Chapter 3

Kinematics

Definition 3.1 (Kinematics) *a branch of dynamics that deals with aspects of motion apart from considerations of force and mass — Websters dictionary*

The Greek word for motion is *kinema*. Ampére used the term *cinématique* to refer to a body of science "...in which movements are considered in themselves ..." We will examine movements in solid bodies all about us, especially in the assemblages called machines [10]. A fundamental aspect of every robot control application is the characterization of spatial relationships. This section will introduce the subject by discussing free bodies in R^3. Following this, we will discuss how robot structures constrain motion and how forces and velocities are propagated along articulated mechanisms. During this discussion, we will develop the governing forward kinematic relationships for kinematic chains consisting of revolute and prismatic joints. We will introduce inverse kinematic analysis (a much harder problem, in general), and describe how Cartesian reference postures can be used to define joint positions for some simple robots. Methods for exploiting redundant degrees of freedom will be introduced allowing more motor flexibility at the expense of control overhead. We will conclude this chapter with a discussion of off-line inverse kinematic models and the pseudoinverse as a means of addressing multiple objectives in an articulated structure.

3.1 Terminology

The individual rigid bodies that collectively form a robot device are referred to as **links**. Links are connected in pairs through kinematic constraints called **joints**. A **prismatic** joint permits translations between links, implemented as a slider moving along a guide link. Typically the guide link is linear so that the slider executes a straight line motion. A **revolute** joint connects two

links through a rotational bearing. If the rotation is about a fixed axis, then the two links execute motions within the plane defined by the links. An assemblage of interconnected links is called a **kinematic chain**. A **mechanism** is formed when one of the links is held fixed and the others may move relative to the fixed link. The fixed link is referred to as the **ground link** or **frame**. Most mechanisms in the machine design literature consist of **closed chains**, that is, kinematic chains with every link connected through joints to two adjacent links. Most robotic devices are **open chains** wherein a link may only be connected to one joint (**unitary link**). Any parameter (length or angle) of an underconstrained mechanism is called a **configuration variable**. Devices for which multiple configuration variables must be specified are described in a **configuration space**. The minimum number of position variables necessary to fully define the configuration of a mechanism is called the **degree of freedom (DOF)** of the system.

3.2 Spatial Relationships

A free body moving in R^6 may translate in R^3 and may execute rotations $\mathbf{R} \in SO(3)$. To fully characterize the instantaneous position of this (rigid) body, it is necessary to specify 6 position variables with respect to some reference frame. A translation is naturally a vector in R^3, but there are several alternatives for representing rotations: exponential coordinates [8], Euler angles, and quaternions. In this chapter, we will consider homogeneous representations, in particular, the homogeneous transform. This representation treats translations and rotations in a uniform manner, and it is easy to compose and invert.

3.2.1 Translations

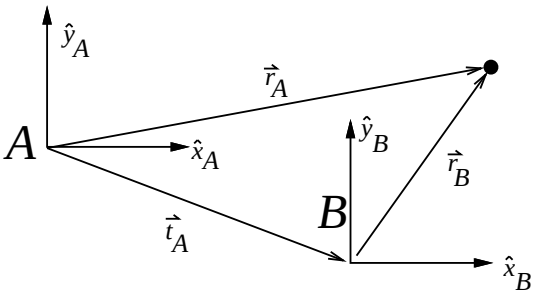


Figure 3.1 *Two Coordinate Frames related through a Pure Translation.*

Frames A and B in Figure 3.1 are related through a pure translation. A position vector expressed in frame B can be expressed in the A coordinate frame through the linear transformation

$$\vec{r}_A = \vec{r}_B + \vec{t}_A$$

where $\vec{t}_A$ represents the pure translation from frame A to frame B written in frame A coordinates.

3.2.2 Rotations

Frames A and B in Figure 3.2 are related through a pure rotation. A position vector expressed in frame B can be expressed in the A coordinate frame by employing the 3×3 transformation matrix ${}_A\mathbf{R}_B$,

$$\begin{aligned} \vec{r}_A &= {}_A\mathbf{R}_B \, \vec{r}_B \\ \begin{bmatrix} rx_A \\ ry_A \\ rz_A \end{bmatrix} &= \begin{bmatrix} \hat{i}_A \cdot \hat{i}_B & \hat{i}_A \cdot \hat{j}_B & \hat{i}_A \cdot \hat{k}_B \\ \hat{j}_A \cdot \hat{i}_B & \hat{j}_A \cdot \hat{j}_B & \hat{j}_A \cdot \hat{k}_B \\ \hat{k}_A \cdot \hat{i}_B & \hat{k}_A \cdot \hat{j}_B & \hat{k}_A \cdot \hat{k}_B \end{bmatrix} \begin{bmatrix} rx_B \\ ry_B \\ rz_B \end{bmatrix} \end{aligned}$$

where $\hat{i}, \hat{j}, \hat{k}$ represent the basis vectors for a coordinate frame.

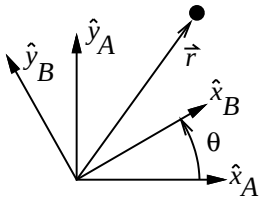


Figure 3.2 *Two Coordinate Frames related through a Pure Rotation.*

This projection of frame B onto frame A clearly converts a position vector, $\vec{r}_B$, written in frame B, into the corresponding coordinates in frame A, $\vec{r}_A$.

Another way to interpret the rotation matrix for this transformation follows from noticing that the rows of ${}_A\mathbf{R}_B$ represent the projection of the basis vectors for frame A onto the basis vectors of frame B. Conversely, the columns of ${}_A\mathbf{R}_B$ represent the basis vectors of frame B projected onto the basis vectors of frame A — the rotation matrix is just the direction cosine matrix. This suggests that the inverse of this matrix is just its transpose.

One way to specify the rotation matrix ${}_A\mathbf{R}_B$ is to write the basis vectors $(\hat{i}, \hat{j}, \hat{k})_B$ in frame A coordinates and to enter the result into the columns of ${}_A\mathbf{R}_B$. If $\hat{x}_B^A$ is a column vector representing the $\bar{x}$ axis of frame B written in frame A coordinates, then

$${}_A\mathbf{R}_B = \begin{bmatrix} \hat{x}_B^A & \hat{y}_B^A & \hat{z}_B^A \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

Notice that the rotation illustrated in Figure 3.2 is a positive sense rotation about the $\hat{z}$ axis. This explains why the direction of the $\hat{z}$ axis is preserved by this transformation — the only coordinate directions that are modified are the $\hat{x}$ and $\hat{y}$ directions. Notice also that Equation 3.1 encodes the transformation from frame A to frame B — ${}_A\mathbf{R}_B = rotation(\hat{z}, \theta)$.

For completeness, we will list the rotation matrix for rotations about all three axes:

$$rot(\hat{x}, \theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\theta) & -sin(\theta) \\ 0 & sin(\theta) & cos(\theta) \end{bmatrix} \tag{3.2}$$

$$rot(\hat{y},\theta) = \begin{bmatrix} cos(\theta) & 0 & sin(\theta) \\ 0 & 1 & 0 \\ -sin(\theta) & 0 & cos(\theta) \end{bmatrix} \tag{3.3}$$

$$rot(\hat{z},\theta) = \begin{bmatrix} cos(\theta) & -sin(\theta) & 0 \\ sin(\theta) & cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.4}$$

3.2.3 The Homogeneous Transformation

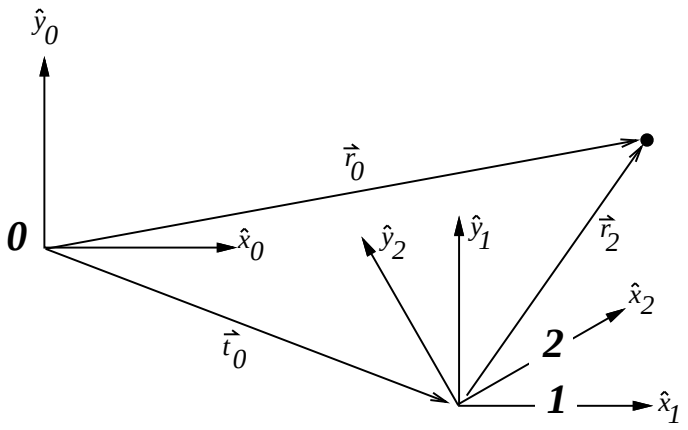


Figure 3.3 *Two Coordinate Frames related through a Rotation and a Translation.*

Now consider the general case where frame 0 and frame 2 are related to one another through both rotation and translation as shown in Figure 3.3. The homogeneous transform is a mechanism for expressing this form of compound transformation. Define transformation, ${}_0\mathbf{T}_2$, to be the compound transformation consisting of a translation from 0 to 1, followed by a rotation from 1 to 2. In vector notation, this homogeneous transformation and corresponding homogeneous position vectors are written:

$${}_0\mathbf{T}_2 = \left[\begin{array}{c|c} {}_1\mathbf{R}_2 & \vec{t}_0 \\ \hline 0\ 0\ 0 & 1 \end{array}\right] \qquad \vec{r}_2 = \begin{bmatrix} r_x \\ r_y \\ r_z \\ 1 \end{bmatrix}_2$$

