ROBOT KINEMATICS

In the analysis of robotic systems, including mechanical manipulators and mobile and space robots, we have to deal with the motion of (1) the parts (links) that constitute the robot and (2) the objects that constitute the robot's environment. The science that studies the motion of a robotic system without considering the forces/torques that produce it is called *kinematics*. On the other hand, the science that studies the forces/torques required to produce motion is called *dynamics*.

Robot kinematics is studied on different levels. On the first, *static* level, we are only concerned with relative positioning and orientation and not with velocities and accelerations. In this case, two major problems have been studied, mainly for serial-link manipulators: forward kinematics and inverse kinematics. To define these two terms formally, certain mathematical tools are given next.

MATRICES

Link *A* Matrices

Fixed-based serial-link rigid robot manipulators consist of a sequence of links connected by joints. A joint variable q_i is associated with each joint i. For revolute and prismatic (extensible) joints, the joint variables are an angle (in degrees) and a length, respectively. The joint vector of an n-link manipulator is defined as $q = [q_1\, q_2\, \cdots\, q_n]^T \in \Re^n$. A robot with n joints has n degrees of freedom (i.e., n independent position variables).

To describe the position and orientation of the manipulator in space, we affix a coordinate frame to each link. The *base frame* is attached to the manipulator base, link 0. The free end of the manipulator is the *end effector*. The *tool frame* is attached to the end effector. The *Danavit–Hartenberg* (DH) convention (1) is commonly used to locate the coordinate frame on the link. Thus, the orientation and translation of link i with respect to link $i - 1$ is specified by the following homogeneous transformation:

$$A_i(q_i) = \begin{bmatrix} R_i & p_1 \\ 0 & 1 \end{bmatrix} \tag{1}$$

where $R_i(q_i)$ is a 3×3 rotation matrix ($R_i^{-1} = R_i^T$) and $p_i(q_i) = [x_i\, y_i\, z_i]^T \in \Re^3$ is a translation vector. R_i and p_i specify the rotation matrix and translation vector of the frame on link i with respect to frame on link $i-1$, respectively. The matrices A_i for a specific manipulator are given by the manufacturer.

Robot *T* Matrix

The position and orientation of the end effector with respect to the base frame are given by the robot T matrix

$$T(q) = A_1(q_1)A_2(q_2)\cdots A_n(q_n) \equiv \begin{bmatrix} R(q) & p(q) \\ 0 & 1 \end{bmatrix} \tag{2}$$

where $R(q) = R_1(q_1)R_2(q_2) \cdots R_n(q_n)$ is the cumulative rotation matrix and $p(q) = [x\, y\, z]^T$ is the Cartesian position of the end effector with respect to the base frame.

Now we are in position to define the robot forward kinematics and inverse kinematics. The former is concerned with computing the Cartesian position and orientation of the end effector (i.e., tool frame) relative to the base frame once a set of joint variables is given. That is, for a given q we compute $T(q)$. On the other hand, the inverse kinematic problem is concerned with computing all possible sets of joint variables that locate the end effector at a prescribed Cartesian position and orientation. Since the kinematic expressions are nonlinear, finding $q \in \Re^n$ given $T(q) \in \Re^{4\times4}$ is not an easy task. Inverse kinematics is an important problem from a practical point of view, and an intensive area of current research.

For most applications, we need to consider a robot in motion. In this case, we deal not only with position but also with time derivatives of the joint variables, mainly velocity and acceleration. An important question arises—how to describe the relationship between velocities $\dot{q}(t)$ in joint space and velocities in $\dot{y}(t)$ Cartesian space. Let a nonlinear transformation from the joint variable $q \in \Re^n$ to another variable $y \in \Re^p$ be given by

$$y(t) = h(q(t)) \tag{3}$$

Differentiating yields

$$\dot{y} = \frac{\partial h}{\partial q}\dot{q} \equiv \mathbf{J}(q)q \tag{4}$$

where the mapping $\mathbf{J}(q)$ from joint-space velocities to Cartesian-space velocities is called the *manipulator Jacobian,* provided that $y(t)$ represents the Cartesian position of the end effector.

Since robot manipulators and mobile robots are intended to perform certain predefined tasks and interact with objects (a workpiece, another robot, obstacles, etc.) in their environment, their motions are constrained to a subset of the set of attainable positions, velocities, and accelerations. We shall focus on these kinematically constrained robotic systems, in particular carlike mobile robots. The contacts between the robot wheels and the ground introduce *nonholonomic* effects such as the impossibility of sideways motion. An example of this behavior is an automobile where the wheels can only roll and spin, but not slide sideways. Nevertheless, we can park an automobile at any desired position and orientation.

ROBOT CONSTRAINTS

We consider a rigid robot M with generalized joint variables $q = [q_1\ q_2\ \cdots\ q_n]^{\mathrm{T}} \in Q \subseteq \Re^n$ moving in a workspace Ω. In the robotic literature Q is called the robot *configuration space.* Suppose that k independent constraints of the form $a_i(q,t) = 0$, $i = 1, 2, \ldots, k$, apply to the motion of M. Grouping these independent scalar constraints in a matrix yields

$$[a_1(q,t) \quad \cdots \quad a_k(q,t)]^{\mathrm{T}} \equiv \mathbf{A}(q,t) = 0 \tag{5}$$

Equation (5) can be used to reduce the order of the configuration space to an $(n - k)$-submanifold of Q. This type of constraint is called a *holonomic* or *integrable* constraint. Obstacles in the robot workspace can be represented as inequality holonomic constraints, that is, $a_i(q, t) \le 0$.

Another kinematic constraint has the form

$$a_i(q,\dot{q},t) = 0 \tag{6}$$

If the kinematic constraint (6) can be expressed as Eq. (5), it is a holonomic or integrable constraint. Otherwise, the constraint is said to be *nonintegrable* or *nonholonomic*. If there are k independent nonholonomic constraints of the form Eq. (6), the space of attainable velocities $\dot{q} \in V_Q$ is reduced to an $(n - k)$-dimensional subspace without changing the dimension of the configuration space Q. Finally, the system may have kinematic inequality constraints of the form $a_i(q, \dot{q}, t) \le 0$. A bounded steering angle of an automobile is a typical kinematic inequality constraint.

In this article we shall assume that all k kinematic equality constraints are independent of time, and can be expressed as

$$\boldsymbol{A}(q)\dot{q} = 0 \tag{7}$$

Robotic systems subject to kinematic constraints are studied by Barraquand and Latombe (2), and Murray et al. (3), among others.

NONHOLONOMIC MOBILE ROBOT

Wheeled vehicles and carlike mobile robots are typical examples of nonholonomic mechanical systems. Many researchers treat the problem of motion under nonholonomic constraints using the kinematic model of a mobile robot, and assuming no disturbances and known dynamics. This simplified representation does not correspond to the reality of a moving vehicle, which has unknown mass, friction, drive train compliance, and backlash effects. In this section we provide a framework that brings the two approaches together: nonholonomic control results that deal with a kinematic steering system, and full servo-level feedback control that takes into account the mobile robot dynamics.

The navigation problem is classified into three basic problems by Canudas de Wit et al. (4): tracking a reference trajectory, following a path, and point stabilization. Some nonlinear feedback controllers have been proposed in the literature [e.g., Kanayama et al. (5)] for solving the first problem. The main idea behind these algorithms is to design velocity control inputs which stabilize the closed-loop system. A reference cart generates the trajectory that the mobile robot is supposed to follow. In path following, as in the previous case, we need to design velocity control inputs that stabilize a carlike mobile robot in a given xy geometric path. The hardest problem is stabilization about a desired posture. One way to solve this problem is given by Samson (6), where the velocity control inputs are time-varying functions. All these controllers consider only the kinematic model (e.g., steering system) of the mobile robot, and perfect velocity tracking is assumed to generate the actual vehicle control inputs.

We need a controller structure that takes into account the specific vehicle dynamics. First, feedback velocity control inputs are designed for the kinematic steering system to make the position error asymptotically stable. Second, a feedback velocity-following control law is designed such that the mobile robot's velocities converge asymptotically to the given velocity inputs. Finally, this second control signal is used by the computed-torque feedback controller to compute the required torques for the actual mobile robot. This control approach can be applied to a class of *smooth* kinematic system control velocity inputs. Therefore, the same design procedure works for all of the three basic navigation problems.

Kinematics and Dynamics of a Mobile Platform

The dynamical model of a mobile robot is given by

$$\mathbf{M}(q)\ddot{q} + \mathbf{V_m}(q,\dot{q})\dot{q} + \mathbf{F}(\dot{q}) + \mathbf{G}(q) + \tau_{\mathrm{d}} = \mathbf{B}(q)\tau - \mathbf{A}^{\mathrm{T}}(q)\lambda \tag{8}$$

where $q \in \Re^3$ is the position and orientation vector, $\mathbf{M}$ is a symmetric, positive definite inertia matrix, $\mathbf{V_m}$ is a centripetal and Coriolis matrix, $\mathbf{F}$ is a friction vector, $\mathbf{G}$ is a gravity vector, τ_{d} is a vector of disturbances including unmodeled dynamics, $\mathbf{B}$ is an input transformation matrix, τ is a control input vector, $\mathbf{A}$ is a matrix associated with the constraints, and λ is a vector of constraint forces. The dynamics of the driving and steering motors should be included in the robot dynamics, along with any gearing.

