole (figure 1.1). A program for this task might look like

```
MOVE TO D WITH
   BEGIN
      FORCE X 0
      FORCE Y 0
      TORQUE X 0
      TORQUE Y 0
   END
```

where D is the goal position of the peg. The coordinate axes specified by force (torque) statements are the compliant axes, translation and rotation of x and y in some pre-determined cartesian coordinate system. The remaining axes, translation and rotation of z, are the non-compliant axes. During execution of the move, the non-compliant axes will be position servoed using a trajectory computed from the move statement, while the compliant axes are force servoed using the force (torque) statements.[1] As long as frictional forces are negligible, the program can be executed without significant forces between peg and hole.

A simple observation will assist in formulating a method for synthesizing such programs. The peg-in-hole has two degrees of freedom, which we may choose to be translation along and rotation about the z-axis. The non-compliant axes are also z-translation and z-rotation, precisely the same as the degrees of freedom.[2] This important property leads to a simple method for synthesizing force control programs — we simply write a move statement specifying the goal position of the end-effector with a force (torque) statement for each axis orthogonal to the degrees of freedom. The control strategy synthesis method developed in succeeding chapters is analogous to the method of this example, but is appropriate for a very general class of task configurations. This generality is obtained by viewing a task configuration in terms of the resulting constraints on motion of the manipulator end-effector.

The main goal of this report is to formulate in precise terms a general method of control strategy synthesis. The logical structure of the report is indicated in figure 1.2. In order to develop the method in a well-defined manner, we will deal with abstract models of the task configuration, the manipulator, and the goal trajectory. These models are the components of the ideal domain. The models and the mechanics of their interaction are well-defined in the ideal

1. For the present it is necessary to appeal partly to intuition for the meaning of this program.
2. The choice of degrees-of-freedom is not the only one possible. For example, we might also have chosen the difference between z-translation and z-rotation, and the sum of z-translation and z-rotation. In any case, the plane spanned by the degrees-of-freedom is the same as that spanned by the non-compliant axes.

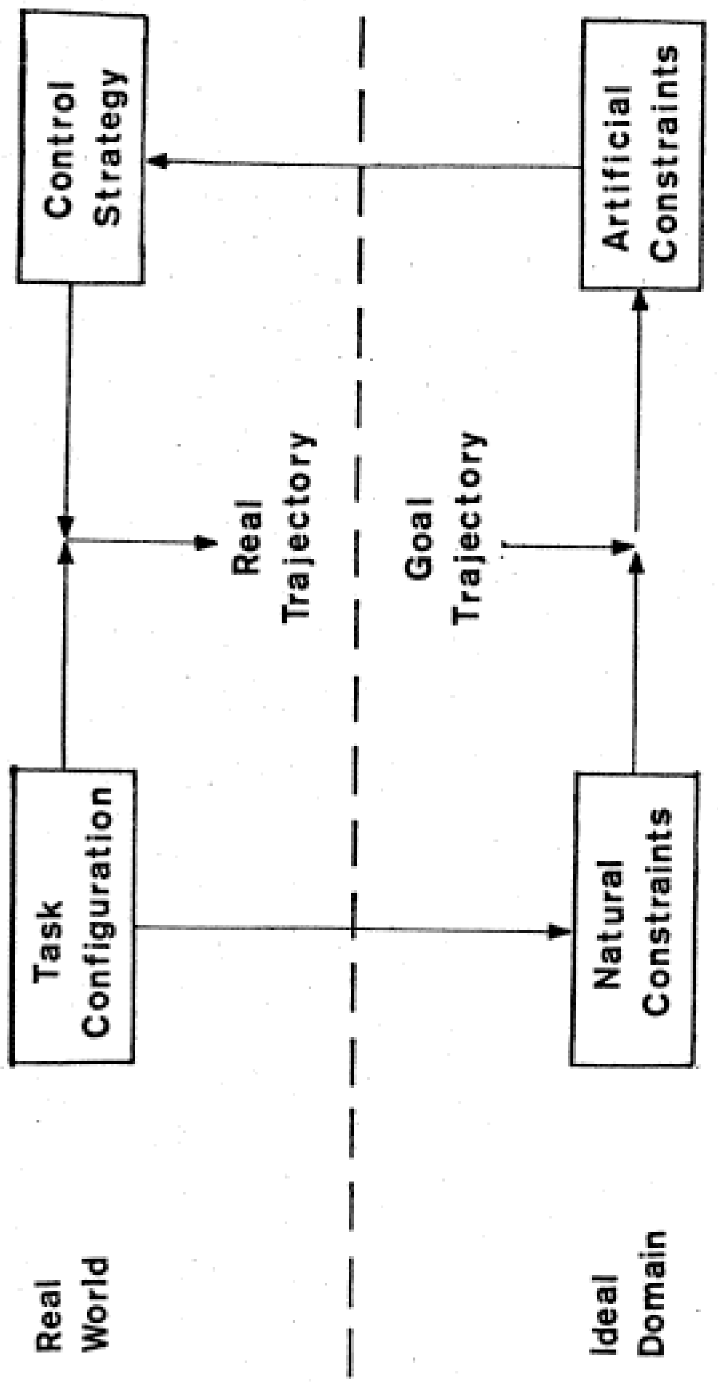

Figure I.2. Overview of Control Strategy Synthesis.

domain, so the synthesis method may be defined and evaluated in precise terms. In the ideal domain, the manipulator is represented by the ideal effector, an object with no physical dimensions, with attributes of position and force only. The task configuration is represented by equations relating the components of ideal effector force and velocity. These equations are referred to as the natural constraints. A control strategy for the ideal effector is represented in the same way, as linear equations on effector force and velocity, called the artificial constraints. The goal trajectory gives the desired ideal effector position as a function of time, which must be consistent with the natural constraints. In the ideal domain the control strategy synthesis problem is to find artificial constraints which will reproduce the desired goal trajectory given the natural constraints.

The synthesis of real-world control strategies consists of three steps. First, the task is modeled as a set of natural constraints in the ideal domain. Second, the synthesis method is applied in the ideal domain to obtain a set of artificial constraints. Finally, the artificial constraints are transformed into a corresponding real-world control strategy. The last step would be accomplished by a low-level manipulator control system which enforces the artificial constraint equations on the manipulator end-effector.

A model of the control system which was well-defined only when the ideal surface coincides with the physical surface would be of little use. In actual practice, the natural constraints would be obtained from a geometric model of the task environment, the planning model. The ideal goal trajectory will therefore lie in the planning model surface, while the actual trajectory must lie in the real-world surface. Hence, the low-level control system is modeled as a projection of the ideal goal trajectory onto the real-world surface. There are two appealing choices: project the trajectory along real-world surface normals or along planning model surface normals. We will opt for the second alternative:

> The desired manipulator trajectory lying in the real-world surface is obtained by projection of the goal trajectory along planning model surface normals.

There are many reasons for this choice. The real-world surface normals are unknown to the control system, while the planning model surface normals are. The use of planning model surface normals also simplifies planning/programming considerably. Given a particular goal trajectory and planning surface, if we project the goal trajectory along planning surface normals, the position of the manipulator depends only on the displacement between the planning surface and the actual surface. However, if the projection is along real-world surface normals, the position of the manipulator depends on the orientation of the real-world surface as well. The difference is illustrated in figure 1.3, which depicts projections from a planning model surface onto several actual surfaces

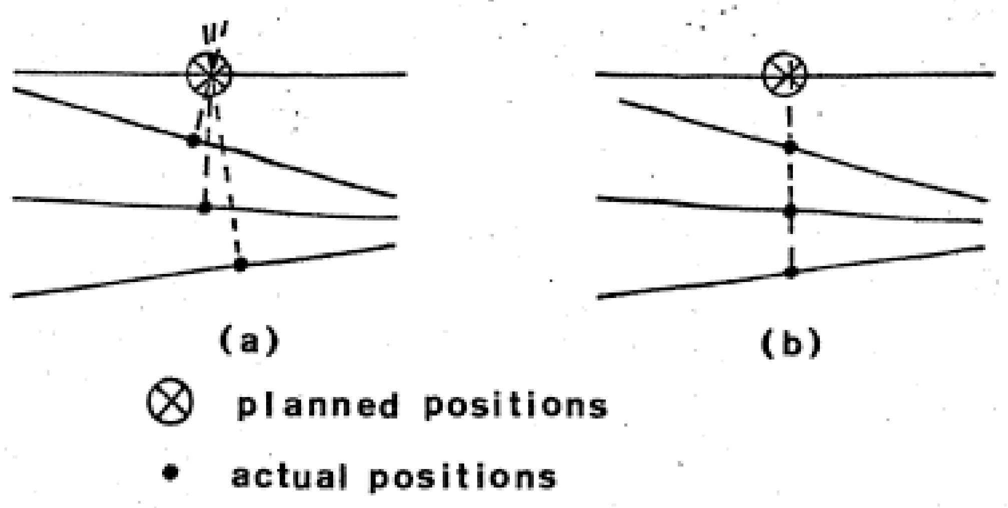

Figure 1.3. Position Projection Along Plan Normals and Real Normals. In (a) the planned position is projected onto real surfaces along the real surface normals. In (b), the projection is along planned surface normals.

using both methods.

Furthermore, if the projection is along planning normals, an implementation which projects velocities rather than positions will work fine, but if the control system projects velocities along real-world normals then the position of the manipulator at any time will depend on the entire trajectory history. This situation is illustrated in figure 1.4, comparing the goal trajectory with the trajectories obtained by projecting velocities along planning normals and along real normals. An error of this sort might go unnoticed in many cases, but some tasks are especially susceptible to error of this kind. For instance, if the manipulator were scribing a circle on a surface, slight irregularities in the surface would prevent the end points of the manipulator trajectory from meeting.

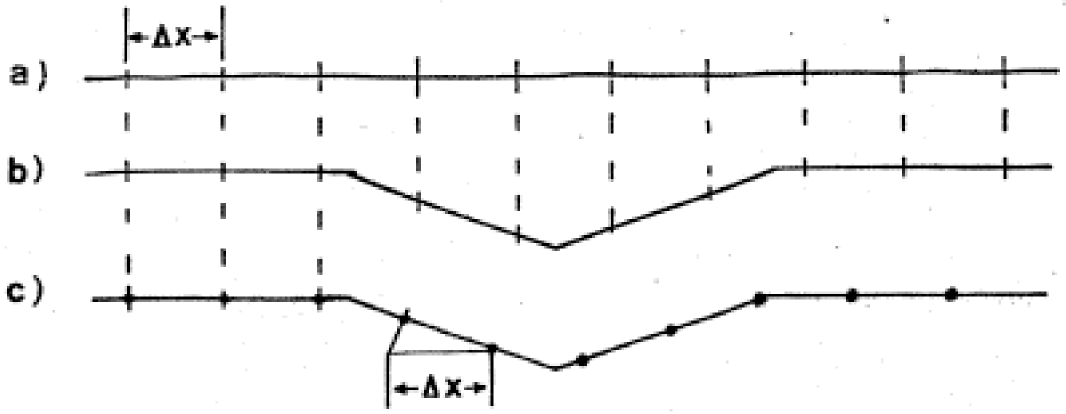

Figure 1.4. Velocity Projection Along Plan Normals and Real Normals. The goal trajectory specifies motion from left to right along the planning surface (a). The trajectory velocity is projected onto real surface (b) using planning surface normals, and onto real surface (c) using real surface normals. The tick marks correspond to the effector position at successive intervals of time.

To summarize, the task configuration and goal trajectory are transformed into a set of natural constraints in the ideal domain. From the natural constraints is obtained a set of artificial constraints for the ideal domain. The artificial constraints are applied to the manipulator end-effector by a low-level manipulator control system. The resulting manipulator trajectory is judged according to how well it satisfies the projection criterion. The problem of obtaining the artificial constraints from the natural constraints and goal trajectory is the subject of chapter two. The transformations between the real world and the ideal domain are developed in chapter three. Chapter four consists of the concluding discussion and suggestions for further work.

1.5 Relationship to Previous Work

This report is directly related to two different areas of manipulation research: manipulator force control, and automatic assembly planning. This section will briefly highlight previous work in these fields, and their relation to the work in this report.

The earliest computer-controlled manipulator is described in [Ernst61]. Although Ernst's system had force sensors mounted on the hand, they were not used to servo effector force, but to detect contact during a motion.

[Inoue71] demonstrated a system which used force control to turn a crank and insert a peg in a hole. For each desired axis of compliance, the user selects a manipulator joint — a free joint — to be force-servoed. Inoue's force control implementation was essentially open-loop; the actuators for the free joints exerted a constant torque for the duration of the movement.

[Paul72] implemented compliant motion using the free-joint method. Paul's system was also open-loop, but with compensation for forces due to link accelerations. Free joint selection was automated; the user specified the direction and magnitude of the force to be applied in cartesian space.

[Silver73] implemented a closed-loop free-joint system using a force-sensing wrist ([Minsky72]). A precision assembly task performed using this system is described in [Inoue74]. Free-joint selection was the responsibility of the user, but the manipulator joints were permanently aligned with cartesian axes, making the choice simple.

[Nevins74a] provides a good general overview of force control. Two methods, the *generalized spring* and *generalized damper* are of particular interest, as they readily provide formal models of force control and provide an interesting contrast with the force control models of this report. The generalized spring method has been realized mechanically [Nevins77] and the generalized damper's implementation in software is considered in detail in [Whitney76].

[Paul76] describes a cloosed-loop method of force control, and [Shimano77] describes an implementation of a variation of this method. The control strategies supported by these systems are very similar to the strategies we will discuss.

