Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

Performance of Surgical Robots with Automatically Generated Spatial Virtual Fixtures

Ming Li and Russell H. Taylor
Department of Computer Science
NSF Engineering Research Center for Computer Integrated Surgical Systems and Technology
The Johns Hopkins University
Baltimore, Maryland 21218, USA
{liming & rht}@cs.jhu.edu

***Abstract* – Our recent work on robotic surgical assistant systems has led us to develop a method to generate spatial motion constraints associated with complex 3D geometry for controlling a robot in a complicated working configuration. An application of specific interest is sinus surgery, in which complicated anatomic structure constrains the motion of the endoscope and other instruments, which are inserted through the nose into a sinus cavity. We have implemented this method in both hands-on cooperative operation and teleoperation control mode. We evaluate and compare user performance in these two operation modes. We show that cooperative operation is more intuitive for the user and easier to use. On the other hand, due to the robot stiffness of our current implementation, teleoperation mode shows more accurate. Both of these two robot-assisted modes significantly improve human's performance compared to the totally freehand motion.**

Index Terms* - *geometric constrain, optimization robot control, performance analysis, virtual fixture

I. INTRODUCTION

Our recent work in developing a robotic surgical assistant system led us to the direction on generating a spatial motion constraints associated with complex 3D geometry for controlling a robot working in a very complicated configuration. An application of specific interest is sinus surgery, in which the endoscope and other instruments are inserted through the nose into a sinus cavity. The complicated anatomic structure constrains the motion of these instruments. During surgery, the instruments or the camera should avoid collisions or excessive force on delicate anatomy while still performing desired motion to accomplish the intended task.

In previous papers [1, 2], we described constrained robot control algorithms and a method to generate virtual fixtures from complex 3D anatomy for a Human Machine Collaborative System (HMCS) to help the user to manipulate surgical tools in a complicated working volume. We reported experimental results for the JHU "Steady Hand" microsurgical robot registered to preoperative skull models derived from CT images. In steady hand cooperative control, both the surgeon and robot hold the surgical tool and the robot moves the tool in response to forces exerted on the tool by the surgeon.

In this paper, we extend our previous work using both steady hand cooperative control operation and more traditional master-slave teleoperation. We present performance evaluation for a simple surgical task using our virtual fixtures with both teleoperation and hands-on cooperative control operation. The simple task is to manipulate an instrument to follow a path inside a cavity. Fig. 1 conceptually illustrates the relationship between the instrument, 3D path and approach aperture to the workspace cavity in our sample task.

Constrained robot control and virtual fixtures have been discussed previously in both telemanipulation and cooperative manipulation contexts. Virtual fixtures [3-5] are computer-generated constraints, which help a robotic manipulator perform a task by limiting its movement into restricted regions and/or influencing its movement along desired paths. The recent work in our center on virtual fixtures [6-10] focused on 2D geometric guidance motion of the tool tip or camera and assumed that the tool or camera itself did not have any other environmental constraints. Funda, Taylor, *et al*. [11] formulated desired motions as sets of task goals in any number of coordinate frames relevant to the task, optionally subject to additional linear constraints in each of the task frames for redundant and deficient robots.

Whether teleoperation or hands-on cooperative control is used, the goal of this work is to provide selective assistance to the human surgeon, while allowing the surgeon to retain ultimate control of the procedure. Each mode has some advantages and some limitations, although both are readily adapted to the same underlying analytical framework. Our purposes in this paper are to demonstrate the ease with which either mode can be incorporated into our formulation and to provide some initial quantitative experimental results demonstrating performance trade-offs. First we briefly review our constrained control algorithm and virtual fixture generation system [1]. We then describe the experiments and results. Finally, we present conclusions and future work.