Now,

$$\begin{aligned} \vec{r}_0 &= {}_0\mathbf{T}_2\vec{r}_2 \\ &= {}_1\mathbf{R}_2\vec{r}_2 + \vec{t}_0 \end{aligned}$$

EXAMPLE: The homogeneous transform provides a convenient means of constructing compound transformations. Figure 3.4 represents the relationship between frames 0 and 4 in terms of three intermediate coordinate frames.

$$ {}_0\mathbf{T}_4 = {}_0\mathbf{T}_1 \ {}_1\mathbf{T}_2 \ {}_2\mathbf{T}_3 \ {}_3\mathbf{T}_4 $$

where:

$$ \begin{aligned} {}_0\mathbf{T}_1 &= translation(\hat{x}_0, 1.0) \\ {}_1\mathbf{T}_2 &= translation(\hat{y}_1, 1.0) \\ {}_2\mathbf{T}_3 &= translation(\hat{z}_2, 1.0) \\ {}_3\mathbf{T}_4 &= rotation(\hat{y}_3, -\pi/4) \end{aligned} $$

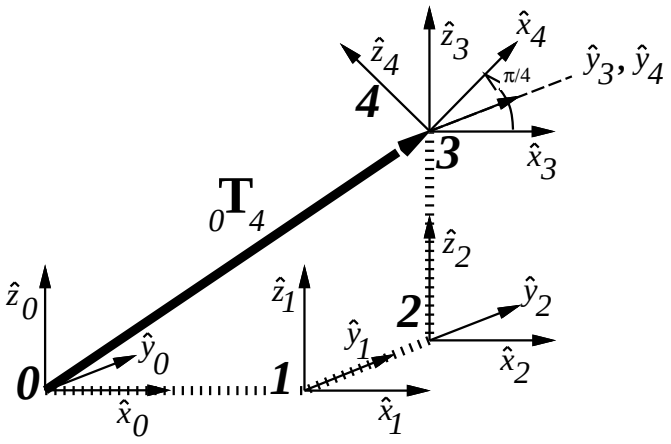


Figure 3.4 *A compound transformation.*

The resulting compound transformation in this example is:

$$ {}_0\mathbf{T}_4 = \begin{bmatrix} 0.707 & 0 & -0.707 & 1 \\ 0 & 1 & 0 & 1 \\ 0.707 & 0 & 0.707 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

It is always a good exercise to check the result, so let's select a couple of position vectors expressed in frame B and check.

CASE 1: let $\vec{r}_4 = (0, 0, 0, 1)$ This position vector locates the origin of frame 4. The corresponding position vector in frame 0 is:

$$ \vec{r}_0 = \begin{bmatrix} 0.707 & 0 & -0.707 & 1 \\ 0 & 1 & 0 & 1 \\ 0.707 & 0 & 0.707 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} $$

CASE 2: let $\vec{r}_4 = (1, 0, 0, 1)$ This position vector locates the endpoint of the $\hat{x}$ axis for frame 4. The corresponding position vector in frame 0 is:

$$ \vec{r}_0 = \begin{bmatrix} 0.707 & 0 & -0.707 & 1 \\ 0 & 1 & 0 & 1 \\ 0.707 & 0 & 0.707 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 1.707 \\ 1 \\ 1.707 \\ 1 \end{bmatrix} $$

Inverting the Homogeneous Transform The inverse of the homogeneous transform is simple to derive, we will simply show you the result. It is a useful exercise to verify that it is correct.

$$ {}_A\mathbf{T}_B = \begin{bmatrix} \hat{x}_B^A & \hat{y}_B^A & \hat{z}_B^A & \vec{t} \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}_B\mathbf{T}_A = [\,{}_A\mathbf{T}_B]^{-1} = \begin{bmatrix} (\hat{x}_B^A)^T & (-\vec{t}\cdot\hat{x}_B^A) \\ (\hat{y}_B^A)^T & (-\vec{t}\cdot\hat{y}_B^A) \\ (\hat{z}_B^A)^T & (-\vec{t}\cdot\hat{z}_B^A) \\ 0\;\;0\;\;0 & 1 \end{bmatrix} $$

3.3 Forward Kinematics

In a typical robotics application several frames of reference may be required to fully specify the state of the robot and the task. Figure 3.5 illustrates several such meaningful coordinate frames for a robotic task. Every application must have an **inertial** reference frame. In Figure 3.5, the **world** frame serves this purpose — all motions and forces are expressed with respect to this reference. The **base** frame for the robot is expressed with respect to the inertial coordinate system. This frame may be a fixed transformation from the world frame, or it may be controlled as in the case of a mobile platform. For manipulators, there is typically a **wrist** frame specified at the end of the manipulator which specifies the endpoint of the controllable degrees of freedom with respect to the base frame. The **tool** frame may be used to locate an important feature of a grasped object which is relevant to a task defined **goal** frame located at some position on an object specified by the **object** frame. Finally, the geometry of the workpiece may be specified relative to a **station** frame.

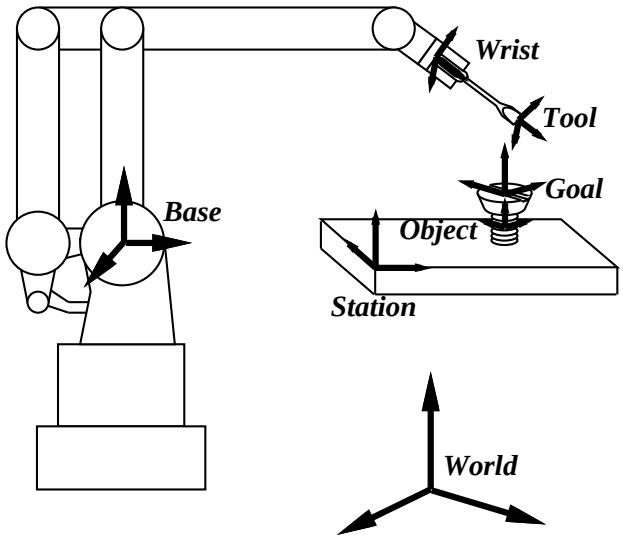


Figure 3.5 *Standard Coordinate Frames Common to Robotics Applications*

Nearly every robotics application begins with a process of kinematic identification — the relative positions and orientations of agents and/or objects in the workspace must be determined. This relationship may be determined *a priori* as, for example, when the station frame is located with respect to the robot's base frame provided that this relationship is static, or it may be determined on-line as, for example, when the goal is located with respect to the station frame. Moreover, these relationships can be identified in one of the Cartesian frames or in the joint angles (or *configuration space*) of the robot. A special case of this kinematic identification, commonly called **forward**

kinematics, involves locating the position and orientation of the wrist frame of the manipulator in Cartesian space as a function of the controllable degrees of freedom. This transformation maps a vector of joint angles, $\vec{\theta}$, into a position and orientation of the manipulator endpoint.

Consider the simple 2 DOF, planar manipulator illustrated in Figure 3.6. The position of the endpoint of this manipulator can be determined by inspection. The position of frame 2 with respect to frame 0 is just $(l_1 cos(\theta_1), l_1 sin(\theta_1)$. The position of frame 3 with respect to frame 2 written in frame 0 coordinates is likewise $(l_2 cos(\theta_1 + \theta_2), l_2 sin(\theta_1 + \theta_2))$, so that the position of frame 3 with respect to frame 0 can be written:

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

The orientation of the endpoint frame can likewise be determined easily. It is just a counterclockwise rotation of the $\hat{x}_0$ and $\hat{y}_0$ axes by an amount $(\theta_1 + \theta_2)$. The direction of the $\hat{z}$ axis is preserved.
This forward kinematic problem can be significantly harder in more general manipulators. In these situations, the homogeneous transform provides a basis for computing the position and orientation of the endpoint. The net transform from the base of the robot to the frame attached to its endpoint can be expressed as a sequence of simple transformations.

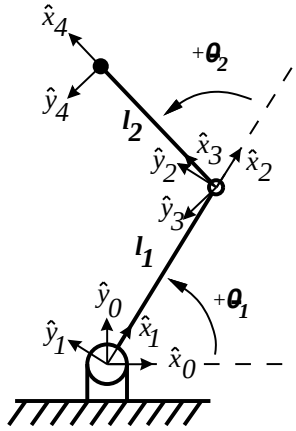


Figure 3.6 *The Two Degree of Freedom, Planar Robot.*

$$\begin{aligned} {}_0\mathbf{T}_4 &= {}_0\mathbf{T}_1\ {}_1\mathbf{T}_2\ {}_2\mathbf{T}_3\ {}_3\mathbf{T}_4 \\ &= rot(\hat{z}_0, \theta_1)\ trans(\hat{x}_1, l_1)\ rot(\hat{z}_2, \theta_2)\ trans(\hat{x}_3, l_2) \\ &= \begin{bmatrix} c_1 & -s_1 & 0 & 0 \\ s_1 & c_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & l_1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} c_2 & -s_2 & 0 & 0 \\ s_2 & c_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & l_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

3.4 Manipulator Workspace

Due to finite link lengths and joint angle ranges, the configuration space of a robot corresponds to a proper subset of Cartesian space through the forward kinematic transformation. This subset is known as the **reachable** workspace. Returning to the 2 DOF planar manipulator example, we may define the reachable workspace by enumerating all positions in Cartesian space that are

realizable endpoint positions for some manipulator configuration. It is clear that every configuration executed in the course of a task must remain within the reachable workspace. Figure 3.7 illustrates the reachable workspace for our 2D planar manipulator as a function of relative link length.