Task level planning has been investigated at Stanford [Finkel74, Taylor76], at Edinburgh [Ambler75], at MIT [Lozano76], and at IBM [Lieberman75]. These systems take task specifications expressed as operations on physical objects, and attempt to derive an appropriate manipulator program written using control system primitives. The last stage of this transformation is to derive the control program from manipulator motion specifications, i.e. control strategy synthesis. [Lozano76] and [Taylor76] discuss this problem in detail.

2. Control Strategy Synthesis in the Ideal Domain

2.1 Manipulator and Task Models

The ideal domain is a model of the real world for which control strategy synthesis is particularly simple. In the ideal domain, the "manipulator" is the ideal effector, defined to have attributes of position and force only. The ideal effector's force and position are each represented as six dimensional vectors over the reals. The six dimensions correspond to translation and rotation around the x-axis, y-axis, and z-axis.

The task configuration is represented by the subset of $\mathbb{R}^6$ consisting of the possible positions of the ideal effector, called the ideal surface. The ideal surface must be connected, and must have a unique tangent space at each point.[1] The dimension of the ideal surface may vary from 0 to 6. The tangent space is a vector flat (a vector subspace which has been displaced from the origin) with the same dimension as the ideal surface. A few examples of ideal surfaces will illustrate their properties. An ideal surface of dimension 0 is a point, corresponding to the absence of any freedom of motion. An ideal surface of dimension 6 contains all of $\mathbb{R}^6$ near the effector position, representing complete freedom of motion. Figures 2.1 and 2.2 show ideal surfaces for a peg in a hole and for a crank. The six axes of $\mathbb{R}^6$ are the three translation components P_x, P_y, P_z, and the three rotation components θ_x, θ_y, and θ_z. The ideal surfaces in figures 2.1 and 2.2 may be decomposed into spaces embedded in $\mathbb{R}^3$. Otherwise displaying these spaces graphically would be very difficult.

1. The spaces described are open connected differentiable sub-manifolds of $\mathbb{R}^6$, but the properties we need will be assumed without reference to the mathematical theory of manifolds.

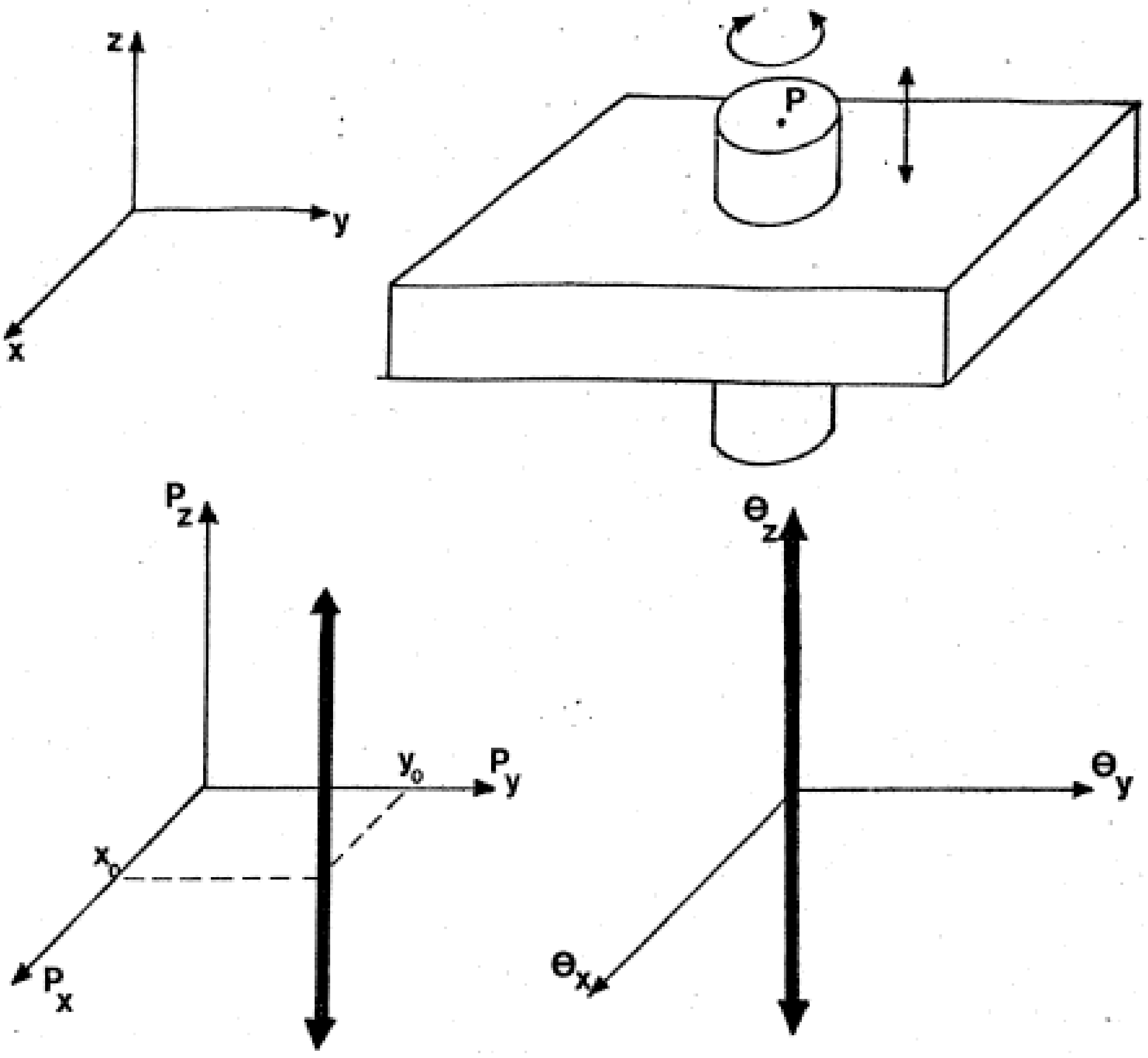

Figure 2.1. Ideal Surface at Center of Peg in Hole. The ideal surface is the plane equal to the cross-product of the two lines depicted here.

Since the ideal effector position lies in the ideal surface at all times, the ideal effector velocity will lie in the vector subspace parallel to the tangent space. This constraint can be expressed as a set of linear equations on the velocity coordinates. Similarly, if tangential forces are neglected, then effector force is restricted to be orthogonal to the tangent space, a constraint expressed as linear equations on ideal effector force coordinates. Figure 2.3 depicts the tangent

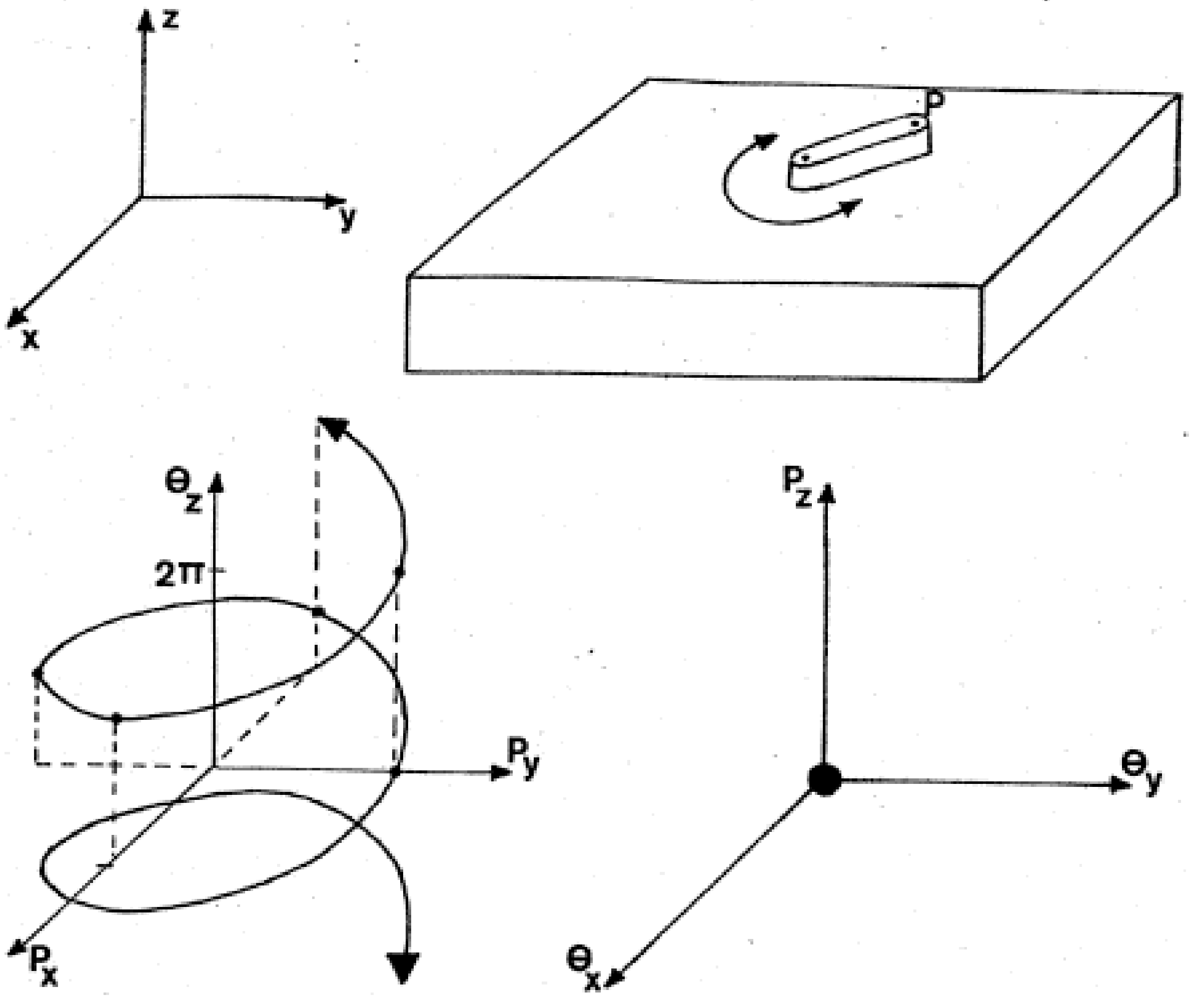

Figure 2.2. Ideal Surface for Crank. The ideal surface is the cross-product of the spiral on the left with the point on the right.

space (dashed line) and the parallel velocity space (solid line) for a crank. These linear equations in ideal effector velocity and force are the natural constraints.

The goal trajectory in the ideal domain is the desired ideal effector position as a function of time, which must lie in the ideal surface. Finally, control strategies in the ideal domain, the artificial constraints, are expressed in the same

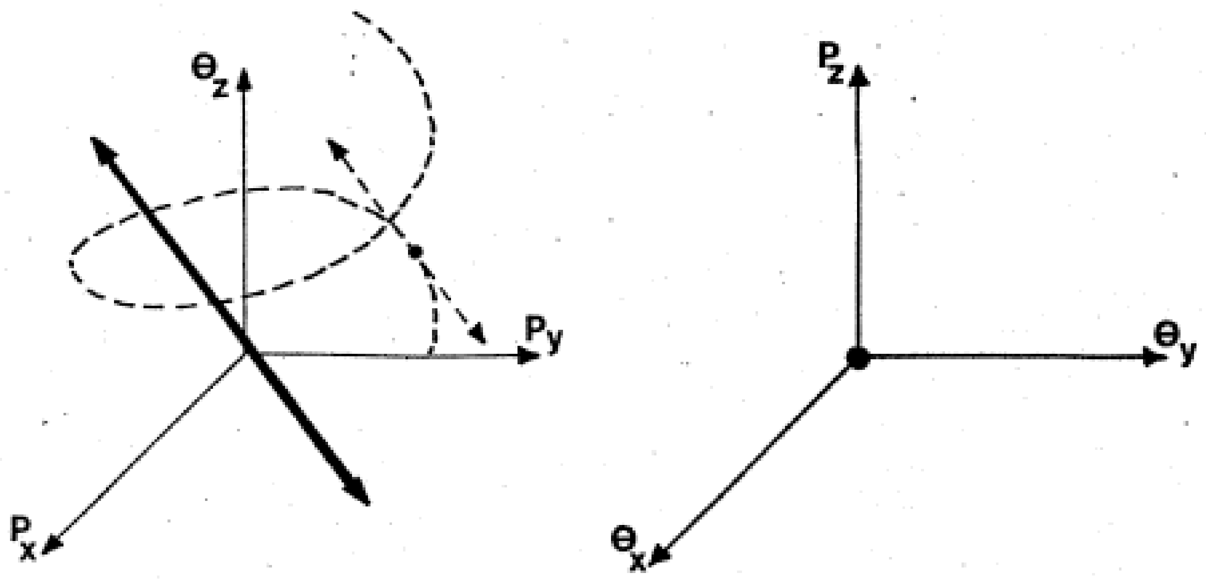

Figure 2.3. Tangent and Velocity Spaces for Crank Surface. The tangent space is the cross-product of the dashed line on the left with the point on the right. The velocity space is the cross-product of the solid line with the point.

form as the natural constraints: equations in components of effector force and velocity. The control strategy synthesis problem is to find artificial constraints which will produce the given goal trajectory for a given surface. This report develops a method for synthesizing the artificial constraints for a very broad range of surfaces, but we begin by considering very simple surfaces. The more general result is obtained by relaxing some of the restrictions on the surfaces.

2.2 Zero-Energy Surfaces

Consider again the ideal surface for a point at the center of a peg in a hole, with the hole fixed in space (figure 2.1). The peg is able to translate along the z-axis and rotate about the z-axis, but cannot translate along or rotate about the other axes. Similarly, force along and torque about the z-axis are zero (if the coefficients of friction are zero), while forces and torques for the other axes are unconstrained. So the natural constraints for this task configuration are

$$\begin{array}{ll} v_x = 0 & \omega_x = 0 \\ v_y = 0 & \omega_y = 0 \\ f_z = 0 & \tau_z = 0 \end{array}$$

where v_x, v_y represent components of ideal effector velocity; ω_x, ω_y components of angular velocity; f_z a component of effector force, and τ_z a component of effector torque.

