Obstacle Avoidance for Redundant Robots Using Configuration Control

R. Colbaugh
Department of Mechanical Engineering, New Mexico State University, Las Cruces, NM 88001

H. Seraji
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109

K. L. Glass
Department of Mechanical Engineering, New Mexico State University, Las Cruces, NM 88001
Received February 12, 1989, accepted July 18, 1989

The article presents a new and simple solution to the obstacle avoidance problem for redundant robots. In the proposed approach, called *configuration control*, the redundancy is utilized to configure the robot so as to satisfy a set of kinematic inequality constraints representing obstacle avoidance, while the end-effector is tracking a desired trajectory. The robot control scheme is very simple, and uses on-line adaptation to eliminate the need for the complex dynamic model and parameter values of the robot. Several simulation results for a four-link planar robot are presented to illustrate the versatility of the approach. These include reaching around a stationary obstacle, simultaneous avoidance of two obstacles, robot reconfiguration to avoid a moving obstacle, and avoidance of rectangular obstacles. The simplicity and computational efficiency of the proposed scheme allows on-line implementation with a high sampling rate for real-time obstacle avoidance in a dynamically varying environment.

本論文は冗長ロボットの障害物回避問題の新しい簡便な解法を述べる。「コンフィグレイション・コントロール」と呼ばれる本提案手法では、エンドエフェクタが目標軌道を追跡中に、障害物回避を表現する運動学的な不等式拘束条件の集合を満たすようロボットを構成するために冗長性が利用される。ロボット制御則は非常に簡単で、複雑な運動モデルやロボットのパラメータ値の必要性をなくするためオンライン適応を用いている。本手法の有効性を示すため4リンク平面ロボットについてのシミュレーション結果が述べられる。これらは動かない障害物への接近、2つの障害物の同時回避、移動する障害物を避けるためのロボット再構成や正方障害物の回避をも含むものである。本手法の簡便さと計算効率によってリアルタイム障害物回避のための高速サンプリングが可能になった。

Journal of Robotic Systems 6(6), 721–744 (1989)
 CCC 0741-2223/89/060721-24$4.00

I. INTRODUCTION

A robot manipulator is defined to be kinematically redundant if it possesses more degrees of freedom than are required to achieve the desired position and orientation of the end-effector. One of the advantages of robot redundancy is the potential to use the "extra" degrees of freedom to maneuver in a congested workspace and avoid collisions with obstacles, where obstacles are defined as objects in the robot workspace. In this article, we consider the problem of controlling a redundant robot so that it closely tracks the desired end-effector trajectory and simultaneously avoids workspace obstacles.

The majority of the work reported to date concerning obstacle avoidance for robot manipulators has dealt with high-level *path planning*, in which the end-effector path is planned *off-line* so as to avoid collision with workspace obstacles. The path planning problem has been studied by Lozano-Perez,[1] Lumelsky,[2] Freund and Hoyer,[3] Gilbert and Johnson,[4] and others, and is not discussed in this article. Alternatively, the obstacle avoidance problem can be solved *on-line* by the robot controller at the low level. Previous approaches to on-line obstacle avoidance have focused on the inverse kinematics portion of the robot control problem. The classical method for solving the inverse kinematics problem for redundant robots is to compute the joint velocities using the pseudoinverse of the Jacobian matrix. The pseudoinverse approach has been applied to the obstacle avoidance problem by many researchers, including Maciejewski and Klein,[5] Nakamura et al.,[6] Kircanski and Vukobratovic,[7] Lovass-Nagy and Schilling,[8] and Walker and Marcus.[9] Baillieul[10] has proposed the "extended Jacobian technique" for solving the inverse kinematics problem, and has applied this technique to obstacle avoidance for a class of planar robots and obstacles. Oh et al[11] have developed a Newton–Raphson numerical procedure for solving the inverse kinematics problem for redundant robots in the presence of obstacles. Sciavicco and Siciliano[12] and Das et al.[13] have derived similar inverse kinematics algorithms for redundant robots that are closed-loop in nature and require only the forward kinematics of the robot, and have applied these algorithms to the obstacle avoidance problem. Finally, Khatib[14] has considered the complete problem of controlling a redundant robot in the presence of obstacles. In Khatib's approach, the redundant robot is controlled directly in Cartesian space using a model-based control law, and obstacle avoidance is achieved using the artificial potential field concept.

In this article, we present a novel approach to obstacle avoidance for redundant robots based on the newly-developed *configuration control* formulation.[15] In this formulation, the obstacle avoidance requirement is represented as a set of kinematic inequality constraints, and the robot control scheme ensures that these constraints are satisfied simultaneously with the desired end-effector motion. The simplicity and computational efficiency of the proposed approach allows real-time obstacle avoidance in a dynamically varying environment.

The article is structured as follows. Section II gives a brief overview of the configuration control scheme for redundant robots. The formulation of the obstacle avoidance problem within the configuration control framework is dis-

cussed fully in Section III. Several simulation examples of obstacle avoidance using a four-link planar robot are presented in Section IV. Section V discusses the results of the article and draws some conclusions.

II. OVERVIEW OF CONFIGURATION CONTROL SCHEME

The robot manipulator under consideration consists of a linkage of rigid bodies with n revolute/prismatic joints. Let $T \in \mathbf{R}^n$ be the vector of joint torques/forces and $\theta \in \mathbf{R}_n$ be the vector of joint rotations/translations. Then the manipulator dynamic model can be derived from Lagrangian mechanics as

$$T = \frac{d}{dt}\left[\frac{\partial L}{\partial \dot{\theta}}(\theta, \dot{\theta})\right] - \frac{\partial L}{\partial \theta}(\theta, \dot{\theta}) \tag{1}$$

$$= H(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + G(\theta) \tag{2}$$

where $L(\cdot) \in \mathbf{R}$ is the Lagrangian of the manipulator, and $H \in \mathbf{R}^{n \times n}$ and $V, G \in \mathbf{R}^n$ are complicated nonlinear functions of θ, $\dot{\theta}$, and the payload. Let $X \in \mathbf{R}^m$ (with $m < n$) define the position and orientation of the end-effector in task space. The relationship between the end-effector coordinate X and the joint coordinate θ can be written as

$$X = f(\theta) \tag{3}$$

where $f: \mathbf{R}^n \rightarrow \mathbf{R}^m$ represents the forward kinematic transformation. Differentiation of (3) with respect to time yields

$$\dot{X} = J_e(\theta)\dot{\theta} \tag{4}$$

where $J_e = \partial f / \partial \theta \in \mathbf{R}^{m \times n}$ is the end-effector Jacobian of the manipulator.

Most of the approaches to redundant robot control proposed to date focus on *local* resolution of redundancy through instantaneous optimization of an objective function using the Jacobian pseudoinverse. A new approach for *global* control of redundant robots is proposed by Seraji.[15]. This approach, termed *configuration control*, prescribes the selection of a generalized coordinate vector $Y \in \mathbf{R}^n$ that is more task-relevant than the joint coordinate vector θ. This configuration vector Y is to be controlled globally across the entire workspace by ensuring that $Y(t)$ tracks a desired trajectory $Y_d(t)$ using the control system shown in Figure 1. The vector Y is defined as:[15]

$$Y = \begin{bmatrix} X \\ \hline Z \end{bmatrix} \tag{5}$$

where $Z \in \mathbf{R}^r$, and $r = n - m$ is the *degree-of-redundancy* of the manipulator.

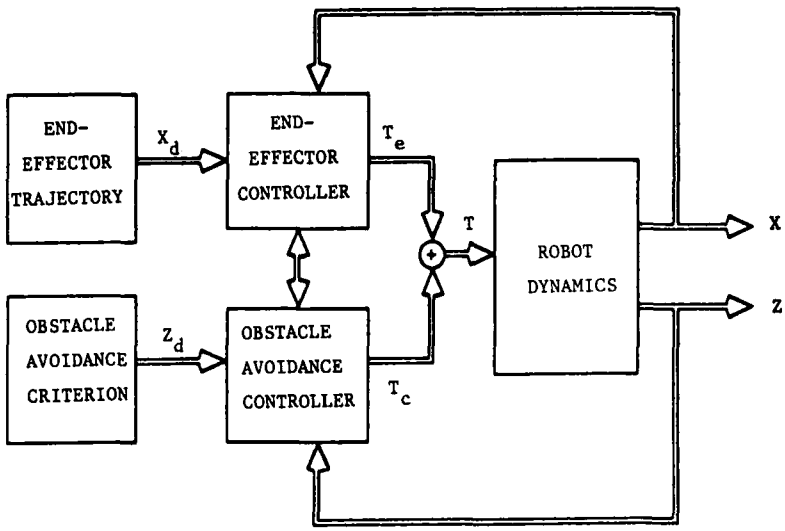

Figure 1. Architecture for configuration control scheme with obstacle avoidance.

