Obstacle Accommodation Motion Planning

Yansong Shan and Yoram Koren, *Senior Member, IEEE*

***Abstract*—A new robot path planning methodology called *obstacle accommodation* has been introduced for robots in cluttered environments. This methodology does not attempt to avoid physical contact between a robot and obstacles. Instead, it controls the contact to prevent damage to the robot. Obstacle accommodation represents a new robotic paradigm in which obstacles can contact robots at unspecified positions, i.e., a contact point can be at any point on any link. Obstacle accommodation requires analysis in kinematics, motion planning, dynamics, and control. This paper deals with the development of the kinematic constraints and with motion planning under these constraints. We begin by providing a general formulation of the motion constraints due to contact with obstacles. Next, a new inverse kinematics is presented that provides joint motion for robots under contact constraints. Finally, the new motion planning algorithm for robot motion in a cluttered environment is provided. The motion planning algorithm is verified by two examples, one on a linkage robot and the other on a mobile robot.**

I. Introduction

As THE field of robotics has progressed, the environmental conditions under which robots must operate have become more and more complicated. It has become clear that one of the key issues to improving and widening the applications of robots is the enhancement of the adaptability of robots to their working environments. Controlling robot motion in challenging environments, especially in obstacle-cluttered environments, has been addressed by many scientists and engineers since the 1960s. To date, two approaches to robot motion planning and control have been used for the situations in which a robot's working environment must be considered. The first approach is obstacle avoidance, e.g., [3] and [10], and the second is hybrid force/position control, e.g., [2], [21], and [22].

Avoiding contact guarantees that no damage will occur to the robot and objects in its environment while in motion. However, in many applications, avoidance creates many difficulties in robot motion control and path planning. In certain very cluttered environments, robot motion may be totally disabled (for example, it is unrealistic to require a person to go through jungles without touching "obstacles"). Contact or collision may also occur because of the imperfect performance of motion controllers, especially when passing through narrow spaces. Therefore, it is desirable (sometimes necessary) to consider the possibility of contact between a robot and obstacles in motion control. The advantage of taking contact into account can also be seen in cases in which the contact provides information about an environment, such as the shape and the "hardness" of the obstacle, i.e., whether it is a piece of grass or a large stone.

The primary reason for obstacle avoidance is to avoid damage due to contact. However, contact is not necessarily dangerous to the robot as long as the contact can be controlled. A new motion control concept and methodology called *obstacle accommodation* has been proposed [19], [20]. This methodology allows contact between a robot and obstacles during motion, and controls the contact so that no damage occurs. This methodology is particularly useful in environments in which strict obstacle avoidance is difficult or impossible. Because true obstacles cannot be distinguished from soft objects by remote sensing, this methodology is essential in situations in which motion controllers do not have adequate knowledge of the true obstacles. For example, a piece of grass may be detected as an obstacle by range detectors such as ultrasonic sensors, and a robot may be unnecessarily controlled to avoid it.

The hybrid force/position control considers the interaction between a robot and its environment, however, these studies are based on a model in which the robot contacts its environment at its end-effector.

This paper studies motion planning for obstacle accommodation. In obstacle accommodation, the task for path planning is to design the end-effector motion and to find the consequent joint motion that will allow a robot to achieve the motion target without violating motion constraints due to obstacles. Here, contact points between a robot and obstacles are unspecified in the sense that they can be at any position on any link. During the motion, the robot may move in free space (no contact with obstacles) in one period of time and contact obstacles in another period of time. The number of contacted obstacles may change as well. Therefore, the challenge for the path generator is to take into consideration the changes in contact properties when generating the motion path. This paper approaches motion planning through the following steps.

First, it studies the formulation of constraints for obstacle accommodation. Two types of motion constraints are discussed: holonomic and the nonholonomic.

Next, it studies the inverse kinematics for robots under these constraints. Since joint motion must satisfy the motion constraints, the inverse kinematics must consider the motion constraints in the solutions. This paper provides a new general inverse kinematic solution to find the joint motion for a robot that is in contact with obstacles at *unspecified points* on *unspecified links*.

Manuscript received February 2, 1993; revised August 24, 1993. This work was presented in part at the International Conference on Intelligent Autonomous Systems, Pittsburgh, PA, February 15–18, 1993. This work was supported by the National Science Foundation under Grant IRI-9112717.

The authors are with the Department of Mechanical Engineering and Applied Mechanics, The University of Michigan, Ann Arbor, MI 48109 USA.

IEEE Log Number 9214698.

1042-296X/95$04.00 © 1995 IEEE

Finally, this paper develops an algorithm to generate the robot motion path. The path planning algorithm is extended for a robot in an environment in which it might contact several obstacles simultaneously.

This paper deals with the kinematics and motion planning issues of obstacle accommodation. One of the key problems involved in obstacle accommodation that is not discussed in this paper is the control of contact forces between the robot and obstacles. In real applications, contact forces may be caused by imperfect knowledge of the environment or any motion error. The contact forces might be very large if the motion constraints are improperly formed. We have conducted preliminary studies on controlling contact forces. From these studies, it is seen that, in general, the contact forces can be controlled at every contact point [19], [20].

II. Kinematic Constraints

Obstacles in a robot's working area restrict its motion. Due to contact with obstacles, the motion of the robot is constrained, and the available motion space is reduced. In addition to the physical constraints due to contact with obstacles, some constraints may be made by the robot designer to further restrict motion. For example, once in contact with an obstacle, the relative motion between a robot and the obstacles might be required to be pure rolling to prevent wear damage due to sliding motion. This type of constraint generally is nonholonomic. This section provides a general method to formulate the constraints for obstacle accommodation. Some studies have been done on the motion constraints for robotic dexterous hand manipulation [4], [5], [7], [11] and for robot force control [2], [13], [21], [22]; however, the kinematic constraints discussed in this paper are due to obstacle contact with a robot at unspecified positions on unspecified links and due to multiobstacle contact.

A. Kinematic Descriptions of Linkages and Obstacles

In this section, we present the kinematic descriptions of a robot linkage and the surfaces involved in contact. Without losing generality, suppose that link i ($i = 1, \cdots, n$, where n is the number of links) is in contact with an obstacle, as shown in Fig. 1. In the analysis (and throughout this paper), we assume that the contact is a point contact and all the contact surfaces are rigid. Further, all the obstacles are assumed to be fixed with the world coordinate system.

As shown in Fig. 1, we assign each joint of the linkage a coordinate system, $\boldsymbol{O}_i$. The first joint coordinate system $\boldsymbol{O}_0$ is attached to the base of the robot, the next joint coordinate $\boldsymbol{O}_1$ is attached to link 1, so forth to the joint coordinate $\boldsymbol{O}_n$ which is attached to link n. The joint coordinate system $\boldsymbol{O}_0$ is also assumed to coincide with the world coordinate system.

Suppose the surface of link i is expressed in terms of $\boldsymbol{O}_i$ as

$$\boldsymbol{x}_i = [x_i(u_i, v_i),\, y_i(u_i, v_i),\, z_i(u_i, v_i)]^{\mathrm{T}}$$

where x_i, y_i, and z_i are the coordinates of any point on the surface of link i with respect to $\boldsymbol{O}_i$, and u_i and v_i are the parameters of the surface. (See [14] for more detailed study on the surface description.) We describe the contact point in terms of the world coordinate system as (see Appendix A)

$$\boldsymbol{x}_{i0} = \boldsymbol{R}_0^i \boldsymbol{x}_i + \boldsymbol{p}_0^i \tag{1}$$

where $\boldsymbol{x}_{i0}$ is a description of the contact position in the world coordinate system, $\boldsymbol{R}_0^i$ and $\boldsymbol{p}_0^i$ are rotational and linear transformation components in homogeneous transformation matrix from $\boldsymbol{O}_i$ to world coordinate system [6], [9].

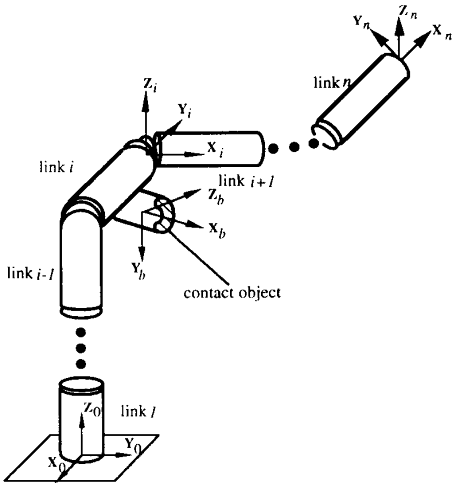


Fig. 1. A robot in contact with an obstacle.

At the contact point, we can have two perpendicular tangent vectors which form a tangent plane to the surface (in some geometric textbooks these tangent vectors are referred to as velocity vectors [14]). Suppose the unit vectors along these two tangent vectors are denoted by $\boldsymbol{t}_{ui}$ and $\boldsymbol{t}_{vi}$, then the unit outward normal vector of the surface at the point $\boldsymbol{t}_{wi}$ is the crossproduct of these two vectors. Here, the subscript i means that these three vectors are expressed in terms of coordinate system $\boldsymbol{O}_i$. We assume that the functions describing the link surfaces in $\boldsymbol{O}_i$ and the obstacle surfaces in $\boldsymbol{O}_b$ are at least first-order differentiable. The three vectors are shown in Fig. 2. $\boldsymbol{t}_{ui}$, $\boldsymbol{t}_{vi}$ are given by

$$\boldsymbol{t}_{ui} = \frac{1}{\sqrt{\left(\frac{\partial x_i(u_i, v_i)}{\partial u_i}\right)^2 + \left(\frac{\partial y_i(u_i, v_i)}{\partial u_i}\right)^2 + \left(\frac{\partial z_i(u_i, v_i)}{\partial u_i}\right)^2}} \cdot \begin{bmatrix} \frac{\partial x_i(u_i, v_i)}{\partial u_i} \\ \frac{\partial y_i(u_i, v_i)}{\partial u_i} \\ \frac{\partial z_i(u_i, v_i)}{\partial u_i} \end{bmatrix} \tag{2a}$$

$$\boldsymbol{t}_{vi} = \frac{1}{\sqrt{\left(\frac{\partial x_i(u_i, v_i)}{\partial v_i}\right)^2 + \left(\frac{\partial y_i(u_i, v_i)}{\partial v_i}\right)^2 + \left(\frac{\partial z_i(u_i, v_i)}{\partial v_i}\right)^2}} \cdot \begin{bmatrix} \frac{\partial x_i(u_i, v_i)}{\partial v_i} \\ \frac{\partial y_i(u_i, v_i)}{\partial v_i} \\ \frac{\partial z_i(u_i, v_i)}{\partial v_i} \end{bmatrix} \tag{2b}$$

and $\boldsymbol{t}_{wi}$ is given by

$$\boldsymbol{t}_{wi} = \boldsymbol{t}_{ui} \times \boldsymbol{t}_{vi}. \tag{2c}$$

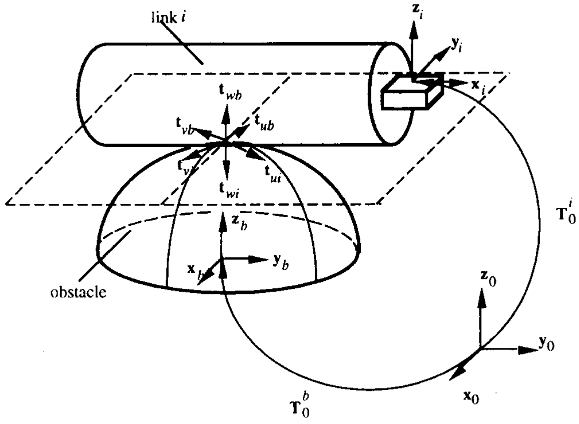


Fig. 2. Coordinate systems on the contacted link and the obstacle.

