Cobot Implementation of Virtual Paths and 3D Virtual Surfaces
Carl Moore, Michael Peshkin, J. Edward Colgate
IEEE Transactions on Robotics and Automation, volume 19 (2003), number 2 pp. 347-351.

Short Papers

Cobot Implementation of Virtual Paths and 3-D Virtual Surfaces

Carl A. Moore, Jr., Michael A. Peshkin, and J. Edward Colgate

***Abstract*—Cobots are devices for human/robot interaction, in which axes of motion are coupled to one another by computer-controlled continuously variable transmissions rather than individually driven by servomotors. We have recently built a cobot with a three-dimensional workspace and a 3-revolute parallelogram-type mechanism. Here we present control methods for the display of virtual surfaces and for free mode in which the cobot endpoint moves as if it were unconstrained. We provide experimental results on the performance of the free, virtual path, and virtual surface controllers.**

***Index Terms*—Cobot, haptics, human/robot collaboration, intelligent assist device (IAD), nonholonomic, virtual constraint.**

I. Introduction

Robotic researchers have suggested that many tasks can be improved through the use of robot-created virtual constraints and surfaces that redirect undesirable user motions to useful directions. For example, Akella [1] found that constraining the motion of a wheeled cart with virtual guide rails could dramatically decrease the effort involved in material-handling operations. Rosenberg [2] has shown that teleoperation tasks, such as remote peg in hole, can be improved through the use of virtual walls that have the effect of filtering out user forces that would drive the slave end-effector off of an acceptable approach. Z-KAT, a surgical robotics firm located in Hollywood, FL, is using the WAM robot from Barrett Technology to render surfaces that constrain the motion of surgical instruments to regions determined preoperatively [3]. Industrial designers are also finding uses for virtual surfaces. For example, Stewart *et al.* [4] of Ford Research Labs is developing a system that virtually renders an automobile surface so that changes to CAD models can be experienced virtually.

One quality measure of a virtual display system is its ability to produce surfaces that are hard and nearly frictionless. A hard virtual surface, which does not deform under user forces, gives the impression that one is interacting with an actual constraint. A nearly frictionless surface does not dissuade a user from interacting with it by dissipating system energy as a dissipative surface might.

Cobots are able to create high quality virtual surfaces by virtue of continuously variable transmissions (CVTs). The CVTs are kinematically coupled to the cobot's joint motions such that the cobot is inherently mobile in at least one degree of freedom (DOF). The user controls the speed of the cobot along this DOF. To display a virtual surface, the cobot "steers" the CVTs so that the allowed DOF is held parallel to the desired virtual surface. The user's intent is monitored, usually using a force sensor. Forces into the surface are passively resisted by kinematic rolling constraints inside the CVTs. Forces off of the surface are responded to by steering the allowed DOF parallel to the user's desired motion. Away from the surface, the allowed DOF is continuously aligned with the user's intended motion, giving the perception that the cobot has as many DOFs as its task-space dimensionality.

Though a robot using force-following control can approximate a cobot's free and surface modes mentioned above, the cobot implementation is quite different. In force following, a robot's joints are actuated to drive its endpoint in the direction of the user's forces, and forces into a constraint are resisted by equal and opposite actuator torques. The entire process is an active one with many well-documented safety and stability issues [5]. In contrast, cobots are generally passive mechanisms that respond to user's forces by redirecting them; no energy is imparted to the endpoint. On the other hand, a passive cobot cannot display the active effects (i.e., springs and masses) desired in many virtual environments. For these applications, active or powered cobots have been created using a chain of parallel-connected CVTs [6]. In an active cobot, each joint's velocity is coupled to a common one, called the *internal motion*, which is directly proportional to the cobot's endpoint speed. Power is added by connecting an actuator to the internal motion. However, since all cobots have fewer mechanical DOFs than their task-space dimensionality, the actuator has fewer DOFs along which it must accelerate the cobot's mass. This fact permits a lower-power actuator to be used. To date, powered cobots have been created using an actuator that has no more power than the average human user. So, while an active cobot inherits many of the stability concerns of a traditional robot, it retains a safety advantage.

In this paper, we develop a virtual path and three-dimensional (3-D) surface controller for a cobot with parallel-connected CVTs. As mentioned above, cobots with parallel-connected CVTs are interesting because they are the basis for active or powered cobots. Section II will cover the design of the Arm cobot and its kinematics. Sections III–V will present the free, path, and surface controllers, respectively, including experimental verifications. Section VI contains conclusions.

