Cobots: Robots for Collaboration with Human Operators
J. Edward Colgate, Witaya Wannasuphoprasit, Michael A. Peshkin
Proceedings of the International Mechanical Engineering Congress and Exhibition,
Atlanta, GA, DSC-Vol. 58, pp. 433-39

COBOTS: ROBOTS FOR COLLABORATION WITH HUMAN OPERATORS

J. Edward Colgate **Witaya Wannasuphoprasit**

Michael A. Peshkin

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

ABSTRACT

A "cobot" is a robotic device which manipulates objects in collaboration with a human operator. A cobot provides assistance to the human operator by setting up virtual surfaces which can be used to constrain and guide motion. While conventional servo-actuated haptic displays may be used in this way also, an important distinction is that, while haptic displays are active devices which can supply energy to the human operator, cobots are intrinsically passive. This is because cobots do not use servos to implement constraint, but instead employ "steerable" nonholonomic joints. As a consequence of their passivity, cobots are potentially well-suited to safety-critical tasks (e.g. surgery) or those which involve large interaction forces (e.g. automobile assembly). This paper focuses on the simplest possible cobot, which has only a single joint (a steerable wheel). Two control modes of this "unicycle cobot", termed "virtual caster" and "virtual wall" control, are developed in detail. Experimental results are also presented.

1. INTRODUCTION

Haptic displays, which are essentially robots designed for direct, physical interaction with human operators, have a great variety of applications. These range from teleoperation, to virtual reality, to robotic surgery. One of the most exciting capabilities of haptic displays is the implementation of *programmable constraint*. For example, Rosenberg (1994) has shown that "haptic virtual fixtures" (hard walls which constrain motion to useful directions) can dramatically improve performance in teleoperation tasks such as remote peg-in-hole insertion. Another example comes from Kelley and Salcudean (1994) who describe the "Magic Mouse", a computer interface device which can constrain an operator's hand to useful directions while interacting with a GUI (to avoid, for instance, "slipping off" a pull-down menu). A third example is a robotic surgery system in which a robot positions a guide for a tool held by a surgeon, and a fourth is automobile assembly in which programmed constraints can help an operator navigate large components (e.g., instrument panels, spare tires, seats, doors) into place without collisions.

These examples have two rather clear commonalities. One, they all involve constraining the motion of a human operator. Two, the source of energy for carrying out the task is the human operator. Related to the latter is a less obvious point: in all cases, the behavior of the haptic display is, ideally, energetically *passive*. Passivity of the haptic display plays an important role in ensuring the stability of the overall system (including the operator), and the safety of the human operator (Colgate and Brown, 1994). Experience has shown, however, that when using conventional approaches to haptic display, constraint and passivity are antagonistic goals. This is because conventional approaches employ servo control to reduce the degrees of freedom (d.o.f.) of a multi-d.o.f. robot to those consistent with the programmed constraint. To implement an effective (perceptually rigid) constraint, a servo controller requires high gains which are incompatible with passivity. While there has been considerable progress made in designing haptic displays which admit high gains (Colgate and Brown, 1994), the problem described is inherent to the servo control approach.

One way around this tradeoff is to use controllable brakes rather than (or in addition to) servoed actuators at the joints of the robot (Russo and Tadros, 1992). Brakes can implement very hard constraints and are completely passive. Brakes, however, suffer from one very serious drawback, illustrated with a simple example in Fig. (1). In this example, a two-axis Cartesian haptic display is contemplated. Here, the brakes are assumed to completely prohibit motion when turned on, or completely allow it when turned off. It is accordingly simple to implement a wall in the y-direction by braking the x-axis, and vice versa. There are, of course, subtleties in practice. For instance, walls are usually unilateral, and therefore force sensing is needed to determine when the display is being pulled away from the wall, so that the brake can be turned off. Subtleties aside, however, there is a much more serious difficulty. Suppose that one wishes to implement a wall at a 45° angle, as illustrated. The only way to achieve this is to approximate the 45°

