Cobot Control

Witaya Wannasuphoprasit, Brent Gillespie, J. Edward Colgate, Michael A. Peshkin

1997 International Conference on Robotics and Automation. Albuquerque, NM

Cobot Control

Witaya Wannasuphoprasit
R. Brent Gillespie
J. Edward Colgate
Michael A. Peshkin

Department of Mechanical Engineering
Northwestern University
Evanston, IL 60208-3111

Abstract

Cobots are a class of mechanically passive robotic devices, intended for direct physical collaboration with a human operator. The operator supplies all motive power while the cobot enforces software-defined guiding surfaces, or constraints. Cobots are intrinsically passive, safe devices. This is because, rather than employ powered actuators to produce constraint forces, cobots use “steerable” nonholonomic joints. Constraint forces are mechanical in origin, yet software defined.

The simplest possible cobot is a unicycle which is steered by a servo system acting under computer control, but which is moved by a human operator. The unicycle cobot requires essentially no consideration of kinematics. Two fundamental control modes of the unicycle cobot, “virtual caster” and “constraint tracking”, are reviewed.

More complicated cobots, such as the three-wheeled “Scooter”, require a set of kinematic transformations relating configuration space to joint space. These transformations play a role in cobot control like that of the jacobian in robot control.

1. Introduction

Co-manipulation robots, or “cobots”, are a class of passive robotic devices which are intended for direct collaboration with a human operator within a shared workspace. Cobots are intended not to enhance human strength, but to provide virtual “guiding” surfaces. The human part of the team supplies all motive power, while the cobot interfaces to a computer.

As an illustrative example, imagine human-cobot collaboration in automobile assembly. Suppose that the task is to put a large, ungainly part, such as an instrument panel, into the vehicle body. The cobot would have two jobs: supporting the instrument panel weight, and setting up virtual guides which, when followed, would ensure that no collisions could occur. The human would have the job of pushing the instrument panel along the virtual guides until properly positioned within the body. The human would also be in charge of fine manipulation tasks, such as “seating” the instrument panel.

Although the constraint forces in this example might be quite large, a cobot would not need to generate large actuator torques at all. This is because a cobot, unlike a conventional robot or haptic interface, has no joint actuators. Instead, a cobot has “steerable” nonholonomic joints whose intrinsic mechanics provide the constraint forces.

The essentials of cobot mechanics may be understood by considering the simplest such device, the unicycle cobot shown in Fig. 1. The cobot mechanism consists of a free-rolling wheel in contact with a working surface. The wheel’s rolling velocity is monitored by an encoder, but it is not driven by a motor. The motor in the figure simply steers the wheel, and cannot cause the cobot to move. Only the operator can cause it to move, by applying forces to the handle. A handle-mounted force sensor monitors these user forces. The linear rails in the figure are part of a Cartesian frame which supports the cobot upright. These rails are also instrumented with linear potentiometers in order to measure the global position of the cobot.

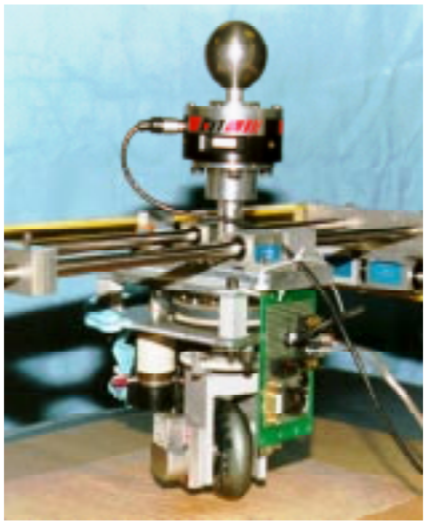


Figure 1. The unicycle cobot.

This cobot has a two-dimensional (planar) configuration space corresponding to all possible locations of the unicycle assembly on the working plane. Although the unicycle has only one degree-of-freedom, it may, by proper steering, reach any point in the planar workspace. Such is the nature of nonholonomic constraint. In operation, however, virtual constraint surfaces may be defined in software to prohibit entry into excluded regions of the plane.