Assume there are k independent nonholonomic constraints of the form (7). Let $\mathbf{S}(q)$ be a full-rank $(n - k)$ matrix (formed by a set of smooth and linearly independent vector fields spanning the null space of $\mathbf{A}(q)$, that is,

$$\mathbf{S}^{\mathrm{T}}(q)\mathbf{A}^{\mathrm{T}}(q) = 0 \tag{9}$$

According to Eq. (7) and Eq. (9), it is possible to find an auxiliary vector time function $v \in \Re^{n-k}$ such that, for all t,

$$\dot{q} = \mathbf{S}(q)v(t) \tag{10}$$

In fact, $v(t)$ often has physical meaning, consisting of two components—the commanded vehicle linear velocity $v_{\mathrm{L}}(t)$, and the angular velocity $\omega(t)$ or heading angle θ. The matrix $\mathbf{S}(q)$ is easily determined independently of the dynamics Eq. (8) from the wheel configuration of the mobile robot. Thus, Eq. (10) is the kinematic equation that expresses the constraints on $\dot{q}(t)$ in terms of the velocity vector $v(t) = [v_{\mathrm{L}}\ \omega]^{\mathrm{T}}$. It does not include dynamical effects, and is known in the nonholonomic literature as the *steering system*. In the case of omnidirectional vehicles, $\mathbf{S}(q)$ is 3×3 and Eq. (10) corresponds to the Newton's law $F = ma$.

The nonholonomic mobile platform shown in Fig. 1 consists of a vehicle with two driving wheels mounted on the same axis, and a passive front wheel. The motion and orientation are achieved by independent actuators (e.g., dc motors) providing the necessary torques to the driving wheels. Another common configuration uses the front wheel for driving and steering.

The position of the robot in an inertial Cartesian frame {O, X, Y} is completely specified by the vector $q = [x\ y\ \theta]$ where (x, y) and θ are the coordinates of the reference point C and the orientation of the basis $\{C, X_C, Y_C\}$ with respect to the inertial basis, respectively. Additionally, Fig. 1 shows the parameters of the mobile base that will be used to develop a mathematical model of the vehicle:

m = mass of the mobile base, including the driving wheels and the dc motors
I = moment of inertia of the mobile base about a vertical axis through C
$2R$ = distance between the driving wheels
r = radius of the driving wheels
P = intersection of the wheel axis with the axis of symmetry
C = reference point in the mobile base
d = distance between P and C

The nonholonomic constraint states that the robot can only move in the direction normal to the axis of the driving wheels, that is, the mobile base satisfies the condition of *pure rolling* [see Sarkar et al. (7)], yielding the kinematic constraint

$$\dot{y}\cos\theta - \dot{x}\sin\theta - d\dot{\theta} = 0 \tag{11}$$

It is easy to verify that $\mathbf{S}(q)$ is given by

$$\mathbf{S}(q) = \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \\ 0 & 1 \end{bmatrix} \tag{12}$$

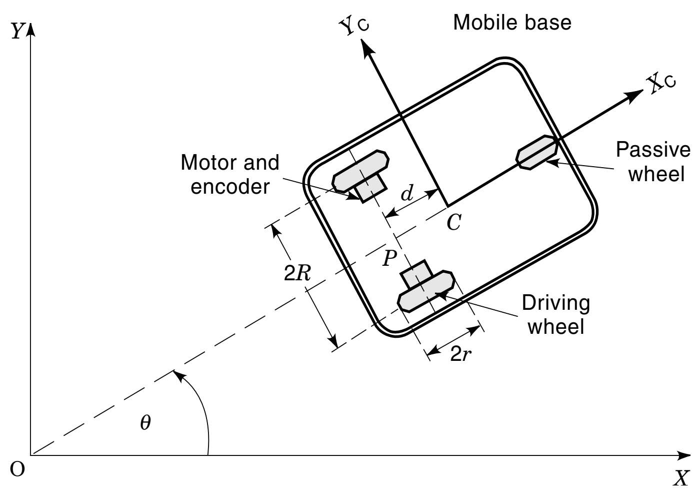

Figure 1. A top view of a differential drive nonholonomic mobile platform. Two dc motors provide the required torques to drive the mobile robot in a two-dimensional space.

Using the above expressions, it is possible to derive the forward kinematics of the mobile base. The kinematic equations of motion of C are

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta & -d\sin\theta \\ \sin\theta & d\cos\theta \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v_L \\ \omega \end{bmatrix} \tag{13}$$

where $|v_L| \leq V_M$ and $|\omega| \leq W_M$, with V_M and W_M the maximum linear and angular velocities of the mobile robot.

For the case of nonholonomic systems, the number of states is greater than the number of control inputs. Therefore, the Jacobian matrix $\mathbf{S}(q)$ is not a square matrix, and it is not possible to find its direct inverse. However, the inverse kinematics can be approximately solved by using the pseudoinverse matrix of the Jacobian; see Zhao and BeMent (8):

$$v = \mathbf{S}^{+}(q)\dot{q} = [\mathbf{S}^{T}(q)\mathbf{S}(q)]^{-1}\mathbf{S}^{T}(q)\dot{q} \tag{14}$$

where

$$\mathbf{S}^{+}(q) = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ \dfrac{-d\sin\theta}{d^2+1} & \dfrac{d\cos\theta}{d^2+1} & \dfrac{1}{d^2+1} \end{bmatrix} \tag{15}$$

Under the assumption that the driving wheels do not slip and the angular displacement of each driving wheel is measured, we can compute the heading angle by using the following relation (7):

$$\theta = \frac{r}{2R}(\theta_r - \theta_l) \tag{16}$$

where θ_r and θ_l are the angular displacements of the right and left driving wheels respectively. The Cartesian position of the mobile robot can be estimated by integrating Eq. (13), where $q_0 = [x_0\ y0\ \theta_0]^T$ is the vector of initial positions:

$$q = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ \theta_0 \end{bmatrix} + \begin{bmatrix} \int_{t_0}^{t} (v_L\cos\theta - \omega d\sin\theta)\,dt \\ \int_{t_0}^{t} (v_L\sin\theta + \omega d\cos\theta)\,dt \\ \int_{t_0}^{t} \omega\,dt \end{bmatrix} \tag{17}$$

The Lagrange formalism is used to derive the dynamic equations of the mobile robot. In this case $\mathbf{G}(q) = 0$, because the trajectory of the mobile base is constrained to the horizontal plane, so that its potential energy U remains constant. The kinetic energy K_E [see for instance Lewis et al. (9)] is given by

$$k_E^i = \tfrac{1}{2}m_i v_i^T v_i + \tfrac{1}{2}\omega_i^T I_i \omega_i, \qquad K_E \equiv \sum_{i=1}^{n_l} k_E^i = \tfrac{1}{2}\dot{q}^T\mathbf{M}(q)\dot{q} \tag{18}$$

The Lagrangian of the mobile platform is given by

$$\begin{aligned} L(q,\dot{q}) &= K_E(q,\dot{q}) - U_{ref} \\ L(q,\dot{q}) &= \tfrac{1}{2}m[(\dot{x} + d\dot{\theta}\sin\theta)^2 + (\dot{y} - d\dot{\theta}\cos\theta)^2] + \tfrac{1}{2}\ {}^{P}I\dot{\theta}^2 \\ L(q,\dot{q}) &= \tfrac{1}{2}m\dot{x}^2 + \tfrac{1}{2}m\dot{y}^2 + m\dot{x}d\dot{\theta}\sin\theta - m\dot{y}d\dot{\theta}\cos\theta + \tfrac{1}{2}I\dot{\theta}^2 \\ I &= {}^{P}I + md^2 \end{aligned} \tag{19}$$

The equations of motion of the nonholonomic mobile base are then given by

$$
\begin{aligned}
\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} &= \tau - \mathbf{A}^{\mathrm{T}}\lambda \\
m\ddot{x} + md(\ddot{\theta}\sin\theta + \dot{\theta}^2\cos\theta) &= \frac{1}{r}(\tau_{\mathrm{r}} + \tau_{\mathrm{l}})\cos\theta + \lambda\sin\theta \\
m\ddot{y} - md(\ddot{\theta}\cos\theta - \dot{\theta}^2\sin\theta) &= \frac{1}{r}(\tau_{\mathrm{r}} + \tau_{\mathrm{l}})\sin\theta - \lambda\cos\theta \\
md(\ddot{x}\sin\theta - \ddot{y}\cos\theta) + I\ddot{\theta} &= \frac{R}{r}(\tau_{\mathrm{r}} - \tau_{\mathrm{l}}) + \lambda d
\end{aligned}
\tag{20}
$$

where τ_{r} and τ_{l} are the torques applied to the right and left driving wheels respectively, and λ is the Lagrange multiplier. Equation (20) can be expressed in the matrix form Eq. (8) where

$$
\begin{aligned}
\mathbf{M}(q) &= \begin{bmatrix} m & 0 & md\sin\theta \\ 0 & m & -md\cos\theta \\ md\sin\theta & -md\cos q & I \end{bmatrix} \\
\mathbf{V_m}(q,\dot{q}) &= \begin{bmatrix} 0 & 0 & md\dot{\theta}\cos\theta \\ 0 & 0 & md\dot{\theta}\sin\theta \\ 0 & 0 & 0 \end{bmatrix}, \qquad \mathbf{G}(q) = \mathbf{0} \\
\mathbf{B}(q) &= \frac{1}{r}\begin{bmatrix} \cos\theta & \cos\theta \\ \sin\theta & \sin\theta \\ R & -R \end{bmatrix}, \qquad \tau = \begin{bmatrix} \tau_{\mathrm{r}} \\ \tau_{\mathrm{l}} \end{bmatrix} \\
\mathbf{A}^{\mathrm{T}}(q) &= \begin{bmatrix} -\sin\theta \\ \cos\theta \\ -d \end{bmatrix} \\
\lambda &= -m(\dot{x}\cos\theta + \dot{y}\sin\theta)\dot{\theta}
\end{aligned}
\tag{21}
$$

Similar dynamical models have been reported in the literature; for instance in Ref. 10 the mass and inertia of the driving wheels are considered explicitly.