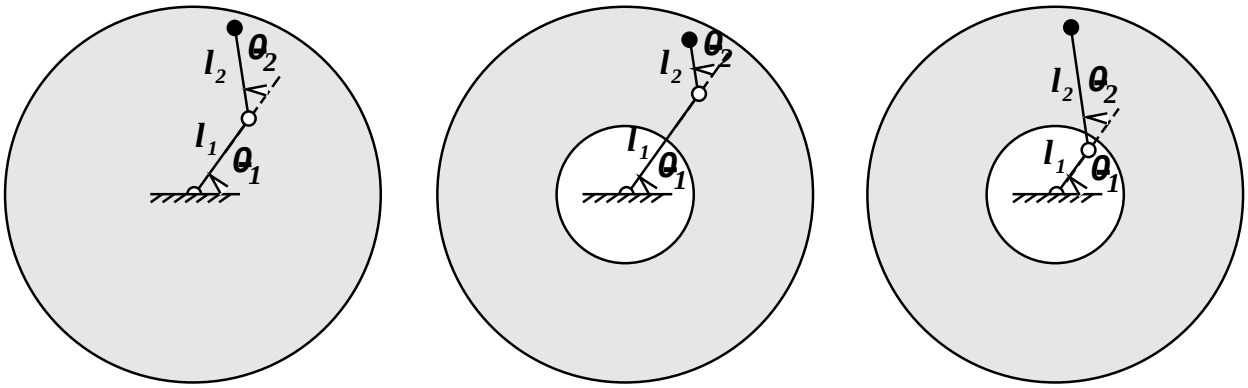


Figure 3.7 *The Reachable Workspace for the Two Degree of Freedom, Planar Robot.*

Simply reaching a reference position is not always good enough. Often geometric task constraints require that the end effector can achieve particular orientations. It would simplify the process of planning a manipulator trajectory significantly if the subset of Cartesian space in which arbitrary orientations could be achieved could be enumerated — this space is sometimes referred to as the **dextrous** workspace. If the trajectory executed in the context of a task remains entirely within the reachable workspace and the goal configuration remains within the dextrous workspace, then a motion planner can be assured that a solution exists. If, on the other hand, either of these conditions is not satisfied, it is possible that subtasks may be required to satisfy all task constraints during execution. Returning to the example in Figure 3.7, we completely characterize the workspace by noting that positions on the outermost boundary of the workspace are indeed reachable, but only when $\theta_2 = 0$. This implies that there is a single joint angle configuration for each of the Cartesian locations on the outside perimeter of the workspace. This is true regardless of the relative link lengths. In these singular configurations, the manipulator can reach the endpoint position, but can only do so with a single endpoint orientation. The workspace region between the inside and outside perimeter is reachable with exactly two alternative configurations. In the case when $l_1 = l_2$, the origin is reachable in an infinite number of configurations and can therefore generate all endpoint orientations in the plane. Therefore, a single point in Cartesian space constitutes the dextrous workspace, and this point exists only when $l_1 = l_2$.

In general, precomputing the dextrous workspace is very difficult. However, the analysis can be facilitated by clever mechanical design. It is always possible to construct a dextrous workspace over

the entire reachable workspace by attaching a spherical wrist to the manipulator endpoint. The spherical wrist allows all endpoint orientations at every reachable position by introducing three orthogonal revolute joints whose axes intersect at a point. This construction is valid only if the center of the spherical joint coincides with the manipulator endpoint. If this is not true, then the dextrous workspace will contract relative to the reachable workspace by an amount proportional to the geometric offset between the endpoint and the center of the spherical wrist. It is interesting to note that the human wrist embodies this design principle with humoral rotation, flexion/extension, and adduction/abduction, corresponding to roll, pitch, and yaw axes respectively, intersecting near the base of the palm.

3.5 Inverse Kinematics

The homogeneous transform provides a tool for computing the forward kinematic transformation for any manipulator. However, the task often implies a Cartesian entity which must be mapped into a joint angle configuration for the robot. This is the so-called inverse kinematic transformation. The mapping from Cartesian space to joint angle configuration, $\vec{X} \mapsto \vec{\Theta}$, is more complicated than the forward kinematic relationship. Certain regions of the Cartesian space are unreachable and others can be achieved by multiple robot configurations.

The inverse kinematic problem has been addressed in a variety of ways: Pieper (ca. 1968) formulated a general inverse kinematic solution to 6 DOF manipulators consisting of 3 revolute or prismatic joints followed by three consecutive joints with rotational axes that intersect at a point, Paul (ca. 1981) demonstrated a technique based on homogeneous transforms for the same class of manipulators treated by Pieper, it is a simple matter to formulate an iterative approach to the inverse kinematic problem, but not without considerable computational expense, and off-line techniques are very common in practical implementations.

For the sake of this discussion, we will consider only geometric techniques of generating complete inverse kinematic solutions. We will analyze a very simple 2 DOF planar manipulator to demonstrate the complexity of the problem.

3.5.1 Geometric Inverse Kinematic Solutions

The inverse kinematic solution for the two degree of freedom, planar manipulator illustrated in Figure 3.6 is considerably more involved than the forward kinematic solution.

The algorithm for computing inverse kinematic solutions must first check for the existence of a solution by verifying that the Cartesian target is inside the reachable workspace.

Figure 3.8 defines the geometric constructions used to solve the inverse kinematics. First, to eliminate θ_1 and solve for two unique θ_2 solutions, we note that $r^2 = x^2+y^2$ and from the forward kinematics, $x = l_1c_1 + l_2c_{12}$ and $y = l_1s_1 + l_2s_{12}$. In the following, we will often employ the trigonometric identities: $s_{12} = s_1c_2 + c_1s_2$ and $c_{12} = c_1c_2 - s_1s_2$.

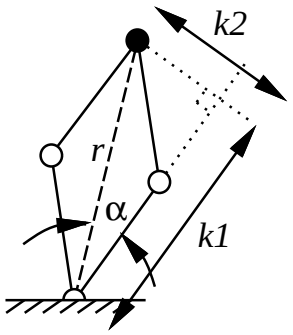


Figure 3.8 *Auxiliary Variables for Computing Joint Angle Configurations.*

$$\begin{aligned} r^2 = x^2 + y^2 &= l_1^2c_1^2 + 2l_1l_2c_1c_{12} + l_2^2c_{12}^2 && (3.5)\\ &\quad +l_1^2s_1^2 + 2l_1l_2s_1s_{12} + l_2^2s_{12}^2 && (3.6)\\ &= l_1^2 + 2l_1l_2c_2 + l_2^2 && (3.7)\end{aligned}$$

We may now rearrange terms to solve for c_2,

$$c_2 = \frac{r^2 - l_1^2 - l_2^2}{2l_1l_2}.$$

This ratio captures geometric aspects of the manipulator and the task and permits us to check to see if the target is reachable. It is clear that $-1 \leq c_2 \leq +1$. If this is not true for a given r, l_1, and l_2, then the target is not reachable.

All target locations that prove to be reachable generate two (possibly repeated) manipulator configurations. The only exception to this rule will be when $l_1 = l_2$ and $r = 0$. This manipulator geometry possesses an infinite number of configurations for the target position. To solve for both θ_2 solutions,

$$\begin{aligned} s_2^2 + c_2^2 &= 1 && (3.8)\\ s_2^2 &= 1 - c_2^2 && (3.9)\\ s_2^{+/-} &= +/-(1 - c_2^2)^{1/2} && (3.10)\\ & && (3.11)\end{aligned}$$

so that,

$$\theta_2^{+/-} = tan^{-1}\frac{s_2^{+/-}}{c_2}. \qquad (3.12)$$

In order to solve for corresponding configurations for θ_1, we note that

$$\alpha^{+/-} = tan^{-1}\frac{k_2^{+/-}}{k_1}$$

where k_1 and k_2 are defined in Figure 3.8

$$k_1 = rc_\alpha = l_1 + l_2c_2 \tag{3.13}$$
$$k_2^{+/-} = rs_\alpha = l_2s_2^{+/-} \tag{3.14}$$

Therefore,

$$x = k_1c_1 + k_2s_1 = (rcos_\alpha)c_1 + (rsin_\alpha)s_1 \tag{3.15}$$
$$= rcos(\alpha + \theta_1) \tag{3.16}$$
$$y = k_1s_1 + k_2c_1 = (rcos_\alpha)s_1 + (rsin_\alpha)c_1 \tag{3.17}$$
$$= rsin(\alpha + \theta_1) \tag{3.18}$$

and

$$tan(\alpha + \theta_1) = \frac{rsin(\alpha + \theta_1)}{rcos(\alpha + \theta_1)} = \frac{y}{x}$$

so that,

$$\theta_1^{+/-} = tan^{-1}\frac{y}{x} - \alpha^{+/-}.$$

Figure 3.9 summarizes the algorithm for computing the complete inverse kinematic solution of the 2R, planar manipulator.