The unicycle cobot has two essential modes of operation:

a. Virtual caster mode is invoked when the cobot's position in its planar workspace is away from all defined constraint surfaces. The cobot should therefore permit any motion that the user attempts to impart. Thus, the wheel must act something like a caster; yet, it is not a caster in the conventional sense. Instead, it has a straight-up shaft like a unicycle. Virtual castering arises when a handle-mounted force sensor detects forces perpendicular to the wheel's rolling direction, and the wheel is steered (by a motor) to minimize these forces. In effect, the wheel is always steered in the direction that the user wants it to roll.

b. Constraint tracking mode is invoked when the user brings the cobot's position in the plane to a place where a constraint surface is defined. At this point, the computer which controls the steering motor no longer attempts to minimize force. Instead, the wheel is steered in a direction that is tangential to the constraint surface. The force sensor still monitors force perpendicular to the wheel's rolling direction. If the force would tend to push the wheel into the constraint, it is ignored. If the force would tend to pull the wheel off of the constraint, is interpreted just as in the virtual caster mode. Thus, is impossible to push the unicycle past a virtual constraint (unless the wheel slips), but the unicycle can easily be pulled off the constraint surface.

The unicycle cobot has some noteworthy characteristics. First, although it is a one d.o.f. device (the wheel fixes the ratio of x and y velocities), it behaves in the virtual caster mode as though it has two d.o.f. Second, although it uses a motor to steer, it is completely passive in the plane of operation. Because the motor exerts torques about an axis that passes through the wheel/ground contact point, it does not generate any reaction forces in the plane.

Cobots with larger dimensional configuration spaces[1] exist, and are discussed in [4]. The remainder of this paper focuses on cobot control and kinematics. We begin, in the next section, with a discussion of unicycle control. We go on, in §3, to discuss the control of Scooter, a tricycle cobot with nontrivial kinematics.

2. Unicycle Control

2.1 Virtual Caster

The ideal caster controller would perceptually eliminate the wheel. In other words, a user manipulating the machine would perceive it to be a point mass in the plane. For a point mass, the acceleration and force vectors are collinear and in fixed proportion. The implication for a unicycle is that, not only must forces in the wheel direction, $F_{||}$, produce accelerations of $a_{||} = F_{||}/M$, but forces normal to the wheel, $F_{\perp}$, must similarly produce accelerations of $a_{\perp} = F_{\perp}/M$. A very simple kinematic analysis, however, shows that a wheel traveling at a speed u with a steering angular velocity ω, has an instantaneous normal acceleration of $a_{\perp} = u\omega$. Thus, we can obtain a prescription for the steering velocity in response to user forces which would result in point-mass-like behavior:

$$\omega = \frac{F_{\perp}}{uM} \tag{1}$$

Note that u in Eq. 1 is not under our control but is determined by the user. This equation indicates that the problem of virtual caster control is fundamentally non-linear: the correct sign of the steering velocity is determined by the product of the signs of $F_{\perp}$ and u, which cannot be approximated by a linear relation. Eq. 1 also indicates that, for a given normal force, the steering velocity scales inversely with the translational velocity. Because of this, there is a singularity at zero speed. At zero speed, it is not physically possible to make the unicycle behave like a particle.

An in-depth discussion of unicycle caster control, including implementation issues (e.g., handling the zero velocity singularity; finite sensor resolution) can be found in [2].

2.2 Path Tracking

In this paper, we will consider only the case of bilateral constraint (path tracking) rather than unilateral constraint (virtual wall). For the unicycle cobot, a bilateral constraint is easily made unilateral with a software switch [1, 2]. We will discuss a nominal, or feedforward, controller first, followed by feedback control. A standing assumption is that the path to be tracked, $R_o(s)$, is parameterized in terms of its own path length, s.