The vector Z is to be chosen as

$$Z = g(\theta) \tag{6}$$

where $g: \mathbf{R}^n \to \mathbf{R}^r$ is a kinematic vector function constructed to reflect the performance of some useful additional task. Observe that specifying g and the desired evolution of Y, namely $Y_d'(t) = [X_d'(t) \mid Z_d'(t)]$, defines the "extra" task to be performed *in addition to* the basic task of desired end-effector motion (here prime denotes transposition). The process of selecting the kinematic function (6) is covered adequately in previous articles.[15,16] For now it is simply noted that this approach to global redundancy resolution is quite general, and can be used to impose either equality or inequality constraints, or to specify the optimization of any desired convex kinematic objective function.

The dynamic model of the manipulator can be derived in terms of the configuration vector Y since this vector is a valid generalized coordinate vector for the manipulator. Proceeding as in (1) and (2) yields

$$F = \frac{d}{dt}\left[\frac{\partial L}{\partial \dot{Y}}(Y, \dot{Y})\right] - \frac{\partial L}{\partial Y}(Y, \dot{Y}) \tag{7}$$

$$= H_y(Y)\ddot{Y} + V_y(Y, \dot{Y}) + G_y(Y) \tag{8}$$

where $F \in \mathbf{R}^n$ is the generalized force vector corresponding to the generalized coordinate vector Y, and $H_y \in \mathbf{R}^{n \times n}$ and $V_y, G_y \in \mathbf{R}^n$ are complicated nonlinear functions of Y, $\dot{Y}$, and the payload. Note that $F' = [F_e' \mid F_c']$, where $F_e \in \mathbf{R}^m$ and $F_c \in \mathbf{R}^r$ are the generalized force vectors corresponding to X and Z, respectively.

Different control strategies can be improvised to compute the control force F that ensures that the manipulator configuration vector $Y(t)$ in (8) tracks the desired trajectory $Y_d(t)$, thereby guaranteeing that the basic task of desired end-effector motion and the additional task requirement are achieved simultaneously. Here the adaptive control scheme developed by Seraji[17] will be adopted to accomplish this trajectory tracking. This control scheme does not require any knowledge of the highly complicated dynamic model (8) or parameter values for the manipulator or the payload, and is computationally very fast. Extensive testing of this adaptive controller at both computer simulation and actual implementation levels has demonstrated its effectiveness.[15–18] The centralized control algorithm that computes the control force F is:[17]

$$F = d(t) + K_p(t)E + K_v(t)\dot{E} + C(t)Y_d + B(t)\dot{Y}_d + A(t)\ddot{Y}_d \tag{9}$$

where $E = Y_d - Y$ is the configuration vector tracking-error, and $d \in \mathbf{R}^n$ and K_p, K_v, C, B, $A \in \mathbf{R}^{n\times n}$ are controller gains which are generated on-line in real-time according to the following simple adaptation laws:

$$\begin{aligned}
q &= W_pE + W_v\dot{E} \\
d(t) &= d(0) + k_1\int_0^t q\,dt + h_1q \\
K_p(t) &= K_p(0) + k_2\int_0^t qE'\,dt + h_2qE' \\
K_v(t) &= K_v(0) + k_3\int_0^t q\dot{E}'\,dt + h_3q\dot{E}' \\
C(t) &= C(0) + k_4\int_0^t qY_d'\,dt + h_4qY_d' \\
B(t) &= B(0) + k_5\int_0^t q\dot{Y}_d'\,dt + h_5q\dot{Y}_d' \\
A(t) &= A(0) + k_6\int_0^t q\ddot{Y}_d'\,dt + h_6q\ddot{Y}_d'
\end{aligned} \tag{10}$$

In (10), the k_i and h_i are positive and nonnegative scalar constant adaptation gains, respectively, which are chosen to provide the desired adaptation rates for the controller terms. The constant (usually diagonal) weighting matrices W_p, $W_v \in \mathbf{R}^{n\times n}$ are selected to reflect the relative significance of the individual elements of the tracking-error vectors E and $\dot{E}$. The configuration control scheme can alternatively be implemented in a decentralized structure, where

each configuration variable is controlled independently.[18] Under the control law (9)–(10), the desired end-effector trajectory $X_d(t)$ is tracked, and the "extra" degrees of freedom are appropriately used to control the evolution of the manipulator configuration through the tracking of the desired kinematic trajectory $Z_d(t)$. Note that the control force F is computed entirely based on the observed performance of the manipulator rather than on the manipulator dynamic model (8). The on-line adaptation of the controller using (10) eliminates the need for the complicated mathematical model of the manipulator dynamics. This relieves the designer from the derivation, on-lined computation, and knowledge of parameters of the complicated robot dynamic model (see Section IV for the complex model of a simple planar arm). Furthermore, the simplicity of the proposed control scheme allows the designer to implement very fast control loops, and thereby improve the system performance.

Observe that the control force F computed in (9)–(10) cannot physically be applied to the manipulator; therefore this desired control force must be mapped to an equivalent joint torque vector T. As shown in the first Appendix, the required $F \rightarrow T$ mapping is given by

$$T = J'F = [J_e' \mid J_c'] \begin{bmatrix} F_e \\ \hdashline F_c \end{bmatrix}$$

or

$$T = J_e' F_e + J_c' F_c \tag{11}$$

where $J_c = \partial g / \partial \theta \in \mathbf{R}^{r \times n}$ is the constraint Jacobian and $J' = [J_e' \mid J_c'] \in \mathbf{R}^{n \times n}$ is the augmented Jacobian matrix. This result is to be expected in view of the fact that the manipulator is nonredundant when described with the configuration vector Y.

In the foregoing analysis, the additional task to be performed due to redundancy is formulated as the kinematic equality constraint $g(\theta) = Z_d(t)$. In some applications, such as obstacle avoidance, the additional task is expressed naturally as a set of kinematic inequality constraints

$$g(\theta) \geq 0 \tag{12}$$

These inequality constraints can readily be incorporated into the configuration control scheme. In order to satisfy the inequality constraints (12), we define the reference trajectory $Z_d(t) \equiv 0$ and form the tracking-errors due to these constraints as

$$E_c = 0, \quad \dot{E}_c = 0 \text{ when } g(\theta) \geq 0 \tag{13}$$

$$E_c = -g, \quad \dot{E}_c = -\dot{g} \text{ when } g(\theta) < 0$$

Therefore, in the additional task controller, the feedforward term is omitted

and the feedback control action is computed as

$$F_c = d(t) + K_p(t)E_c + K_v(t)\dot{E}_c \tag{14}$$

where d, K_p, K_v are the adaptive controller terms given in (10) and updated based on the tracking-errors (13). It is important to note that both equality and inequality constraints can exist simultaneously in a given additional task provided the total number of "active" constraints is no greater than the degree-of-redundancy r. Using this formulation, the additional task to be performed by the redundant manipulator can be decomposed into a number of subtasks with different sets of r kinematic constraints, such as different obstacles to be avoided. In the execution of each subtask, the appropriate kinematic constraint is satisfied in addition to the desired end-effector motion.

Since the configuration control scheme involves the $F \rightarrow T$ mapping (11) through the augmented Jacobian matrix J, the singularities of J are of interest. The matrix J will be singular at any joint configuration for which the end-effector Jacobian J_e is rank-deficient. In addition, J will be singular at those values of θ for which the constraint Jacobian J_c loses full rank. The complete analysis of the singularities of J is a subject of current research.[19] It is important to realize that at a Jacobian singularity, no combination of joint movements will produce motion along certain directions in task space, and therefore the effective number of degrees of freedom of the robot is reduced.

Finally, it is interesting to note that when the number of kinematic constraints c is less than r, the configuration control scheme will automatically use the $r - c$ "extra" degrees-of-redundancy to minimize the robot kinetic energy integrated over the entire trajectory. This is proved in the second Appendix, and is a very desirable feature of global optimality in many applications.

III. FORMULATION OF OBSTACLE AVOIDANCE PROBLEM

In this section, we consider the problem of causing the robot to closely track some desired end-effector trajectory while simultaneously ensuring that none of the robot links collide with workspace obstacles. Our approach to this problem is to formulate the obstacle avoidance criterion as a set of kinematic inequality constraints in task space, and then to use the configuration control scheme to ensure that these constraints are satisfied while the desired end-effector trajectory is tracked.

We shall suppose that all workspace obstacles can be enclosed in convex volumes, and that each volume defines a *space of influence* (SOI) for the control law. It is necessary to define each SOI with a sufficient "slack" for the obstacle, so that transient errors of the control system do not damage the obstacle. Selection of SOIs will be discussed in Section IV.

