Christiaan J. J. Paredis
Pradeep K. Khosla
Department of Electrical and Computer Engineering
The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213-3890

Kinematic Design of Serial Link Manipulators From Task Specifications

Abstract

The Reconfigurable Modular Manipulator System (RMMS) consists of modular links and joints that can be assembled into many manipulator configurations. This capability allows the RMMS to be rapidly reconfigured to custom tailor it to specific tasks. An important issue related to the RMMS is the determination of the optimal manipulator configuration for a specific task. This article addresses the problem of mapping kinematic *task specifications into a* kinematic *manipulator configuration. For the design of two-degrees-of-freedom (2-DOF) planar manipulators, an* analytical *solution is derived. Because analytical solutions become impractical for problems with more than two design parameters, we have also developed a numerical approach for the design of 6-DOF manipulators. The numerical procedure determines the Denavit-Hartenberg (D-H) parameters of a nonredundant manipulator with joint limits that can reach a set of specified positions/orientations in an environment that may include parallelepiped-shaped obstacles. Finally, this approach is demonstrated with a three-dimensional example for a 6-DOF manipulator.*

1. Introduction

In theory, robot manipulators are flexible and can be reprogrammed for new tasks. However, each manipulator's configuration makes it capable of only a limited number of applications. A solution to this problem is to have a set of joints of various performance characteristics and a set of links of various lengths that can be assembled into a configuration appropriate for each task. This set of joints and links of various specifications is called the *Reconfigurable Modular Manipulator System* (RMMS) (Schmitz et al. 1988).

To effectively utilize the capabilities provided by the RMMS concept, it is important to address the issue of configuring a manipulator from task requirements.

The International Journal of Robotics Research,
Vol. 12, No. 3, June 1993,

Different tasks require different manipulator configurations (Schmitz et al. 1988); therefore, it is important to establish methodologies that generate an appropriate manipulator configuration, given a set of kinematic and dynamic task specifications. Kinematic requirements are those task requirements that affect only the kinematic structure of the manipulator, while dynamic requirements affect both its kinematics and dynamics. Examples of kinematic requirements are workspace volume, maximum reach, and maximum positional error. Examples of dynamic requirements are maximum payload, maximum joint velocities, and maximum joint accelerations.

Just as the task requirements can be classified as kinematic or dynamic requirements, the design procedure can also be split into two parts: *kinematic* and *dynamic* design (Krishnan and Khosla 1989). Kinematic design determines the kinematic configuration of the manipulator, while dynamic design determines the dynamic configuration. Because the dynamic design possibly requires a change in kinematic structure, a few iterations may be necessary to find a manipulator that satisfies both kinematic and dynamic requirements, as illustrated in Figure 1. Some initial results have already been obtained in mapping dynamic task specifications into the dynamic

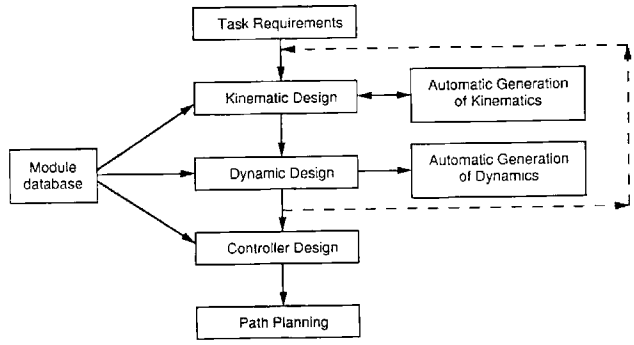

Fig. 1. Schematic of iterative design procedure.

configuration of a manipulator (Krishnan and Khosla 1989).

In the past decade, there has been an increasing interest in the kinematic properties of robot manipulators, such as workspace features (Lee and Yang 1983; Yang and Lee 1983; Gupta 1986; Rastegar and Deravi 1987), dexterity criteria (Uchiyama et al. 1985; Yoshikawa 1985; Chiu 1988; Kim and Khosla 1991), and inverse kinematics of general manipulators (Tsai and Morgan 1985; Raghavan and Roth 1989). Also, the kinematics of modular manipulators have been researched (Kelmar and Khosla 1988; Benhabib et al. 1989). In parallel with the research efforts on kinematic *analysis* of manipulators, the problem of kinematic *design* has been addressed. Tsai and Soni (1984), Vijaykumar et al. (1986), and Paden and Sastry (1988) concluded that an elbow manipulator with zero link offsets is optimal with respect to working volume and dexterity. In Park and Brockett (1989), harmonic maps are used to determine the link lengths that are optimal with respect to a newly defined kinematic dexterity measure. However, the optimality measures considered in the articles mentioned are task independent: they do not guarantee that there is no better manipulator for a *specific* task. Tsai and Soni (1985) include task specificity, assuming, however, that an elbow configuration is optimal for any task; only the size and the base position of the manipulator are considered as design parameters.

Although a major contribution to the design of manipulators was made in these articles, none of them addressed the problem of determining the *total* kinematic structure, including twist angles. This article presents both analytical and numerical approaches for determining the total kinematic structure of a manipulator, given a set of task specifications. Based on the same approach, Au (1992) has been able to design the kinematic structure of *fault-tolerant* manipulators (i.e., a manipulator that is still able to meet the task specifications even if any one of its joints fails and is frozen at any arbitrary angle).

An exact description of the kinematic design problem as addressed in this article, is given in Section 2. Using the concept of the kinematic hypersurface, the design problem is formalized mathematically in Section 3. Section 4 develops a generalization of the inverse kinematics that enables us to formulate the kinematic task specifications as a set of equalities and inequalities. In Section 5, this formulation is translated into an analytical solution for a 2-DOF planar manipulator. To be able to solve design problems with more than two design parameters, a numerical approach is developed in Section 6. In Section 7, this approach is extended to include obstacle avoidance as an additional design criterion. As a comprehensive example, a three-dimensional problem for a 6-DOF manipulator is solved in Section 8. Finally, results are summarized in Section 9.

2. Problem Statement

The exact problem solved in this article is the determination of the kinematic structure of a manipulator that satisfies the given kinematic requirements. Because the kinematic structure of a manipulator can be described unambiguously by its Denavit-Hartenberg (D-H) parameters (Denavit and Hartenberg 1955), we can translate the problem into the determination of the D-H parameters. The kinematic requirements taken into consideration are:

1. *Reachability*. The specified set, W_R, of positions/orientations, $\mathbf{p} = (x, y, z, \vartheta, \varphi, \psi) \in W_R$, has to be inside the reachable workspace of the manipulator.
2. *Joint angles*. The joint angles must remain within the specified limits.
3. *Obstacles*. When reaching the points $\mathbf{p} \in W_R$, the manipulator should not pass through any of the specified parallelepiped-shaped obstacles.

To make the problem solvable, we make the following assumptions:

1. All the design parameters can vary continuously.
2. Only nonredundant serial link rotary manipulators are considered.
3. The robot base position is fixed and known.
4. The last three axes of 6-DOF spatial manipulators intersect at a point.
5. No self-collision is considered.
6. The robot can be represented by a set of straight-line segments with thickness zero.

In a first part of this article, the problem statement is limited to reachability and joint limit specifications. This enables us to clearly explain the basic concepts of the proposed design method. In Section 7, obstacle avoidance is added as a possible task specification, demonstrating the generality of our approach. It has been shown (Paredis 1990) that one can also include manipulability as a task specification and eliminate the assumption of nonredundancy so that the kinematic structure of *redundant* manipulators can be designed. Moreover, the requirement that the robot base be fixed and known can be relaxed, as was shown by Kim (1992), who addressed the problem of kinematic synthesis and base position synthesis simultaneously.

3. General Mathematical Formulation

3.1. Kinematic Space

According to the Denavit-Hartenberg convention, three parameters are needed to describe each link of a rotary manipulator (Craig 1989). These three parameters are the

link length a_{i-1}, the link twist α_{i-1}, and the link offset d_i. A fourth D-H parameter, the joint angle θ_i, does not characterize any dimensional parameter of a rotary serial link manipulator, but merely its position; therefore, it is not a design parameter.