2.2.1 Feedforward Control

An appropriate feedforward controller can be derived by assuming perfect path tracking. In such a case, the unicycle would at all times be tangent to the path, as illustrated in Fig. 2. Moreover, the speed, u, of the

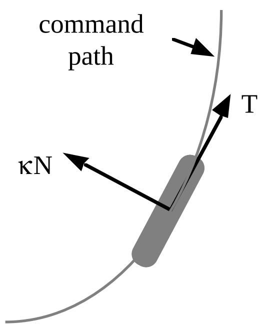


Figure 2. Top view of a unicycle cobot tracking a path

[1] One is tempted to write "higher degree-of-freedom cobots", but all cobots, regardless of the configuration space dimension, have one degree-of-freedom.

unicycle along the path, and the curvature, κ, of the path would together determine the necessary steering velocity:

$$\omega = \kappa u \tag{2}$$

Unfortunately, a number of non-idealities including steering dynamics and sideslip, will conspire to ensure that this feedforward controller alone will not keep the unicycle on an intended path. Thus, it is necessary to consider feedback corrections.

2.2.2 Feedback Control

Fig. 3 is an illustration of a unicycle which is off of its intended path. In such a case, the C-space configuration of the unicycle cobot ($R = [x,y]^T$) is not what the controller is expecting ($R_o(s)$, where s is the measured path length). Moreover, the cobot heading[2] (T) is, in general, not parallel to the expected heading (T_o). The control policy that corrects for these errors has two components:

1. s is replaced with s', where $R_o(s')$ is the closest point on the command path to the cobot's actual configuration (R).
2. κ in Eq. 2 is replaced with $\kappa + \delta\kappa$, where $\delta\kappa$ is determined on the basis of the configuration and heading errors, and is intended to steer the unicycle back toward the command path.

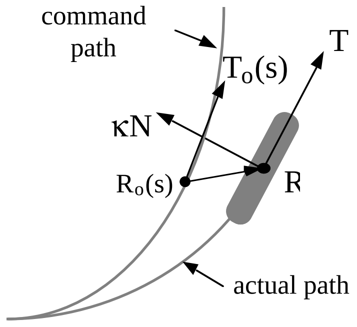


Figure 3. Imperfect path tracking

The closest point computation required to find s' will, in general, be complicated to perform. However, if we make the reasonable assumption that deviations from the path are small (i.e., our path tracking controller works well), then it is evident (see Fig. 3) that the path length error (s' - s) is simply the length of $(R - R_o(s))$ projected onto T_o. Thus, we define:

$$s' = s + (R - R_o(s)) \cdot T_o(s) \tag{3}$$

When the actual cobot configuration is compared to $R_o(s')$ there are of course still errors. These errors are of two types:

- Displacement — $\Delta R = R - R_o(s')$ is a vector approximately normal to the command path representing the distance that the cobot is off the path.
- Heading — $\Delta T = T - T_o(s')$ is a vector approximately normal to the command path representing the error in cobot heading.

[2] The "heading" of a cobot is represented by a unit tangent vector in C-space.

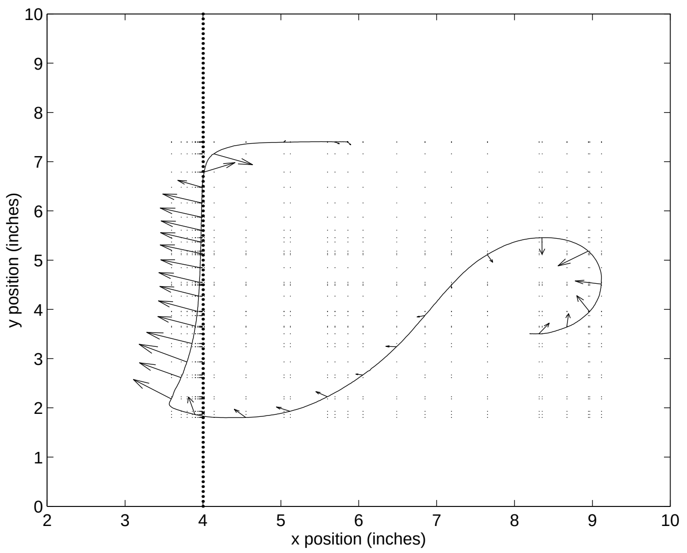
A standing assumption is that the cobot to be controlled is outfitted with the sensors necessary to measure these errors. In the case of the cobot pictured in Fig. 1, linear potentiometers provide C-space configuration (R) and a steering shaft encoder provides heading (T).