In the theoretical development of this section, it is assumed for convenience that the SOI volumes are spheres in three-dimensional space and disks in two-dimensional space; however, this assumption is not necessary and is relaxed in one of the computer simulation examples in Section IV. It is further assumed

that the location of the center and the radius of each SOI are known. Presumably this information can be provided by some higher level planner with access to appropriate sensory data; e.g., vision data.

The basic strategy that we shall adopt for obstacle avoidance is somewhat similar to the one proposed in Ref. 5; however, our approach to obstacle avoidance is different and more general because we implement this strategy within the configuration control framework. The strategy may be summarized as follows: If any point on the robot enters the SOI of any obstacle, the robot redundancy is utilized to inhibit the motion of that point in the direction toward the obstacle. Note that this strategy is simple and intuitively appealing.

Figure 2 shows a general robot link i of length l_i and a general obstacle j and associated SOI, in three-dimensional space. Define $(X_c)_{ij} \in \mathbf{R}^3$ to be the position of the *critical point* on link i relative to obstacle j (measured in the robot base frame), where the critical point ij is that point on link i currently at a minimum distance from obstacle j. Here $i = 1, 2, \ldots, n$ and we assume that there are k obstacles, so that $j = 1, 2, \ldots, k$. Let $(X_0)_j \in \mathbf{R}^3$ and $(r_0)_j$ denote the position of the center and the radius of obstacle j, respectively. Then, defining $[d_c(\theta)]_{ij} = \|(X_c)_{ij} - (X_0)_j\| = [((X_c)_{ij} - (X_0)_j)'((X_c)_{ij} - (X_0)_j)]^{1/2}$, the criterion for obstacle avoidance may be expressed as a set of inequality constraints:

$$[d_c(\theta)]_{ij} - (r_0)_j \equiv g_{ij}(\theta) \geq 0; \quad i = 1, 2, \ldots, n, \quad j = 1, 2, \ldots, k \tag{15}$$

For a moving obstacle, $(X_0)_j$ and therefore g_{ij} are functions of time, and hence inequality (15) must be satisfied *for all time t*. The constraint ij given by (15) is defined to be *active* if $g_{ij} < 0$, and we assume that no more than r constraints are simultaneously active at any time. If the number of active constraints c is less than r, the configuration control scheme automatically uses the $r - c$ "extra" degrees-of-redundancy to minimize the robot kinetic energy integrated over the

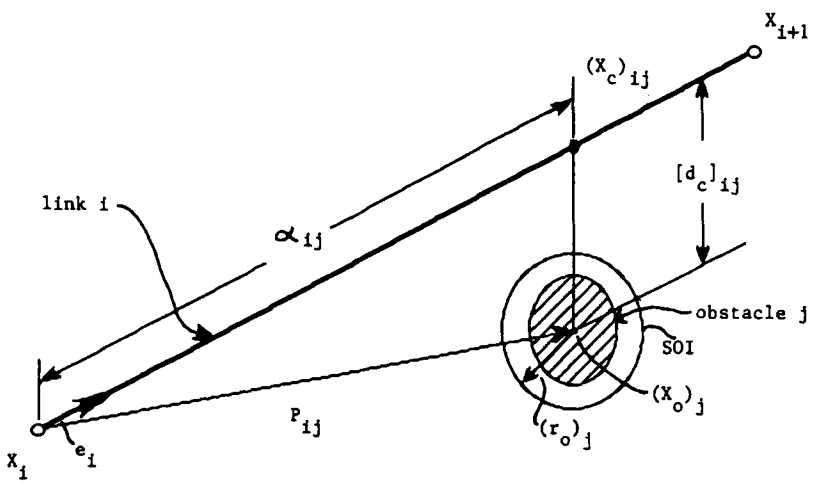

Figure 2. General representation of a robot link-obstacle pair.

entire trajectory (see second Appendix). In the event that more than r constraints are simultaneously active ($c > r$), our policy is to abort the task, because in general there is no safe trajectory solution to the tracking problem in this case. Observe that this does not prevent us from successfully maneuvering in a workspace containing more than r obstacles, provided that only r obstacles are handled at any one time. In view of this policy, we can stack the currently active constraints into the constraint vector

$$g(\theta, t) \geq 0 \tag{16}$$

with $g: \mathbf{R}^n \rightarrow \mathbf{R}^c$ where $c \leq r$ and the explicit time-dependency of g is shown to accommodate moving obstacles. The inequality constraint (16) is identical to (12), and therefore this constraint relationship and the desired end-effector trajectory can be tracked simultaneously using the configuration control law (9)–(14). In order to implement this control scheme, however, an efficient method for locating the active critical points must be developed, and expressions for E_c, $\dot{E}_c$, and J_c must be derived.

Locations of Active Critical Points

In constructing an algorithm to locate active critical points, it is important to note that the locations of the critical points vary during the robot task and must be continually updated. Thus the algorithm used to locate these critical points must be computationally efficient. We proceed by first locating all of the critical points (the points on each link closest to each of the obstacles) and then determining which (if any) of these critical points are within obstacle SOIs and are therefore active.

Referring to Figure 2, we define $X_i \in \mathbf{R}^3$ to be the location of joint i relative to the base frame, $\alpha_{ij} \in \mathbf{R}^+$ to be the distance measured along link i from joint i to critical point ij, $p_{ij} = (X_0)_j - X_i$, and $e_i = (X_{i+1} - X_i)/l_i$. These definitions as shown in Figure 2 may be used to derive the following recursive algorithm for computing the location of all active critical points:

$$\alpha_{ij} = e_i' p_{ij} = \frac{1}{l_i}[X_{i+1} - X_i]'[(X_0)_j - X_i]$$

$$\text{if} \quad \alpha_{ij} \leq 0 \quad \text{then} \quad \alpha_{ij} = 0 \tag{17}$$

$$\text{if} \quad \alpha_{ij} \geq l_i \quad \text{then} \quad \alpha_{ij} = l_i$$

$$(X_c)_{ij} = X_i + \alpha_{ij} e_i \tag{18}$$

$$(d_c)_{ij} = [((X_c)_{ij} - (X_0)_j)'((X_c)_{ij} - (X_0)_j)]^{1/2} \tag{19}$$

$$\text{if} \quad (d_c)_{ij} < (r_0)_j \quad \text{then} \quad (X_c)_{ij} \quad \text{is an active critical point,} \tag{20}$$

otherwise it is not active.

Calculation of E_c, $\dot{E}_c$, and J_c

Having located the active critical points using the algorithm (17)–(20), the constraint vector (16) may be readily constructed from definition (15). The fact that all of the constraints included in the constraint vector (16) are active makes calculating $E_c \in \mathbf{R}^c$ and $\dot{E}_c \in \mathbf{R}^c$ straightforward (i.e., the case $E_{c,i} = 0$ need not be considered):

$$E_c = [E_{c,i}] \quad i = 1, 2, \ldots, c; \quad \text{with} \quad E_{c,i} = (r_0)_i - (d_c)_i \tag{21}$$

$$\dot{E}_c = [\dot{E}_{c,i}] \quad i = 1, 2, \ldots, c; \quad \text{with} \quad \dot{E}_{c,i} = -\dot{g}_i = -\left(\frac{\partial g_i}{\partial \theta}\dot{\theta} + \frac{\partial g_i}{\partial t}\right)$$

$$= -\frac{1}{(d_c)_i}[(X_c)_i - (X_0)_i]'\left[\frac{\partial (X_c)_i}{\partial \theta}\dot{\theta} - (\dot{X}_0)_i\right], \tag{22}$$

where the subscript i refers to element i of g in (16); e.g., $(X_c)_i$ and $(X_0)_i$ are the critical point—obstacle pair corresponding to element i of g, and $(d_c)_i$ is the distance between them. Note that $\dot{g}_i$ is simply the projection of the critical point—obstacle *approach velocity* $(\dot{X}_c)_i - (\dot{X}_0)_i$ onto the unit vector pointing from $(X_0)_i$ to $(X_c)_i$.

The constraint Jacobian matrix $J_c \in \mathbf{R}^{c \times n}$ may be computed row by row through direct differentiation of the elements of g in (16):

$$(J_c)_i = \frac{\partial g_i}{\partial \theta} = \frac{1}{(d_c)_i}[(X_c)_i - (X_0)_i]'\frac{\partial (X_c)_i}{\partial \theta}, \quad i = 1, 2, \ldots, c. \tag{23}$$

The c rows $(J_c)_i$ are then stacked to form J_c. The matrix $J_{X_{ci}} = \partial (X_c)_i / \partial \theta \in \mathbf{R}^{3 \times n}$ is recognized as the Jacobian of the critical point $(X_c)_i$ in the base frame. The marix $J_{X_{ci}}$ can be computed very efficiently for any $(X_c)_i$ once the Jacobians of all of the robot joints are known, as illustrated in Section IV. Therefore the rows of J_c can be efficiently computed as

$$(J_c)_i = \frac{1}{(d_c)_i}[(X_c)_i - (X_0)_i]'J_{X_{ci}}, \quad i = 1, 2, \ldots, c. \tag{24}$$

Note that the general nature of the constraint (16) (i.e., inequality constraint, explicitly time dependent) and Jacobian J_c in (24) do not effect the validity of the $F \rightarrow T$ map (11) in the configuration control scheme, as proved in the first Appendix.