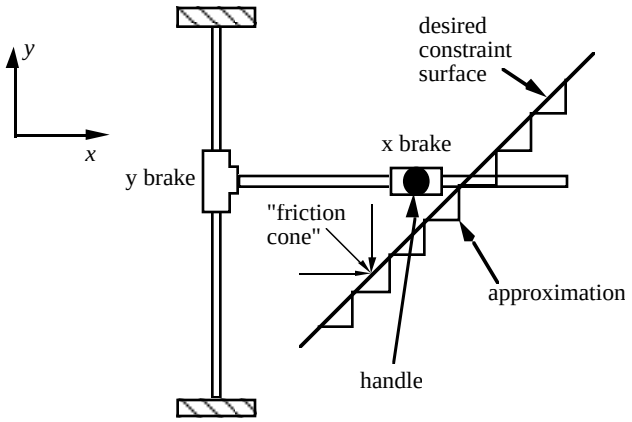


Figure 1. A passive haptic display using brakes. Forces applied within a 90° cone centered about the surface normal will result in a stuck mechanism.

smooth wall with a series of steps. The user is certain to perceive these steps. Moreover, this wall exhibits a behavior not unlike friction: any force in a 90° cone angle centered about the wall's outward normal will result in both brakes being activated, and the mechanism becoming stuck.

Book and his students have made considerable progress in using brakes by improving upon the control algorithm (braking force is made proportional to the velocity, not just the configuration) and by adding clutches which couple the degrees of freedom (Charles, 1994, Davis, 1996). Nonetheless, there is some question of how well this approach extends to higher numbers of d.o.f. (two d.o.f. require four brakes/clutches and a differential). Moreover, some degree of "jaggedness" seems inherent to the approach.

Delnondedieu and Troccaz (1995) describe an energetically passive manipulator named "PADyC" which uses overrunning clutches rather than brakes. At each joint are two such clutches, each of which runs on a motor-driven drum. One drum rotates clockwise and the other counterclockwise. The rotational speeds of these drums determine the maximum clockwise and counterclockwise joint angular velocities which an operator can generate without engaging a clutch. Thus, an operator is effectively speed-limited in the joint space. As in the example discussed above, however, limited directions of constraint are available so that achieving a smooth feel is an inherently difficult problem.

In this paper, we discuss a new approach to implementing programmable constraint which produces smooth, hard, passive constraints. The basic idea is diametrically opposed to the conventional approach: we begin with a mechanism that has either zero or one d.o.f., and use feedback control to make it behave as though is has additional d.o.f., as necessary to be consistent with the programmed constraint. The key to implementing this strategy is the use of nonholonomic joints.

In the next section, the concept of a cobot is briefly introduced. A more detailed discussion is available in (Colgate, et al., 1996). In Sections 3-5, aspects of cobot control are discussed and experimental results are presented. In Section 6, our ongoing work is briefly described.

2. UNICYCLE AND BICYCLE COBOTS

Figure 2 is a photograph of a prototype cobot. This is a unicycle device which consists of a single steerable wheel that rolls on the plane (a horizontal surface). The wheel is held upright by a passive mechanism, and a human operator grabs onto and pushes a handle mounted directly above the unicycle shaft. There are two modes of operation:

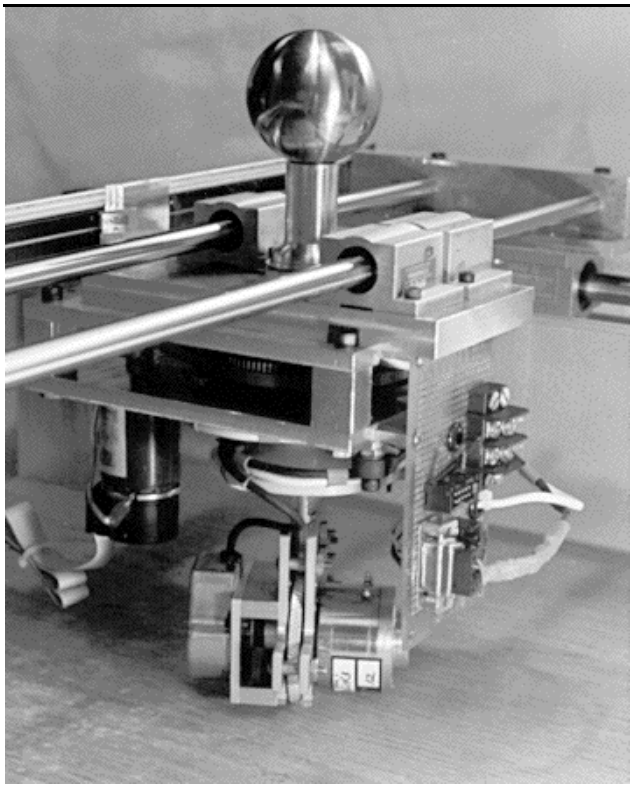


Figure 2. Photograph of the unicycle cobot prototype. Components that can be seen include the handle; the xy frame, instrumented with linear potentiometers; the steering motor and transmission; and the wheel assembly including a high resolution encoder and a particle brake (not in use, currently).

Mode 1: "Virtual Caster" In this mode the wheel acts like a caster so that it doesn't constrain motion at all. The wheel is not a caster in the conventional sense. Instead, it has a straight-up shaft like a unicycle, but this shaft is instrumented with a force sensor. If the sensor detects forces perpendicular to the wheel's rolling direction, the wheel is steered (by a motor) to minimize these forces. In effect, the wheel turns so that it can roll in the direction it is pushed, and so, from the user's point of view, it is like a free particle which he or she can move around the plane at will.

Mode 2:"Virtual Wall" When the user moves the shaft to the edge of the free region (to the constraint surface), the computer which controls the steering motor no longer does so in such a way as to minimize force. Instead, the steering motor is used to turn the wheel so that its rolling direction is tangential to the constraint. The force sensor mentioned above still monitors forces perpendicular to the wheel. If the forces would tend to push the wheel into the constraint, they are ignored. If the forces would tend to pull the wheel off of the constraint, they are interpreted just as in the free space mode. This means that it is impossible to push the unicycle past a virtual

constraint (unless the wheel slips), but that the unicycle can easily be pulled off of the constraint surface.

This machine has some interesting and desirable characteristics. First, although it is a one d.o.f. device (the wheel fixes the ratio of x and y velocities), in the virtual caster mode, it behaves 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. It is important to note that motorized steering of a conventional offset caster would not be passive: this is the reason that we have chosen to design a virtual caster.

For the virtual caster and virtual wall behaviors to succeed, the steering control system must be carefully conceived. For instance, for virtual caster operation, it is important that the control system be able to keep lateral forces on the wheel nulled regardless of operator behavior. This problem will be discussed further in the next section. For virtual wall operation, it is important that the absolute location and orientation of the wheel be known at all times. It is possible to achieve this by starting motion in a known location, measuring wheel speed and direction, and integrating. This approach, however, is not robust to wheel slip. An alternative is to measure the absolute position of the device directly. A variety of technologies exist for doing this. For the prototype pictured in Fig. (2), we have outfitted the planar kinematic mechanism that holds the wheel upright with position sensors. This mechanism also serves to absorb reaction forces generated by the steering motor.

A unicycle cobot can constrain motion in x and y, but it cannot constrain orientation. In many applications (e.g., robot-assisted surgery) orientation is very important. A "bicycle cobot", shown in Fig. (3), can implement x, y, and angular constraint. This machine consists of two independently steerable wheels whose shafts are held a fixed distance from one another. Both are controlled in a manner comparable to that described for the unicycle cobot.