An intuitive approach to feedback control is simply to adjust the steering velocity (ω) in order to compensate for both types of errors. In fact, it is somewhat more powerful to think in terms of adjusting a curvature command (κ), because the path velocity u is solely at the discretion of the operator. Thus, we would expect that a reasonable form for the controller would be:

$$\omega = u\left[\kappa_o(s') - N_o(s')\cdot\left(\frac{G_1}{L^2}\Delta R + \frac{G_2}{L}\Delta T\right)\right] \tag{4}$$

$\kappa_o(s')$ is the feedforward term from §2.2.1. G_1 and G_2 are control gains, and L is a scale factor used to ensure consistent units. L may also be interpreted as a lookahead distance [2, 3].

Gain selection as well as the stability of this type of controller have been discussed in depth in [2]. Further discussion of these issues will be omitted here, due to space limitations. Fig. 4, however, shows recorded data in both caster and constraint modes for the unicycle cobot.



3. Control of Scooter

3.1 Scooter

Scooter, a redundant tricycle cobot, is pictured in Fig. 5. Scooter has been built primarily as a testbed for exploring the kinematics and control of higher dimensional cobots.

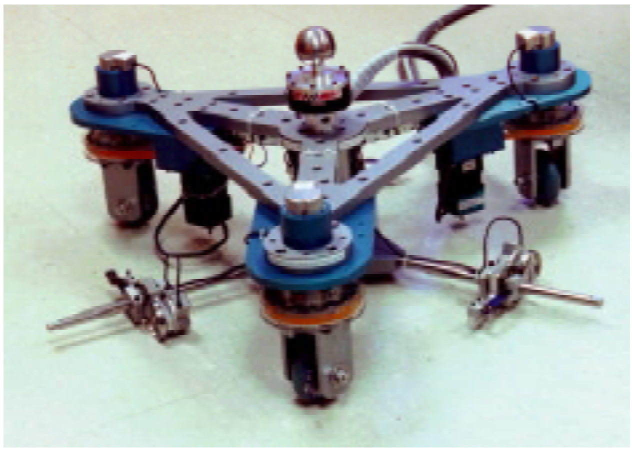


Figure 5. Scooter

The configuration space of Scooter is that of a planar rigid body ($R = [x, y, l\theta]^T$). Only two wheels are needed to produce one degree-of-freedom motion in this space; however, Scooter is outfitted with a third wheel to eliminate the need for external support, and to eliminate a singularity that occurs with only two wheels [1].

It is our assumption that, in controlling Scooter or other high dimensional cobots, it will be desirable to plan paths and constraint surfaces in C-space, rather than in the space of individual joints. Thus, the first topic we address here is kinematic transformations from C-space to joint space.

3.2 Kinematics

In the case of Scooter, the C-space configuration may be described as the position (x, y) and orientation (θ) of the handle location (in the middle of the equilateral triangle), measured relative to a known starting location in a global frame. The orientation, however, is generally scaled by a characteristic length measure, *l*, to minimize numerical difficulties. Thus, $R = [x, y, l\theta]^T$.

The path followed by wheel i can easily be derived from a knowledge of R and of the vector that runs from the handle location to the wheel steering axis. The operations involved are simply translation and rotation. Thus, it is possible to write:

$$r_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} L_{ix}(R) \\ L_{iy}(R) \end{bmatrix} = L_i(R), \quad i = 1,2,3 \tag{5}$$

Eq. 5 indicates that, unlike the unicycle cobot, Scooter exhibits non-trivial kinematics relating its C-space path to its wheel paths. Like the unicycle, however, it is necessary to steer individual wheels in order to control Scooter. This suggests that several C-space-to-joint-space kinematic transformations are necessary (one for each wheel). In the following, we develop the transformation relations necessary to relate the C-space path tangent to the joint (wheel) path tangents, and the C-space path curvature to the joint path curvatures.