To facilitate the development of our design approach, we define the *D-H configuration space* as a $3n$-dimensional space with the $3n$ D-H parameters as base elements (three D-H parameters for each of the n links). A manipulator with n rotary joints can be represented by a vector, $\mathbf{v}_{dh} = [a_0, \alpha_0, d_1, \ldots, a_{n-1}, \alpha_{n-1}, d_n]^T$, of $3n$ D-H parameters that corresponds to one specific point in the D-H configuration space, and vice versa: each point in the D-H configuration space corresponds to one specific n-DOF rotary manipulator. The vector, $\mathbf{q} = [\theta_1, \ldots, \theta_n]^T$, of n joint variables can be viewed as a point in the n-dimensional *joint space*. Each point in this space represents one specific posture of the manipulator, and conversely, each manipulator posture corresponds to exactly one point in the joint space. A point, $\mathbf{p} = [x, y, z, \vartheta, \varphi, \psi]^T$, in the six-dimensional *task space* (or Cartesian space), corresponds to a physical point and direction in the base frame of the manipulator.

The kinematic relationships of a manipulator involve all three spaces mentioned above. A specific manipulator (an element of the D-H configuration space) in a specific posture (an element of the joint space) positions the end effector in one specific point/orientation (an element of the task space). This relation is usually expressed as a mapping from the joint space into the task space.[1]

To make the kinematic relation[2] easier to use in different forms (e.g., as in inverse kinematics), it is useful to introduce the idea of the *kinematic space*. The $(4n + 6)$–dimensional kinematic space is the Cartesian product of the D-H configuration space, the joint space, and the task space. The base elements of the kinematic space thus include the $3n$ D-H parameters, the n joint variables, and the six position/orientation coordinates of the end effector. This implies that a manipulator in a certain posture at a certain point can be represented as one point in the kinematic space.

Having introduced the idea of kinematic space as a tool to formulate the kinematic relation of serial manipulators, we develop this idea further in the next section and demonstrate its use with an example. Finally, we formulate the kinematic design problem using the framework of the kinematic space.

1. Notice that this mapping depends on the D-H parameters of the manipulator.

2. By *kinematic relation* we mean the relationship between joint positions and end effector position, and not the Jacobian relations between joint velocities and end-effector velocities.

3.2. Mathematical Formulation Using the Concept of the Kinematic Hypersurface

As mentioned before, the kinematic relation can be viewed as a mapping from the joint space into the task space. The disadvantage of this representation is that the mapping is different for each manipulator. The concept of kinematic space can be exploited to overcome this disadvantage, as we can formulate, in this space, all the kinematic relations of all the n-DOF rotary manipulators. Indeed, the kinematics of any n-DOF manipulator can be described by the following equation, which corresponds to a $4n$-dimensional hypersurface (henceforth called *kinematic hypersurface*) in the kinematic space:

$$ {}^0_1T\,{}^1_2T\,{}^2_3T \cdots {}^{n-1}_{n}T = {}^0_nT, \tag{1} $$

where ${}^{i-1}_{i}T$ is the transformation matrix from the $(i-1)$th frame to the ith frame and is described by:

$$ {}^{i-1}_{i}\mathrm{T} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i\cos\alpha_{i-1} & \cos\theta_i\cos\alpha_{i-1} & -\sin\alpha_{i-1} & -\sin\alpha_{i-1}d_i \\ \sin\theta_i\sin\alpha_{i-1} & \cos\theta_i\sin\alpha_{i-1} & \cos\alpha_{i-1} & \cos\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2} $$

In equation (1), ${}^0_nT$ is the position/orientation of the end effector of the manipulator:

$$ {}^0_nT = \begin{bmatrix} & & & x \\ & R(\vartheta,\varphi,\psi) & & y \\ & & & z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{3} $$

where $[x, y, z]^T$ is the Cartesian position vector of the end effector with respect to the base frame of the manipulator, and $R(\vartheta, \varphi, \psi)$ is the orientation matrix of the end effector with respect to the base frame:[3]

$$ \mathrm{R}(\vartheta,\varphi,\psi) = \begin{bmatrix} c\vartheta c\varphi & c\vartheta s\varphi s\psi - s\vartheta c\psi & c\vartheta s\varphi c\psi + s\vartheta s\psi \\ s\vartheta c\varphi & s\vartheta s\varphi s\psi + c\vartheta c\psi & s\vartheta s\varphi c\psi - c\vartheta s\psi \\ -s\varphi & c\varphi s\psi & c\varphi c\psi \end{bmatrix}. \tag{4} $$

An example of the use of the kinematic hypersurface is the determination of the workspace of a specific manipulator. Specifying the $3n$ D-H parameters, α_{i-1}, a_{i-1}, d_i $(i = 1, \ldots, n)$ corresponds to intersecting the $4n$-dimensional kinematic hypersurface with $3n$ of $(4n + 5)$–dimensional hyperplanes. This yields an n-dimensional solution; the dimensionality of the solution corresponds to the n joint angles that are still free to be chosen. To determine the reachable workspace, we can

3. We use the X-Y-Z fixed-angle convention, also referred to as *roll-pitch-yaw convention*.

project this n-dimensional hypersurface on the (x, y, z) space.

Instead of obtaining the workspace, given the kinematic parameters of a manipulator, we are interested in exactly the opposite problem: given a required workspace, W_R, determine the kinematic configuration. This problem can be formulated mathematically using the concept of the kinematic hypersurface. We achieve this in two steps. First, consider the set of all the kinematic configurations that can reach *one* specific point/orientation, $\mathbf{p}_j = [x_j, y_j, z_j, \vartheta_j, \varphi_j, \psi_j]^T$. This set is the intersection of the kinematic hypersurface with six $(4n + 5)$–dimensional hyperplanes, given by the following equations:

$$\begin{array}{ll} x = x_j & \vartheta = \vartheta_j \\ y = y_j & \varphi = \varphi_j \\ z = z_j & \psi = \psi_j \end{array} \tag{5}$$

The solution of this set of equations will, in general, be an infinite set of manipulators. We can find such a set of manipulators for each point in W_R.

In the second step, we take the intersection of all the solutions—each for a different position $\mathbf{p}_j$—found in step one. This yields the set of manipulators able to reach every point in W_R. An exact mathematical formulation of the problem is thus:

> Find $\mathbf{v}_{dh} = [a_0, \alpha_0, d_1, \ldots, a_{n-1}, \alpha_{n-1}, d_n]^T$ such that for each $\mathbf{p}_j \in W_R : \exists \theta_i (i = 1, n)$, for which
>
> $$ {}^0_1T\,{}^1_2T\,{}^2_3T \cdots {}^{n-1}_{n}T = \begin{bmatrix} & & & x_j \\ & R(\vartheta_j, \varphi_j, \psi_j) & & y_j \\ & & & z_j \\ \mathbf{0} & \mathbf{0} & 0 & 1 \end{bmatrix} \tag{6}$$
>
> and
>
> $$\theta\mathrm{min}_i \le \theta_i \le \theta\mathrm{max}_i \qquad (i = 1, n) \tag{7}$$
>
> where $\theta\mathrm{min}_i$ and $\theta\mathrm{max}_i$ are the joint limits of joint i.

4. Generalized Inverse Kinematics

4.1. Mathematical Formulation

Section 3.2 translated the problem of kinematic design of serial manipulators into a general mathematical formulation. This formulation is particularly useful because of its generality, but it does not lend itself to an easy solution procedure. Therefore, in this section, we will transform the general formulation such that it becomes solvable. To achieve our goal, we will exploit the availability of the closed-form inverse kinematics solution and assume that the manipulator is nonredundant.

We can rewrite equation (6) in the form of the inverse kinematics:

$$\mathbf{q} = \mathbf{g}(\mathbf{p}, \mathbf{v}_{dh}) \tag{8}$$

where $\mathbf{q} = [\theta_1, \ldots, \theta_n]^T$ is the vector of the joint variables and $\mathbf{g}(\cdot)$ is a vector function describing the mapping from task space to joint space.

This is a set of highly nonlinear equations that, in general, has a nonunique solution. A first attempt to solve this set of equations was made by Pieper and Roth (1969). They found a closed-form solution in the form of a fourth-order polynomial, when three consecutive axes of a manipulator intersect at a point. Only recently has a closed-form solution been derived for a *general* 6R manipulator (Tsai and Morgan 1985; Lee and Liang 1988; Raghavan and Roth 1989). This solution appears as a 16th-order polynomial in the tangent of the half-angle of one of the joint variables.

Using the inverse kinematics, equation (8), we now restate the mathematical formulation of the previous section as follows:

> Find $\mathbf{v}_{dh}$ such that for each $\mathbf{p}_j \in W_R : \exists \theta_i (i = 1, n)$, for which the following constraints are satisfied:
>
> $$\theta_i = g_i(\mathbf{p}_j, \mathbf{v}_{dh}) \qquad (i = 1, n) \tag{9}$$
>
> $$\theta\mathrm{min}_i \le \theta_i \le \theta\mathrm{max}_i \qquad (i = 1, n) \tag{10}$$
>
> where $\theta\mathrm{min}_i$ and $\theta\mathrm{max}_i$ are the joint limits of joint i.