The system Eq. (8) is now transformed into a more appropriate representation for control purposes. Differentiating Eq. (10), substituting this result in Eq. (8), and then multiplying by $\mathbf{S}^{\mathrm{T}}$, we can eliminate the constraint term $\mathbf{A}^{\mathrm{T}}(q)\lambda$. The complete equations of motion of the nonholonomic mobile platform are given by

$$\dot{q} = \boldsymbol{S}v \tag{22}$$

$$\mathbf{S}^{\mathrm{T}}\mathbf{M}\mathbf{S}\dot{v} + \mathbf{S}^{\mathrm{T}}(\mathbf{M}\dot{\mathbf{S}} + \mathbf{V_m}\mathbf{S})v + \overline{\mathbf{F}} + \overline{\tau}_{\mathrm{d}} = \mathbf{S}^{\mathrm{T}}\mathbf{B}\tau \tag{23}$$

where $v(t) \in \Re^{n-k}$ is a velocity vector. By appropriate definitions we can rewrite Eq. (23) as follows:

$$\overline{\mathbf{M}}(q)\dot{v} + \overline{\mathbf{V}}_{\mathbf{m}}(q,\dot{q})v + \overline{\mathbf{F}}(v) + \overline{\tau}_{\mathrm{d}} = \overline{\mathbf{B}}(q)\tau, \qquad \overline{\tau} \equiv \overline{\mathbf{B}}\tau \tag{24}$$

The true model of the vehicle is thus given by combining Eq. (22) and Eq. (24). However, in the latter equation it turns out that $\overline{\mathbf{B}}$ is square and invertible, so that standard computed-torque techniques can be used to compute the required vehicle control τ.

Control Design

Nonholonomic systems are a special class of nonlinear systems. They exhibit certain control properties that are worth mentioning. It has been shown that a nonholonomic system, under appropriate assumptions, is controllable; nevertheless, its equilibrium point $x_{\mathrm{e}} = 0$ cannot be made asymptotically stable by any smooth static state feedback. This is discussed in detail by Bloch et al. (11). It has been shown by Yamamoto and Yun (10) that a system with nonholonomic constraints is not input-state linearizable, but it may be input–output linearizable if the output function is chosen properly.

Let u be an auxiliary input. Then by applying the nonlinear feedback

$$\tau = \overline{\mathbf{B}}^{-1}(q)[\overline{\mathbf{M}}(q)u + \overline{\mathbf{V}}_{\mathbf{m}}(q,\dot{q})v + \overline{\mathbf{F}}(v)] \tag{25}$$

one can convert the dynamic control problem into the kinematic control problem

$$
\begin{aligned}
\dot{q} &= \mathbf{S}(q)v \\
\dot{v} &= u
\end{aligned}
\tag{26}
$$

Equation (26) represents a state-space description of the nonholonomic mobile robot and constitutes the basic framework for defining its nonlinear control properties. See Refs. 11 and 12 and the references therein.

In performing the input transformation Eq. (25), it is assumed that all the dynamical quantities (e.g., $\overline{\mathbf{M}}, \overline{\mathbf{F}}, \overline{\mathbf{V}}_{\mathrm{m}}$) of the vehicle are exactly known and $\overline{\tau}_{\mathrm{d}} = 0$. It is required to incorporate robust/adaptive control techniques if this is not the case.

Many approaches exist to selecting a velocity control, denoted by $v_{\mathrm{c}}(t)$, for the steering system Eq. (22). In this section, we desire to convert such a prescribed velocity control into a torque control $\tau(t)$ for the actual physical cart. It is desirable to have a common design algorithm capable of dealing with the three basic nonholonomic navigation problems defined next:

Tracking a Reference Trajectory. The trajectory tracking problem for nonholonomic vehicles is posed as follows. Let there be prescribed a reference vehicle

$$
\begin{aligned}
&\dot{x}_{\mathrm{r}} = v_{\mathrm{r}}\cos\theta_{\mathrm{r}}, \qquad \dot{y}_{\mathrm{r}} = v_{\mathrm{r}}\sin\theta_{\mathrm{r}}, \qquad \dot{\theta}_{\mathrm{r}} = \omega_{\mathrm{r}} \\
&q_{\mathrm{r}} = [x_{\mathrm{r}} \quad y_{\mathrm{r}} \quad \theta_{\mathrm{r}}]^{\mathrm{T}}, \qquad \mathbf{v}_{\mathrm{r}} = [v_{\mathrm{r}} \quad \omega_{\mathrm{r}}]^{\mathrm{T}}
\end{aligned}
\tag{27}
$$

with $v_{\mathrm{r}} > 0$ for all t. Find a smooth velocity control $v_{\mathrm{c}}(t)$ such that $\lim_{t\to\infty}(q_{\mathrm{r}} - q) = 0$. Then compute the auxiliary input $u(t)$ and the torque input $\tau(t)$ such that $v \to v_{\mathrm{c}}$ as $t \to \infty$.

Path Following. Given a path $P(x, y)$ in the plane and the mobile robot linear velocity $v_{\mathrm{L}}(t)$, find a smooth (angular) velocity control input $\omega_{\mathrm{c}}(t)$ such that $\lim_{t\to\infty} e_\theta = 0$ and $\lim_{t\to\infty} b(t) = 0$, where e_θ and $b(t)$ are the orientation error and the distance between a reference point in the mobile robot and the path P, respectively. Then compute the auxiliary input $u(t)$ and the torque input $\tau(t)$ such that $\omega \to \omega_{\mathrm{c}}$ as $t \to \infty$.

Point Stabilization. Given an arbitrary configuration q_{r}, find a velocity control input $v_{\mathrm{c}}(t)$ such that $\lim_{t\to\infty}$ (q_{r} −

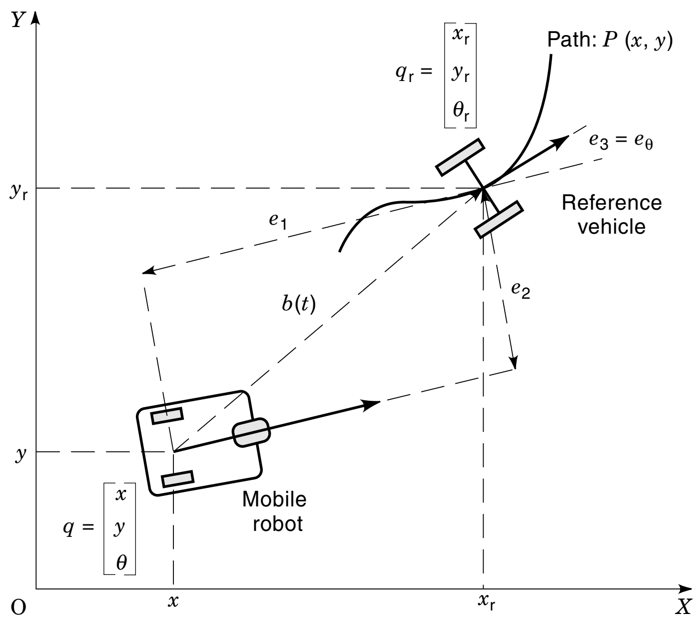

Figure 2. Relative position of the nonholonomic mobile robot with respect to a reference vehicle. The three basic navigation problems deal with the method of realizing stable control algorithms such that the position and orientation errors tend to zero.

$q) = 0$. Then compute the auxiliary input $u(t)$ and the torque input $\tau(t)$ such that $v \to v_c$ as $t \to \infty$.

Figure 2 illustrates the three basic navigation problems.

Tracking a Reference Trajectory

A general structure for the tracking control system is presented in Fig. 3. In this figure, complete knowledge of the dynamics of the cart is assumed, so that Eq. (25) is used to compute $\tau(t)$ given $u(t)$.

It is assumed that a solution to the steering system tracking problem is available. This is denoted by $v_c(t)$. Thus, the tracking error vector is expressed in a frame linked to the mobile platform:

$$e_p = T_e(q_r - q)$$
$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_r - x \\ y_r - y \\ \theta_r - \theta \end{bmatrix} \tag{28}$$

A velocity control input that achieves tracking for Eq. (26) is given by Kanayama et al. (5):

$$v_c = \begin{bmatrix} v_r \cos e_3 + k_1 e_1 \\ \omega_r + k_2 v_r e_2 + k_3 v_r \sin e_3 \end{bmatrix} \tag{29}$$

where $k_1, k_2, k_3 > 0$ are design parameters. Then the proposed nonlinear feedback acceleration control input is

$$u = \dot{v}_c + K_c(v_c - v) \tag{30}$$

where K_c is a positive definite, diagonal matrix. It is common in the literature to assume simply that $u = \dot{v}_c$, called *perfect velocity tracking,* which cannot be assured to yield tracking for the actual cart. The asymptotic stability of the system with respect to the reference trajectory can be proved by using standard Lyapunov methods (see LYAPUNOV METHODS). Define an auxiliary velocity error

$$e_c = v - v_c = \begin{bmatrix} e_4 \\ e_5 \end{bmatrix} = \begin{bmatrix} v_L - v_{c1} \\ \omega - v_{c2} \end{bmatrix} \tag{31}$$

By using Eq. (30), we obtain

$$\dot{e}_c = -K_c e_c \tag{32}$$

Then the velocity vector of the mobile base satisfies $v \to v_c$ as $t \to \infty$. Then, consider the following Lyapunov function candidate:

$$V = k_1(e_1^2 + e_2^2) + \frac{2k_1}{k_2}(1 - \cos e_3) + \frac{1}{2k_4}\left(e_4^2 + \frac{k_1}{k_2 k_3 v_r} e_5^2\right) \tag{33}$$

where $V \geq 0$, and $V = 0$ only if $e_p = 0$ and $e_c = 0$. It can be shown that $\dot{V} \leq 0$ and the entire error $e = [e_p^T\ e_c^T]^T \to 0$ as $t \to \infty$. Therefore, the closed-loop system is uniformly asymptotically stable. Note that Eq. (33) takes into account the velocity error produced by the dynamic computed-torque controller.