II. Arm Cobot Design and Kinematics

The Arm cobot (Fig. 1) is a three-joint parallelogram link manipulator [7]. It has a reach of over 90 cm, and the origin of its three joints is 1.5 m above the floor. A force sensor is attached to the end-effector to measure the user's intent, and the links are counterbalanced by masses on the opposite sides of two joints.

The three joint rotations are mechanically coupled to the drive rollers of three rotational CVTs [7]. Each CVT's drive roller, steering rollers, and common wheel are in compressed rolling contact with the central sphere. The rolling constraints combine so that for each CVT, the ratio of the common wheel velocity $\dot{\theta}_0$ to drive roller velocity ω_i is a function of the steering roller angle γ_i

$$\mathbf{G}_i = \tan\gamma_i = \frac{\omega_i r_d}{\dot{\theta}_0 r_0} \tag{1}$$

where r_d and r_0 are the radii of the drive roller and common wheel, respectively. $\mathbf{G}_i$ is called the CVT transmission ratio.

The Arm cobot is inherently mobile in one DOF. To permit arbitrary motion of the Arm's endpoint, the controller must steer the angles γ so that the allowed DOF is brought parallel to a desired direction $\mathbf{T}$.

Manuscript received December 20, 2001; revised June 27, 2002. This paper was recommended for publication by Associate Editor H. Arai and Editor A. De Luca upon evaluation of the reviewers' comments. This work was supported in part by the Office of Naval Research under Grant N00014-92-J-1252. This paper was presented in part at the 2002 IEEE Robotics and Automation Conference and in part at the 2000 7th Mechatronics Forum International Conference.

C. A. Moore, Jr. is with the Mechanical Engineering Department, Florida A&M and Florida State University, Tallahassee, FL 32310 USA (e-mail: camoore@eng.fsu.edu).

M. A. Peshkin and J. E. Colgate are with the Mechanical Engineering Department, Northwestern University, Evanston, IL 60208 USA (e-mail: peshkin@northwestern.edu; colgate@northwestern.edu).

Digital Object Identifier 10.1109/TRA.2003.808866

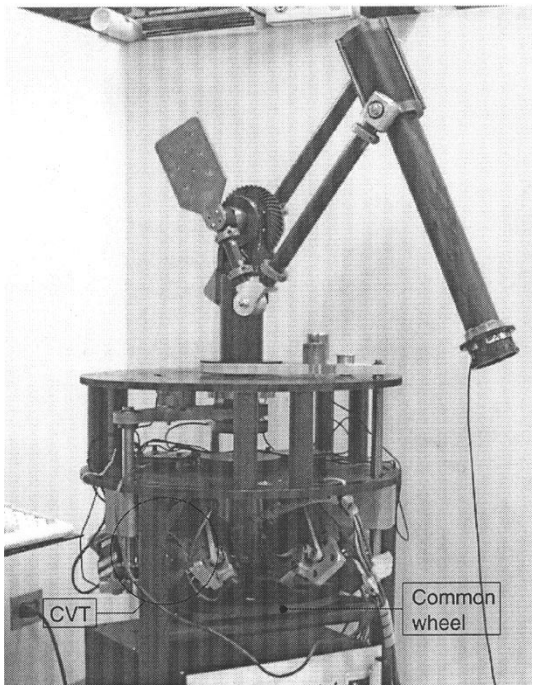

Fig. 1. Arm cobot with three rotational CVTs connected in parallel through a common wheel.

We determine the relationship between the steering angles and $\mathbf{T}$ by substituting the Jacobian relating the differential motion of the CVT drive rollers to the differential motion of the endpoint ($\dot{\mathbf{x}} = \mathbf{J}_{xd}\omega$) into the CVT transmission ratios equation

$$\mathbf{G} = \frac{r_d \mathbf{J}_{xd}^{-1}\dot{\mathbf{x}}}{\dot{\theta}_0 r_0}. \tag{2}$$

With (2), it is possible to calculate the necessary CVT steering angles for a particular endpoint velocity and common wheel speed. A variable k is defined as the ratio of endpoint speed to common wheel speed [8]

$$k|\dot{\mathbf{x}}| = \dot{\theta}_0 r_0. \tag{3}$$

Substituting (3) into (2) yields a relationship between endpoint motion direction $\mathbf{T}$ and the required CVT transmission ratios

$$\mathbf{G} = \frac{r_d \mathbf{J}_{xd}^{-1}}{k}\frac{\dot{\mathbf{x}}}{|\dot{\mathbf{x}}|} = \tilde{k}\mathbf{J}_{xd}^{-1}\mathbf{T} \tag{4}$$

where $\tilde{k}$ is r_d/k.