However, one major problem exists for this formulation. In the event that the point $\mathbf{p}_j$ is outside the reach of the manipulator, equation (9) has no solution. To cope with this problem, we use a computation scheme that we call *generalized inverse kinematics*, which, in the case of nonreachable points, generates *complex* joint angles. In the next section, we present this approach in more detail.

4.2. Generalized Inverse Kinematics

The problem addressed in this section is the following: How can we adapt the inverse kinematics such that they yield a solution, even when the specified position/orientation is outside the reach of the manipulator? To achieve our objective, we use the following substitution in the inverse kinematics solution (Pieper and Roth 1969; Tsai and Morgan 1985; Raghavan and Roth 1989):

$$u = \tan\left(\frac{\theta}{2}\right). \tag{11}$$

This substitution transforms a trigonometric equation in $\cos(\theta)$ and $\sin(\theta)$ into a polynomial equation in u,

which then can be solved using a root-finding algorithm. Furthermore, this substitution is a one-to-one mapping between an angle $\theta \in [-\pi, \pi]$ and a real number $u \in [-\infty, \infty]$; this is important for mapping u back into θ.

Using the transformation (11), we can express the inverse kinematics as a set of polynomial equations. The difference between the normal inverse kinematics and the generalized inverse kinematics (GIK) is that in the GIK all the roots of the polynomial equations are considered, including complex roots, whereas in the normal inverse kinematics, only real roots are considered. Thus, using the GIK, the joint angles θ_i are, in general, complex quantities. A complex joint angle does not make physical sense and thus should be excluded from the admissible solution set. This is achieved by adding the constraint

$$\mathrm{Imag}(\theta_i) = 0 \tag{12}$$

in the above problem statement.

The inclusion of complex roots implies that a solution for the inverse kinematics will always exist (even if the desired position of the end effector is outside the reach of the manipulator) and that the number of solutions is invariable—16 solutions for a general 6R serial manipulator (Raghavan and Roth 1989)—which results in a simplified implementation of the GIK.

The final problem statement becomes:

> Find $\mathbf{v}_{dh}$ such that for each $\mathbf{p}_j \in W_R : \exists \theta_i (i = 1, n)$, for which the following constraints are satisfied:
>
> $$\begin{cases} \theta_i = g_i^G(\mathbf{p}_j, \mathbf{v}_{dh}) \\ \mathrm{Imag}(\theta_i) = 0 \\ \theta\mathrm{min}_i \le \theta_i \le \theta\mathrm{max}_i \end{cases} \quad (i = 1, n) \tag{13}$$
>
> where $g_i^G (i = 1, n)$ is a set of complex functions representing the GIK.

5. Analytical Solution for a 2-DOF Planar Manipulator

5.1. Example

We consider the problem of designing a 2-DOF planar manipulator having only two design (or D-H) parameters: the link lengths l_1 and l_2. Figure 2 shows these parameters and the other variables used in this section.

The exact design problem addressed is: Given a finite set of points $(x, y) \in W_R$ and limits on the joint angles θ_1 and θ_2, find a pair of link lengths (l_1, l_2) that determine a planar manipulator that is able to reach all the points in W_R. More specifically, the joint limits are $\pm\pi/4$ for θ_1 and $\pm 5\pi/6$ for θ_2. The points $(x, y) \in W_R$ are depicted by an "X" in Figure 3.

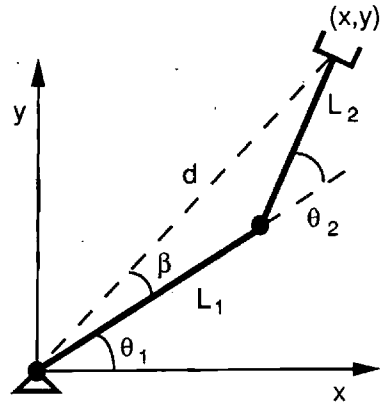

Fig. 2. A simple 2-DOF example.

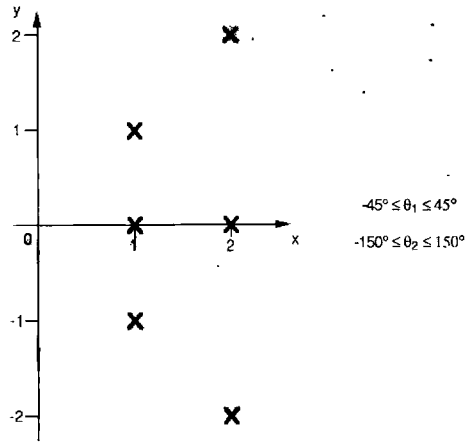

Fig. 3. Graphical depiction of the problem statement.

5.2. The Inverse Kinematics

The inverse kinematics of this planar manipulator can be expressed using inverse trigonometric functions:

$$\theta_1 = \arctan 2(y, x) - \arccos\left(\frac{x^2 + y^2 + l_1^2 - l_2^2}{2 l_1 \sqrt{x^2 + y^2}}\right) \tag{14}$$

$$\theta_2 = \arccos\left(\frac{x^2 + y^2 - l_1^2 - l_2^2}{2 l_1 l_2}\right) \tag{15}$$

This formulation has two disadvantages. First, arccos() is not defined for values of its argument outside the interval $[-1, +1]$. Second, we find only one of the two postures to reach (x, y) (elbow down) using this formula. Consequently, it is more convenient to use the GIK by making the substitution (11). We rewrite (14) and (15) as a function of u and v, where:

$$v = \tan\left(\frac{\theta_1}{2}\right), \tag{16}$$

$$u = \tan\left(\frac{\theta_2}{2}\right). \tag{17}$$

From equation (15) we obtain an expression for u:

$$u^2 = -\frac{(x^2+y^2)-(l_1+l_2)^2}{(x^2+y^2)-(l_1-l_2)^2}. \tag{18}$$

Once the solutions for u are known, it is possible to find an expression for v starting with the following equations:

$$\begin{cases} x &= k_1\cos\theta_1 - k_2\sin\theta_1 \\ y &= k_1\sin\theta_1 + k_2\cos\theta_1 \\ x^2+y^2 &= k_1^2+k_2^2 \end{cases} \tag{19}$$

where

$$k_1 = l_1 + l_2\cos\theta_2, \tag{20}$$

$$k_2 = l_2\sin\theta_2. \tag{21}$$

The solution for this set of equations is

$$v = \frac{y-k_2}{x+k_1} = \frac{k_1-x}{k_2+y}. \tag{22}$$

5.3. Mathematical Formulation of the Task Specifications

The task specifications considered in this example are *reachability* and *joint limits*.

The GIK, as given by equations (18) and (22), will, in general, yield complex solutions for u. Indeed, u^2 is negative when

$$\sqrt{x^2+y^2} < |l_1 - l_2| \tag{23}$$

or

$$\sqrt{x^2+y^2} > l_1 + l_2, \tag{24}$$

which corresponds to the cases for which the point (x, y) is outside the reach of the manipulator. The reachability constraint can be expressed as

$$\mathrm{Imag}(u) = 0. \tag{25}$$

If we compute k_1 and k_2 using

$$\theta_2 = 2\arctan(\mathrm{Real}(u)), \tag{26}$$

it is not necessary to demand that

$$\mathrm{Imag}(v) = 0, \tag{27}$$

since

$$k_1, k_2 \in \Re \Rightarrow v \in \Re. \tag{28}$$

The mathematical formulation of the whole problem is then:

Find (l_1, l_2) such that for each point $(x, y) \in W_R : \exists(\theta_1, \theta_2)$ and (v, u) for which:

$$\begin{cases} \mathrm{Imag}(u) = 0 \\ \theta_1 \geq \theta\mathrm{min}_1 \\ \theta_2 \geq \theta\mathrm{min}_2 \\ \theta_1 \leq \theta\mathrm{max}_1 \\ \theta_2 \leq \theta\mathrm{max}_2 \end{cases} \tag{29}$$

5.4. Analytical Solution

In this section, we develop expressions for the bounds on the feasible region in the $\{l_1, l_2\}$ plane (i.e., the region containing the admissible solution sets (l_1, l_2)). We derive analytical expressions for the following:

1. Bounds due to reachability constraints
2. Bounds due to joint limits on θ_1
3. Bounds due to joint limits on θ_2

The reachability bounds border the region, determined analytically by equations (23) and (24) or equation (25). These equations define the admissible solution pairs (l_1, l_2), that describe a manipulator *without* joint limits that is able to reach the point (x, y). The region is bounded by three straight lines:

$$l_1 + l_2 = d \tag{30}$$

$$l_1 - l_2 = d \tag{31}$$

$$l_2 - l_1 = d \tag{32}$$

where $d = \sqrt{x^2+y^2}$. For the point $(x = 1, y = 0)$, the reachability bounds are labeled "a" in Figure 4.