GIVEN (x,y) endpoint position goal:

$r^2 = x^2 + y^2$
$c_2 = (r^2 - l_1^2 - l_2^2)/(2l_1l_2)$
if $(-1 \leq c_2 \leq +1)$
    $s_2^{+/-} = +/- (1 - c_2^2)^{1/2}$
    $\theta_2^{+/-} = tan^{-1}\frac{s_2^{+/-}}{c_2}$
    $k_1 = l_1 + l_2c_2$
    $k_2^{+/-} = l_2s_2^{+/-}$
    $\alpha^{+/-} = tan^{-1}\frac{k_2^{+/-}}{k1}$
    $\theta_1^{+/-} = tan^{-1}\frac{y}{x} - \alpha^{+/-}$
else "out of reach"

Figure 3.9 *The IK algorithm for the 2R, planar manipulator*

3.6 The Manipulator Jacobian

The planar, two degree of freedom manipulator treated in Section 3.3 (see Figure 3.6) yielded expressions that defined the endpoint position, (x, y), as a function of the joint angle configuration, (θ_1, θ_2),

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

This forward kinematic relationship is clearly nonlinear, since the mapping cannot be expressed in the form $\vec{X} = A\vec{\Theta}$. This is precisely the reason that the inverse kinematic transformation is difficult to compute — it is not a simple linear inverse relationship. However, in the vicinity of the current state, the first derivative of the forward kinematic relationship defines a locally linear relationship

$$d\vec{X} = Jd\vec{\Theta}.$$

This transformation is a linear (and invertible) approximation of the mapping from joint space *velocities*, $d\vec{\Theta}$, to Cartesian space endpoint velocities, $d\vec{X}$. However, the relationship is valid only for the instantaneous configuration of the manipulator.

The Jacobian of the planar manipulator is:

$$\partial x = -l_1 sin(\theta_1)\partial\theta_1 - l_2 sin(\theta_1 + \theta_2)\partial\theta_1 - l_2 sin(\theta_1 + \theta_2)\partial\theta_2 \quad (3.19)$$

$$\partial y = l_1 cos(\theta_1)\partial\theta_1 + l_2 cos(\theta_1 + \theta_2)\partial\theta_1 + l_2 cos(\theta_1 + \theta_2)\partial\theta_2 \quad (3.20)$$

which can be written:

$$\begin{bmatrix} \partial x \\ \partial y \end{bmatrix} = \begin{bmatrix} -l_1 s_1 - l_2 s_{12} & -l_2 s_{12} \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} \end{bmatrix} \begin{bmatrix} \partial\theta_1 \\ \partial\theta_2 \end{bmatrix}. \quad (3.21)$$

With the manipulator Jacobian, we may compute the Cartesian endpoint velocity that will result from a given joint space velocity. However, most often we are interested in computing the joint space velocities that will generate a desired Cartesian endpoint velocity, $\dot{\vec{\Theta}} = J^{-1}\dot{\vec{X}}$. This is indeed possible for the planar manipulator since its Jacobian is 2×2, but the inverse relation is ill-conditioned in the neighborhood of singularities. Singularities occur whenever the Jacobian looses rank, i.e., whenever the determinate of the Jacobian is zero. Geometrically, this happens when: (1) the endpoint position lies on the axis of a joint — velocities in this joint will not generate any endpoint velocity, and (2) when the endpoint lies within the plane defined by two joint axes — velocities in either of these joints generate collinear endpoint velocities. In these situations, a desired endpoint velocity can require infinite joint angle velocities.

EXAMPLE: If we compute the determinate of the Jacobian for the planar, 2D manipulator defined in Equation 3.21

$$\begin{vmatrix} -l_1 s_1 - l_2 s_{12} & -l_2 s_{12} \\ l_1 c_1 + l_2 c_{12} & l_2 c_{12} \end{vmatrix} = -l_1 l_2 s_1 c_{12} - l_2^2 s_{12} c_{12} + l_1 l_2 c_1 s_{12} + l_2^2 c_{12} s_{12}$$
$$= l_1 l_2 (c_1 s_{12} - s_1 c_{12})$$
$$= l_1 l_2 s_2.$$

It follows that the Jacobian is singular when $sin(\theta_2) = 0$, or when $\theta_2 = 0, \pi$. These configurations of the manipulator generate the boundary of the workspace: $\theta_2 = 0$ generates the outside- and $\theta_2 = \pi$ the inner-radius of the reachable workspace (Figure 3.7). In these singular configurations, the endpoint of the manipulator can not move instantaneously in two dimensions — it is constrained to move in a one dimensional subspace. If requested endpoint velocities are not in these subspaces, then infinite joint space velocities can result.

Suppose that this manipulator is to execute an endpoint velocity, $V = 1\frac{m}{sec}$ in the $\hat{x}_0$ direction as illustrated in Figure 3.10. Since the Jacobian for the 2D, planar manipulator (Equation 3.21) is 2×2, it is a simple matter to invert: $J^{-1} = \frac{1}{det(J)}[cofactors\ of\ J]^T$, yielding the mapping $\dot{\vec{\Theta}} = J^{-1}\dot{\vec{X}}$. The result is:

$$J^{-1} = \frac{1}{l_1 l_2 s_2} \begin{bmatrix} l_2 c_{12} & l_2 s_{12} \\ -l_1 c_1 - l_2 c_{12} & -l_1 s_1 - l_2 s_{12} \end{bmatrix}$$

so that,

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \frac{1}{l_1 l_2 s_2} \begin{bmatrix} l_2 c_{12} & l_2 s_{12} \\ -l_1 c_1 - l_2 c_{12} & -l_1 s_1 - l_2 s_{12} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$
$$= \begin{bmatrix} \frac{c_{12}}{l_1 s_2} \\ -\frac{c_1}{l_2 s_2} - \frac{c_{12}}{l_1 s_2} \end{bmatrix} \frac{rad}{sec}$$

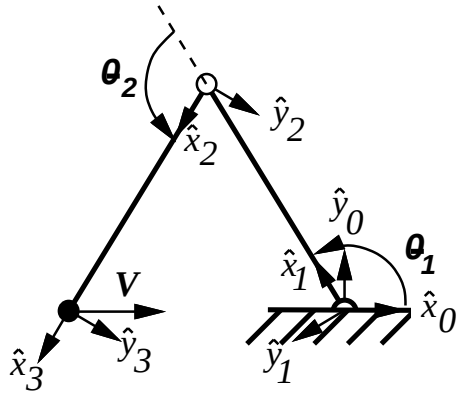


Figure 3.10 *The 2D, planar manipulator and a specific endpoint velocity command.*

Once again, note that when $s_2 \to 0$, $\dot{\vec{\theta}} \to \infty$ — this is the singularity in the Jacobian. Therefore, when $\theta_2 = 0, \pi$, the inverse Jacobian is singular. In practice, velocity controllers based on inverse Jacobians will have difficulty tracking Cartesian trajectories in the neighborhood of kinematic singularities. Cartesian path planners must either compute paths that avoid kinematic singularities, or they must be robust with respect to deviations from the path.

3.6.1 Principle Kinematic Transformations

> "...posture variation is a means through which motion and strength characteristics of the arm is made compatible with the task [1]."

The Jacobian transforms the unit sphere in joint velocity space:

$$\|\dot{\theta}\|^2 = \dot{\theta}_0^2 + \dot{\theta}_1^2 + \ldots + \dot{\theta}_m^2 \leq 1$$

to an ellipsoid in Cartesian space, since $\dot{x} = J\dot{\theta}$, and

$$\dot{\theta}^T\dot{\theta} = \|\dot{\theta}\|^2 = (J^{-1}\dot{x})^T(J^{-1}\dot{x}) = \dot{x}^T[(J^{-1})^T J^{-1}]\dot{x} = \dot{x}^T(JJ^T)^{-1}\dot{x}.$$

We find that $(JJ^T)^{-1}$ behaves like a configuration dependent amplifier from joint velocities to Cartesian endpoint velocities.

In the force domain,

$$\tau^T\tau = (J^T f)(J^T f) = f^T(JJ^T)f$$

so that the force amplifier is (JJ^T). The eigenvectors of (JJ^T) and $(JJ^T)^{-1}$ are identical. Moreover, the eigenvalues of (JJ^T) (force amplifier) are reciprocals of eigenvalues of $(JJ^T)^{-1}$ (velocity amplifier).

The principle axes of the conditioning ellipsoid, $\vec{e}$ are characterized by the eigenvectors $(\hat{e}_i)$ of (JJ^T). The singular values of J are the square roots of the eigenvalues of (JJ^T), $\sigma_i = \sqrt{e_i}$.

