NORTHWESTERN UNIVERSITY

COBOTS : COLLABORATIVE ROBOTS

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the Degree

DOCTOR OF PHILOSOPHY

Field of Mechanical Engineering

By

Witaya Wannasuphoprasit

EVANSTON, ILLINOIS

June 1999

CHAPTER 1

INTRODUCTION

1.1 Motivation

Humans have the unique ability to adjust, correct, and perform tasks within uncertain environments. For example, in automobile assembly where there is positioning variation for each vehicle body, a human operator can easily insert and tighten a screw in the vehicle body even on the moving line. The human moreover recognizes vehicle models and colors and inserts the right kind of screw or other component. Perhaps the most important feature of human operator is that he or she can handle unexpected problems. This because humans have the senses of touch, hearing, feel, and vision, and of course the intelligence to act purposefully on all this sensory input. To perform a similar task, robots would certainly require extensive sensors and complicated control algorithms, none of which is cost effective today. Nevertheless, humans do have limitations: limited strength, physical fatigue, and metal fatigue. Humans are prone to injuries, especially those arising from repetitive stress. In addition, training is usually required when a human performs a new task.

This thesis is in the area of human-robot collaboration. The main objective of this research is to combine human intelligence with strengths of robots. This involves direct physical interaction between a human and a robot in a shared workspace. While the human serves mainly

to provide sensory input and intelligent decision-making, the robot provides assistance such as gravity compensation, "*inertia management*[1]" (i.e., guiding the motion of large loads), and guidance via *virtual surfaces*.

This new class of collaborative robots is known in the automotive industry as "Intelligent Assist Devices" (IADs). IADs respond to the need for improved ergonomics, productivity, and flexibility (programmability) in automobile manufacturing. One of the major challenges in IAD development is to provide solutions to inertia management and guidance, which are intuitive to use, yet intrinsically safe.

1.2 Introduction

Recently human-robot collaboration research has grown significantly, especially in the decade of the 90s. One exciting research area has been *haptic interface*, which later will be reviewed. A *haptic interface* is a device that lets a user touch, feel and manipulate a virtual environment. A good sampling of recent work in this field can be found in [8,13,16,17,25], which covers topics ranging from mechanical design to real-time simulation, to psychophysics and neurophysiology. Applications of *haptic interface* include teleoperation, rapid prototyping, computer-assisted surgery, and training.

One of key features of haptic interface is an ability to implement *virtual surfaces*. Virtual surfaces are software defined, computer generated, constraint surfaces. Virtual walls do not physically exist, but the human operator perceives and feels them as if they were quite real. The

[1] A problem associated with change in speed (momentum) and direction of awkward loads.

virtual surface concept has a number of applications. For example, Rosenberg [25] used *virtual fixtures* to improve efficiency in teleoperation. In assembly processes, *virtual surfaces* such as *virtual funnels* can guide operators to the target faster and prevent unwanted collisions. In computer assisted surgery, a surgeon may perform an operation while virtual walls prevents inadvertent damage to sensitive tissue. In short, the benefits of virtual surfaces include increasing in productivity, improving quality, and improving ergonomics.

Current haptic interface approaches for implementing virtual surfaces rely on active joint actuation. Creating smooth, frictionless, and hard (stiff) virtual surfaces requires high gain servo control. As pointed out by Colgate and Brown [5], humans can readily induce instability in such systems. As a consequence, safety is a big concern, especially with high power systems, as might be required in automobile assembly applications.

For this reason, haptic interfaces that are energetically passive have great appeal. One way to build a passive interface is to use controllable brakes rather than motors at the joints; however, this turns out to have some shortcomings for virtual surface implementation. As will be explained later in detail, the virtual walls implemented by brakes are limited in the orientations they may take on. Brakes also absorb energy. This makes them less appealing for the implementation of frictionless walls because they do not conserve momentum.

A related approach is “PADyC”, introduced by Delnondedieu and Troccaz [30]. PADyC is a manipulator using overrunning clutches. 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 that 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 work, we present *cobots (collaborative robots),* a novel technology for intelligent assist devices, haptic interface, human-robot collaboration, and human-robot interaction. Cobots were specifically conceived for the implementation of virtual surfaces.

Cobots are a new class of robotic devices, intended to work with human operator in a share workspace. 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 interfaces may be used in this way also, an important distinction is that, while haptic interfaces are active devices that 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).

In the remainder of this chapter, we explain virtual surfaces in detail. Then, we review haptic interfaces. Finally, we provide an overview of the rest of the thesis at the end of the chapter.

1.3 Virtual Surfaces

In this section we present *virtual surfaces* as a tool for effectively assisting human operators to perform tasks with less operating force, in less time, with lower ergonomic stress, and with fewer errors.

A virtual surface is a software-defined construct that, via a haptic device, takes on the properties of a physical constraint. The virtual surface does not physically exist in the sense that we normally understand surfaces to exist, but a human operator should perceive it as if it were a real surface.

There are several applications of virtual surfaces. For example, Rosenberg [25] 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 [16] 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). Yet another is a robotic surgery system in which a robot positions a guide (a constraint) for a tool held by a surgeon.

Another important application for virtual surfaces is to assist a human operator in an assembly process, especially when manipulating a heavy and bulky load. In some automobile assembly operations, for instance, a human operator is required to transfer a heavy component and assemble it in a vehicle. Even with current *assist devices* such as pneumatic balancers and overhead rail systems, the human operator may have to exert considerable force to manipulate the load. Part of the reason for this is that existing systems do little to aid *inertia management*—change in direction and speed of large loads—or to help overcome friction. One solution is to use

a virtual surface such as a *virtual funnel*, which directs workpart motion to a specific task location. The virtual funnel serves two purposes: to prevent collisions during the process, and to provide guidance to the operator (who can slide the workpart along the virtual surface).

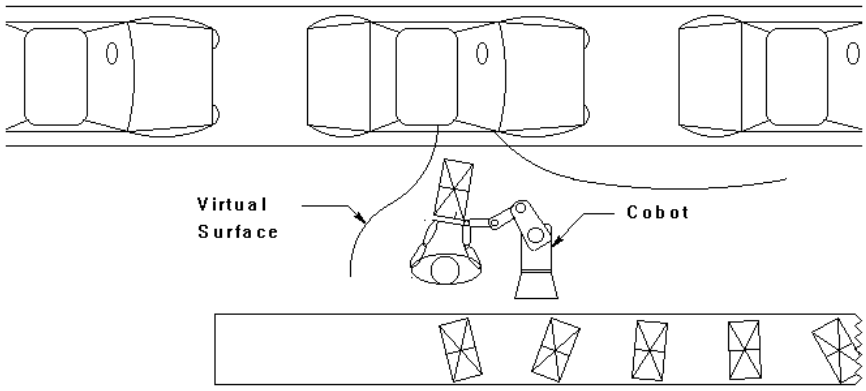


Fig 1.1. Picture of human and cobot assembly a part through virtual funnel (drawing courtesy of Carl Moore)

The advantages of a virtual surface are significant. Virtual funnels increase productivity according to Fitt's Law, which may be paraphrased as "when you hit a bigger target you can hit it faster". Rejection rates are reduced due to the collision free operation. In addition, overall ergonomic conditions improve because the operator uses a virtual surface for guidance and thus requires smaller manipulating forces. The other benefit is flexibility, since virtual surfaces are programmable. One advantage of this is that several vehicle models can be assembled on the same line. Moreover, the training time for a new operator is reduced.

1.4 Virtual Surfaces via Haptic Interfaces

Conventional approaches to haptic interfaces [2,13,15,17] usually rely on actuators to generate forces during interaction with people. Because of this safety is a troublesome issue. Consider the seemingly simple task of simulating a smooth and hard virtual wall. An effective simulation, which lets the human operator perceive a real physical constraint, requires a high gain servo controller with sizable actuators. While there has been significant progress made in the design of haptic interfaces that admit high gain [5], some problems inherent to the high gain controllers still exist, such as the trade-off between controller performance and stability robustness. Stability is not usually compatible with a high gain system. An unstable haptic display is potentially an unsafe device. In addition to this problem, hardware or software failures may result in injury.

For the above reasons, haptic interface designs that are energetically passive have great appeal. A passive design is intrinsically safe. It cannot strike the operator or support unstable oscillations. An example of a passive haptic interface is one that uses controllable brakes. For instance, Russo and Tadros [26] couple magnetic particle brakes to the joints of a manipulator. Brakes are completely passive and can simulate hard constraint. This approach seems to be a good candidate, but there are some problems. Since brakes are passive; i.e. they cannot actively generate force, they can implement only those constraints which align with their axes of motion. Additionally, for unilateral constraints such as a wall, a force sensor is needed to determine when the brake is to be turned on or off. The former can be shown by considering the example in Fig 1.2. Simulation of a wall in the x or y direction can be implemented easily. For an arbitrary direction, however, brakes can result in a stuck mechanism (this happens when both brakes are turned on). Suppose that the wall is implemented at 45 degrees as illustrated. Since the wall is

not aligned with the brake directions, one way to simulate this wall is to approximate it with a series of steps. If the handle (at position A) is pushed with a force vector that lies in region 1, the y brake is turned on and the handle moves rightward in the x direction (position B). Then the y brake is turned off and the x brake is turned on. Now the handle moves upward to position C. If the force is still applied, the handle will move to positions D, E, and so on. Thus, any force in region 1 or 3 will intermittently slide the handle up or down along the step-like wall. However, for any force in region 2, both brakes must be activated to prevent penetration. This results in a stuck mechanism.

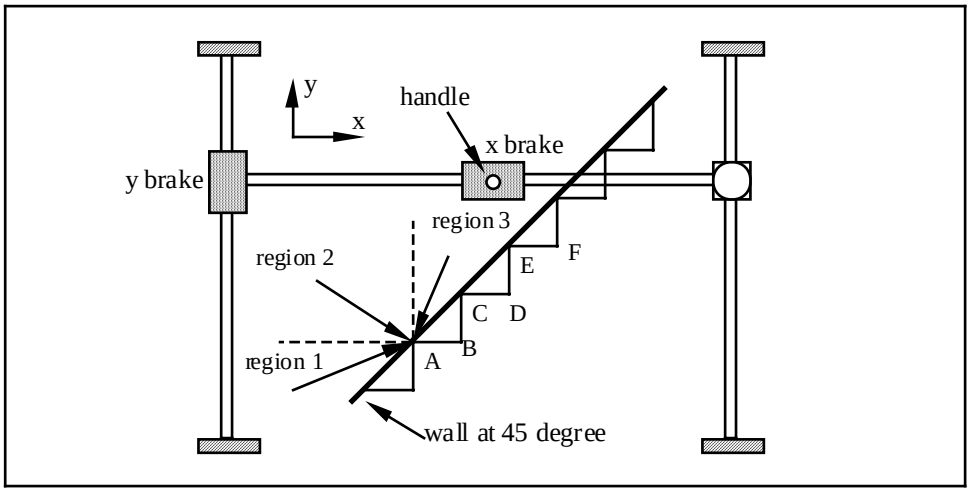


Fig 1.2. x and y brakes supported by H frame exhibit a wall at 45 degree

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 [4]. Nonetheless, there is some question of how well this approach extends to higher numbers of degrees-of-

freedom (two d.o.f. requires 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 that 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.

1.5 Cobots: Collaborative Robots

In this work, we present *cobots*, *collaborative robots*, a novel approach to implementing *virtual constraints*. In the physical world, constraints act to reduce the degrees of freedom of a mechanical system. Similarly, in conventional haptic interface, servo control reduces the apparent degrees of freedom to implement the virtual constraint. Here, however we use the opposite approach. Beginning with a mechanism that has either one or zero DOF, we use feedback control to *increase* the degrees of freedom. This mechanism has a number of DOF less than the dimension of the configuration space. In *free* mode, the mechanism behaves as if it has full degrees of freedom, while in *constraint* mode it loses some of those degrees of freedom. The key to implementing this strategy is the use of nonholonomic joints. By using a nonholonomic

mechanism for haptic interface, it turns out that we can effectively implement smooth, hard, passive virtual surfaces.

Cobots are quite different from conventional robots in both hardware architecture and control algorithm. A traditional robot actuates each joint by a servomotor, which is usually equipped with a position sensor such as an encoder. In general, robot paths are planned in the task space. Then the controller transforms task space parameters into the joint space via the robot's Jacobian and issues a torque command to each joint. Cobots on the other hand do not control each joint individually, but control the relation between them. More specifically, cobots control the ratio of joint velocities between any two joints.

Consider a 2 DOF Cartesian robot shown Fig 1.3, The robot consists of an x-y frame equipped with position sensors (not shown). Each axis is powered by a servomotor via a timing belt mechanism. A force sensor, attached to the handle, is used to measure the operator-applied force.

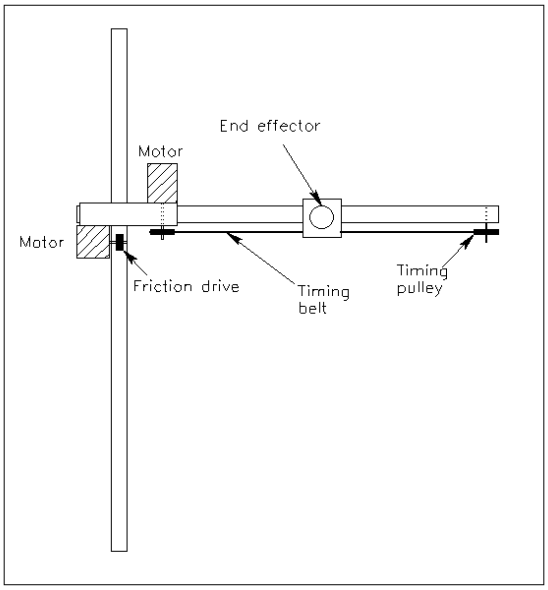


Fig 1.3. A 2 DOF Cartesian robot

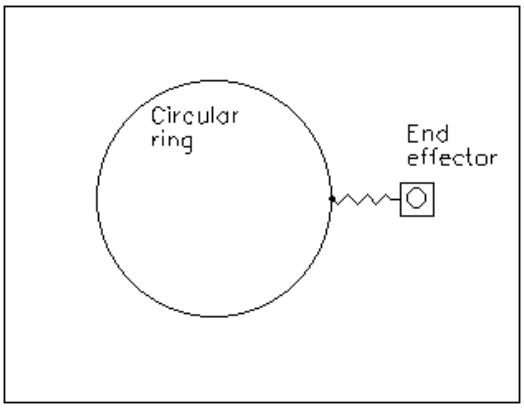


Fig 1.4. A circular virtual path simulated by the robot shown in Fig 1.3.

In order to simulate a circular path, for example, we can compute motor commands using a stiff virtual spring, which connects the end effector to a circular ring (depicted in Fig1.4). Thus the command for the motors can be written as:

$$\begin{bmatrix} f_x \\ f_y \end{bmatrix} = \begin{bmatrix} k\,d_x \\ k\,d_y \end{bmatrix}, \qquad 1.1$$

where f is force command in taskspace, d_x *and* d_y are positioning errors, and k is the spring stiffness. Subscripts x and y are associated with the x and y axes.

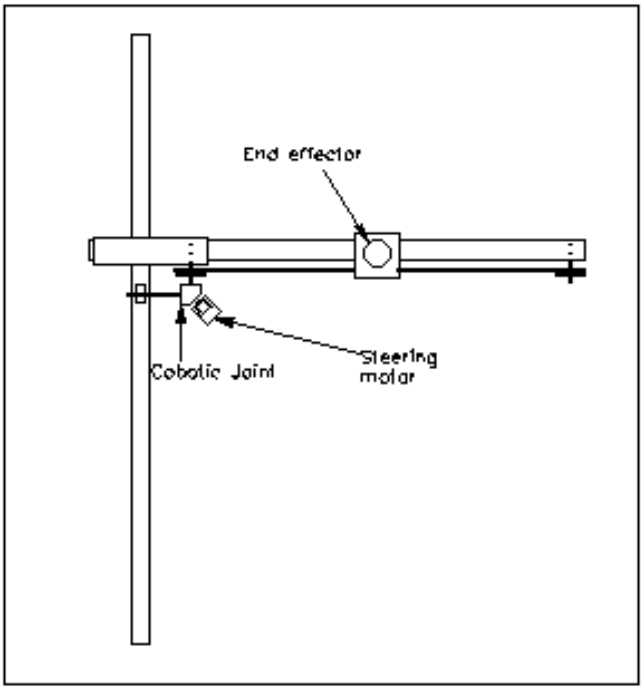


Fig 1.5. A Cartesian cobot

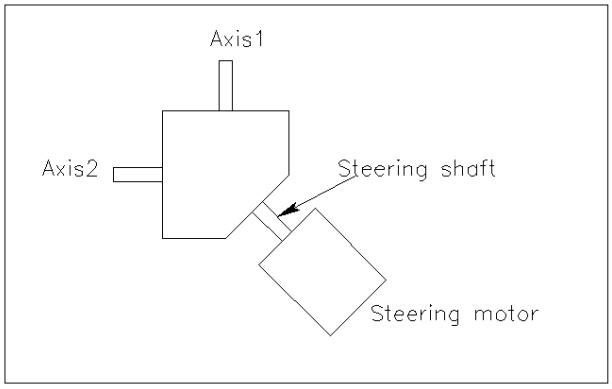


Fig 1.6. A Cobotic joint

Now consider a Cartesian cobot shown in Fig 1.5. The axes are not powered by motors but are coupled together by a so-called "*cobotic joint*". Referring to Fig 1.6, a *cobotic joint* is a servo *steered* joint, which consists of two input shafts (axes 1 and 2) and one steering shaft. The steering shaft is steered by a small servo motor. The inputs are mechanically coupled together and velocity ratio between them is set and can be varied by changing the steering shaft position. This velocity ratio is continuously adjustable and can take on any value. For this reason, cobotic joints can be viewed as members of the well know class of transmission elements called *CVTs*, *Continuously Variable Transmissions*. The details of cobotic joints and cobot architecture are explained in Chapter 2.

In this case the velocity relationship between the two axes can be written as

$$\frac{v_y}{v_x} = m = f(\alpha_s), \qquad 1.2$$

where v_x and v_y are velocity in the x and y axes respectively, m is a transmission ratio, and α_s is the steering angular position.

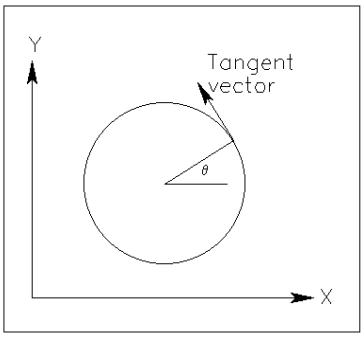


Figure 1.7. A circular path.

To implement a virtual circular path, the cobot controls only one parameter, *T*, which, in a sense, is the path tangent. Referring to Fig 1.7, the tangent vector (unit vector) of the circular path can be written as

$$T = \cos(\theta)\, y - \sin(\theta)\, x \qquad 1.3$$

To recapitulate, the robot keeps the operator on a virtual surface by generating reaction forces. The cobot, on the other hand, keeps the operator on a virtual path by steering. The cobot, in essence, "*redirects*" reaction forces. Since the direction of the tangent vector is orthogonal to the path, steering power can not be transmitted to the human operator. This makes the cobot inherently passive. In addition, the cobot requires less overall power and a smaller number of actuators. In the example above, the cobot would need only one small motor (less than 20 watts in practice). Consider, as an example, an operator handling a 25 kg load and moving in a 50 cm diameter circle with a constant velocity of 2 m/s. This task would require an 800 watt robot compared to a 20 watt cobot.