To find an expression for the bounds due to the joint limits on θ_1, we will first derive the equation, describing the manipulators that can reach the point (x, y) with a fixed joint angle θ_1. If we define the angle β as in Figure 2, the following expression holds:

$$(d\sin\beta)^2 + (d\cos\beta - l_1)^2 = l_2^2, \tag{33}$$

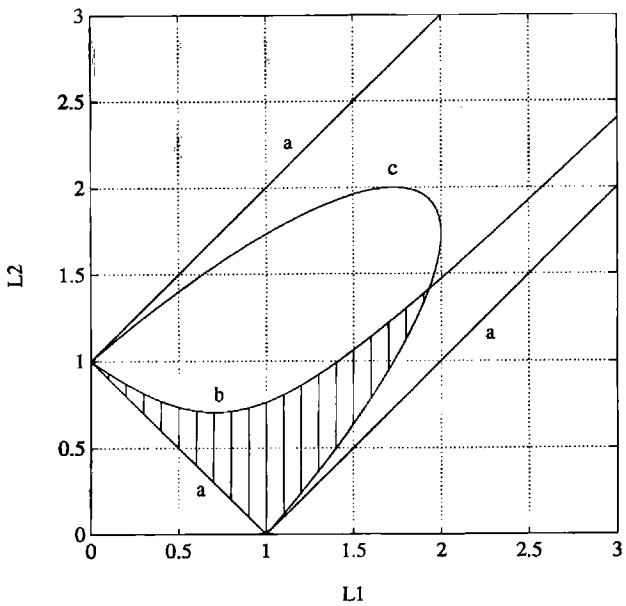

Fig. 4. Bounds of the feasible region for the point ($x = 1$, $y = 0$). a, Bounds due to reachability constraints. b, Bounds due to joint limits on θ_1, $|\theta_1| \leq \pi/4$. c, Bounds due to joint limits on θ_2, $|\theta_2| \leq 5\pi/6$.

or

$$\left(\frac{l_2}{d\sin\beta}\right)^2 - \left(\frac{d\cos\beta - l_1}{d\sin\beta}\right)^2 = 1. \qquad (34)$$

Equation (34) represents a hyperbola with both asymptotes at 45 degrees, focal axis parallel to the l_2 axis, center at $(d\cos\beta, 0)$, and a vertex distance of $d\sin\beta$. The bound due to joint limits on θ_1 can now be found by choosing the maximum permissible β-value. This hyperbola is labeled "b" in Figure 4.

The starting point for the deduction of the equation representing the bounds due to joint limits on θ_2 is equation (18). If we perform a rotation of 45 degrees, transforming the $\{l_1, l_2\}$ D-H configuration space into the space with base variables $\{p = \frac{l_1+l_2}{\sqrt{2}}, q = \frac{l_1-l_2}{\sqrt{2}}\}$, we obtain:

$$u^2 = \frac{\sin^2(\frac{\theta_2}{2})}{\cos^2(\frac{\theta_2}{2})} = -\frac{d^2 - 2p^2}{d^2 - 2q^2}. \qquad (35)$$

The above expression can be written as:

$$2p^2\cos^2\left(\frac{\theta_2}{2}\right) + 2q^2\sin^2\left(\frac{\theta_2}{2}\right) = d^2\cos^2\left(\frac{\theta_2}{2}\right) + d^2\sin^2\left(\frac{\theta_2}{2}\right) = d^2, \qquad (36)$$

or:

$$\left(\frac{p}{\frac{d}{\sqrt{2}\cos(\frac{\theta_2}{2})}}\right)^2 + \left(\frac{q}{\frac{d}{\sqrt{2}\sin(\frac{\theta_2}{2})}}\right)^2 = 1. \qquad (37)$$

This is the equation of an ellipse with an axis of $d/(\sqrt{2}\cos(\theta_2/2))$ in the p direction and an axis of $d/(\sqrt{2}\sin(\theta_2/2))$ in the q direction. The bound, labeled "c" in Figure 4, corresponds to the maximum permissible value of $|\theta_2|$.

For each point (x, y), we thus obtain five bounding curves. However, one can see in Figure 4 that the bound due to joint limits on θ_2 will always be more restrictive than the bounds given by equations (31) and (32). The bounds delimiting the feasible region are called "active" (e.g., bound "c"), while "passive" bounds do not touch the feasible region at all (i.e., a more restrictive active bound exists). For problems with several points in W_R, the number of bounds can become very large, but the set of active bounds will usually remain small.

The analytical solution for the example considered in this section is shown in Figure 5. One can pick any pair (l_1, l_2) inside the feasible region as a solution to the design problem. However, in general, for spatial manipulators, the D-H configuration space has $3n$ dimensions, and the analytical bounds are highly nonlinear hypersurfaces. In this case, it becomes very difficult to construct the set of active bounds and even more difficult to find a tuple that is "inside" all the active bounds, as we cannot visualize the solution anymore. A numerical solution procedure is more appropriate here.

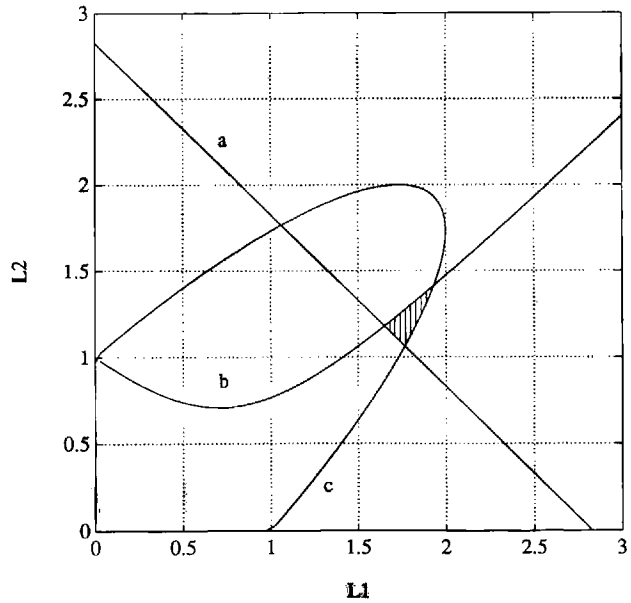

Fig. 5. Analytical solution for example 1. The bounds are: a, Bound due to reachability constraints. b, Bound due to joint limits on θ_1, $|\theta_1| \leq \pi/4$. c, Bound due to joint limits on θ_2, $|\theta_2| \leq 5\pi/6$.

In the next section, we develop a numerical approach that can be used for the design of spatial manipulators. We show its efficacy by applying it to the 2-DOF design problem and comparing the results with the analytical approach.

6. Numerical Solution

6.1. Methods to Solve a Set of Nonlinear Equations

Section 4.2 formulated the design problem in terms of the GIK. At first sight, the constraints (13) look like a simple set of nonlinear equations to which standard solution methods are applicable. However, as we mentioned earlier, the GIK have multiple solutions (eight for a 6R serial manipulator that has the last three axes intersecting at a point). Each solution corresponds to a possible posture of the manipulator. Thus, for each point $\mathbf{p} \in W_R$ and a given set of D-H parameters, the GIK yield eight sets of joint variables: $(\theta_1^{(1)}, \ldots, \theta_6^{(1)}), (\theta_1^{(2)}, \ldots, \theta_6^{(2)}), \ldots (\theta_1^{(8)}, \ldots, \theta_6^{(8)})$, where the superscript denotes the posture number. For each posture, a separate set of (in)equalities, equation (13), representing the task requirements is formulated. The manipulator fulfills all the task requirements if the set of (in)equalities is satisfied for at least one set $(\theta_1^{(k)}, \ldots, \theta_6^{(k)})$. For instance, the reachability task specification is satisfied

when a point, $\mathbf{p}_j$, is reachable in at least one posture but does not require that $\mathbf{p}_j$ has to be reachable in all eight postures possible. This implies that the problem is not formulated as a set of (in)equalities that have to be satisfied simultaneously, but merely as a juxtaposition of subsets of simultaneous (in)equalities for which only at least one subset has to be satisfied.