These equations are homogeneous linear equations in the components of force and velocity, with the force equations orthogonal to the velocity equations. Formally, we define a *zero-energy* surface to be an ideal surface giving a consistent set of homogeneous linear equations in the components of effector velocity, and an orthogonal set of homogeneous linear equations of effector force. These are called zero-energy surfaces since it is impossible for the effector to exert a force in the same direction that it is moving.

Zero-energy surfaces correspond intuitively to real-world tasks comprised of stiff, immobile solids with negligible coefficients of friction. The definition also excludes any asymmetry of the constraints with respect to direction. That is, if motion is impossible in one direction, it must be impossible in the opposite direction. A simple example of a physical configuration violating this restriction is a peg resting against the bottom of a hole, illustrated in figure 2.4. In this case, translation along the positive z-axis is possible, but not along the negative z-axis. In fact, this configuration cannot be modeled directly as an ideal surface, since there is no unique tangent space. The transformation necessary to model

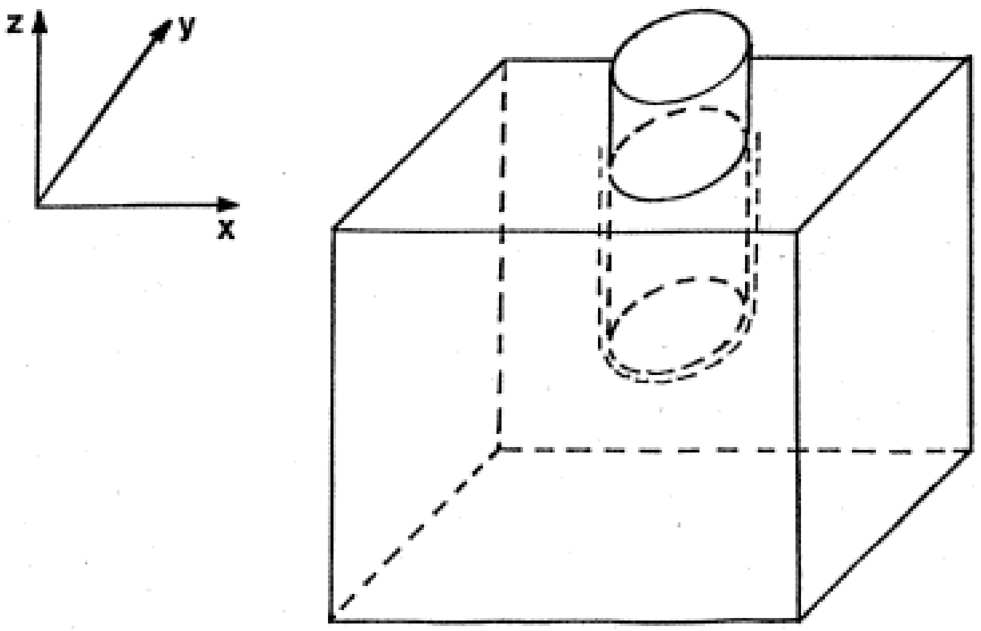

Figure 2.4. Peg at Bottom of Hole. The ideal effectors **A** and **B** are rigidly attached to O.

this situation is developed in section 2.5.

The effector velocity and effector force are represented as vectors in a six-dimensional vector space over the reals ($\mathbf{v}^T$ denotes the transpose of $\mathbf{v}$):

$$\mathbf{v} = (v_x\ v_y\ v_z\ \omega_x\ \omega_y\ \omega_z)^T$$
$$\mathbf{f} = (f_x\ f_y\ f_z\ \tau_x\ \tau_y\ \tau_z)^T$$

If the constraints on **v** and **f** are represented in matrix notation, then for the peg in hole example of figure 2.1, we have

$$\mathbf{A}\mathbf{v} = \mathbf{0} \qquad (2.1a)$$
$$\mathbf{B}\mathbf{f} = \mathbf{0} \qquad (2.1b)$$

where **A** and **B** are given by

$$\mathbf{A} = \begin{pmatrix} 1&0&0&0&0&0 \\ 0&1&0&0&0&0 \\ 0&0&0&0&0&0 \\ 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&0&0&0 \end{pmatrix} \qquad \mathbf{B} = \begin{pmatrix} 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \\ 0&0&1&0&0&0 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&0 \\ 0&0&0&0&0&1 \end{pmatrix}$$

Equations of the form of 2.1a and 2.1b restrict the effector force and velocity to lie in vector subspaces, denoted $\mathbf{S_v}$ and $\mathbf{S_f}$ respectively. For a zero-energy surface, $\mathbf{S_v}$ consists of all vectors which are orthogonal to every vector in $\mathbf{S_f}$, and vice versa. In other words $\mathbf{S_v}$ and $\mathbf{S_f}$ are <u>orthogonal</u> <u>complements</u>.

$$(\mathbf{S_v})^{\perp} = \mathbf{S_f} \qquad\qquad (\mathbf{S_f})^{\perp} = \mathbf{S_v}$$

For the configuration of figure 2.1, $\mathbf{S_v}$ is the v_z-ω_z plane, while $\mathbf{S_f}$ is spanned by the f_x, τ_x, f_y and τ_y axes. These two subspaces are indeed orthogonal complements.

The artificial constraints have the same form as the natural constraints, except that the equations need not be homogeneous. There are two criteria to determine whether the artificial constraints will produce the desired goal trajectory:

> The combined (artificial + natural) force constraints must be consistent.
>
> The combined velocity constraints must be non-singular with the goal trajectory as the unique solution.

There are many possible sets of artificial constraints which satisfy these conditions. We will focus on just those which are *orthogonal* to the natural constraints. Just as the natural velocity constraints could be represented as a vector subspace $\mathbf{S_v}$, the artificial velocity constraints may be represented as a vector *flat* $((\mathbf{S_v})^{\perp},\mathbf{v})$ which is the result of displacing the subspace $(\mathbf{S_v})^{\perp}$ from the origin by $\mathbf{v}$, $\mathbf{v} \in \mathbf{S_v}$. Now, an elementary result of linear algebra is that any velocity vector is uniquely represented as the sum of a vector in $\mathbf{S_v}$ and a vector in $(\mathbf{S_v})^{\perp}$, that is, $\mathbb{R}^6$ is the *direct sum* of $\mathbf{S_v}$ and $(\mathbf{S_v})^{\perp}$. The component of velocity along $(\mathbf{S_v})^{\perp}$ is determined by the task geometry to be $\mathbf{0}$. The component along $\mathbf{S_v}$ is equal to the goal trajectory velocity, but by construction, it is also the displacement $\mathbf{v}$ associated with the artificial constraint velocity flat $((\mathbf{S_v})^{\perp},\mathbf{v})$. Hence the artificial velocity constraints are completely determined by the natural constraints and the goal trajectory.

The situation is not as clear for the force constraints. The effector force is uniquely represented as the sum of a vector in $\mathbf{S_f}$ and a vector in $(\mathbf{S_f})^{\perp}$. Again, the force component along $(\mathbf{S_f})^{\perp}$ is determined by the task geometry to be $\mathbf{0}$, but the component along $\mathbf{S_f}$ is determined by neither the natural constraints nor the goal trajectory. The magnitudes of the forces normal to the surfaces of this simple model are irrelevant to the satisfaction of the goal trajectory, so any values will do.[1] We may choose the artificial force constraints to be an orthogonal vector flat $((\mathbf{S_f})^{\perp},\mathbf{f})$ with $\mathbf{f} \in \mathbf{S_f}$, but $\mathbf{f}$ is not completely determined. Any particular value of $\mathbf{f}$ will work for a zero-energy surface.

---

1. Since this is clearly not true in the real world, some explanation may be helpful. The zero-energy domain abstracts away some properties of manipulators such as actuator torque limits, and properties of task configurations such as limits on how much force can be applied without upsetting the task, so that force magnitudes become unimportant. Upsetting the task configuration is a problem which will be addressed in later sections of this chapter, but considerations which depend on the particular manipulator are better left out of the ideal domain, and will not be considered until the next chapter.

So the artificial constraints determine the component of velocity along S_v and the component of force along S_f. For the configuration of figure 2.1, these considerations lead us to the artificial constraints

$$\begin{array}{ll} f_x = 0 & \tau_x = 0 \\ f_y = 0 & \tau_y = 0 \\ v_z = k_1 & \omega_z = k_2 \end{array}$$

where k_1, k_2 depend on the goal trajectory, and may vary with time. In this particular case, force values of zero were used.[1]

2.3 Non-Homogeneous Surfaces

The definition of zero-energy surface constraints allows an asymmetry between the artificial and natural constraints — the artificial constraints are not required to be homogeneous. There is no reason not to allow similar generality in the natural velocity constraints. In fact, such a generalization will extend the scope of our analysis considerably. A non-homogeneous surface is defined to be a surface giving a set of natural constraints which may be non-homogeneous. The non-homogeneous surfaces include the zero-energy surfaces.

Recall that for zero-energy surfaces the natural constraints may be represented by the vector subspaces S_v and S_f containing the possible effector velocities and forces respectively. For non-homogeneous surfaces the same role is

1. The magnitudes might also be left indeterminate in the artificial force constraints, to be resolved when the artificial constraints are transformed into a real-world control strategy. Such a technique might be useful in a planning system or in manipulator-independent programming. The planning system might specify only those aspects of the effector's behavior which are relevant to the task, and allow other aspects to be resolved later with reference to the particular manipulator.

played by the vector flats $(\mathbf{S_v},\mathbf{v}_0)$ and $(\mathbf{S_f},\mathbf{f}_0)$ with $\mathbf{v}_0 \in \mathbf{S_v}^{\perp}$, $\mathbf{f}_0 \in \mathbf{S_f}^{\perp}$. As with zero-energy surfaces, $\mathbf{S_v}$ and $\mathbf{S_f}$ are orthogonal complements. The artificial constraints are unchanged in form, and we will still choose the velocity constraints to be the unique orthogonal set satisfying the goal trajectory. Likewise, the artificial force constraints will be chosen to be orthogonal to the natural force constraints, with the force magnitudes still not determined uniquely by the goal trajectory.

An example of the use of non-homogeneous natural velocity constraints is the problem of interaction between a manipulator and a moving belt on an assembly line. For instance, if the manipulator were to insert a peg in a moving hole (see figure 2.5) then the natural constraints would be

$$\begin{array}{ll} v_x = v_{belt} & \omega_x = 0 \\ v_y = 0 & \omega_y = 0 \\ f_z = 0 & \tau_z = 0 \end{array}$$

where the belt is moving in the x direction with velocity v_{belt}. The corresponding artificial constraints are

$$\begin{array}{ll} f_x = 0 & \tau_x = 0 \\ f_y = 0 & \tau_y = 0 \\ v_z = k_1 & \omega_z = k_2 \end{array}$$

which are the same as the artificial constraints obtained for a stationary peg-in-hole in section 2.2. The motion along the x-axis is taken care of by the force constraint along the x-axis. This constraint forces the peg to follow the hole whether stationary or moving.

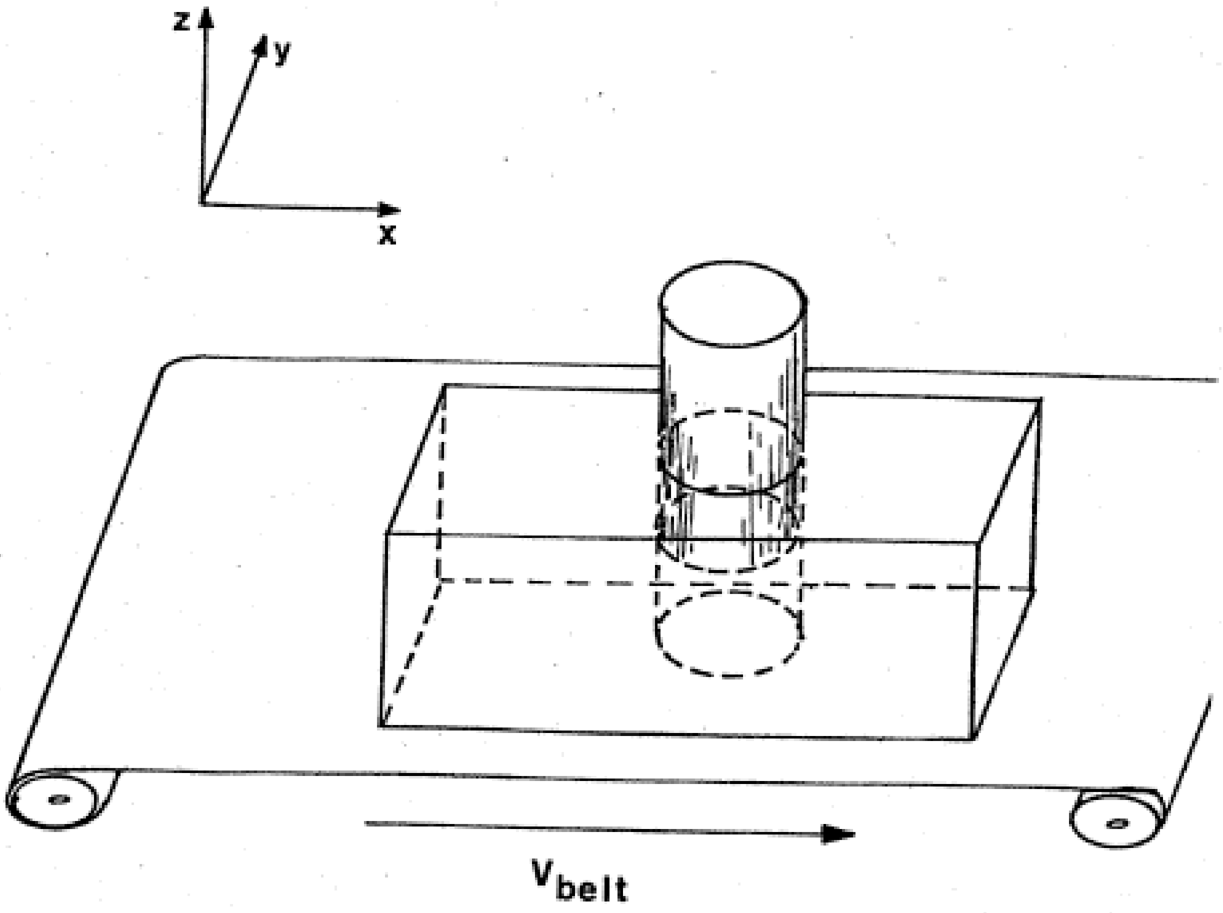

Figure 2.5. Compliance with Moving Belt.