Equation (4) will now be used to design a free-mode controller that gives the user the perception that the Arm cobot has three instantaneous DOFs instead of one.

III. Free-Mode Control

In free mode, the controller must steer the CVTs so that the Arm's end-effector (inertia $= M$) is free to accelerate according to the user's force $F = Ma$. Because a cobot cannot resist motion parallel to its current allowed DOF $\mathbf{T}_0$, components of force parallel to it, $F_{\|}$, require no CVT change to produce the desired endpoint acceleration $a_{\|} = M^{-1}F_{\|}$. However, components of force perpendicular to $\mathbf{T}_0$, $\mathbf{F}_{\perp}$, require a new heading to produce the acceleration $a_{\perp} = M^{-1}\mathbf{F}_{\perp}$. The heading change $\Delta\mathbf{T}$ can be represented by a desired curvature $k\mathbf{N}$ in task space

$$\mathbf{T} = \mathbf{T}_0 + k\mathbf{N}|\dot{\mathbf{x}}|dt. \tag{5}$$

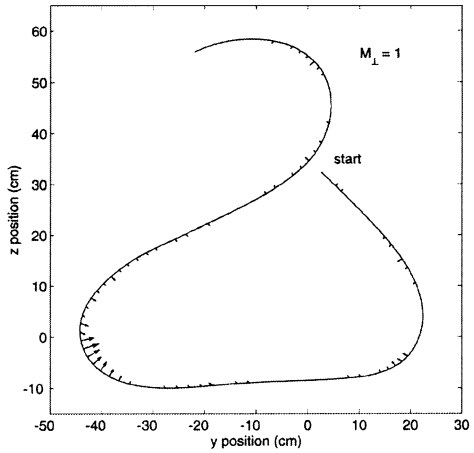

Fig. 2. Path of end-effector under free-mode control (Y-Z projection) includes overlay of user forces with $M_{\perp} = 1$.

Curvature is equal to the perpendicular acceleration divided by the speed squared, so it is related to the user's intent through the perpendicular component of force

$$k\mathbf{N} = \frac{\mathbf{F}_{\perp}M_{\perp}^{-1}}{|\dot{\mathbf{x}}|^2}. \tag{6}$$

Equations (5) and (6) can now be combined with (4) to create a prescription for a CVT steering controller that permits a cobot's endpoint moving at speed $|\dot{\mathbf{x}}|$, with instantaneous heading $\mathbf{T}_0$, to follow the user's intent

$$\mathbf{G} = \tilde{k}\mathbf{J}_{xd}^{-1}\left(\mathbf{T}_0 + \frac{\mathbf{F}_{\perp}M_{\perp}^{-1}}{|\dot{\mathbf{x}}|}dt\right). \tag{7}$$

The variable $M_{\perp}$ is a gain variable that modifies the responsiveness of the endpoint to the user's forces. Lower values of $M_{\perp}$ increase the responsiveness of the cobot's endpoint to perpendicular user forces, and higher ones decrease it.

Note that the desired curvature (6) is not well defined for near-zero velocity. When the velocity is near zero, it is preferable to determine $\mathbf{T}$ without calculating the desired curvature. Since the tangent always points in the direction of motion, the new tangent can be found by integrating the user-desired acceleration $\mathbf{F}M^{-1}$

$$\mathbf{T} = \frac{|\dot{\mathbf{x}}_0|\mathbf{T}_0 + \left(\mathbf{F}_{\|}M_{\|}^{-1} + \mathbf{F}_{\perp}M_{\perp}^{-1}\right)dt}{\left||\dot{\mathbf{x}}_0|\mathbf{T}_0 + \left(\mathbf{F}_{\|}M_{\|}^{-1} + \mathbf{F}_{\perp}M_{\perp}^{-1}\right)dt\right|}. \tag{8}$$

$M_{\|}$ is the parallel analog to the responsiveness control variable $M_{\perp}$. It ensures consistent units and scaling.