The algorithms found in the literature to solve a set of nonlinear equalities and inequalities cannot handle the above formulation. However, there is one approach that can be adapted to this more complex formulation: the penalty function method (Fletcher 1987).

6.2. *Formulating the Problem of Kinematic Design Using a Penalty Function*

The basic idea of the penalty function approach is to translate the problem of finding a feasible solution for a set of constraints into a minimization problem. For instance, consider the following set of constraints:

$$\begin{cases} c_i(\mathbf{x}) = 0 & (i = 1, l) \\ c_j(\mathbf{x}) \leq 0 & (j = 1, m) \end{cases} \tag{38}$$

This set of constraints is transformed into the penalty function:

$$F(\mathbf{x}) = \sum_{i=1}^{l} [c_i(\mathbf{x})]^2 + \sum_{j=1}^{m} \left[\max\left(0, c_j(\mathbf{x})\right)\right]^2 \tag{39}$$

with the property that all the constraints are satisfied simultaneously if and only if the function $F(\mathbf{x})$ is equal to zero. Otherwise, $F(\mathbf{x})$ has a strictly positive value.

For the constraints in equation (13), a possible penalty function is:

$$\begin{aligned} F(\mathbf{q}^{(k)}) = \textstyle\sum_{i=1}^{n} \; \{ & \; [\mathrm{Imag}(\theta_i^{(k)})]^2 \\ & + [\max(0, \theta\mathrm{min}_i - \mathrm{Real}(\theta_i^{(k)}))]^2 \\ & + [\max(0, \mathrm{Real}(\theta_i^{(k)}) - \theta\mathrm{max}_i)]^2 \\ & \} \end{aligned} \tag{40}$$

where $\mathbf{q}^{(k)} = [\theta_1^{(k)}, \theta_2^{(k)}, \ldots, \theta_8^{(k)}]^T$, the joint vector of posture j.

Because the constraints have to be satisfied for only one of the eight postures, we can evaluate $F(\mathbf{q}^{(k)})$ for each posture, j, and consider the minimum of these eight penalties:

$$F_{\mathrm{point}\,p}(\mathbf{v}_{dh}) = \min\left\{F(\mathbf{q}^{(1)}), F(\mathbf{q}^{(2)}), \ldots, F(\mathbf{q}^{(8)})\right\} \tag{41}$$

where $\mathbf{v}_{dh}$ is the vector of D-H parameters. Indeed, this function is zero when at least one of the postures has a zero penalty and thus satisfies all the constraints. The total penalty for a certain manipulator is obtained by summing over all the points $\mathbf{p}_j \in W_R$:

$$F_{\mathrm{tot}}(\mathbf{v}_{dh}) = \sum_{p \in W_R} \min\left\{F(\mathbf{q}^{(1)}), F(\mathbf{q}^{(2)}), \ldots, F(\mathbf{q}^{(8)})\right\}. \tag{42}$$

Experiments with this penalty function showed the existence of a large number of local minima. Therefore, to find a solution that satisfies all the constraints, we need to use a global optimization procedure.

6.3. *Methods to Solve a Global Optimization Problem*

In recent years, a multitude of solution methods for global optimization problems have been developed (Torn and Žilinskas 1989). Each method has its own characteristics, and an appropriate choice of a specific method can only be made by matching both the characteristics of the method and the problem. The problem we wish to solve here has a constrained search space and a fairly high dimensionality. The objective function (i.e., the penalty function) has no analytical expressions for the derivatives. It has many local minima and regions of global minima where the function value equals zero. Furthermore, the function is computationally expensive to evaluate. Based on these characteristics, we choose to implement two random search algorithms: simulated annealing (Kirkpatrick et al. 1983; van Laarhoven and Aerts 1987) and random line search (Bremermann 1970).

Simulated annealing was first proposed as a global optimization algorithm for combinatorial problems (Kirkpatrick et al. 1983), but it can be easily adapted to continuously varying optimization variables. The method is based on an algorithm from statistical physics developed by Metropolis et al. (1953). The method is basically a random iterative improvement algorithm with the modification that, under certain conditions, an *increase* in the objective function is accepted. A basic flow chart of the simulated annealing algorithm is shown in Figure 6. A new trial point is generated randomly in the neighborhood of the current guess. The condition for acceptance of this trial point is:

$$\begin{cases} \Delta F_{\mathrm{obj}} \leq 0 & \Longrightarrow \text{accept} \\ \exp(-\Delta F_{\mathrm{obj}}/T) > \text{random } [0, 1) & \Longrightarrow \text{accept,} \end{cases} \tag{43}$$

which depends on a control variable, T, the temperature. The algorithm is started at a high temperature, for which most new trial configurations are accepted. After each iteration, the temperature is decreased until no new acceptable trial point can be found; the optimization is then *frozen*. We adapted this basic algorithm to the special properties of our objective function. The algorithm is stopped when a new trial point has an objective function value equal to zero, even when the optimization is not yet frozen. Also, the best-guess-so-far is stored, because it is

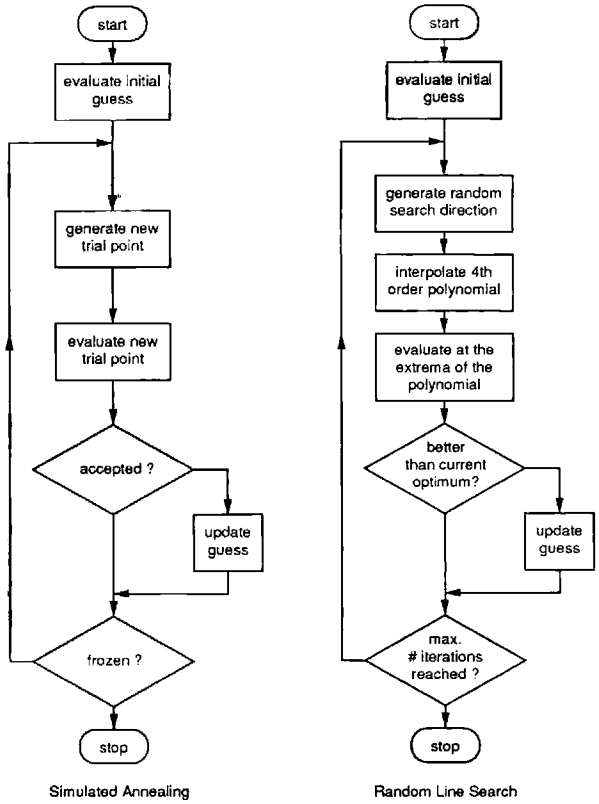

Fig. 6. Flow charts of the simulated annealing algorithm and the line search algorithm.

possible in simulated annealing to diverge from this best guess.

The random line search algorithm was developed by Bremermann (1970) as an improvement of pure random sampling. The motivation is that the probability of a line intersecting a neighborhood, B_0, of the minimum is proportional to the relative surface of B_0, which may be essentially larger than the relative volume of B_0 (Piefke 1978). A flow chart of the algorithm is given in Figure 6. In our implementation, we added the extra stopping rule for zero objective function values.

The optimization procedures discussed above are compared through a simple 2-DOF example in the next section.

6.4. Numerical Solution

In Section 5, it was illustrated that analytical bounds for the feasible region can be found when the number of design parameters is small. Here, we continue the 2-DOF example of Section 5 and compare the analytical solution with the numerical solution.

Equation (29) expresses the task requirements as a set of equalities and inequalities. As in Section 6.2, the problem of solving this set of (in)equalities is converted

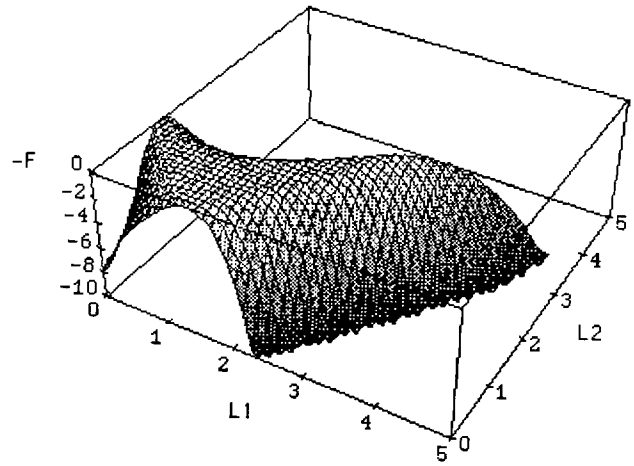

Fig. 7. Penalty function when only one point is specified.

into an optimization problem with the following penalty function:

$$
\begin{aligned}
F(l_1, l_2) = \sum_{(x,y)\in W_R} \min_{j=1,2} \; \{ & \; [\mathrm{Imag}(u^{(k)})]^2 \\
& + [\max(0, \theta min_1 - \theta_1^{(k)})]^2 \\
& + [\max(0, \theta_1^{(k)} - \theta max_1)]^2 \\
& + [\max(0, \theta min_2 - \theta_2^{(k)})]^2 \\
& + [\max(0, \theta_2^{(k)} - \theta max_2)]^2 \\
& \}
\end{aligned}
\tag{44}
$$

Figure 7 shows the penalty function when only one point is specified.