1.6 Thesis Overview

Chapter 2 describes cobotic joints and the conceptual designs of cobot architecture. Section 2.1 presents the two major types of cobotic joints: wheel and spherical joints. Section 2.2.1 introduces a cobot with a remarkably simple architecture (1 DOF and a 2 dimensional C-space) using a wheel joint, called a *unicycle* cobot. Section 2.2.2 describes a two-wheel cobot, which can be used to implement virtual surfaces in a planar configuration space (x, y, and θ). Although it has two wheels, the cobot naturally has 1 DOF motion, which can be described as a rotation around a point where the axes of both wheels intersect. As will be shown, like most other robotic mechanisms, the two-wheel machine exhibits a singularity. An additional wheel is introduced in Section 2.2.3 and shown to solve the singularity problem. Another benefit of the third wheel is that the device is self-supporting. The device still has 1 DOF when all three wheel axes intersect at the same point (instantaneous center of rotation). Section 2.2.4 explains an *extreme joystick* cobot and Section 2.2.5 describes a wheel joint cobot with a higher dimensional configuration space. In Section 2.2.6, we present an arm-like cobot using spherical joints. Some cobot architectures require transmission mechanisms, which also increase friction to the system. *Power assist* can be added to help the operator overcome friction. Section 2.3 briefly discusses *power assist* cobots in general.

To explore cobotic concepts, we built a prototype of the *unicycle* cobot. Chapter 3 provides details of the unicycle cobot design and construction. Kinematics and control of the unicycle are also presented as an introduction to a more general, higher DOF, cobot control in

Chapter 6. We describe control in three modes: free mode, path tracking mode, and virtual surface mode. In addition, we provide experiment results.

Based on promising results obtained with the unicycle cobot, we designed and constructed *Scooter*, a three-wheel cobot, as a testbed to explore cobot kinematics and control in higher DOF. Chapter 5 describes the design and construction in detail. One key design issue for Scooter is the velocity and global position sensing technique. We implemented an odometry technique via *glide wheel* sensors. Section 5.4 explains the velocity estimation algorithm in detail. Section 5.5 describes a feedback control called COR agreement control that guarantees 1 DOF behavior of Scooter.

Chapter 6 presents cobot control more generally. We develop kinematic transformations for cobots and describe these in detail in Section 6.2. Section 6.2 to Section 6.4 present control modes of operation including free mode, path tracking mode, and virtual surface mode. Experimental results are provided as well. We, in addition, discuss analogies between robot and cobot controls.

With a view toward practical implementation of cobots, we have extended our efforts from basic research to industrial applications. Chapter 7 presents two industrial cobot prototypes. Under co-development of Northwestern University and General Motors Corporations, we built the first industrial floor-based cobot as a successor to Scooter. The application for this cobot is to remove a door from a vehicle in an assembly line. In order to avoid damage (i.e scratch paint, collision), the task requires a very accurate approach and escape trajectory with a specific orientation. The door unloader cobot illustrates remarkable performance. It reduces cycle time by

50% and requires only a few pounds of force to maneuver. We also briefly described an overhead rail cobot prototype, which is currently under evaluation at Ford Motor Company.

The last chapter concludes and discusses all work presented in this thesis, as well as proposing future work. This includes global position research for floor-based cobots, joint velocity sensing, development of improved cobotic joints, path and virtual surface generation, and optimal control to minimize user input forces.

CHAPTER 2

CONCEPTUAL DESIGNS FOR COBOT ARCHITECTURES

In this chapter, we present a variety of potential cobot architectures. The chapter begins (Section 2.1) with a review of the two principal types of cobot joints, the steered wheel on a plane, and the spherical CVT. Section 2.2 then presents a number of passive cobot architectures based on these two joints.

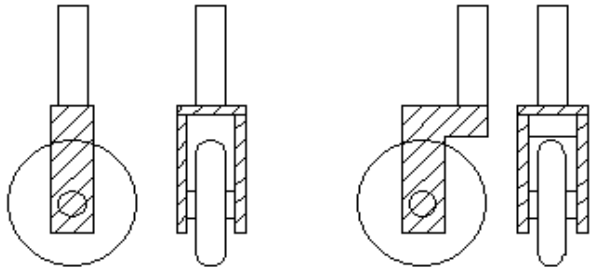
By way of introduction, Section 2.2.1 describes the conceptual design of a unicycle cobot. The unicycle cobot has 1 DOF motion in an x-y configuration space. It consists of a single steerable wheel that rolls on a horizontal surface. Since the unicycle cobot can not support itself upright, a Cartesian rail system is provided to restrict the machine to planar motion. The unicycle cobot can constrain translational motion, but not orientation. Section 2.2.2 develops the design of a two-wheel cobot. Despite having two wheels, the bicycle cobot has 1 DOF motion. We show that the motion of this device can be viewed as a rotation around an instantaneous center. When both wheels are parallel, however, the machine exhibits a singularity. In section 2.2.3, this problem is solved by adding a third wheel. Another benefit of the third wheel is that the machine becomes self supporting. Section 2.2.5 briefly presents conceptual designs of higher dimensional C-Space cobots using wheel joints. In section 2.2.6, a conceptual design of an arm-like cobot using spherical CVTs is presented.

All cobots described in Section 2.2 are intrinsically passive and can not move themselves. In some application there is a need for a *power assist*. For example, an arm-like cobot can not support itself against gravity. Depending on the cobot architecture, a *power assist* can be effectively added to the cobots. Section 2.3 describes a general architecture for *power assist* cobots. Further details on cobot architecture can be found in Peshkin et al [24].

2.1 Cobotic Joints

2.1.1 Translational Joints

Cobot joints are examples of nonholonomic joints. In other words, they establish non-integrable relationships between velocity variables. Due to non-integrability, the constraint imposed in velocity space does not imply a similar constraint in configuration space. The cobotic wheel joint illustrates this clearly.



Cobotic wheel joint Caster wheel

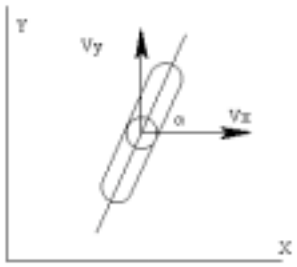


Figure 2.1 Cobotic wheel joint and caster wheel

The cobotic wheel joint is shown in Fig. 2.1. It consists of a steering shaft, wheel axis shaft and a wheel. The wheel rolls freely about the wheel axis shaft. The wheel heading direction is controlled by a servomotor (not shown in the picture) attached to the steering shaft. Unlike a caster (Fig 2.1 top right) there is no offset between the wheel axis and the steering shaft axis. This is a very important concept, because the torque generated by the servomotor cannot transmit power to the rolling motion of the wheel. The joint actuator, therefore, cannot drive joint motion – it can only adjust the available direction of motion. Because of this, we say that the cobot joint is intrinsically passive.

Associated with the wheel is a 2 dimensional configuration space – the plane on which it rolls (having *x* and *y* coordinates). However, the wheel has only one DOF; that is, only a one-dimensional velocity space. At any instant in time, the wheel can be moved only in its rolling direction. This is a consequence of the nonholonomic constraint imposed by the wheel-plane interface.

The wheel's direction couples *x* and *y* translational velocities. Referring to Fig 2.1, the steering angle α sets a transmission ratio between these two translational axes. The velocity relation between the *x* and *y* axes can be described as

$$v_y/v_x = tan\alpha \qquad 2.1$$

Since ratio $v_y : v_x$ can be set continuously from -1 to 1, the wheel joint is classified as a *continuously variable transmission* (CVT). Because it relates two translational velocities, the wheel is sometimes referred to as a "translational CVT" or a "translational joint". The translational joint plays an important role for planar cobots. Cobot architectures using translational joints are presented in Sections 2.2.1, 2.2.2, 2.2.3, and 2.2.4.

2.1.2 Rotational Joints

In addition to the wheel joint, another interesting cobotic joint is the spherical CVT illustrated in Fig. 2.2. Unlike the wheel joint, which relates *translational* velocities, the rotational CVT relates *angular* velocities. The relation between the angular velocity of two shafts can be varied by turning a steering roller axis. Spherical CVTs are essential for revolute serial link or arm-like cobots.

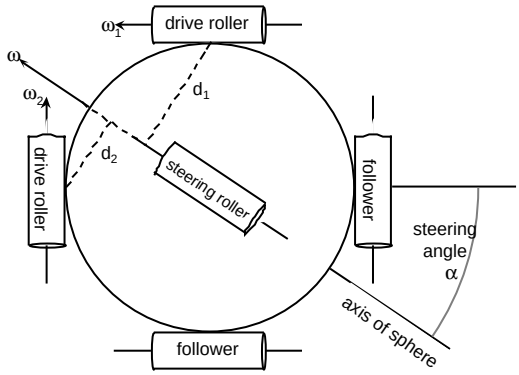


Fig 2.2. Cobotic spherical joint

As shown in Fig 2.2, the spherical joint has six rollers preloaded around a sphere. (In practice, only four rollers are used (Peshkin et al., 1996)). Two of them, the **drive rollers,** are connected to the revolute shafts whose angular velocities are to be related. Two **follower rollers** are used only to confine and preload the sphere (and are absent in the four roller design, see Fig 2.3). On the top and the bottom of the sphere are two **steering rollers**. These two rollers are mechanically connected together (not shown in the picture), so that both of them are at all times steered to the same angle.

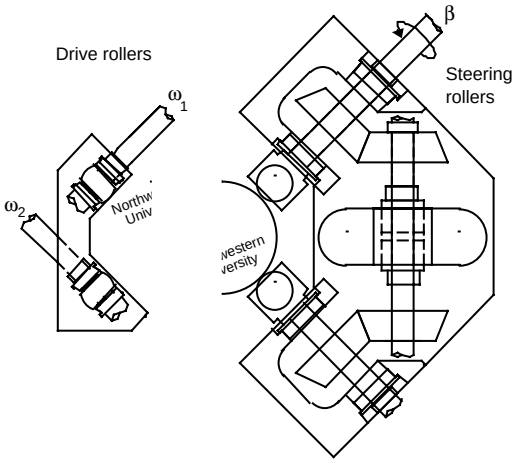


Fig 2.3. CVT spherical joint (drawing courtesy of Carl Moore)

Rolling contact constraints require that the sphere's axis of rotation must be in the same plane as the roller axes. The drive and follower rollers form a common plane (parallel to the paper), and the steering rollers form the other plane (normal to the paper). The sphere's axis of rotation (NOT shown in Fig. 2.3) is the line where these two planes intersect. From geometry, one may find

$$\omega_2 / \omega_1 = d_2 / d_1, \qquad 2.2$$

$$\text{or } \omega_2 / \omega_1 = \tan(\alpha). \qquad 2.3$$

For further details on the spherical CVT, we recommend the reader to the work of Peshkin, Colgate, and Moore [23].

2.2 Conceptual Designs of Cobot Architectures

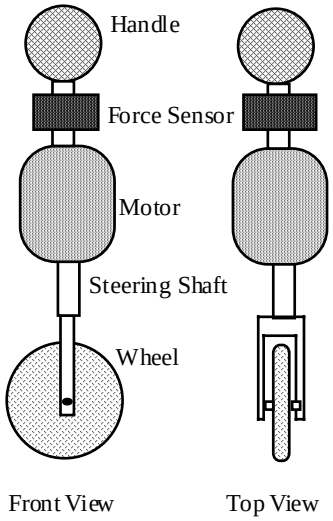


Figure 2.4. Unicycle cobot

2.2.1 Unicycle Cobot

The simplest cobot is a unicycle cobot. The unicycle cobot (Fig 2.4) is built upon a single translational joint, and like that joint has 1 DOF and a 2 dimensional *C*-space. In addition to the translational joint, the unicycle cobot has a handle mounted directly above the steering shaft, and an *x*-*y* force sensor mounted below the handle. In addition, to keep the wheel assembly upright, the unicycle cobot incorporates a gantry-style support structure. The gantry frame is also equipped with position sensors that provide an absolute position of the machine. The position of the machine may also be calculated by odometry (integrating along the wheel's path), however, this approach is not robust to the wheel slip. Fig 2.5 illustrates a unicycle machine with a Cartesian gantry frame.

Because of it simplicity, we built this first cobot as a test bed to explore cobot kinematics and control. Chapter 3 presents the unicycle cobot prototype in detail.

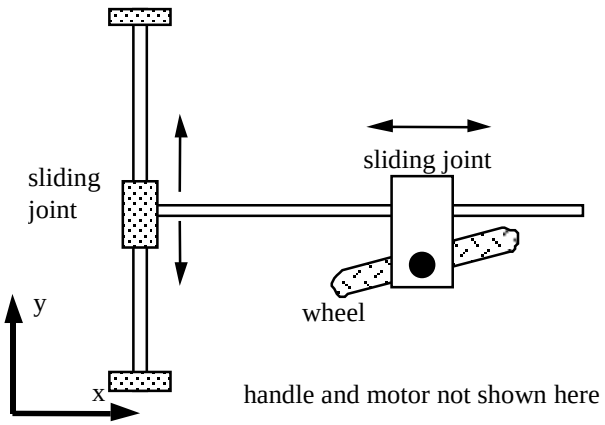


Figure 2.5. Unicycle machine with Cartesian frame

2.2.2 Bicycle Cobot

A unicycle cobot can constrain translations, but it cannot constrain orientation. In many areas of application, orientation is very important. For this purpose, a two-wheel cobot is introduced (Fig 2.6). The bicycle cobot has two independently steerable wheels whose shafts are held a fixed distance from one another. Like the unicycle cobot, the bicycle cobot cannot support itself. Thus a Cartesian frame similar to that in Section 2.2.1 may be used to support the machine and allow only planar motion. The machine has one DOF, but a three dimensional C-space (x, y, and θ). As shown in Fig 2.6, the single DOF motion can be described as a rotation about an instantaneous center of rotation. By steering the two wheels, that center of rotation can be placed anywhere in the plane.

This bicycle cobot exhibits a singularity when a center of rotation is specified on the line that passes through both steering shafts. Both wheels will be steered perpendicular to this line. In

this configuration, the machine can move translationally along the wheel direction, or rotate around any point in that line. Thus, in its singular configuration, the machine has two DOF. This problem can be solved by adding a third wheel whose shaft is not in-line with the other two.

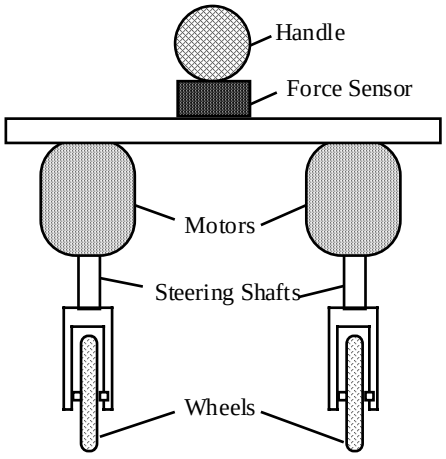


Figure 2.6. Bicycle cobot

Section 2.2.3 describes a tricycle cobot in more detail.

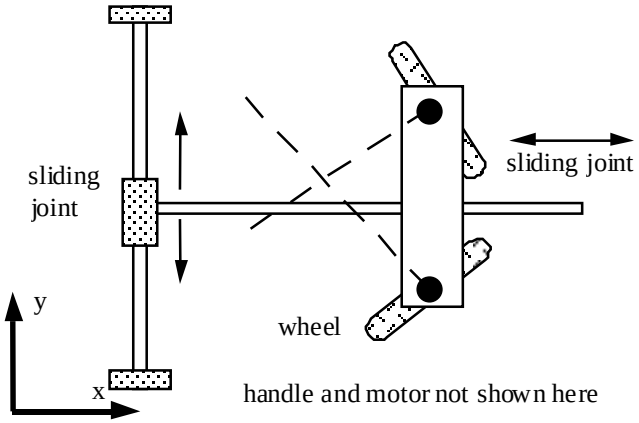


Figure 2.7. Bicycle Cobot

2.2.3 Tricycle Cobot

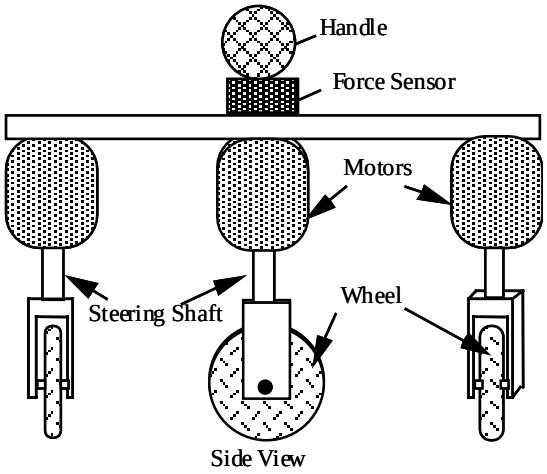


Fig 2.8. Tricycle Cobot

A Tricycle cobot is a planar redundant cobot (3 joints in 3 *C* space). It has a three dimensional C-space (x,y,θ). A third wheel not only eliminates the singularity of the bicycle cobot, but it has the practical advantage of allowing the cobot to stand on its own wheels, thus eliminating the need for a frame. As illustrated in Fig 2.8, the three wheel units connect together via a triangular base. Depending on the axle alignment, the machine has either one or zero DOF. For instance, the machine as shown in Fig 2.9a has one DOF. This motion can be viewed as a rotation about the instantaneous center of rotation (the point where all wheel axes intersect.). The cobot in Fig 2.9b has zero DOF because there is no center of rotation.

Even when the intent is to have alignment at a common point, there is typically a small error triangle generated by the axes. By actively minimizing the size of the error triangle, the cobot behavior closely approximates 1 DOF.

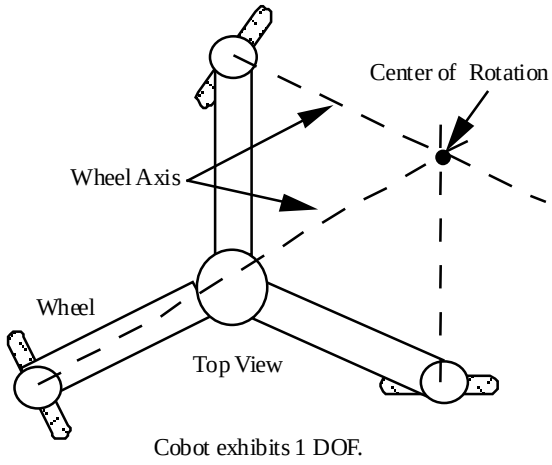


Figure 2.9a. 1 DOF Tricycle Cobot

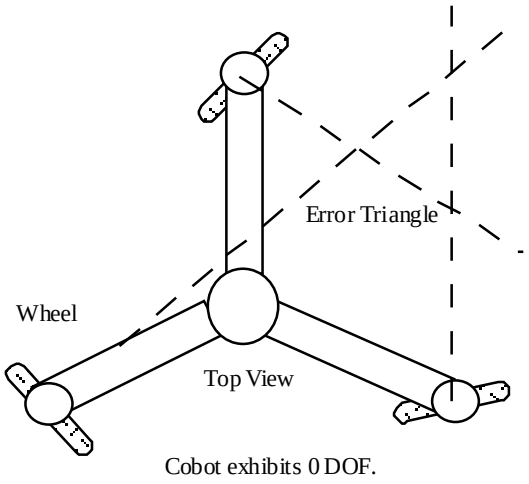


Figure 2.9b. 0 DOF Tricycle Cobot

Even though the mechanism of the tricycle cobot is relatively simple, there are several design issues. For instance, since the cobot requires continuous rotation of the steering shafts, sensing wheel-rolling velocity imposes a challenge.

We built a tricycle cobot prototype as a testbed to explore high-DOF cobotic kinematics and controls. The details of the prototype will present in Chapter 5.