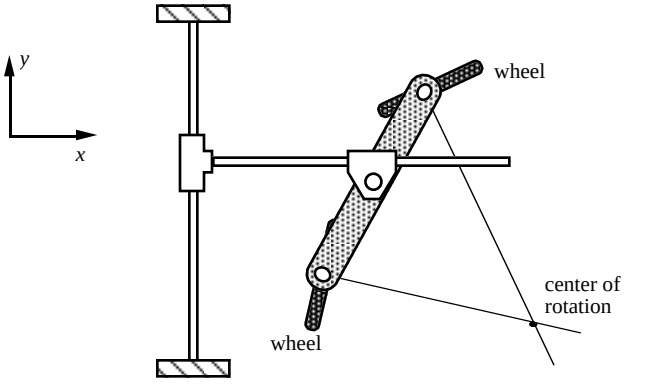


Figure 3. Bicycle PCM. The operator grasps a handle that protrudes from any point on the plate connecting the two wheels.

With this example we can begin to see that, like other robotic mechanisms, cobots exhibit singularities. Any motion of the bicycle cobot can be viewed as a rotation about the instantaneous center of rotation (see Fig. (3)). It is not possible, however, to specify a center of rotation on the line that passes through the two wheel shafts. If we attempt to do so, the two wheels will both be aimed perpendicular to this line. In this configuration, the cobot actually gains a degree of freedom, going from one to two (of course, we usually think of singularities as reducing the d.o.f.).

One way to solve this problem is to add a third wheel whose shaft is not collinear with the other two. This would also have the benefit of making the machine statically stable, eliminating the need for a frame. The design of higher d.o.f. constraint machines is discussed in (Peshkin, et al., 1996).

3. THE VIRTUAL CASTER

3.1 Theory

Proper operation of the virtual caster is obviously the key to the concept we have described: without it, we cannot *add* degrees of freedom to a nonholonomically constrained device. In this section, we discuss virtual caster control for the unicycle cobot.

The ideal caster controller would perceptually eliminate the wheel. In other words, a user manipulating the machine would perceive it to be a single rigid body. In the case of a unicycle machine, it is useful to think of that body as a point mass. 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 velocity ω, has an instantaneous normal acceleration of $a_{\perp} = u\omega$. Thus, we can obtain a prescription for the steering velocity which would result in particle-like behavior:

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

Equation 1 indicates that the problem of virtual caster control is fundamentally nonlinear: 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. Equation 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.

3.2 Implementation

The unicycle cobot pictured in Fig. (2) was used to collect the data presented in this paper. The unicycle is supported upright by an xy frame which is instrumented for position. The unicycle assembly includes a dc motor for steering; an optical encoder that measures steering angle; another optical encoder that measures wheel rotation (this is used to derive translational velocity, u); and a handle-mounted sensor that measures x and y components of the user-applied force (this is not pictured in Fig. (2)). Feedback control is implemented on a Pentium computer, at a controller update rate of 1 kHz.

The caster controller follows the form of Eq. (1), but is modified for torque control and finite sensor resolution. Because the steering motor is torque controlled, an "inner" loop is first closed around steering velocity, ω. Due to the high update rate, however, this controller and the "outer" steering angle controller are in fact implemented together.

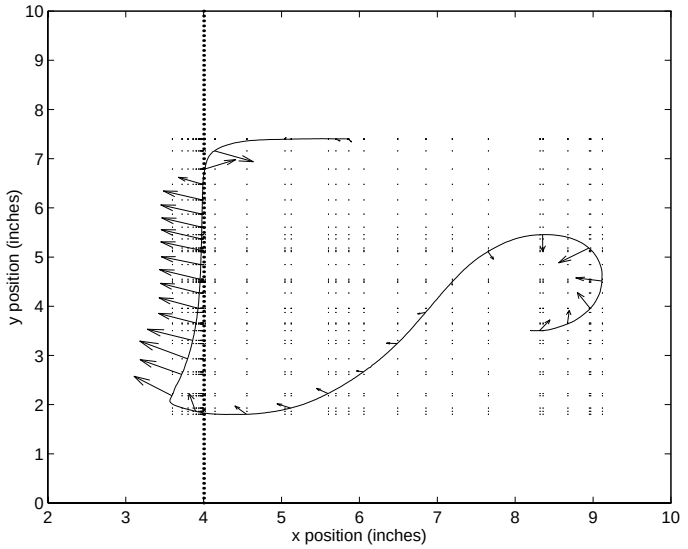
Due to finite sensor resolution and the singularity at zero translational velocity, the denominator of Eq. (1) must also be

modified to prevent overflow, excessively large control signals, and instability. The form of the virtual caster controller which has been implemented is:

$$\Gamma = \frac{K_1 F_\perp}{\text{sign}(u)\ \max(|u|, \varepsilon\ \text{sign}(u))} - K_2\omega \qquad (2)$$