IV. SIMULATION EXAMPLES

The configuration control scheme with obstacle avoidance capability is given in (9)–(14), where the terms E_c, $\dot{E}_c$, and J_c are defined in (21), (22), and (24), respectively. This control scheme will now be applied to a direct-drive four-link

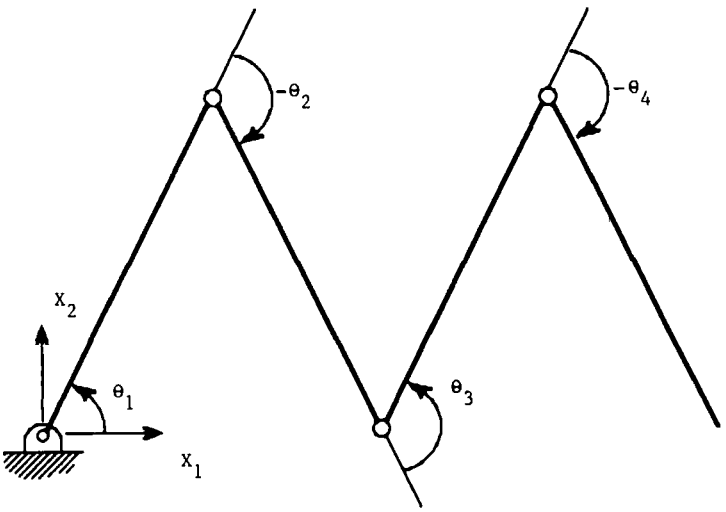

Figure 3. Four-link robot in horizontal plane.

planar robot in a series of computer simulations. The results presented here are samples selected from a comprehensive computer simulation study that is carried out to test the performance of the proposed controller. These examples are chosen for presentation because they illustrate the flexibility and versatility of the configuration control approach to obstacle avoidance.

Consider the four-link robot in a horizontal plane shown in Figure 3. The robot parameters are link lengths $l_1 = l_2 = l_3 = l_4 = 1.0$ meter, link masses $m_1 = m_2 = m_3 = m_4 = 10.0$ kg, and joint viscous friction coefficients $c_1 = c_2 = c_3 = c_4 = 40.0\ \text{Nt} \cdot \text{m/rad} \cdot \text{s}^{-1}$; the link inertias are modeled by thin uniform rods. The robot dynamic equation that relates joint torques $T \in \mathbf{R}^4$ and joint angles $\theta \in \mathbf{R}^4$ is given in Ref. 20.

$$T = H(\theta)\ddot{\theta} + V(\theta, \dot{\theta}) + C\dot{\theta} \tag{25}$$

where the mass matrix $H = [h_{ij}] \in \mathbf{R}^{4\times4}$, Coriolis and centrifugal torque vector $V = [v_i] \in \mathbf{R}^4$, and viscous friction coefficient matrix $C = \text{diag}(c_i) \in \mathbf{R}^{4\times4}$ have the following representations (here $c_2 = \cos\theta_2$, $s_{23} = \sin(\theta_2 + \theta_3)$, etc.):

$$\begin{aligned}
h_{11} &= 73.33 + 50c_2 + 30c_3 + 30c_{23} + 10c_4 + 10c_{34} + 10c_{234} \\
h_{12} &= h_{21} = 40 + 25c_2 + 15c_{23} + 30c_3 + 10c_4 + 10c_{34} + 5c_{234} \\
h_{13} &= h_{31} = 16.67 + 10c_4 + 5c_{34} + 5c_{234} + 15c_3 + 15c_{23} \\
h_{14} &= h_{41} = 3.33 + 5c_4 + 5c_{34} + 5c_{234} \\
h_{22} &= 40 + 30c_3 + 10c_4 + 10c_{34} \\
h_{23} &= h_{32} = 16.67 + 10c_4 + 5c_{34} + 15c_3
\end{aligned} \tag{26}$$

$$h_{24} = h_{42} = 3.33 + 5c_4 + 5c_{34}$$
$$h_{33} = 16.67 + 10c_4$$
$$h_{34} = h_{43} = 3.33 + 5c_4$$
$$h_{44} = 3.33$$

$$\begin{aligned} v_1 = &-50\dot{\theta}_1\dot{\theta}_2 s_2 - 30\dot{\theta}_1\dot{\theta}_3 s_3 - 30\dot{\theta}_1(\dot{\theta}_2 + \dot{\theta}_3)s_{23} \\ &- 10\dot{\theta}_1\dot{\theta}_4 s_4 - 10\dot{\theta}_1(\dot{\theta}_3 + \dot{\theta}_4)s_{34} \\ &- 10\dot{\theta}_1(\dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4)s_{234} - 25\dot{\theta}_2^2 s_2 - 15\dot{\theta}_2(\dot{\theta}_2 + \dot{\theta}_3)s_{23} \\ &- 30\dot{\theta}_2\dot{\theta}_3 s_3 - 10\dot{\theta}_2\dot{\theta}_4 s_4 \\ &- 10\dot{\theta}_2(\dot{\theta}_3 + \dot{\theta}_4)s_{34} - 10\dot{\theta}_2(\dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4)s_{234} \\ &- 10\dot{\theta}_3\dot{\theta}_4 s_4 - 5\dot{\theta}_3(\dot{\theta}_3 + \dot{\theta}_4)s_{34} \\ &- 5\dot{\theta}_3(\dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4)s_{234} - 15\dot{\theta}_3^2 s_3 - 15\dot{\theta}_3(\dot{\theta}_2 + \dot{\theta}_3)s_{23} - 5\dot{\theta}_4^2 s_4 \\ &- 5\dot{\theta}_4(\dot{\theta}_3 + \dot{\theta}_4)s_{34} - 5\dot{\theta}_4(\dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4)s_{234} \end{aligned}$$

$$\begin{aligned} v_2 = &-30\dot{\theta}_1\dot{\theta}_3 s_3 - 10\dot{\theta}_1\dot{\theta}_4 s_4 - 10\dot{\theta}_1(\dot{\theta}_3 + \dot{\theta}_4)s_{34} \\ &- 5\dot{\theta}_1(\dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_4)s_{234} - 30\dot{\theta}_2\dot{\theta}_3 s_3 \\ &- 10\dot{\theta}_2\dot{\theta}_4 s_4 - 10\dot{\theta}_2(\dot{\theta}_3 + \dot{\theta}_4)s_{34} - 10\dot{\theta}_3\dot{\theta}_4 s_4 - 5\dot{\theta}_3(\dot{\theta}_3 + \dot{\theta}_4)s_{34} - 15\dot{\theta}_3^2 s_3 \\ &- 5\dot{\theta}_4^2 s_4 - 5\dot{\theta}_4(\dot{\theta}_3 + \dot{\theta}_4)s_{34} + 25\dot{\theta}_1^2 s_2 + 15\dot{\theta}_2^2 s_{23} + 5\dot{\theta}_1^2 s_{234} \end{aligned}$$

$$\begin{aligned} v_3 = &-10\dot{\theta}_1\dot{\theta}_4 s_4 - 10\dot{\theta}_2\dot{\theta}_4 s_4 - 10\dot{\theta}_3\dot{\theta}_4 s_4 - 5\dot{\theta}_4^2 s_4 + 15\dot{\theta}_1^2 s_3 + 15\dot{\theta}_1^2 s_{23} \\ &+ 5\dot{\theta}_1^2 s_{34} + 5\dot{\theta}_1^2 s_{234} + 15\dot{\theta}_2^2 s_3 + 30\dot{\theta}_1\dot{\theta}_2 s_3 + 5\dot{\theta}_2^2 s_{34} + 10\dot{\theta}_1\dot{\theta}_2 s_{34} \end{aligned}$$

$$\begin{aligned} v_4 = &\, 5\dot{\theta}_1^2 s_4 + 5\dot{\theta}_1^2 s_{34} + 5\dot{\theta}_1^2 s_{234} + 5\dot{\theta}_2^2 s_4 + 5\dot{\theta}_2^2 s_{34} + 5\dot{\theta}_3^2 s_4 \\ &+ 10\dot{\theta}_1\dot{\theta}_2 s_4 + 10\dot{\theta}_1\dot{\theta}_2 s_{34} + 10\dot{\theta}_1\dot{\theta}_3 s_4 + 10\dot{\theta}_2\dot{\theta}_3 s_4 \end{aligned}$$

$$c_1 = c_2 = c_3 = c_4 = 40.0$$

Note that the gravity vector is orthogonal to the plane of motion of the robot, so that no gravity torques appear in (25). It must be emphasized that the dynamic model (25)–(26) is used only to simulate the robot behavior and is *not* used in the control law formulation. The forward kinematics $X = f(\theta)$ and end-effector Jacobian matrix J_e for the robot shown in Figure 3 are

$$\begin{aligned} X_1(t) &= c_1 + c_{12} + c_{123} + c_{1234} \\ X_2(t) &= s_1 + s_{12} + s_{123} + s_{1234} \end{aligned} \tag{27}$$

$$J_e = \begin{bmatrix} -(s_1 + s_{12} + s_{123} + s_{1234}) & -(s_{12} + s_{123} + s_{1234}) & -(s_{123} + s_{1234}) & -s_{1234} \\ (c_2 + c_{12} + c_{123} + c_{1234}) & (c_{12} + c_{123} + c_{1234}) & (c_{123} + c_{1234}) & c_{1234} \end{bmatrix}. \tag{28}$$