The symmetry between natural and artificial constraints is complete for non-homogeneous surfaces, and there is nothing but viewpoint to distinguish between an arm control system and a task geometry. This unification allows an immediate extension to tasks involving multiple effectors.

2.4 Multiple Effectors

There is more than one way to develop controls for multiple effectors. The first method we will present is based on a generalization of the notion of orthogonal complement, and resembles previous use of force control for coordinated use of multiple manipulators ([Nakano74], [Ishida77]). A set of vector subspaces of R^6 are *orthogonal-complementary* if their direct sum is R^6 and if they are mutually orthogonal. Several manipulators may be coordinated by choosing the artificial constraint sets so that, taken with the natural constraint set, they form an orthogonal-complementary set.

Consider the two-dimensional example illustrated in figure 2.6. In this example, two effectors, A and B, are rigidly attached to the beam, O. Viewing the task first from A's viewpoint, if there are no natural velocity constraints on O, then the space spanned by the x, y, and θ axes is the subspace of possible velocities, and the artificial constraints at effector A may be three velocity constraints. Now if we view the configuration from effector B, the set of natural velocity constraints corresponds to the original natural velocity constraints plus the artificial velocity constraints of effector A. So, from B's viewpoint, the subspace of possible velocities is just the 0-vector, and the artificial constraints at B must be three force constraints. Clearly, the roles of A and B may be switched. It is also possible to use velocity constraints at A along the x-axis and at B along the other two axes, or any other suitable mixture. However, all of

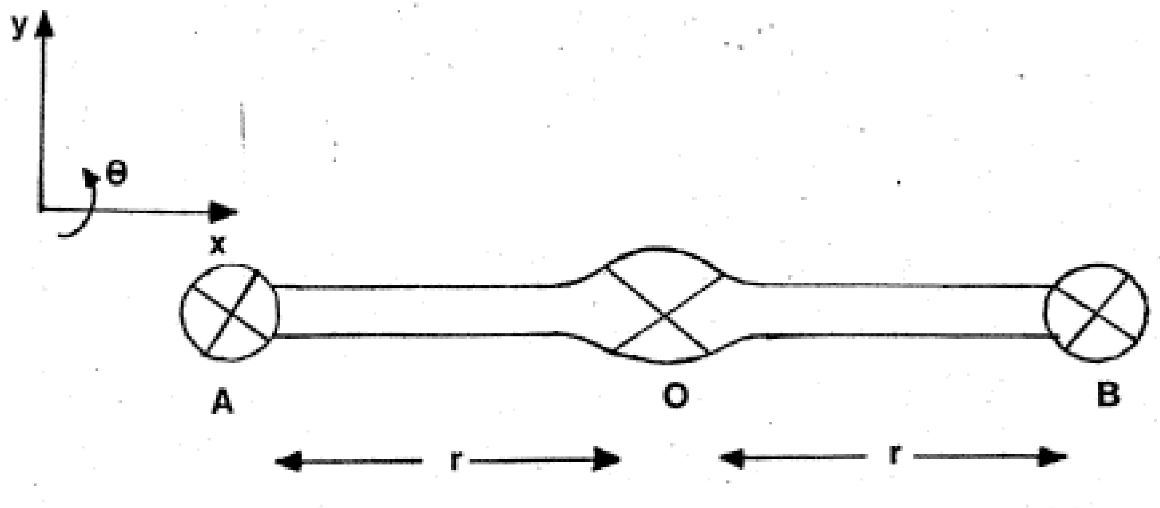

Figure 2.6. Multiple Effector Task.

these possibilities produce asymmetry in the behavior of the effectors. This is of no consequence in the ideal domain, but the effects are observable in the real world, where the manipulators would react differently to unexpected disturbances.

A more appealing approach exists which is symmetric with respect to the roles of the two effectors. First, we find the natural constraints. The rigid attachment between the object O and the manipulators A and B gives the following identities between velocities.

$$v_{Ax} = v_{Ox}$$
$$v_{Bx} = v_{Ox}$$
$$v_{A\theta} = v_{O\theta}$$
$$v_{B\theta} = v_{O\theta}$$
$$\frac{v_{Ay} + v_{By}}{2} = v_{Oy}$$
$$\frac{v_{By} - v_{Ay}}{r} = v_{O\theta}$$

If we assume that the only forces acting on effector A or B are the result of contact with O, then by balancing forces and moments we have

$$f_{Ax} + f_{Bx} = f_{Ox}$$
$$f_{Ay} + f_{By} = f_{Oy}$$
$$2\,\frac{f_{By} - f_{Ay}}{r} + f_{A\theta} + f_{B\theta} = f_{O\theta}$$

So far we have nine equations, obtained strictly from consideration of the interconnections of the manipulators and the object. An additional three equations will result from the interaction of O with the task geometry. By leaving these *external* constraints unspecified, the results we obtain will be applicable independent of the external task configuration. Some manipulation of the nine *internal* natural constraints gives us:

$$v_{Ax} - v_{Bx} = 0 \quad (2.2a)$$
$$v_{A\theta} - v_{B\theta} = 0 \quad (2.2b)$$
$$\frac{v_{By}}{r} - \frac{v_{Ay}}{r} - v_{A\theta} = 0 \quad (2.2c)$$

and

$$v_{Ax} = v_{Ox} \quad (2.3a)$$
$$v_{A\theta} = v_{O\theta} \quad (2.3b)$$
$$\frac{v_{Ay} + v_{By}}{2} = v_{Oy} \quad (2.3c)$$
$$f_{Ax} + f_{Bx} = f_{Ox} \quad (2.3d)$$
$$f_{Ay} + f_{By} = f_{Oy} \quad (2.3e)$$
$$2\,\frac{f_{By} - f_{Ay}}{r} + f_{A\theta} + f_{B\theta} = f_{O\theta} \quad (2.3f)$$

Equations 2.2 can be viewed as natural constraints on the velocity components of the effector pair. Equations 2.3 give us a transformation from velocity and force of the object to velocity and force of the effector pair. These can be used to transform external constraints on O to constraints on the manipulator pair. These transformed constraints, taken with the first three equations (equations 2.2), can be treated like any other set of natural constraints. Note that the first three, though, are independent of the external constraints on O. They result strictly from the geometry linking the effectors together. Hence, we can determine three of the artificial constraints independent of the external constraints on O. They are

$$f_{Ax} - f_{Bx} = 0 \quad (2.4a)$$
$$f_{A\theta} - f_{B\theta} = 0 \quad (2.4b)$$
$$\frac{f_{By}}{r} - \frac{f_{Ay}}{r} - f_{A\theta} = 0 \quad (2.4c)$$

Again, force values other than zero could be used, but zero gives an appealing symmetry. The remaining three artificial constraints depend on the external constraints on O, and are determined using the transformations above. For example, suppose O is pinned at its center so that only rotations are possible. Specifically, the natural constraints on O are

$$v_{Ox} = 0 \tag{2.5a}$$

$$v_{Oy} = 0 \tag{2.5b}$$

$$f_{O\theta} = 0 \tag{2.5c}$$

Applying the transformations (equations 2.3) we obtain natural constraints on A and B.

$$v_{Ax} = 0 \tag{2.6a}$$

$$\frac{v_{Ay} + v_{By}}{2} = 0 \tag{2.6b}$$

$$2\,\frac{f_{By} - f_{Ay}}{r} + f_{A\theta} + f_{B\theta} = 0 \tag{2.6c}$$

The corresponding artificial constraints are

$$f_{Ax} = 0 \tag{2.7a}$$

$$\frac{f_{Ay} + f_{By}}{2} = 0 \tag{2.7b}$$

$$2\,\frac{v_{By} - v_{Ay}}{r} + v_{A\theta} + v_{B\theta} = k \tag{2.7c}$$

where k is determined by the goal trajectory. The final set of artificial constraints is the conjunction of equations 2.4 and 2.7. A little cleaning up yields:

$$f_{Ax} = 0 \tag{2.8a}$$

$$f_{Bx} = 0 \tag{2.8b}$$

$$f_{Ay} + f_{By} = 0 \tag{2.8c}$$

$$f_{A\theta} - f_{B\theta} = 0 \tag{2.8d}$$

$$\frac{f_{By} - f_{Ay}}{r} - f_{A\theta} = 0 \tag{2.8e}$$

$$2\,\frac{v_{By} - v_{Ay}}{r} + v_{A\theta} + v_{B\theta} = k \tag{2.8f}$$

These constraints prevent compression or torsion of the bar linking the two effectors, they prevent any component of force at the pin in the x-y plane, and provide for compliant rotation of the bar about the pin.

This demonstrates how to transform a set of task constraints and a linking geometry into constraints for multiple effectors. We obtain two sets of natural constraints, one set due to the geometry connecting the effectors, and one set transformed from the external task constraints. The corresponding sets of artificial constraints play different roles in the resulting system. The artificial constraints obtained from the connecting geometry are used to determine the "mode" of cooperation of the effectors. In the example, they provide for symmetry, prohibiting any opposition between the manipulators. The artificial constraints obtained from the external constraints are used to obtain the desired external function.

2.5 Half-Surfaces

A simple example of a task not directly modeled by an ideal surface was shown in figure 2.4. In this example the peg is at the bottom of a hole, and may move upward along the z-axis, but not downward. This gives rise to constraints expressed not as equations but as inequalities:

$$v_z \geq 0 \qquad f_z \leq 0$$

There are two possible modes of motion possible in this case: withdraw the peg, or rotate the peg while maintaining contact with the bottom of the hole. The first mode corresponds to traversing the interface between two ideal surfaces of differing dimensions. A control strategy to accomplish such motions falls into the class of guarded moves (section 1.3), and is not treated in this report. It is the second mode which interests us. In order to apply the analysis of preceding sections, we use as a natural constraint

$$v_z = 0$$

and as the corresponding artificial constraint

$$f_z = \epsilon$$

In previous sections, the value of ϵ was irrelevant, but this is no longer so. In order to obtain a consistent set of force constraints, ϵ must be non-positive. In practice, a small negative value for ϵ, a small *bias force*, should be used to allow for error in the control system.

The result is that the ambiguity introduced at a half-surface may be resolved by examining the goal trajectory. If the trajectory moves away from the half-surface, the motion must be initiated using a guarded move. If the trajectory lies along the half-surface, we can pretend it is a "full" surface, as long as we guarantee continued contact by using a bias force.

For example, consider the task shown in figure 2.7, and suppose the goal trajectory specifies motion along the surface. The natural constraints are

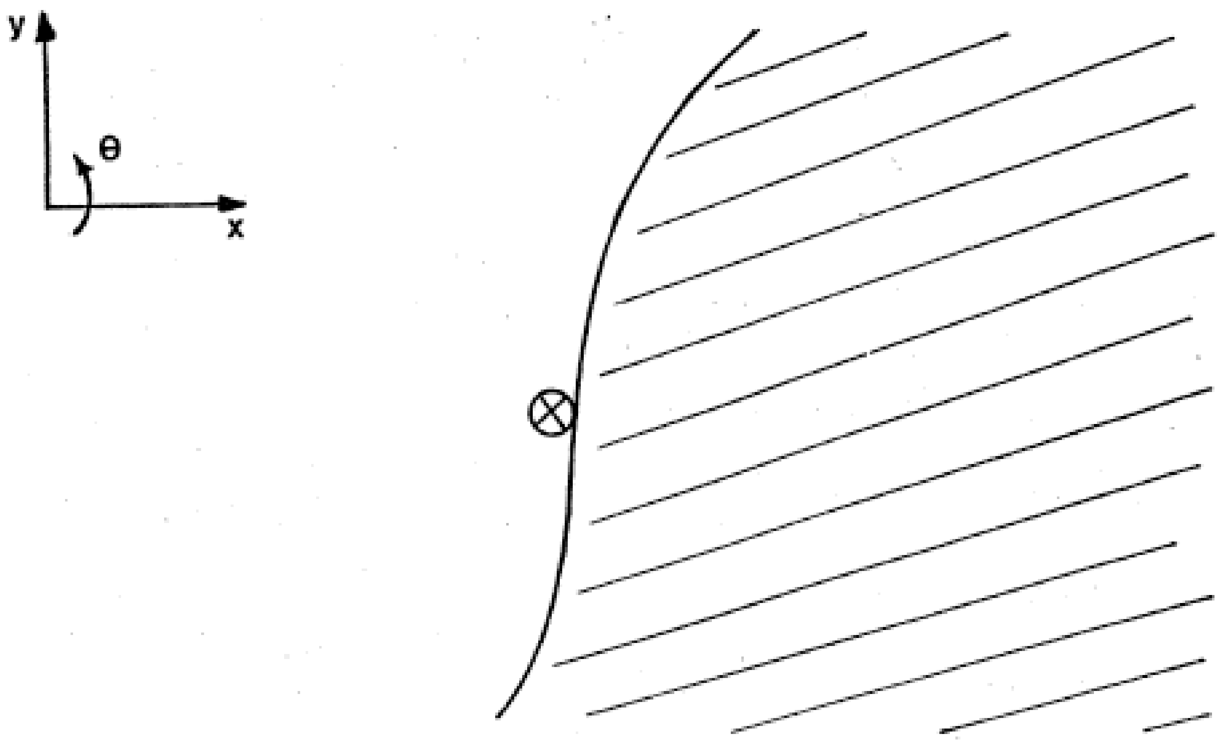

Figure 2.7. Two Dimensional Surface.