Γ is the motor torque, K_1 is an adjustable gain which replaces $1/M$ in Eq. (1), and K_2 is a gain associated with the steering velocity controller. ε is an adjustable parameter which places a lower limit on the denominator magnitude (ε is of the same order as the velocity resolution). These gains are adjusted for performance and stability. u and ω are estimated by digital differentiation and digital filtering of the associated angular measures. $F_\perp$ is computed based on the measured x and y components of the user-applied force and the steering angle.

Figure 4 displays a set of experimental data. The cobot trajectory (curved line) and, at selected points, the operator-applied force, are shown for both virtual caster and virtual wall operation. During virtual caster operation, the force remains quite small except when performing fairly sharp, low-speed changes in direction.



4.1 Theory

The problem of tracking a virtual constraint is very similar to that of steering an autonomous vehicle along a predetermined path. Because the latter problem has been studied in considerable depth, there is an ample literature to follow. Our derivation follows relatively closely that of Ollero and Heredia (1995). As in their paper, we will consider the case of bilateral constraint rather than unilateral constraint (virtual wall) tracking. A bilateral constraint is easily made unilateral with a software switch as discussed in Section 5.

The basic idea is quite simple: the nominal steering velocity, ω, should be set to u/ρ, where ρ is the instantaneous radius of curvature of the constraint; in addition, ω should be corrected appropriately whenever the cobot strays off of the constraint. To derive a control law which implements such corrections, it is first helpful to explore the dynamics of cobotic path tracking.

Several assumptions will be made to simplify the dynamic analysis. First, it will be assumed that the path to be tracked is circular with radius of curvature ρ. Second, it will be assumed that the cobot is being pushed at a constant translational speed, $u = u_o$. Third, it will be assumed that the steering velocity controller has first order dynamics with a time constant τ.

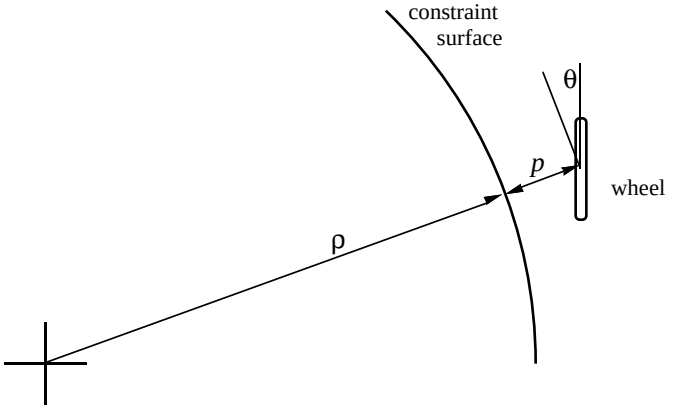


Figure 5. Kinematics of constraint tracking

Under these conditions, the state of the unicycle is described by its "penetration", p, normal to the constraint surface, its "out-of-tangency", θ, and its steering velocity, ω, as illustrated in Fig. (5). The state equations are easily shown to be:

$$\frac{dp}{dt} = u_o \sin\theta \qquad (3a)$$

$$\frac{d\theta}{dt} = \frac{u_o \cos\theta}{\rho + p} - \omega \qquad (3b)$$

$$\frac{d\omega}{dt} = \frac{1}{\tau}(\omega_c - \omega) \qquad (3c)$$

The first two equations stem directly from the kinematics shown in Fig. (5), and the third equation from the assumption of first order steering dynamics. ω_c is the commanded steering velocity.