The configuration control scheme that is applied to the four-link robot is given in (9)–(14), and is implemented using the control structure shown in Figure 1. The terms E_c and $\dot{E}_c$ are defined in (21), (22), where the active critical point locations and all necessary obstacle avoidance parameters are given by the recursive algorithm (17)–(20). The matrix J_c is computed row by row using (24). Note that for the four-link robot with two end-effector coordinates to be controlled, we have $r = 2$ so that J_c can possess at most two rows and therefore at most two critical points can be active simultaneously. The construction of J_c using (24) requires that $J_{X_{ci}} \in \mathbf{R}^{2\times 4}$ be computed for each active critical point. These base frame Jacobians are readily written once the active critical points are located (here α denotes the distance α_{ij} in (17) associated with the critical point under consideration):

critical point on link 1:

$$J_{X_{ci}} = \begin{bmatrix} -\alpha s_1 & 0 & 0 & 0 \\ \alpha c_1 & 0 & 0 & 0 \end{bmatrix} \tag{29}$$

critical point on link 2:

$$J_{X_{ci}} = \begin{bmatrix} -s_1 - \alpha s_{12} & -\alpha s_{12} & 0 & 0 \\ c_1 + \alpha c_{12} & \alpha c_{12} & 0 & 0 \end{bmatrix} \tag{30}$$

critical point on link 3:

$$J_{X_{ci}} = \begin{bmatrix} -s_1 - s_{12} - \alpha s_{123} & -s_{12} - \alpha s_{123} & -\alpha s_{123} & 0 \\ c_1 + c_{12} + \alpha c_{123} & c_{12} + \alpha c_{123} & \alpha c_{123} & 0 \end{bmatrix} \tag{31}$$

critical point on link 4:

$$J_{X_{ci}} = \begin{bmatrix} -s_1 - s_{12} - s_{123} - \alpha s_{1234} & -s_{12} - s_{123} - \alpha s_{1234} & -s_{123} - \alpha s_{1234} & -\alpha s_{1234} \\ c_1 + c_{12} + c_{123} + \alpha c_{1234} & c_{12} + c_{123} + \alpha c_{1234} & c_{123} + \alpha c_{1234} & \alpha c_{1234} \end{bmatrix}. \tag{32}$$

These $J_{X_{ci}}$ are then used to construct J_c very efficiently using the projection algorithm (24).

In order to demonstrate the effectiveness of the configuration control scheme with obstacle avoidance capability, four different simulation examples will be considered: "wrap" simulation, two obstacle simulation, moving obstacle simulation, and rectangular obstacle simulation. Throughout these simulations, the unit of length is meter, the unit of angle is radian, and the unit of time is second.

'Wrap' Simulation

The task requirements for this simulation are depicted graphically in Figure 4(a), and consist of having the robot end-effector track a circular trajectory of radius 2.4 with center at $X_{\text{center}} = [1.0 \quad 0]'$, while simultaneously avoiding collision with a stationary obstacle. The desired circular end-effector trajectory is quantified as

$$X_d(t) = \begin{bmatrix} 1.0 + 2.4 \sin 0.25t \\ -2.4 \cos 0.25t \end{bmatrix}, \quad t \subset [0, 14].$$

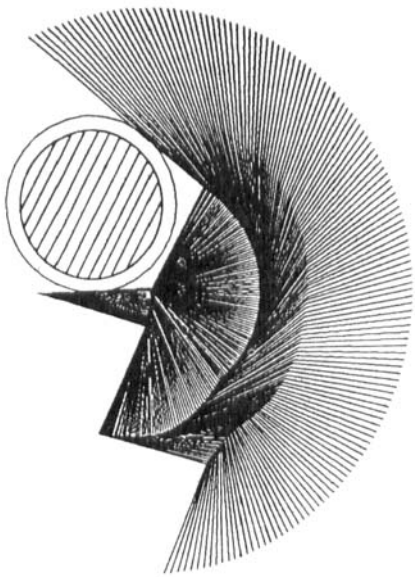

Figure 4(a). Robot configurations in the wrap simulation.

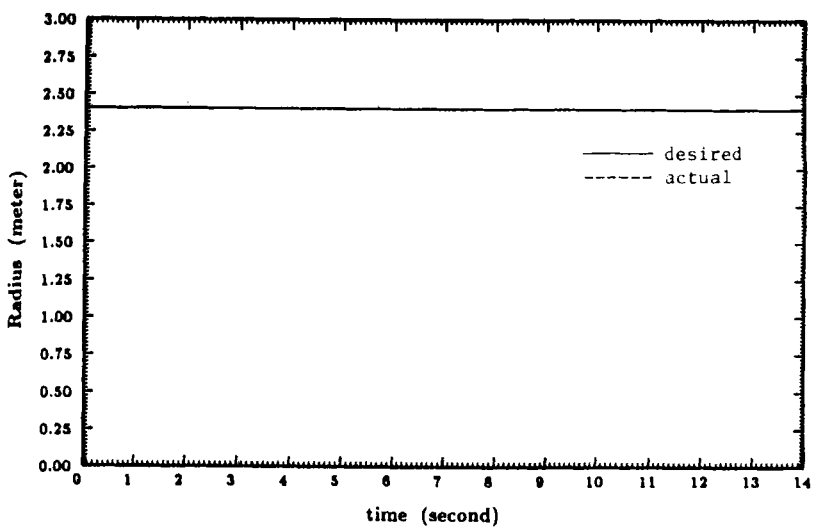

Figure 4(b). Variation of radius of end-effector path in wrap simulation.

The obstacle is a disk of radius 0.65 with center at $X_0 = [0.50 \quad 0.75]'$. The obstacle is assigned an SOI radius of $r_0 = 0.75$ to provide "slack" for the controller, as described in Section III. Note that here the SOI radius is selected heuristically, and that many approaches for specifying this quantity can be proposed. For example, it may be appropriate to define the SOI radius to be a function of the robot link—obstacle approach velocity. The initially configuration of the robot for this simulation is $\theta(0) = [-0.313 \quad -1.73 \quad 1.73 \quad -1.73]'$, and the robot is initially at rest.

End-effector trajectory tracking and obstacle avoidance are achieved simultaneously using the configuration control structure shown in Figure 1. The end-effector position $X(t)$ is controlled using the algorithm (9)–(11). The controller gains d, K_p, K_v, C, B, and A are all set to zero initially. The adaptation gains h_i and k_i are assigned the following values: $h_i = 0$, for $i = 1, 2, \ldots, 6$, $k_1 = k_4 = k_5 = k_6 = 0.5$, $k_2 = k_3 = 2.0$. The weighting matrices are chosen as $W_p = \text{diag}(6000)$, $W_v = \text{diag}(700)$.

The inequality constraint (16) corresponding to the obstacle avoidance criterion is satisfied using the inequality constraint control law (12)–(14) together with the $F \rightarrow T$ map (11). The controller gains are all set to zero initially, and the adaptation gains are assigned the following values: $h_i = 0$, for $i = 1, 2, \ldots, 6$, $k_1 = k_2 = k_3 = 2.0$, $k_4 = k_5 = k_6 = 0$. The weighting matrices are defined as $W_p = \text{diag}(3000)$, $W_v = \text{diag}(350)$. The terms E_c, $\dot{E}_c$, and J_c are computed as specified in (21), (22), and (24), respectively. Note that the inequality constraint (16) is of dimension zero, one, or two, at different times in the simulation depending on whether the obstacle SOI is colliding with no, one, or two links of the robot, respectively. The dimension of the inequality constraint control input F_c in (14) must change according to the dimension change in constraint (16), so that at various times during the trajectory the control input F_c is zero, a scalar, or a two-dimensional vector. Note that when the dimension of F_c is zero or one, the configuration control scheme automatically uses the remaining degrees-of-redundancy to minimize the robot kinetic energy integrated over the trajectory.