The tangent (8) could be called $\mathbf{T}_{\text{slow}}$ since it is used for small values of endpoint speed. It is defined for near-zero endpoint speed unless the endpoint force is also near zero. A software switch is used to make sure that at near-zero speed and force the tangent direction remains aligned to the current. In practice, the controller activates this software switch when endpoint speed is less than 2.54 cm/s and user force is less than 0.045 Kg. These settings enable the Arm cobot to respond smoothly at the start of motion.

$\mathbf{T}_{\text{slow}}$ would be equal to the tangent $\mathbf{T}$ calculated from curvature if the instantaneous endpoint inertia was used to calculate $M_{\|}$. In practice, instead of calculating the cobot's endpoint inertia, we set $M_{\|}$ equal to one, and use an arctangent blending function to switch the controller between $\mathbf{T}_{\text{slow}}$ and $\mathbf{T}$. Satisfactory performance was achieved by setting the inflection point of the blending function to 2.0 cm/s.

To test the response of the free-mode controller, a user attempted to pull the Arm's end-effector along an arbitrary path. The resulting motion is plotted in Fig. 2 along with an overlay of the perpendicular user forces applied during the motion [9]. The path is projected onto

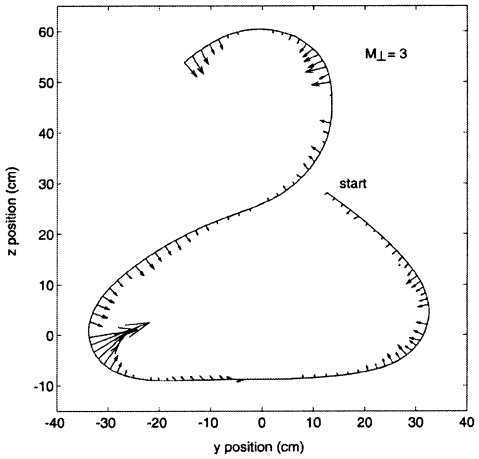

Fig. 3. Path of end-effector under free-mode control (Y-Z projection) includes overlay of user forces with $M_\perp = 3$.

the Y-Z plane for plotting because the majority of the motion occurred there. This and all subsequent experiments in this paper were conducted without using the common wheel to add power.

To verify the relationship between $M_\perp$ and the responsiveness of the cobot, a user attempted to pull the Arm along the same path as that shown in Fig. 2 using a higher $M_\perp$ setting. The effect on the required perpendicular forces is displayed in Fig. 3. As expected, the effort required to execute the path with $M_\perp = 3$ is larger, as represented by the force arrows, than that for $M_\perp = 1$.

Since the endpoint follows any user-desired direction in free mode, it is not possible to retrace a path exactly. However, care was taken to trace nearly the same path at nearly the same speed for both trials above. The average speeds and standard deviations were $\overline{v} = 60.0$ cm/s, $\sigma = 10.0$ cm/s for $M_\perp = 1$ and $\overline{v} = 57.0$ cm/s, $\sigma = 10.0$ cm/s for $M_\perp = 3$. These values suggest that the increase in required force for $M_\perp = 3$ is a result of the larger virtual mass, not a speed differential.

IV. Path Mode

One goal of cobot technology is to enable people to interact with predefined and arbitrarily oriented paths through space. For example, in medical rehabilitation, a patient might exercise a damaged limb by moving it through curved arcs that simulate everyday motions. Cobot technology is an attractive choice for path-constrained human–robot interaction because it enables the creation of paths that are smooth, stable, and infinitely adjustable. However, to follow a predefined path, the cobot requires a "path controller" which keeps it on the path while the user applies forces not parallel to the path.

A. Feedforward Control

Assuming no errors, a feedforward steering controller that confines a cobot moving at speed $|\dot{\mathbf{x}}|$, on a path with instantaneous tangent $\mathbf{T}_p$, and curvature $k_p\mathbf{N}_p$ can be created from (7)

$$\mathbf{G}_i = \tilde{k}\mathbf{J}_{xd}^{-1}\left(\mathbf{T}_p + k_p\mathbf{N}_p|\dot{\mathbf{x}}|dt\right). \tag{9}$$

Of course, it is not possible to have error-free path following using only feedforward control. Nonidealities including steering dynamics and CVT roller slip (lateral and longitudinal creep [10]) create errors that must be corrected using feedback control.

B. Feedback Control

The feedback control used to eliminate path errors is different from that used in traditional robotics. The joint actuators of a robot apply torques to drive the endpoint back to the desired path. Cobots, being generally passive, cannot propel their endpoints. Instead, cobots reduce path error by redirecting user-imposed velocities. The feedback control derivation is as follows.