In Fig. 2, if we assign a coordinate system O_b to the obstacle, then the surface of the obstacle can be expressed in terms of O_b as

$$\boldsymbol{x}_b = [x_b(u_b, v_b), y_b(u_b, v_b), z_b(u_b, v_b)]^{\mathrm{T}}.$$

Any contact point on the obstacle surface can also be expressed in terms of the world coordinate system as (see Appendix A)

$$\boldsymbol{x}_{b0} = \boldsymbol{R}_0^b \boldsymbol{x}_b + \boldsymbol{p}_0^b \tag{3}$$

where $\boldsymbol{x}_{b0}$ is a description of the contact position in the world coordinate system, $\boldsymbol{R}_0^b$ and $\boldsymbol{p}_0^b$ are rotational and linear transformation components in homogeneous transformation matrix from O_b to world coordinate system.

We can calculate the unit tangent vectors and outward normal of the obstacle surface at the contact point in the same way as for the link surface. Denoting $\boldsymbol{t}_{ub}$ and $\boldsymbol{t}_{vb}$ as the unit tangent vectors along the two tangent vectors (velocity vectors) and $\boldsymbol{t}_{wb}$ as the unit outward normal vector to the obstacle surface, then

$$\boldsymbol{t}_{ub} = \frac{1}{\sqrt{\left(\frac{\partial x_b(u_b, v_b)}{\partial u_b}\right)^2 + \left(\frac{\partial y_b(u_b, v_b)}{\partial u_b}\right)^2 + \left(\frac{\partial z_b(u_b, v_b)}{\partial u_b}\right)^2}} \cdot \begin{bmatrix} \frac{\partial x_b(u_b, v_b)}{\partial u_b} \\ \frac{\partial y_b(u_b, v_b)}{\partial u_b} \\ \frac{\partial z_b(u_b, v_b)}{\partial u_b} \end{bmatrix} \tag{4a}$$

$$\boldsymbol{t}_{vb} = \frac{1}{\sqrt{\left(\frac{\partial x_b(u_b, v_b)}{\partial v_b}\right)^2 + \left(\frac{\partial y_b(u_b, v_b)}{\partial v_b}\right)^2 + \left(\frac{\partial z_b(u_b, v_b)}{\partial v_b}\right)^2}} \cdot \begin{bmatrix} \frac{\partial x_b(u_b, v_b)}{\partial v_b} \\ \frac{\partial y_b(u_b, v_b)}{\partial v_b} \\ \frac{\partial z_b(u_b, v_b)}{\partial v_b} \end{bmatrix} \tag{4b}$$

$$\boldsymbol{t}_{wb} = \boldsymbol{t}_{ub} \times \boldsymbol{t}_{vb}. \tag{4c}$$

In summary, (1) and (3) give the transformations of a point from O_i and O_b to O_0 while (2) and (4) define the unit outward normal vectors of the contact surfaces. These equations will be used in developing the motion constraints.

B. Holonomic Constraints

In this section, we formulate the physical constraints on the robot motion that attempt to merge the material of two contact objects (obstacle and robot link) at the contact point. As will be seen later, the motion constraints are holonomic because they can be written in the form of

$$\boldsymbol{C}(\boldsymbol{\Theta}, \boldsymbol{u}) = 0$$

where $\boldsymbol{C}$ is an $r \times 1$ vector, if r constraints are imposed on the robot motion (as will be seen later, $r = 5$ for a single contact point)

$$\boldsymbol{\Theta} = [\theta_1, \theta_2, \cdots, \theta_n]^{\mathrm{T}}$$

is the joint variable vector, and

$$\boldsymbol{u} = [u_i, v_i, u_b, v_b]^{\mathrm{T}}$$

is the contact surface parameter vector.

The unit outward normal of the link surface can be transformed into O_0 by

$$\boldsymbol{t}_{wi0} = \boldsymbol{R}_0^i \boldsymbol{t}_{wi} = \boldsymbol{R}_0^i(\boldsymbol{t}_{ui} \times \boldsymbol{t}_{vi}). \tag{5}$$

Similarly, the direction of the unit outward normal of the obstacle surface can be transformed into O_0 by

$$\boldsymbol{t}_{wb0} = \boldsymbol{R}_0^b \boldsymbol{t}_{wb} = \boldsymbol{R}_0^b(\boldsymbol{t}_{ub} \times \boldsymbol{t}_{vb}). \tag{6}$$

Since at each contact point, these two outward normal vectors must lie on the same line but in opposite directions, i.e., $\boldsymbol{t}_{wb0} = -\boldsymbol{t}_{wi0}$, we have

$$\boldsymbol{R}_0^b(\boldsymbol{t}_{ub} \times \boldsymbol{t}_{vb}) = -\boldsymbol{R}_0^i(\boldsymbol{t}_{ui} \times \boldsymbol{t}_{vi}). \tag{7}$$

In addition to the constraint on the outward normal of the contact surfaces described in (7), we have another set of constraints based on the fact that the contact point on the link surface and on the obstacle surface coincide in the world coordinate system. Based on (1) and (3) we have

$$\boldsymbol{R}_0^b \begin{bmatrix} x_b(u_b, v_b) \\ y_b(u_b, v_b) \\ z_b(u_b, v_b) \end{bmatrix} + \boldsymbol{p}_0^b = \boldsymbol{R}_0^i \begin{bmatrix} x_i(u_i, v_i) \\ y_i(u_i, v_i) \\ z_i(u_i, v_i) \end{bmatrix} + \boldsymbol{p}_0^i. \tag{8}$$

Equations (7) and (8) give the constraints on the variables u_i, v_i, u_b, and v_b as well as the linkage joint angles. In the equations, we have $n + 4$ unknowns: θ_js ($j = 1, \cdots, n$), u_i, v_i, u_b, and v_b. Each side of (7) gives the projections of the outward normal vector onto the world coordinate system, since only two of projections are independent in $\boldsymbol{R}^3$ space, there are only two equations in (7) that are independent. In (8), we have three independent equations. So, (7) and (8) give a total of 5 independent equations.

Since (7) and (8) do not involve the time derivatives of θ_js and $\boldsymbol{u}$, these two sets of equations can be rearranged into a standard holonomic constraint form

$$\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{u}) = 0.$$

Ideally, eliminating u_i, v_i, u_b, and v_b by the four equations in (7) and (8) and substituting them into the last independent equation, we can have a holonomic constraint equation on the joint variables

$$N(\boldsymbol{\Theta}) = 0.$$

Usually, it is difficult to eliminate surface parameters from (7) and (8) and to form a constraint that contains only the joint variables $N(\boldsymbol{\Theta}) = 0$. Therefore, in general, we are more concerned with the differential form of the constraints in the inverse kinematic analysis and motion planning. Differentiating (7) and (8) and eliminating $\dot{u}_i$, $\dot{v}_i$, $\dot{u}_b$, and $\dot{v}_b$, we can have a constraint for each contact as

$$c_i(\boldsymbol{\Theta}, \boldsymbol{u}_i)\dot{\boldsymbol{\Theta}} = 0$$

where c_i is an $1 \times n$ vector. Every contact will generate one constraint equation, so when the linkage contacts several obstacles, more holonomic constraints are generated. Suppose there are r number of constraints, then the holonomic constraint, in vector form, can be defined by

$$\boldsymbol{C}(\boldsymbol{\Theta}, \boldsymbol{U})\dot{\boldsymbol{\Theta}} = \begin{bmatrix} c_1(\boldsymbol{\Theta}, \boldsymbol{u}_1) \\ c_2(\boldsymbol{\Theta}, \boldsymbol{u}_2) \\ \vdots \\ c_r(\boldsymbol{\Theta}, \boldsymbol{u}_r) \end{bmatrix} \dot{\boldsymbol{\Theta}} = \boldsymbol{0} \tag{9}$$

where

$$\boldsymbol{U} = [\boldsymbol{u}_1^{\mathrm{T}}, \boldsymbol{u}_1^{\mathrm{T}}, \cdots, \boldsymbol{u}_r^{\mathrm{T}}]^{\mathrm{T}}$$

includes the contact surface parameters of all the contact points. Note that during the motion, the number r may change because the robot may contact different numbers of obstacles in different motion periods.

In this study, we only consider the case where the r constraints are independent. For r independent constraint functions, the robot can only have mobility in $(n - r)$-dimensional space. If $r = n$ then the motion is totally blocked. Note that $r \not> n$ because the maximum number of independent constraints on an n DOF linkage is n. In summary, the holonomic constraints are assumed to satisfy the following condition: the constraint functions are independent in the sense that the $r \times n$ matrix

$$\boldsymbol{C}(\boldsymbol{\Theta}, \boldsymbol{U}) \in \boldsymbol{R}^{r \times n}$$

has full rank r.

C. Nonholonomic Constraints

The holonomic motion constraints developed in the last section can be considered to be *natural* or *physical* since they are based on the fact that the contact object will not merge at contact points. The assumption is true for any rigid body contact, since the constraints developed must be *physically satisfied* at every contact point.