The control law is applied to the robot dynamic model (25)–(26) through computer simulation on a SUN 3/50 computer with a sampling period of one millisecond; all integrals required by the controller are implemented using a simple trapezoidal integration rule with a time step of one millisecond. The results of this simulation are given in Figures 4(a) and 4(b), and indicate that the desired end-effector trajectory is closely tracked and the obstacle is successfully avoided.

Two Obstacle Simulation

The task requirements for this simulation are depicted graphically in Figure 5(a), and consist of having the robot end-effector track a straight-line trajectory of length 1.2 while simultaneously avoiding collisions with two stationary obstacles. The desired end-effector trajectory is $X_d(t) = [2.6 - 0.6 \cos 0.5t \quad 0]'$ for $t \subset [0, 2\pi]$. The two obstacles are disks, the smaller one having a radius of 0.07 and center location $(X_0)_1 = [0.375 \quad \sqrt{3}/4]'$ and the larger one having a radius

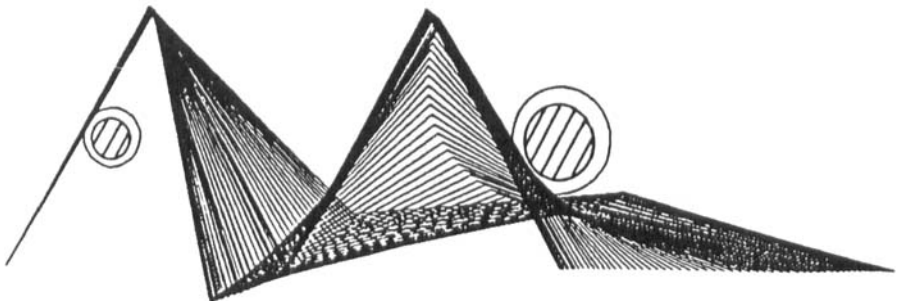

Figure 5(a). Robot configurations in the two obstacle simulation.

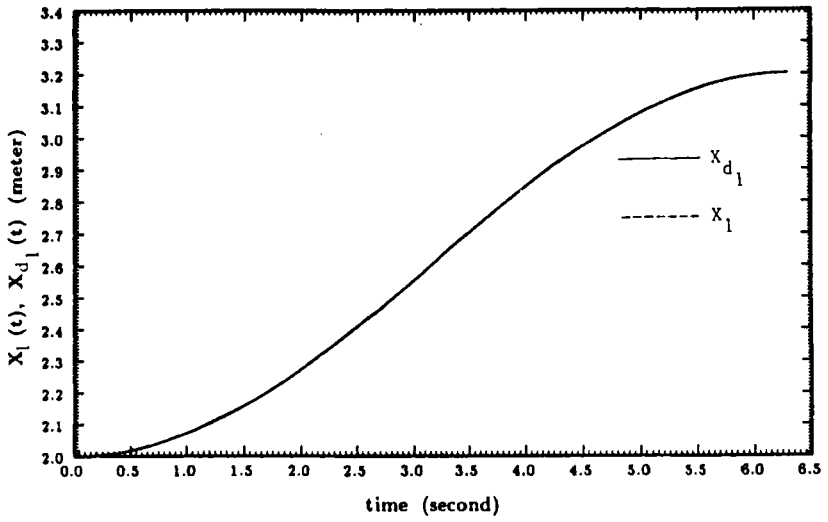

Figure 5(b). End-Effector coordinate X_1 in the two obstacle simulation.

of 0.125 and a center location $(X_0)_2 = [2.0 \quad \sqrt{3}/4]'$. The smaller obstacle is assigned an SOI radius of $(r_0)_1 = 0.1$ while the larger obstacle is assigned an SOI radius of $(r_0)_2 = 0.175$. The initial configuration of the robot for this simulation is $\theta(0) = [\pi/3 \quad -2\pi/3 \quad 2\pi/3 \quad -2\pi/3]'$, and the robot is initially at rest. The control law is implemented exactly as described in the previous simulation. The results of the simulation are given in Figures 5(a) and 5(b), and indicate that the desired end-effector trajectory is closely tracked and both obstacles are successfully avoided throughout the motion.

Moving Obstacle Simulation

The task requirements for the simulation are depicted graphically in Figure 6(a), and consist of having the robot maintain its end-effector at the initial

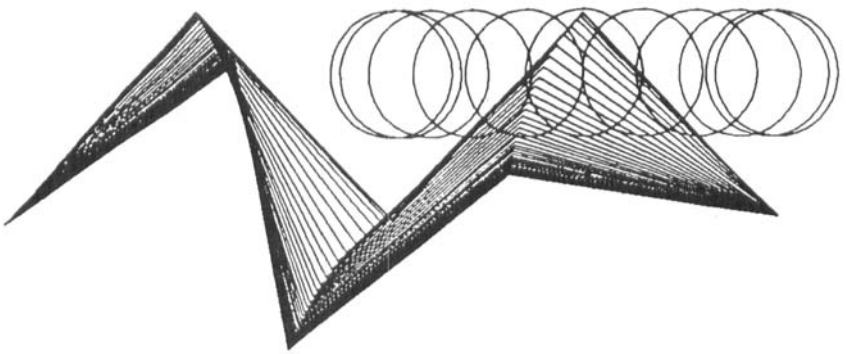

Figure 6(a). Robot configurations in the moving obstacle simulation.

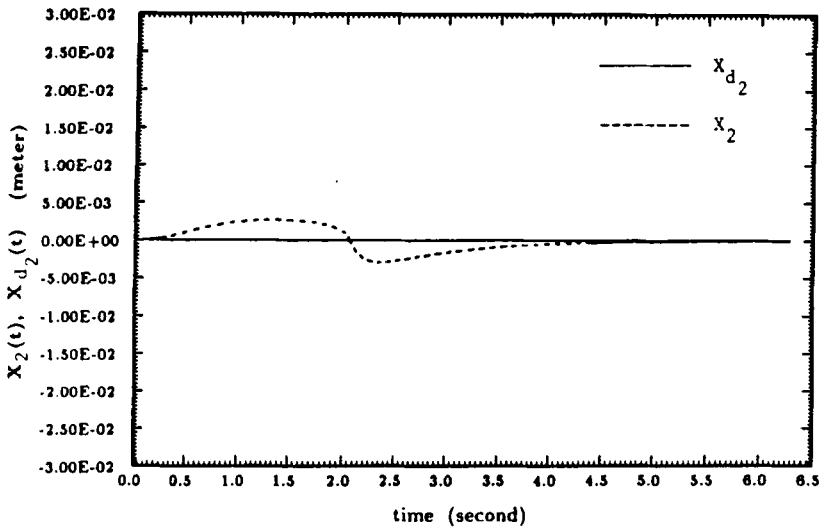

Figure 6(b). End-effector coordinate X_2 in the moving obstacle simulation.

location of $X_d(t) = [2\sqrt{2} \quad 0]'$ for $t \subset [0, 2\pi]$ while avoiding a moving obstacle. The obstacle is a disk of radius 0.2 and center location defined by the trajectory $X_0(t) = [(\sqrt{2}/2)(3 + \cos 0.5t) \quad 0.5]'$; thus the obstacle is initially close to link four of the robot and subsequently moves toward the robot base. The obstacle is assigned an SOI radius of $r_0 = 0.225$ (for clarity of presentation only the SOI and not the actual obstacle is shown in Figure 6(a)). The initial configuration of the robot for this simulation is $\theta(0) = [\pi/4 \quad -\pi/2 \quad \pi/2 \quad -\pi/2]'$, and the robot is initially at rest. The control law is implemented exactly as described in the first simulation. The results of this simulation are given in Figures 6(a) and 6(b), and indicate that the desired end-effector position is maintained and the moving obstacle is successfully avoided.

Rectangular Obstacle Simulation

The task requirements for this simulation are depicted graphically in Figure 7(a), and consist of having the robot end-effector track a vertical straight-line trajectory and come into contact with a rectangular obstacle, while simultaneously avoiding collision between this obstacle and any of the robot links. The desired end-effector trajectory is $X_d(t) = [2\sqrt{2} \quad 0.45 - 0.45\cos 0.5t]'$ for $t \subset [0, 2\pi]$. The obstacle is a semi-infinite rectangle with corner point A having the location $X_A = [\sqrt{2} \quad 0.9]'$. Note that this obstacle is assigned a "boundary layer" SOI of thickness 0.05 instead of a circular SOI, and for clarity of presentation this boundary layer is not shown. The initial configuration of the

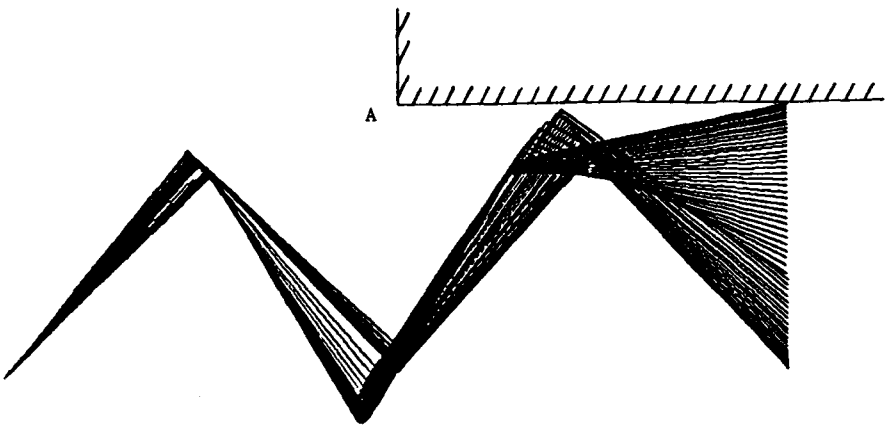

Figure 7(a). Robot configurations in the rectangular obstacle simulation.