Moreover, the singular vectors of J^T are the same as the eigenvectors of (JJ^T). An introduction to the singular value decomposition is presented in Appendix **??**. The eigenvectors of (JJ^T) represent the directions in Cartesian space of the principle transformations from joint torques to endpoint forces. The square root of the corresponding eigenvalue indicates the relative degree of amplitude or precision in force transformation.
The velocity ellipsoid is characterized by the principal axes, $(1/\sqrt{e_i})\hat{e}_i$. A small JJ^T eigenvalue amplifies comparatively small $\dot{\theta}$ into large $\dot{x}$. Large eigenvalues imply a direction in space along which comparatively large joint velocities correspond to small endpoint velocities. These configurations, although unfavorable from the perspective of velocity amplitude, can lead to precision velocity control since joint velocity sensors have higher effective resolution in these directions.

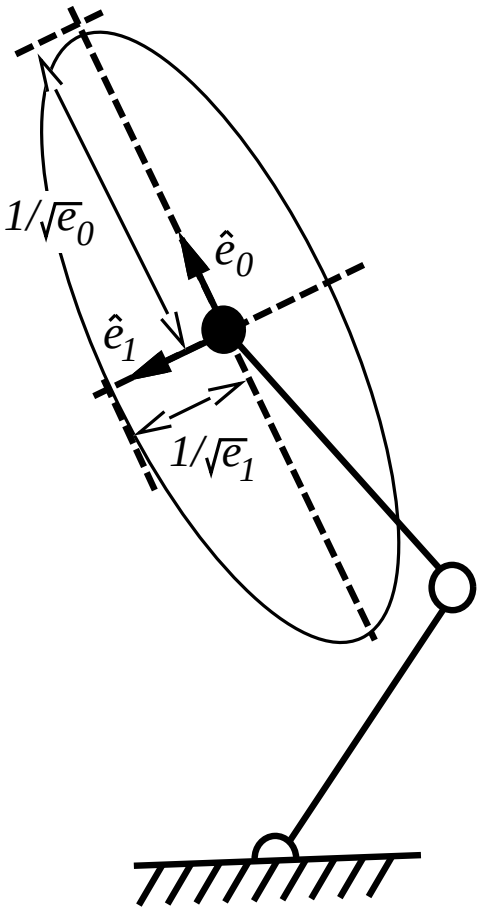


Figure 3.11 *A Velocity Ellipsoid derived from (JJ^T) for the 2D Manipulator Jacobian.*

The Jacobian can thus be thought of as a configuration dependent amplifier from joint space to Cartesian space — effectively trading amplitude for precision in the velocity domain. Several scalar metrics have been proposed in the literature to characterize this transformation.

3.6.2 Singularity Avoidance by Back Projection

A Cartesian velocity can be linearly decomposed into components along basis vectors defined by the singular vectors of J. The singular values represent a configuration dependent ability to transform joint angle velocities into Cartesian velocities. Subspaces in which this transformation is ill-conditioned correspond to the space spanned by singular vectors with singular values less than some threshold. Small singular values suggest that velocities in these directions require relatively

large joint angle velocities to track Cartesian velocity commands. However, if errors in the resulting Cartesian trajectory can be tolerated, then Cartesian velocities that project onto ill-conditioned subspaces can be ignored and a modified Cartesian velocity reference can be executed. The result is a command Cartesian velocity that remains within relatively well-conditioned subspaces of the manipulator Jacobian. A scheme like this requires that the errors in the resulting velocity commands do not accumulate. If a deviation in the planned trajectory occurs, then subsequent velocity commands must compensate. Note also that a mechanism such as this one for avoiding singularities will never be able to converge to a singular configuration.

3.6.3 Scalar Kinematic Conditioning Metrics

Since the Jacobian captures the map from configuration space to Cartesian space in a locally linear transformation, it is amenable to to a large variety of tools from linear systems theory to characterize the local quality of the transformation. Scalar conditioning metrics have been used as a design tool for relating manipulator geometry to a task domain [11], and as scalar objective fields for optimizing the inverse kinematic solution [1, 2, 5, 6, 7, 9, 12, 13]. In this section, we will introduce commonly used scalar metrics and illustrate how these metrics can be used to address kinematic conditioning as a control task. In Section 3.7.1, we will introduce a method for addressing kinematic conditioning in the context of redundant manipulators.

Minimum Singular Value The minimum singular value of the manipulator Jacobian indicates how close the system is to a kinematic singularity where the Jacobian will loose rank. As the manipulator approaches a singularity, the minimum singular value decreases until it reaches zero at the singular configuration.

Condition Number If $\dot{\tilde{\theta}}$ is an approximate solution to $J\dot{\theta} = \dot{x}$, then we may write $d\dot{x} = \dot{x} - J\dot{\tilde{\theta}}$ so that $\dot{\theta} - \dot{\tilde{\theta}} = J^{-1}d\dot{x}$. Therefore,

$$||\dot{\theta} - \dot{\tilde{\theta}}|| = ||J^{-1}d\dot{x}|| \leq ||J^{-1}||\ ||d\dot{x}||.$$

Since $\dot{x} = J\dot{\theta}$, $||\dot{x}|| \leq ||J||\ ||\dot{\theta}||$, and $||\dot{\theta}|| \geq ||\dot{x}||/||J||$, so that

$$\begin{aligned} \frac{||\dot{\theta} - \dot{\tilde{\theta}}||}{||\dot{\theta}||} &\leq \frac{||J^{-1}||\ ||d\dot{x}||}{||\dot{x}||/||J||} \\ &\leq ||J||\ ||J^{-1}||\frac{||d\dot{x}||}{||\dot{x}||} \end{aligned}$$

and

$$\frac{||d\dot{\theta}||}{||\dot{\theta}||} = \kappa(J)\frac{||d\dot{x}||}{||\dot{x}||} \qquad \kappa(J) = ||J||||J^{-1}||.$$

The *condition number*, $1 \leq \kappa(J) \leq \infty$, describes the error amplification capacity of the Jacobian J. When $\kappa = 1$, the manipulator configuration is spatially isotropic, small errors in $\dot{\theta}$ correspond to small errors in $\dot{x}$. As $\kappa \to \infty$, the manipulator is approaching a kinematic singularity.

The condition number of matrix J can also be defined as the ratio of the maximum to the minimum singular value of J. The condition number of the manipulator Jacobian, therefore, describes the eccentricity of the conditioning ellipsoid. The inverse of the condition number is perhaps more relevant,

$$\frac{1}{\kappa} = \frac{\sigma_{min}(J)}{\sigma_{max}(J)} \tag{3.22}$$

which varies continuously between zero (singular configurations) and unity (isotropic configurations).

Determinant of the Manipulator Jacobian The square root of the determinant of JJ^T (or the product of the singular values of J) is an approximation of the volume of the conditioning ellipsoid. The volume of the ellipsoid can be appreciable even though the configuration may be approaching a singularity, but in general, volume increases as the conditioning ellipsoid becomes more spherical — spatially isotropic. The *manipulability* metric

$$p = \sqrt{detJJ^T}$$

is a common choice as a scalar conditioning metric.

Figure 3.1 depicts the scalar manipulability function for the three distal finger joints of the Utah/MIT dextrous hand [4]. The relative link lengths are illustrated in Figure 3.12. The manipulability field was computed in the plane of the last three joints and joint range limits were ignored. The finger is redundant since there are more joints $(\theta_1, \theta_2, \theta_3)$ than dimensions in the plane (x, y) (we will expand on the notion of redundancy in Section 3.7). The function illustrated shows only the most manipulable configuration of the finger at each position in the workspace. The model so obtained is an off-line inverse kinematic model since for any position in the plane for the fingertip, the model returns the optimally manipulable finger configuration. These kind of inverse kinematic models are very efficient at run-time, and therefore very common in practice.

In addition to providing an inexpensive inverse kinematic map, these metric spaces can be used to advantage as kinematic *controllers*. In the context of a dextrous hand that is interacting with a

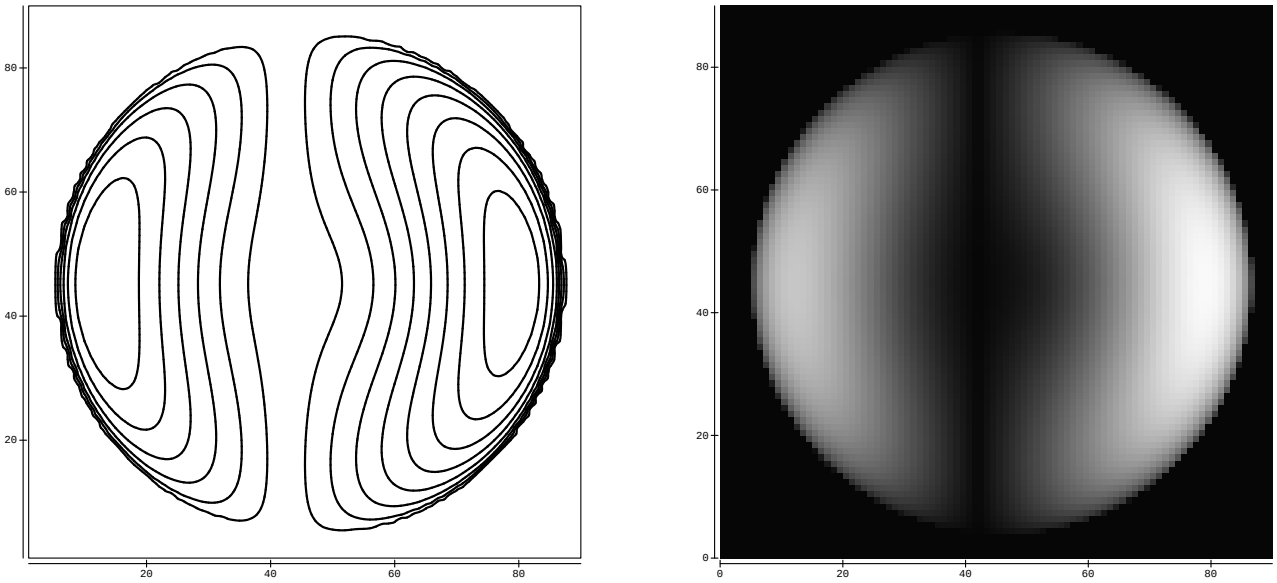


Figure 3.1: Maximum manipulability model for the Utah/MIT finger.

workpiece, it is often useful to optimize the kinematic isotropy of the finger by executing displacements in the arm. Such an approach preserves the ability of the finger to execute fine motions and/or delicate contact forces.