2.2.4 Extreme Joystick Cobot

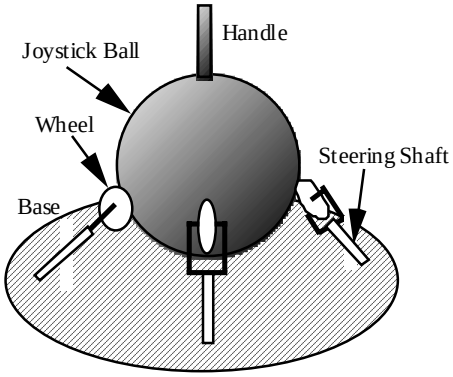


Figure 2.10. Joystick Cobot

The joystick cobot (Ref [23]) pictured in Fig. 2.10 is also a three-wheel cobot. It consists of three servo-steered wheel units, a joystick ball, a force sensor, and a handle. It has a three dimensional work space (θ, α, ψ), all rotations. Unlike the tricycle cobot, all steering wheel assemblies are fixed to a grounded base. The steering shafts can be rotated but not translated. The moving parts are the handle and the joystick ball. Similar to the tricycle cobot, the joystick cobot also exhibits a singularity if all wheel planes (the planes that pass through the wheel axes) intersect at a common axis in space. The extreme joystick cobot is currently under development (see Fig 2.11). The details of the extreme joystick cobot can be found in Ref [28].

Figure 2.11. Picture of extreme joystick cobot (pitcure courtesy of Julio Santos Munne')

2.2.5 A Higher Dimensional C-Space Cobot

It is difficult to expand the wheel-based cobot to a higher dimensional C-space, but it is not impossible. Peshkin, Colgate, and Moore [23] presented a six dimensional C-space cobot. The six dimensional C-space parallel manipulator, involving a Merlet platform geometry, is shown in Fig 2.12. It consists of a central cylindrical shell which is free to rotate, six wheel joints, six push rods, six legs, and the end-effector platform.

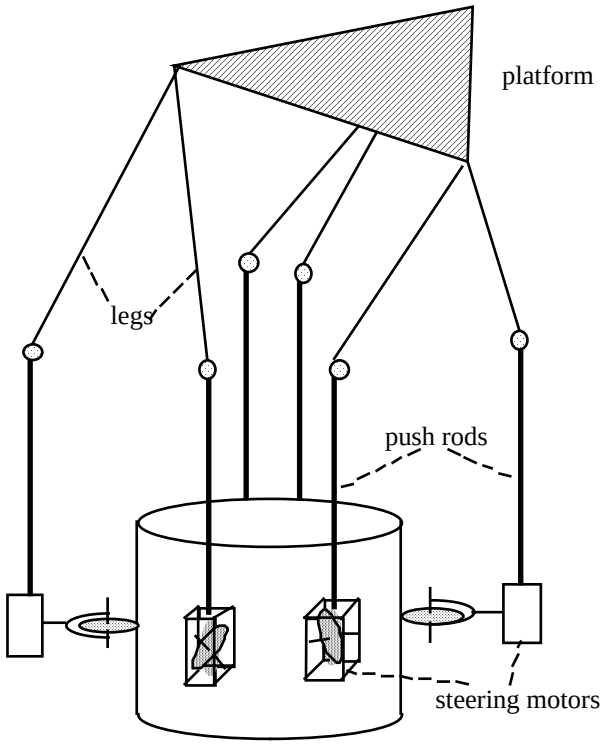


Fig 2.12. A 6 C-space Cobot

2.2.6 Serial Link or Armlike Cobot

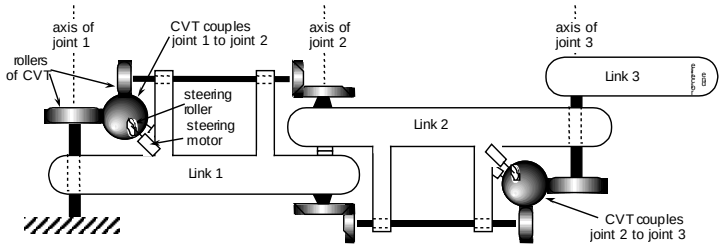


Figure 2.13. A serial link cobot. Steering axis and motors not shown here

In the previous sections we described several cobot architectures based on translational joints. This section reviewed a serial link cobot [23] using rotational joints based on the spherical

CVT. Figure 2.13 shows an arm-like cobot. The cobot has a 3 dimensional workspace (x,y,θ). There are two spherical CVT joints. The first joint couples link 1 and link 2, and the other couples link 2 and link 3. There are three main problems associated with this arm-like cobot: gravity acts on the cobot structure, error accumulates via serial transmissions, and like the bicycle it has a fairly pronounced singularity. The first problem could be solved by attaching a counterbalance mechanism to the linkage. Using feedback control can reduce the error accumulations. Eliminating the singularity can be accomplished by adding a CVT that couples link1 and link 3, though this may prove difficult in practice. Another, and probably better solution to this problem is to connect all CVTs to a common joint, as in the 6 dimensional cobot described above, or the power-assist cobots described below.

2.3 Powered Assist Cobots

The cobots presented above are passive cobots and mainly provide virtual guidance. The basic architecture of these cobots (except the 6 dimensional cobot) is illustrated in Fig 2.14. As this figure illustrates, a typical n C-Space cobot would need n-1 CVTs[2]. Each CVTs couple two joints. For instance, the 3 DOF serial cobot, in Fig 2.13, uses 2 spherical CVTs.

[2] Except redundant cobots such as the tricycle cobot.

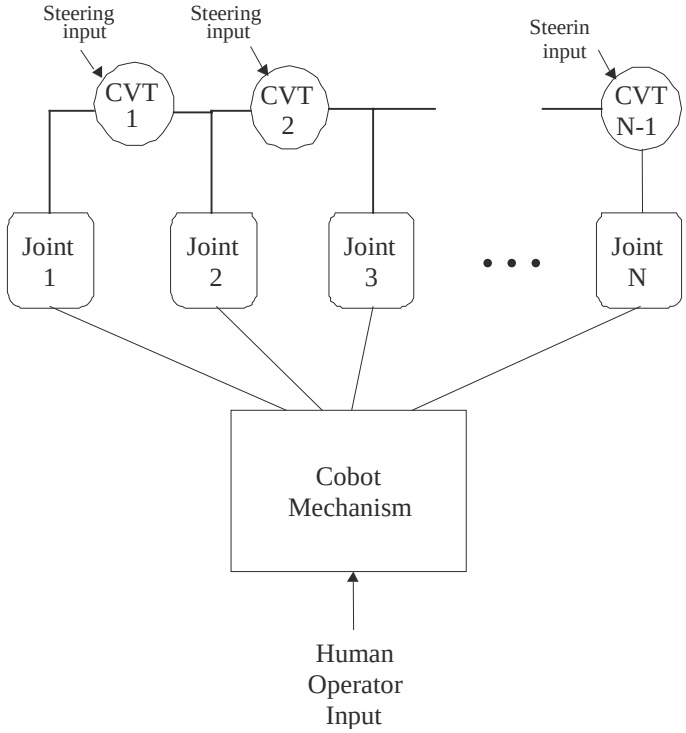


Figure 2.14. A serial chain cobot structure.

For serial chain cobot structures, the cobot endpoint velocity is a function of products of transmission ratios. Thus error accumulates along the mechanism. The other disadvatage is degeneracy. If one of velocity components is equal to zero, the cobot exhibits a singularity. Similar to the bicycle cobot, when two wheel are parallel, it gains additional DOF.

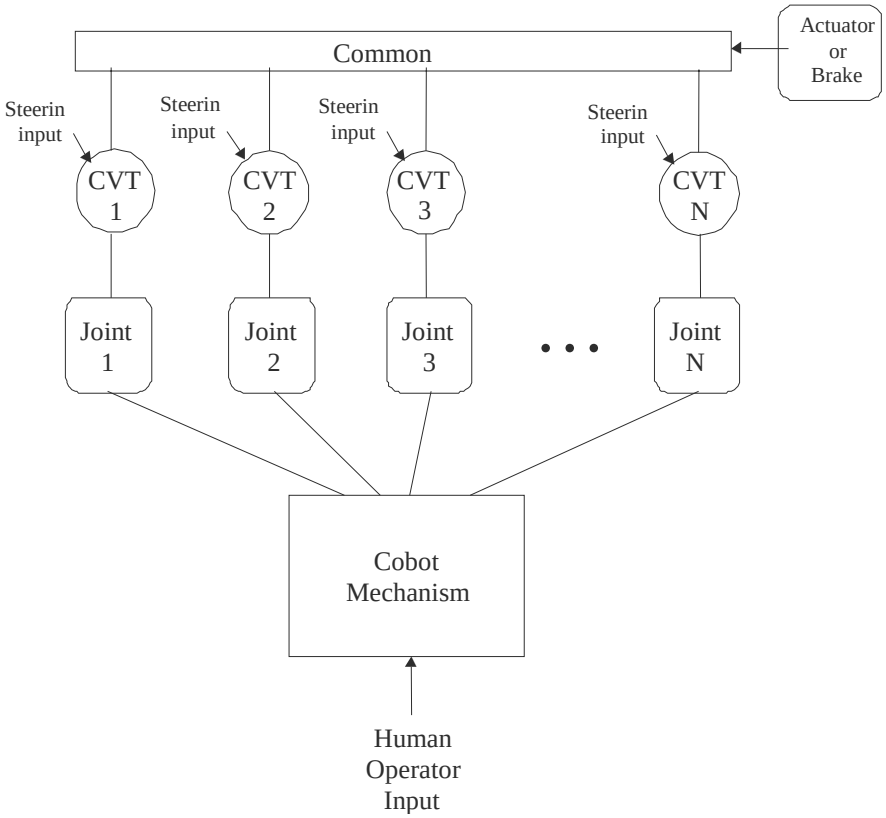


Figure 2.15. Parallel cobot structure : Powered assist cobot

To overcome these problems, let us consider a parallel cobot structure as shown in Fig. 2.15. For this parallel structure, an n C-Space requires n CVTs. One axis of each CVT is connected to a joint of the cobot mechanism. The other axes are connected to a common link. This allows the cobot to relate any two joint velocities with never more than two transmission ratios. A key advantage of this parallel structure is an ability to add power assist. An actuator (or brake) can be used to drive the common link.

An interesting feature of this architecture is that, to add power assist, only one additional actuator is required, regardless of the number of C-Space dimensions. This makes power assist

cobots have great appeal over a regular power assist servo system[3]. Since cobots at all times have one DOF (i.e., one available direction of motion), cobots need only one actuator to assist in that direction.

Figure 2.16 shows a power assist parallelogram arm cobot. The cobot consists of three spherical CVT units, a common link driven by a motor, and a parallelogram mechanism. The prototype is being build at Northwestern University. For detials please refer to [20].

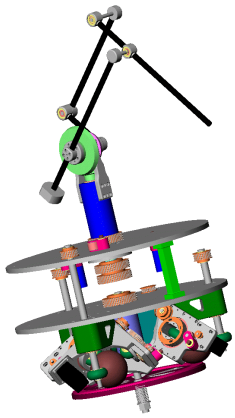


Figure 2.16. A power assist parallelogram cobot (drawing courtesy of Carl Moore).

[3] A regular n DOF power assist system normally requires n power assist motors

CHAPTER 3

UNICYCLE COBOT PROTOTYPE

This chapter presents the design and the construction of the unicycle cobot prototype. The prototype is designed to be simple, portable, expandable, and reliable. The primary purposes of this unicycle cobot prototype are to investigate the kinematics and control of the simplest cobot.

3.1 Unicycle Cobot Prototype

The unicycle cobot is a one-wheel device. It has 1 DOF and a 2 dimensional C-space. By using *virtual caster* control, which will be described later in this Chapter, the unicycle cobot can exhibit to the user two DOF (*x*-*y* in a plane). The unicycle cobot consists of the wheel unit, a T-frame, electronic circuits (signal processors and amplifiers), computer and computer interfaces.

The wheel unit consists of a rubber composite wheel, a wheel digital encoder, a wheel support, a steering shaft, a handle, timing pulleys and timing belts, a motor, a motor encoder, supporting plates, and bearing housings. Refer to Fig 3.1. The wheel rolls on a bearing, which is tightly pressed on a shaft. The shaft is fixed inside the wheel support. The wheel encoder is mounted on one side of the wheel support.

Fixed on the top of the wheel support is a vertical steering shaft. The axis of the steering shaft passes through the wheel-ground contact point. Thus, unlike a caster, there is no offset between the shaft axis and the ground contact. The reason for this design is

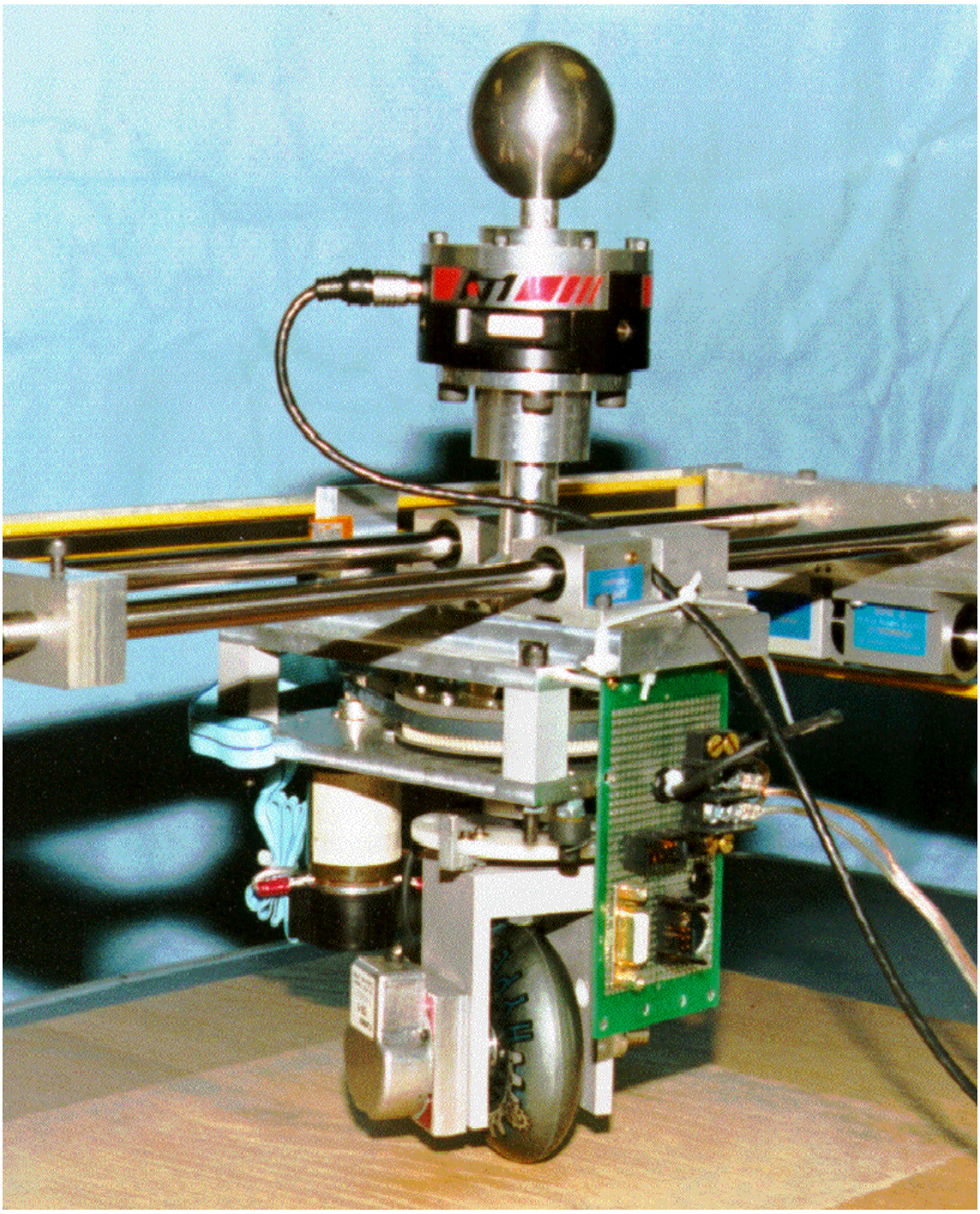


Figure 3.1. The Unicycle Cobot Prototype

that motor torques intended for steering are not transmitted to planar motion. This makes the device energetically passive in the configuration space. A large pulley is mounted on the steering

shaft and coupled to a small pulley on the motor shaft via a timing belt. The transmission ratio of these pulleys is set to maximize the achievable angular acceleration of the steering system (Section 3.2 provides the details). A handle is placed on the top of the chassis and aligned with the shaft axis. A force sensor is mounted to the handle to monitor user intend forces. Two linear bearing housings are also mounted on the top of the chassis.

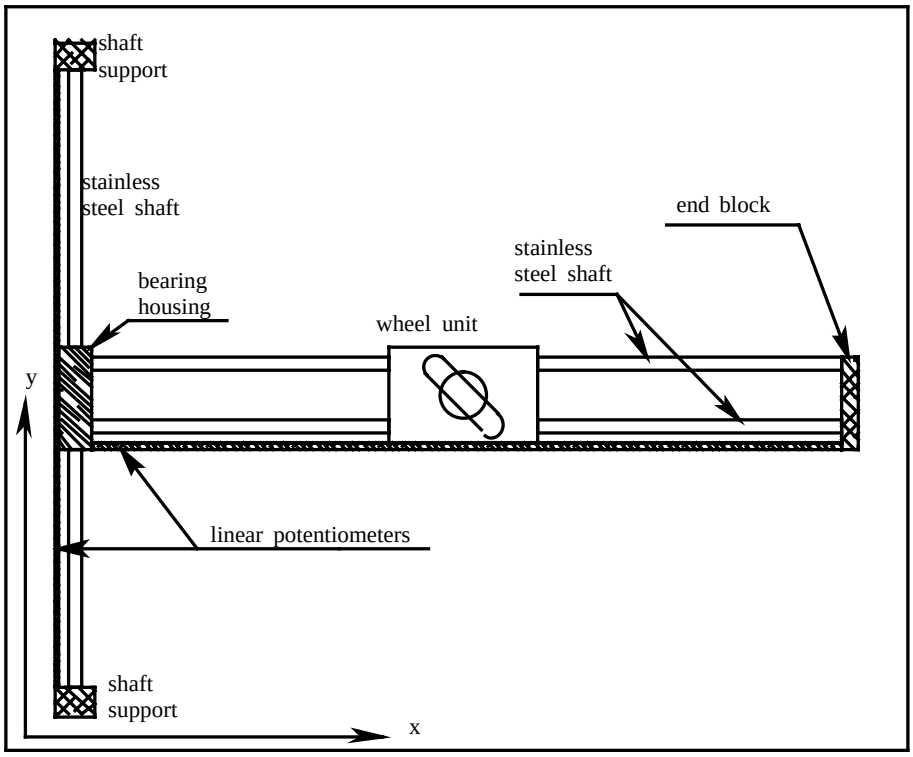


Figure 3.2. Unicycle cobot diagram

The T-frame, shown in Fig 3.2, provides support for the unicycle and allows *x*-*y* motion. In the *y* direction, a single shaft is supported at both ends. The bearing housing is placed on the shaft and can slide between the shaft supports. There are two parallel shafts in the *x* direction. The steering assembly rides on these shafts. The T-frame is instrumented with linear potentiometers, which measure the absolute position of the machine. Using information from the wheel encoders, one can calculate the machine position, however this is not robust to wheel slip.