There are some situations in which certain extra constraints need to be imposed onto the motion. These constraints are considered to be *artificial* in the sense that they are made by the robot designer. For instance, if the relative motion between a robot link and an obstacle is allowed only to be rolling (the translational motion usually causes wear damage on the link surface), certain additional constraints must be imposed on the robot motion control. These constraints are of nonholonomic type since, as will be seen later, these constraints cannot be written in the form of (7) or (8) while they have to be expressed in terms of joint velocities and the time derivative of surface parameters as

$$\sum_{i=1}^{n} a_{ji}\, d\theta_i + \sum_{l=1}^{m_u} b_{jl}\, du_l + a_{jt}\, dt = 0 \qquad j = 1, 2, \cdots, m \tag{10}$$

where m is the number of total constraints, the as and bs are, in general, functions of θ s and $\boldsymbol{U}$. u_l represents every one of the components in $\boldsymbol{U}$, and m_u is the total number of us in $\boldsymbol{U}$. Furthermore, if (10) is nonholonomic, this differential expression is characterized by *being not integrable*.

Cai and Roth [4], Montana [15], and Cole, Hauser, and Sastry [5] have studied the motion of contacted objects in general. Their studies provide kinematic relations including both holonomic and nonholonomic types. Based on their studies, certain nonholonomic constraints are usually involved in motion constraints when pure rolling between two objects is expected.

To formulate the nonholonomic constraints for obstacle accommodation, we express the translational and rotational velocities of joint coordinate system $\boldsymbol{O}_i$ with respect to the world coordinate system by $\boldsymbol{v}_i$ and ω_i. $\boldsymbol{v}_i$ and ω_i can be calculated in terms of joint variables as

$$\begin{bmatrix} \boldsymbol{v}_i \\ \omega_i \end{bmatrix} = \boldsymbol{J}_{\boldsymbol{O}_i}\dot{\boldsymbol{\Theta}}_i \tag{11}$$

where $\boldsymbol{J}_{\boldsymbol{O}_i}$ is the Jacobian matrix for joint coordinate frame $\boldsymbol{O}_i$ and

$$\dot{\boldsymbol{\Theta}}_i = [\dot{\theta}_1, \dot{\theta}_2, \cdots, \dot{\theta}_i]^{\mathrm{T}}.$$

Note that $\boldsymbol{v}_i$ and ω_i are functions of joint variables $\theta_1, \theta_2, \cdots$, and θ_i as well as their velocities.

Contact point velocity in terms of the world coordinate system can be calculated through

$$\boldsymbol{v}_c = \boldsymbol{v}_i + \omega_i \times \boldsymbol{R}_0^i \boldsymbol{x}_i. \tag{12}$$

Since pure rolling motion between the contact link and the obstacles is required, the velocity of a contact point is required to be zero. Therefore, the constraints on the motion are

$$\boldsymbol{v}_i + \omega_i \times \boldsymbol{R}_0^i \boldsymbol{x}_i = 0. \tag{13}$$

Equation (13) consists of a set of equations involving joint velocities.

Differentiating (13) yields (noticing $\dot{\boldsymbol{p}}_0^b = 0$ and $\boldsymbol{O}_b$ is fixed with respect to world coordinate)

$$\boldsymbol{R}_0^b \dot{\boldsymbol{x}}_b = \omega_i \times \boldsymbol{R}_0^i \boldsymbol{x}_i + \boldsymbol{R}_0^i \dot{\boldsymbol{x}}_i + \dot{\boldsymbol{p}}_0^i \tag{14}$$

here

$$\dot{\boldsymbol{p}}_0^i = \boldsymbol{v}_i.$$

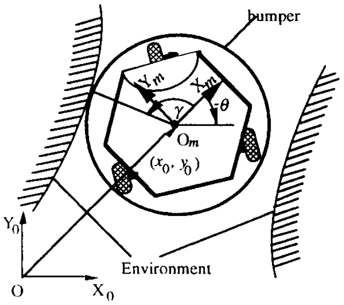


Fig. 3. A mobile robot moving through a narrow path.

From (13), we have

$$\boldsymbol{R}_0^b \dot{\boldsymbol{x}}_b = \boldsymbol{R}_0^i \dot{\boldsymbol{x}}_i. \tag{15}$$

Equation (15) can be written in terms of surface parameters of both links and obstacle as

$$\boldsymbol{R}_0^b\left(\frac{\partial \dot{\boldsymbol{x}}_b}{\partial u_b}\dot{u}_b + \frac{\partial \dot{\boldsymbol{x}}_b}{\partial u_b}\dot{v}_b\right) = \boldsymbol{R}_0^i\left(\frac{\partial \dot{\boldsymbol{x}}_i}{\partial u_i}\dot{u}_i + \frac{\partial \dot{\boldsymbol{x}}_i}{\partial u_i}\dot{v}_i\right). \tag{16}$$

Equation (16) provides constraints in terms of the surface variables. In general, (13) and (16) cannot be integrated into a form that consists of only joint variables and surface parameters without their time derivatives involved. Therefore, (13) or (16) provides the motion for a set of nonholonomic constraints for the motion.

D. Constraints on the Motion of a Wheeled Mobile Robot

In this section, we discuss the constraints on a wheeled mobile robot when it passes through an area where contact with obstacles is expected. Fig. 3 shows a wheeled mobile robot that has a circular bumper. The scenario represents the case in which the robot is supposed to pass through a narrow opening (narrow door or narrow path). Because of the imperfect performance of the controller or inaccurate knowledge of the environment during the motion, the robot may make contact with the environment.

If a denotes the radius of the bumper, a contact point on the bumper can be determined in terms of coordinate $\boldsymbol{O}_0$ through angle γ by

$$x_c = x_0 + a\cos(\gamma + \theta)$$

$$y_c = y_0 + a\sin(\gamma + \theta) \tag{17}$$

where θ is the orientation of $\boldsymbol{O}_m$ with respect to world coordinate $\boldsymbol{O}_0$ and γ is the angle of contact point position vector in terms of $\boldsymbol{O}_m$. If the environment is described in $\boldsymbol{O}_0$ as

$$y = G(x)$$

then we have

$$y_0 + a\sin(\gamma + \theta) = G(x_0 + a\cos(\gamma + \theta)). \tag{18}$$

The outward normal of the environment's surface at the contact point can be calculated by

$$N_{b0} = \frac{1}{\sqrt{1 + \left(\frac{\partial G}{\partial x}\right)^2}}\begin{bmatrix} -\left(\frac{\partial G}{\partial x}\right) \\ 1 \end{bmatrix}. \tag{19}$$

The outward normal of the robot bumper at the contact point in terms of $\boldsymbol{O}_m$ can be calculated as

$$N_m = \begin{bmatrix} \cos\gamma \\ \sin\gamma \end{bmatrix}. \tag{20}$$

The N_m can be represented in terms of the world coordinate by

$$N_{m0} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}\begin{bmatrix} \cos\gamma \\ \sin\gamma \end{bmatrix}. \tag{21}$$

From (19) and (21), we have

$$\frac{1}{\sqrt{1 + \left(\frac{\partial G}{\partial x}\right)^2}}\begin{bmatrix} -\left(\frac{\partial G}{\partial X}\right) \\ 1 \end{bmatrix} = \boldsymbol{R}(\theta)\begin{bmatrix} \cos\gamma \\ \sin\gamma \end{bmatrix} \tag{22}$$

where

$$\boldsymbol{R}(\cdot) = \begin{bmatrix} \cos(\cdot) & -\sin(\cdot) \\ \sin(\cdot) & \cos(\cdot) \end{bmatrix}.$$

Equations (18) and (22) are holonomic constraints on the motion of the robot in contact with the environment. From simple kinematic analysis, the contact point velocity is

$$\boldsymbol{v}_c = \begin{bmatrix} \dot{x}_0 \\ \dot{y}_0 \end{bmatrix} + \begin{bmatrix} -a\dot{\theta}\sin(\gamma + \theta) \\ a\dot{\theta}\cos(\gamma + \theta) \end{bmatrix} = 0. \tag{23}$$

It can be easily seen that, if the shape of the bumper is more complicated than a circular shape, a is no longer a constant, (23) will be a more complicated nonholonomic constraint equation.

III. Inverse Kinematics

In this section, a general inverse kinematics solution is provided that calculates the joint motion for any desired end-effector motion for a robot under any kinematic constraints (both holonomic and nonholonomic).

In the analysis, we present both holonomic constraints and nonholonomic constraints in a unified form

$$\boldsymbol{H}(\boldsymbol{\Theta}, \boldsymbol{U})\dot{\boldsymbol{\Theta}} = 0 \tag{24}$$

where $\boldsymbol{H}$ is an $r \times n$ matrix, r is the number of constraints, n is the number of system variables, and $\boldsymbol{U}$ is the contact surface parameter vector. Throughout this section, the constraints are assumed to satisfy the following conditions: a) the number of constraint functions is less than the degrees-of-freedom, i.e., $r < n$; b) the solution set $\{\dot{\boldsymbol{\Theta}} \in R^n \mid \boldsymbol{H}(\boldsymbol{\Theta}, \boldsymbol{U})\dot{\boldsymbol{\Theta}} = 0\}$ is not empty; and c) the constraints are independent in the sense that the $r \times n$ matrix $\boldsymbol{H}$ has full rank of r.

A. Differential Inverse Kinematics

In developing the inverse kinematics, we consider the differential inverse kinematics. We do so for the following reasons. a) In real applications, a robot may contact obstacles at any time. Since the robot must satisfy the constraints at all points along its trajectory, the differential analysis is more appropriate. b) The contact properties between a robot and obstacles change during the motion, so the motion constraints change. In some cases, certain motion constraints only appear in one period of the motion. In these situations, the general inverse kinematics solution may not exist. For a robot to satisfy the changing constraints at all times, only the instantaneous kinematics is applicable.

In general, the forward kinematics give a relationship between the variables of the linkage end-effector (position and orientation) and the joint coordinate variables. Let

$$\boldsymbol{x} = [x_1, x_2, \cdots, x_m]^{\mathrm{T}}$$

denote the position and the orientation of a linkage end-effector, and let $\boldsymbol{\Theta}$ denote the vector of joint variables. The relationship between $\boldsymbol{x}$ and $\boldsymbol{\Theta}$ can be generally expressed as

$$\boldsymbol{x} = \boldsymbol{F}(\boldsymbol{\Theta}) \tag{25}$$

where $\boldsymbol{F} \in \boldsymbol{R}^{m\times 1}$ is a set of functions of $\boldsymbol{\Theta}$. Differentiating (25) yields the following equation:

$$\dot{\boldsymbol{x}} = \frac{\partial \boldsymbol{F}(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \dot{\boldsymbol{\Theta}} \tag{26a}$$

or

$$\dot{\boldsymbol{x}} = \boldsymbol{J}(\boldsymbol{\Theta})\dot{\boldsymbol{\Theta}} \tag{26b}$$

where

$$\boldsymbol{J}(\boldsymbol{\Theta}) = \frac{\partial \boldsymbol{F}(\boldsymbol{\Theta})}{\partial \boldsymbol{\Theta}} \in \boldsymbol{R}^{m\times n}$$

is the linkage Jacobian matrix. If the robot works in an obstacle-free environment, the solution for $\dot{\boldsymbol{\Theta}}$ can be found for the following three cases separately.