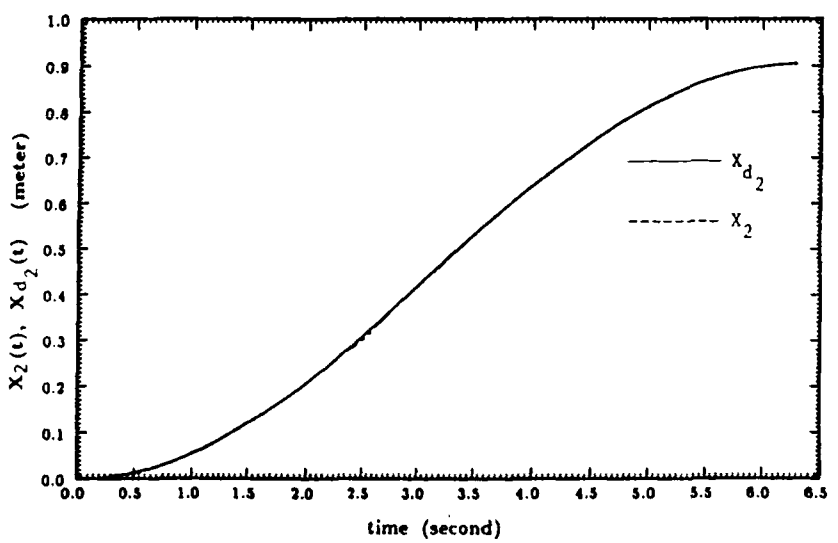

Figure 7(b). End-effector coordinate X_2 in the rectangular obstacle simulation.

robot is $\theta(0) = [\pi/4 \quad -\pi/2 \quad \pi/2 \quad -\pi/2]'$, and the robot is initially at rest. The control law is implemented exactly as described in the first simulation. The results of the simulation are given in Figures 7(a) and 7(b), and indicate that the desired end-effector trajectory is tracked closely and the rectangular obstacle is successfully avoided. Note that the actual trajectory deviates slightly from the desired trajectory at $t = 2.5$ due to the inequality constraint becoming active at that time.

In this example, we have demonstrated the obstacle avoidance capability in a variety of simulations by using a simple control scheme. The control scheme does not use the highly complicated dynamic model of the robot (25)–(26) and does not require knowledge of the robot dynamic parameters.

V. CONCLUSIONS

A simple and direct method for obstacle avoidance of redundant robots is presented in this article. The scheme is developed within the configuration control framework, whereby a set of kinematic inequality constraints representing obstacle avoidance are satisfied, while the end-effector is tracking a desired trajectory. In contrast to previous methods, the configuration control scheme which achieves end-effector motion and obstacle avoidance does not require either the complex dynamic model or the complicated inverse kinematics transformation of the robot. The method is illustrated by extensive computer simulation studies.

The proposed control scheme can be implemented at the low-level servo control loops of the robot, in contrast to previous approaches to obstacle avoidance at the high-level task planning stage. This provides the capability of avoiding obstacles in a dynamically varying environment, where *a priori* task planning is not feasible. The simplicity and computational efficiency of the proposed scheme allows real-time implementation of control loops with high sampling rates, yielding improved dynamic performance.

The research described in this article is supported in part through contracts 06-5630 and 05-9859 with Sandia National Laboratories and in part through a contract with the National Aeronautics and Space Administration through the Goddard Space Flight Center in support of the Flight Telerobotic Servicer Project.

APPENDIX: PROOF THAT $T = J_a' F_a$ IS VALID FOR GENERAL KINEMATIC CONSTRAINTS

In this Appendix, we show that the $F \rightarrow T$ map $T = J_a' F_a$ is a valid one for any control input F_a corresponding to the simultaneous tracking of a general end-effector trajectory and a general kinematic constraint.

Let $c \leq r$ general kinematic equality and/or inequality constraints be currently active, and denote these constraints as $Z(t) = g(\theta, t)$. Construct the generalized

coordinate vector $Y \in \mathbf{R}^n$ as follows:

$$\begin{aligned} Y(t) &= [X'(t) \mid Z'(t) \mid q'(t)]' \\ &= [X_a'(t) \mid q'(t)]' \end{aligned} \tag{A1}$$

where $X_a = [X' \mid Z']'$ as before and $q \in \mathbf{R}^{r-c}$ is any additional coordinate vector which makes Y a valid generalized coordinate vector for the robot. Let the forward kinematic relationship between Y and θ be defined by f_y: $\mathbf{R}^n \rightarrow \mathbf{R}^n$ as

$$Y(t) = f_y(\theta) \tag{A2}$$

Let $F = [F_a' \mid F_q']'$ be the generalized force vector corresponding to the generalized coordinate vector Y, where it is understood that $F_q \equiv 0$ since no control action is generated for the coordinate vector q. From Lagrangian dynamics,

$$T = \frac{d}{dt}\left[\frac{\partial L}{\partial \dot{\theta}}\right] - \frac{\partial L}{\partial \theta} \tag{A3}$$

$$F = \frac{d}{dt}\left[\frac{\partial L_y}{\partial \dot{Y}}\right] - \frac{\partial L_y}{\partial Y} \tag{A4}$$

where L, $L_y \in \mathbf{R}$ are the Lagrangians of the robot written in terms of θ and Y, respectively. Now $L = L_y$ because the Lagrangian is invariant under coordinate transformation. Thus, rewriting (A3) using L_y yields:

$$T = \frac{d}{dt}\left[\frac{\partial L_y}{\partial Y}\frac{\partial Y}{\partial \dot{\theta}} + \frac{\partial L_y}{\partial \dot{Y}}\frac{\partial \dot{Y}}{\partial \dot{\theta}}\right] - \left[\frac{\partial L_y}{\partial Y}\frac{\partial Y}{\partial \theta} + \frac{\partial L_y}{\partial \dot{Y}}\frac{\partial \dot{Y}}{\partial \theta}\right] \tag{A5}$$

Noting that, from (A2),

$$\dot{Y} = \frac{\partial f_y}{\partial \theta}\dot{\theta} + \frac{\partial f_y}{\partial t} = J\dot{\theta} + \frac{\partial f_y}{\partial t}$$

where $J = \partial f_y/\partial\theta = [(\partial X_a/\partial\theta)' \mid (\partial q/\partial\theta)']' = [J_a' \mid J_q']'$, allows (A5) to be written as

$$T = J'\left[\frac{d}{dt}\frac{\partial L_y}{\partial \dot{Y}} - \frac{\partial L_y}{\partial Y}\right] + \left[\frac{d}{dt}\left(\frac{\partial f_y}{\partial \theta}\right)' - \frac{\partial}{\partial \theta}\left(\frac{\partial f_y}{\partial \theta}\dot{\theta} + \frac{\partial f_y}{\partial t}\right)'\right]\frac{\partial L_y}{\partial \dot{Y}} \tag{A6}$$

Finally, (A6) reduces to

$$T = J'F = [J_a' \mid J_q']\begin{bmatrix} F_a \\ \hline 0 \end{bmatrix} = J_a'F_a$$

because of (A4) and the following identity

$$\frac{d}{dt}\left(\frac{\partial f_y}{\partial \theta}\right) - \frac{\partial}{\partial \theta}\left(\frac{\partial f_y}{\partial \theta}\dot{\theta} + \frac{\partial f}{\partial t}\right) = 0 \tag{A7}$$

This completes the proof. Note that the result (A7) may be verified by considering first the case in which $f_y \in \mathbf{R}$, and then proceeding to the vector case by induction.

APPENDIX: PROOF OF GLOBAL MINIMIZATION OF ROBOT KINETIC ENERGY

In this Appendix, we show that in the configuration control approach, the redundant degrees of freedom which are not explicitly controlled in the task space are *automatically* used to globally minimize the robot kinetic energy.