A signal flow diagram for the unicycle machine is depicted in Fig 3.3. The diagram comprises the wheel unit with T-frame, electronic circuits, and a computer, as well as computer interfaces. Inside the electronic circuit box, there is a motor amplifier, a potentiometer supply, and a signal conditioner. The computer interface consists of an A/D & D/A board and an encoder board. The absolute position of the unicycle is measured by the potentiometers. All analog signals are filtered in the electronics box and then fed to the A/D board. Encoder signals are directly sent to the encoder board. The amplifier receives the motor command from the D/A board and produces the current that drives the motor.

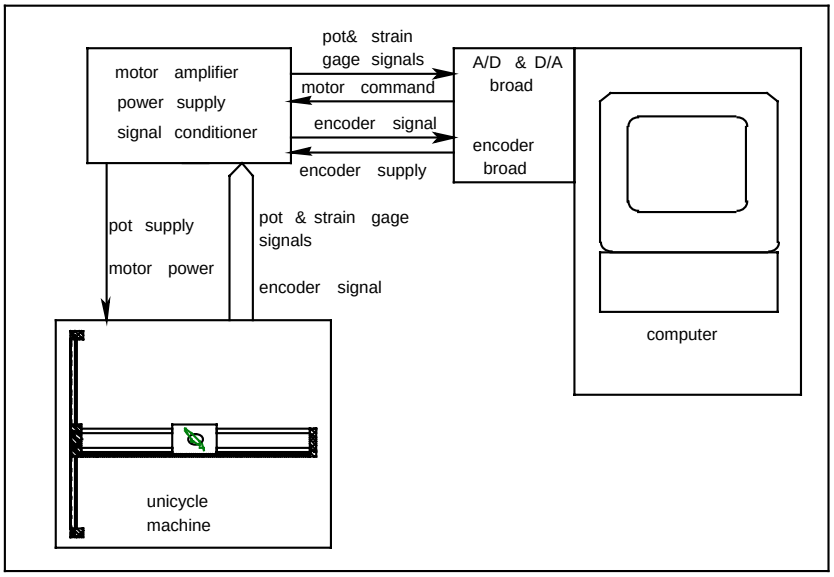


Figure 3.3. Unicycle cobot signal diagram

3.2 Transmission Selection Analysis

The purpose of the analysis is to maximize the angular acceleration capability of the wheel steering axis. This is done to ensure a highly responsive steering system.

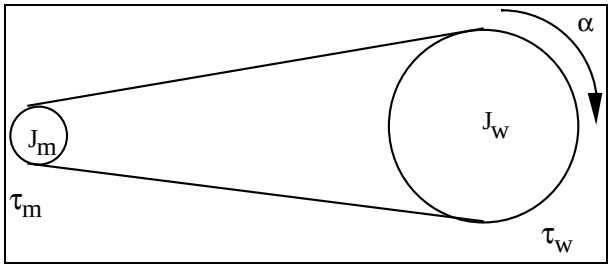


Figure 3.4. Timing belt transmission

Refer to Fig 3.4, the equation of motion can be written as:

$$J_w \ \alpha \ + \ \tau_f \ = \ n[\ \tau_m \ - \ n \alpha \ J_m \ ] \ , \qquad 3.1$$

where J_m is mass moment of inertia of the motor; J_w is mass moment of inertia of the wheel assembly; n is transmission ratio; τ_m = motor torque, τ_f = friction torque at the wheel-floor contact; and α = angular acceleration.

Rearrange Equation 3.1:

$$\alpha (J_w + n^2 \ J_m) \ = \ n \ \tau_m - \tau_f \qquad 3.2$$

α can be written as:

$$\alpha \quad = \quad \frac{n \ \tau_m - \tau_f}{(J_w + n^2 \ J_m)} . \qquad 3.3$$

Maximize α with respect to n.

$$\frac{d\alpha}{dn} \ = \ 0 \ = \ \frac{\tau_m (J_w + n^2 \ J_m) - 2 n \ J_m (n \ \tau_m - \tau_f)}{(J_w + n^2 \ J_m)^2} \qquad 3.4$$

or
$$0 = \tau_m (J_w + n^2 J_m) - 2 n J_m (n \tau_m - \tau_f) \quad 3.5$$

$$0 = -n^2 - 2 n \frac{\tau_f}{\tau_m} + \frac{J_w}{J_m} \quad 3.6$$

The transmission ratio n is:

$$n = \frac{\tau_f}{\tau_m} + \sqrt{(\frac{\tau_f}{\tau_m})^2 + \frac{J_w}{J_m}} \quad 3.7$$

Note that if the friction is small, n can be approximated as:

$$n = \sqrt{\frac{J_w}{J_m}} \quad 3.8$$

3.3 Introduction to Unicycle Cobot Control

In this section, we provide an introduction to unicycle cobot control. The completed analysis of the unicycle cobot control will be provided in Chapter 4.

Recall 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. ***Virtual wall 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 that 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. 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.

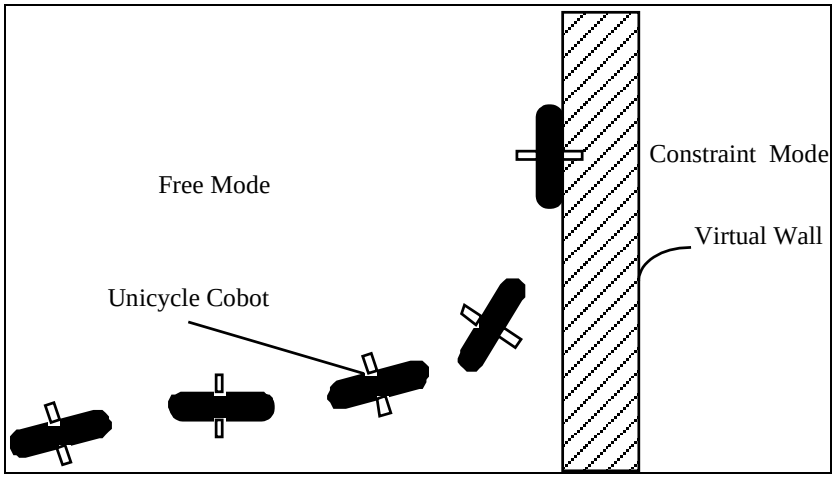


Figure 3.5 Unicycle Cobot exhibits free mode and virtual wall mode

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 DOF. 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.

The completed details of unicycle cobot control including experiment results are presented in Chapter 4.

CHAPTER 4

UNICYCLE COBOT CONTROL

This chapter presents unicycle cobot control. Section 4.1 describes the virtual caster control. The purpose of virtual caster to eliminate the wheel so that the operator feels there is no wheel at all. Section 4.2 presents path-tracking analysis. Since there are slip on the wheel, feedback control is used in addition to feed forward control. The last Section presents implementation of unicycle cobot controls.

4.1 Virtual Caster Control

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, $F_{\parallel}$, produce accelerations of $a_{\parallel} = F_{\parallel}/M$, 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 that would result in particle-like behavior:

$$\omega = \frac{F_{\perp}}{uM} \qquad 4.1$$

Equation 4.1 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. Equation 4.1 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.

4.2 Path Tracking Analysis

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. 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.

4.2.1 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. 4.1. Moreover, the speed, u, of the unicycle along the path, and the curvature, κ, of the path would together determine the necessary steering velocity:

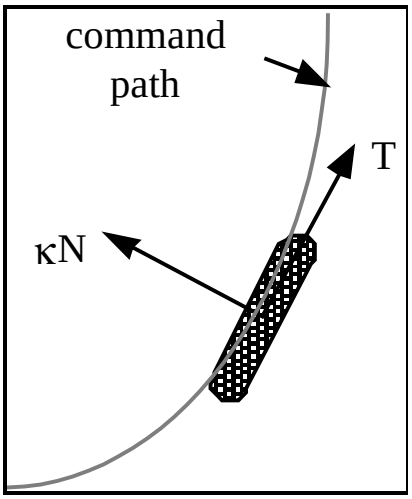


Figure 4.1. Top view of a unicycle cobot tracking a path

$$\omega = \kappa u \qquad 4.2$$

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.

4.2.1 Feedback Control

Fig. 4.2 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 (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 Equation 4.2 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.

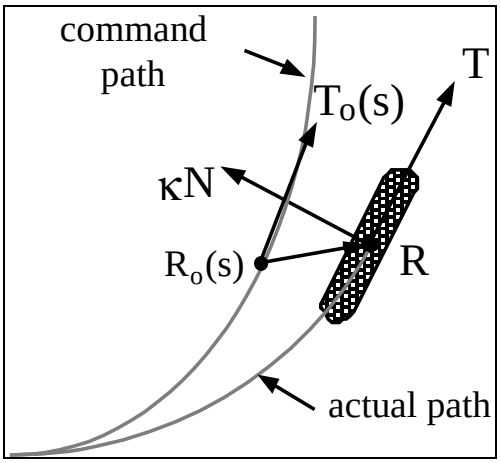


Figure 4.2. Imperfect path tracking

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. 4.2) that the path length error (s'

- s) is simply the length of (R - $R_o(s)$) projected onto T_o. Thus, we define:

$$s' = s + (R - R_o(s)) \cdot T_o(s) \qquad 4.3$$

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.

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 unicycle cobot, 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:

$$\omega = u\left[\kappa_o(s') - N_o(s') \bullet \left(\frac{G_1}{L^2}\Delta R + \frac{G_2}{L}\Delta T\right)\right] \qquad 4.4$$

$\kappa_o(s')$ is the feedforward term from Equation 4.2. 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 [7,22]. For further details of kinematics and stability associated with gain selections, follows work done by Colgate, Wannasuphoprasit, and Peshkin [7].

4.2 Unicycle Cobot Implementation

4.3.1 Virtual caster

The unicycle cobot was used to collect the data presented in this chapter. The virtual caster controller follows the form of Equation 4.1, 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 Equation 4.1 must also be modified to prevent overflow, excessively large control signals, and instability. The caster controller has therefore taken the form:

$$\Gamma = \frac{K_1 F_\perp}{\text{sign}(u)\,\max(|u|, \varepsilon\,\text{sign}(u))} - K_2\omega \qquad 4.5$$

Γ is the motor torque, K_1 is an adjustable gain which replaces $1/M$ in Equation 4.1, 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.

4.3.2 Virtual Wall

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.

Figure 4.3 and 4.4 shows recorded data in both caster and constraint modes for the unicycle cobot. 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. The results was very pleasing. The virtual caster is fairly responsive. The virtual walls are hard smooth and very convincing.

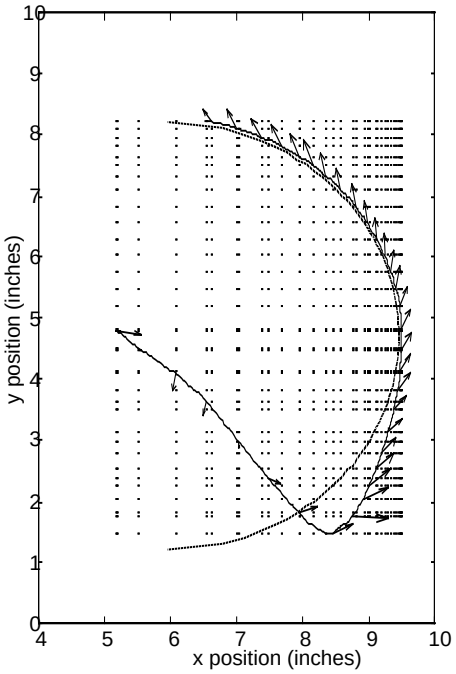


Figure 4.3. Circular wall. The cobot exhibit virtual caster mode inside the circle. When the collision occurs the controller switch to the path tracking mode.

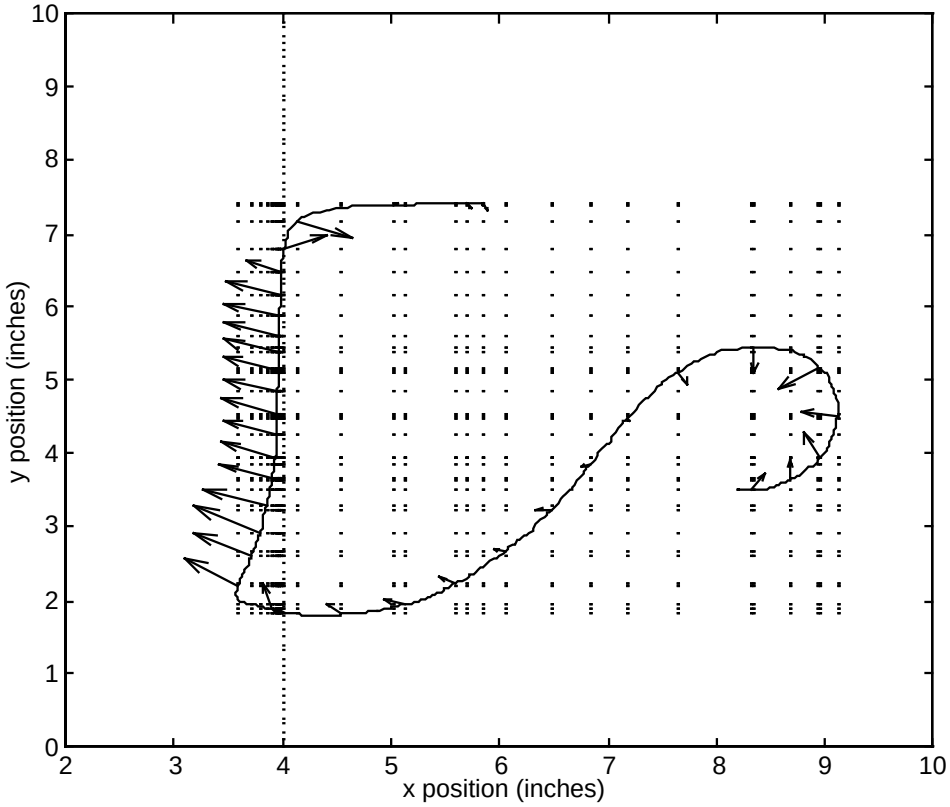


Figure 4.4. Trajectory and applied forces for a unicycle cobot. A virtual wall is located at x = 4.

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. 4.5 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.

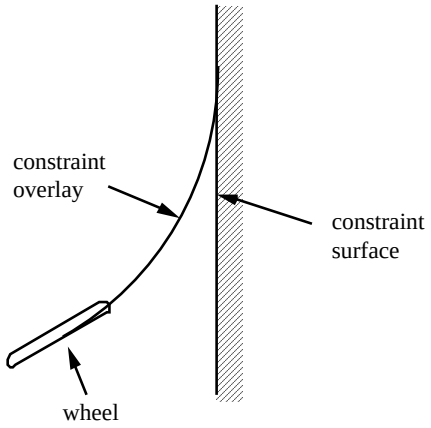


Figure 4.5. Overlay constraint

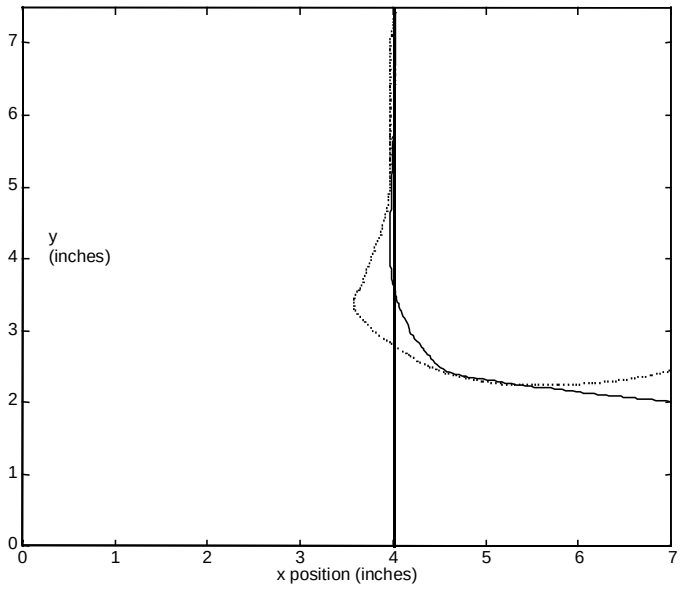


Figure 4.6. Unicycle cobot trajectories with and without constraint overlay

In Fig. 4.6, 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.

CHAPTER 5

SCOOTER: A TRICYCLE COBOT

This chapter describes the design and construction of the cobot “Scooter”. Scooter is a benchmark for testing higher DOF cobot control. Not only intended to be a laboratory research device, Scooter can be used for light industrial applications such as assisting a human in an assembly task. A major design objective for Scooter was to permit unlimited steering axis rotation (no joint limits). Among other consequences, this led to a novel velocity sensor design, *the glide wheel*. Glide wheels and the problem of velocity estimation are discussed in detail in Section 5.2.

It has been stated in this thesis that all cobots have one DOF; i.e., one instantaneously available direction of motion. For Scooter, however, if all three wheel axis do not intersect at a common point, there is no available direction of motion – the device is stuck. Section 5.3 presents a COR (center of rotation) agreement control algorithm. The COR agreement controller minimizes the error that results from imperfect intersection of all three wheel axes. This insures that the cobot always has one DOF.

5.1 Design and Construction of Scooter

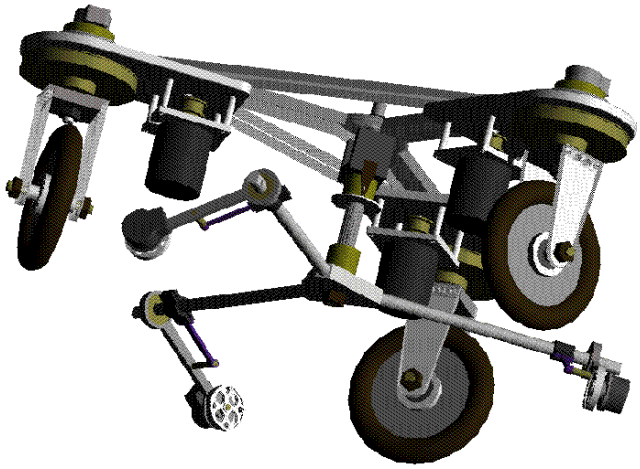


Figure 5.1: 3D CAD drawing of Scooter

Scooter (Fig 5.1) is a planar cobot. It has three dimensional work space (x,y,θ). The purpose of building Scooter was to understand higher work space cobots and explore cobot control in higher DOF. The scooter is designed not only to be a laboratory research device but also a pilot test for light-medium applications. Based on the experience of building, controlling and using the unicycle cobot, the main design objectives were as follows:

- responsive steering: The effectiveness of virtual caster control, especially at low speed, and the ability of cobot to change direction upon striking a virtual wall, are both limited by the responsiveness of the steering controller.

- continuous steering rotation: it is extremely important that there are no limits to the steering angle – running into a limit produces a behavior which is disconcerting and non-intuitive.

- high resolution velocity sensing: virtual caster control algorithms are quite sensitive to wheel speed, especially at low velocity; thus, high resolution sensing is important.

- handle force sensing: two potentially useful locations for force sensing are at handle and on the wheel assemblies. The former is preferred because it gives a better indication of user "intent".

The design of Scooter has been greatly influenced by these four points. For instance, in keeping with points 1, 2 and 4, the wheel subassemblies are not instrumented except for an optical encoder that detects steering shaft angle. No instrumentation at all is placed on the steered portion of the assembly. Because of this, the steered inertia is low, permitting rapid response. Also, no signals or power need be supplied to the wheel assemblies, allowing unlimited rotation to be easily achieved (i.e., no slip rings, telemetry, or other method is needed to transmit signals between bodies in relative rotation).

In addition to the above specifications, Scooter is designed to be simple, modular, and rugged. The design also permits relatively straightfoward modification, such that it can be used for several applications with payload up to 250 lbs.