When several points are specified, the penalty function becomes more complex in shape and has local minima. Therefore, a global optimization procedure is needed. For this example, three optimization methods were used: simulated annealing, random line search, and crude random sampling. Although the convergence properties of all three methods are different, their final solutions are very similar.

The results for 100 runs of the simulated annealing algorithm are shown in Figure 8. Each "·" represents a final solution of the optimization procedure. The initial guesses for the 100 runs are generated using a random number generator and are uniformly distributed over the region $\{(l_1, l_2) | 0 \leq l_1 \leq 3, 0 \leq l_2 \leq 3\}$. The solutions of the optimization procedure are all inside the feasible region bordered by the analytical bounds found in Section 5.4. This implies that the approach presented in this section performs well for this simple example. In Section 8, a more complex example is presented to prove the validity and generality of the approach.

We tested the convergence of the simulated annealing and random line search algorithms and compared them with crude random sampling. The results are shown in Table 1. The random line search produces the best results

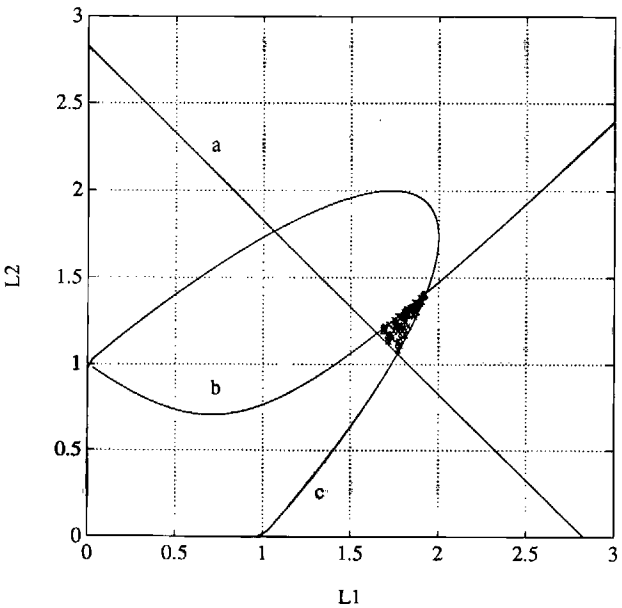

Fig. 8. Numerical solutions (indicated with "·"). Analytical bounds: a, Bound due to reachability constraints. b, Bound due to joint limits on θ_1, $|\theta_1| \leq \pi/4$. c, Bound due to joint limits on θ_2, $|\theta_2| \leq 5\pi/6$.

Table 1. Comparison between random sampling, simulated annealing and random line search, expressed in number of function evaluations*

	Number of Function Evaluations		
	Random Sampling	*Simulated Annealing*	*Line Search*
Mean	236	204	104
Std. dev.	204	172	67
Minimum	6	6	8
Maximum	1034	761	313

*The statistical population consists of 100 runs.

for this simple example.[4] However, our experience is that for more complex design problems, the random line search algorithm does not always converge. This can be explained by the fact that only transitions to *better* guesses are allowed. Therefore, simulated annealing is preferred for these problems.

Thus far, we have developed a procedure for designing a manipulator with joint limits that is able to reach a specified set of positions/orientations. Before we solve a three-dimensional example for a 6-DOF manipulator, we will demonstrate the generality of the penalty function approach by adding obstacles as a possible task specification.

7. Including Obstacles in the Workspace

All the tasks performed by robots are performed in an environment. To make sure that a certain task is performable by a manipulator, we have to take this environment into consideration. For instance, if we want to pick up an object from a table, we have to make sure that we can reach this object without passing *through* the table. Usually, the obstacle avoidance problem is considered as a planning problem. Here, we will approach it in the context of design. Thus, we will not answer the question, "Which path do we have to follow to reach this object?" but merely the question, "Which manipulator can be used to accomplish the task in the presence of obstacles?" To answer this question, we need an adequate model of both the environment and of the manipulator. We will use simple but reasonable models. Specifically, we limit ourselves to parallelepiped-shaped obstacles. This limitation is not as restrictive as might be thought, because we allow the definition of multiple obstacles so that oddly shaped obstacles can always be approximated by a combination of several parallelepipeds. Furthermore, we assume that the manipulator can be represented by a set of straight-line segments with thickness zero. Again, we can get around this limitation by increasing the sizes of the obstacles, similar to "growing" the obstacles (Lozano-Pérez and Wesley 1979).

In the previous section, we described a technique to translate the manipulator design problem into an optimization problem by using a penalty function. Now, we will demonstrate how we use this same concept to include *obstacle avoidance* in the design.

The key for the penalty function approach is that the penalty has to be in some way proportional to the violation of the constraint and that it must be zero when the constraint is satisfied. The constraint in the case of obstacle avoidance is that the manipulator should not pass through the obstacles. Thus, the penalty function should be (1) zero if the manipulator does not intersect with any obstacle, and (2) "proportional" to the largeness of intersections. The first criterion is well defined and easy to achieve, but the second criterion is rather vague. The way in which "proportionality" is defined affects both the computation time of the penalty function and the convergence properties of the optimization. This can be illustrated by a simple two-dimensional example with only one obstacle.[5]

4. A possible explanation for this result is given by Žilinskas (1986).

5. In case there is more than one obstacle, each obstacle can be considered independently and the resulting penalties added together.

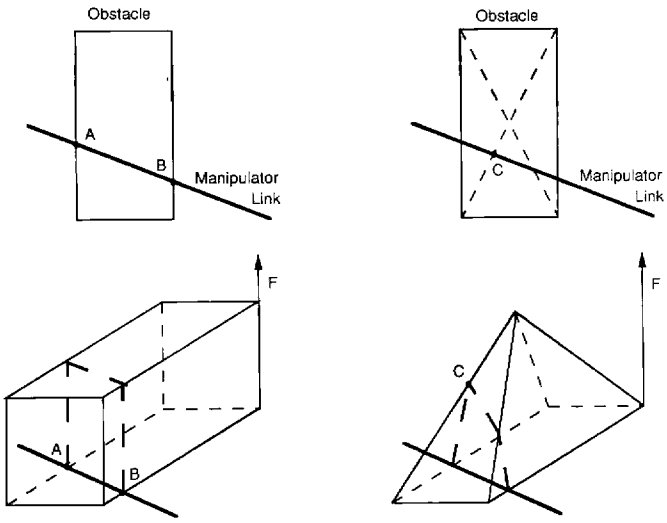

Fig. 9. Different types of penalties for obstacles.

In two dimensions, a parallelepiped reduces to a rectangle. We will compare the following two penalties, depicted in Figure 9:

F_1 is the length of intersection (i.e., the distance between the points A and B). This corresponds to the *integral* over the length of manipulator of a function that has the value of unity inside the obstacle and zero outside.

F_2 is the *maximum* value, achieved over the length of the manipulator, of a pyramidal function (point C).

Notice that both penalties have the property that they are zero when the manipulator does not intersect the obstacle. Both are also proportional to the largeness of intersection but in a different way.

To solve the design problem, we minimize the penalties. Minimizing F_1 results in a reduction of the length of intersection. This causes the line AB to become perpendicular to the sides of the obstacle. Here, a local minimum is reached. Moving the link parallel to itself toward the edge of the obstacle improves the solution—that is, it brings us closer to the perfect solution, where the link does not intersect the object at all—but does not result in a decrease of the penalty F_1. The penalty F_2, on the other hand, can only decrease when the manipulator link moves away from the top of the pyramid (i.e., the center of the obstacle). Thus, it is impossible to "get stuck" in a local minimum. The penalty function F_2 not only yields better convergence properties, but also is faster to compute than the function F_1. Fast computation of the function value of point C, in Figure 9, is possible because of the following property of the pyramidal function.