For a single finger, the postural Jacobian is derived:

$$\frac{dM}{d\vec{\theta}_{arm}} = \underbrace{\frac{\partial M}{\partial \vec{x}}}_{gradient\ field} \quad {}_f\mathbf{R}_{arm} \quad \underbrace{\frac{\partial \vec{X}}{\partial \theta_{arm}}}_{arm\ Jacobian}$$

where $\vec{x}$ is a Cartesian displacement of the finger in it's local coordinate frame, $\vec{X}$ is a Cartesian displacement of the arm in it's local coordinate system, and ${}_f\mathbf{R}_{arm}$ is just the rotation matrix from arm coordinates to finger coordinates. A controller can now be devised that adjusts the arm configuration to optimize the finger posture by ascending the gradient expressed by the postural Jacobian and holding the finger fixed in space.

$$\Delta\vec{\theta}_{arm} = K_m * \frac{dM}{d\vec{\theta}_{arm}}$$

Figure 3.12 illustrates the reachable workspace for the Utah/MIT finger and shows the trajectories of steepest ascent in the postural Jacobian toward the optimal finger configuration.

3.7 Redundant Manipulators

A manipulator is *redundant* if it possesses more controllable degrees of freedom than are required to regulate the $n \leq 6$ dimensional Cartesian state of the endpoint. Redundant manipulators can have an infinite number of inverse kinematic solutions for a given reference position and may execute an infinite variety of $\dot{\vec{\theta}}$ in order to generate a given $\dot{\vec{x}}$. Under these circumstances, the inverse kinematic problem is ill-posed, leading to the a so-called *null space* in the manipulator Jacobian. Motions in the null space produce no endpoint velocity as they move across the *self-motion manifold* for the given endpoint position[1].

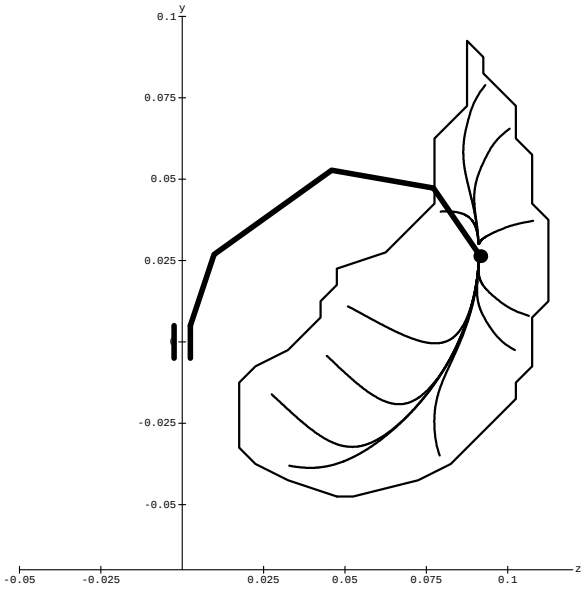


Figure 3.12 *Relative fingertip trajectories resulting from the manipulability-based postural Jacobian.*

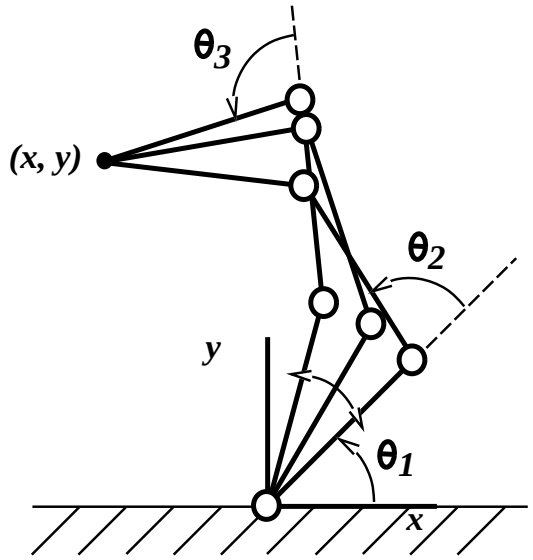


Figure 3.13 *Configurations on the same Self-Motion for a 3R manipulator.*

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

[1]The availability of multiple inverse kinematic solutions is not equivalent to the notion of dextrous workspace presented earlier. Although multiple solutions exist in non-redundant manipulators, they are distinct points in configuration space, while redundant systems may move continuously along the self-motion manifold.

therefore, the *self-motion manifold* is defined by:

$$\begin{Bmatrix} \dot{x} \\ \dot{y} \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \end{Bmatrix} = [J] \begin{Bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix}$$

$$= \begin{bmatrix} -l_1 s_1 - l_2 s_{12} - l_3 s_{123} & -l_2 s_{12} - l_3 s_{123} & -l_3 s_{123} \\ l_1 c_1 + l_2 c_{12} + l_3 c_{123} & l_2 c_{12} + l_3 c_{123} & l_3 c_{123} \end{bmatrix} \begin{Bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{Bmatrix}$$

For this case, internal motions along the self-motion manifold are constrained to be in the *null space* of the manipulator Jacobian:

$$\dot{\vec{\theta}}_{null} = \begin{Bmatrix} l_2 l_3 sin(\theta_3) \\ -l_2 l_3 sin(\theta_3) - l_1 l_3 sin(\theta_2 + \theta_3) \\ l_1 l_2 sin(\theta_2) + l_1 l_3 sin(\theta_2 + \theta_3) \end{Bmatrix}$$

If we follow this null space vector through a sequence of configurations, we can generate the self-motion manifold. A motion along the self-motion manifold is called an *internal motion*. The resulting flexibility can be exploited in many ways. The configuration space trajectory can be chosen to avoid kinematic singularities, to address force and velocity constraints imposed by the task, or to optimize the configuration with respect to a cost function during the execution of a Cartesian task.

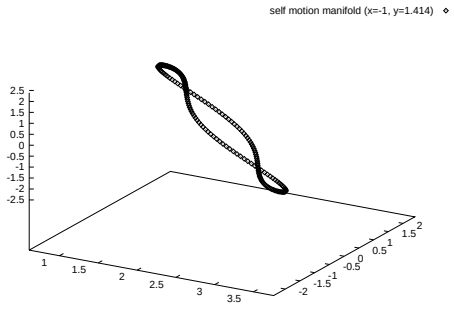


Figure 3.14 *A trace along the Self-Motion for a 3R manipulator.*

3.7.1 The Pseudoinverse

Consider the case when the Jacobian is not square,

$$\dot{\vec{x}}_{n\times 1} = J_{n\times m} \dot{\vec{\theta}}_{m\times 1}$$

Given the desired end-effector path in Cartesian space, we would like to find a corresponding path in configuration space. To do so, we define the residual $\vec{\rho} = \dot{\vec{x}} - J\dot{\vec{\theta}}$ from which a quadratic error is computed,

$$E = \left(\dot{\vec{x}} - J\dot{\vec{\theta}}\right)^T \left(\dot{\vec{x}} - J\dot{\vec{\theta}}\right).$$

The Hessian of E is positive semidefinite for any given J, so J defines a unique minimum. Error is minimized with respect to $\dot{\vec{\theta}}$ when

$$\begin{aligned} \frac{\partial E}{\partial \dot{\vec{\theta}}} &= \vec{0} \\ J^T(\dot{\vec{x}} - J\dot{\vec{\theta}}) &= \vec{0} \\ J^T\dot{\vec{x}} &= J^T J\dot{\vec{\theta}} \\ \dot{\vec{\theta}} &= \left[J^T J\right]^{-1} J^T\dot{\vec{x}} \\ &= J^{\#}\dot{\vec{x}} \end{aligned}$$

where:

$$\begin{aligned} J^{\#} &= \left[J^T J\right]^{-1} J^T \quad (m < n) \\ &= J^T\left[JJ^T\right]^{-1} \quad (m \geq n). \end{aligned}$$

$J^{\#}$ is the pseudoinverse or Moore-Penrose generalized inverse of J. In the first case ($m < n$), there are fewer independent degrees of freedom in the manipulator than are necessary to execute general Cartesian velocities in the output space. The inverse transformation is overconstrained (more equations than unknown joint velocities) and exact solutions may not exist. In this case, $J^{\#}$ produces the joint velocity vector that minimizes the squared error between the solution and the reference Cartesian trajectory.