$v_x = 0$ (2.9a)

$f_y = 0$ (2.9b)

$f_\theta = 0$ (2.9c)

and the artificial constraints will be

$f_x = \epsilon$ (2.10a)

$v_y = k_1$ (2.10b)

$v_\theta = k_2$ (2.10c)

where ϵ is a small positive bias force and k_1, k_2 are determined by the goal trajectory. This trick of making assumptions to change the natural constraints from inequalities into equations, and then using the effector program to guard the assumptions, will also be used in the next section to handle sliding friction.

2.6 Force Thresholds

A half-surface gives rise to an ambiguity — we have the freedom to treat it as a surface or not, as we choose. Sliding friction produces a similar phenomenon by virtue of its threshold nature. A common model for sliding friction [Simunovic75] is given by

$$f_t = f_0 \frac{\mathbf{v}}{|\mathbf{v}|} \text{ if } |\mathbf{v}| > 0$$
$$|f_t| \leq f_0 \text{ otherwise}$$

where f_t is the force component along the surface tangent, $\mathbf{v}$ is the velocity vector, and f_0 is the *force threshold*. f_0 is sometimes the product of the normal force and the coefficient of friction.

This type of constraint can be treated in a fashion similar to that used for half-surfaces. In this case, the choice of model depends on whether or not motion relative to the surface is specified by the goal trajectory, and the assumptions can be guarded by choosing either a non-zero velocity or a force with magnitude less than f_0. In contrast to the situation with half-surfaces, both modes are readily described by ideal surfaces. Returning to the example of figure 2.7, suppose the coefficient of friction of the surface is μ. Then the new natural constraints are

$$v_x = 0 \qquad (2.11a)$$
$$f_y = \mu\, f_x \frac{v_y}{|v_y|} \qquad (2.11b)$$
$$f_\theta = 0 \qquad (2.11c)$$

if v_y is non-zero, or

$$v_x = 0 \tag{2.12a}$$
$$|f_y| \le \mu\, f_x \tag{2.12b}$$
$$f_\theta = 0 \tag{2.12c}$$

if $v_y = 0$. If the goal trajectory specifies motion upward along the surface, then the artificial constraints will be precisely the same as in the section 2.5, equations 2.10. On the other hand, if the goal trajectory specifies a velocity of zero along the y-axis, then the artificial constraints will be

$$f_x = \epsilon \tag{2.13a}$$
$$f_y = 0 \tag{2.13b}$$
$$v_\theta = k \tag{2.13c}$$

where ϵ is a bias force chosen as in section 2.6. As for a half-surface, the ambiguity can be resolved to satisfy the goal trajectory. The alternatives for the example are represented in table form below.

Motion Desired	Natural Constraints	Artificial Constraints
$v_y \ne 0$	$v_x = 0$ $f_y = \mu\, f_x \frac{v_y}{\lvert v_y\rvert}$ $f_\theta = 0$	$f_x = \epsilon$ $v_y = k_1$ $v_\theta = k_2$
$v_y = 0$	$v_x = 0$ $\lvert f_y\rvert \le \mu\, f_x$ $f_\theta = 0$	$f_x = \epsilon$ $f_y = 0$ $v_\theta = k_1$

The use of force threshold constraints may also be a convenient way to prevent excessive forces which might upset the task configuration.

2.7 Tangential Forces

The friction example of the previous section exhibited components of force along the surface tangent. Masses and springs also give rise to forces along surface tangents. The resulting natural constraints can be extremely complicated. Fortunately, some surfaces have the useful property that the normal forces are unconstrained, and in particular do not depend on the tangential forces. If this is the case the artificial constraints may be selected as before with assurance of consistency with the natural constraints, and the complicated nature of the tangential natural constraints may be ignored.

Suppose that we are given a set of natural velocity constraints, and suppose that the artificial velocity constraints are chosen to be perpendicular to the natural velocity constraints, and the artificial force constraints are parallel to the natural velocity constraints. Clearly the combined velocity constraints are non-singular. Likewise, if normal forces are not naturally constrained the combined force constraints will be consistent and independent.

For example, consider the task shown in figure 2.8. In this example, a two-dimensional bead is constrained to move along a wire. The natural constraints are

$$v_y = 0$$
$$v_\theta = 0$$
$$f_x = m \frac{d}{dt} v_x$$

where m is the bead's mass, a constant. This example satisfies the crucial assumption — there is no constraint on forces along the y-axis or θ-axis. Accordingly, the artificial constraints of the form

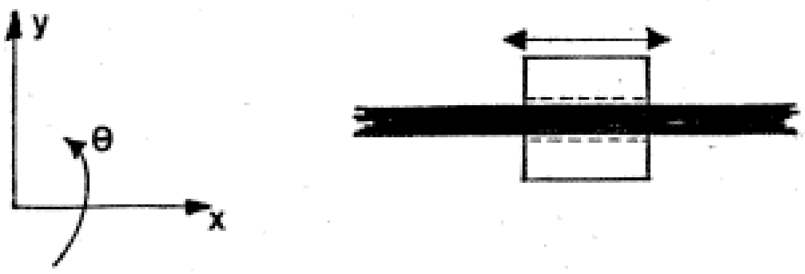

Figure 2.8. Bead on Wire.

$$f_y = 0$$
$$f_\theta = 0$$
$$v_x = k$$

(where, as usual, k may be a function of time) are non-singular.

In general, determining the natural constraints may not be possible. However, the upshot of this section is that the natural force constraints need not be determined, as long as we may assume that there is no natural constraint on the normal force. The artificial constraints may be chosen solely on the basis of the natural velocity constraints. The details of the relation between forces and the position trajectory along the surface tangent will not come up until we get to implementation of real-world control strategies (section 3.2).

The critical assumption of unconstrained normal force is valid for a broad range of task configurations. An example violating this assumption is the sliding friction example of the previous section, where normal force f_x was constrained by the equation

$$f_y = \mu \, f_x \frac{v_y}{|v_y|}$$

Another interesting example which violates the assumption is a spinning top, which transforms a force along one axis into a velocity along an orthogonal axis.

3. Transformations

3.1 From Real-World Tasks to Natural Constraints

Many aspects of the transformation from real-world tasks to natural constraints were suggested in the previous chapter. In particular, natural constraints for surfaces, friction, masses, and springs were developed. Rather than belabor these aspects of task configuration, we will proceed to consider objects with positive physical dimensions (section 3.1.1) and time-varying constraints (section 3.1.2).

3.1.1 Dealing with Objects of Positive Extent

The methods of the previous chapter require that the task be represented as constraints on the velocity of the ideal effector, which has no physical dimensions. The real world is composed of objects for which the physical constraints are distributed in space. Before applying the ideal effector methods, it will be necessary to collect these constraints at a single point.

The approach described here follows a common approach to the analysis and synthesis of mechanical linkages [Hartenburg64]. Let us assume we have the task represented as a set of rigid bodies, *links*, which are connected to one another by one of the *lower pairs*, exhibited in figure 3.1. The contacting surfaces of the two links of a pair are called the *elements* of the pair. A pair imposes a constraint on the relative velocity between the elements of the pair. In fact, the relative velocity is constrained to lie in a vector subspace. For example, the screw pair gives the following constraints:

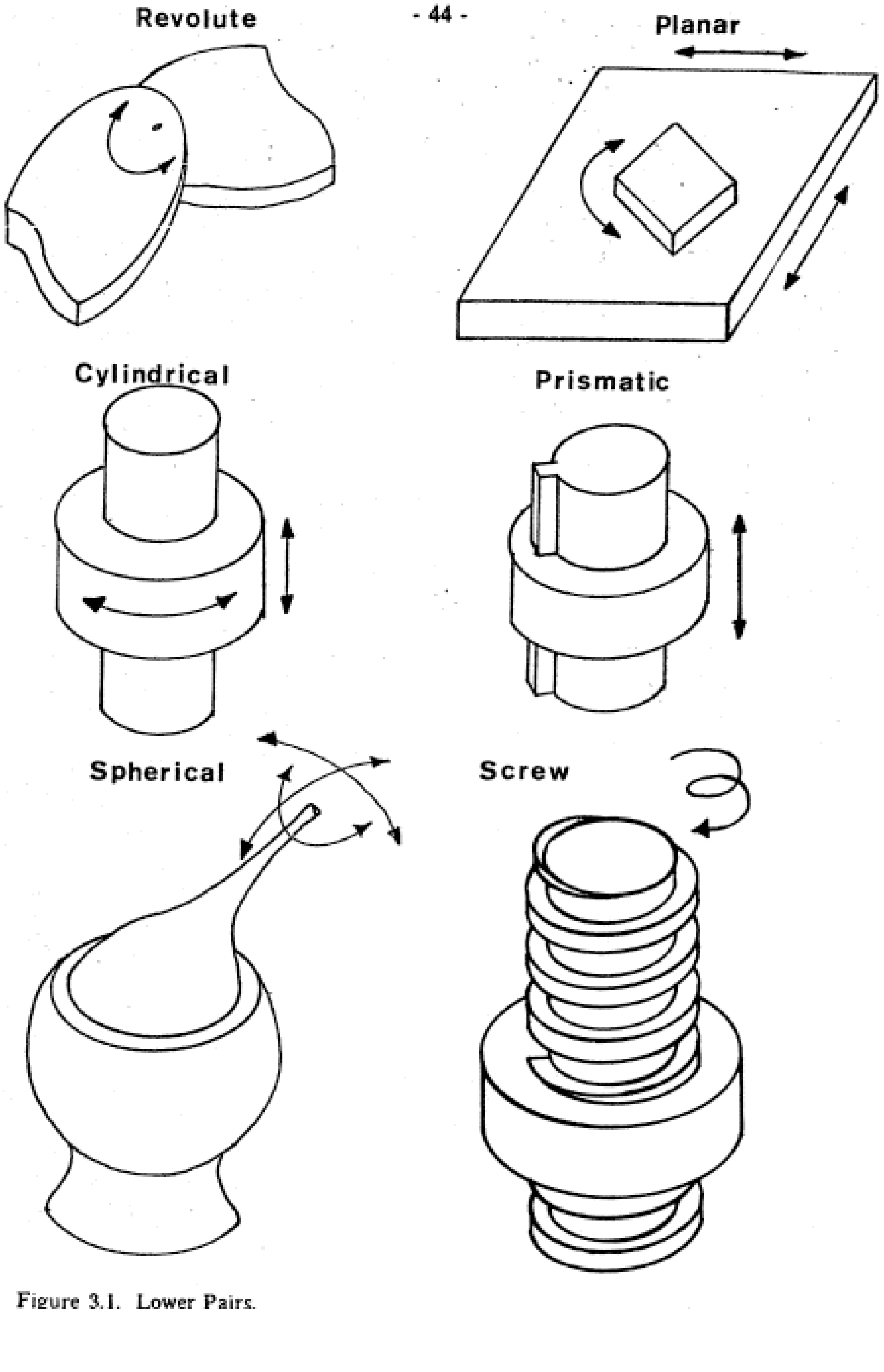

Figure 3.1. Lower Pairs.

$$v_x = v_y = \omega_x = \omega_y = 0$$
$$v_z - \frac{L}{2\pi}\omega_z = 0$$

where $\mathbf{v}_{pair} = (v_x\ v_y\ v_z\ \omega_x\ \omega_y\ \omega_z)^T$ is the relative velocity of the pair (the difference between the velocities of the pair elements), and L is the lead of the screw, the advance per revolution.

The natural constraints in the ideal domain for a task consisting of links connected by pairs may be determined by propagating velocity constraints to a common point at the effector. The propagation of constraints is handled by two operations: translation along links, and passing through pairs. For translation of velocity constraints along links, let us assume that A and B are two points rigidly attached to a link. If $\mathbf{S_A}$ and $\mathbf{S_B}$ are the velocity subspaces for points **A** and **B** respectively, then $\mathbf{S_B}$ is the image of $\mathbf{S_A}$ under the linear transformation **T**.

$$\mathbf{S_B} = \mathbf{T}(\mathbf{S_A})$$

where **T** is represented in matrix form by

$$\mathbf{T} = \left(\begin{array}{c|ccc} & 0 & R_z & -R_y \\ I_3 & -R_z & 0 & R_x \\ & R_y & -R_x & 0 \\ \hline 0 & & I_3 & \end{array}\right)$$

where $\mathbf{R} = (R_x\ R_y\ R_z)$ is a 3-vector from point A to point B.

Passing constraints through a pair is slightly more difficult. If the velocity constraints for one element of a pair are known, and if the relative velocity constraints for the pair are known, we can determine the resulting velocity constraints for the second element of the pair. If the relative velocity constraint subspace is $\mathbf{S_P}$, and the velocity constraint space of element **A** is $\mathbf{S_A}$, we have

$$(\mathbf{v}_B - \mathbf{v}_A) \in S_P$$
$$\mathbf{v}_A \in S_A$$

and the resulting constraint on $\mathbf{v}_B$ is

$$\mathbf{v}_B \in S_P \oplus S_A$$

where $S_P \oplus S_A$ consists of all vectors which may be expressed as the sum of a vector from S_P and a vector from S_A.