Case A—$m = n$: In this case, the Jacobian matrix is an $n \times n$ square matrix. Assuming the Jacobian has full rank of n, the solution can be obtained simply through

$$\dot{\boldsymbol{\Theta}} = \boldsymbol{J}^{-1}\dot{\boldsymbol{x}}. \tag{27}$$

Case B—$m > n$: In this case, the number of equations is greater than the number of unknowns, and the Jacobian matrix is an $m \times n$ matrix. The solution can be obtained through minimizing the least-square error

$$\boldsymbol{E}(\dot{\boldsymbol{\Theta}}) = (\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}})(\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}})^{\mathrm{T}} \tag{28}$$

and the solution is

$$\dot{\boldsymbol{\Theta}} = \boldsymbol{J}^{I}\dot{\boldsymbol{x}} \tag{29}$$

where

$$\boldsymbol{J}^{I} = (\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{J}^{\mathrm{T}}$$

is the pseudoinverse Jacobian matrix. Here we assume that matrix $(\boldsymbol{J}^{T}\boldsymbol{J})$ has full rank of m.

Case C—$m < n$: In this case, the number of equations is less than the number of unknowns. The solution can be found through minimizing

$$L(\dot{\boldsymbol{\Theta}}) = \dot{\boldsymbol{\Theta}}^{\mathrm{T}}\dot{\boldsymbol{\Theta}} \tag{30}$$

subject to

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{\Theta}}. \tag{31}$$

The solution depends on the pseudoinverse Jacobian matrix $\boldsymbol{J}^{I} = \boldsymbol{J}^{\mathrm{T}}(\boldsymbol{H}\boldsymbol{H}^{\mathrm{T}})^{-1}$, and

$$\dot{\boldsymbol{\Theta}} = \boldsymbol{J}^{\mathrm{T}}(\boldsymbol{J}\boldsymbol{J}^{\mathrm{T}})^{-1}\dot{\boldsymbol{x}}. \tag{32}$$

Here, we assume that matrix $(\boldsymbol{J}\boldsymbol{J}^{\mathrm{T}})$ has full rank of n. In the next section, we will develop the inverse kinematics equations to solve $\dot{\boldsymbol{\Theta}}$ in terms of $\dot{x}$ for robots under constraints due to contact with obstacles.

B. Inverse Kinematics Under Constraints

When a robot linkage system is subject to constraints due to contact with obstacles, the solution to the differential inverse kinematics is more complicated since the solution has to satisfy the contact constraints. Suppose the number of motion constraints is r. If $m+r > n$, then the motion requirements are more than the number of joint variables (we see constraints as motion requirements), and the solution cannot be absolutely obtained. Just as with the case of $m > n$ for robots in free space, the solution is obtained through optimization techniques. If $m + r = n$, the number of requirements of motion is the same as the joint variables, and the solution can be found through manipulating n equations for n unknowns. If $m + r < n$, the system is considered to be redundant in the sense that the number of system unknown variables is greater than the number of motion requirements. Just as with the case of $n > m$ for the robot moving in free space, the inverse kinematics is obtained through optimization techniques. In the following, we will develop the mathematics to solve the differential inverse kinematics for these three cases separately.

Inverse Kinematics When $m + r > n$: When $m + r > n$, the number of task dimensions and the number of motion constraints are greater than the number of system unknowns; therefore, the exact solution does not exist. The problem can be solved by minimizing the least square error

$$E(\dot{\boldsymbol{\Theta}}) = (\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}})^{\mathrm{T}}(\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}}) \tag{33}$$

subject to

$$\boldsymbol{H}\dot{\boldsymbol{\Theta}} = 0 \tag{34}$$

where $\boldsymbol{H}$ is the motion constraints involving both holonomic and nonholonomic types of constraints. The solution to this optimization problem [1], [12] is (see Appendix B)

$$\dot{\boldsymbol{\Theta}} = (\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}[\boldsymbol{J}^{\mathrm{T}} - \boldsymbol{H}^{\mathrm{T}}[\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{H}^{\mathrm{T}}]^{-1}\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{J}^{\mathrm{T}}]\dot{\boldsymbol{x}} \tag{35}$$

In (35), all the inverse operations are assumed on an invertible matrix.

Inverse Kinematics When $m + r = n$: When $m + r = n$, since the number of unknowns equals the number of motion requirements and the number of constraints, the exact solution exists. In this case, n equations available, in which m equations are from motion tasks, i.e.,

$$\dot{\boldsymbol{x}} = \boldsymbol{J}^{m \times n} \dot{\boldsymbol{\Theta}}$$

and r from the motion constraints, i.e.,

$$\boldsymbol{H}^{r \times n} \dot{\boldsymbol{\Theta}} = 0$$

the solution can be found by directly solving the following equation (see Appendix B)

$$\dot{\boldsymbol{\Theta}} = \begin{bmatrix} \boldsymbol{J}^{m \times n} \\ \boldsymbol{H}^{r \times n} \end{bmatrix}^{-1} \begin{bmatrix} \dot{\boldsymbol{x}}^{m \times 1} \\ \boldsymbol{0}^{r \times 1} \end{bmatrix}. \tag{36}$$

In (36), all the inverse operations are assumed on an invertible matrix.

Inverse Kinematics When $m + r < n$: When $m + r < n$, the number of system unknowns is greater than the number of motion task requirements (number of task dimensions) and the number of motion constraints; therefore, the system is undetermined. We solve the inverse kinematics by minimizing the value of

$$L = \dot{\boldsymbol{\Theta}}^{\mathrm{T}} \dot{\boldsymbol{\Theta}} \tag{37}$$

subject to

$$\dot{\boldsymbol{x}} = \boldsymbol{J} \dot{\boldsymbol{\Theta}} \tag{38}$$

and

$$\boldsymbol{H} \dot{\boldsymbol{\Theta}} = 0. \tag{39}$$

The solution to this optimization problem is (see Appendix B)

$$\begin{aligned} \dot{\boldsymbol{\Theta}} = {} & [\boldsymbol{J}^{\mathrm{T}} - \boldsymbol{H}^{\mathrm{T}} (\boldsymbol{H} \boldsymbol{H}^{\mathrm{T}})^{-1} (\boldsymbol{H} \boldsymbol{J}^{\mathrm{T}})] \\ & \cdot \{\boldsymbol{J}[\boldsymbol{J}^{\mathrm{T}} - \boldsymbol{H}^{\mathrm{T}} (\boldsymbol{H} \boldsymbol{H}^{\mathrm{T}})^{-1} (\boldsymbol{H} \boldsymbol{J}^{\mathrm{T}})]\}^{-1} \dot{\boldsymbol{x}}. \end{aligned} \tag{40}$$

In (40), all the inverse operations are assumed on an invertible matrix.

IV. Path Planning

A. Relative Motion between a Robot and Obstacles

To make motion planning for a robot in an obstacle-cluttered environment, we need to consider the relative motion between the robot and the obstacles. For any particular obstacle, the relative motion between the robot and the obstacle will be one of the following situations:

1) The robot moves towards the obstacle;
2) The robot moves while remaining in contact with the obstacle;
3) The robot moves away from the obstacle.

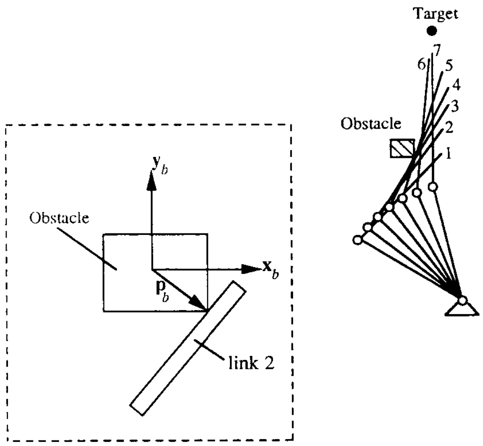


Fig. 4. Motion toward a target.

Fig. 4 shows these three types of relative motion between a two-link robot and an obstacle. As can be seen, during stage 1 to stage 3, the robot moves close to the obstacle but yet is in free space; from stage 3 to stage 5, the robot moves in contact with the obstacle (slides along the obstacle); finally during stage 5 to stage 7, the robot moves away from the obstacle.

These relative motions can be interpreted in the robot joint configuration space. Recall that once a robot is in contact with certain obstacles, a set of holonomic constraints is established on the robot motion. Denoting the undifferentiated form of the holonomic constraints as

$$\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{U}) = 0 \tag{41}$$

we can divide the motion space into two parts: $\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{U}) > 0$, where the robot can move without contacting the obstacles, and $\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{U}) < 0$, into which any robot motion is forbidden. Note that $\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{U}) = 0$ represents the sub-space in which the motion of the robot is in contact with obstacles. Fig. 5 describes the relationship in joint configuration space. (In the figure, $\dot{\boldsymbol{\Theta}}^{a}$ is the velocity calculated for the robot under motion constraints, and $\dot{\boldsymbol{\Theta}}^{0}$ is the velocity calculated as if the robot is in obstacle free space. They will be used in later discussions). As shown in the figure, from time $t = t_0$ to $t = t_2$ the robot moves in free space. From $t = t_2$ to $t = t_m$, the robot moves in contact with the obstacle, and from $t = t_m$ to $t = t_n$, the robot moves in free space again. At t_n the robot reaches the target.

In the motion, if $\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{U}) = 0$ is satisfied, the robot is in contact with the obstacles. In this case, $\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{U}) = 0$ should be considered in joint trajectory planning so that the motion planning can generate a robot path without violating the kinematic constraints. If relative pure rolling motion is required, the nonholonomic constraints must be added and the trajectory planning must be under both the holonomic and nonholonomic constraints. The challenge for the motion planning is to generate a unified algorithm that generates joint motion under any specified motion constraint (both holonomic and nonholonomic).

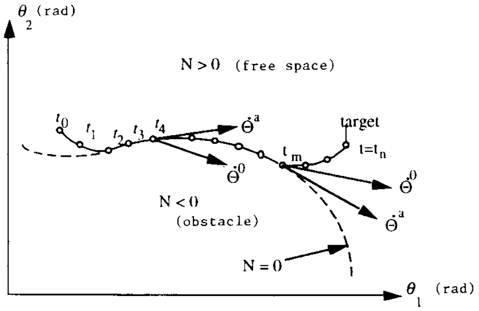


Fig. 5. Three relationships between a robot and an obstacle.