In the second case, the vector $\dot{\vec{\theta}} = J^{\#}\dot{\vec{x}}$ defines the minimum length solution, that is the l_2 norm of the resulting $\dot{\vec{\theta}}$ solution is the minimum over all candidate solutions. Since singularities lead to large joint angle velocities, the pseudoinverse exploits manipulator redundancy when possible to avoid kinematic singularity.

But choosing the minimum length joint angle velocity is only one option for selecting from among the alternatives in a redundant manipulator. It is also possible to further exploit redundancy to address scalar performance indices. In doing so, we establish a prioritized behavior for the manipulator. The controller is expressed as:

$$\dot{\theta} = J^{\#}\dot{x} + (I - J^{\#}J)\kappa \tag{3.23}$$

where κ is an *internal* or null space motion that does not disturb the minimum length solution, $\dot{\theta} = J^{\#}\dot{x}$, and optimizes a performance metric:

$$\kappa = K\frac{\partial p}{\partial \theta}. \tag{3.24}$$

The first *priority* task in the control expressed by Equation 3.23 is the Cartesian velocity command. If the redundant manipulator has excess degrees of freedom, then the posture is adjusted to optimize the secondary performance metric, p.

3.7.2 The SR-inverse

Nakamura *et.al.* [9] proposed the SR-inverse as an alternative to the more traditional pseudoinverse. The observation that the pseudoinverse solves for the $\min_{\dot{\theta}} \|\dot{\theta}\|$ among all $\dot{\theta}$ that satisfy $\min_{\dot{\theta}} \|\dot{x} - J\dot{\theta}\|$ guarantees the Cartesian precision (exactness) of the solution, but may still produce infeasible solutions since the joint space velocity may still get quite large in the vicinity of a singularity. If we are willing to compromise the precision of the inverse kinematic solution, then we may also consider the feasibility of the solution. Suppose that the objective function consists of a weighted combination of precision ($\dot{x} - J\dot{\theta}$) and the feasibility of the solution ($\dot{\theta}$). Under these circumstances, the objective function becomes:

$$E = w_1(\dot{x} - J\dot{\theta})^T(\dot{x} - J\dot{\theta}) + w_2(\dot{\theta})^T(\dot{\theta}) \tag{3.25}$$

where w_1 and w_2 are weights representing the relative emphasis on the "exactness" and "feasibility" of the solution, respectively. The result is:

$$J^* = (J^T J + kI)^{-1} J^T = J^T (JJ^T + kI)^{-1} \tag{3.26}$$

where k now represents the relative magnitudes of w_1 and w_2 ($k = 0$ yields the pseudoinverse). It has been reported that this approach can improve performance in the neighborhood of singularities, but as mentioned earlier, introduces some error when tracking a Cartesian trajectory.

3.8 Homework Exercises

1. **Inverting the Homogeneous Transform**
   Given the general expression for the homogeneous transform and its inverse:

$$ {}_A T_B = \left[\begin{array}{ccc|c} & & & \\ \bar{x} & \bar{y} & \bar{z} & \vec{p} \\ & & & \\ \hline 0 & 0 & 0 & 1 \end{array}\right]_{4\times 4} \qquad {}_B T_A = {}_A T_B^{-1} = \left[\begin{array}{ccc|c} & & & \\ \bar{x}^T & & & -\vec{p}\cdot\bar{x} \\ \bar{y}^T & & & -\vec{p}\cdot\bar{y} \\ \bar{z}^T & & & -\vec{p}\cdot\bar{z} \\ & & & \\ \hline 0 & 0 & 0 & 1 \end{array}\right]_{4\times 4} $$

   Prove that ${}_B T_A$ is the inverse of ${}_A T_B$, that is, that ${}_B T_A \; {}_A T_B = I_{4\times 4}$ ($=$ ${}_B T_B$).

2. **Forward Kinematics**
   A 2 DOF planar robot is illustrated in Figure 3.15.

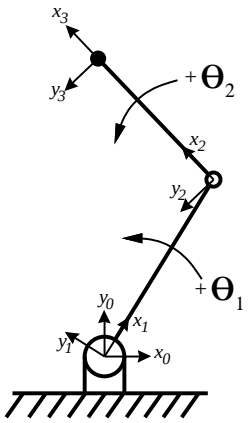


Figure 3.15 *The 2 DOF planar robot*

(a) Derive the forward kinematics of the the manipulator, $\vec{\theta} \mapsto \vec{x}$. Write the homogeneous transform ${}_0T_3$.

(b) Write the Jacobian for the 2 DOF manipulator.

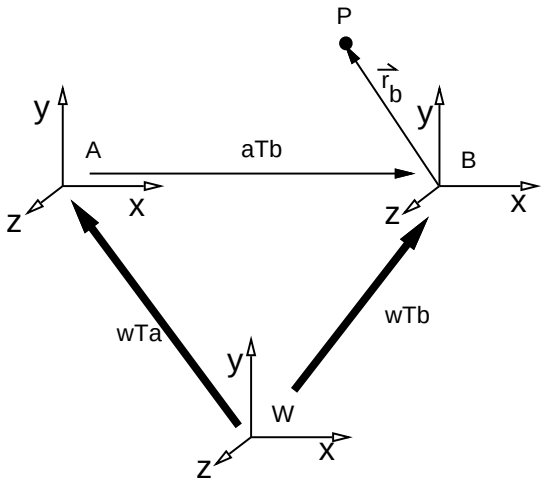
i. Compute the joint angle velocities required to execute an instantaneous velocity of 1 m/s in the x-direction from any initial joint angle configuration, i.e

$$\dot{\vec{\theta}} = f(\vec{\theta}) = J^{-1} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

ii. Compute an expression for the torques on the joints of the manipulator necessary to apply 1 N endpoint force in the $-\hat{y}$ direction from any reachable initial posture.

(c) Compute the eigenvalues and eigenvectors for JJ^T (the squared velocity transformation). Draw the velocity ellipsoid at $\theta_1 = \pi/4,\ \theta_2 = \pi/2$, assuming $l_1 = l_2 = 1$.

3. The Figure illustrates three coordinate frames: W (world), A and B.



$$ {}_W T_A = \begin{bmatrix} 1 & 0 & 0 & -3 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad and \quad {}_W T_B = \begin{bmatrix} 1 & 0 & 0 & 3 \\ 0 & 1 & 0 & 4 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

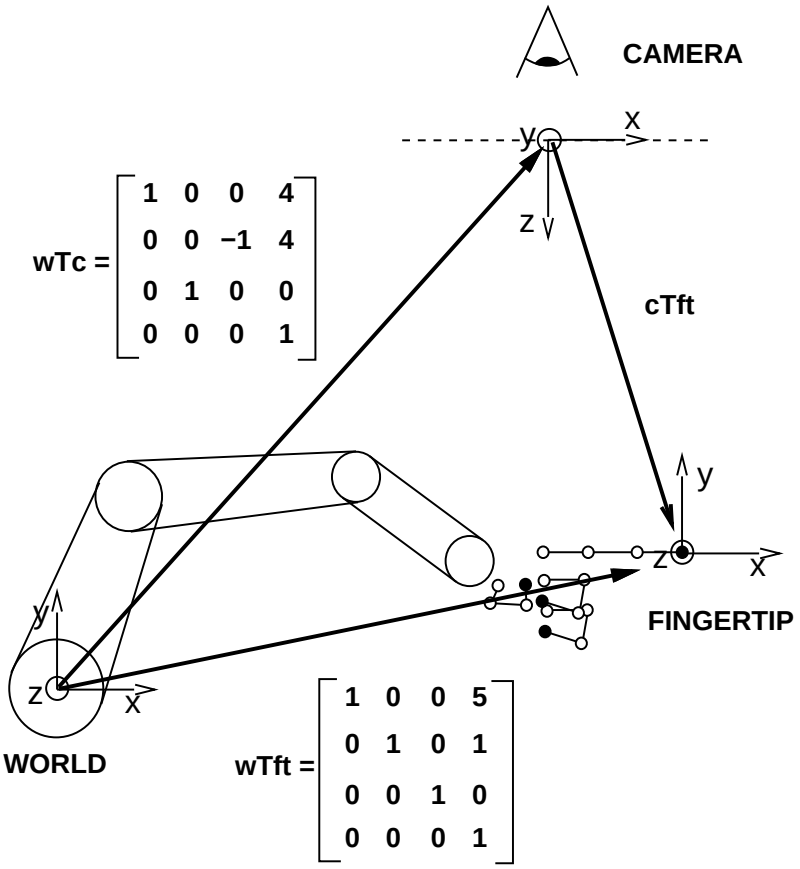
(a) Use the given transforms for ${}_W T_A$ and ${}_W T_B$ to calculate the transform from A to B, ${}_A T_B$.

(b) The position vector which locates point P with respect to coordinate frame B is given by $\vec{r}_B^T = [\ -1,\ 1.5,\ 0,\ 1\ ]$. Solve for the position vector which locates point P with respect to frame A, $\vec{r}_A$.