5.1.1 Wheel unit subassemblies

The wheel unit subassemblies are basically unicycle cobots without force sensors or handles, as illustrated in Figure 5.2. Each wheel unit consists of a supporting plate, a bearing housing, a motor and a motor support, timing pulleys, a timing belt, an optical encoder, a steering shaft, steering shaft bearings, a wheel, a wheel support, and wheel bearings. The major difference between Scooter's wheel units and those of the unicycle is relocation of the velocity

sensor. The glide wheel velocity sensors and the matter of velocity estimation will be explained in Section 5.2.

As is evident in Figure 5.2, the wheel unit design is very straightforward. A small brushless dc motor drives the steering shaft via a timing belt transmission that provides a ratio of approximately 8:1. A high resolution optical encoder measures the steering angle. As previously mentioned, there are no limits to the steering angle.

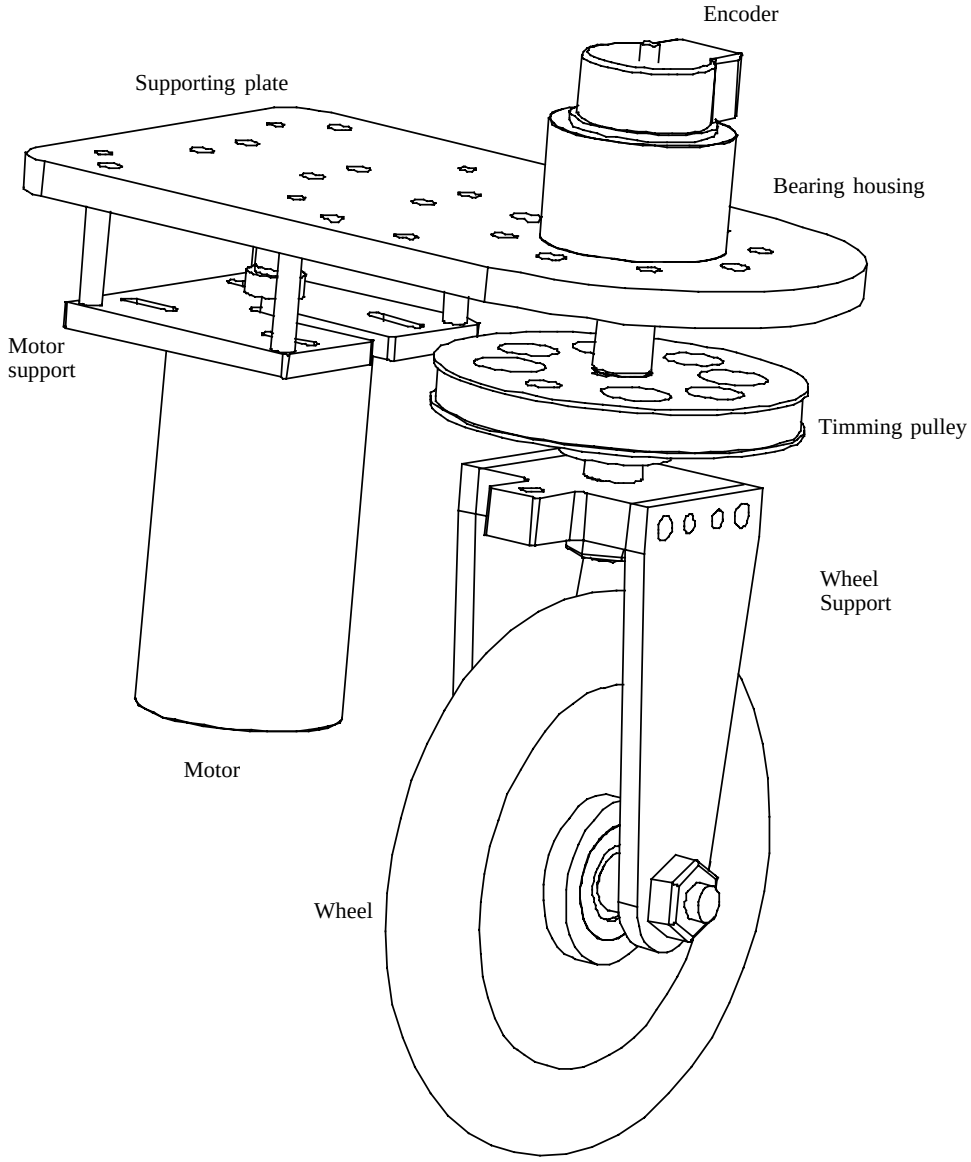


Figure 5.2. Wheel unit assembly

5.1.2 Platform

The platform is the main structural component of Scooter. It connects all three wheel units together, and also supports the velocity sensor assemblies. The platform is made from a 5/8 inch thick aluminum plate in the shape of an equilateral triangle with a circular ring attached to each corner. There are several cutaways to reduce weight.

5.1.3 Glide Wheel Sensors

The decision to remove all instrumentation from the wheel assemblies creates one significant difficulty. It is no longer possible, as a consequence of this decision, to measure velocity by placing sensors directly on the wheels. It is instead necessary to infer wheel velocities (and the velocity of the cobot in general) by employing a separate sensor system.

Here, there are many options, including systems based on optical flow, gyros, integrated accelerometer signals, differentiated displacement signals (displacement measured, for instance, by a mouse), and so on. We have chosen an approach of the latter type, but based on a novel component called the "glide wheel" rather than a computer mouse.

A glide wheel (Fig 5.4) is simply a wheel which, like the three steering wheels of Scooter, runs on the planar substrate. Unlike the steering wheels, however, glide wheels are not intended to provide constraint. In fact, they are designed to slip easily. Because they are designed to slip, glide wheels can be fixed in the tricycle's body frame without impeding its motion. An encoder monitors rotation of the glide wheel.

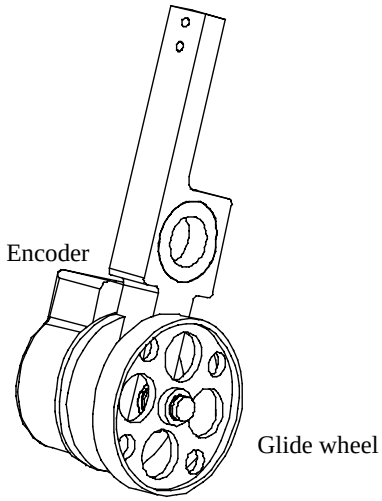


Figure 5.4. Glide wheel sensor

A glide wheel responds to a component of the translational velocity of its center: the component, which is parallel to the wheel's rolling direction. Thus, if the point on the frame at which the glide wheel is mounted happens to be moving nearly perpendicular to the rolling direction, the glide wheel will primarily be skidding, and will rotate slowly. This behavior is somewhat more subtle than it might at first appear, and becomes ideal only if there is no resistance to turning (bearing or encoder resistance, for instance) When the skidding velocity is large compared to the velocity in the rolling direction, even a very small bearing resistance will be sufficient to prevent rotation of the wheel, and produce an (incorrect) reading of zero.

Scooter uses three glide wheel sensors to estimate the velocities. As mentioned before, at some configurations the glide wheel gives incorrect reading of the velocity. However, the estimator described in the next section makes use of geometric insights to obtain a sufficiently accurate measure.

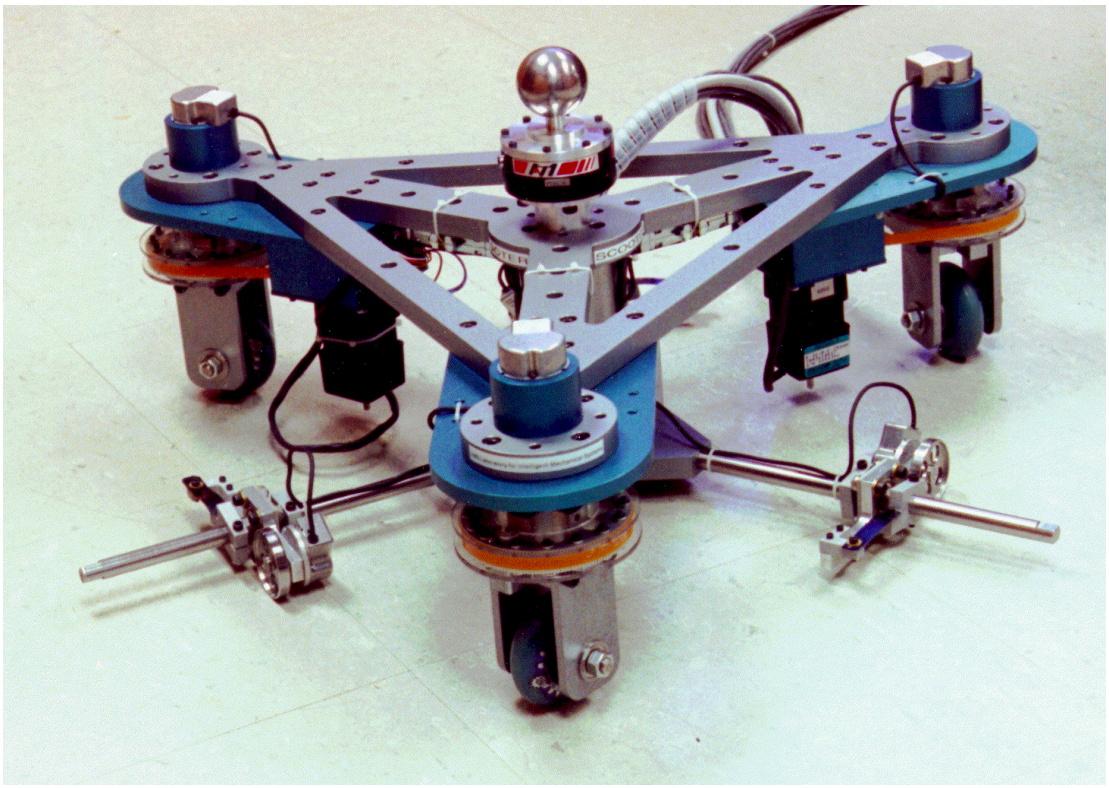


Figure 5.5. Picture of Scooter

5.2 Estimating Velocity

Three parameters are needed to specify the velocity of a planar rigid body. Common choices include the *x* and *y* velocities at a selected point on the body along with the angular velocity, and the *x* and *y* coordinates of the COR along with the angular velocity. For present

purposes, we will assume that the objective is to obtain the former parameters for a point at the center of the tricycle cobot.

The most straightforward approach to this is to use the three glide wheel signals as illustrated in Fig. 5.6. If we assume that the velocity and angular velocity vectors at the center of the cobot are $\boldsymbol{v} = (v_x, v_y, 0)$ and $\underline{\omega} = (0, 0, \omega)$ respectively, then the expected glide wheel measurements are:

$$u_i = (\boldsymbol{v} + \omega\hat{\mathbf{z}} \times \mathbf{r}_i) \cdot \hat{\mathbf{t}}_i \qquad i = 1,2,3 \qquad 5.1$$

These equations are linear in v_x, v_y and ω, and are easily solved to yield cobot velocity. Once the cobot velocity is known, it is straightforward to solve for the steering wheel velocities.

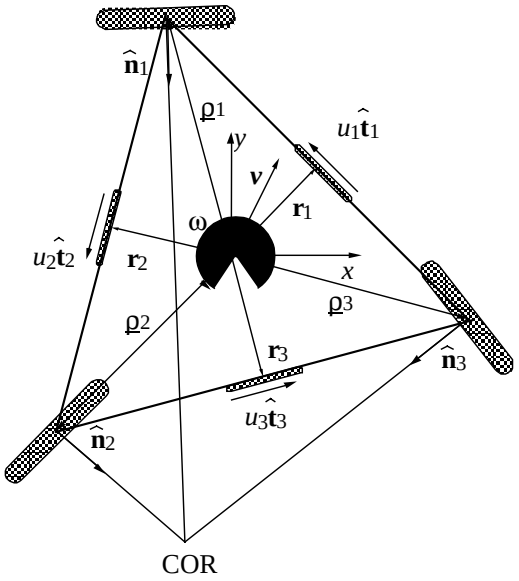


Figure 5.6. Scooter geometry.

$\boldsymbol{v}$ and ω represent the translational and angular velocities of the cobot center; $\mathbf{r}_i$ is a vector from the cobot center to the i^{th} glide wheel (all three vectors have length r); $\hat{\mathbf{t}}_i$ is a unit vector in the rolling direction of the i^{th} sensor wheel; and u_i is the speed measure obtained from the i^{th} glide wheel. ; $\underline{\rho}_i$ is a vector from the cobot center to the i^{th} steering wheel (all three vectors have length ρ); $\hat{\mathbf{n}}_i$ is a unit normal vector for the i^{th} steering wheel which points toward the COR.

There remains the question of whether this is a good approach to velocity estimation. It can be shown that Equation 5.1 are always well-conditioned, therefore, there is no reason to expect otherwise if the glide wheel readings are all accurate. As was discussed above, however, we can expect a glide wheel reading to degrade as the wheel's velocity becomes approximately normal to its rolling direction. Referring to Fig. 5.6, we see that this happens when the COR is on or near the line (shown as a dashed line in the figure) that passes through the wheel and is parallel to the wheel's rolling direction. Due to the arrangement of the glide wheels, however, the COR can lie near this line for at most two of the wheels. Thus, when one, or at most two of the glide wheels are principally in sliding, the remaining wheel or wheels should be principally in rolling, *and* should have a much higher rolling velocity. Thus, the reading from this wheel or wheels should be the primary contributor to the cobot velocity estimation.

In general, glide wheel sensors perform relatively well at medium and high speed. However error increases as speed decreases. There is, however, another approach to velocity estimation. Because the steering angles are known accurately (high resolution optical encoders are used to measure steering angle, as discussed below), it is possible to estimate the location of the COR. Once this is known, only a single variable, the angular velocity about the COR, needs to be measured. This can be obtained from the fastest-moving glide wheel, ensuring a low-noise

measurement. A possible problem with this method, however, is that the COR can be expected to pass through infinity on a regular basis, and even linger there during straight-line movements. This can be solved by estimate the infinity with a reasonable large number.

The approach is to calculate redundant velocities and weight them by a weight function, which is proportional to the glide wheel velocity. For example, the first set of cobot velocities $V_1 = (v_{x1}, v_{y1}, \omega_1)$ can be estimated by using three know variables:

$$V_1 = f(\alpha_3, \alpha_1, v_{g1}), \qquad 5.2$$

where α_1, α_3 are steering angles of wheel unit 1 and 3 respecitvely, and v_{g1} is the glide wheel velocity unit 1. Thus one can write V_2 and V_3 as:

$$V_2 = f(\alpha_1, \alpha_2, v_{g2}), \qquad 5.3$$

$$V_3 = f(\alpha_2, \alpha_3, v_{g3}) \qquad 5.4$$

Using a weight function, the cobot velocity can be written as:

$$V = \frac{v_{g1}V_1 + v_{g2}V_2 + v_{g3}V_3}{v_{g1} + v_{g2} + v_{g3}} \qquad 5.5$$

5.3 COR Agreement Control

The proper operation of Scooter relies heavily on the assumption that the three steering wheels will always agree upon a COR (unless they are intentionally misaligned for the purposes of braking).

There exist a variety of functions of the three steering angles which are zero for only those configurations which do agree upon a COR. The form of such a function is:

$$f(\theta_1,\theta_2,\theta_3) = 0 \qquad 5.6$$

Selecting an appropriate function is not a trivial task, unfortunately. Many seemingly obvious choices exhibit singularities; i.e., COR locations at which the function is either insensitive to COR disagreement, or extremely sensitive. One function which we have found to work well is the perimeter of the triangle formed by the intersection of the three wheel normals, appropriately scaled. The perimeter of this triangle is, of course, zero when the three normals all pass through the COR. The perimeter of the triangle grows smoothly with any sort of disagreement. Scaling is necessary because even minor disagreements can cause large changes in perimeter when the COR is very far from the cobot center.

It is useful to think of f as defining a two-dimensional surface in the three-dimensional $(\theta_1,\theta_2,\theta_3)$ space. COR agreement corresponds to a point on this surface, and valid COR slews are trajectories on this surface. When the configuration does not lie on the surface (i.e. f is not equal to 0), the COR agreement controller should drive it back, *preferably without interfering with movements along the surface*. In other words, corrective movements should be normal to the surface. The normal is given by the gradient of f:

$$\nabla f = \left(\frac{\partial f}{\partial q_1}, \frac{\partial f}{\partial q_2}, \frac{\partial f}{\partial q_3}\right) \qquad 5.7$$

A velocity servo acts on each of the steering axes, therefore, it is necessary to convert the information in f and ∇f into angular velocity commands for the three axes. One reasonable approach is:

$$\underline{\omega}_{COR} = -\frac{1}{\tau}\frac{f}{|\nabla f|^2}\nabla f \qquad 5.8$$

Here, $\underline{\omega}_{COR}$ is a vector of angular velocity commands for the three wheels, which is seen to be proportional to f and in the direction of the surface normal. τ is a control gain with the units of time. The expression is normalized by $|\nabla f|^2$ so that the time constant for reaching the surface (assuming ideal steering velocity servos) is τ. To minimize interference with higher levels of control, τ is made as fast as possible.

CHAPTER 6

COBOT CONTROL

This chapter focuses on cobot control in higher DOF. To gain a better understanding of cobot control, we developed and implemented control algorithms on Scooter [31]. Although Scooter has only a 3 dimensional C-Space, the cobot control theory outlined here was developed for C-Space of an arbitrary dimension.

6.1 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,

$$R = [x, y, l\theta]^T, \qquad 6.1$$

where R is a configuration vector in C-Space.

The path followed by wheel i can easily be derived from 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:

$$r_i = \begin{bmatrix} xi \\ yi \end{bmatrix} = \begin{bmatrix} Lix(R) \\ Liy(R) \end{bmatrix} = L_i(R), \quad i = 1,2,3 \qquad 6.2$$

Equation 6.2 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.