II. CONSTRAINED ROBOT CONTROL

A. Constrained control algorithm

Our goal is to place absolute bounds on the motion of the instrument in the constrained working environment. Within these bounds, the controller should try to place the instrument tip as close to the desired position as possible. Our assumption is that the robot is holding the surgical instrument, and a model of the patient's anatomy have been obtained and registered to the coordinate frame of the robot. The basic control loop may be summarized as follows:

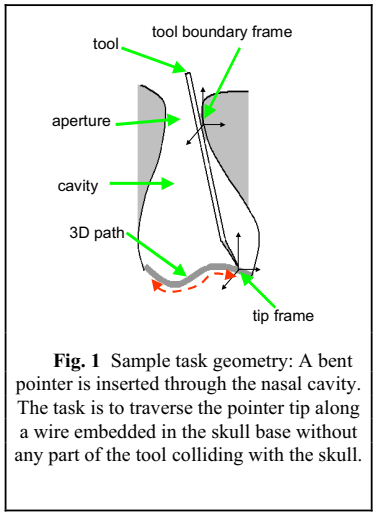

Fig. 1 Sample task geometry: A bent pointer is inserted through the nasal cavity. The task is to traverse the pointer tip along a wire embedded in the skull base without any part of the tool colliding with the skull.

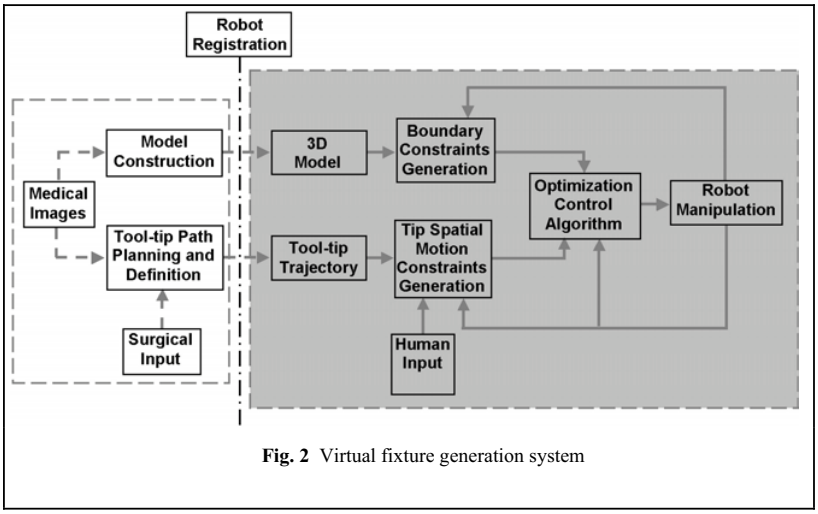

Fig. 2 Virtual fixture generation system

Step 1: Describe a desired incremental motion of the surgical instrument, based upon user inputs, such as obtained from a joystick or hands-on cooperative force control. This description may include both an objective function describing desired outcomes (e.g., move as close as possible to a target) and motion constraints (e.g., avoid collisions, avoid exceeding robot joint limits, prevent position errors to exceed specified limits, restrict tip motion to remain within a desired envelope, etc.). The desired incremental motion is described as the solution to the constrained optimization problem.

Step 2: Use the robot and task kinematic equations to produce a new quadratic optimization problem, in which instrument motion variables and other task variables have been projected onto incremental joint variables. This problem has the general form:

$$\begin{aligned} &\arg\min_{\Delta q} \left\| W \cdot \Delta x - f \right\| \\ &st. \quad H \cdot \Delta x \geq h \\ &\quad \Delta x = J \cdot \Delta q \end{aligned} \tag{1}$$

where Δq is the desired incremental motion of the joint variables and Δx is an arbitrary vector of task variables. W is a diagonal matrix for weights and f is the desired incremental motion. Different components of the optimization function may be assigned different relative weights, so that the errors of critical motion elements are close to zero, while errors in other non-critical motions simply stay as low as possible within tolerances allowed by the constraint set.

Step 3: Use known numerical methods [12] to compute incremental joint motions Δq, and use these results to move the robot.

Step 4: Go back to Step 1.

B. Constraints generation

Fig. 2 shows the virtual fixture generation system for our task. Our constrained robot motion control is based on two important components: tool-tip spatial motion constraints generation and the tool shaft boundary constraints generation.