3.2.1 Tangent Transformation

C-space and joint space unit tangent vectors are defined as follows (note that "joint space" here refers to the space associated with a single wheel):

$$T = \frac{dR}{ds}; \quad t_i = \frac{dr_i}{ds_i}, \quad i = 1,2\ 3 \tag{6}$$

Here, s_i is the arc length along the path traced out by wheel i. A relation between these two types of tangents can be obtained by differentiating Eq. 5 with respect to s_i:

$$t_i = \frac{dr_i}{ds_i} = \frac{\partial L_i}{\partial R}\frac{dR}{ds}\frac{ds}{ds_i} \tag{7}$$

The first term on the right hand side can be recognized as an ordinary 2×3 Jacobian, which we will name J_i, from cobot C-space to the two-dimensional path space of wheel i. The second term is the C-space tangent, and the third term is a local path length expansion/contraction ratio. This ratio is a scalar, and its magnitude is necessarily that which makes t_i a unit vector. Thus:

$$t_i = \frac{J_i\ T}{|J_i\ T|} \tag{8}$$

3.2.2 Curvature Transformation

C-space and joint space curvatures (κ, κ_i) and unit normal vectors (N, n_i) are defined as follows:

$$\kappa N = \frac{dT}{ds}; \quad \kappa_i n_i = \frac{dt_i}{ds_i}, \quad i = 1,2\ 3 \tag{9}$$

An expression for $\kappa_i n_i$ can be obtained by differentiating Eq. 8 with respect to s_i. Because Eq. 8 involves a quotient of terms, all of which depend upon s, the operations are fairly tedious, though straightforward. Details will be omitted. The result is:

$$\kappa_i n_i = \frac{I - t_i t_i^T}{|J_i T|^2}\left[\begin{bmatrix} T^T H_{ix} T \\ T^T H_{iy} T \end{bmatrix} + J_i \kappa N \right] \tag{10}$$

H_{ix} and H_{iy} are 3×3 matrices defined as:

$$H_{ix} = \frac{\partial^2 L_{ix}}{\partial R^2}; \; H_{iy} = \frac{\partial^2 L_{iy}}{\partial R^2} \tag{11}$$

They may be thought of as representing the spatial rate of change of the Jacobian, J_i. This effect, even in the absence of curvature in C-space, can result in joint space curvature (i.e., curvature of the path that a wheel traces out on the ground). The term $I - t_i t_i^T$ may be recognized as a projection matrix, which ensures that n_i will be normal to t_i.

3.3 Virtual Caster

Unlike the unicycle cobot, the apparent dof of Scooter may vary from 1 to 3 (if the redundant wheel is misaligned, even zero dof is a possibility). Although quite interesting and important in its own right, the mathematics involved in setting up 2 dof behaviors, and more importantly, switching between 1, 2 and 3 dof behaviors, goes beyond the scope of this paper. Here, we will restrict attention to a fully castered, 3 dof mode. In the next section, we will focus on 1 dof path tracking.

C-space caster control for Scooter is much like caster control for the unicycle cobot. In addition to instantaneous configuration (which figures into Jacobians, etc.), two key measurements are necessary: velocity (both magnitude, u, and direction, T), and operator-applied force ($F = [F_x, F_y, l^{-1}\tau]^T$).

In order to make the cobot behave as an unconstrained rigid body, the force measurement can be used to generate a *desired acceleration.* For instance, this acceleration might be $A = [F_x/m,\ F_y/m,\ \tau l/I]^T$, where m and I are estimates of cobot mass and moment, respectively. These estimates need not be terribly accurate. A more general interpretation of them is as caster controller gains. Moreover, this is only one prescription for generating a desired acceleration vector. More general ones involving non-diagonal inertia matrices can be readily imagined.