Because the translational velocity is assumed constant, it is possible (and somewhat more convenient) to express the dynamics in terms of arc length, s, rather than time. If we define the instantaneous curvature as $\kappa = \omega/u_o$, the equations may be written as:

$$\frac{dp}{ds} = \sin\theta \qquad (4a)$$

$$\frac{d\theta}{ds} = \frac{\cos\theta}{\rho + p} - \kappa \qquad (4b)$$

$$\frac{d\kappa}{ds} = \frac{1}{u_o\tau}(\kappa_c - \kappa) \tag{4c}$$

A final modification involves the representation of curvature. It is useful to replace κ with the "curvature error", $\tilde{\kappa} = \kappa - \rho^{-1}$. The commanded curvature is likewise altered, resulting in:

$$\frac{dp}{ds} = \sin\theta \tag{5a}$$

$$\frac{d\theta}{ds} = \frac{\cos\theta}{\rho + p} - \frac{1}{\rho} - \tilde{\kappa} \tag{5b}$$

$$\frac{d\tilde{\kappa}}{ds} = \frac{1}{u_o\tau}(\tilde{\kappa}_c - \tilde{\kappa}) \tag{5c}$$

Now we turn to the matter of control. The nominal controller for this system would simply be $\tilde{\kappa}_c = 0$. As is intuitively clear, however, the resulting controller is not stable (in the sense that it will not asymptotically track the circular path), because it in no way accounts for errors. As an aside, it would be possible to verify this instability simply by linearizing Eq. (5a-c) about the desired equilibrium state (0,0,0), and solving for the eigenvalues. We will take this approach below.

A straight forward way to derive a stabilizing controller is simply to include terms proportional to the penetration and out-of-tangency errors. Greater insight can be gained, however, by envisioning a modified path that the unicycle could follow to return to course whenever errors occur. This is illustrated in Fig. (6). The modified path is determined by the instantaneous errors (p, θ) and a "lookahead distance", L, (or, equivalently, a lookahead arc angle, ϕ) which describes the distance allotted for recovery. Stabilization can be achieved by replacing the nominal controller with a command based on the curvature of the modified trajectory.

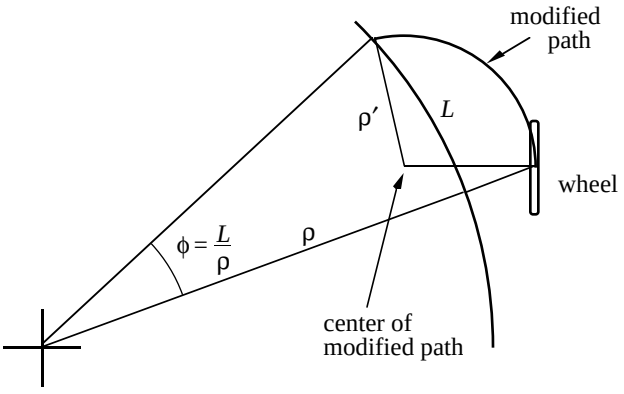


Figure 6. A modified path for correction of tracking errors. The modified path is tangent to and passes through the center of the wheel. It passes through the original path after an arclength of L.

If the modified path is taken to be circular, the following expression for $\kappa_c = 1/\rho'$ can be found after some straight forward geometry:

$$\kappa_c = \frac{p\cos\theta + \rho(\sin\phi\sin\theta + (1 - \cos\phi)\cos\theta)}{0.5p^2 + (1 - \cos\phi)(\rho^2 + p\rho)} \tag{6}$$

By further making the assumption that p and θ are small, and obtaining the Taylor series expansion of Eq. (6), the following linear controller can be found:

$$\tilde{\kappa}_c = \frac{\cos\phi}{\rho^2(1 - \cos\phi)}p + \frac{\sin\phi}{\rho(1 - \cos\phi)}\theta \tag{7}$$

This control law shows, as intuition would suggest, terms proportional to p and to θ. It additionally, however, gives us some guidance in selecting the gains associated with these terms. In the case of small ϕ (lookahead distance small relative to the radius of curvature), the control law simplifies even further:

$$\tilde{\kappa}_c = \frac{2}{L^2}p + \frac{2}{L}\theta \tag{8}$$

Here it is easily seen that the lookahead distance is essentially a gain parameter: the smaller L, the higher the gain.