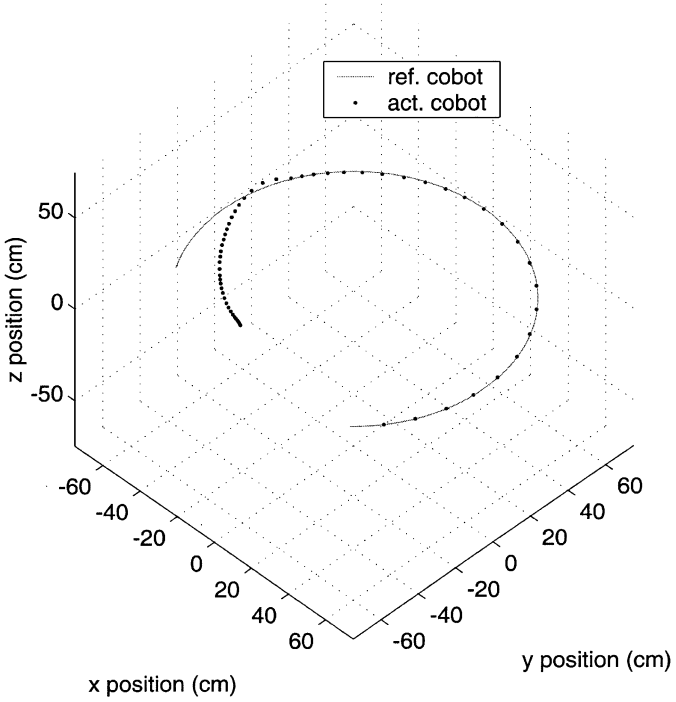

Fig. 4. Virtual path: approach and constrained by a virtual arc.

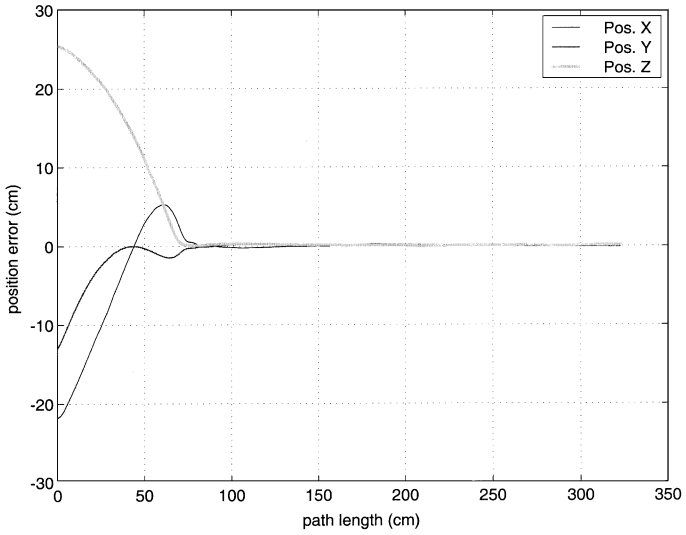

Fig. 5. Virtual path: position error versus path length.

Assume that the actual cobot is at position $\mathbf{R}$ with instantaneous heading $\mathbf{T}$. The expected position $\mathbf{R}_p$ represents the location of a reference cobot that always lies on the planned path with a heading equal to the path tangent $\mathbf{T}_p$ at that point. The point $\mathbf{R}_p$ is the closest point on the desired path to the actual cobot position. The path errors are of two types [11].

Position Error: $\Delta\mathbf{R} = \mathbf{R}_p - \mathbf{R}$ is a vector approximately normal to the planned path and proportional to the distance that the cobot is from it.

Tangent Error: $\Delta\mathbf{T} = \mathbf{T}_p - \mathbf{T}$ is a vector representing the heading error.