Once a desired acceleration, A, is available, caster control is straightforward. The idea is to use A, T and u to predict a C-space path. κN for this path is found and inserted, along with T, into Eq. 10 to produce the necessary joint space curvatures, from which wheel steering velocities are determined. The following relations between the time and path derivatives of R are useful:

$$\frac{dR}{dt} = Tu \tag{12}$$

$$\frac{d^2R}{dt^2} = A = \kappa N u^2 + T\dot{u} \tag{13}$$

Also useful is the following result, which may be found by straightforward differentiation of u:

$$\dot{u} = T^T A \tag{14}$$

Combining Eqs. 13 and 14, the C-space curvature vector is found:

$$\kappa N = \frac{1}{u^2}\,[I - TT^T]\,A \tag{15}$$

This result is pleasingly simple: the component of the desired acceleration which is normal to the C-space path is selected by the projection matrix and scaled by u^2 to produce the curvature. Eq. 10 can then be used to compute the joint space curvatures.

The steering velocity commands are each the product of a curvature and a speed. The applicable speed is, of course, the path speed at the joint, u_i. It is readily shown that:

$$t_i u_i = J_i T u \qquad i = 1,2,3 \tag{16}$$

As a final step, it must be recognized that, in the case of Scooter, the steering motors are mounted in the moving frame of Scooter rather than the ground frame. Thus, the steering velocity commands must be adjusted for the rotational velocity of Scooter:

$$\omega_i = \left| J_i T u \times \frac{I - t_i t_i^T}{|J_i T|^2} \left[\begin{bmatrix} T^T H_{ix} T \\ T^T H_{iy} T \end{bmatrix} + J_i \frac{1}{u^2}[I - TT^T]\,A \right] \right| - \dot{\theta} \qquad i = 1,2,3 \tag{17}$$

3.4 Path Tracking

In the above discussion, we saw that C-space caster control of Scooter is in large part analogous to caster control of the unicycle cobot: a desired normal acceleration is determined, and used to command steering velocities. It will be shown in this section that the analogy is equally strong in the case of constraint tracking. Moreover, we are motivated to investigate constraint tracking in C-space by the belief that path planning will be more naturally accomplished in C-space than in joint space. As with the unicycle, we will break the discussion into feedforward and feedback control.

3.4.1 Feedforward Control

Feedforward control is developed on the basis of ideal path tracking. In this case, the cobot heading, T, and curvature vector, κN, are assumed to be identical to those of the command path (T_o and $\kappa_o N_o$), and the cobot is assumed to be on the path. As before, the command path is assumed to be parameterized in terms of its own path length. The feedforward command for each joint can then be determined by using Eqs. 10 and 16, and following the procedure decribed in the previous section.

3.4.2 Feedback Control

The control policy that corrects for configuration and heading errors has two components. These components are analogous to those for the unicycle cobot, however, rather than make adjustments to the scalar curvature, it becomes necessary to make adjustments to the curvature vector (i.e., both κ and N):

1. s is replaced with s', where $R_o(s')$ is the closest point on the command path to the cobot's actual configuration (R).
2. κN is replaced with $\kappa_o(s')N_o(s') + \delta(\kappa N)$, where $\delta(\kappa N)$ is determined on the basis of the configuration and heading errors, and is intended to steer Scooter back toward the command path in C-space.

s' continues to be specified according to Eq. 3, and when the cobot configuration is compared to $R_o(s')$ there continue to be two types of errors: displacement, ΔR, and heading, ΔT. These vectors, while approximately normal to the path, will generally *not* be parallel to $N_o(s')$. Thus, the correction to be made will not be simply a scaling of the curvature vector, but also a redirecting of this vector.

A reasonable form for $\delta(\kappa N)$ would be similar to the curvature correction employed for unicycle control (Eq. 4):

$$\delta(\kappa N) = -\frac{G_1}{L^2}\Delta R - \frac{G_2}{L}\Delta T \tag{18}$$

Somewhat greater insight into Eq. 4 can be obtained by considering the terms individually. Let's begin with the displacement term. Associated with it is a circle that is tangent to T and lies in the plane common to T and ΔR. The magnitude of this term also determines the circle's curvature. These points are illustrated in Fig. 6. Note that $N_o(s')$ does not generally lie in the plane of the circle.