Initially, consider only square obstacles with sides

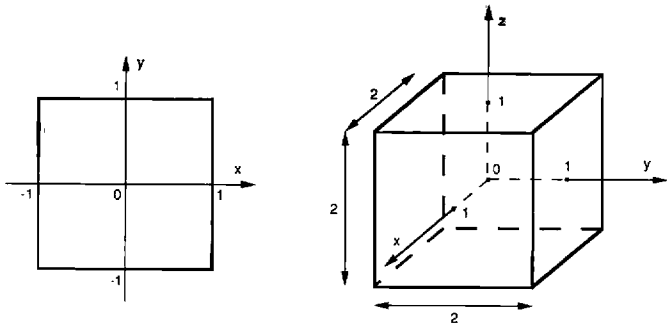

Fig. 10. Basic two-dimensional and three-dimensional obstacle.

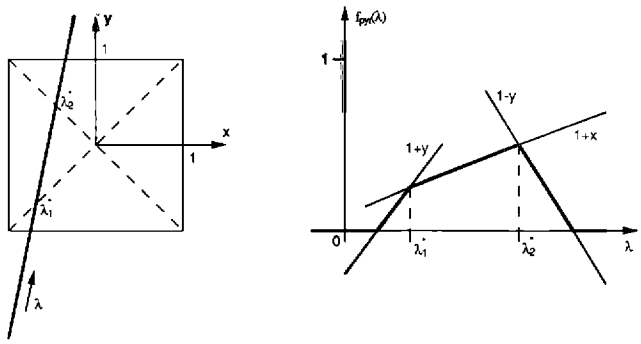

Fig. 11. A two-dimensional pyramidal function evaluated over a straight line.

of two units, as shown in Figure 10. For this case, the pyramidal function can be formulated as:

$$F_{\text{pyr}} = \min\left[\ \max(0, 1 - |x|),\ \max(0, 1 - |y|)\ \right] \tag{45}$$

This function, evaluated over a straight line passing through the obstacle, is depicted in Figure 11. The one-dimensional function, $f_{\text{pyr}}(\lambda)$, is piecewise linear and always reaches its maximum at λ_1^* or λ_2^*. Mathematically, these points correspond to:

$$\begin{aligned} x &= y \qquad \text{or} \\ x &= -y. \end{aligned} \tag{46}$$

Thus, to find the maximum, we only have to check the value of the pyramidal function for two points per line segment. Because the manipulator is represented by a set of line segments, the maximum of the pyramidal function over the total length of the manipulator is the maximum of the maxima for all the line segments considered separately.

This approach can be extended to obstacles of an arbitrary size in an arbitrary position/orientation by performing a frame transformation to the center of the obstacle, followed by a scaling operation. The extension of the pyramidal function to a cube in three dimensions, as

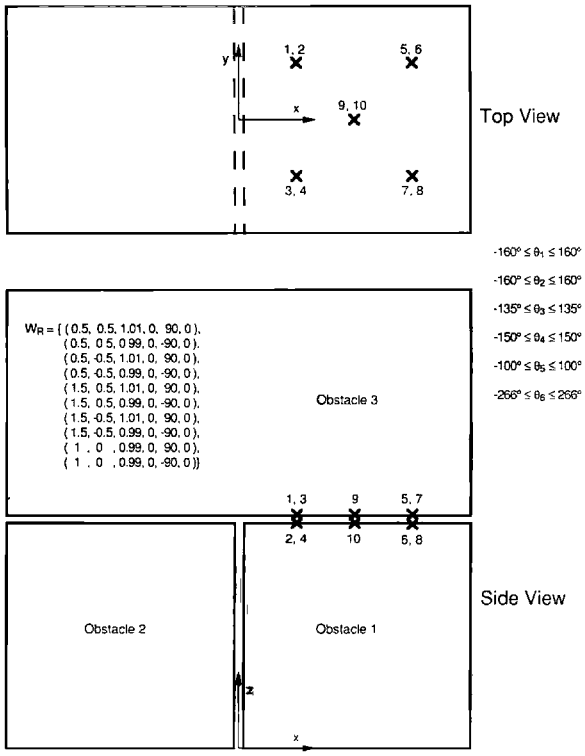

Fig. 12. Problem statement for the three-dimensional problem.

shown in Figure 10, is:

$$F_{\text{pyr}} = \min\left[\ \max(0, 1 - |x|),\ \max(0, 1 - |y|),\ \max(0, 1 - |z|)\ \right]. \tag{47}$$

Similarly, this function reaches a maximum over a straight line where:

$$\begin{aligned} & |x| = |y| \\ \text{or} \quad & |y| = |z| \\ \text{or} \quad & |z| = |x| \end{aligned} \tag{48}$$

8. A Three-Dimensional Example

In this section, it will be shown how the design methodology introduced in this article can be used to solve a comprehensive nonredundant design problem. The problem statement is given in Figure 12. The manipulator has to reach 10 points/orientations situated between two obstacles that are separated only by 0.02 m. The gap between the obstacles is 1 m above the floor. To complicate the situation even more, a third obstacle is positioned behind the manipulator. The limits on the joints of the manipulator are as those for a commercial PUMA 560 Mark II robot.

A 6-DOF manipulator has 17 D-H parameters. Because our implementation of the inverse kinematics assumes that the last three axes of the manipulator intersect at one point, we have to fix $a_4 = a_5 = d_5 = 0$. It must be noted that this is a limitation of our implementation of the inverse kinematics algorithm and not of the proposed design methodology, so it can be overcome by using numerical inverse kinematics (Au 1992). Furthermore, from a practical point of view, it is reasonable that one wrist is used to accomplish a whole group of tasks. Therefore, the twist angles of the wrist are fixed to their optimal values (Tsai and Soni 1984)—namely, $\alpha_4 = \pi/2$ and $\alpha_5 = -\pi/2$. The remaining 12 design parameters are indicated with a star in Table 2.

Table 2. The D-H parameters for spatial example 1

	DOF 1	DOF 2	DOF 3	DOF 4	DOF 5	DOF 6
			Initial Guess			
d	0.5*	0.1*	0.1*	0.0*	0.0	0.1*
a	0.0*	1.0*	1.0*	0.0	0.0	0.0*
α	90.0*	180.0*	−90.0*	90.0	−90.0	—
			Solution #1			
d	1.01	0.0	0.0	0.0	0.0	0.0
a	0.77	0.55	0.44	0.0	0.0	0.0
α	0.22	87.4	23.5	90.0	−90.0	—
			Solution #2			
d	1.00	0.01	0.0	0.0	0.0	0.0
a	0.87	0.00	0.90	0.0	0.0	0.01
α	−17.5	−160.4	−169.7	90.0	−90.0	—
			Solution #3			
d	1.00	0.0	0.0	0.0	0.0	0.0
a	0.82	0.11	0.85	0.0	0.0	0.0
α	−0.4	151.3	−98.7	90.0	−90.0	—

*Design parameters.

The initial guess, depicted in Figure 13, is an elbow-configured robot that has a vertical working plane and is unable to reach the specified points without passing through the obstacles. Figure 13 also shows three solutions for which the D-H parameters are given in Table 2.

The first solution corresponds roughly to Selectively Compliant Assembly Robot Arm (SCARA)–configured manipulator such as CMU DDArm II. This type of configuration has a horizontal working plane, which, for this particular design, is exactly between obstacles 1 and 3. The second solution has a spherical joint with a small angle between both rotation axes. It is intuitively clear that both solutions are able to reach the 10 points in W_R without passing through any obstacles. The third solution is less intuitive. However, detailed analysis of this manipulator shows that it meets all the task requirements. Therefore, one can argue that, for this specific task, this solution is no better or worse than a SCARA manipulator

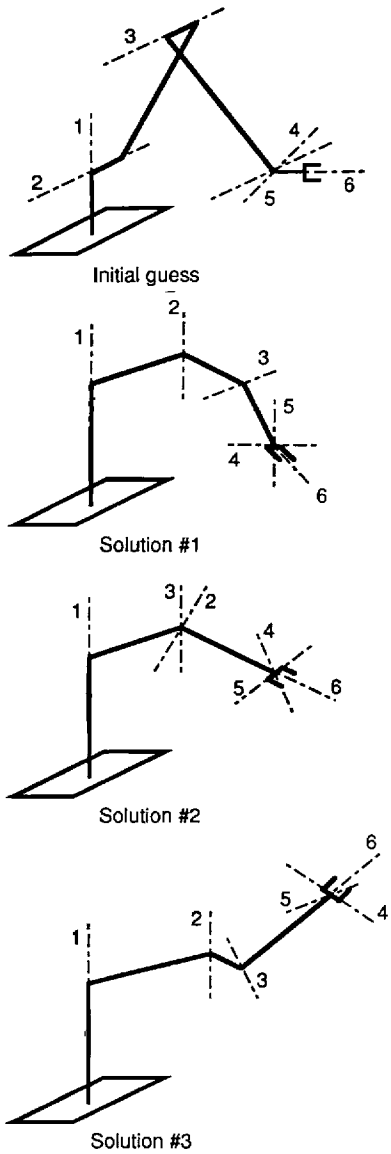

Fig. 13. Initial guess and three solutions.

or any other manipulator that satisfies the task requirements.