Tool shaft boundary constraints: In order to avoid the collision of the tool shaft and the anatomy, we need address the geometry relation between the anatomy and the tool. Boundary constraints are generated from closest-point pairs P_b and P_k on the geometry boundary and the tool respectively. We use 3D triangulated surface models of anatomy to develop real-time constraints on tool motion. The models are geometrically complex; generating constraints in real time can be a challenge. In the current work, we model the surgical tool as one or more cylinders representing the tip and the tool shaft. We use a covariance tree data structure [13] to search for the closest point on the surface to the tool. A covariance tree is a variant of a k-dimensional binary tree (k-D tree). In our covariance tree, each sub-space is defined in the orthogonal coordinate system of the eigenvectors centered at the center of mass of the point set, and is recursively partitioned along this local coordinate frame. We rotate each local reference frame so that the *x-axis* is parallel to the eigenvector with the largest eigenvalue and the *z-axis* is parallel to the eigenvector with the smallest eigenvalue. An important advantage of covariance trees is that the bounding boxes tend to be much tighter than those found in conventional k-D trees and tend to align with surfaces, thus producing a more efficient search. Details of tree building and searching are discussed in [1].

Tool-tip spatial motion constraints: This component is responsible for relating the defined task with the user's input. We model the tool tip as a Cartesian "robot" whose position $x(t) \in SE(3)$ and velocity $v(t) \in R^3$ are related to the robot's joint positions $q(t)$ and the relations

$$x(t) = Kins(q(t)); \; v(t) = J(q(t))\dot{q}(t); \; J_{ij}(t) = \frac{\partial Kins(q)_i}{\partial q_j} \tag{2}$$

In the hands-on cooperative operation approach, the tip's motion is controlled by applying forces and torques on the handle of the instrument. The forces and torques are transformed to produce forces $f \in R^3$ resolved in coordinate frame of the tool tip. In the teleoperation approach, the readout of the joystick, which is aligned with tool tip coordinate frame, controls the tip motion. Given a

reference direction of motion $D = D(t)$, following [10], we define the preferred direction as

$$D_c(x) = (1 - k_d)[D]f + k_d \|f\| \langle D \rangle u \quad (3)$$

where $[D]$ and $\langle D \rangle$ are the span and kernel operators of D respectively; u is a signed distance from the tool tip to the task motion target; k_d is a blending coefficient with $0 \le k_d \le 1$, which governs how quickly the tool is moved toward the reference direction. We then define an admittance control law

$$v_{tip-desired} = k([D_c] + k_\tau \langle D_c \rangle) \quad (4)$$

where $v_{tip-desired}$ is the desired tip velocity, k is the admittance gain and k_τ ($0 \le k_\tau \le 1$) is an admittance ratio that attenuates the non-preferred component of the force input. Our desired 3D Curve D is defined by B-spline curve. At each control loop, we search the closest point on the B-spline to the current tool-tip position and compute the tangent direction of the B-spline at that point. If t_x, t_y and t_z are the components of the tangent to the curve, and n_x, n_y and n_z are the components of the vector from current position to the closest point on the curve, then we have $D = (t_x \quad t_y \quad t_z)^T$ and $u = (n_x \quad n_y \quad n_z)^T$.

C. Control Algorithm Implementation

At each time step, our goal is to compute incremental joint motions Δq, which then are fed to low-level position servos. We compute the desired tool tip velocity and convert this to an incremental 6-DOF tool motion $\Delta P_{tip-des} = ((v_{tip-desired} \Delta t) \| (0,0,0))^T$, where Δt is the sample interval. We identify three classes of requirements: in the tip frame, tool boundary frame, and in joint space. For each, we define an objective function ζ to be minimized and a set of linearized constraints.

Tool tip motion: We require that an incremental tool tip motion be as close as possible to some desired value. We express this as:

$$\zeta_{tip} = \|\Delta P_{tip} - \Delta P_{tip-des}\|^2 ; \; \Delta P_{tip-des}{}^T \cdot \Delta P_{tip} \ge (1 - \varepsilon)\|\Delta P_{tip-des}\|^2 \quad (5)$$

where ε is a small positive number (0.01 in our experiments). We relate tip frame motion to joint motion via the Jacobean relationship $\Delta P_{tip} = J_{tip}(q)\Delta q$. We rewrite (5) as

$$\zeta_{tip} = \|W_{tip} \cdot (J_{tip}(q)\Delta q - \Delta P_{tip-des})\|^2 ; \; H_{tip-des} J_{tip}(q)\Delta q \ge h_{tip} \quad (6)$$

where $H_{tip-des} = \Delta P_{tip-des}{}^T$, $h_{tip} = (1 - \varepsilon)\|\Delta P_{tip-des}\|^2$ and $W_{tip} = diag\{w_{tip}\}$ denotes a diagonal matrix of weighting factors specifying the relative importance of each component of ΔP_{tip}. We set w_{tip} to a fairly high value (1 in the current experiments) to track the path tightly.