Control Design by Feedback Linearization. This control design technique has been investigated by a number of researchers (7,10). In this section, we apply this approach to our simplified mobile robot. Moreover, we show that controllability of the system is lost at the intersection point of the wheel axis and the axis of symmetry (P in Fig. 1).

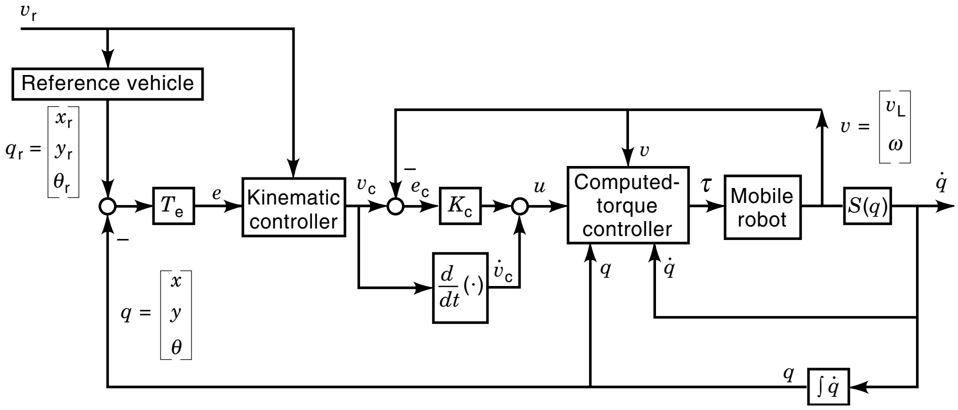

Figure 3. Kinematic and dynamic computed-torque control structure. The computed-torque controller generates the inputs to the actual mobile base such that the linear and angular velocities converge to the corresponding velocities generated by the kinematic controller.

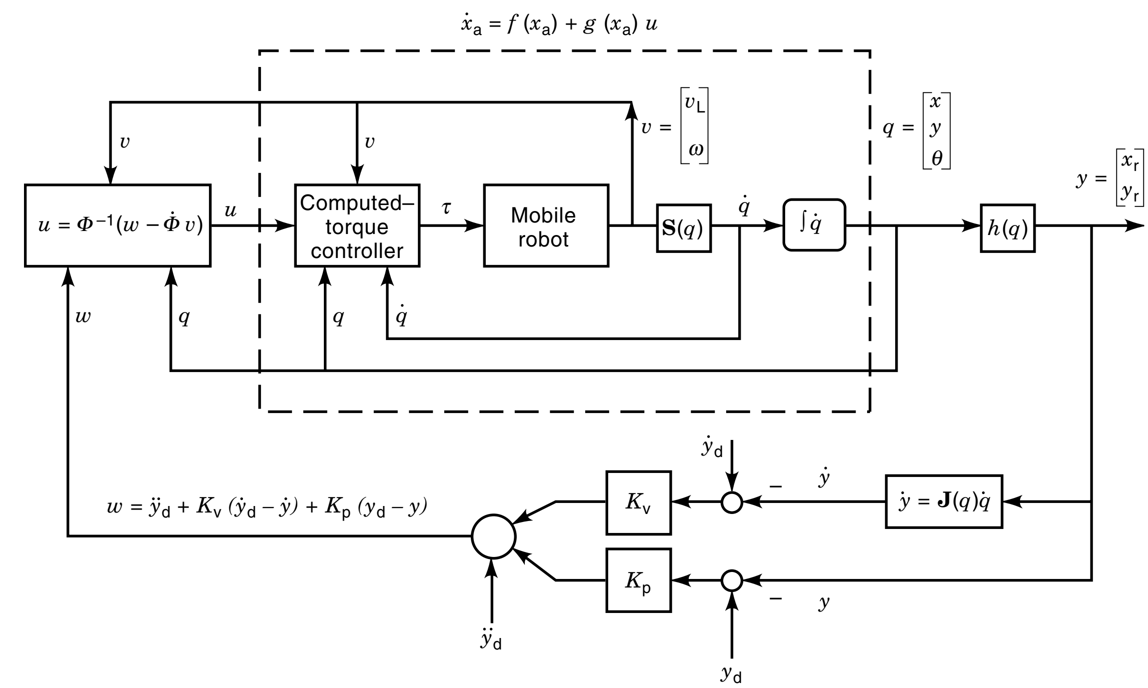

Figure 4. Control structure for trajectory tracking based on input–output feedback linearization. It is assumed that the reference point (x_m, y_m) is not chosen on the wheel axis. After performing input–output feedback linearization, standard linear control design techniques (e.g., PD) can be applied.

Figure 4 depicts the block diagram of the input–output feedback linearization control scheme. Equation (25) can be rewritten as

$$\begin{aligned} \dot{x}_a &= f(x_a) + g(x_a)u \\ y &= h(x_a) \end{aligned} \tag{34}$$

where the augmented state vector is $x_a = [q^T\ v^T]^T = [x\ y\ \theta\ v_L\ \omega]^T$ and

$$f(x_a) = \begin{bmatrix} \mathbf{S}(q)v \\ 0 \end{bmatrix}, \qquad g(x_a) = \begin{bmatrix} 0 \\ \mathbf{I} \end{bmatrix} \tag{35}$$

Since we are interested in position control, the selected output is a function of the mobile robot position $q(t)$. The dimension of the output vector is at most $n - k$ with $q \in \Re^n$ and k nonholonomic constraints. We also choose a reference point in the mobile base denoted by (x_m, y_m). Thus, the output equation becomes

$$\begin{aligned} y = h(q) &= [h_1(q) \quad h_2(q)]^T = \begin{bmatrix} x_r \\ y_r \end{bmatrix} \\ &= \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_m \\ y_m \end{bmatrix} \end{aligned} \tag{36}$$

Applying the standard approach of input–output linearization in Ref. 13—that is, differentiate $y(t)$ until $u(t)$ appears, and design $u(t)$ to cancel the nonlinearities—we obtain

$$\ddot{y} = \dot{\Phi}(q)v + \Phi(q)u \tag{37}$$

The system Eq. (37) is input–output-linearizable if and only if the decoupling matrix $\Phi(q)$ is full-rank. The decoupling matrix is given by

$$\Phi(q) = \mathbf{J}(q)\mathbf{S}(q) = \begin{bmatrix} \cos\theta & -y_m\cos\theta - (d + x_m)\sin\theta \\ \sin\theta & -y_m\sin\theta + (d + x_m)\cos\theta \end{bmatrix} \tag{38}$$

where $\mathbf{J}(q) = \partial h/\partial q$ is the Jacobian matrix and $|\Phi(q)| = d + x_m$. Thus, the input–output linearization problem is solvable if $x_m \neq -d$, that is, the reference point cannot be chosen on the wheel axis of the mobile base.

If the state feedback

$$u = \Phi^{-1}(w - \dot{\Phi}v) \tag{39}$$

is applied, Eq. (37) becomes

$$\ddot{y} = w \tag{40}$$

where $w(t)$ is an auxiliary control input. Let $y_d(t)$ denote the reference trajectory; then the auxiliary control input is given by

$$w = \ddot{y}_d + K_v(\dot{y}_d - \dot{y}) + K_p(y_d - y) \tag{41}$$

where K_p, $K_v < 0$ are design parameters. Defining the tracking error as $e = y - y_d$, it is clear from Eq. (40) and Eq. (41) that $e > 0$ as $t > \infty$.

Simulation Result. We have discussed two approaches that solve the trajectory tracking problem for a nonholonomic mobile robot. The computed-torque algorithm is the same in both methods; however, the way of designing the acceleration control input is different. The first method uses a technique called *backstepping* to extend a kinematic controller to a dynamic controller; see Refs. 14 and 15. In the second approach such an extension is carried out by input–output feedback linearization. Nevertheless, the two control structures given in Fig. 1 and Fig. 2 may exhibit similar behavior. Suppose the mobile robot has to follow a straight line with initial coordinates (0, 0) and inclination 45°. The initial state of the mobile platform is $q_0 = [3\ 1\ 0°]$. A typical mobile robot trajectory using either method is depicted in Fig. 5. Note that the mobile base describes a trajectory that satisfies the nonholonomic constraints without any path planning involved.

Path Following. As in the trajectory-tracking case, the kinematic–dynamic control law that solves the path-following problem can be implemented by either a nonlinear feedback design or an input–output feedback linearization design. In path following the geometry of the path $P(x, y)$ is given and the control objective is to follow the path as close as possible. For this purpose, the kinematic model of the mobile robot is transformed into a new set of coordinates which includes the geometry of the path. Let C be a reference point in the mobile base, and D be the *orthogonal projection* of C on $P(x, y)$. The signed distance $\overline{\mathrm{CD}}$ is denoted by $b(t)$. It is assumed that the linear velocity $v_{\mathrm{L}} \neq 0$ and the curvature of the path $\mu(s)$ is smooth and bounded. Let s denote the signed arc length along the path from a predefined initial position to D. The orientation error is denoted by e_θ (see Fig. 2).