6.2 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):

$$T = \frac{dR}{ds}; \quad t_i = \frac{dr_i}{ds_i}, \quad i = 1,2\ 3 \qquad 6.3$$

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 Equation 6.2 with respect to s_i:

$$t_i = \frac{dr_i}{ds_i} = \frac{\partial Li}{\partial R}\frac{dR}{ds}\frac{ds}{ds_i} \qquad 6.4$$

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, *T*, 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:

$$t_i = J_i T \frac{ds}{ds_i} = \frac{JiT}{|JiT|} \qquad 6.5$$

6.3 Curvature Transformation

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

$$\kappa N = \frac{dT}{ds}; \quad \kappa_i n_i = \frac{dt_i}{ds_i}, \quad i = 1,2,3 \qquad 6.6$$

An expression for $\kappa_i n_i$ can be obtained by differentiating Equation 6.5 with respect to s_i.

$$\kappa_i n_i = \frac{d^2 r_i}{ds_i^2} = \frac{\partial}{\partial s_i}\left[J_i T \frac{ds}{ds_i} \right] \qquad 6.7$$

$$\kappa_i n_i = \frac{\partial J_i}{\partial s_i} T \frac{ds}{ds_i} + J_i \frac{dT}{ds_i}\frac{ds}{ds_i} + J_i T \frac{d^2 s}{ds_i^2} \qquad 6.8$$

$$\kappa_i n_i = \frac{\partial J_i}{\partial s_i} T \frac{ds}{ds_i} + \kappa J_i N \frac{ds}{ds_i}\frac{ds}{ds_i} + J_i T \frac{d^2 s}{ds_i^2} \qquad 6.9$$

Let us now consider some of the constituent terms of Eq. 6.9:

$$\frac{ds}{ds_i} = \frac{1}{|\, JiT \,|} = \frac{1}{\left(T^T J^T J\, T\right)^{1/2}} \tag{6.10}$$

$$\frac{\partial J_i}{\partial s_i} = \left[\frac{T^T H_{ix}}{T^T H_{iy}}\right]\frac{ds}{ds_i} = \left[\frac{T^T H_{ix}}{T^T H_{iy}}\right]\frac{1}{|\, JiT \,|}\,, \tag{6.11}$$

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

$$H_{ix} = \frac{\partial^2 L_{ix}}{\partial R^2}\,;\; H_{iy} = \frac{\partial^2 L_{iy}}{\partial R^2}. \tag{6.12}$$

$$\frac{d^2 s}{ds_i^{\,2}} = \frac{-0.5}{\left(T^T J^T J\, T\right)^{3/2}}\left[\begin{array}{l} \kappa N^T J^T J\, T\dfrac{ds}{ds_i} + T^T\left[\dfrac{(T^T H_{ix})^T}{(T^T H_{iy})^T}\right] JT\dfrac{ds}{ds_i} + T^T J^T\left[\dfrac{T^T H_{ix}}{T^T H_{iy}}\right] T\dfrac{ds}{ds_i} \\ \quad ... + T^T J^T J\,\kappa N\dfrac{ds}{ds_i} \end{array}\right] \tag{6.13}$$

$$\frac{d^2 s}{ds_i^{\,2}} = \frac{-1}{|\, JiT \,|^4}\left[\; T^T J^T\left[\frac{T^T H_{ix}}{T^T H_{iy}}\right] T + \kappa T^T J^T J\, N\right] \tag{6.14}$$

Substituting Equations 6.10, 6.11, and 6.14 into 6.9:

$$\begin{aligned} \kappa_i n_i = {} & \left[\frac{T^T H_{ix}}{T^T H_{iy}}\right] T\frac{1}{|\, JiT \,|^2} + \kappa J_i N\frac{1}{|\, JiT \,|^2} - J_i T T^T J_i^{\,T}\left[\frac{T^T H_{ix}}{T^T H_{iy}}\right] T\frac{1}{|\, JiT \,|^4} \\ & - \kappa J_i T T^T J_i^{\,T} J_i\, N\frac{1}{|\, JiT \,|^4} \end{aligned} \tag{6.15}$$

Note that

$$\frac{J_i T T^T J_i^T}{|J_i T|^2} = \frac{J_i T T^T J_i^T}{T^T J_i^T J_i T} = t_i t_i^T \qquad 6.16$$

Thus,

$$\kappa_i n_i = \frac{1}{|JiT|^2}\left(I - t_i t_i^T\right)\left[\frac{T^T H_{ix}}{T^T H_{iy}}\right]T + \frac{1}{|JiT|^2}\left(I - t_i t_i^T\right)J_i \kappa N \qquad 6.17$$

Rewriting Equation 6.17:

$$\kappa_i n_i = \frac{\left(I - t_i t_i^T\right)}{|JiT|^2}\left[\left[\frac{T^T H_{ix}}{T^T H_{iy}}\right]T + J_i \kappa N\right] \qquad 6.18$$

H 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.

6.4 Virtual Caster Control

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

$$A = [F_x/m, F_y/m, \tau d/I]^T, \qquad 6.19$$

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 Equation 6.18 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:

$$\frac{dR}{dt} = Tu \qquad 6.20$$

$$\frac{d^2R}{dt^2} = A = \kappa N u^2 + T\dot{u} \qquad 6.21$$

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

$$\dot{u} = T^T A \qquad 6.22$$

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

$$\kappa N = \frac{1}{u^2} [I - TT^T] A \qquad 6.23$$

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. Equation 6.19 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:

$$t_i u_i = J_i T u \quad i = 1,2,3 \qquad 6.24$$

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:

$$\omega_i = \left[J_i Tu \times \frac{\left(I - t_i t_i^T\right)}{|\, JiT \,|^2} \left[\left[\frac{T^T H_{ix}}{T^T H_{iy}} \right] T + J_i \frac{1}{u^2} \left[I - TT^T \right] A \right] \right] - \dot{\theta} \qquad \text{i = 1,2,3} \qquad 6.25$$

6.5 Path Tracking and Virtual Wall Control

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.

6.5.1 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 Equation 6.18 and 6.24, and following the procedure described previously in Chapter 4.

6.5.2 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 Equation 4.19, 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:

$$\delta(\kappa N) = -\frac{G_1}{L^2}\Delta R - \frac{G_2}{L}\Delta T \qquad 6.26$$

It is necessary only to introduce the curvature correction (Equation 6.26) into the transformation relation (Equation 6.18), and follow the procedure developed for caster control. The resulting control law is:

$$\omega_i = \left[J_i Tu \times \frac{\left(I - t_i t_i^T\right)}{|JiT|^2} \left[\left[\frac{T^T H_{ix}}{T^T H_{iy}} \right] T + J_i \left(\kappa_o(s') N_o(s') + \delta(\kappa N) \right) \right] \right] - \dot{\theta}$$

$$i = 1,2,3 \qquad 6.27$$

6.6 Experiments with Scooter

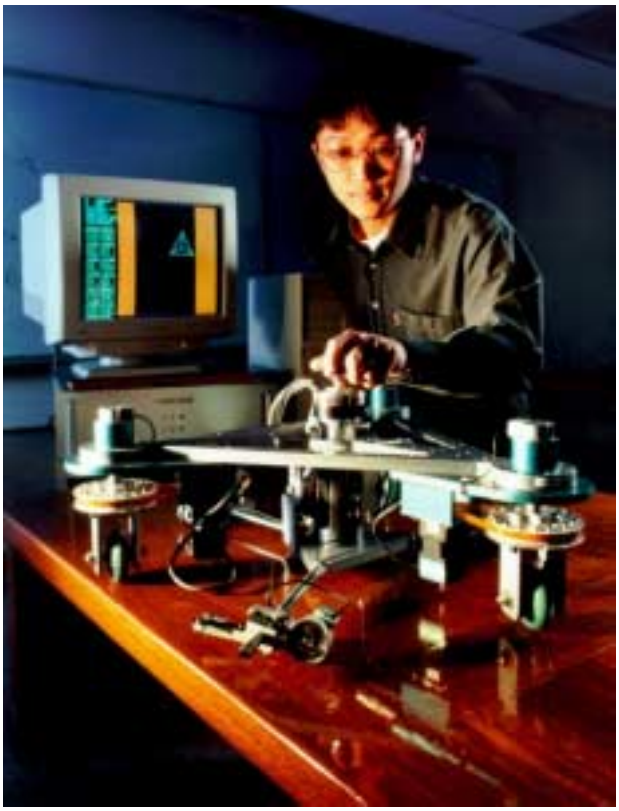


Figure 5.2 Scooter in virtual wall mode. The virtual walls are parallel to the table edges and also displayed on the computer monitor

6.6.1 Virtual caster

Virtual caster is an essential mode of Scooter operation. Figure 6.2 displays an experiment result of Scooter 3D path in virtual caster mode. The path is very smooth. Scooter found to be very easy to maneuver. Only small applied force requires to change its directions. Figures 6.3-6.5 are projections of this path into x-y, x-θ, and y-θ planes respectively. For each sample points, the applied force vectors are overlaid on the projected paths. The force magnitudes are scaled so that 1 graph unit equals to 1 lb of force (or 1-in-lb of torque).

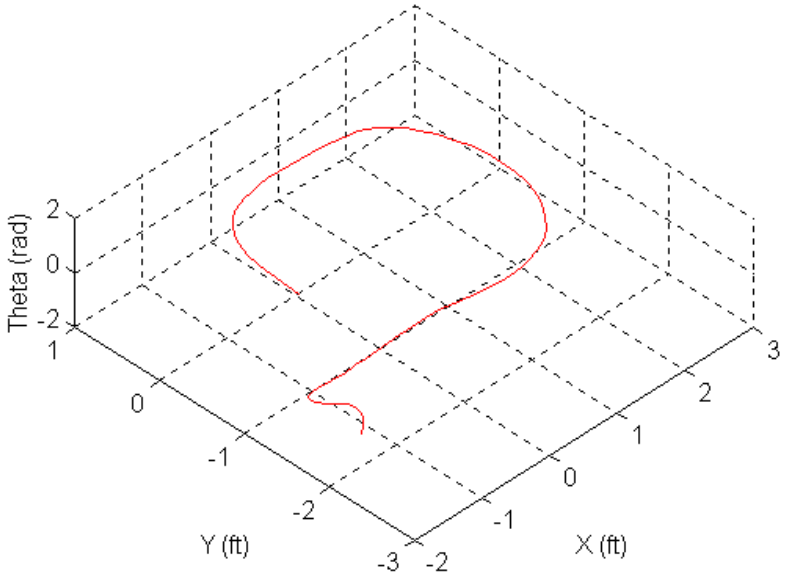


Fig 6.2. Scooter's path in virtual caster mode in *x-y-θ* coordinate.

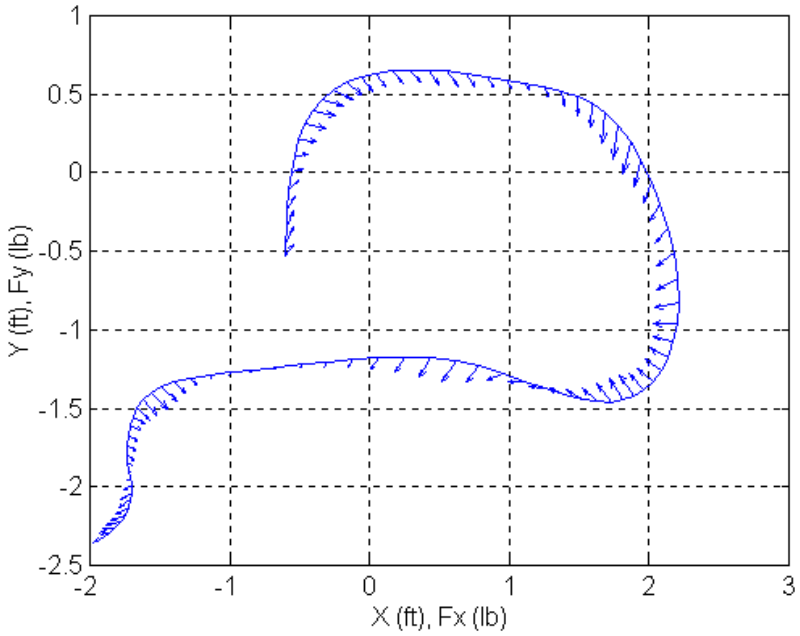


Figure 6.3 *x-y* projection of Scooter path in Figure 6.2 overlay with operator's force vectors.

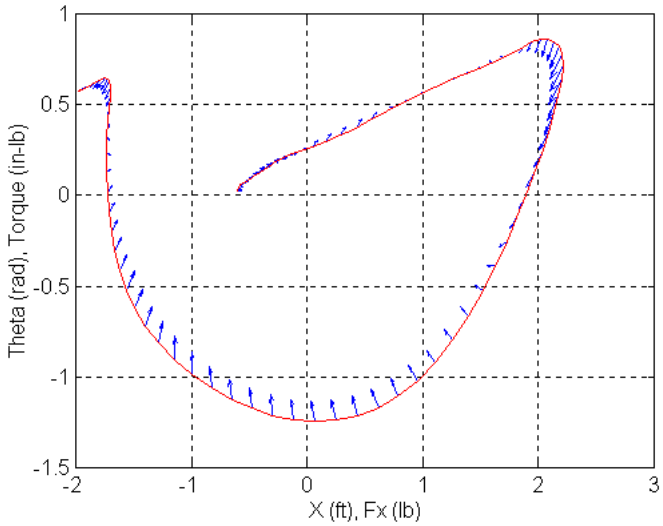


Figure 6.4 *x-θ* projection of Scooter path in Figure 6.2 overlay with operator's force-torque vectors.

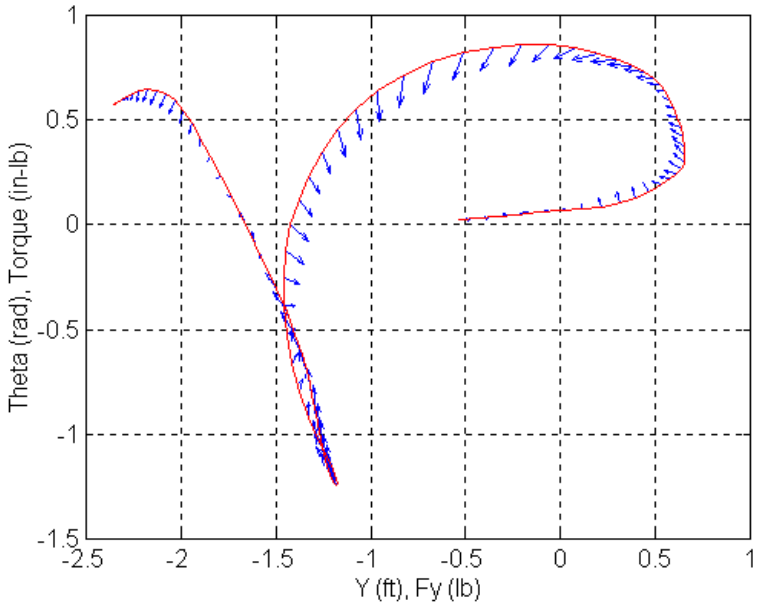


Figure 6.5 *y-θ* projection of Scooter path in Figure 6.2 overlay with operator's force-torque vectors**.**

6.6.2 Path Tracking

Figure 6.6 displays the location of Scooter as it tracks a helical path in C-Space. The solid line is the commanded path, and the dotted line the actual path of Scooter. Scooter tracks the path extremely well (the dot and the solid line are very close together).

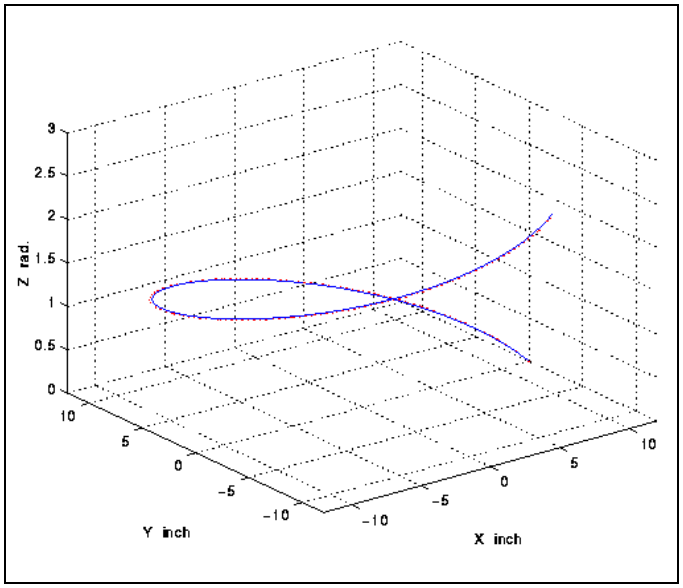


Figure 6.6. The solid line is a commanded helix path. The dot line demonstrates the actual scooter's tracking path.

Scooter also performs well in tracking arbitrary paths. For instance, Fig 6.7 shows Scooter tracking an S-shaped path with a fix orientation. Figure 6.8 displays Scooter tracking a similar S-shaped path, but one in which orientation varies as a function of path length.

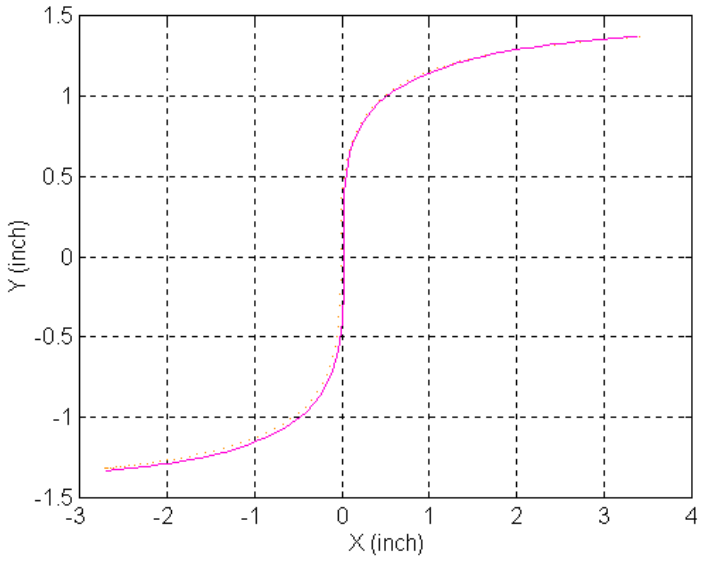


Figure 6.7. Scooter tracks S path with a fix orientation

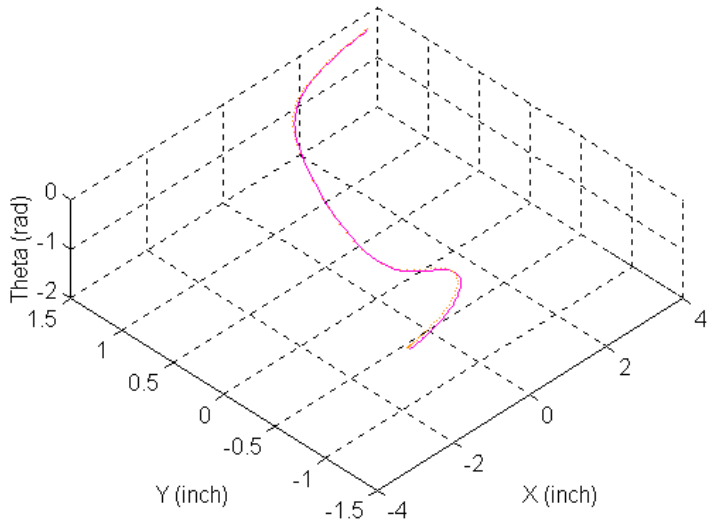


Figure 6.8. S path similar to Figure 6.7 but orientation varies as a function of path length.

6.6.3 Virtual Walls

Implementing virtual wall basically involves a software switch between the virtual caster and path tracking controllers. If Scooter's C-space position is outside the virtual wall, virtual caster control is implemented. When Scooter's position is inside the wall, the path tracking control is implemented. If the dot product of the force vector and the wall's outward-pointing normal vector becomes greater than zero, however, the control switches back to virtual caster control.

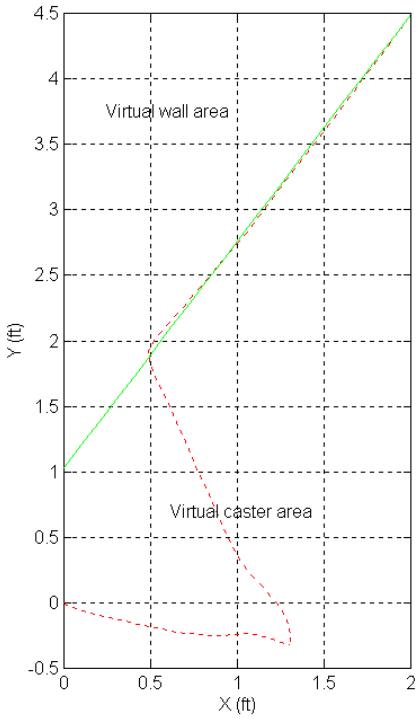


Fig 6.9. Scooter strikes a 60 degree straight wall

Figure 6.9 displays Scooter striking a 60 degree virtual wall in x-y plane. The dash line is the actual Scooter path.

Scooter, in addition, can enforce a 3-D wall. For instance, in Fig 6.10 Scooter implemented a 60 degree wall with an orientation (θ) limit at 0.5 radian.