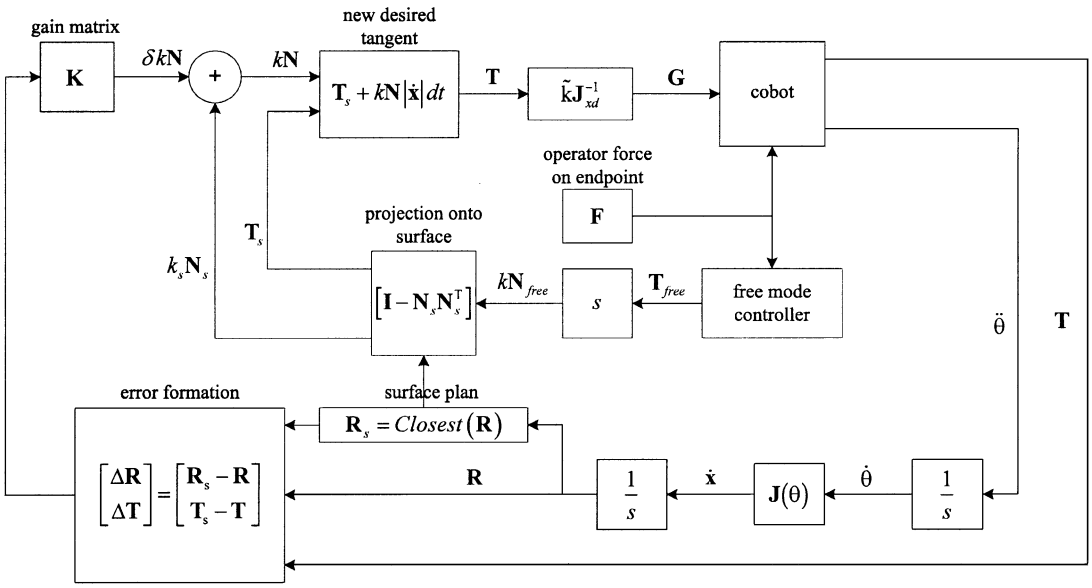

Fig. 6. Surface controller.

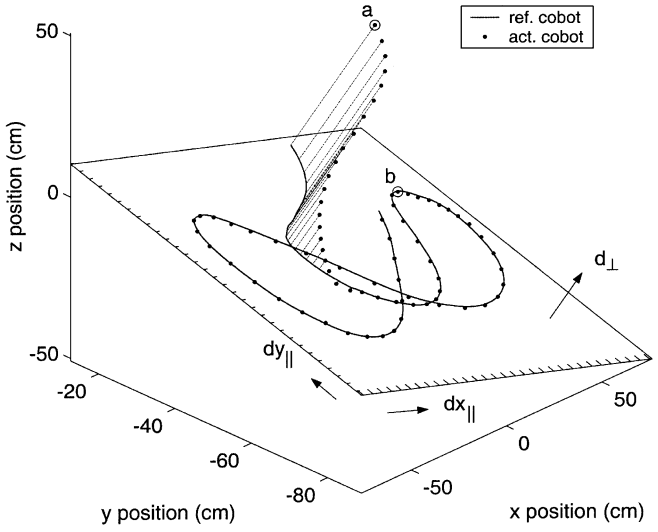

Fig. 7. Path of Arm cobot as it approaches and becomes confined to a virtual planar surface.

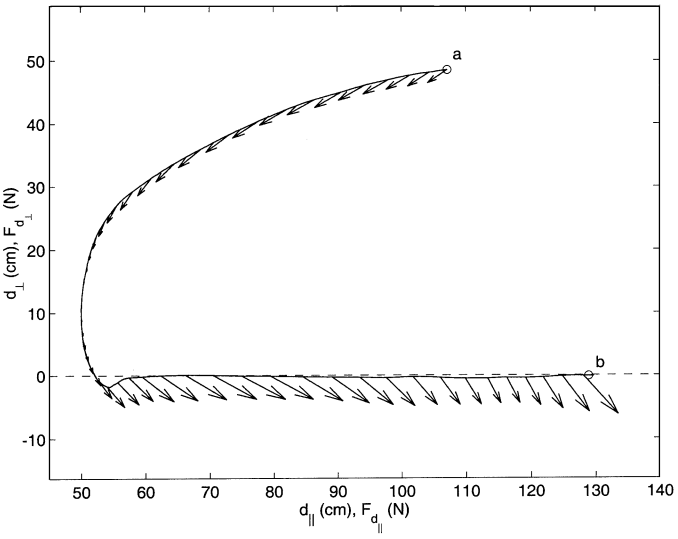

Fig. 8. Virtual surface ($dx_{\|} - d_{\perp}$ projection): endpoint path with an overlay of user forces.