Boundary constraints: We want to ensure that the instrument, which is inserted into a cavity, will not collide with the cavity boundary as a result of the motion. For each potential contact point pair we get a constraint of the general form

$$n_b{}^T \cdot (P_k + \Delta P_k - P_b) \ge \varepsilon \quad (7)$$

where P_b and P_k are the position of the potential collision point on the surface and tool, respectively. They can be generated by method described in section B. n_b is the normal of the contact point on the surface, and ε is a small positive number (0.001 in our experiments). The constraint described in (7) indicates that the angle between $(P_k + \Delta P_k - P_b)$ and $n_b{}^T$ is less than 90°. We can also define an objective function $\zeta_k = \|W_k \Delta P_k\|$ expressing the desirability of minimizing extraneous motion of the tool near the boundary, and can again rewrite these formulae in terms of Δq:

$$\zeta_k = \|W_k \Delta P_k\|, \; H_k J_k(q)\Delta q \ge h_k \quad (8)$$

We use very low values (0.01) for w_k and rely mainly on the inequality constraints. An alternative would have been to leave the ζ_k term out of the optimization altogether. The number of boundary constraints is dynamically changed. It depends on how many closest-point pairs we generate based on the relative position of the tool and the geometry constraint.

Joint limits: Finally, we want to ensure that none of the joint limits are exceeded as a result of the motion. This requirement can be stated as, $q_{\min} - q \le \Delta q \le q_{\max} - q$, where q is the vector of the current values of the joint variables, and $q_{\min}$ and $q_{\max}$ denote the vectors of the lower and the upper bounds on the joint variables, respectively. We also want to minimize the total motion of the joints. This can be rewritten in the form

$$\zeta_{joints} = \|W_{joints} \Delta q\|^2 ; \; H_{joints} \Delta q \ge h_{joints} \quad (9)$$

where $H_{joints} = \begin{bmatrix} I \\ -I \end{bmatrix}$, $h_{joints} = \begin{bmatrix} q_{\min} - q \\ -(q_{\max} - q) \end{bmatrix}$. Again, we set w_{joints} to 0.01 and simply enforce the inequality constraints.

Putting it together: We combine all the task constraints and objective functions, and then obtain the overall optimization problem, which is:

$$\zeta = \arg\min_{\Delta q} \left\| \begin{bmatrix} W_{tip} & & \\ & W_k & \\ & & W_{joints} \end{bmatrix} \cdot \left(\begin{bmatrix} J_{tip}(q) \\ J_k(q) \\ I \end{bmatrix} \Delta q - \begin{bmatrix} \Delta P_{tip-des} \\ 0 \\ 0 \end{bmatrix} \right) \right\|^2 \quad (10)$$

$$\text{subject to} \begin{bmatrix} H_{tip} & & \\ & H_k & \\ & & H_{joints} \end{bmatrix} \cdot \begin{bmatrix} J_{tip}(q) \\ J_k(q) \\ I \end{bmatrix} \Delta q \ge \begin{bmatrix} h_{tip} \\ h_k \\ h_{joints} \end{bmatrix} \quad (11)$$

which can be solved numerically using NNLS method in [12] for the set of joint displacements Δq, satisfying the constraint (11) and minimizing the error norm of (10).

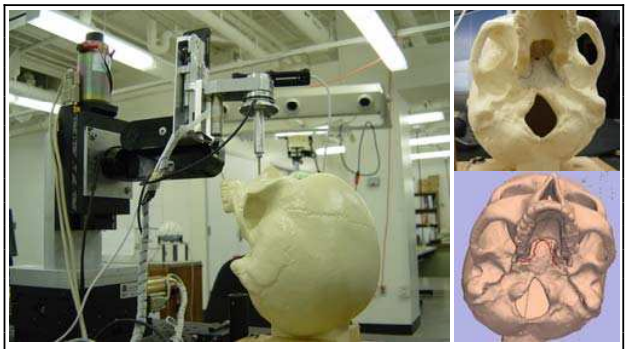

Fig. 3 (left) Experimental Setups, (right up) phantom skull and wire serving as path, (right down) reconstructed 3D skull model and path