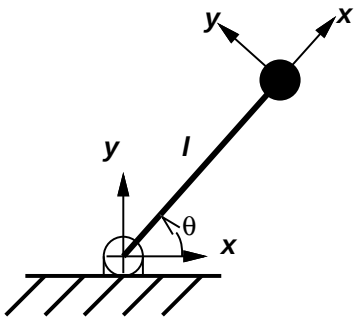
4. Using the transforms given:

(a) solve for ${}_c T_{ft}$

(b) compute the position of the hand's fingertip in the camera's coordinate frame.



5. A simple 1 DOF manipulator is illustrated below.



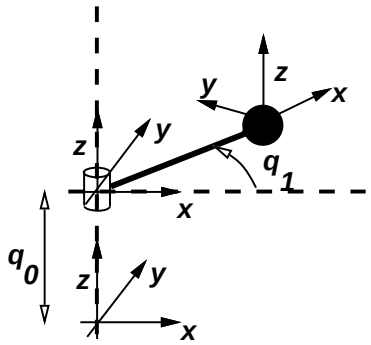
(a) Compute the forward kinematics, i.e., the homogeneous transform which defines the end-point position and orientation, for the manipulator. NOTE: use the defined coordinate systems.

(b) What is the reachable workspace for the manipulator?

(c) Write the Jacobian, J_{2x1}, that maps $\Delta\theta$ to $(\Delta x, \Delta y)$, i.e.,

$$\begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} = \begin{bmatrix} \frac{\delta x}{\delta \theta} \\ \frac{\delta y}{\delta \theta} \end{bmatrix} \Delta\theta \; = \; \mathbf{J}_{2X1} \; [\Delta\theta]$$

6. A simple 2 DOF manipulator is illustrated below.



(a) determine the homogeneous transform which defines the endpoint position and orientation, for the manipulator. NOTE: use the defined coordinate systems.

(b) What is the reachable workspace for the manipulator?

(c) Write the Jacobian, $\Delta\vec{x} = J_{3x2}\Delta\vec{q}$

7. Homogeneous transforms are not the most compact representation for describing spatial relationships, in fact, much of the information in a homogeneous transform is redundant. Given the following homogeneous transform;

$$T \; = \; \begin{bmatrix} ? & 0 & -1 & 0 \\ ? & 0 & 0 & 1 \\ ? & -1 & 0 & 2 \\ ? & 0 & 0 & 1 \end{bmatrix}$$

find the elements designated by the question marks.

8. **Introduction to Roger-the-Crab**

Roger-the-Crab started out as a purely kinematic device proposed by Churchland to study the kinematics of hand-eye coordination. We have a dynamic simulation of Roger in the code for the class and we will use the Roger simulator for several homework projects during the semester. Roger has eyes that independently track a single feature in the field of view. This simple visual system permits us to compute depth from *vergence* — triangulation given known baseline between the eyes and eye orientation. Consequently, we may then use kinematic models of Roger's manipulator to "touch" the visual feature.

Figure 3.16 shows Roger's kinematic structure.

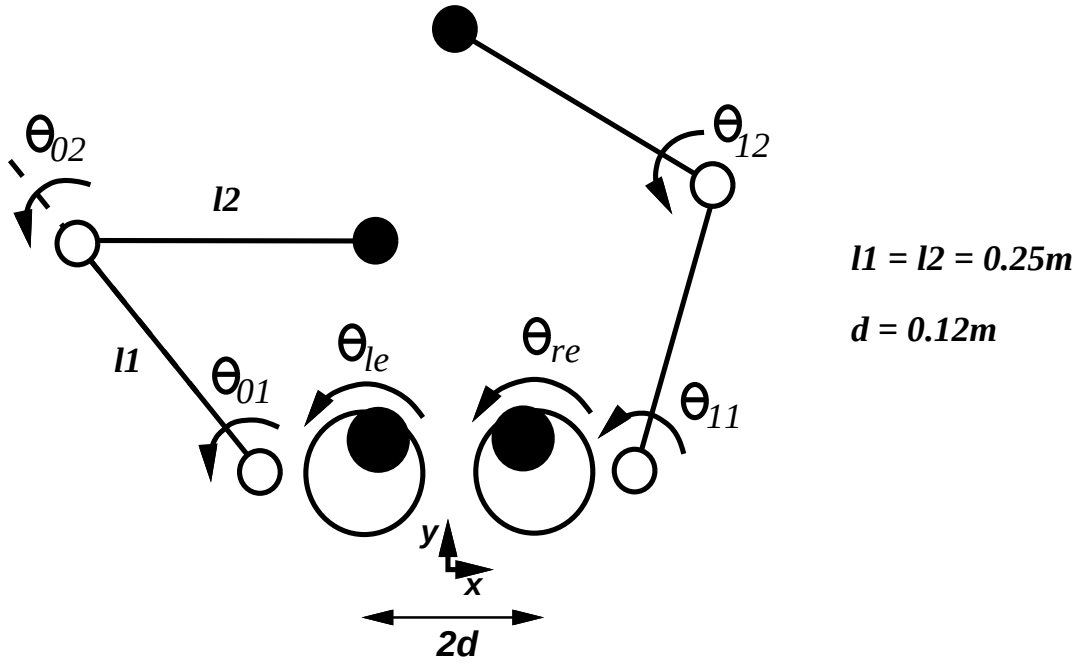


Figure 3.16 *The Anatomy of Roger-the-Crab*

Ultimately, we are interested in the mapping from eye orientations, Φ, to a manipulator configuration, Θ.

Write the function that maps eye orientation to Cartesian position, $\Phi \mapsto X$, analytically. Then, write the inverse kinematic relation, $X \mapsto \Theta$.

9. **Rhino Interface**

The Rhino robot is a five degree-of-freedom educational manipulator with a servoed gripper. The controller has an on-board microprocessor that handles all motor operations. Robot/host communication uses the standard RS-232C interface and can be run from any computer that is equipped with such an interface.

Four instructions provide a comprehensive operational capability:

(a) start a motor and move it a certain number of steps.

(b) stop a motor.

(c) query the status of the six microswitches.

(d) compute the motor position relative to its zero position.

Our Rhinos are hosted by Sun workstation running X Windows. Two sample programs are provided in /usr/local/rhino:

rhino.c: Interactive RS-232 communication between the Sun and the Rhino XR Mark III Controller.

(a) To compile type: *make rhino*

(b) To run type: *rhino*

Anything you type on the Sun will be sent to the Rhino. Typing "~" will exit the program.

rhino_home.c: Skeleton (non-interactive) procedure to drive the Rhino.

(a) To compile type: *make rhino_home*

(b) To run type: *rhino_home*

This project involves building yourself an interface to the Rhino which allows you to:

(a) "home" the robot — move it to a landmark in joint space defined where all the microswitches are closed,

(b) keep track of the *state* of the robot by transforming the encoder values on the robot to joint angles expressed in radians.

(c) define forward- and inverse-kinematic models so that you can map tasks from the world frame to joint space and vise versa.

Bibliography

[1] S.L. Chiu. Control of redundant manipulators for task compatibility. In *Proceedings of the 1987 Conference on Robotics and Automation*, volume 3, pages 1718–1724, Raleigh, NC, April 1987. IEEE.

[2] S.L. Chiu. Kinematic characterization of manipulators: An approach to defining optimality. In *Proceedings of the 1988 Conference on Robotics and Automation*, volume 2, pages 828–833, Philadelphia, PA, April 1988. IEEE.

[3] A. Gelb, editor. *Applied Optimal Estimation.* Technical Staff — The Analytical Sciences Corporation, The MIT Press, Cambridge, MA, 1986.

[4] R. Grupen and K. Souccar. Manipulability-based spatial isotropy: A kinematic reflex. In *Workshop on Mechatronical Computer Systems for Perception and Action*, Halmstad, SWEDEN, June 1-3 1993.

[5] C. Klein and B. Blaho. Dexterity measures for the design and control of kinematically redundant manipulators. *Journal of Robotics Research*, 6(2):72–83, Summer 1987.

[6] Z. Li and S. Sastry. Task-oriented optimal grasping by multifingered robot hands. *IEEE Journal of Robotics and Automation*, 4(1):32–44, February 1988.

[7] Z. Li and S. Sastry. Task-oriented optimal grasping by multifingered robot hands. *IEEE Transactions Systems, Man, and Cybernetics*, 11(10):681–689, 1988.

[8] Sastry Murray, Li. *Robotic Manipoulation: A Mathematical Approach.*

[9] Y. Nakamura and H. Hanafusa. Optimal redundancy control of robot manipulators. *Journal of Robotics Research*, 6(1), Spring 1987.

[10] B. Paul. *Kinematics and Dynamics of Planar Machinery.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1979.

[11] J.K. Salisbury. *Kinematic and Force Analysis of Articulated Hands.* PhD thesis, Stanford University, May 1982.

[12] T. Yoshikawa. Analysis and control of robot manipulators with redundancy. In *Robotics Research: The First International Symposium*, pages 735–747, 1984.

[13] T. Yoshikawa. Manipulability of robotic mechanisms. *Journal of Robotics Research*, 4(2):3–9, Summer 1985.