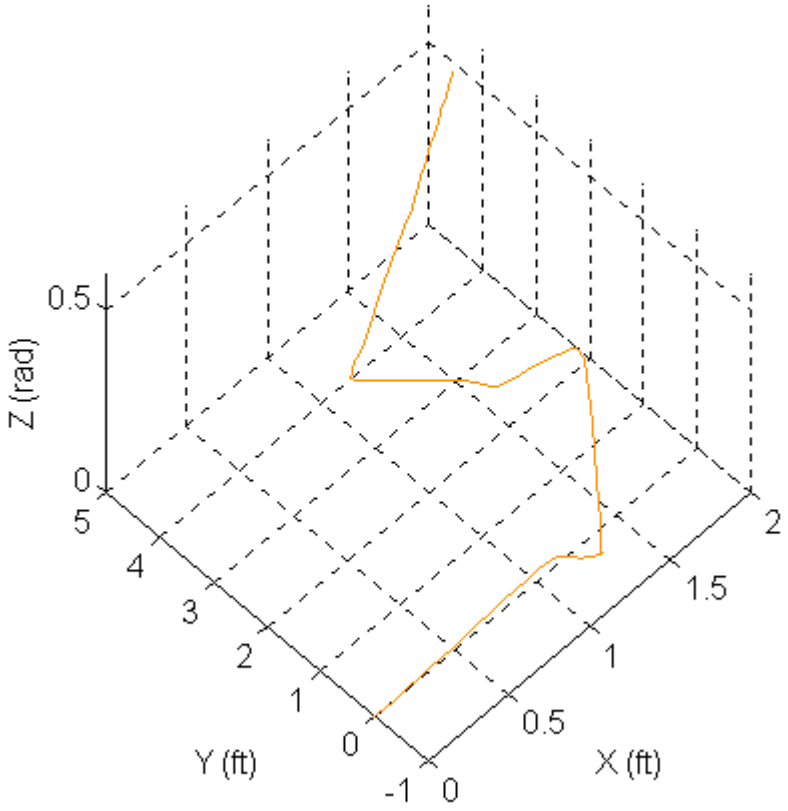


Figure 6.10. A 3-D wall : 60 degree x-y wall with q wall at 0.5 rad.

In addition to straight wall, Scooter implemented a virtual ellipse. Refer to Fig. 6.11. The area inside the ellipse is the free area and the circumference is the virtual wall that preventing Scooter out of the ellipse boundary.

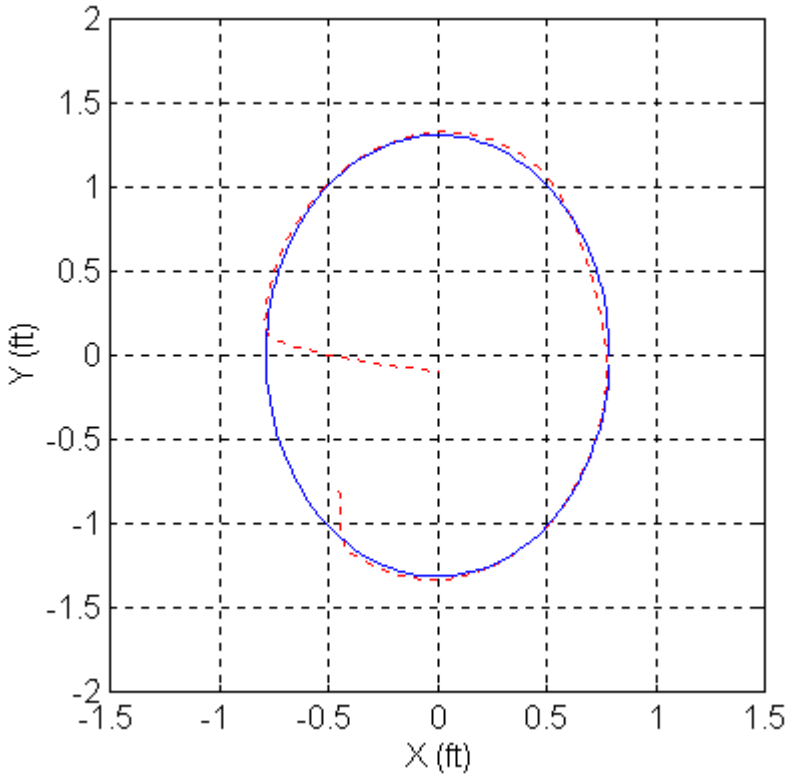


Figure 6.11. Virtual ellipse

In some applications, there are a need for using both virtual walls for gross motion and path tracking for fine motion with better accuracy. To accomplish this virtual funnel can be use to ensure a smooth transition. Figure 6.12 shows Scooter actual path. The virtual funnel begins with a large entry and gradually reduces to a narrow channel, which can eventually become a path. In this application, Scooter also implemented overlay constraints, which described early in Chapter 4.

Users of Scooter typically report that interaction with virtual walls is very pleasing. The walls are hard, smooth, and very inviting. Because the cobot conserves momentum by redirecting

motion rather than inhibing motion, the apparent mass of the cobot (with or without a payload) is generally quite small. An operator requires a few pounds of applied force to maneuver Scooter with 250 lbs payload.

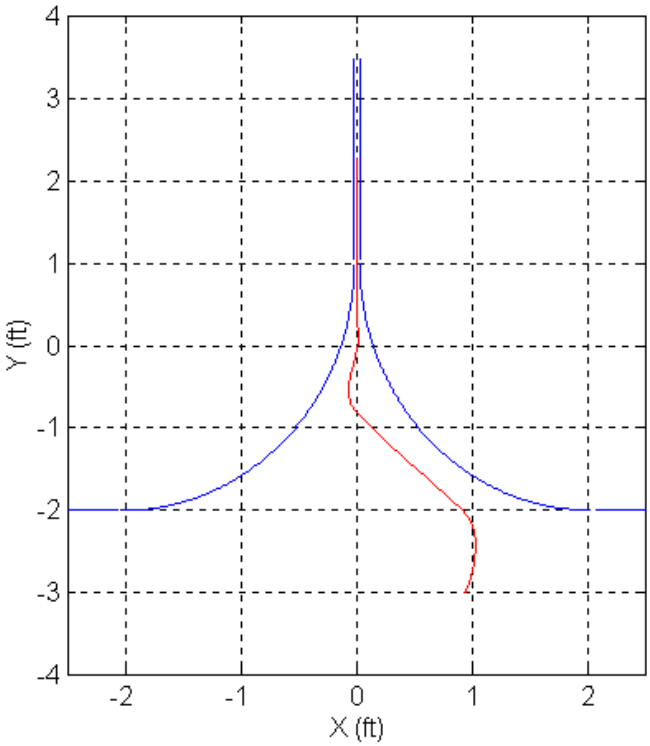


Figure 6.12. Virtual funnel

CHAPTER 7

INDUSTRIAL COBOTS

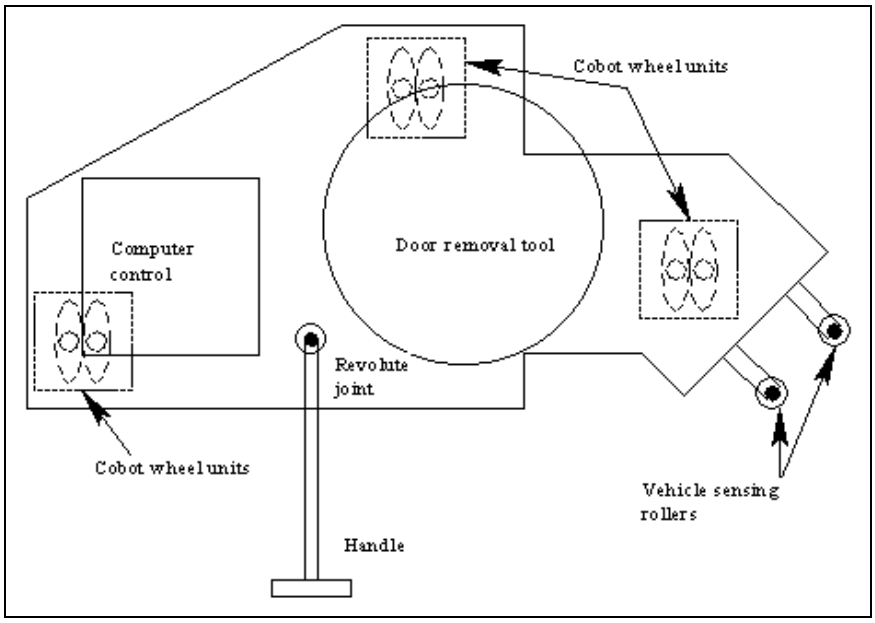
With the promising performance of Scooter, we extended our efforts to the industrial world. We developed a prototype industrial cobot for use on an automobile assembly line where virtual walls and virtual guidance play important roles in improving ergonomics, increasing productivity and quality. Section 7.1 presents the first industrial cobot prototype, the "Cobot door unloader". The cobot door unloader is a wheel-based device designed to take a door off of a vehicle on an assembly line. Section 7.1.1 and 7.1.2 provide details of the cobot and its task. Experimental results are given in Section 7.1.3.

A second industrial prototype, an overhead rail cobot, is also described. This cobot uses spherical joints. Section 7.2. briefly described this prototype.

7.1 Cobot Door Unloader for an Automobile Assembly Line

7.1.1 Description

In collaboration with General Motors our group at Northwestern University built and tested a proof-of-concept floor-based cobot, which is now in a process validation laboratory at GM's Tech Center in Warren, MI. Our application was the "doors-off[4]" task in which the vehicle's doors are removed from the empty auto body, just after painting and prior to assembly. Manually or with conventional assist devices, the task is problematic due to tight tolerances, highly curved body surfaces, and the need for a vehicle-specific "escape trajectory" to avoid damage. The task requires rotational motion as well as translation, and also involves issues of locating the unloader with respect to an imprecisely situated car and working with a moving line.



[4]Automobiles are typically assembled in three phases: Body shop where the sheet metal is welded, Paint shop where it is painted and General assembly where all sub-systems are mated with the painted shell. To maximize paint quality, the shell of the car is loosely integrate at the end of the body shop and sent in to be painted simultaneously. However, in order to improve production efficiencies (by keeping assembly costs down and by permitting workers access through the entire door opening), doors are taken off the car as soon as it exits the paint shop and enters general assembly. This process for removing the door is called the "doors-off" process. It is one of many steps in the "General Assembly Bill of Process."

Figure 7.1 Top view of cobot door unloader

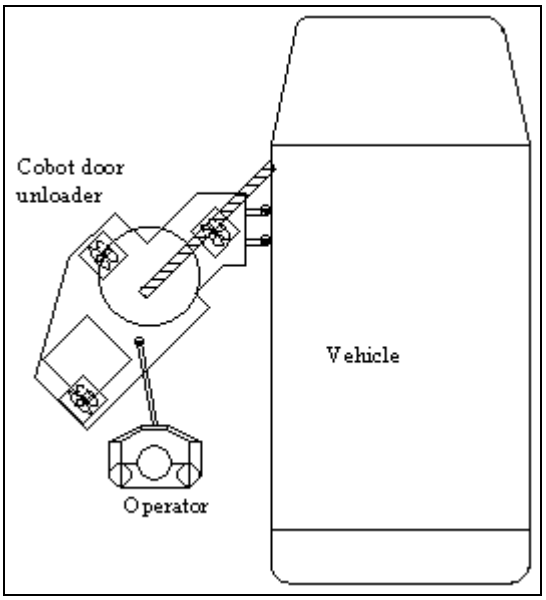


Figure 7.2. An operator removes the door under cobot guidance

The door unloader (See Fig. 7.1) consists of a cobotic platform, computer controller, a handle, door removal tool (a task-specific “tooling” module to grasp and lift the door), and a vehicle sensing system.

The cobot module is a ruggedized Scooter. Each wheel units consists of two wheels, which make it easy to turn and handle larger loads. They are equipped with two set of sensors (not shown in the Figure), a wheel angle sensor, and a wheel steering angle sensor. This cobot also uses a dead-reckoning technique similar to Scooter to calculate its position and velocity.

The handle is no longer a force sensor as it was on Scooter. The operator’s interface to the cobot, by which his or her motion-intention is made known to the controller, is for this application a freely settable handle whose angle is read by a RVDT. The controller reads the

operator's intent expressed via the handle angle, and may modify its motion based on this input, or ignore it, depending on mode of operation.

The vehicle locating system consists of two roller/sliders, which measure the relative geometry (distance, orientation and velocity) between the door unloader and the vehicle as the door is being lifted off. With this information the cobot can position itself relative to the car. The location system plays a significant role in ensuring that the door hinge pins lift off cleanly.

The "tooling" module is designed to lift the door off of its two hinges while ensuring that the door is securely held by the gripper.

Figure 7.3 The cobot door unloader (courtesy of General Motors Company).

7.1.2 Task Implementations

The cobot's tasks are to

- Direct the operator towards the vehicle and later to the door drop-off station, maintaining the proper orientation for each.

- Assume the correct orientation and lateral distance with respect to the vehicle to permit the lifting off of the door.
- Perform direction changes at the operator's command while mitigating the apparent inertia of the door unloader.

The task cycle (see Fig. 7.4) is a fairly simple one. The operator starts from the home position (typically line-side) with the cobot in virtual caster mode. The operator steers the device towards the vehicle, while the cobot automatically orients[5] itself with respect to the car via a *gross approach path*. Once the vehicle sensing system engages the side of the car, the unloader switches to *fine approach path mode*, adjusting its orientation to match that of the particular vehicle. This mode also controls the offset distance between the vehicle and the unloader. The operator pushes a button to grasp the door and another to lift it. Upon door lift-off the velocities of the unloader and the vehicle become independent. The system triggers on this signal to execute an *escape path*, which guides the door away from the vehicle as quickly and safely as possible. The operator now regains control of the unloader and steers it in virtual caster mode

[5] The relative orientation between the door unloader and the vehicle is optimized to ensure that the door does not hit any "Class A" surface on the front fender during the operation. In the test vehicle that we were using, the desired angle was 63°.

towards the drop off station. The unloader orients itself with respect to the drop off station as it approaches. When the vehicle sensing system engages, the unloader executes a *fine return path* that tunes its orientation and position for dropping off the door. The operator transfers the door to the door trim line and is then ready to repeat the cycle.

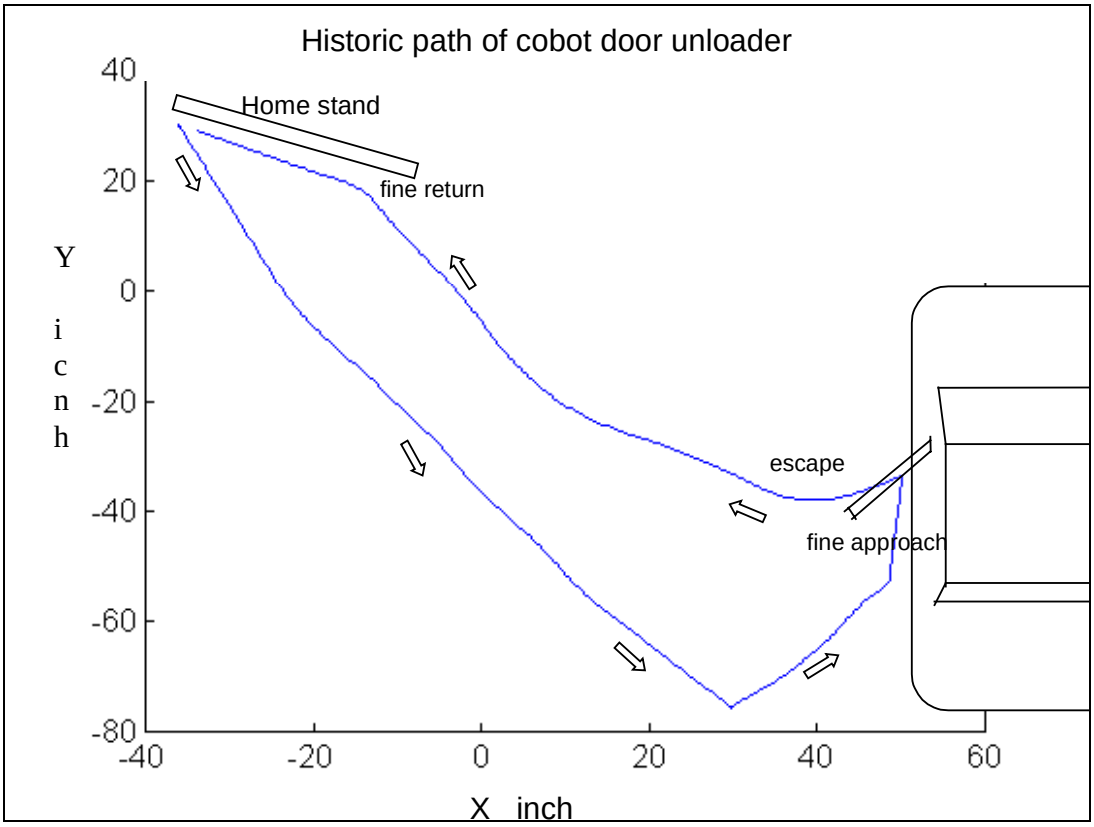


Figure 7.4. A typical trajectory followed by the cobotic door unloader. For purposes of visualization, the vehicle and the home stand part of the drop off station are also shown (though, not to scale).

The door unloader uses dead-reckoning (based on the rotation of the wheels) to calculate its position at any instant of time. This method is susceptible to accumulated errors. To overcome this problem we exploited the fact that the device has a fixed point during every cycle -- at the

drop off station. Thus, as the door is being transferred to the door trim line the device is 'zeroed' out. Figure 7.4 shows a typical path followed by the operator during a cycle.

One motivation for this work was inertia management -- handling motions so that the apparent inertia that the operator feels is minimized. Despite the design team's concern about a loaded mass in excess of 136 kgs (300 lbs), most operators reported finding the door unloader to be very easy to maneuver -- startup force was typically less than 25N (5 pounds). Low rolling friction contributes to this good result, and equally importantly the cobot does not "waste" momentum – changes of direction are handled by steering rather than braking. The operator, consequently, does not have to supply acceleration and deceleration forces that commonly cause fatigue.

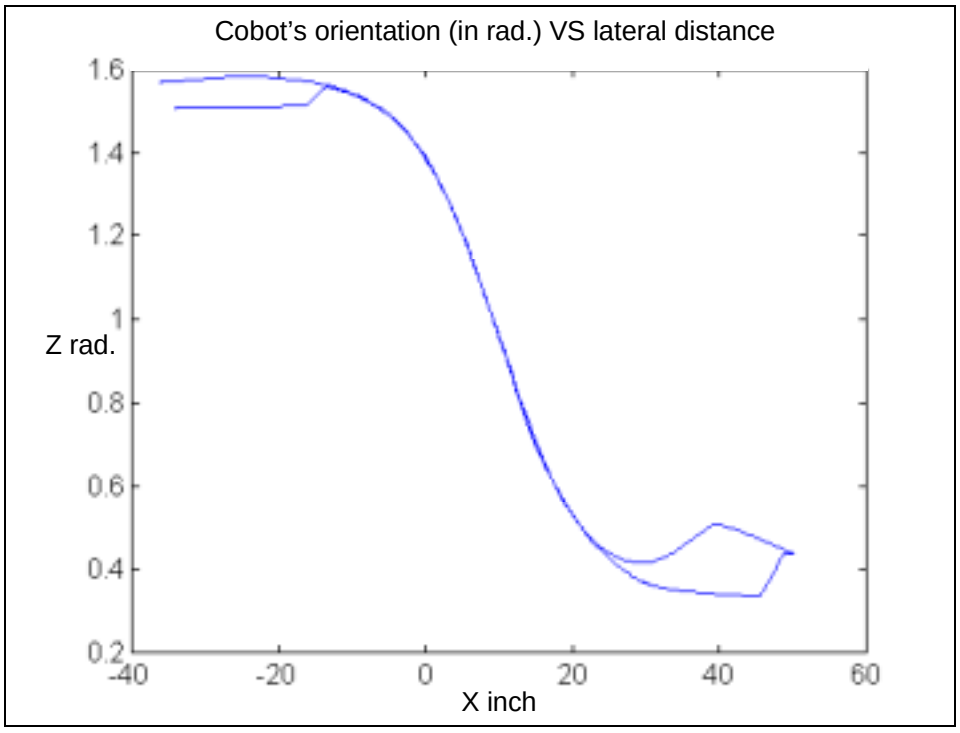


Figure 7.5. The orientation trajectory corresponding to the (x,y) trajectory shown in Figure 7.4.