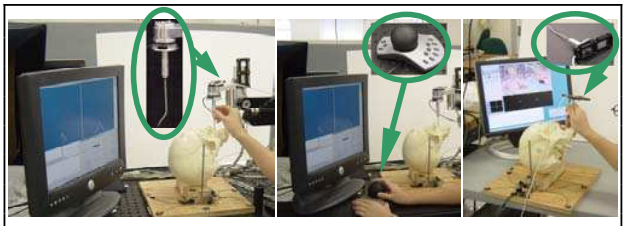

Fig. 4 Three experiment control modes: (left) hands-on, (middle) remote, and (right) freehand

III. Experiments

The experimental setup is shown in Fig. 3. Our experimental system is based on the JHU Steady-Hand Robot [14]. This robot has a 7-DOF remote-center-of-motion (RCM) kinematic structure, 10μm position resolution, and (relatively) high stiffness. The robot was designed for cooperative control of microsurgical tasks requiring low tremor and high precision. A force sensor is integrated at the end-effector, and the operator holds a tool mounted at the end-effector. The robot motion responds to the applied force, allowing the operator to have direct control over robot motion. For the experiments with teleoperation, we added a simple teleoperator interface in which desired tool motions are commanded through a SpaceBall™ mouse (3D Connexion, Germany).

A thin wire attached inside the nasal cavity of a plastic skull serves as the target path. The target path with respect to CT space is defined by tracing the wire with the tip of an Optotrak pointer. A 5-th degree b-spline curve, which interpolates the gathered sample points, represents the target path. Five small spherical fiducials were attached to the skull, which was then CT scanned. A surface model of the skull was created from the CT images and the positions of the fiducials in CT coordinates were determined via standard methods. We computed the transformation from the robot and skull to the pre-operative CT coordinate frame by a standard fiducial-based registration method. The control system operated at 100Hz. We chose the admittance ratio k_τ in (4) as 0 to enforce the tool-tip motion only along the preferred direction. The control gain k_d in (3) as 0.2.

Ten graduate students and faculties in our laboratory participated in the experiment. These subjects were varied in experience with Steady-Hand Robot from novice to expert. All of the subjects were asked to manipulate a surgical tool with an OPTOTRAK® rigid body affixed through the plastic nasal cavity, follow a certain path (a wire attached to the bottom of nasal cavity serve as the path) as close as possible while avoiding the collision between the tool-shaft and the skull. A 3D visualization interface provided the user with visual information about the tool position, orientation, and the reference target path. The subjects were asked to perform five trials for each of three different modes below. (Pictures are shown in Fig. 4)

1. Freehand mode: the users held the tool and manipulated it without any assistance. No robot or other mechanical constraint was involved.
2. Steady-Hand robot guidance in hands-on cooperative operation mode: The users cooperatively manipulated the tool attached to the robot directly. The robot provided assistance to avoid collisions and aid to follow the path along the wire.
3. Steady-Hand robot guidance in teleoperation mode: The users controlled the tool motion through the SpaceBall™. The users only controlled the motion of the tool tip. The robot provided assistance to avoid collisions and aid to follow the path along the wire.

We recorded the tip position error of each trial during the path following task both from the robot encoders and OPTOTRAK® tracking system. The tip position error is the distance from tool tip position to the reference curve in a coordinate frame. The execution times for each trial were recorded on the computer. The average execution time and average error for all ten subjects were tested to determine the statistical difference between different modes. The average error is defined as the total error divided by the number of samples throughout the task. We compared the performance of different control modes in both robot context and OPTOTRAK context. In robot context, we evaluated subjects' performance based on the amount of the error inside robot envelope. All constraints were transformed into the robot coordinate frame, and the tool tip position was measured using the robot encoders and its kinematics. In this case, we assume there is no other external error: error is only caused by the control algorithms and the robot controller. In OPTOTRAK® context, the tracking system provided an independent measurement of tool tip position relative to the target path. In this case, we measure the error on the system level.