The path-following problem consists in finding a control law such that $\lim_{t\to\infty} e_\theta = 0$ and $\lim_{t\to\infty} b(t) = 0$, given a path $P(x, y)$ and the linear velocity $v_{\mathrm{L}}(t)$. In Ref. 4, the following kinematic model is given:

$$\begin{aligned} \dot{b} &= v_{\mathrm{L}} \sin e_\theta \\ \dot{e}_\theta &= \xi \end{aligned} \tag{42}$$

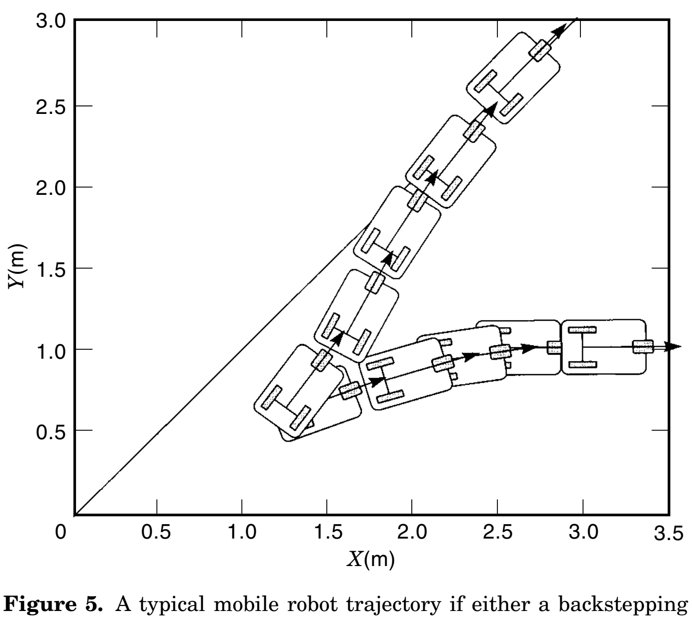

Figure 5. A typical mobile robot trajectory if either a backstepping controller or a feedback linearizing controller is utilized.

where

$$\xi = \omega - v_{\mathrm{L}} \cos e_\theta \frac{\mu(s)}{1 - \mu(s)b} \tag{43}$$

A *kinematic* nonlinear feedback that renders the system Eq. (42) asymptotically stable is given by

$$\xi = -k_1 b \frac{\sin e_\theta}{e_\theta} - k_2 e_\theta \tag{44}$$

where k_1, $k_2 > 0$ are design parameters. The new auxiliary velocity control input $v_c(t)$ becomes

$$v_{\mathrm{c}} = \begin{bmatrix} v_{\mathrm{L}} \\ v_{\mathrm{L}} \cos e_\theta \dfrac{\mu(s)}{1 - \mu(s)b} - k_1 b \dfrac{\sin e_\theta}{e_\theta} - k_2 e_\theta \end{bmatrix} \tag{45}$$

Now the acceleration input $u(t)$ in Eq. (30) is computed using vc in Eq. (45). Finally u will be used by the computed-torque dynamic controller to compute the required motor torques.

To apply input–output feedback linearization, the output function $y = h(\cdot)$ in Eq. (34) must be an appropriate function of the distance between the mobile base and the given path $P(x, y)$. Some choices for $h(\cdot)$ are given in Ref. (13).

Asymptotic Stabilization of a Class of Nonholonomic Systems

Feedback stabilization deals with finding feedback control laws such that an equilibrium point of the closed-loop system is asymptotically stable. Unfortunately, the linearization of nonholonomic systems about any equilibrium point is not asymptotically stabilizable. Moreover, there exists *no* smooth time-invariant state-feedback that makes an equilibrium point of the closed-loop system locally asymptotically stable (16). Therefore, feedback linearization techniques cannot be applied to nonholonomic systems directly.

A variety of techniques have been proposed in the nonholonomic literature to solve the asymptotic stabilization problem. A comprehensive summary of these techniques and other nonholonomic issues is given by Kolmanovsky and McClamroch (17). These techniques can be classified as (1) continuous time-varying stabilization (CTVS), (2) discontinuous time-invariant stabilization (DTIS), and (3) hybrid stabilization (HS). In CTVS the feedback control signals are smooth and time-periodic. In contrast, DTIS uses piecewise continuous controllers and sliding mode controllers. HS consists in designing a discrete-event supervisor and a set of low-level continuous controllers. The discrete-event supervisor coordinates the low-level controllers (by mode switching) to make an equilibrium point asymptotically stable. We shall discuss CTVS, since it can be implemented directly using the control structure shown in Fig. 3.

Continuous Time-Varying Stabilization. Smooth time-periodic control laws for nonholonomic mobile robots were introduced by Samson (6,18). In this section we solve the asymptotic stabilization problem as an extension of the trajectory tracking problem, that is, using the control structure shown in Fig. 3. The trajectory tracking problem is given by Eq. (27). It is assumed, without loss of generality, that the reference cart moves along the x axis, that is,

$$\dot{x}_{\mathrm{r}} = v_{\mathrm{r}}, \qquad q_{\mathrm{r}} = [x_{\mathrm{r}} \quad 0 \quad 0]^{\mathrm{T}}, \qquad \mathbf{v}_{\mathrm{r}} = [v_{\mathrm{r}} \quad 0]^{\mathrm{T}} \tag{46}$$

Therefore, the point stabilization problem reduces to finding a reference velocity $v_{\mathrm{r}}(t)$ and a velocity control $v_{\mathrm{c}}(t)$ such that $\lim_{t\to\infty}(q_{\mathrm{r}} - q) = 0$ and $\lim_{t\to\infty} x_{\mathrm{r}} = 0$. Then compute the auxiliary input $u(t)$ and the torque input $\tau(t)$ such that $v \to v_{\mathrm{c}}$ as $t \to \infty$. A possible solution has been proposed in Ref. 4, where

$$v_{\mathrm{r}} = -k_5 x_{\mathrm{r}} + g(e_{\mathrm{p}}, t) \tag{47}$$

and

$$g(e_{\mathrm{p}}, t) = \|e_{\mathrm{p}}\|^2 \sin t \tag{48}$$

The velocity control $v_{\mathrm{c}}(t)$ and its derivative are be given by

$$v_{\mathrm{c}} = \begin{bmatrix} v_{\mathrm{r}} \cos e_3 + k_1 e_1 \\ k_2 v_{\mathrm{r}} \dfrac{\sin e_3}{e_3} e_2 + k_3 e_3 \end{bmatrix} \tag{49}$$

$$\dot{v}_{\mathrm{c}} = \begin{bmatrix} \dot{v}_{\mathrm{r}} \cos e_3 \\ k_2 \dot{v}_{\mathrm{r}} \dfrac{\sin e_3}{e_3} e_2 \end{bmatrix} + \begin{bmatrix} k_1 & 0 & -v_{\mathrm{r}} \sin e_3 \\ 0 & k_2 v_{\mathrm{r}} \dfrac{\sin e_3}{e_3} & k_2 v_{\mathrm{r}} \dfrac{e_3 \cos e_3 - \sin e_3}{e_3^2} e_2 + k_3 \end{bmatrix} \dot{e} \tag{50}$$

where k_1, k_2, k_3, $k_5 > 0$ are design parameters. The acceleration input and the control torque are computed by Eq. (30) and Eq. (25), respectively. Typical behavior of this smooth time-periodic control law is depicted in Fig. 6(a).

Note that the rates of convergence provided by smooth time-periodic laws are at most $t^{-1/2}$, that is, nonexponential. Thus nonsmooth feedback laws with exponential rate of convergence have been proposed in the literature; see for instance M'Closkey and Murray (19). In this approach the change of coordinates

$$\begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ 0 & 0 & 1 \\ \sin\theta & -\cos\theta & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{51}$$

is used to transform the kinematic model of the mobile robot given in Eq. (26) to a *chained* form

$$\begin{aligned} \dot{\eta}_1 &= v_1 \\ \dot{\eta}_2 &= v_2 \\ \dot{\eta}_3 &= n_1 v_2 \\ \dot{v}_1 &= u_1 \\ \dot{v}_2 &= u_2 \end{aligned} \tag{52}$$

A periodic time-varying control law that renders the equilibrium of Eq. (52) globally exponentially stable is given by

$$v_{\mathrm{c}}(t) = \begin{bmatrix} v_{\mathrm{c1}} \\ v_{\mathrm{c2}} \end{bmatrix} = \begin{bmatrix} -\eta_1 + \dfrac{\eta_3}{\rho(\eta)} \cos t \\ -\eta_2 - \dfrac{\eta_3^2}{\rho^3(\eta)} \sin t \end{bmatrix} \tag{53}$$

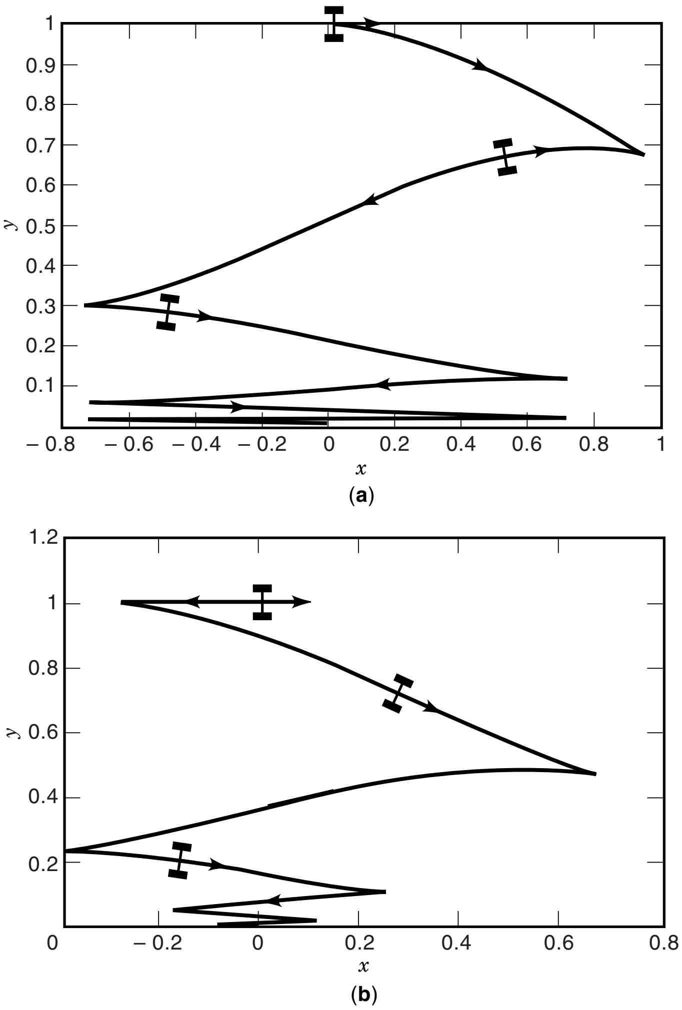

Figure 6. Mobile robot's trajectory. (a) The rate of convergence using a smooth time-varying feedback is at most $t^{-1/2}$. (b) On the other hand, on applying the controller Eq. (53) the robot's state converges to the origin exponentially.