This circle may be considered the instantaneous "return path" associated with the displacement term. Indeed, the circle will intersect the command path tangent, $T_o(s')$, at a "return distance" of $L(2/G_1)^{1/2}$.

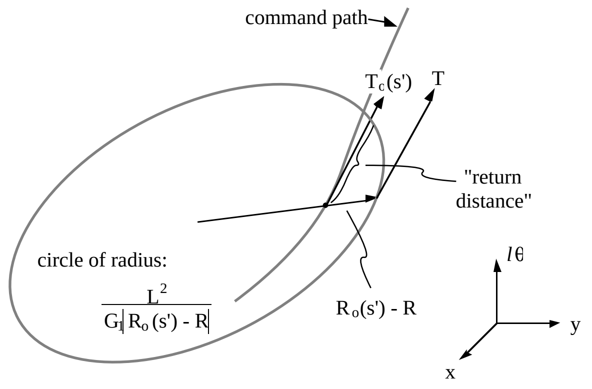


Figure 6. Geometry of circular "return path" induced by displacement term of feedback controller.

There is a similar circle associated with the heading term, and it produces a "return distance" of $2L/G_2$. If we make the gain selection $G_1 = G_2 = 2$, then it is appropriate to speak of L as a "lookahead distance", i.e., a characteristic C-space path length for error recovery.

Returning now to the design of the constraint tracking controller, it is necessary only to introduce the curvature correction (Eq. 18) into the transformation relation (Eq. 10), and follow the procedure developed for caster control. The resulting control law is:

$$\omega_i = \Bigg| J_i T u \times \frac{I - t_i t_i^T}{|J_i T|^2}\left[\begin{bmatrix} T^T H_{ix} T \\ T^T H_{iy} T \end{bmatrix} + J_i\left(\kappa_o(s')N_o(s') + \delta(\kappa N)\right)\right]\Bigg| - \dot{\theta} \qquad i = 1,2,3 \tag{19}$$

A discussion of gain selection and stability will be deferred to future publications. These matters, as well as an experimental investigation involving Scooter, are the subject of current work.

4. Conclusions

The fundamentals of virtual caster and constraint tracking control of cobots have been introduced. A key concept is that of a path in cobot C-space parameterized by path length, and having tangent and curvature vectors. In addition, tangent and curvature transformations between C-space and joint space are seen to play a key role. The joint space of a cobot is unlike that of a conventional robot: associated with each cobot joint is a two-dimensional path (e.g., the path that a wheel traces out on the ground).

It is interesting to note that most of the ideas that have been introduced here apply to cobots of higher dimensional C-space, and even cobots which employ steerable nonholonomic joints other than wheels (such as those discussed in [4]).

For more information about cobots, the reader is invited to visit http://www.ece.nwu.edu/~peshkin/cobot/.

Acknowledgements

The authors gratefully acknowledge the support of the GM Foundation and the National Science Foundation. We are particularly grateful for the intellectual input of Prasad Akella, and all the members of the cobot group at Northwestern.

References

[1] Colgate, J. E., M. A. Peshkin and W. Wannasuphoprasit. *Nonholonomic Haptic Display.* IEEE International Conference on Robotics and Automation. Minneapolis. pp. 539-544, 1996.

[2] Colgate, J. E., W. Wannasuphoprasit and M. A. Peshkin. *Cobots: Robots for Collaboration with Human Operators.* International Mechanical Engineering Congress and Exposition. Atlanta. pp. 433-440, ASME, 1996.

[3] Ollero, A. and G. Heredia. *Stability Analysis of Mobile Robot Path Tracking.* IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Pittsburgh, PA. pp. 461-466, 1995.

[4] Peshkin, M., J. E. Colgate and C. Moore. *Constraint Machines Based on Continuously Variable Transmissions, for Haptic Interaction with People.* IEEE International Conference on Robotics and Automation. pp. 551-556, 1996.