In the discussion below, the same as in Section III, we use

$$H(\boldsymbol{\Theta}, \boldsymbol{U})\dot{\boldsymbol{\Theta}} = 0 \tag{42}$$

to represent any specified motion constraint (both for holonomic and nonholonomic). Note that the holonomic constraints are involved in $\boldsymbol{H}(\boldsymbol{\Theta}, \boldsymbol{U})\dot{\boldsymbol{\Theta}} = 0$ through their differential form (14).

It is noted that (56) can be differentiated and the surface parameter changes, $\dot{\boldsymbol{U}}$, can be derived in terms of $\dot{\boldsymbol{\Theta}}$ as

$$\dot{\boldsymbol{U}} = \boldsymbol{P}(\boldsymbol{\Theta}, \dot{\boldsymbol{U}})\dot{\boldsymbol{\Theta}}. \tag{43}$$

Throughout this section, we assume that the robot is equipped with tactile skin sensors so that the position of contact can be detected.

B. Path Planning

The tasks for obstacle accommodation motion planning are:

1) To make a real-time decision whether: (a) to contact or not to contact the obstacle (if previously not in contact), (b) to remain in contact with the obstacle or to separate from the obstacle (if previously in contact);

2) To formulate the constraints;

3) To find the robot motion trajectory under formulated constraints.

As the first task in motion planning, we provide a method for robot controllers to make the decision. Once a decision is made, we provide an algorithm (based on our inverse kinematics) that provides joint motion for accomplishing assigned tasks. As previously discussed, contacting or not contacting the obstacle gives totally different mathematical models for motion planning. During motion involving contact with obstacles, certain holonomic and nonholonomic constraints need to be imposed on the path planning. If the controller decides not to contact obstacles, motion planning is the same as that for a robot in free space.

In this section, we consider the case in which one obstacle exists in the moving area of a robot. As in Section III, we write the Jacobian of the robot as

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{\Theta}}$$

where

$$\dot{\boldsymbol{x}} \in \boldsymbol{R}^m$$

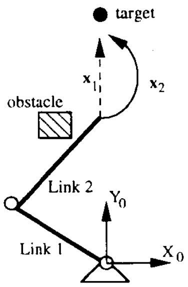


Fig. 6. Two paths for the end-effector.

is the end-effector task and

$$\dot{\boldsymbol{\Theta}} \in \boldsymbol{R}^n$$

is the joint velocity vector. For a wheeled mobile robot, $\boldsymbol{x}$ can be the position and orientation of the robot, and $\dot{\boldsymbol{\Theta}}$ can be the motion control variables such as wheel velocities. The trajectory planning algorithm is as follows.

Step 1: Generate a desired motion path for the end-effector as the robot works in obstacle free space $\dot{\boldsymbol{x}}$. The simplest algorithm for $\dot{\boldsymbol{x}}$ can be

$$\dot{\boldsymbol{x}} = \boldsymbol{k}(\boldsymbol{x}^d - \boldsymbol{x}) \tag{44}$$

where $\boldsymbol{x}^d$ is the target position, $\boldsymbol{x}$ is current end-effector position and orientation. $\boldsymbol{k} \in \boldsymbol{R}^{m \times m}$ is a constant gain matrix. The calculated $\dot{\boldsymbol{x}}$ will be used to find $\dot{\boldsymbol{\Theta}}$ through inverse kinematics.

It is noted that in obstacle free space, (44) generates the path to the target in the shortest distance (straight line). However, when obstacles exist, it may not provide the best choice for finding $\dot{\boldsymbol{\Theta}}$. For example, in Fig. 6, the dashed line represents the trajectory $\boldsymbol{x}_1$, which is generated through (44). It can be seen that the trajectory represented by the solid line $\boldsymbol{x}_2$ may be better than $\boldsymbol{x}_1$ (the straight line) in the sense that it leads to a motion in which the robot has less contact with the obstacle.

Step 2: If the robot is in free space, the joint path is found through (27) for the case in which $n = m$, (29) for the case in which $m > n$, or (32) for the case in which $m < n$.

Step 3: If the robot is in contact with the obstacle, we first calculate the differential changes of the joint variables, $\dot{\boldsymbol{\Theta}}^0$ based on (27), (29), and (32), then check whether $\boldsymbol{\Theta}(t + \Delta t)$ violates the kinematic constraints (here, Δt is the sampling time) by using the holonomic constraints $\boldsymbol{N}(H\Theta, \boldsymbol{U}) = 0$. If

$$\boldsymbol{N}(\boldsymbol{\Theta}(t + \Delta t), \boldsymbol{u}(t + \Delta t)) > 0 \tag{45}$$

motion into free space is allowed for the desired end-effector trajectory $\dot{\boldsymbol{x}}$. Calculated $\dot{\boldsymbol{\Theta}}^0$ is then used to generate the trajectory for the robot motion. Note in (45)

$$\boldsymbol{\Theta}(t + \Delta t) = \boldsymbol{\Theta}(t) + \dot{\boldsymbol{\Theta}}^0 \Delta t$$

$$\boldsymbol{u}(t + \Delta t) = \boldsymbol{u}(t) + \dot{\boldsymbol{u}}^0 \Delta t \tag{46}$$

and

$$\dot{\boldsymbol{u}}^0 = \boldsymbol{P}(\boldsymbol{\Theta}, \boldsymbol{u})\dot{\boldsymbol{\Theta}}^0.$$

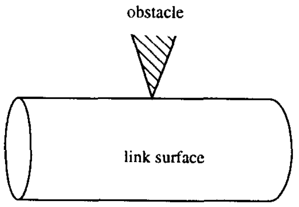


Fig. 7. The pin-tip type of contact.

However, if

$$N(\boldsymbol{\Theta}(t+\Lambda t),\, \boldsymbol{u}(t+\Lambda t)) \leq 0 \tag{47}$$

the trajectory generated by (27), (29), or (32) violates the motion constraints, i.e., the robot can not achieve $\dot{\boldsymbol{x}}$ without contacting the obstacle. In this case the joint trajectory is generated by (35), for the case in which $n < m + r$; for the case in which $n = m + r$, (36) can be used for the inverse kinematics; for the case in which $n > m + r$, (40) is used for solving the inverse kinematics.

Step 4: Update the surface parameter vector $\boldsymbol{u}$. There are two methods to update $\boldsymbol{u}$.

a) Measuring $\boldsymbol{u}$ in real time from contact sensors. If a tactile sensor is equipped, and the contact is assumed to be pin-tip type, then $\boldsymbol{u}$ only consists of the link surface parameters. The surface parameters of contact links can be measured directly.

b) Using (43), we can find the $\dot{\boldsymbol{u}}$ and update $\boldsymbol{u}$ by

$$\boldsymbol{u}(t+\Delta t) = \boldsymbol{u}(t) + \dot{\boldsymbol{u}}\,\Delta t. \tag{48}$$

Step 5: Repeat step 1 till the motion task is fulfilled.

Note that in real applications, we usually do not have accurate knowledge about the obstacle surface. In the calculation, it is easier if we assume that the contact point is the pin-tip type, as shown in Fig. 7. Based on the pin-tip contact assumption, the obstacle surface parameters do not appear in the constraints.

As shown in Fig. 5, we denote the joint velocity developed by (27), (29), or (32) as $\dot{\boldsymbol{\Theta}}^0$ (motion without obstacle contact) and the joint velocity calculated by (35), (36), or (40) as $\dot{\Theta}^a$ (motion with obstacle contact). It is seen from the algorithm that when the robot is in contact with the obstacle, we should first calculate the joint velocities as if the robot is in free space. The reason is that if the motion in free space is allowed, $\boldsymbol{N}(\boldsymbol{\Theta}^0(t+\Delta t), \boldsymbol{u}(t+\Delta t)) > 0$, the robot should move into the free space by selecting $\dot{\boldsymbol{\Theta}} = \dot{\boldsymbol{\Theta}}^0$. The motion in free space usually is more efficient and less likely to cause damage. For example, in Fig. 5, when the robot moves to the point denoted as t_m, the motion toward the target without concerning the obstacle is allowed as $\dot{\boldsymbol{\Theta}}^0$, and the motion command is selected as the robot moves in free space.

C. Illustrative Examples

In this section, we apply the motion planning algorithm to a wheeled mobile robot and a robot manipulator for motion control in environments containing an obstacle.

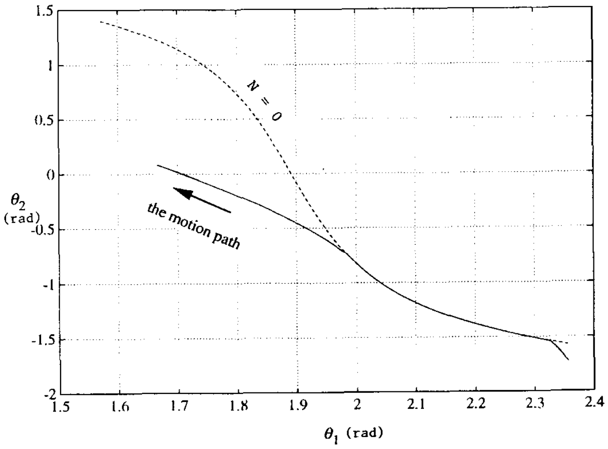


Fig. 8. Simulation results of the path in $\theta_1\theta_2$ configuration space.

Example 1—Path Planning for a 2 DOF Manipulator: If we neglect the width of the links of the robot shown in Fig. 4, the holonomic constraint can be formulated as

$$-p_{bx}\sin(\theta_1+\theta_2) + p_{by}\cos(\theta_1+\theta_2) + L_2\sin(\theta_2) = 0 \tag{49}$$

where $p_{bx} = 0.3536$ and $p_{by} = 1.061$ are the coordinates of the contact point in $\boldsymbol{O}_0$. For simplicity, the contact is assumed to be the pin-tip type. In this example, we generate $\dot{\boldsymbol{x}}$ for the robot to move toward the target in a straight line, i.e.,

$$\dot{\boldsymbol{x}} = \boldsymbol{k}(\boldsymbol{x}^d - \boldsymbol{x})$$

where $\boldsymbol{k}$ is a 2×2 gain matrix. Here $m = 2$, $r = 1$, and $n = 2(m + r > n)$, thus, (35) is used to calculate the constrained joint motion. We only consider the holonomic constraints on the motion, which means that the relative sliding motion between the second link and the obstacle is allowed. Fig. 8 shows a simulation result based on the algorithm developed in this section. In Fig. 8, the solid line is the simulation result of the motion planning; the dashed line is the plot of the holonomic constraint function for the motion of the two link arm due to the obstacle. It can be seen from the figure that, in the first period of time, the robot (link 2) moves close to the obstacle but yet is in free space. Then the robot moves in contact with the obstacle. Finally, the robot moves apart from the obstacle and into the free space again. The motion sequence is depicted in Fig. 9.