Preliminary tests indicate that the prototype door unloader promises significant improvements in

(1) **ergonomics**, by minimizing the operator's twisting and lateral forces;

(2) **productivity**, by decreasing the time to master the use of the device and by reducing cycle time;

(3) **quality,** by reducing the scope for human error;

(4) **safety,** because of the passivity of the cobot; and

(5) **flexibility,** because it is programmable. It allows several models to assembly in the same line.

Efforts to quantify these improvements are on-going.

7.2 An Overhead Rail Cobot

This thesis has primarily focused on planar wheel-based cobots. This section, however, describes a spherical joint cobot, which is mounted on a typical industrial rail system. The rail system consists of two parallel stationary rails. Riding on these rails are one or two “bridge” rails which span the distance between the stationary rails. The rail system is unpowered and uses low-friction trolleys and bearings. Our objective was to implement virtual surfaces in the rail system’s large workspace by making it into a cobot.

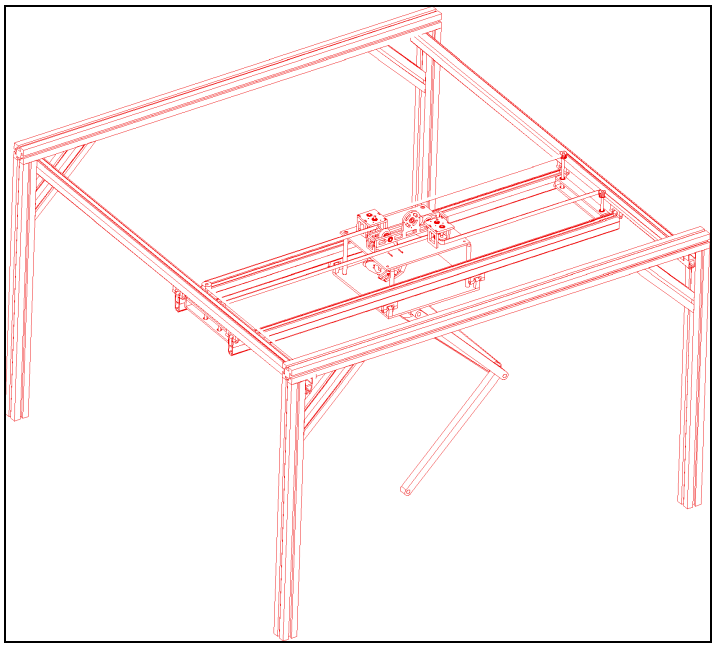


Figure 7.6 Overhead rail cobot

The overhead rail cobot has a 2D translational workspace (x,y), just like unicycle cobot described earlier. The spherical joints (CVTs) are mounted on a carriage that is attached to the rail system (Fig. 7.6). One drive roller from each CVT is connected to a pulley, and the angular velocity of these pulleys are designated ω_1, and ω_2, respectively. The other drive rollers are connected together by a short belt, which can be driven by a 200 watt *power assist* motor. As shown in Fig. 7.7, ω_1 and ω_2 are coupled to the rail system by timing belts. In terms of the timing belt geometry, the translational velocities V_x and V_y can be written as: $V_x = \omega_1 - \omega_2$, and $V_y = \omega_1$

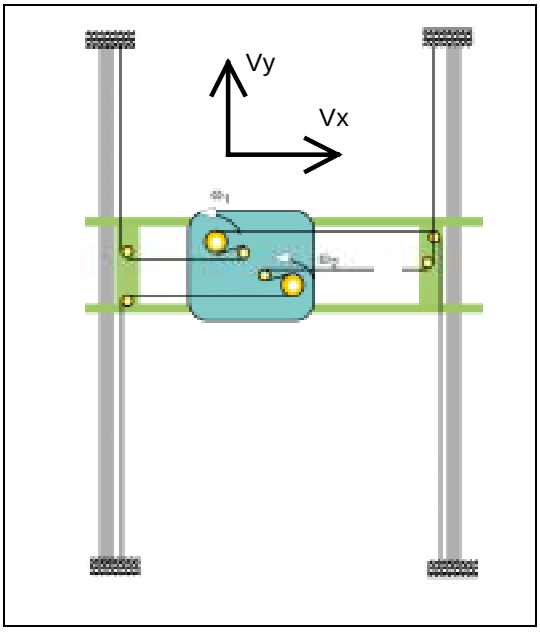


Figure 7.7 Timing belt mechanism coupled together by two CVTs.

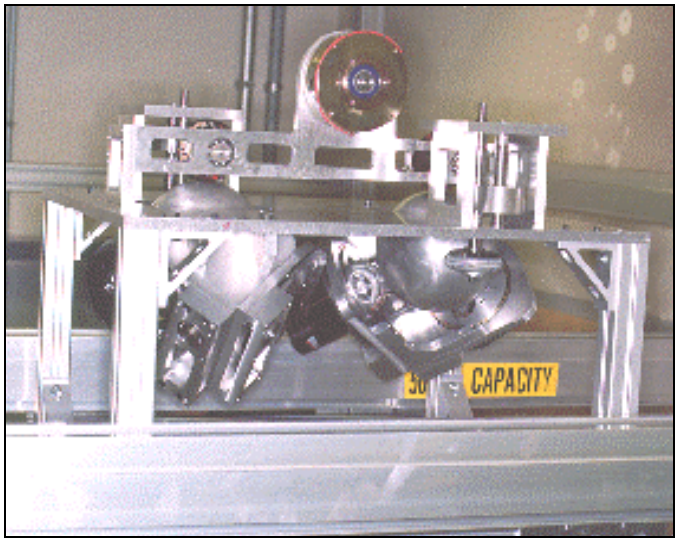


Figure 7.8 CVT mechanism of the overhead rail cobot (courtesy of Ford Motor Company)

Unlike cobot door unloader, this cobot can add provide some energy to the motion of the payload. The purpose of this "power assist" is to overcome friction in the timing belt mechanism. Moreover the power assist also makes the 400-lb load significantly easier to move. This powered rail cobot is presently at Ford Motor Company's Advanced Manufacturing Technology Division. It has demonstrated significant advantages relative to a power assist system based strictly on servo technology. For example, a 2,900 watt motor would be required to move this size of payload at 2m/s speed in a circle path of 50 cm radius. The cobot can accomplish this task with only a 200 watt motor primarily because it is the CVTs and not the motor which are responsible for generating constraint forces. Further details of the overhead rail cobot and other higher dimensional cobots are available in (Peshkin et al., 1998).

CHAPTER 8

CONCLUSIONS AND FUTURE WORK

8.1 Conclusions

We have presented cobots as a novel technology for human-robot collaboration in the context of material handling. We introduced the virtual wall concept and demonstrated its implementation through cobot prototypes. The benefits of virtual walls are significant. They can increase productivity and quality, and improve ergonomics. Since cobots employ steered joints, virtual walls generated by cobots are smooth, hard, frictionless, and intrinsically safe.

There are two types of cobotic joints, the translational joint (wheel joint) and the rotational joint (spherical joint). Since the transmission ratios are infinitely adjustable, both cobotic joints are classified as a *continuously variable transmissions* (CVTs). With these joints as foundation, we presented a variety of cobot architectures. The simplest 2 dimensional C-Space device is the unicycle cobot, which consists of one wheel joint. The bicycle cobot has two wheel joints and hence has a 3 dimensional C-Space. However, to get rid of a singularity and provide a supporting frame, the tricycle cobot employs 3 wheel joints. In addition to wheel joints, we described an arm-like cobot using spherical joints. Even though cobots' primary purpose is to implement virtual walls and guidance, power assist can be added to help an operator start/stop a load as well as overcome friction. Power assist cobots are very efficient because they require

only a single motor to assist the operator along the instantaneous direction of motion. Virtual surfaces continue to be implemented via cobotic steering.

We built the unicycle cobot prototype to test our concepts. The unicycle cobot can implement virtual walls in the *x*-*y* plane. In order to explore cobots in higher dimensional C-Space, we built Scooter. Scooter can implement virtual walls in x-y-θ space. Since continuous rotation of the steering axis is very important, Scooter uses glide wheel sensors to infer wheel and cobot velocities.

We also developed cobot control for higher dimensional C-Space, especially the control of Scooter. Rather than time as the independent variable, cobots use a path length parameter for path planning and virtual walls. This is appropriate because the human operator is in control of the time course. In order to control each wheel properly, we developed curvature transformations, which enable C-Space curvature to project into wheel space curvature.

Scooter performed extremely well. Virtual caster control is very smooth and pleasing. Virtual walls implemented by Scooter are very convincing. They are hard, smooth, and inviting. Using them, an operator can comfortably operate Scooter at up to 2 m/s with a payload up to 200 lbs.

With the success of Scooter, General Motors Corporation and the Northwestern team extended our effort to industrial application. We build the first floor based industrial cobot, the cobot door unloader. The unloader is currently displayed at a GM facility. The preliminary data show that the unloader significantly improves ergonomics, reduces cycle time by 50%, and improves process quality by eliminating collisions between parts.

Cobots conserve energy by steering rather than braking or actuating motion. Changing the direction of a payload does not require force from the operator. Virtual walls implemented by cobots are smooth, frictionless, and intrinsically passive. Cobots can increase productivity and quality, and improve ergonomics. We believe that this work makes a significant contribution to the fields of human-robot interaction, haptic interface, material handling, ergonomics, and intelligent assist devices.

8.2 Future Work

Global positioning / Part referenced sensing

Floor based cobots such as Scooter currently use dead reckoning to estimate velocity and position. The disadvantage of this method is error accumulation. One solution to this problem is global positioning. The global positioning system may not need to be very accurate because the cobot velocity can be estimated as it is done currently, by high-resolution sensors like the glide wheels.

For some repeated-cycle applications, such as general assembly in automobile plants, the dead reckoning technique may be adequate, provided that there is a home reset position for every cycle. In addition, it is often useful to measure the cobot configuration relative to a part, such as an automobile chassis. For both purposes, a general approach to part-referenced sensing would be useful.

Velocity sensor

Cobots require continuous rotation of the CVT steering axes. This discouraged us from placing a conventional sensor such as an encoder at the wheel. Slip rings can be used to transmit the sensor signal but noise and lifetime are concerns. A novel velocity sensor which would transmit signals across the rotating steering shaft or through a hollow steering shaft, would simplify wheel unit designs.

Spherical joint development

Cobots take advantage of the reaction force generated by friction. There is a need to study materials, especially for use in the spherical CVT, which can provide high friction and long lifetime. Another interesting topic is to quantify the power handling of CVTs.

Controls

There are several issues related to the control and programming of cobots. A GUI (graphical user interface) can potentially be quite helpful in programming virtual walls and paths. Another interesting area of research is planning cobot paths in order to minimize the operator applied force and guarantee continuity in curvatures.

REFERENCES

1 Akella, P., Peshkin, M., Colgate, J. E., Wannasuphoprasit, W., Nidamaluri, N., Wells, J., Holland, S., Pearson, T., and Peacock B., (1999), “Cobots for the Automobile Assembly Line”, I IEEE International Conference on Robotics and Automation. Detroit, Michigan.

2 Baraff, D. (1992), “Dynamic Simulation of Non-Penetrating Rigid Bodies,” Ph.D. dissertation, Cornell Univ., Dept. of Computer Science.

Baraff, D. (1994), “Fast Contact Computation for Nonpenetrating Rigid Bodies,” In Proceedings, SIGGRAPH'94, pp. 23-34.

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

5 Colgate, J.E., and Brown , J.M. (1994), “Factors Affecting the Z-witdh of a Haptic Display,” Proc. International Conference on Robotics and Automation, San Diego, CA: IEEE R&A Society, 4, pp. 3205-10.

6 Colgate, J.E., Peshkin, M.A. and Wannasuphoprasit, W., (1996), ”Nonholonomic Haptic Display,” IEEE International Conference on Robotics and Automation. Minneapolis, pp. 539-544.

7 Colgate, J. E., W. Wannasuphoprasit and M. A. Peshkin, (1997), “Cobots: Robots for Collaboration with Human Operators,” International Mechanical Engineering Congress and Exposition ASME, Atlanta. pp. 433-440.

8 Colgate, J.E., Stanley, M.C., & Brown , J.M. (1995), “Issues in the Haptic Display of Tool use” IEEE/RSJ International Conference on Intelligent Robots and Systems, Pittsburgh, Pennsylvania, pp. 140-145.

Colgate, J.E., Schenkel, G.G., (1997), “Passivity of a Class of Sampled-Data Systems: Application to Haptic Interface,” Journal of Robotic Systems, Vol. 14, No. 1, pp. 37-47.

10 Gillespie, R., B., Colgate, J., E. and Peshkin, M., (1999), “A General Framework for Cobot Control”, IEEE Transactions of Robotics and Automations., TRAS99060.

Gillespie, R., B., Moore C. A., Peshkin, M. and, Colgate, J., E (1999), “Kinematic Creep in Continuously Variable Transmissions”, IEEE Transactions of Robotics and Automations.

Gillespie, R., B., Freeman, R., Wannasuphoprasit, W., Colgate, J., E. and Peshkin, M., (1999), “Stable Path Following Control for Cobots”, Submitted to IEEE Transactions of Robotics and Automations.

13 Hannaford, B., & Anderson, R., (1988), “Experimental and Simulation Studies of Hard Contact in Force Reflecting Teleoperation,” IEEE. CH2555-1/88/0000/0584, pp. 584-9.

14 Hirota, K., & Hirose, M. “Development of Surface Display,” IEEE. 0-7803-1363-1/93, (pp. 256-62).

15 Kazerooni, H., (1996), "The Human Power Amplifier Technology at the University of California, Berkeley", Journal of Robotics and Autonomous Systems, Elsevier, Volume 19, pp. 179-187.

16 Kelley, A. J., & Salcudean, S.E. (1994). “On the Development of a Force-Feedback Mouse and Its Integration into a Graphical User Interface,” In C.J. Radcliffe (Ed.), International Mechanical Engineering Congress and Exposition, DSC 55-1 (pp. 287-94). Chicago

17 Massie, T.H., Salisbury, J.K., (1994). The PHANToM Haptic Interface: A Device for Probing Virtual Objects. In C.J. Radcliffe (Ed.) International Mechanical Engineering Exposition and Congress, DSC 55-1, (pp. 295-302). Chicago

18 Millman, P.A., Stanley M.A., & Colgate, J.E. (1993). Design of A High Performance Haptic Interface to Virtual Environment. In IEEE Virtual Reality Annual International Symposium, (pp 216-22) Seattle Washington.

19 Mirtich, B., & Canny, J. Impulse-Based Simulation University of California, Berkeley, Department of Computer Science.

Moore, C. A., Peshkin M. A. and Colgate J. E., (1999), “Design of a 3R Cobot Using Continuously Variable Transmissions”, Proceedings of the IEEE International Conference on Robotics and Automation.

21 Moore, C. A., (1997), “Continuously Variable Transmission for Serial Link Cobot Architectures ”, Master Thesis, Mechanical Engineering, Northwestern University.

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

23 Peshkin, M., J. E. Colgate and C. Moore, (1996), “Constraint Machines Based on Continuously Variable Transmissions, for Haptic Interaction with People,” IEEE International Conference on Robotics and Automation. pp. 551-556.

24 Peshkin, M., Colgate, J. E., Wannasuphoprasit, W., Moore, C., Gillespie B., Akella, P., "Cobot Architechture," (1999) IEEE Transactions on Robotics and Automation. TRAS99061.

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

26 Russo, M., & Tadros, A. (1992). Controlling Dissipative Magnetic Particle Brakes in Force Reflective Devices. In H. Kazerooni (Ed.), ASME Winter Annual Meting, (pp. 63-70). Anaheim, California.

27 Salcudean, S. and Wong, N.M. A Force Reflecting Teleoperation System with Magnetically Levitated Master and Wrist. Proc. IEEE ICRA. (pp. 1420-26) Nice, France

28 Santos, M. J., “Extreme Joystick : A Cobot with Stored Energy”, 1997 Ph.D. Proposal, Mechanical Engineering Department, Northwestern University.

29 Schimmels, J.M., & Peshkin, M.A. (1991). Force-Assemblaility: Insertion of a Workpiece into a Fixture Guided by Contact Forces Alone. In IEEE 1991 International Conference on Robotics and Automation., (pp. 1296-1301). Sacramento, California.

30 Troccaz, J., & Delnondedieu Y, “ PADyC: A Passive Arm with Dynamic Constraints,” In IEEE. Second Annual International Symposium on Medical Robotics and Computer Assisted Surgery, (pp. 173-80). Baltimore, Maryland.

31 Wannasuphoprasit, W., Gillespie, R. Brent, Colgate, J. E. and Peshkin, M. A., (1997), "Cobot Control," IEEE International Conference on Robotics and Automation,Albuquerque, Vol. 4, pp. 3571-3576.

32 Wannasuphoprasit, W., Akella, P., Peshkin, M. A. and Colgate, J. E., (1998),"Cobots, A Novel Material Handling Technology," 98-WA/MH-2, Presented at the ASME International Mechanical Engineering Congress & Exposition, Anaheim, California. [ASME Material Handling Engineering Division Best Conference Paper]

VITA

Witaya Wannasuphoprasit, also know as “Wit”, born and grew up in a machine shop at Bangkok, Thailand. He received B.M.E. from KingMongkut’s Institute of Technology (Ladkrabang) with honors in 1989. After two years on teaching in Mechanical Engineering Department at Chulalongkorn University (Bangkok), he came to the United States. He received his M.S. in 1993 and Ph.D. in 1999 at Northwestern University both in Mechanical Engineering. In 1998, he won the first ever best paper award at 1998 International Mechanical Engineering Congress and Exposition by ASME International Material Handling Engineering Division. In addition, he was a co-author of the paper that won the 1996 best conference paper award at IEEE International Conference on Robotics and Automation.

PUBLICATIONS

Wannasuphoprasit, W., Akella, P., Peshkin, M. A. and Colgate, J. E., (1998), "Cobots, A Novel Material Handling Technology," 98-WA/MH-2, Presented at the ASME International Mechanical Engineering Congress & Exposition, Anaheim, California.

Wannasuphoprasit, W., Gillespie, R. Brent, Colgate, J. E. and Peshkin, M. A., (1997), "Cobot Control," IEEE International Conference on Robotics and Automation,Albuquerque, Vol. 4, pp. 3571-3576.

Colgate, J. E., W. Wannasuphoprasit and M. A. Peshkin, (1996), “Cobots: Robots for Collaboration with Human Operators,” International Mechanical Engineering Congress and Exposition ASME, Atlanta. pp. 433-440.

Colgate, J.E., Peshkin, M.A. and Wannasuphoprasit, W., (1996), ”Nonholonomic Haptic Display,” IEEE International Conference on Robotics and Automation. Minneapolis, pp. 539-544.

Colgate, J.E., Matsumoto, H. and Wannasuphoprasit, W., (1993), “Linear Electrostatic Actuators: Gap Maintenance Via Fluid Bearing,” Robotics & Computer-Integrated Manufacturing, Vol 10, No 5, pp. 365-376.