where

$$\rho(\eta) = (\eta_1^4 + \eta_2^4 + \eta_3^2)^{1/4} \tag{54}$$

Typical behavior of this exponentially stabilizing time-periodic control law is depicted in Fig. 6(b).

Current Topics of Research in Nonholonomic Systems

Most of the control techniques for nonholonomic systems (e.g., nonholonomic mobile robots) assume that the dynamics of the system is perfectly known. Therefore the nonlinear feedback given by Eq. (25) can be used to convert the system into Eq. (26). This is a major simplification that may not hold in practice. Surprisingly, there are few references that consider adaptive/robust approaches for nonholonomic mechanical systems. In this subsection we provide a basic framework that can accommodate the so-called intelligent control techniques such as neural networks and fuzzy logic systems; see for instance Ref. 20.

Given the desired control velocity $v_{\mathrm{c}}(t)$ [(e.g., Eq. (29) or Eq. (49)], define now the velocity tracking error as

$$e_{\mathrm{c}} = v_{\mathrm{c}} - v \tag{55}$$

Differentiating Eq. (55) and using Eq. (24), the nonholonomic robot dynamics may be written in terms of the velocity tracking error as

$$\overline{\mathbf{M}}(q)\dot{e}_c = -\overline{\mathbf{V}}_{\mathbf{m}}(q,\dot{q})e_c - \overline{\tau} + f + \overline{\tau}_d \tag{56}$$

where the important *nonlinear robot function* is

$$f = \overline{\mathbf{M}}(q)\dot{v}_c + \overline{\mathbf{V}}_{\mathbf{m}}(q,\dot{q})v_c + \overline{\mathbf{F}}(v) \tag{57}$$

The function $f(\cdot)$ contains all the nonholonomic robot parameters such as masses, moments of inertia, friction coefficients, and so on. These quantities are often imperfectly known and difficult to determine. Therefore, a suitable control input for velocity following is given by the computed-torque-like control

$$\overline{\tau} = \hat{f} + K_c e_c - \gamma \tag{58}$$

with K_c a diagonal, positive definite gain matrix, and $\hat{f}$ an *estimate* of the robot function f that is provided by an *adaptive network* (e.g., neural network or fuzzy logic). The robustifying signal γ is required to compensate unmodeled dynamics and/or unmodeled unstructured disturbances. Using this control in Eq. (56), the closed-loop system becomes

$$\overline{\mathbf{M}}\dot{e}_c = -(K_c + \overline{\mathbf{V}}_{\mathbf{m}})e_c + \tilde{f} + \overline{\tau}_d + \gamma \tag{59}$$

where the velocity tracking error is driven by the *functional estimation error* $\tilde{f} = f - \hat{f}$. The robustifying signal γ can be selected by several techniques, including sliding-mode and others, under the general aegis of *robust control* methods. (See ROBUST CONTROL.)

For good performance, the bound on $e_c(t)$ should be in some sense small enough, because it will directly affect the position error $e_p(t)$. Thus, the nonholonomic control system consists of two subsystems: (1) a kinematic controller, and (2) a dynamic controller. The dynamic controller provides the required torques, so that the robot's velocity tracks a reference velocity input. As perfect velocity tracking does not hold in practice, the dynamic controller generates a velocity error $e_c(t)$, which is assumed to be bounded by some known constant. This error can be seen as a disturbance for the kinematic system; see Fig. 7. Assuming that the nonholonomic constraints are not violated, it is possible to compute the bound on the position error $e_p(t)$. Unfortunately, systematic methods to design adaptive/robust controllers for nonholonomic systems are unknown.

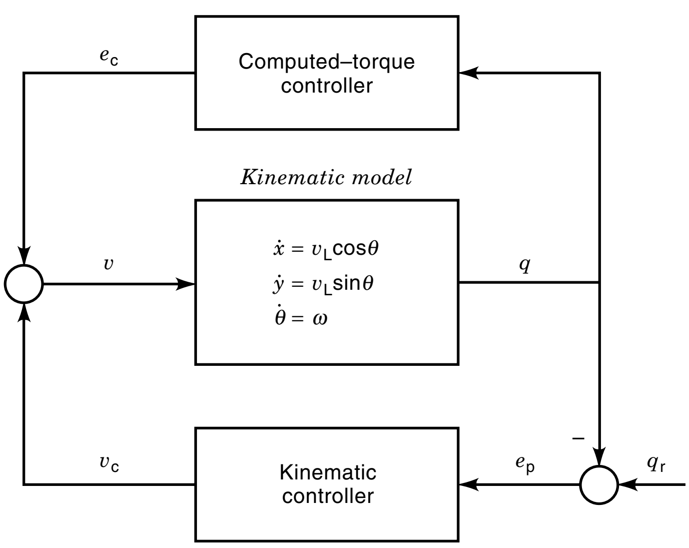

Figure 7. Block diagram of the closed-loop nonholonomic control system. The velocity error $e_c(t)$ produced by the dynamic controller may be considered as a disturbance of the closed-loop kinematic system.

KINEMATICS OF A HYPERREDUNDANT ROBOT

In a *kinematically redundant* robot the number of degrees of freedom (DOF) is larger than the dimension of the workspace. This means that the robot has more joints than the minimum required to perform a given task. The extra DOF can be used to avoid obstacles and singularities in the workspace, optimize the robot's motion with respect to a cost function, and move in a highly constrained environment. The forward kinematics of a redundant manipulator is solved in a similar fashion to that of a nonredundant manipulator. However, in the case of redundant manipulators the inverse kinematics problem is ill posed; that is, there may be an infinite number of solutions for a given tool-frame position and orientation. Moreover, for a given reference trajectory of the end effector $y_d(t) \in \Re^m$, there may exist an infinite number of trajectories $q(t) \in \Re^n$ in joint space. We need a performance index to choose among these possible solutions of the inverse kinematic problem. Optimization techniques have traditionally been used to find a solution of the kinematic problem in redundant manipulators. The main idea is to find a joint velocity vector $\dot{q}$ that minimizes the weighted cost function

$$J_{\dot{q}} = \tfrac{1}{2}\dot{q}^{\mathrm{T}}\mathbf{W}\dot{q} \tag{60}$$

subject to

$$\dot{y}_d = \mathbf{J}(q)\dot{q} \tag{61}$$

where $\mathbf{W} \in \Re^{n\times n}$ is a positive definite symmetric weighting matrix, and $\mathbf{J} \in \Re^{m\times n}$ is the manipulator Jacobian. The solution to this optimization problem is given by

$$\dot{q} = \mathbf{W}^{-1}\mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{W}^{-1}\mathbf{J}^{\mathrm{T}})^{-1}\dot{y}_d \equiv \mathbf{J}_{\mathrm{W}}^{\dagger}\dot{y}_d \tag{62}$$

where $\mathbf{J}_{\mathrm{W}}^{\dagger} \in \Re^{n\times m}$ is called the *weighted pseudoinverse*.

The result of a very large degree of kinematic redundancy is a *hyperredundant manipulator*. Hyperredundant robots are analogous to snakes, elephant trunks, and tentacles. Some examples of hyperredundant robot morphologies are depicted in Fig. 8. Figure 8(a) shows a redundant robot which consists of a large number of rigid links. Figure 8(b) presents a continuously deformable robot. Finally, Fig. 8(c) depicts a variable-geometry truss (VGT) robot. These hyperredundant robots are useful in performing tasks in highly constrained workspaces. These tasks may include inspection of space stations, active endoscopy for noninvasive surgery, or inspection of nuclear reactor cores. In the last twenty years many implementations and practical applications of hyperredundant robots have been reported. See for instance Ref. 21 and the references therein.