The most obvious approach to reducing cobot path error is to steer back to the point $\mathbf{R}_p$. The steering command produced by the feedforward control is a function of the planned path curvature k_p. Therefore, it is intuitive to create the feedback controller by augmenting k_p by a delta curvature δk, which is a function of the position and heading errors

$$\delta k = K_1 \Delta \mathbf{R} + K_2 \Delta \mathbf{T}. \tag{10}$$

The variables K_1 and K_2 are control gains (proportional and derivative), which also ensure consistent units. The resulting form of the feedback controller is

$$\mathbf{G} = \tilde{k}\mathbf{J}_{xd}^{-1}\left(\mathbf{T}_p + \mathbf{N}_p \cdot (k_p + K_1 \Delta \mathbf{R} + K_2 \Delta \mathbf{T})|\dot{\mathbf{x}}|dt\right). \tag{11}$$

A virtual arc was used to test the path-mode controller. Starting with nonzero position and heading error, the user pushed the cobot while the controller attempted to eliminate the path error. The results are shown in Figs. 4 and 5.

In this experiment, the path controller reduced the path error to 0.2 cm on a virtual arc with a 65.0-cm radius. Backlash in the Arm's joints prohibits significant further reduction of path error.

V. Surface Mode

To create a virtual surface, the Arm's endpoint is confined to a path that lies on the surface. The path is found by projecting the cobot's heading $\mathbf{T}$ and the user's intent $k\mathbf{N}$ onto the surface at a point $\mathbf{R}_s$ closest to the endpoint. Given a surface with normal $\mathbf{N}_s$ at point $\mathbf{R}_s$, the projections result in vectors that are tangent to and perpendicular to a path on the surface

$$\mathbf{T}_s = \left[\mathbf{I} - \mathbf{N}_s \mathbf{N}_s^T\right] \mathbf{T} \tag{12}$$

$$k_s \mathbf{N}_s = \left[\mathbf{I} - \mathbf{N}_s \mathbf{N}_s^{\mathrm{T}}\right] k\mathbf{N}. \tag{13}$$

The virtual surface controller is constructed from the path controller, where the desired path tangent is $\mathbf{T}_s$ instead of $\mathbf{T}_p$, and the desired

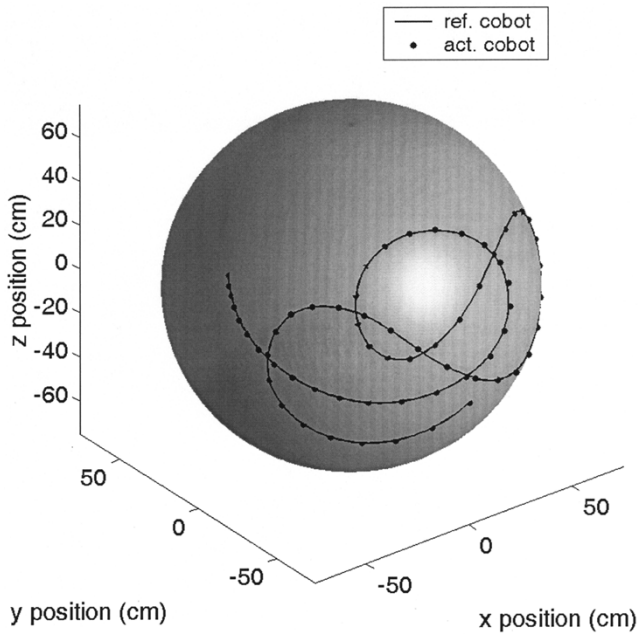

Fig. 9. Virtual surface: constrained to a virtual sphere with radius= 74.0 cm.

curvature is $k_s\mathbf{N}_s$ instead of $k_p\mathbf{N}_p$. A block diagram of the surface controller is shown in Fig. 6.

The surface mode controller confines the cobot's endpoint to the surface while permitting free mode-like motion on the surface.

Displaying a virtual plane tested the surface controller; results are shown in Fig. 7. The user starts to approach the plane from position (a). At this point, the controller is in free mode. The controller switches to surface mode when the endpoint passes through the desired virtual plane. The endpoint is steered back to surface and the remainder of the user's motion is constrained to lie in the plane.

A projection of the path onto the $dx_{\|} - d_{\perp}$ plane (Fig. 8) better illustrates the dip below the plane and includes an overlay of the user's forces applied from the start of motion at position (a) to position (b).

The force vectors are scaled such that one graph unit equals one Newton of force. The largest force recorded during the experiment was 8.75 N.

A virtual sphere demonstrates the ability of the controller to produce curved virtual surfaces as well as planar ones. The plot of the path followed on the virtual sphere is illustrated in Fig. 9.