IV. Results

For all trials of ten subjects in robot-assisted modes (hands-on cooperative and teleoperative mode), during the path following task execution, we found the tool shaft itself did not hit the skull phantom by observation.

A. Robot Context

The error profiles during task execution of the two modes in robot context are shown in Fig. 5. Although the error in hands-on cooperative mode (0.204mm) is slightly better than the error performance in teleoperative mode (0.219mm), there is no significant difference between two modes (paired t-test, p=0.31) as shown in TABLE I.

TABLE I ERROR AND TIME IN HANDS-ON COOPERATIVE MODE AND TELEOPERATIVE MODE IN ROBOT CONTEXT

	Avg Error (mm)	p-value	Time (s)	p-value
Hands-on	0.204 ± 0.01	0.3065	19.00 ± 2.31	0
Teleoperative	0.219 ± 0.02		24.17 ± 4.14	

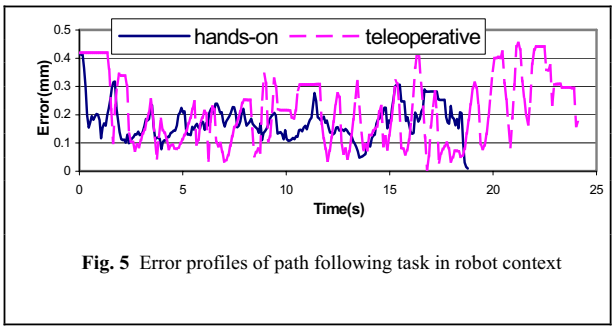

Fig. 5 Error profiles of path following task in robot context

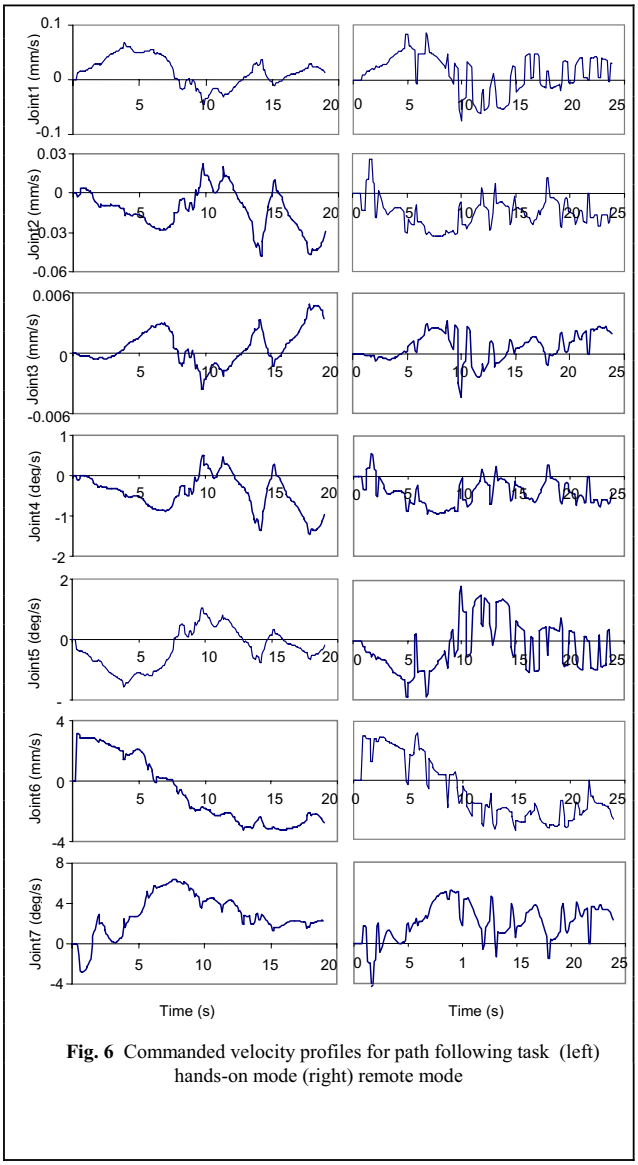

Fig. 6 Commanded velocity profiles for path following task (left) hands-on mode (right) remote mode

However the execution time in hands-on cooperative mode (19.00s) is significantly better than in teleoperative mode (24.17s). All subjects moved faster in hands-on cooperative mode.