Example 2—Path Planning for a Wheeled Mobile Robot: Fig. 10 shows a specific case of Fig. 3, in which the obstacle in contact with the mobile robot has the contour of a cylinder with a radius of r_b. Neglecting the detailed deduction, the holonomic constraints due to the contact are written as

$$\theta + \gamma = \pi + \alpha \tag{50a}$$

and

$$x_0 + r_a\cos(\theta+\gamma) = r_b\cos\alpha$$

$$y_0 + r_a\sin(\theta+\gamma) = r_b\sin\alpha. \tag{50b}$$

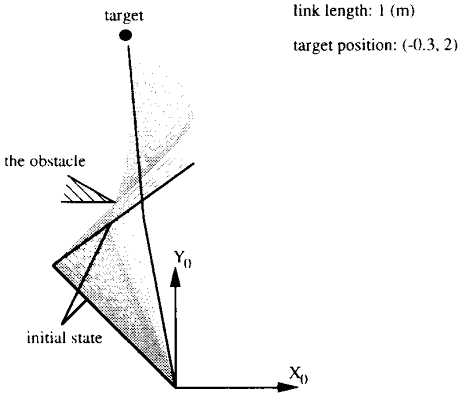


Fig. 9. The motion sequence.

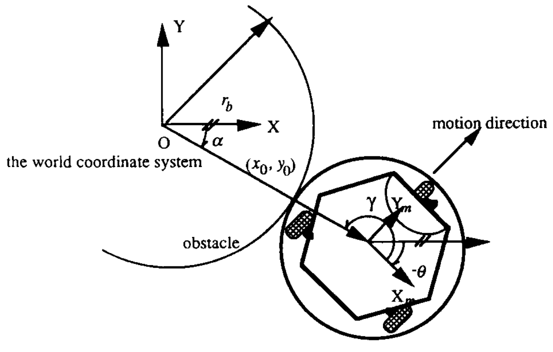


Fig. 10. A mobile robot in contact with a cylindrical obstacle.

Restricting the sliding motion to prevent wear damage on the robot surface (bumper), a nonholonomic constraint can be built based on the factor that the velocity at the contact point is zero. Here, we assume that the robot has an actuator to control bumper rotation.

$$\dot{x}_0 - r_a \sin(\theta + \gamma)\dot{\theta} = 0$$

$$\dot{y}_0 + r_a \cos(\theta + \gamma)\dot{\theta} = 0. \qquad (51)$$

Differentiating (50) yields

$$\dot{\theta} + \dot{\gamma} = \dot{\alpha}$$

$$\dot{x}_0 - r_a \sin(\theta + \gamma)\dot{\theta} - r_a \sin(\theta + \gamma)\dot{\gamma} = -r_b \sin(\alpha)\dot{\alpha}$$

$$\dot{y}_0 + r_a \cos(\theta + \gamma)\dot{\theta} + r_a \cos(\theta + \gamma)\dot{\gamma} = -r_b \cos(\alpha)\dot{\alpha}. \qquad (52)$$

From (51) and (52), we have

$$r_a \sin(\theta + \gamma)\dot{\gamma} = -r_b \sin(\alpha)\dot{\alpha}$$

$$r_a \cos(\theta + \gamma)\dot{\gamma} = r_b \cos(\alpha)\dot{\alpha}. \qquad (53)$$

It can be found that the five equations involved in (52) and (53) are linearly independent. Choosing four independent constraints from the five equations in (52) and (53), we have the constraints for the mobile robot in the motion as

$$\begin{bmatrix} 0 & 0 & 1 & 1 & -1 \\ 1 & 0 & -r_a \sin(\theta + \gamma) & 0 & 0 \\ 0 & 1 & r_a \cos(\theta + \gamma) & 0 & 0 \\ 0 & 0 & 0 & r_a \sin(\theta + \gamma) & r_b \sin(\alpha) \end{bmatrix} \cdot \begin{bmatrix} \dot{x}_0 \\ \dot{y}_0 \\ \dot{\theta} \\ \dot{\gamma} \\ \dot{\alpha} \end{bmatrix} = \mathbf{0} \qquad (54)$$

Fig. 11(a) and Fig. 11(b), show the simulation results of x_0, y_0, and θ for the robot in the processing of moving toward the target. Fig. 12 depicts the motion sequence.

The examples used in this section are based on simplified models and simplified environments. In real applications, the situations will be more complicated. For example, obstacle and link surface properties will be more complicated and motion constraints are generally more difficult. One of our future studies will focus on improving the algorithm and designing a new robot structure to make obstacle accommodation more viable for real-time application.

D. Motion Planning under Multiobstacle Constraints

In many cases, there may be more than one obstacle contacting the robot ($r > 1$). In multiobstacle environments, we use an algorithm similar to that for the robot in contact with a single obstacle except for Step 3. The motion planning algorithm needs to decide in real-time whether: 1) to contact or not to contact certain obstacles (if they are previously not in contact); 2) to remain in contact with certain obstacles or to separate them from (if they are previously in contact); and 3) to contact fewer obstacles or to contact more obstacles. This section will provide an algorithm to make these decisions and provide the corresponding motion path.

In the following, we will use the example shown in Fig. 13 to explain the path planning algorithm for a robot in a multiobstacle environment.

After the end-effector motion, $\dot{\boldsymbol{x}}$, is designed, we determine the joint motion as follows:

First, we calculate the joint motion as the robot is in free space. We denote the solution as $\dot{\boldsymbol{\Theta}}_0^0$. If $\dot{\boldsymbol{\Theta}}_0^0$ does not violate the constraints, the robot can move toward $\dot{\boldsymbol{x}}$ without contacting anything in the next step. We mark the joint motion as $\dot{\boldsymbol{\Theta}}_0^{0^*}$ and use it as the planned motion path for the next motion step.

If $\dot{\boldsymbol{\Theta}}$ violates the motion constraints, the robot has to contact some obstacles in the next motion step (we mark the joint motion as $\dot{\boldsymbol{\Theta}}_0^{0\sim}$). However, we still demand that the robot contact as few obstacle as possible in the next motion step. In this case, we plan the joint motion as follows.

We take off the constraints due to Obstacle 1, and form a new constraint $\boldsymbol{H}_2$, concerning only Obstacle 2. Then, we design the joint motion under the constraints of $\boldsymbol{H}_2$, denote the resulting joint motion as $\dot{\boldsymbol{\Theta}}_2^0$, and if $\dot{\boldsymbol{\Theta}}_2^0$ satisfies the constraints $\boldsymbol{N}_1 > 0$ ($\boldsymbol{N}_1 = 0$ is the holonomic constraints due to Obstacle 1), although the robot is in contact with Obstacle 2, it can

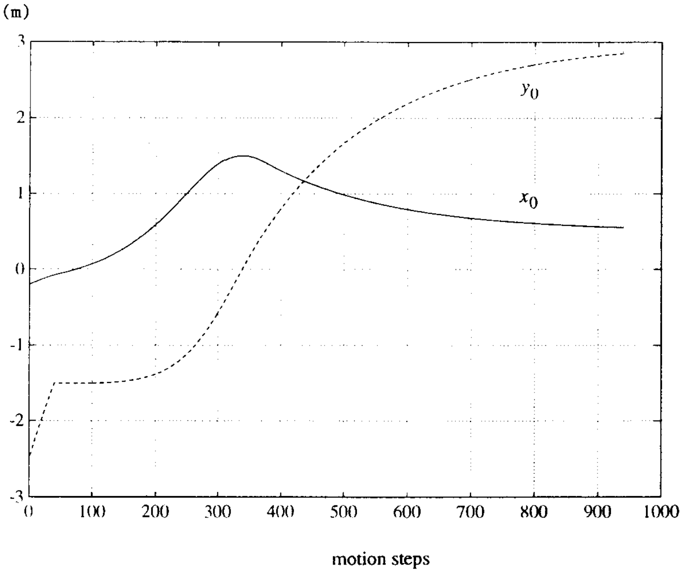


Fig. 11(a). The simulation results of x_0 and y_0.

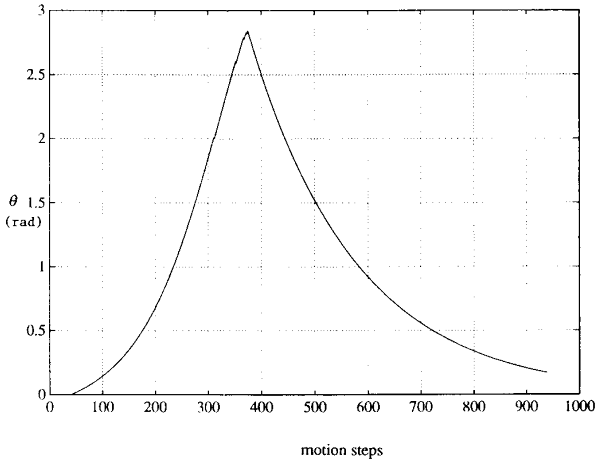


Fig. 11(b). The simulation results of θ.

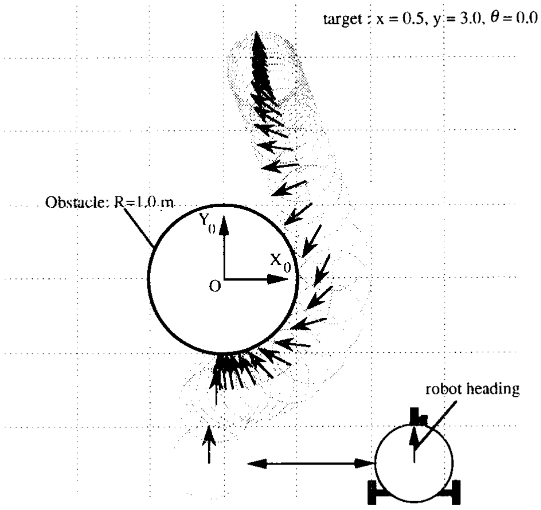


Fig. 12. Motion toward the target.

move without contacting Obstacle 1. In this case, we mark the joint motion as $\dot{\boldsymbol{\Theta}}_2^{0*}$. If $\dot{\boldsymbol{\Theta}}_2^{0}$ violates the holonomic constraints due to Obstacle 1 ($\boldsymbol{N}_1 \leq 0$), we mark it as $\dot{\boldsymbol{\Theta}}_2^{0\sim}$ (that means no motion is possible when the robot is contacting only Obstacle 2).