Using these operations the velocity constraints may be transformed to a common point. If two different constraint sets are obtained at a single point, the corresponding velocity subspace is obtained by intersecting the two original subspaces. Accordingly, the method proposed is to trace every chain of links and pairs from "ground" (presumably the workbench) to the effector. The intersection of the resulting subspaces gives the appropriate velocity constraint for the effector. As an example, consider the task configuration of figure 3.2. The constraint at A allows translation along the x-axis and rotation about the z-axis, while the constraint at B allows translation along the y-axis and rotation about the z-axis. We wish to find the velocity constraints on point P. If O is assumed motionless, then we have at A:

$$v_{Ay} = 0$$
$$v_{Az} = 0$$
$$\omega_{Ax} = 0$$
$$\omega_{Ay} = 0$$

and at B:

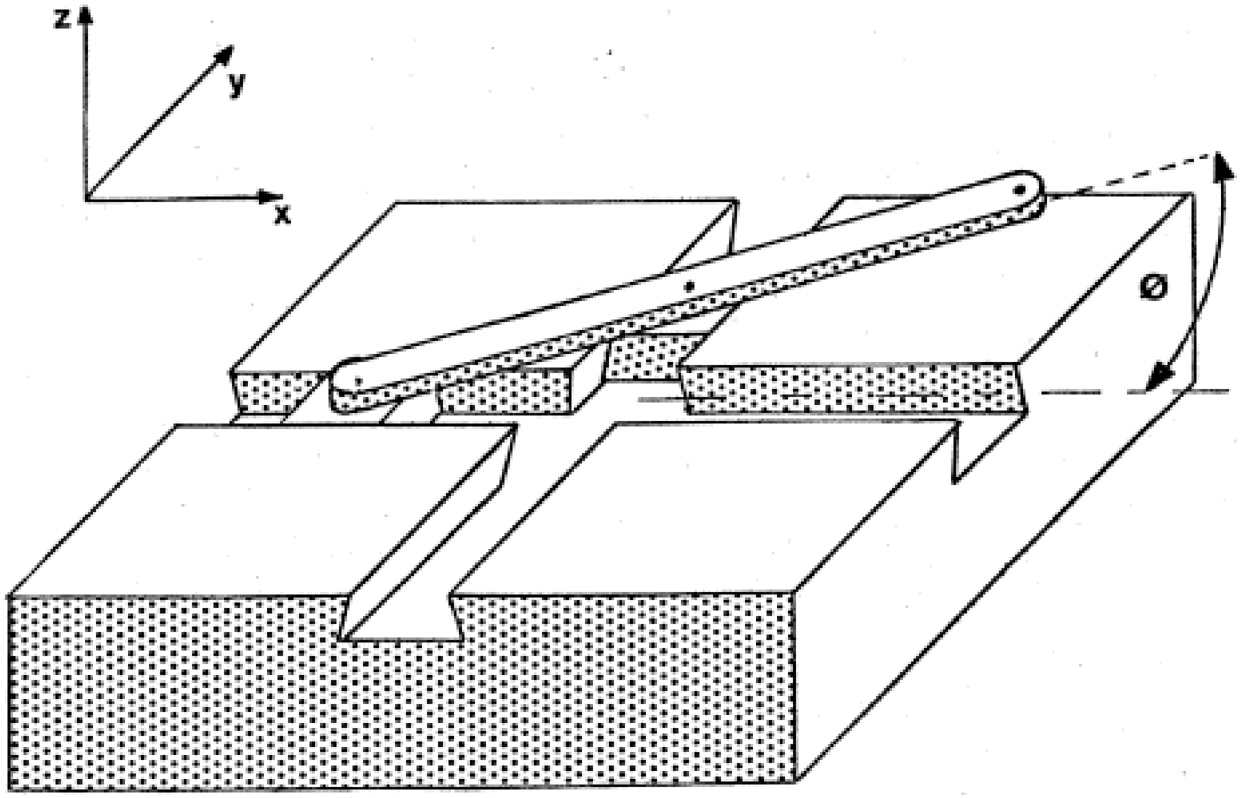

Figure 3.2. Example of Velocity Constraint Determination.

$$v_{Bx} = 0$$
$$v_{Bz} = 0$$
$$\omega_{Bx} = 0$$
$$\omega_{By} = 0$$

Letting R_A and R_B be the 3-vectors from A to P and from B to P,

$$R_A = (2\cos(\phi), 2\sin(\phi), 0)^T$$
$$R_B = (\cos(\phi), \sin(\phi), 0)^T$$

the appropriate linear transformations T_A and T_B are

$$\mathbf{T_A} = \left(\begin{array}{c|ccc} \mathbf{I} & \begin{matrix} 0 \\ 0 \\ 2\sin(\phi) \end{matrix} & \begin{matrix} 0 \\ 0 \\ -2\cos(\phi) \end{matrix} & \begin{matrix} -2\sin(\phi) \\ 2\cos(\phi) \\ 0 \end{matrix} \\ \hline \mathbf{0} & & \mathbf{I} & \end{array}\right)$$

$$\mathbf{T_B} = \left(\begin{array}{c|ccc} \mathbf{I} & \begin{matrix} 0 \\ 0 \\ \sin(\phi) \end{matrix} & \begin{matrix} 0 \\ 0 \\ -\cos(\phi) \end{matrix} & \begin{matrix} -\sin(\phi) \\ \cos(\phi) \\ 0 \end{matrix} \\ \hline \mathbf{0} & & \mathbf{I} & \end{array}\right)$$

Now $\mathbf{S_A}$ is spanned by the vectors $\mathbf{i}_1 = (1\ 0\ 0\ 0\ 0\ 0)^T$ and $\mathbf{i}_6 = (0\ 0\ 0\ 0\ 0\ 1)^T$ and $\mathbf{S_B}$ is spanned by $\mathbf{i}_2 = (0\ 1\ 0\ 0\ 0\ 0)^T$ and $\mathbf{i}_6$. So $\mathbf{S_P}$ is obtained by intersection of the space spanned by $\mathbf{T_A}(\mathbf{i}_1)$ and $\mathbf{T_A}(\mathbf{i}_6)$ with the space spanned by $\mathbf{T_B}(\mathbf{i}_2)$ and $\mathbf{T_B}(\mathbf{i}_6)$. Restated as vector equations we have

$$\mathbf{v_P} = k_1\ \mathbf{T_A}(\mathbf{i}_1) + k_2\ \mathbf{T_A}(\mathbf{i}_6)$$
$$\mathbf{v_P} = k_3\ \mathbf{T_B}(\mathbf{i}_2) + k_4\ \mathbf{T_B}(\mathbf{i}_6)$$

Expanding these into scalar equations and eliminating the unwanted variables, we finally obtain

$$v_{Px} + \sin(\phi)\ \omega_{Pz} = 0$$
$$v_{Py} - 2\cos(\phi)\ \omega_{Pz} = 0$$
$$v_{Pz} = 0$$
$$\omega_{Px} = 0$$
$$\omega_{Py} = 0$$

which are the natural velocity constraints for the point P.

Unfortunately, not all task configurations are as easily modeled as this one. The problem is that the relative velocity constraints introduced at a pair depend on the orientation of the pair. This in turn may depend on the relative positions of any number of pairs in the mechanism. Hence application of this method must in general be preceded by a positional analysis of the task. There is no general algebraic method of kinematic analysis available, although methods exist for various special classes of mechanisms (for example see [Duffy74]). If numerical methods are considered, the situation is somewhat improved [Uicker64].

3.1.2 Time-Varying Constraints

In order for a control strategy to be useful, its built-in assumptions of state should hold for some reasonable amount of time. For some tasks, the natural constraints are constant; for others they vary with time, either because the task surface is in motion, or because the manipulator is moving on the surface. Since the artificial constraints are orthogonal to the natural constraints, the artificial constraints will also be functions of manipulator position and time.

As an example, consider the door-closing problem in two dimensions. Figure 3.3 illustrates the problem. The effector is rigidly attached to the doorknob. If we assume any friction is negligible, the natural constraints at the instant shown are

$$v_x = 0$$
$$v_y - R\, v_\theta = 0$$
$$f_y + \frac{1}{R}\, f_\theta = 0$$

Unfortunately, the instant the door moves, the natural constraints will change. In this case, the natural constraints are easily expressed as a function of θ_D, the angle the door makes with the horizontal. Specifically,

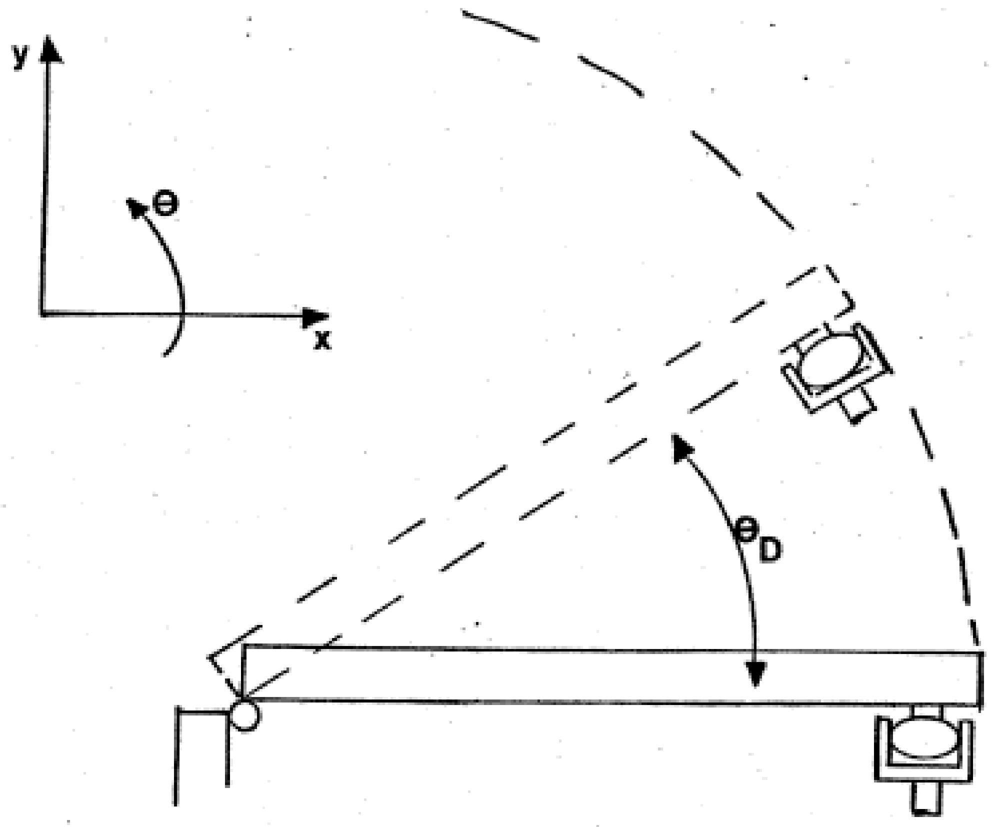

Figure 3.3. Door Closing Problem.

$$\cos(\theta_D)\, v_x + \sin(\theta_D)\, v_y = 0 \qquad (3.1a)$$
$$-\sin(\theta_D)\, v_x + \cos(\theta_D)\, v_y - R\, v_\theta = 0 \qquad (3.1b)$$
$$-\sin(\theta_D)\, f_x + \cos(\theta_D)\, f_y + \frac{1}{R}\, f_\theta = 0 \qquad (3.1c)$$

The artificial constraints may also be expressed in terms of θ_D, obtaining a single set of artificial constraints which is appropriate for any state of the door.

A simple method for specifying a similar control strategy is obtained by observing that the natural constraints as measured in the coordinate frame of the effector are constant. So the artificial constraints will also be constant in the effector frame [Paul76]. If the effector frame is initially aligned with the external coordinate frame then the artificial constraints are

$$f_x = 0 \tag{3.2a}$$

$$f_y - R\, f_\theta = 0 \tag{3.2b}$$

$$v_y + \frac{1}{R}\, v_\theta = 0 \tag{3.2c}$$

where the velocities and forces are measured in the effector frame.[1] If the artificial constraints are expressed in terms of θ_D, a method for determining θ_D is needed. There are three reasonable alternatives:

> Use the planned value for θ_D, computed from the goal trajectory.
>
> Compute θ_D from the actual manipulator's z-rotation in real time.
>
> Compute θ_D from the manipulator's position in the x-y plane in real time.

Since the manipulator will undoubtedly stray slightly from the planned trajectory, these three methods will give different results. Usually the first method is best, because it leads naturally to trajectories satisfying the goal as formulated in section 1.4. However, if the information on the door is incomplete, and large trajectory errors are possible, one of the latter two methods would be preferable.

---

1. To be precise, the velocities are measured in an external frame and transformed to the effector frame. If they were measured directly in the effector frame, the effector velocities would always be zero.

3.2 From Artificial Constraints into Control Strategies

The last step in the synthesis of control strategies is the transformation from artificial constraints to real-world control strategies. Since the ideal effector corresponds to the end-effector in the real world, the artificial constraints are interpreted as equations to be satisfied by the end-effector. The role of "equation enforcer" can be performed by a system which resembles control systems already in use [Paul76, Shimano77]. This section is primarily concerned with the design of such control systems, which leads into a discussion of error in the real world.

The problems of low-level control of manipulators are very complex. A detailed description of a low-level control system capable of enforcing artificial constraints is beyond the scope of this report. Nonetheless, success of the methodology developed in the preceding sections depends on the prospects of such a control system. While it is clear that the artificial constraints will produce the goal trajectory in the ideal domain, it is critical that they work in the real world. The nature of an implementation must be considered to properly assess this question.

There are three fundamental problems of classical mechanics which have been solved for many research laboratory manipulators. They are:

> *Kinematics.* Given the position of the end effector in cartesian coordinates, determine the corresponding manipulator joint positions.
>
> *Statics.* Given the force exerted by the effector of the manipulator, determine the corresponding forces developed at the manipulator joint actuators.