As a next step, it is useful to insert the control law (7) into the state equations (5a-c), and study stability. Lyapunov's second method can be applied by linearizing the resulting equations (5c is linear, 5a,b are not) and investigating pole locations. The characteristic equation of the linearized system is:

$$\lambda^3 + u\tau\lambda^2 + \left[\frac{\sin\phi}{u\tau\rho(1 - \cos\phi)} + \frac{1}{\rho^2}\right]\lambda + \left[\frac{\cos\phi}{u\tau\rho^2(1 - \cos\phi)} + \frac{1}{u\tau\rho^2}\right] = 0 \tag{9}$$

The Routh-Hurwitz method can be used to find the conditions under which all roots lie in the left half plane. The condition is:

$$\frac{u\tau}{\rho} < \tan\phi \tag{10}$$

In the case of small ϕ this reduces to:

$$u\tau < L \tag{11}$$

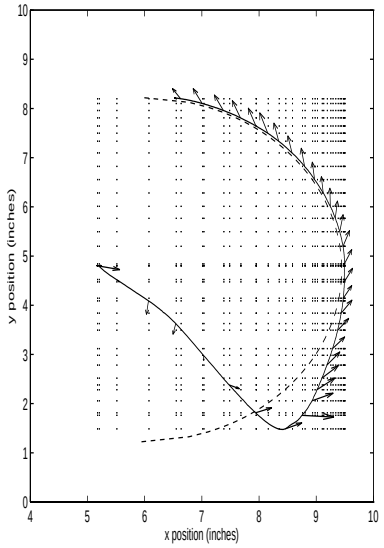
This relation suggests that the steering controller should be as responsive as possible (small τ), the translational speed should be small (or at least below some threshold), and the lookahead distance should be as large as possible. Of course, it is desirable to keep the lookahead distance well below the radius of curvature, ρ. As L approaches ρ, two of the poles approach the imaginary axis and path tracking becomes quite poor.

4.2 Implementation

Both straight and circular virtual walls have been implemented using the control law in Eq. (8). In addition, we have found that somewhat better performance can be obtained by tuning the gain parameters for penetration and out-of-tangency independently. This is neither surprising nor troubling, as the analysis above was

based on a circular modified path, whereas other shapes (e.g. polynomial) can be readily defined and will give rise to different gain weightings.

Typical behavior is illustrated in Fig. (4) and Fig. (7). In both cases, it is seen that, after an initial penetration (an issue discussed further in Section 5), the cobot approaches the constraint surface in a smooth and stable fashion. Both cases, however, also exhibit some steady state error (amounting to about 0.01 inches). This is possibly the result of the wheel's intrinsic slip angle. In the case of circular constraint, the error has been minimized by modestly increasing the wheel's nominal steering velocity command. A more effective approach may be to implement a form of integral control. This is currently under investigation.



5.1 Theory

The virtual caster and constraint tracking controllers discussed in the previous sections are the basic elements of a virtual wall controller. A virtual wall can be implemented by switching between the two controllers as follows:

- Switch from caster to constraint when a "collision" is detected (i.e., when cobot crosses boundary of virtual wall);
- Switch from constraint to caster when operator-applied force points away from virutal wall.

A limitation of this simple strategy, however, is that the wheel must reorient upon collision with a virtual wall. In the worst case, the reorientation would involve a 90° turn. During the time required to execute such a turn, fairly deep wall penetration may well occur.

This problem may be avoided by monitoring impending collisions, and beginning to turn the wheel before the collision occurs. As a conceptual framework for this, the "constraint overlay" illustrated in Fig. (8) is quite useful. The constraint overlay is a circular sector which is tangent to both the wheel and the constraint surface. When the radius of this sector falls below a pre-determined minimum, it becomes active, "overlaying" the existing constraint.