Similarly, we design the joint motion under constraints due to Obstacle 1 but without considering Obstacle 2 and denote the solution as either $\dot{\boldsymbol{\Theta}}_1^{0^*}$ or $\dot{\boldsymbol{\Theta}}_1^{0\sim}$ depending on whether the solution $\dot{\boldsymbol{\Theta}}_1^{0}$ satisfies $\boldsymbol{N}_2 > 0$ or $\boldsymbol{N}_2 \leq 0$, respectively.

If $\dot{\boldsymbol{\Theta}}_1^{0*}$ or $\dot{\boldsymbol{\Theta}}_2^{0^*}$ exists, then the next motion step can be without touching Obstacle 1 or Obstacle 2. If $\dot{\boldsymbol{\Theta}}_2^{0^*}$ exists, $\dot{\boldsymbol{\Theta}}_2^{0^*}$ is used as the motion planning for the next motion step. If neither $\dot{\boldsymbol{\Theta}}_1^{0^*}$ nor $\dot{\boldsymbol{\Theta}}_2^{0^*}$ exists, the motion has to be in contact with both Obstacles 1 and 2. In this case, the joint motion is designed under constraints $\boldsymbol{H}$, which involves all the obstacle contact.

It is seen that, in designing $\dot{\boldsymbol{\Theta}}_j^{0}$ ($j = 1, 2$) there are three combinations of constraints due to obstacles: Obstacle 1 only, Obstacle 2 only, and both Obstacle 1 and Obstacle 2. For r obstacles, there are total

$$s = \sum_{j=1}^{r} C_r^j \tag{55}$$

number of total combinations of obstacles, where C_r^j is the number of combinations of r for j. Suppose we have s^0 number of solutions, $\dot{\boldsymbol{\Theta}}_j^{0^*}$ (here, $j = 1, \cdots, s^0$) out of s combinations of constraints, the next motion step can be

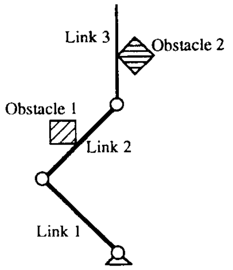


Fig. 13. A robot in contact with two obstacles.

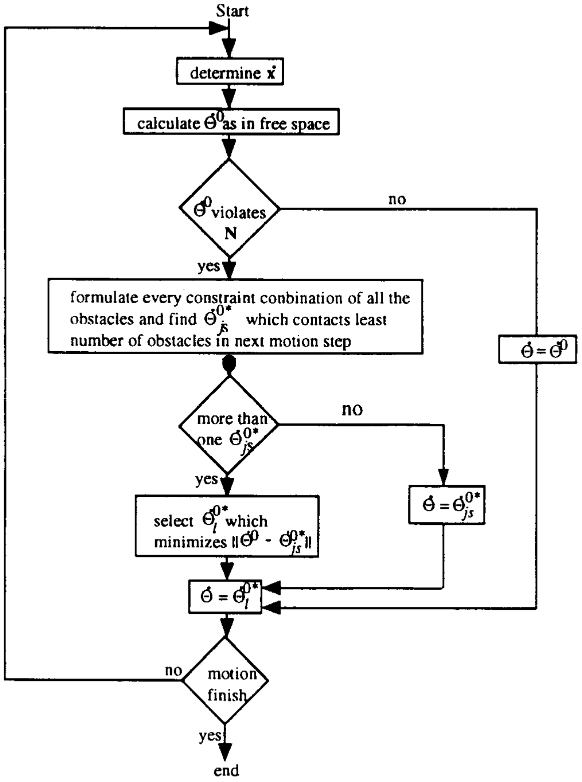

Start
determine ẋ
calculate Θ̇⁰ as in free space
Θ̇⁰ violates N
no
yes
formulate every constraint conbination of all the obstacles and find Θ̇⁰*js which contacts least number of obstacles in next motion step
Θ̇ = Θ̇⁰
more than one Θ̇⁰*js
no
yes
Θ̇ = Θ̇⁰*js
select Θ̇⁰*l which minimizes ||Θ̇⁰ - Θ̇⁰*js||
Θ̇ = Θ̇⁰*l
no
motion finish
yes
end

Fig. 14. The path planning diagram.

without contacting all the obstacles. We select the one in s^0 solutions that contacts the least number of obstacles as the final solution. If there is more than one solution that contacts the least number of obstacles, the path is determined based on the one that is closest to $\dot{\boldsymbol{\Theta}}^0$ (the joint motion designed as in free space), i.e., the one that minimizes the value $|\dot{\boldsymbol{\Theta}}^0 - \dot{\boldsymbol{\Theta}}_j^{0^*}|$. The path planning algorithm in the flow-chart diagram shown in Fig. 14 summarizes the aforementioned discussion.

V. Conclusions

This paper introduces a motion planning methodology for obstacle accommodation. To achieve the motion planning, the paper first developed a general formulation for motion constraints due to obstacles. Two types of motion constraints are discussed: holonomic type and nonholonomic type. The nonholonomic constraints are developed to prevent the robot from wear damage from obstacle contact.

Next, this paper provides a general inverse kinematics solution for robots in contact with obstacles. According to different constraint situations ($m + r < n$, $m + r = n$, and $m + r > n$) different equations of the inverse kinematics need to be selected. The inverse kinematics study is new and important because it is the first to provide a general inverse kinematics for robots in contact with obstacles at unspecified points on unspecified links.

When a robot moves among obstacles, the controller has to generate joint paths according to whether it moves in a free space or in contact with obstacles, as well as according to different motion constraints and then use the corresponding equations to find the inverse kinematic solutions. If the robot moves in free space, then (27), (29), or (32) should be properly selected according to whether $n = m$, $n < m$, or $n > m$, respectively. If the robot moves in contact with obstacles, i.e., $\boldsymbol{N}(\boldsymbol{\Theta}, \boldsymbol{U}) = 0$, the path generator should formulate the motion constraints $\boldsymbol{H}(\boldsymbol{\Theta}, \boldsymbol{U})\dot{\boldsymbol{\Theta}} = 0$ and use (35), (36), or (40) to solve the inverse kinematics for the cases $m + r > n$, $m + r = n$, or $m + r < n$, respectively.

Finally, this paper develops an algorithm for path planning based on motion constraint formulation and inverse kinematics. The task for path planning is to design end-effector motion and to find the consequent joint motion that will allow a robot to achieve motion towards the target without violating motion constraints due to obstacles. During the motion, the robot may move in free space (no contact with obstacles) in one period of time and contact obstacles in another period of time. The number of contacted obstacles may change as well. Therefore, the challenge for the path generator is to take into consideration the changes in contact properties when generating the motion path. Two computer simulation results based on this algorithm are presented. One of the simulations is performed on a two-link robot and the other one is performed on a wheeled mobile robot.

Appendix A
Contact Point Description

In our study, the homogeneous transformation is used to describe the coordinate transformation between the joints. Suppose the homogeneous transformation matrix from $\boldsymbol{O}_{i-1}$ to $\boldsymbol{O}_i$ is $\boldsymbol{T}_{i-1}^i$ [10], [16]

$$\boldsymbol{T}_{i-1}^i = \begin{bmatrix} \boldsymbol{R}_{i-1}^i & \boldsymbol{p}_{i-1}^i \\ 0\ 0\ 0 & 1 \end{bmatrix} \tag{56}$$

then the transformation matrix from $\boldsymbol{O}_0$ to $\boldsymbol{O}_i$ can be expressed as

$$\boldsymbol{T}_0^i = \boldsymbol{T}_0^1 \boldsymbol{T}_1^2 \cdots \boldsymbol{T}_{i-1}^i = \begin{bmatrix} \boldsymbol{R}_0^i & \boldsymbol{p}_0^i \\ 0\ 0\ 0 & 1 \end{bmatrix}. \tag{57}$$

Since $\boldsymbol{O}_i$ is rigidly attached to the ith link, the ith link surface can be described in $\boldsymbol{O}_i$. Suppose the surface of link i is expressed in terms of $\boldsymbol{O}_i$ as

$$\boldsymbol{x}_i = [x_i(u_i, v_i), y_i(u_i, v_i), z_i(u_i, v_i)]^{\mathrm{T}}$$

where x_i, y_i, and z_i are the coordinates of any point on the surface of link i with respect to $\boldsymbol{O}_i$, and where u_i and v_i are the parameters of the surface, then the point can be determined in the world coordinate system $\boldsymbol{O}_0$ as

$$\begin{bmatrix}\boldsymbol{x}_{i0}\\ 1\end{bmatrix} = \begin{bmatrix}\boldsymbol{R}_0^i & \boldsymbol{p}_0^i\\ 0\ 0\ 0 & 1\end{bmatrix}\begin{bmatrix}\boldsymbol{x}_i\\ 1\end{bmatrix} \tag{58}$$

where

$$\boldsymbol{x}_{i0} = [x_{i0},\, y_{i0},\, z_{i0}]^{\mathrm{T}}$$

is the vector from origin of the world coordinate system to the point. According to (3), we have

$$\boldsymbol{x}_{i0} = \boldsymbol{R}_0^i\boldsymbol{x}_i + \boldsymbol{p}_0^i. \tag{1}$$

Similarly, if we assign a coordinate system $\boldsymbol{O}_b$ to the obstacle, then the surface of the obstacle can be expressed in terms of $\boldsymbol{O}_b$ as

$$\boldsymbol{x}_b = [x_b(u_b,\, v_b),\, y_b(u_b,\, v_b),\, z_b(u_b,\, v_b)]^{\mathrm{T}}.$$

Any contact point on the obstacle surface can also be expressed in terms of the world coordinate system as

$$\begin{bmatrix}\boldsymbol{x}_{b0}\\ 1\end{bmatrix} = \boldsymbol{T}_0^b\begin{bmatrix}\boldsymbol{x}_b\\ 1\end{bmatrix} \tag{59}$$

where

$$\boldsymbol{x}_{b0} = [x_{b0},\, y_{b0},\, z_{b0}]^{\mathrm{T}}$$

is the coordinates of the contact point on the obstacle surface in terms of the world coordinate and $\boldsymbol{T}_0^b$ is the homogeneous transformation matrix from the world coordinate system to the obstacle coordinate system $\boldsymbol{O}_b$

$$\boldsymbol{T}_0^b = \begin{bmatrix}\boldsymbol{R}_0^b & \boldsymbol{p}_0^b\\ 0\ 0\ 0 & 1\end{bmatrix}. \tag{60}$$

Equation (6) can also be written as

$$\boldsymbol{x}_{b0} = \boldsymbol{R}_0^b\boldsymbol{x}_b + \boldsymbol{p}_0^b. \tag{3}$$

Appendix B
Derivation of Equations (35) and (40)