Traditional methods for solving the inverse kinematics of redundant manipulators [e.g., the pseudoinverse (51)] are not adequate for many hyperredundant structures, where the large number of DOF makes these methods computationally impractical. This section follows the lines and notation of Chirikjian and Burdick (22,23). They have developed efficient

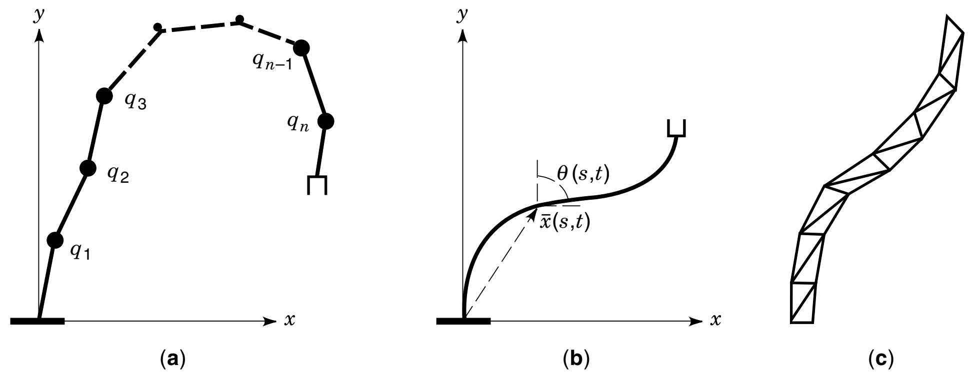

Figure 8. Examples of hyperredundant robot morphologies. The special features of this class of mechanical systems make them suitable for performing a variety of tasks in highly constrained environments. (Modified from Ref. 23, Copyright © IEEE 1995.)

kinematic methods for hyperredundant robots based on configuration optimization rather than trajectory optimization. For the hyperredundant case where the robot is built in a modular fashion [see Fig. 8(c)], configuration optimization seems to provide a more convenient framework than trajectory optimization. To be specific, a function of the robot's configuration is minimized while satisfying end-effector position constraints

$$\min_q J_{\mathrm{q}}(q) = \tfrac{1}{2} q^{\mathrm{T}} \mathbf{W} q \tag{63}$$

subject to

$$C(q) = F(q) - y_{\mathrm{d}} = 0 \tag{64}$$

In order to solve this optimization problem we can use, for instance, the Lagrange–Newton approach. The Lagrangian is given by

$$L(q, \lambda) = J_{\mathrm{q}}(q) + \lambda^{\mathrm{T}} C(q) \tag{65}$$

where λ denotes the Lagrange multipliers. Local extrema are found by solving the following matrix equation with initial estimates q_0 and λ_0:

$$\begin{bmatrix} \delta q_k \\ \delta \lambda_k \end{bmatrix} = - \begin{bmatrix} \mathbf{P}(q_k, \lambda_k) & \mathbf{J}^{\mathrm{T}}(q_k) \\ \mathbf{J}^{\mathrm{T}}(q_k) & \mathbf{0} \end{bmatrix}^{-1} \begin{bmatrix} \nabla J_{\mathrm{q}}(q_k) + \mathbf{J}^{\mathrm{T}}(q_k)\lambda_k \\ C(q_k) \end{bmatrix} \tag{66}$$

where

$$\mathbf{P}(q_k, \lambda_k) = \nabla^2 J_{\mathrm{q}}(q_k) + \sum_{i=1}^{m} \lambda_i \nabla^2 C_i(q) \tag{67}$$

The new estimates of the extrema are given by the update rule

$$\begin{aligned} q_{k+1} &= q_k + \delta q_k \\ \lambda_{k+1} &= \lambda_k + \delta \lambda_k \end{aligned} \tag{68}$$

The convergence of the algorithm is quadratic if the initial estimates are good. However, the computation of the Hessian matrix [i.e., the second derivatives of $F(q)$] for nonserial hyperredundant manipulators [e.g., Fig. 8(b,c)] may be extremely difficult.

The Backbone-Curve Representation

The optimization procedures outlined in the previous section can be very difficult to apply to continuous-morphology hyperredundant robots [e.g., Fig. 8(b)]. Therefore, some alternative techniques have been developed by a number of authors. We briefly discuss the technique developed by Chirikjian and Burdick (23), which provides a fairly general framework for computing the kinematics of most of hyperredundant manipulator topologies. The key idea is that many hyperredundant robot morphologies can be represented by a geometric abstraction (Fig. 9) called the *backbone-curve model.* The backbone curve may be *extensible* or *inextensible,* depending on the robot's mechanical design. Furthermore, a *backbone reference set* is defined by a backbone-curve parametrization and a set of reference frames that evolve along the curve. Thus, the kinematic problem is solved in two steps. First, the time evolution of the backbone reference set is determined. Then, the backbone kinematic solution is used to specify the joint variables (i.e., displacements) of the actual robot. In the case of modular hyperredundant manipulators as shown in Fig. 8(c),

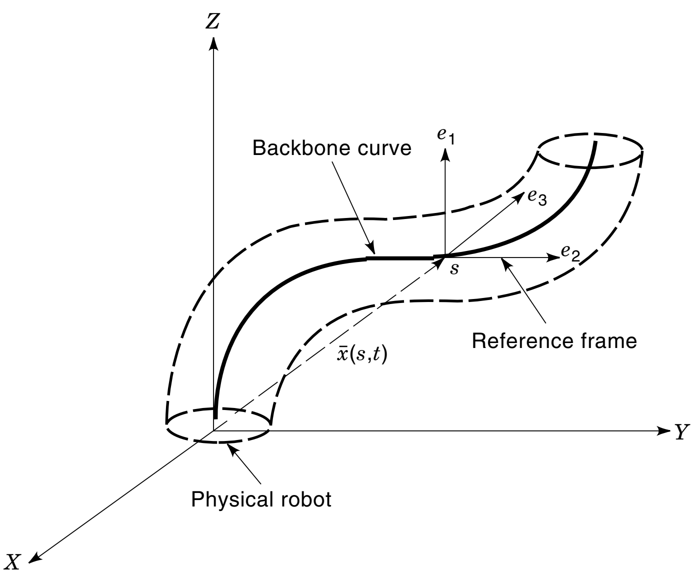

Figure 9. Backbone-curve abstraction. The backbone curve provides a framework to study the kinematics of many hyperredundant robots in a unified and systematic manner. (Reproduced from Ref. 23, Copyright © IEEE 1995.)

a *fitting procedure* is required to determine the actuator displacements that make the robot fit the backbone curve as closely as possible.

FREE-FLOATING SPACE MANIPULATORS

Robotic technologies for space servicing and exploration have become an intensive area of research in the last past two decades; see Bejczy et al. (24). Since a space robot has to perform tasks in an environment different from the surface of the earth, new problems unique to space robotics have emerged. A space robot has to be economical in power consumption, volume, and mass. It has to carry out tasks under zero gravity. Furthermore, space robots are flexible due to their light weight.

A *free-flying* or *free-floating* space manipulator consists of a free base (e.g., a spacecraft) and a manipulator mounted on it. In a free-flying case the position and attitude of the free base may be controlled by thrusters. Some control and path-planning problems arise from the fact that the manipulator motion may disturb the position and attitude of the free base. On the other hand, the base's position and attitude are not actuated during the manipulator operation and can move freely. It has been pointed out by Dubowsky and Papadopoulos (25) that free-floating robots may have *dynamic singularities,* a unique feature not present in conventional terrestrial manipulators. Various control and planning methods for space robots have been developed as extensions of methods applied to conventional manipulators; see for instance Ref. 26. These techniques can be grouped into four classes according to Dubowsky and Papadopoulos (25):

1. The free-base position and attitude are fixed. The spacecraft actuators compensate for disturbances caused by the manipulator motion.
2. The spacecraft's attitude is controlled, but not its translation.
3. The spacecraft can rotate and translate as a result of manipulator motions. This case corresponds to a free-floating manipulator where the space robot is controlled by its joint actuators only.
4. The last category consists of free-flying robots. In this case the spacecraft and the manipulator are controlled in a coordinated way such that a desired location and orientation in space is reached by the robot's end effector.

We focus our analysis on free-floating space robots. Unlike free-flying robots, free-floating systems do not have the disadvantage of a limited life because of the use of jet fuel.

Modeling of Free-Floating Space Manipulators

In this section we follow the notation in Ref. 25 to determine the kinematic and dynamic model of free-floating space robots. Figure 10 shows a general model of an N-DOF space manipulator with revolute joints. It is assumed that the manipulator is composed of rigid bodies and no external forces or torques act on the system, so that the total momentum is conserved. Link 0 denotes the spacecraft, and the notation is as follows:

$q \in \Re^n$: joint variables
$\dot{q} \in \Re^n$: manipulator joint rates
$\tau \in \Re^n$: joint torques
CM: system center of mass
r_{cm}: position of CM with respect to the inertial frame
r_{E}: position of end effector with respect to the inertial frame
r_{s}: position of spacecraft with respect to the inertial frame
v_{s}: linear velocity of the spacecraft
$\dot{r}_{\text{E}}$: linear velocity of the end effector
a_i, b_i: link vectors (see Fig. 10)
$\vec{\text{k}}_i$: unit vector along the rotational axis of joint i

The CM linear velocity is ${}^{0}\dot{r}_{\text{cm}} = 0$ for free-floating manipulators where the linear momentum is constant. The position of the end effector is given in Yoshida and Umetani (27) as

$$r_{\text{E}} = r_{\text{s}} + b_0 + \sum_{i=1}^{n} (a_i + b_i) \tag{69}$$

Differentiating yields

$$\dot{r}_{\text{E}} = v_{\text{s}} + \omega_0 \times (r_{\text{E}} - r_{\text{s}}) + \sum_{i=1}^{n} [\vec{\text{k}}_i \times (r_{\text{E}} - r_i)]\dot{q}_i \tag{70}$$

The angular velocity of the end effector is given by

$$\omega_{\text{E}} = \omega_0 + \sum_{i=1}^{n} \vec{\text{k}}_i \dot{q}_i \tag{71}$$

The basic kinematics of the free-floating space robot can be expressed as

$$\begin{bmatrix} \dot{r}_{\text{E}} \\ \omega_{\text{E}} \end{bmatrix} \equiv v_{\text{E}} = \mathbf{J}_{\text{s}} \begin{bmatrix} \dot{r}_{\text{s}} \\ \omega_0 \end{bmatrix} + \mathbf{J}\dot{q} = \mathbf{J}^*\dot{q} \tag{72}$$

where $\mathbf{J}^* \in \Re^{6\times n}$ is called the *generalized Jacobian matrix* (GJM). The GJM for free-floating systems is an extension of the Jacobian matrix $\mathbf{J}$ for the fixed-base manipulators. Control algorithms applied to fixed-base (i.e., terrestrial) manipulators can be used in controlling a free-floating robot provided that dynamic singularities are avoided. It is worth mentioning that if the mass m_0 and the inertia tensor $\mathbf{I}_0$ of the spacecraft approach infinity, then $\mathbf{J}^* \to \mathbf{J}$, that is, the GJM converges to the conventional Jacobian matrix.