TABLE II ERROR AND TIME IN THREE DIFFERENT MODE IN OPTOTRAK CONTEXT

	Avg Error (mm)		Time (s)	
	mean	st. dev.	mean	st. dev.
Hands-on	0.993	0.136	19.00	2.31
Teleoperative	0.720	0.112	24.17	4.14
Free hand	2.468	0.981	27.56	8.82

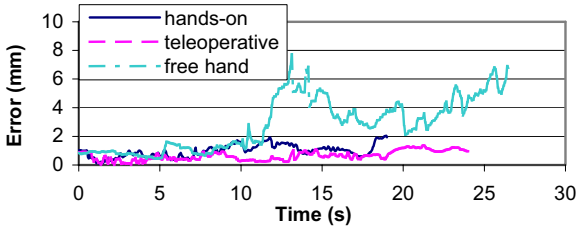

Fig. 7 Error profiles of path following task in OPTOTRAK® tracking system

In both modes, our control optimization solved practically identical problems to determine commanded joint velocities – i.e., the only differences in the constraints and objective functions were those relating to the user command interface (i.e., force compliance *vs.* SpaceBall input). Velocity profiles in Fig. 6 show that commanded joint velocities change more smoothly in hands-on cooperative mode than in our simple teleoperative mode. The users' SpaceBall input that is used to calculate the joint velocities is not as smooth as the force input. The teleoperative control via SpaceBall is harder than hands-on cooperative control. The subjects also commented that it is easier to perform in hands-on cooperative mode. This can provide some explanation on why execution time of the hands-on cooperative mode is less.

B. OPTOTRAK Context

We compared the errors in hands-on cooperative mode, the teleoperative mode, as well as the free hand mode in the OPTOTRAK® tracking coordinate system. As shown in TABLE II, the errors in both robot-assisted modes (0.99mm in hands-on cooperative mode, 0.72mm in teleoperative mode) are significantly better than in free hand mode (2.47mm). Similarly, the execution times in robot-assisted modes (17.00s in hands-on cooperative mode, 24.17s in teleoperative mode) are better than free hand mode (27.56s). The error profiles of the three modes during task execution are shown in Fig. 7. Because the relative position of the second part of the path with respective to the cavity is complicated, with the freehand mode, the subjects needed to tilt and rotate the tool simultaneously to keep the tool tip on the path. This complicated freehand tool control made the position error much bigger in the second half of the task.

As might be expected, the error of both robot-assisted modes in OPTOTRAK® context is much bigger than in robot context. In the former case, besides the control algorithm and robot controller, the robot calibration and system registration errors are other two main sources of the tip motion error. In our experiment, the residual registration

error measured across the five fiducials was 0.473mm. Moreover, although it is small, the accuracy of the OPTOTRAK® tracking system (<0.2mm in space) also contributes to the error.

Although the hands-on cooperative mode shows less error than teleoperative mode when the robot is used to measure its own performance, the relative accuracy of the teleoperative mode is better when an independent means (the OPTOTRAK®) is used to measure path tracking (paired t-test, $p<0.000001$). We believe that the main reason for this is the robot stiffness. In hands-on cooperative mode, subjects held the tool that is mounted on the robot end-effector. The hand forces used to command robot motion themselves produced some robot deflection. This factor was perhaps exacerbated by a tendency of users to push rather harder than was necessary to cause the desired motions. In teleoperation mode, of course, the users exerted no forces on the tool.

V. Conclusion

Robot-guidance (both hands-on cooperative and remote teleoperative control mode) employed spatial motion constraints generated by virtual fixtures derived from complex geometry can assist users in skilled manipulation tasks, while maintaining desirable properties such as collision avoidance and safety. The results showed significant improvement in both accuracy and execution time, compared to free-hand instrument manipulation.

The identical constrained robot control method can be used for both hands-on cooperative control and teleoperative control. We compared the performance of hands-on cooperative operation and teleoperation to control a tool manipulated in a complicated working volume. The experiment results show that hands-on cooperative operation is more intuitive for people to use. The execution time with hands-on cooperative operation is shorter than that with teleoperation. Without considering external errors, the performance of cooperative mode shows high accuracy than teleoperative mode. On the system level, although the registration error contributes same error in both control modes, teleoperation mode shows more performance accuracy due to the robot stiffness. The contact between the users and the tools that is mounted on the robot end-effector introduces perturbations into the system.