When $m + r > n$, the number of task dimensions and the number of motion constraints are greater than the number of system unknowns; therefore, the exact solution does not exist. The problem can be solved by minimizing the least-square error

$$E(\dot{\boldsymbol{\Theta}}) = (\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}})^{\mathrm{T}}(\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}}) \tag{33}$$

subject to

$$\boldsymbol{H}\dot{\boldsymbol{\Theta}} = 0 \tag{34}$$

where $\boldsymbol{H}$ is the motion constraints involving both holonomic and nonholonomic types of constraints. Using Lagrange multipliers, a new objective function can be formed as

$$L(\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}})^{\mathrm{T}}(\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}}) + \lambda^{\mathrm{T}}H \tag{61}$$

where

$$\lambda = [\lambda_1,\, \lambda_2, \cdots, \lambda_r]^{\mathrm{T}}.$$

Expending the equation L yields

$$L = (\boldsymbol{J}\dot{\boldsymbol{\Theta}})^{\mathrm{T}}(\boldsymbol{J}\dot{\boldsymbol{\Theta}}) - 2\dot{\boldsymbol{x}}^{\mathrm{T}}(\boldsymbol{J}\dot{\boldsymbol{\Theta}}) + \dot{\boldsymbol{x}}^{\mathrm{T}}\dot{\boldsymbol{x}} + \lambda^{\mathrm{T}}\boldsymbol{H}\dot{\boldsymbol{\Theta}}. \tag{62}$$

To find the optimal solution, we have to solve the following equation:

$$\frac{\partial L}{\partial\dot{\boldsymbol{\Theta}}} = 2\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J}\dot{\boldsymbol{\Theta}} - 2\boldsymbol{J}^{\mathrm{T}}\dot{\boldsymbol{x}} + \boldsymbol{H}^{\mathrm{T}}\lambda = 0. \tag{63}$$

Assuming that $(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})$ has full rank of n, solving (63), we obtain

$$\dot{\boldsymbol{\Theta}} = (\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\left(\boldsymbol{J}^{\mathrm{T}}\dot{\boldsymbol{x}} - \frac{1}{2}\boldsymbol{H}^{\mathrm{T}}\lambda\right). \tag{64}$$

Substituting (64) into (34), we have

$$\lambda = 2[\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{H}^{\mathrm{T}}]^{-1}\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{J}^{\mathrm{T}}\dot{\boldsymbol{x}}. \tag{65}$$

Here, it is assumed that matrix $\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{H}^{\mathrm{T}}$ has full rank r. From (64) and (65), we obtain the solution

$$\dot{\boldsymbol{\Theta}} = (\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}[\boldsymbol{J}^{\mathrm{T}} - \boldsymbol{H}^{\mathrm{T}}[\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{H}^{\mathrm{T}}]^{-1}\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\boldsymbol{J})^{-1}\boldsymbol{J}^{\mathrm{T}}]\dot{\boldsymbol{x}}. \tag{35}$$

When $m+r < n$, the number of system unknowns is greater than the number of motion task requirements (number of task dimensions) and the number of motion constraints; therefore, the system is undetermined. We solve the inverse kinematics by minimizing the value of

$$L = \dot{\boldsymbol{\Theta}}^{\mathrm{T}}\dot{\boldsymbol{\Theta}} \tag{37}$$

subject to

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{\Theta}} \tag{38}$$

and

$$\boldsymbol{H}\dot{\boldsymbol{\Theta}} = 0. \tag{39}$$

Let

$$\mu = [\mu_1,\, \mu_2, \cdots, \mu_r]^{\mathrm{T}}$$

and

$$\lambda = [\lambda_1,\, \lambda_2, \cdots, \lambda_r]^{\mathrm{T}}$$

be the Lagrange multiplier vectors. The new objective function is

$$L(\dot{\boldsymbol{\Theta}},\, \mu,\, \lambda) = \dot{\boldsymbol{\Theta}}^{\mathrm{T}}\dot{\boldsymbol{\Theta}} + \mu^{\mathrm{T}}(\boldsymbol{J}\dot{\boldsymbol{\Theta}} - \dot{\boldsymbol{x}}) + \lambda^{\mathrm{T}}\boldsymbol{H}\dot{\boldsymbol{\Theta}}. \tag{66}$$

The necessary condition that the optimal solution satisfies is

$$\frac{\partial L}{\partial\dot{\boldsymbol{\Theta}}} = 2\dot{\boldsymbol{\Theta}} + \boldsymbol{J}^{\mathrm{T}}\mu + \boldsymbol{H}^{\mathrm{T}}\lambda = 0. \tag{67}$$

Thus, by solving (67) we have

$$\dot{\boldsymbol{\Theta}} = -\frac{1}{2}(\boldsymbol{J}^{\mathrm{T}}\mu + \boldsymbol{H}^{\mathrm{T}}\lambda). \tag{68}$$

Substituting (68) into (38) and (39) yields

$$\boldsymbol{H}(\boldsymbol{J}^{\mathrm{T}}\mu + \boldsymbol{H}^{\mathrm{T}}\lambda) = 0 \tag{69}$$

and

$$\frac{J(J^{\mathrm{T}}\mu + H^{\mathrm{T}}\lambda)}{2} = -\dot{x}. \qquad (70)$$

Solving for μ and λ from (69) and (70) and substituting them into (68), we have

$$\dot{\theta} = [J^{\mathrm{T}} - H^{\mathrm{T}}(HH^{\mathrm{T}})^{-1}(HJ^{\mathrm{T}})] \cdot \{J[J^{\mathrm{T}} - H^{\mathrm{T}}(HH^{\mathrm{T}})^{-1}(HJ^{\mathrm{T}})]\}^{-1}\dot{x}. \qquad (40)$$

In (40), all the inverse operation are assumed on an invertible matrix.

REFERENCES

[1] J. S. Arora, *Introduction to Optimum Design.* New York: McGraw-Hill, 1989.

[2] H. Asada and J.-J. E. Slotine, *Robot Analysis and Control.* New York: Wiley, 1981.

[3] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst. Man Cybern.*, vol. 19, pp. 1179–1187, Sept. 1989.

[4] C. Cai and B. Roth, "On the spatial motion of rigid bodies with point contact," in *Proc. IEEE Int. Conf. Robotics Automat.*, 1987, pp. 686–695.

[5] A. Cole, J. Hauser, and S. Sastry, "Kinematics and control of a multifingered hand with rolling contact," *IEEE Trans. Automat. Cont.*, vol. 34, no. 4, 1989.

[6] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower pair mechanisms based on matrices," *J. Appl. Mech.*, vol. 22, no. 2, pp. 215–221, 1955.

[7] C. Y. Ho and J. Sriwattanathamma, *Robot Kinematics: Symbolic Automation and Numerical Synthesis.* Ablex Publishing, 1990.

[8] K. H. Hunt, *Kinematic Geometry of Mechanisms.* New York: Oxford University Press, 1978.

[9] Y. Koren, *Robotics for Engineers.* New York: McGraw-Hill, 1985.

[10] J. C. Latombe, *Robot Motion Planning.* Norwell, MA: Kluwer Academic, 1991.

[11] Z. Li and J. Canny, "Motion of two rigid bodies with rolling constraints," *IEEE Trans. Robotics Automat.*, vol. 6, Feb. 1990.

[12] D. G. Luenberger, *Linear and Nonlinear Programming.* Norton, MA: Addison-Wesley, 1984.

[13] M. T. Mason, "Compliance and force control for computer controlled manipulators," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-11, June 1981.

[14] H. Millman and G. D. Parker, *Elements of Differential Geometry.* Englewood Cliffs, NJ: Prentice-Hall, 1978.

[15] D. J. Montana, "The kinematics of contact and grasp," *Int. J. Robotics Res.*, vol. 7, no. 3, pp. 17–32, June 1988.

[16] R. Paul, *Robot Manipulator: Mathematics, Programming, and Control.* Cambridge, MA: M.I.T. Press, 1981.

[17] J. Philips, *Freedom in Machinery—Volume 1: Introducing Screw Theory.* Cambridge, UK: Cambridge University Press, 1984.

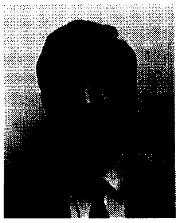
[18] G. R. Pieper, "The kinematics of manipulators under computer control," Ph.D. dissertation, Stanford University, Stanford, CA, 1968.

[19] Y. Shan and Y. Koren, "Obstacle accommodated motion control of a planar manipulator," in *Proc. Int. Conf. Cont. Robotics*, 1992.

[20] Y. Shan, "Obstacle accommodation: Mechanics, control, and applications," Ph.D. dissertation, University of Michigan, Ann Arbor, MI, 1992.

[21] H. West, "Kinematics analysis for the design and control of braced manipulators," Ph.D. dissertation, M.I.T., Cambridge, MA, 1968.

[22] D. E. Whitney, "Quasi-static assembly of compliantly supported rigid parts," *J. Dynamic Syst., Measure., Cont.*, vol. 104, pp. 65–77, Mar. 1982.



Yansong Shan received the B.S.M.E. from Shanghai Jiao-Tong University, China, in 1982, the M.S.M.E. degree from the University of Tokyo, Japan, in 1986, and the Ph.D. degree from the University of Michigan, Ann Arbor, MI, in 1992, all in mechanical engineering.

From 1988 to 1990, he was a development engineer at CardioPulmonic Inc. After finishing his Ph.D., he joined the Department of Mechanical Engineering at the University of Michigan, Ann Arbor, where he is currently conducting research on manufacturing systems, robotics, and man-machine systems.

He was a Rackham Predoctoral Fellow at the University of Michigan during his Ph.D. study and he is a Member of Tau Beta Pi.

Yoram Koren (M'76–SM'88) received the B.Sc. and M.Sc. degrees in electrical engineering and the D.Sc. in mechanical engineering from Technion, Israel Institute of Technology in Haifa, Israel, in 1967 and 1970, respectively.

He is a Professor in the Department of Mechanical Engineering at the University of Michigan, Ann Arbor. He has 25 years of research, teaching, and consulting experience in the automated manufacturing field.

Dr. Koren is the author of more than 120 technical papers and three books, and holds three U.S. patents in robotics. His book, *Computer Control of Manufacturing Systems* (New York: McGraw Hill, 1983) is used as a textbook at major universities and received the 1984 Textbook Award from SME. His book, *Robotics for Engineers* (New York: McGraw Hill, 1985) was translated into Japanese and French and is used by engineers throughout the world. He is a Senior Member of IEEE and a Fellow of ASME, an Active Member of CIRP, and a Fellow of SME/Robotics-International.