The dynamics of a free-floating manipulator obtained using a Lagrangian approach are

$$\mathbf{M}^*(q)\ddot{q} + \mathbf{V}^*_{\mathbf{m}}(q, \dot{q})\dot{q} = \tau \tag{73}$$

where $\mathbf{M}^*(q)$ is the symmetric, positive definite inertia matrix, $\mathbf{V}^*_{\mathbf{m}}$ is a centripetal and Coriolis matrix, and τ is a control input vector. If the control task is to move the manipulator with respect to its free base, then conventional computed-torque control techniques can be applied. For a comprehensive analysis of computed-torque methods for rigid manipulators the reader is referred to Lewis et al. (9). Furthermore, if the control task requires one to drive the end effector to a fixed position and orientation, any control algorithm devel-

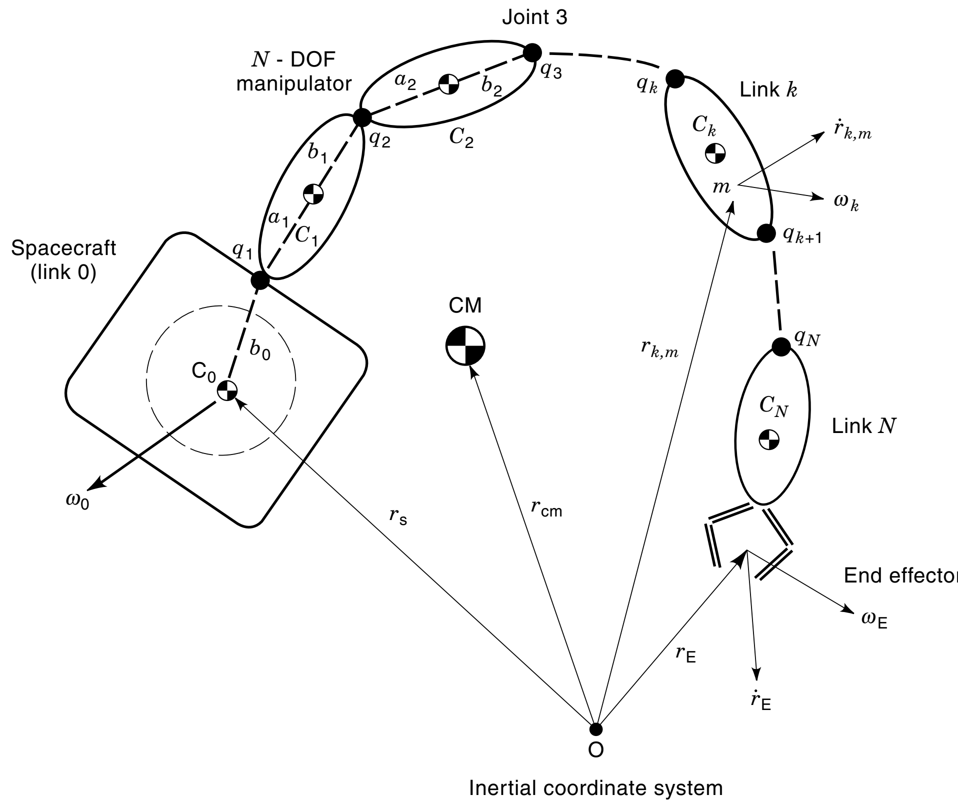

Figure 10. A free-floating space manipulator. The spacecraft (link 0) can rotate and translate freely as a result of manipulator motions. The space robot is only controlled by its joint actuators.

oped for terrestrial manipulators will work for free-floating space robots, provided that the dynamic singularities are avoided, that is, det $\mathbf{J}^* \neq 0$. Details are presented in Ref. 25.

BIBLIOGRAPHY

1. J. J. Craig, *Introduction to Robotics, Mechanics and Control,* 2nd ed., Reading, MA: Addison-Wesley, 1989.
2. J. Barraquand and J.-C. Latombe, Nonholonomic multibody mobile robots: Controllability and motion planning in the presence of obstacles, *Proc. IEEE Int. Conf. Robot. Autom.,* 1991, pp. 2328–2335.
3. R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation,* Boca Raton, FL: CRC Press, 1994.
4. C. Canudas de Wit et al., Nonlinear control design for mobile robots, in Y. F. Zheng (ed.), *Recent Trends in Mobile Robots,* Singapore: World Scientific, 1993, pp. 121–156.
5. Y. Kanayama et al., A stable tracking control method for an autonomous mobile robot, *Proc. IEEE Int. Conf. Robot. Autom.,* 1990, pp. 384–389.
6. C. Samson, Time-varying feedback stabilization of car-like wheeled mobile robots, *Int. J. Robot. Res.,* **12** (1): 55–64, 1993.
7. N. Sarkar, X. Yun, and V. Kumar, Control of mechanical systems with rolling constraints: Application to dynamic control of mobile robots, *Int. J. Robot. Res.,* **13** (1): 55–69, 1994.
8. Y. Zhao and S. L. BeMent, Kinematics, dynamics and control of wheeled mobile robots, *Proc. IEEE Int. Conf. Robot. Autom.,* 1992, pp. 91–96.
9. F. L. Lewis, C. T. Abdallah, and D. M. Dawson, *Control of Robot Manipulators,* New York: Macmillan, 1993.
10. Y. Yamamoto and X. Yun, Coordinating locomotion and manipulation of a mobile manipulator, in Y. F. Zheng (ed.), *Recent Trends in Mobile Robots,* Singapore: World Scientific, 1993, pp. 157–181.
11. A. M. Bloch, M. Reyhanoglu, and N. H. McClamroch, Control and stabilization of nonholonomic dynamic systems, *IEEE Trans. Autom. Control,* **37**: 1746–1757, 1992.
12. G. Campion, B. d'Andréa-Novel, and G. Bastin, Controllability and state feedback stabilizability of nonholonomic mechnical systems, in C. Canudas de Wit (ed.), *Lecture Notes in Control and Information Science,* Berlin: Springer-Verlag, 1991, pp. 106–124.
13. J. E. Slotine and W. Li, *Applied Nonlinear Control,* Englewood Cliffs, NJ: Prentice-Hall, 1991.
14. R. Fierro and F. L. Lewis, Control of a nonholonomic mobile robot: Backstepping kinematics into dynamics, *J. Robot Syst.,* **14** (3): 149–163, 1997.
15. I. Kolmanovsky and N. H. McClamroch, Application of integrator backstepping to nonholonomic control problems, *Proc. IFAC Nonlinear Control Syst. Des. Symp.,* 1995, pp. 747–752.
16. R. W. Brockett, Asymptotic stability and feedback stabilization, in R. W. Brockett, R. S. Millman, and H. J. Sussmann (eds.), *Differential Geometric Control Theory,* Boston: Birkhäuser, 1983, pp. 181–191.
17. I. Kolmanovsky and N. H. McClamroch, Developments in nonholonomic control problems, *IEEE Trans. Control Syst. Technol.,* **15**: 20–36, 1995.
18. C. Samson, Velocity and torque feedback control of a nonholonomic cart, in C. Canudas de Wit (ed.), *Lecture Notes in Control and Information Science,* New York: Springer-Verlag, 1991, pp. 125–151.
19. R. T. M'Closkey and R. M. Murray, Extending exponential stabilizers for nonholonomic systems from kinematic controllers to dynamic controllers, *Proc. IFAC Symp. Robot. Control,* 1994, pp. 211–216.
20. R. Fierro and F. L. Lewis, Practical point stabilization of a nonholonomic mobile robot using neural networks, *Proc. IEEE Conf. Dec. Control,* 1996, pp. 1722–1727.

21. S. Hirose and A. Morishima, Design and control of a mobile robot with an articulated body, *Int. J. Robot. Res.,* **9** (2): 99–114, 1990.
22. G. S. Chirikjian and J. W. Burdick, A modal approach to hyper-redundant manipulator kinematics, *IEEE Trans. Robot. Autom.,* **10**: 343–354, 1994.
23. G. S. Chirikjian and J. W. Burdick, Kinematically optimal hyper-redundant manipulator configurations, *IEEE Trans. Robot. Autom.,* **11**: 794–806, 1995.
24. A. K. Bejeczy, S. T. Venkataraman, and D. Akin (eds.), Special issue on space robotics, *IEEE Trans. Robot. Autom.,* **9**: 1993.
25. S. Dubowsky and E. Papadopoulos, The kinematics, dynamics and control of free-flying and free-floating space robotic systems, *IEEE Trans. Robot. Autom.,* **9**: 531–543, 1993.
26. Y. Xu and T. Kanada (eds.), *Space Robotics: Dynamics and Control,* Boston: Kluwer, 1993.
27. K. Yoshida and Y. Umetani, Control of space manipulators with generalized Jacobian matrix, in Y. Xu and T. Kanada (eds.), *Space Robotics: Dynamics and Control,* Boston: Kluwer, 1993, pp. 165–204.

RAFAEL FIERRO
FRANK L. LEWIS
The University of Texas at Arlington

ROBOT MANIPULATOR. See MANIPULATORS.