The surface controller was able to hold the cobot's endpoint on the surface of the sphere with small error. The average resulting radius was 74.15 cm with a standard deviation of 0.28 cm.

VI. Conclusion

The controllers for free, path, and surface mode were successful in displaying the desired virtual effects. Creating a surface controller by augmenting the path controller with projections of the current heading and the desired curvature onto the desired surface proved to be a sound technique, one that is applicable to other cobots with parallel CVT architectures.

A future research goal is to further magnify the user's force by driving the common wheel with a power actuator.

References

[1] P. Akella *et al.*, "Cobots for the automobile assembly line," in *Proc. IEEE Int. Conf. Robotics and Automation*, 1999, pp. 728–733.

[2] L. B. Rosenberg, "Virtual fixtures: Perceptual overlays enhance operator performance in telepresence tasks," Ph.D. dissertation, Dept. Mech. Eng., Stanford Univ., Stanford, CA, 1994.

[3] A. E. Quaid and R. A. Abovitz, "The use of haptic information displays for assisting in the execution of image-guided surgery plans," in *Proc. Computer Assisted Orthopedic Surgery Meeting*, 2001, pp. 339–340.

[4] P. Stewart, Y. Chen, and P. Buttolo, "Direct integration of haptic user interface in CAD systems," in *Proc. Dynamic Systems Control Division (ASME)*, 1997, pp. 93–99.

[5] J. E. Colgate and J. M. Brown, "Factors affecting the Z-width of a haptic display," in *Proc. IEEE Int. Conf. Robotics and Automation*, 1994, pp. 3205–3210.

[6] M. A. Peshkin, J. E. Colgate, W. Wannasuphoprasit, C. A. Moore, B. Gillespie, and P. Akella, "Cobot architecture," *IEEE Trans. Robot. Automat.*, vol. 17, pp. 377–390, Aug. 2001.

[7] C. A. Moore, M. A. Peshkin, and J. E. Colgate, "A three revolute cobot using CVTs in parallel," in *Proc. Int. Mechanical Engineering Congr. Expo. (ASME)*, 1999.

[8] ——, "A controller for simulating freedom of motion for a cobot," in *Proc. 7th Mechatronics Forum Int. Conf.*, 2000.

[9] W. Wannasuphoprasit, B. Gillespie, J. E. Colgate, and M. A. Peshkin, "Cobot control," in *Proc. IEEE Int. Conf. Robotics and Automation*, 1997, pp. 3571–3576.

[10] R. B. Gillespie, C. A. Moore, M. A. Peshkin, and J. E. Colgate, "Kinematic creep in continuously variable transmissions," *ASME J. Mech. Design*, submitted for publication.

[11] R. B. Gillespie, J. E. Colgate, and M. A. Peshkin, "A general framework for cobot control," *IEEE Trans. Robot. Automat.*, vol. 17, pp. 391–401, Aug. 2001.

Robust Scene Reconstruction From an Omnidirectional Vision System

Roland Bunschoten and Ben Kröse

***Abstract*—In this paper, we present an efficient multibaseline stereo algorithm for panoramic image data. We derive a parameterization of epipolar curves in terms of inverse depth. As a result, the search for image correspondences across multiple images can be performed efficiently. Furthermore, depth estimates are obtained directly, thus bypassing the need to perform explicit stereoscopic triangulation. We apply our method to obtain a three-dimensional reconstruction of an environment from a set of panoramic images. The images are acquired by a single omnidirectional vision sensor mounted on top of our mobile robot during navigation. Experimental results demonstrate the effectiveness of our approach.**

***Index Terms*—Multibaseline stereo vision, omnidirectional vision, scene reconstruction.**

I. Introduction

RECENTLY, researchers in the robotics community have begun to consider omnidirectional vision sensors which provide images covering a large part of the hemisphere. Catadioptric omnidirectional vision sensors are quickly gaining popularity. These sensors consist of a camera and a carefully selected mirror–lens combination. They have been proven to be useful for robot environment modeling, both in the

Manuscript received March 25, 2002. This paper was recommended for publication by Associate Editor D. Kriegman and Editor S. Hutchinson upon evaluation of the reviewers' comments.

The authors are with the Intelligent Autonomous Systems Group, Computer Science Institute, Faculty of Science, University of Amsterdam, The Netherlands (e-mail: bunschot@science.uva.nl; krose@science.uva.nl).

Digital Object Identifier 10.1109/TRA.2003.808850