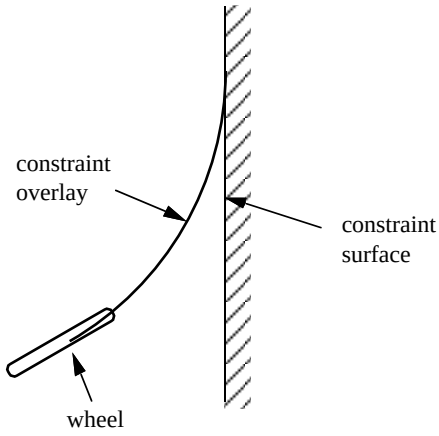
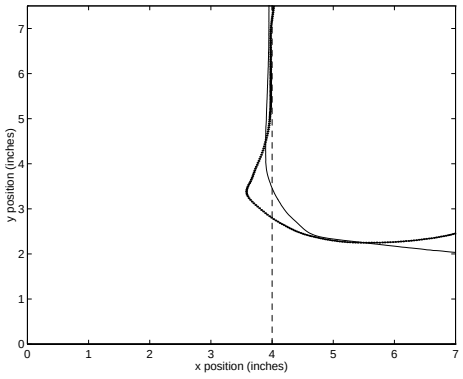


Figure 8. Constraint overlay

5.2 Implementation

In Fig. (9), wall strike trajectories are illustrated with and without constraint overlay. As is evident, the constraint overlay initiates the turn prior to collision and minimizes the penetration that occurs.



used only for steering, and not for direct interaction. Thus, a current focus of our research is on using cobots for comanipulation of a large components (such as seats, wheels, instrument panels, etc.) in automobile assembly. We are currently developing several cobotic manipulators, including a three-wheeled device, a joystick, and a serial-link device.

ACKNOWLEDGEMENTS

This work has been supported by a grant from the GM Foundation. We are particularly grateful for the support of Prasad Akella, Nick Caruso and Steve Holland. Thanks also go to Brent Gillespie for many stimulating discussions of cobot control and for suggesting the term “cobot”.

REFERENCES

Charles, R. A., 1994, "The Development of the Passive Trajectory Enhancing Robot," Master of Science in Mechanical Engineering, Georgia Institute of Technology.

Colgate, J. E. and Brown, J. M., 1994, "Factors Affecting the Z-width of a Haptic Display," *International Conference on Robotics and Automation*, IEEE R&A Society, San Diego, CA, Vol. 4, pp. 3205-10.

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

Davis, H. T., Jr., 1996, "An Investigation of Passive Actuation for Trajectory Control," Master of Science in Mechanical Engineering, Georgia Institute of Technology.

Delnondedieu, Y. and Troccaz, J., 1995, "PADyC: a Passive Arm with Dynamic Constraints; a prototype with two degrees of freedom," *Medical Robotics and Computer Assisted Surgery*, IEEE, Baltimore, Maryland, pp. 173-180.

Kelley, A. J. and Salcudean, S. E., 1994, "On the Development of a Force-Feedback Mouse and Its Integration Into a Graphical User Interface," *International Mechanical Engineering Congress and Exposition*, C. J. Radcliffe, ed., ASME, Chicago, Vol. DSC 55-1, pp. 287-294.

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

Peshkin, M., Colgate, J. E. and Moore, C., 1996, "Constraint Machines Based on Continuously Variable Transmissions, for Haptic Interaction with People," *IEEE International Conference on Robotics and Automation*, , Vol. 1, pp. 551-556.

Rosenberg, L. B., 1994, "Virtual Fixtures: Perceptual overlays enhance operator performance in telepresence tasks," Ph.D. dissertation, Stanford Univ., Dept. of Mechanical Engineering.

Russo, M. and Tadros, A., 1992, "Controlling Dissipative Magnetic Particle Brakes in Force Reflective Devices," *ASME Winter Annual Meeting*, H. Kazerooni, ed., Anaheim, California, pp. 63-70.