The experiments reported here are not designed as definitive experiments comparing these two paradigms in general. In particular, we simply used an available 3D joystick as the teleoperation master hand controller. No attempt was made to produce an optimized ergonomic design. Different specific designs could significantly improve the overall performance of either mode.

Nevertheless, the experiments show that it is possible to apply our control formulation to either paradigm and to achieve good performance while doing so. We are currently considering how to integrate these components into a full frontal sinus surgery system. Meanwhile we are evaluating other robot registration methods to improve the registration accuracy.

Acknowledgment

Partial funding of this research was provided of National Science Foundation under grants EEC9731748 (CISST ERC), IIS9801684, IIS0099770, and IIS0205318. The authors also gratefully acknowledge the advice and collaboration of Dr. Masaru Ishii, M.D., Dr. Allison Okamura, Dr. Gregory Hager, Ankur Kapoor and all the subjects for these experiments.

References

[1] M. Li, and R. H. Taylor, "Spatial Motion Constraints in Medical Robot Using Virtual Fixtures Generated by Anatomy," *Proc. IEEE International Conference on Robotics and Automation*, pp. 1270-1275, 2004.

[2] M. Li, and R. H. Taylor, "Optimum Robot Control for 3D Virtual Fixture in Constrained ENT Surgery," *Proc. Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pp. 165-172, 2003.

[3] L. B. Rosenberg, "Virtual fixtures: perceptual tools for telerobotic manipulation," *IEEE Virtual Reality International Symposium*, pp. 76-82, 1993.

[4] F. Lai, and R. D. Howe, "Evaluating control modes for constrained robotic surgery," *Proc. IEEE International Conference on Robotics and Automation*, 2000.

[5] Z. Stanisic, S. Payandeh, E. Jackson, "Virtual Fixture as an aid for teleoperation," *9th Canadian Aero*, 1996.

[6] R. Kumar, T. M. Goradia, A. Barnes, P. Jensen, L. M. Auer, L. L. Whitcomb, D. Stoianovici, R. H. Taylor, "Performance of robotic augmentation in microsurgery-scale motions," *Proc. Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pp. 1108-1115, 1999.

[7] R. Kumar, G. D. Hager, A. Barnes, P. Jensen, R. H. Taylor, "An augmentation system for fine manipulation," *Proc. Medical Image Computing and Computer-Assisted Interventions (MICCAI)*, pp. 956-965, 2000.

[8] A. Bettini, S. Lang, A. Okamura, G. Hager, "Vision assisted control for manipulation using virtual fixtures," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1171-1176, 2001.

[9] A. Bettini, S. Lang, A. M. Okamura, and G. D. Hager, "Vision assisted control for manipulation using virtual fixtures: Experiments at macro and micro scales," *Proc. IEEE International Conference on Robotics and Automation*, pp. 3354-3361, 2002.

[10] P. Marayong, M. Li, A. M. Allison, G. D. Hager, "Spatial motion constraints: theory and demonstrations for robot guidance using virtual fixtures," *Proc. IEEE International Conference on Robotics and Automation*, pp. 1954-1959, 2003.

[11] J. Funda, R. H. Taylor, B. Eldridge, S. Gomory, K. G. Gruben, "Constrained Cartesian motion control for teleoperated surgical robots," *IEEE Transactions on Robotics and Automation*, vol. 1996, pp. 453-465, 1996.

[12] C. Lawson, and R. Hanson, "Solving Least Squares Problems," *Englewood Cliffs, NJ: Prentice-Hall*, 1974.

[13] J. Williams, R. Taylor, L. Wolff, "Augmented k-d techniques for accelerated registration and distance measurement of surfaces," *Computer Aided Surgery: Computer-Integrated Surgery of the Head and Spine*, pp. 01-21, 1997.

[14] R. H. Taylor, J. Patrick, L.L. Whitcomb, A. Barnes, R. Kumar, D. Stoianovici, P. Gupta, Z. Wang, E. deJuan, L. Kavoussi, "A Steady-Hand robotic system for microsurgical augmentation," *International Journal of Robotics Research*, vol. 18(12), pp. 1201-1210, 1999.