> *Dynamics.* Given the position, velocity, and acceleration of each manipulator link, determine the corresponding forces developed at the manipulator joint actuators.

A solution to the first problem enables a system to transform a goal trajectory defined in cartesian space to a trajectory in joint space. Analytic solutions for this problem are known for most research arms [Pieper68, Paul72, Horn74]. The solution to the second problem is the simplest, because the transformation from forces in cartesian space to forces in joint space is linear. If $\mathbf{f}_c$ is a vector of forces in cartesian space, and $\mathbf{f}_j$ is a vector of forces in joint space, then we have

$$\mathbf{f}_j = \mathbf{F}\ \mathbf{f}_c$$

where $\mathbf{F}$ is the *force resolution matrix.* It so happens that the force resolution matrix bears a simple relation to the solution to the kinematic problem. The Jacobian is a matrix whose i-jth entry is the partial derivative of the ith cartesian position coordinate with respect to the jth joint position variable.

$$\mathbf{dx} = \mathbf{J}\ \mathbf{d}\theta$$

where $\mathbf{dx}$ is the differential cartesian position vector, $\mathbf{d}\theta$ is the differential joint position vector, and $\mathbf{J}$ is the Jacobian. The force resolution matrix is just the transpose of the Jacobian matrix [Groome72].

$$\mathbf{F} = \mathbf{J}^T$$

The solution to the dynamic problem is more complicated [Kahn69, Paul72, Bejczy74, Horn75]. Although sophisticated dynamic models are available, most control systems are based on simplified models of the dynamics.

Given solutions to the three problems outlined above, an open-loop control system could be constructed as in figure 3.4. The control system translates the goal trajectory from cartesian coordinates to joint coordinates by inverting the kinematic equations. The static equations are used to derive actuator torques from the effector forces. To these are added the actuator torques obtained from the joint motions using the dynamic equations. The resulting torques would be applied at the appropriate actuators. This system works precisely the same whether force control or position control is desired. The two different control modes are only distinguished in the manner in which feedback is used.

The open-loop method is not generally used due to the presence of error. The planning model cannot represent the real task with complete accuracy. The static and dynamic equations are also inaccurate.[1] The use of sensory feedback

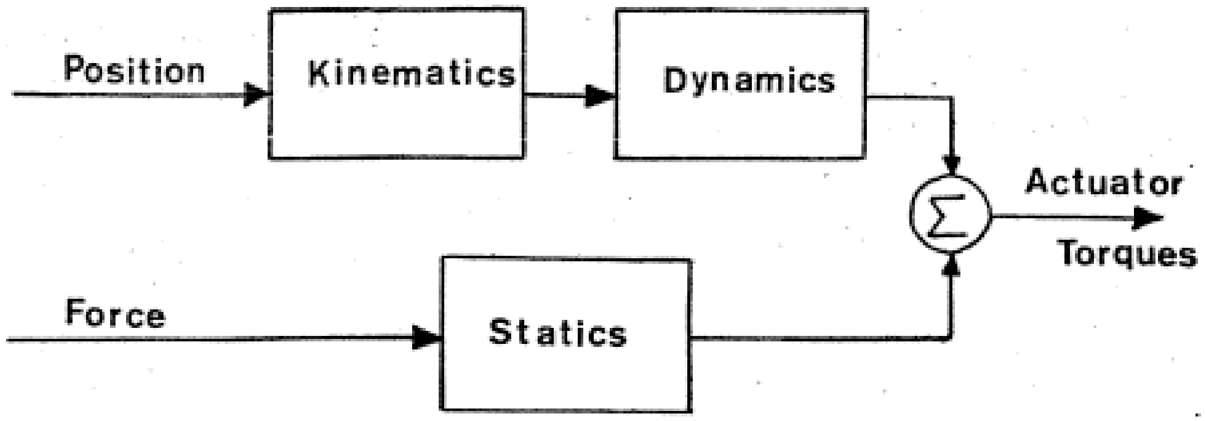

Figure 3.4. Open Loop Manipulator Control System.

1. Most control systems use very simple dynamic models. [Raibert77] and [Horn77b] describe a control method using a better dynamic model, but in any case error will still be a factor.

compensates for the inaccuracies of the task and manipulator models. However, when the loop is closed the force and position control systems are no longer identical. For one thing, a closed loop force control system would naturally require force sensors, while a position control system would require position sensors. For a hybrid system, combining force control of some axes with position control of other axes, both position and force error must be determined. However, the position error is reckoned only along the position axes, and the force error only along the force axes. In the language of the ideal domain, the force error is projected onto the planned surface normal space, and the position error is projected onto the planned surface tangent space. The resulting calculated error is the distance from the artificial constraint spaces, as it should be. Note that the planned surface is used for the error projections, consistent with the definition of the goal trajectory (section 1.4).

The calculations required for the implementation described above are very time-consuming. Some of these calculations may be avoided using the *free joint* method [Paul72, Paul76, and Shimano77]. This approach is easily grasped by considering a task with the property that each force or velocity constraint happens to be aligned with a manipulator joint.[1] In that case, the force axes can be servoed on force and the position axes on position in an independent fashion, which is precisely the method used in [Inoue71] and [Silver73]. Even if the joints are only approximately aligned with the desired constraints, good results may still be obtained. Based on this observation, the control system in [Paul72] accepts arbitrarily oriented force constraints and approximates them by constraints

1. The alignment of manipulator joints with natural constraints might occur surprisingly often, since most assemblies require compliant motions along one of only a few possible lines of approach [Nevins77]. Hence a manipulator might be used with its joints aligned with these lines of approach.

aligned with manipulator joints. The constraints and joints are matched by using the force resolution matrix to find the joint most sensitive in the direction of the constraint. The corresponding actuator torque is held constant, without regard for any normal forces contributed by the other manipulator joints. Similarly, the position axes are servoed without regard for tangential motions contributed by free joints. The resulting errors limit the usefulness of the system.

These problems were remedied in [Paul76]. Forces due to position-servoed joints are automatically accounted for by using force sensors to read the actual forces at the effector. Tangential motions due to free joints are calculated and used in the position servos. [Shimano77] describes an implementation of a similar system. Neither of these schemes will accept a force constraint which includes components of torque and force, such as the constraint

$$f_y - R\, f_\theta = 0$$

obtained in section 3.1.2 for the door-closing example. Also, neither system projects the goal trajectory using planning surface normals in all circumstances, although these differences could probably be eliminated without too much difficulty.

4. Conclusions

4.1 Discussion

The primary goal of this work was to develop a method for synthesizing control strategies for compliant motion, using a precise language to describe force control. To that end, the formal models for surface, manipulator, and control strategy were defined, and a precise formulation of goal trajectory obtained. The synthesis problem is considerably simpler for these abstract objects, and the control strategies obtained are appropriate for many compliant motion tasks in the real world. The most important aspects of this method are the representations of surface and manipulator. The use of sub-manifolds of $\mathbb{R}^6$ to represent surfaces focuses directly on the most important characteristic of a compliant motion task — the degrees of freedom of the manipulator. The unified treatment of rotational and translational variables led to the synthesis of control strategies which couple rotation and translation of the manipulator effector, an aspect of compliance which was previously neglected. The precise formulation of goal trajectory and the associated criteria for the manipulator trajectory (i.e. the projection of the goal trajectory along planning surface normals) has not been discussed in substance previously, although the desirability of some such projection property is mentioned in an example in [Paul76].

The use of artificial constraints orthogonal to the natural constraints is an important aspect of the synthesis method. For velocity constraints, this is the only choice that will produce a trajectory satisfying the goal trajectory criterion (section 1.4). The value of orthogonal force constraints is illustrated in figure 4.1. The planning model force constraint is represented by a solid line, the real-world force constraint by a dashed line. The orthogonal artificial constraint, pictured on the left, gives much better results than the artificial constraint

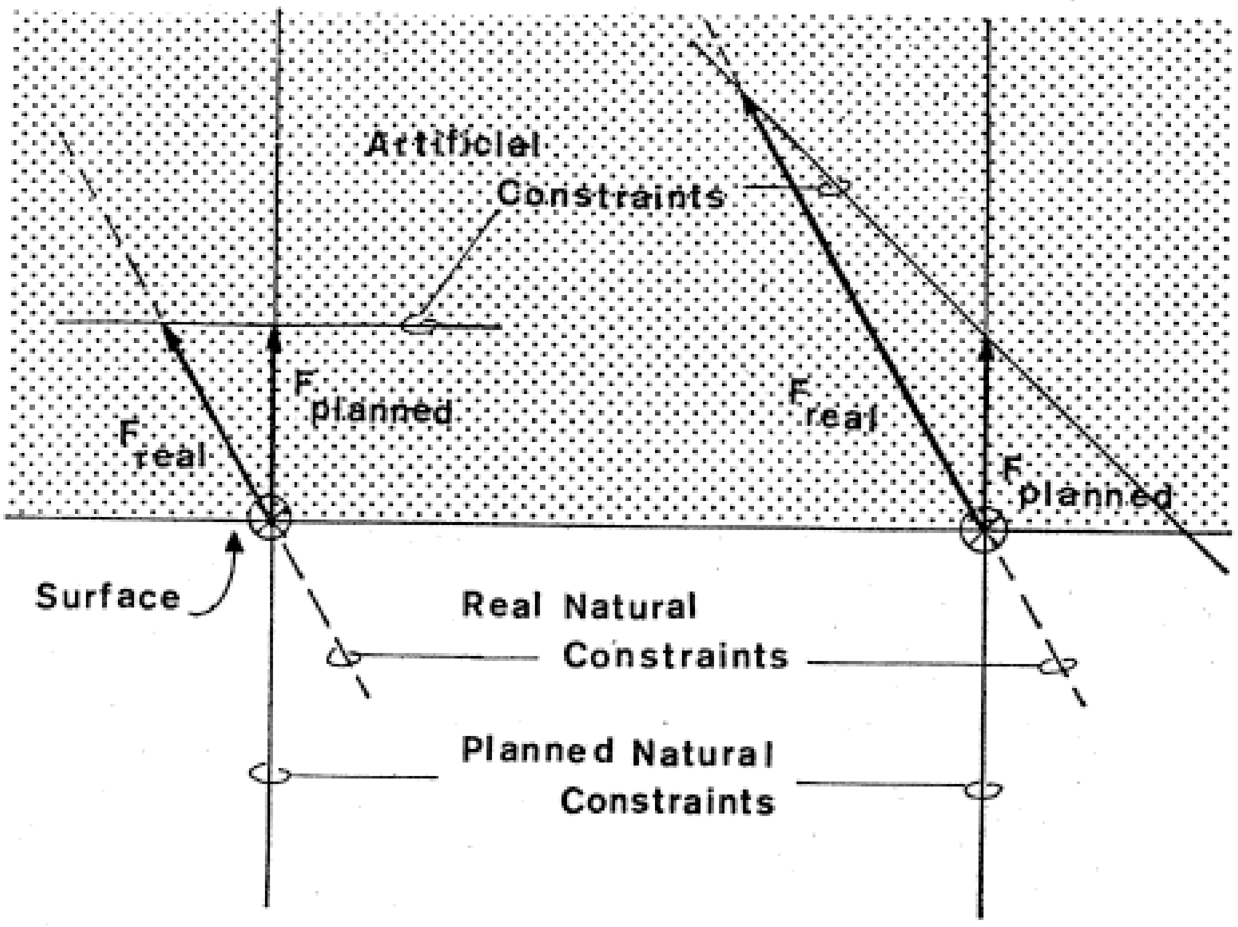

Figure 4.1. Comparison of Orthogonal and Non-Orthogonal Force Constraints. The effect of an error in the planned natural force constraints is compared for an orthogonal artificial constraint (left) and a non-orthogonal artificial constraint (right).

pictured on the right. If bounds for the normal force were determined in the planning stage, the non-orthogonal artificial force constraint might result in a force violating the bounds, thereby losing surface contact or upsetting the task.

An interesting question arises in connection with the representation of ideal surfaces — given an arbitrary ideal surface, is there a corresponding task configuration realizable in the real world. The answer is yes, because the transformation from artificial constraints to real-world control strategies is a method for constructing a physical realization for an arbitrary ideal surface. In this sense, one can think of the manipulator as a programmable ideal surface. In the vicinity of the effector position the planning surface and "manipulator surface" will have exactly one common point. The two surfaces are orthogonal at this point, and the position of this intersection (as a function of time) is the goal trajectory. The real trajectory will be the intersection of the real task surface with the manipulator surface. The projection property is a simple consequence of the use of a manipulator surface which is orthogonal to the planning surface. In this view of compliant motion, force control is just a manipulator surface, and the control strategy synthesis method says simply to use a manipulator surface which is orthogonal-complementary to the planning surface.

The ideal effector has potential applications in other areas of manipulation. It is used implicitly in previous task planning systems (particularly [Udupa77], [Lozano78]), and seems to be a good way to develop manipulator-independent programs. Much of the planning process can be decomposed into a stage concerned primarily with task level planning in terms of the ideal effector, and a stage to interpret an ideal effector program using some particular real-world manipulator. The ideal effector can also model objects in the environment other than the manipulator end-effector. In fact, in this report the ideal effector was often identified with some geometrical solid which was not a manipulator. This generality is important for planning purposes, for the desired compliant motion often does not involve the end effector directly, but rather an object grasped by the effector.