9. Summary

This article has proposed an approach to solve the kinematic design problem (i.e., the determination of the Denavit-Hartenberg parameters of a nonredundant manipulator with joint limits that can reach a set of specified points/orientations). This problem was first formulated mathematically using the concept of the kinematic hypersurface in the kinematic space. The problem statement was further particularized through a generalized formulation of the inverse kinematics that resulted in an analytical solution for the design of 2-DOF planar manipulators. To extend the capabilities of the solution procedure to spatial problems with six degrees-of-freedom, a numerical approach was developed. Using a global optimization procedure, the penalty of a manipulator design was minimized, resulting in an optimal kinematic configuration to perform the specified task. A simple 2-DOF planar manipulator example demonstrates that the numerical solutions correspond perfectly with the analytical solution. It was then shown how to extend the numerical procedure to include obstacle avoidance as a design criterion. Finally, a more comprehensive three-dimensional example demonstrated the power of the approach.

Acknowledgments

This research was funded in part by DOE under grant DE-F902-89ER14042, by Sandia under contract AC-3752-A, by the Department of Electrical and Computer Engineering, and by The Robotics Institute.

References

Au, W. K. F. 1992. Fault tolerant manipulator design. M.S. thesis. Electrical and Computer Engineering Department, Carnegie Mellon University.

Benhabib, B., Zak, G., and Lipton, M. G. 1989. A generalized kinematic modeling method for modular robots. *J. Robot. Sys.* 6(5):545–571.

Bremermann, H. 1970. A method of unconstrained global optimization. *Math. Biosci.* 9:1–15.

Chiu, S. L. 1988. Task compatibility of manipulator postures. *Int. J. Robot. Res.* 7(5):13–21.

Craig, J. J. 1989. *Introduction to Robotics: Mechanics and Control.* 2nd ed. Reading, MA: Addison-Wesley.

Denavit, J., and Hartenberg, R. S. 1955. A kinematic notation for lower-pair mechanisms based on matrices. *J. Appl. Mech.* 22(2):215–221.

Fletcher, R. 1987. *Practical Methods of Optimization.* 2nd ed. New York: John Wiley & Sons.

Gupta, K. C. 1986. On the nature of robot workspace. *Int. J. Robot. Res.* 5(2):112-121.

Kelmar, L., and Khosla, P. K. 1988 (Apr. 24–29, Philadelphia). Automatic generation of kinematics for a reconfigurable modular manipulator system. *Proc. 1988 IEEE Int. Conf. Robot. Automat.* Washington DC: IEEE, pp. 663–668.

Kim, J.-O. 1992. Task based kinematic design of robot manipulators. Ph.D. thesis. Robotics Ph.D. program, Carnegie Mellon University.

Kim, J.-O., and Khosla, P. K. 1991 (Nov. 3–5, Osaka). Dexterity measures for design and control of manipulators. *IROS'91: Int. Workshop on Intelligent Robots and Systems*. New York: IEEE, pp. 758–763.

Kirkpatrick, S., Gelatt, C. D. Jr., and Vecchi, M. P. 1983. Optimization by simulated annealing. *Science* 220(4598):671–680.

Krishnan, A., and Khosla, P. K. 1989 (Oct. 23–27, Dayton). A methodology for determining the dynamic configuration of a reconfigurable manipulator system. *Proc. 5th Annual Aerospace Applications of AI Conference*.

Lee, H.-Y., and Liang, C.-G. 1988. Displacements analysis of the general spatial 7-Link 7-R mechanism. *Mech. Mach. Theory* 23(3):219–226.

Lee, T. W., and Yang, D. C. H. 1983. On the evaluation of manipulator workspace. *ASME J. Mech. Trans. Automat. Des.* 105:70–77.

Lozano-Pérez, T., and Wesley, M. A. 1979. An algorithm for planning collision-free paths among polyhedral obstacles. *Comm. ACM* 22(10):560–570.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E. 1953. Equation of state calculations by fast computing machines. *J. Chem. Phys.* 21:1087–1092.

Paden, B., and Sastry, S. 1988. Optimal kinematic design of 6R manipulators. *Int. J. Robot. Res.* 7(2):43–61.

Paredis, C. J. J. 1990. An approach for mapping kinematic task specifications into a manipulator design. M.S. thesis. Electrical and Computer Engineering Department, Carnegie Mellon University.

Park, F. C., and Brockett, R. W. 1989 (Dec. 13–15, Tampa). Harmonic maps and the optimal design of mechanisms. *Proc. 28th Conf. on Decision and Contr.* New York: IEEE, pp. 206–210.

Piefke, F. 1978. Beziehungen zwichen Sehnenlangenverteilung und der Verteilung der Abstand zweier zufalliger Punkte im Eikorper. *Zeitsch. Wahrschein. Theorie verwendte Gebiete* 43:129–134.

Pieper, D. E., and Roth, B. 1969 (Zakopane, Poland). The kinematics of manipulators under computer control. *Proc. 2nd Int. Cong. on the Theory of Mach. and Mech.* pp. 159–169.

Raghavan, M., and Roth, B. 1989 (Aug. 28–31, Tokyo). Kinematic analysis of the 6R manipulator of general geometry. In Miura, H., and Arimoto, S. (eds.): *Robot. Res.: Fifth Int. Symp.* Cambridge, MA: MIT Press, pp. 314–320.

Rastegar, J., and Deravi, P. 1987. Methods to determine workspace, its subspaces with different number of configurations, and all the possible configurations of a manipulator. *Mech. Mach. Theory* 22(4):343–350.

Schmitz, D. E., Khosla, P. K., and Kanade, T. 1988 (Nov. 6–10, Sydney). The CMU reconfigurable modular manipulator system. *Proc. 19th Int. Symp. Exp. Rob. (ISIR)*. New York: Springer-Verlag, pp. 473–488.

Torn, A., and Žilinskas, A. 1989. *Global Optimization—Lecture Notes in Computer Science*. Vol. 350. New York: Springer-Verlag.

Tsai, L.-W., and Morgan, A. 1985. Solving the kinematics of the most general six- and five-degree-of-freedom manipulators by continuation methods. *ASME J. Mech. Trans. Automat. Des.* 107:189–200.

Tsai, Y. C., and Soni, A. H. 1984. The effect of link parameter on the working space of general 3R robot arms. *Mech. Mach. Theory* 19(1):9–16.

Tsai, Y. C., and Soni, A. H. 1985. Workspace synthesis of 3R, 4R, 5R and 6R robots. *Mech. Mach. Theory* 20(6):555–563.

Uchiyama, M., Shimizu, K., and Hakomori, K. 1985 (Aug. 20–23, Kyoto). Performance evaluation of manipulators using the Jacobian and its application to trajectory planning. In Hanafusa, H., and Inoue, H. (eds.): *Robot. Res.: Second Int. Symp.* Cambridge, MA: MIT Press, pp. 447–454.

van Laarhoven, P. J. M., and Aerts, E. H. L. 1987. *Simulated Annealing: Theory and Applications*. Boston: D. Reidel.

Vijaykumar, R., Waldron, K. J., and Tsai, M. J. 1986. Geometric optimization of serial chain manipulator structures for working volume and dexterity. *Int. J. Robot. Res.* 5(2):91–103.

Yang, D. C. H., and Lee, T. W. 1983. On the workspace of mechanical manipulators. *ASME J. Mech. Trans. Automat. Des.* 105:62–69.

Yoshikawa, T. 1985 (Aug. 20–23, Kyoto). Manipulability of robotic mechanisms. In Hanafusa, H., and Inoue, H. (eds.): *Robot. Res.: Second Int. Symp.* Cambridge, MA: MIT Press, pp. 439–446.

Žilinskas, A. G. 1986. *Global Optimization—Axiomatics of Statistical Models, Algorithms and Their Applications*. Vilnius: Mokslas.