Let $c < r$ inequality constraints be active during some time interval $t \subset [0, t_f]$, and let the corresponding constraint relationships be used to augment the robot task space as prescribed in the configuration control approach. Denote the augmented forward kinematics and differential kinematics as

$$X_a = f_a(\theta)$$

$$\dot{X}_a = J_a(\theta)\dot{\theta}$$

where $X_a \in \mathbf{R}^{m+c}$, $J_a = \partial f_a / \partial \theta \in \mathbf{R}^{(m+c)\times n}$, and $f_a: \mathbf{R}^n \to \mathbf{R}^{m+c}$. In the configuration control scheme, a control input $F_a \in \mathbf{R}^{m+c}$ is constructed to track the desired configuration trajectory $(X_a)_d$, and this control input is applied to the robot by mapping it to an equivalent joint torque vector T as

$$T = J_a' F_a \tag{A8}$$

Note that this map is derived in the first Appendix and corresponds to the case in which $F_q \equiv 0$, since the degrees of freedom represented by $q \in \mathbf{R}^{r-c}$ are not explicitly controlled. Here we show that the $F \to T$ map (A8) satisfies the necessary condition for optimizing the following constrained problem for the case in which gravity and friction effects are negligible:

$$\text{minimize} \int_0^{t_f} \frac{1}{2}\dot{\theta}' H \dot{\theta}\, dt \tag{A9}$$

$$\text{subject to the constraint } X_a - f_a(\theta) = 0$$

In other words, the $F \to T$ map (A8) prescribed in the configuration control approach satisfies a necessary condition for minimizing robot kinetic energy integrated over the task time interval, subject to the requirement that the

desired forward kinematics relationship be maintained. Thus the map (A8) is the minimum kinetic energy map for any F_a provided that $F_q \equiv 0$. It is noted that if gravity and friction effects are not negligible, the map (A8) provides an approximation to the minimum kinetic energy trajectory. A thorough discussion of the effects of gravity and friction on the map (A8) is presented in Ref. 21.

The optimization problem (A9) may be analyzed using the calculus of variations.[22] First, the intermediate function $\psi(\theta, \dot{\theta}, \lambda) \in \mathbf{R}$ is constructed as

$$\psi = \frac{1}{2}\dot{\theta}' H \dot{\theta} + \lambda'[X_a - f_a(\theta)] \tag{A10}$$

where $\lambda \in \mathbf{R}^{m+c}$ is the Lagrange multiplier vector. The necessary conditions on (A10) for optimality of (A9) are

$$\frac{\partial \psi}{\partial \lambda} = 0, \qquad \frac{\partial \psi}{\partial \theta} - \frac{d}{dt}\left(\frac{\partial \psi}{\partial \dot{\theta}}\right) = 0 \tag{A11}$$

Substituting (A10) into the necessary conditions (A11) yields, after some simplification

$$\ddot{X}_a - J_a\ddot{\theta} - \dot{J}_a\dot{\theta} = 0$$

$$H\ddot{\theta} + \dot{H}\dot{\theta} - J_a'\lambda - \frac{1}{2}\partial(\dot{\theta}' H \dot{\theta})/\partial\theta = 0 \tag{A12}$$

Equations (A12) may be solved for $\ddot{\theta}$ as[21]

$$\ddot{\theta} = H^{-1}J_a'(J_aH^{-1}J_a')^{-1}[\ddot{X}_a - \dot{J}_a\dot{\theta}] - [I_n - H^{-1}J_a'(J_aH^{-1}J_a')^{-1}J_a]H^{-1}V \tag{A13}$$

where $I_n \in \mathbf{R}^{n\times n}$ is the identity matrix. Note that the solution (A13) to the problem (A9) has been obtained, independently, by Kazerounian and Wang.[23]

The necessary condition (A13) for optimization of (A9) can be expressed as an $F \rightarrow T$ map by using the joint-space and Cartesian-space dynamic models summarized below:

$$T = H\ddot{\theta} + V + G \tag{A14}$$

$$F_a = (J_aH^{-1}J_a')^{-1}[\ddot{X}_a - \dot{J}_a\dot{\theta}] + (J_aH^{-1}J_a')^{-1}J_aH^{-1}[V + G]$$

where $G \in \mathbf{R}^n$ represents the torques due to gravity and friction effects. Substituting the dynamic models (A14) into the result (A13) yields, after some simplification, the minimum kinetic energy $F \rightarrow T$ map as

$$T = J_a'F_a + [I - J_a'(J_aH^{-1}J_a')^{-1}J_aH^{-1}]G \tag{A15}$$

Finally, the condition that gravity and friction effects are negligible gives $G \approx 0$ so that (A15) reduces to (A8). This completes the proof.

References

1. T. Lozano-Perez, "Automatic planning of manipulator transfer movements," *IEEE Trans. Systems, Man, and Cybernetics*, **SMC-11**, 681–698 (1981).
2. V. J. Lumelsky, "Effect of kinematics on motion planning for planar robot arms moving amidst unknown obstacles," *IEEE Journal of Robotics and Automation*, **RA-3**, 207–223 (1987).
3. E. Freund and H. Hoyer, "Collision avoidance for industrial robots with arbitrary motion," *Journal of Robotic Systems*, **1**, 317–329 (1984).
4. E. G. Gilbert and D. W. Johnson, "Distance functions and their application to robot path planning in the presence of obstacles," *IEEE Journal of Robotics and Automation*, **RA-1**, 21–29 (1985).
5. A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *The Intern. Journal of Robotics Research*, **4**, 109–117 (1985).
6. Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulators," *The Intern. Journal of Robotics Research*, **6**, 3–15 (1987).
7. M. Kircanski and M. Vukobratovic, "Contribution to control of redundant robotic manipulators in an environment with obstacles," *The Intern. Journal of Robotics Research*, **5**, 112–119 (1986).
8. V. Lovass-Nagy and R. Schilling, "Control of Kinematically Redundant Robots Using {1}-Inverses," *IEEE Trans. Systems, Man, and Cybernetics*, **SMC-17**, 4, 644–649 (1987).
9. I. Walker and S. Marcus, "Subtask performance for redundancy resolution for redundant robot manipulators," *IEEE Journal of Robotics and Automation*, **4**, 350–354 (1988).
10. J. Baillieul, "Avoiding obstacles and resolving kinematic redundancy," in *Proc. IEEE Intern. Conf. on Robotics and Automation*, San Francisco, CA., April 1986, pp. 1698–1704.
11. S. Oh, D. Orin, and M. Bach, "An inverse kinematic solution for kinematically redundant robot manipulators," *Journal of Robotic Systems*, **1**, 235–249 (1987).
12. L. Sciavicco and B. Siciliano, "A solution algorithm to the inverse kinematic problem for redundant manipulators," *IEEE Journal of Robotics and Automation*, **4**, 403–410 (1988).
13. H. Das, J-J. Slotine, and T. Sheridan, "Inverse kinematic algorithms for redundant systems," in *Proc. IEEE Intern. Conf. on Robotics and Automation*, Philadelphia, PA. April 1988, pp. 43–48.
14. O. Khatib, "Real-time obstacle avoidance for manipulators and mobile robots," *The Intern. Journal of Robotics Research*, **5**, 90–98 (1986).
15. H. Seraji, "Configuration control of redundant manipulators: Theory and implementation," *IEEE Trans. on Robotics and Automation*, **5**, 472–490 (1989).
16. H. Seraji, R. Colbaugh, K. Glass, and T. Lee, "Case studies in configuration control for redundant robots," in *Proc. 28th IEEE Conference on Decision and Control*, Tampa, FL., December 1989.
17. H. Seraji, "Direct adaptive control of manipulators in cartesian space," *Journal of Robotic Systems*, **4**, 157–178 (1987).
18. H. Seraji, "Decentralized adaptive control of manipulators: Theory, simulation, and experimentation," *IEEE Trans. on Robotics and Automation*, **5**, 183–201 (1989).
19. J. Burdick and H. Seraji, "Characterization and control of self-motions in redund-

ant manipulators," in *Proc. NASA Conference on Space Telerobotics*, Pasadena, CA., January 1989.
20. R. Colbaugh and K. Glass, "Adaptive cartesian position/force control of redundant robots," *Final Report for Sandia Contract Number 06-5630*, New Mexico State University, Engineering Research Center, November 1988.
21. R. Colbaugh, "Dynamic performance optimization of redundant robot manipulators," *Preprints 842nd Meeting of the American Mathematical Society*, Las Cruces, NM, April 1988.
22. D. E. Kirk, *Optimal Control Theory*, Prentice-Hall, Englewood Cliffs, NJ, 1970.
23. K. Kazerounian and Z. Wang, "Global versus Local Optimization in Redundancy Resolution of Robotic Manipulators," *The Intern. Journal of Robotics Research*, **7**, 3–12 (1988).