It must be emphasized that there are other types of control strategies which might be used for compliant motion. Two in particular are the *generalized spring* and the *generalized damper* methods. In the generalized spring method, the control strategy enforces the relation

$$\mathbf{f} = \mathbf{K}\ (\mathbf{p} - \mathbf{p}_0)$$

where $\mathbf{f}$ is the effector force, $\mathbf{p}$ is the effector position, and $\mathbf{p}_0$ is the nominal position, which is input supplied from the planning system or user program. $\mathbf{K}$ is the *stiffness* matrix, which relates forces observed at the effector to deviations from the nominal position. The stiffness matrix can be chosen to optimize performance of a particular task. This approach was pursued at the Draper Lab and culminated in a mechanical implementation capable of extremely fast peg insertions.

The generalized damper method is similar in form. The control strategy enforces the relation

$$\mathbf{f} = \mathbf{B}\ (\mathbf{v} - \mathbf{v}_0)$$

where $\mathbf{f}$ is the effector force, $\mathbf{v}$ is the effector velocity, and $\mathbf{v}_0$ is the nominal velocity, which is input from the planning system or user program. $\mathbf{B}$ is the *damping* matrix, in this case relating effector force to deviations from the nominal velocity. A generally useful choice for $\mathbf{B}$ is just the identity matrix times some negative damping coefficient. Other choices may exhibit interesting properties. For instance, it is possible to choose $\mathbf{B}$ so that the manipulator will climb over any obstacles it encounters. Many early force compliance schemes [Groome72, Silver73] are conveniently viewed as damping schemes. Both the generalized spring and damper methods are discussed in [Nevins74a] and the generalized damper is discussed further in [Whitney76]. We could also define a generalized inertia control strategy, which would be a convenient model for the free-joint method of force control.

These control strategies (when a reasonable stiffness or damping matrix is used) have the advantage of being consistent with almost any naturally occurring set of natural constraints. However, their effect is more difficult to model than the ideal surface strategies presented in this report. For this reason, they are not as convenient from the planning point of view. From the implementation point of view, though, these models are indispensable. For instance, an implementation of an ideal surface strategy might look like a generalized spring with the position axes very stiff and the force axes very loose.

4.2 Suggestions for Further Work

This report touches on many aspects of manipulation, so many of the ideas and problems have been only partially developed. The control strategies developed were not compared in any precise sense with other strategies for compliance, the generalized spring and damper methods in particular. Likewise the design choice restricting artificial constraints to be orthogonal to the natural constraints was only addressed superficially in the report. This is particularly interesting, since many of the published examples of compliance programs correspond to a set of non-orthogonal constraints.

Integrating the control strategy synthesis technique into a complete planning system is an interesting problem. A trajectory traversing several surfaces might be decomposed into several segments, each lying in a single surface. The transition from one surface to another requires a guarded-move type of control strategy. The synthesis of guarded-move type strategies for the interface between two surfaces is an interesting problem in its own right.

The report emphasizes the use of the ideal domain for synthesizing artificial constraints, but it was felt that the chapter on transformations between the ideal domain and the real world was necessary for logical completeness. The particular transformation methods described were meant to play the role of a constructive existence proof. A more serious effort on this problem would be justified. Velocity analysis of mechanisms has not been neglected in the kinematics literature (for example see [Denavit65]), but the current work is focused on single degree-of-freedom closed kinematic chains, with limited applicability to manipulation tasks.

The difficulty of implementing an active force control scheme should not be judged by the scant attention it receives in this report. There are important unresolved issues both in hardware and software design. Ultimately any compliant motion method will be judged by the velocity achieved without exceeding a specified force error, which depends on the frequency characteristics of the force control loop. In [Nevins74b] this argument is pursued to suggest the use of redundant degrees of freedom to provide compliance. The remote-center compliance device [Nevins77] uses passively compliant redundant degrees of freedom. A programmable active compliance is necessary to implement the compliant motion primitives developed in this report, but so far no active compliance using redundant degrees of freedom has been built.

References

[Ambler75]

Ambler, A. P., Barrow, H. G., Brown, C. M., Burstall, R. M., and Popplestone, R. J., "A Versatile System for Computer-Controlled Assembly", Artificial Intelligence, vol. 6, 1975.

[Barbera76]

Barbera, J. A., Albus, J. S., and Evans, J. M., "Control Strategies for Industrial Robots Systems", Society of Manufacturing Engineers Technical Paper MR76-616, 1976.

[Bejczy74]

Bejczy, A. K., Robot Arm Dynamics and Control, Jet Propulsion Laboratory Technical Memo 33-669, February, 1974.

[Bolles73]

Bolles, R. C., and Paul, R., "The Use of Sensory Feedback in a Programmable Assembly System", Stanford Artificial Intelligence Laboratory Memo AIM-220, October, 1973.

[Darringer75]

Darringer, J. A., and Blasgen, M. W., "MAPLE: A High Level Language for Research in Mechanical Assembly", IBM Research Report RC-5606, September, 1975.

[Denavit65]

Denavit, J., Hartenberg, R. S., Razi, R., and Uicker, J. J., "Velocity, Acceleration, and Static-Force Analyses of Spatial Linkages", Journal of Applied Mechanics, December, 1965.

[Duffy74]

Duffy, J., and Rooney J., "Displacement Analysis of Spatial Six-Link, 5R-C Mechanisms", Journal of Applied Mechanics, September, 1974.

[Dunne77]

Dunne, M. J., "An Advanced Assembly Robot", Society of Manufacturing Engineers Technical Paper MS77-755, 1977.

[Ernst61]

Ernst, H. A., MH-1, A Computer-Operated Mechanical Hand, Sc. D. Thesis, MIT, December, 1961.

[Finkel74]

Finkel, R., Taylor, R., Bolles, R., and Feldman, J., "AL, A Programming System for Automation", Stanford Artificial Intelligence Laboratory Memo Number AIM-243, November, 1974.

[Finkel76]

Finkel, R., "Constructing and Debugging Manipulator Programs", Stanford Artificial Intelligence Laboratory Memo 284, August, 1976.

[Freund76]

Freund, E., and Syrbe, M., "Control of Industrial Robots by Means of Microprocessors", Colloques IRIA, Analyse de Systemes et ses Orientations Nouvelles, 1976.

[Goto74]

Goto, T., Inoyama, T., and Takeyasu, K., "Precise Insert Operation by Tactile Controlled Robot", 2nd Conference on Industrial Robot Technology, March, 1974.

[Groome72]

Groome, R. C., Force Feedback Steering of a Teleoperator System, M. S. Thesis, MIT Department of Aeronautics and Astronautics, August 14, 1972.

[Hartenburg64]

Hartenburg, R. S., and Denavit, J., Kinematic Synthesis of Linkages, McGraw-Hill, 1964.

[Hollerbach77]

Hollerbach, J. M., "The Minimum Energy Movement for a Spring Muscle Model", MIT Artificial Intelligence Laboratory Memo No. 424, September, 1977.

[Horn74]

Horn, B. K. P., and Inoue, H., "Kinematics of the MIT-AI-VICARM Manipulator", MIT Artificial Intelligence Laboratory Working Paper 69, May 1974.

[Horn75]

Horn, B. K. P., "Kinematics, Statics, and Dynamics of Two-D Manipulators", MIT Artificial Intelligence Laboratory Working Paper 99, June 1975.

[Horn77a]

Horn, B. K. P., Hirokawa, K., and Vazirani, V. V., "Dynamics of a Three Degree of Freedom Kinematic Chain", MIT Artificial Intelligence Laboratory Working Paper No. 155, October, 1977.

[Horn77b]

Horn, B. K. P., and Raibert, M. H., Configuration Space Control, MIT Artificial Intelligence Laboratory Memo 458, December, 1977.

[Inoue71]

Inoue, H., "Computer Controlled Bilateral Manipulator", Bulletin of the JSME, vol. 14, no. 69, 1971.

[Inoue74]

Inoue, H., "Force Feedback in Precise Assembly Tasks", MIT Artificial Intelligence Laboratory Memo no. 308, August 1974.

[Ishida77]

Ishida, T., "Force Control in Coordination of Two Arms", Proceedings, Fifth International Conference on Artificial Intelligence, August, 1977.

[Kahn69]

Kahn, M. E., "The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains", Stanford Artificial Intelligence Memo Number 106, December, 1969.

[Lewis74]

Lewis, R. A., "Autonomous Manipulation on a Robot: Summary of Manipulator Software Functions", Jet Propulsion Laboratory

Technical Memorandum 33-679, March 1974.

[Lieberman75]

Lieberman, L. I., Wesley, M. A., "AUTOPASS: A Very High Level Programming Language for Mechanical Assembler Systems", IBM Research Report RC 5599, August, 1975.

[Lozano76]

Lozano-Perez, T., The Design of a Mechanical Assembly System, MIT Artificial Intelligence Laboratory Technical Report 397, 1976.

[Lozano78]

Lozano-Perez, T., and Wesley, M. A., "An Algorithm for Planning Collision-Free Paths Amongst Polyhedral Obstacles", IBM Research Report RC 7171, June, 1978.

[Minsky72]

Minsky, M., Manipulator Design Vignettes, MIT Artificial Intelligence Laboratory Memo 267, October, 1972.

[Nakano74]

Nakano, E., Ozaki, S., Ishida, T., and Kato, I., "Cooperational control of the Anthropomorphous Manipulator 'Melarm'", Proceedings, 4th International Symposium on Industrial Robots, November 19-21, 1974.

[Nevins74a]

Nevins, J. L., and Whitney, D. E., "The Force Vector Assembler Concept", First IFToMM Symposium on Theory and Practice of Robots and Manipulators, 1974.

[Nevins74b]

Nevins, J. L., Whitney, D. E., et al, Exploratory Research in Industrial Modular Assembly, Report R-800, Draper Lab, Cambridge, MA, March, 1974.

[Nevins74c]

Nevins, J. L., Whitney, D. E., and Woodin, A. E., "A Scientific Approach to the Design of Computer Controlled Manipulators", Report R-837, Draper Lab, Cambridge, MA, August, 1974.

[Nevins75a]

Nevins, J. L., and Whitney, D. E., "Adaptable-Programmable Assembly Systems: an Information and Control Problem", Proceedings, 5th International Symposium on Industrial Robots, September, 1975.

[Nevins75b]

Nevins, J. L., Whitney, D. E., et. al., "Exploratory Research in Industrial Modular Assembly," R-921, The Charles Stark Draper Laboratory, Inc., October 21, 1975.

[Nevins77]

Nevins, J. L., and Whitney, D. E., "Research on Advanced Assembly Automation", Computer, December, 1977.

[Paul72]

Paul, R., "Modelling, Trajectory Calculation and Servoing of a Computer Controlled Arm", Stanford Artificial Intelligence Laboratory Memo AIM-177, November 1972.

[Paul75]

Paul, R., "Manipulator Path Control", _Proceedings, 1975 International Conference on Cybernetics and Society_, 1975.

[Paul76]

Paul, R., and Shimano, B., "Compliance and Control", _Proceedings, 1976 Joint Automatic Control Conference_, 1976.

[Pieper68]

Pieper, D. L., "The Kinematics of Manipulators Under Computer Control", Stanford Artificial Intelligence Memo No. 72, October, 1968.

[Raibert77]

Raibert, M. H., _Motor Control and Learning by the State Space Model_, MIT Artificial Intelligence Laboratory Technical Report No. 439, September, 1977.

[Shimano77]

Shimano, B. E., "Force Control", _Exploratory Study of Computer Integrated Assembly Systems_, Binford, T. O., et al, Stanford Artificial Intelligence Laboratory Memo 285.4, June 1977.

[Silver73]

Silver, D., "The Little Robot System", MIT Artificial Intelligence Laboratory Memo No. 273, January, 1973.

[Simunovic75]

Simunovic, S. N., "Force Information in Assembly Processes", _Proceedings 5th International Symposium on Industrial Robots_, September 22-24, 1975.

[Simunovic76]

Simunovic, S. N., Task Descriptors for Automated Assembly, M. S. Thesis, MIT Department of Mechanical Engineering, January, 1976.

[Takase74]

Takase, K., Inoue, H., Sato, K., and Hagiwara, S., "The Design of an Articulated Manipulator with Torque Control Ability", Proceedings, 4th International Symposium on Industrial Robots, 19-21 November, 1974.

[Taylor76]

Taylor, R. H., "A Synthesis of Manipulator Control Programs from Task-Level Specifications", Stanford Artificial Intelligence Memo 282, July, 1976.

[Taylor77]

Taylor, R. H., "Planning and Execution of Straight-line Manipulator Trajectories", IBM Research Report RC-6657, July 25, 1977.

[Udupa77]

Udupa, S., Collision Detection and Avoidance in Computer Controlled Manipulators, California Institute of Technology PhD Thesis, 1977.

[Uicker64]

Uicker, J. J., Denavit, J., and Hartenberg, R. S., "An Iterative Method for the Displacement Analysis of Spatial Mechanisms", Journal of Applied Mathematics, Transactions of the ASME, June, 1964.

[Watson75]

Watson, P. C., and Drake, S. H., "Pedestal and Wrist Force Sensors for Automatic Assembly", Proceedings 5th International Symposium on Industrial Robots, September 22-24, 1975.

[Whitney69]

Whitney, D. C., "Resolved Motion Rate Control of Manipulators and Human Prostheses", IEEE Transactions on Man-Machine Systems, vol. MMS-10, no. 2, June, 1969.

[Whitney72]

Whitney, D. E., "The Mathematics of Coordinated Control of Prostheses and Manipulators", ASME Journal of Dynamic Systems, Measurement, and Control, December, 1972.

[Whitney76]

Whitney, D. E., "Force Feedback Control of Manipulator Fine Motions," Proceedings, 1976 Joint Automatic Control Conference.

[Will75]

Will, P. M., and Grossman, D. D., "An Experimental System for Computer Controlled Mechanical Assembly," IEEE Transactions on Computers, vol. c-24, no. 9, September, 1975.

[Young78]

Young, K. D., "Controller Design for a Manipulator Using Theory of Variable Structure Systems", IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-8, no. 2, February, 1978.