Spatial Motion Constraints Using Virtual Fixtures Generated by Anatomy

Ming Li, *Member, IEEE*, Masaru Ishii, MD, and Russell H. Taylor, *Fellow, IEEE*

***Abstract*— This paper describes a spatial motion constraints generation approach for human-machine collaborative surgical assistant system from registered CT models. We extend constrained optimization formulation incorporating task goals, anatomy-based constraints, "no fly zones", etc. We use a fast potential collision constraint detection method based on 3D surface model and covariance tree data structure. These boundary constraints, along with task behaviors and joint limits, serve as constraint conditions for constrained robot control. We are able to follow a complex path inside a human skull phantom represented by a surface model composed of 99,000 vertices and 182,000 triangles in real time. Our approach enables real-time task-based control of surgical robot in a precise interactive minimally invasive surgery task.**

We illustrate our approach based on two example tasks which are analogous to the procedures in endoscopic sinus surgery and analyze the users performance on both teleoperation and cooperative control for one of the example tasks. The experimental results show that a robotic assistant employing our approach on spatial motion constraints can assist user in skilled manipulation tasks, while maintaining desired properties. Our approach is equally applicable to teleoperative and cooperative controlled robots.

***Index Terms*— virtual fixtures, anatomy-based constraint, optimization robot control, surgical robot assistant**

I. Introduction

MINIMALLY invasive surgery, compared to open surgery, offers many benefits to patients. However minimally invasive surgery presents a constrained working environment for both surgeons and mechanical devices designed to assist them.

In a minimally invasive approach, such as for throat surgery or endoscopic sinus surgery, long thin instruments or endoscope are inserted through anatomic opening reaching the pathological area. The operating volume is very limited. The instruments or endoscope have some degrees of translational and rotational freedom but their motions are constrained by anatomic structure.

Although the primary focus of this paper is development of techniques for controlling the motion of teleoperated and cooperatively controlled surgical robots in the presence of complex geometric constraints associated with patient anatomy, we have chosen endoscopic sinus surgery as a focusing example. Figure 2 conceptually illustrates the relationship between the instrument, target (such as 3D path, inspection target) and approach aperture to the workspace cavity in endoscopic sinus surgery. During such a surgical procedure, the inserted tools should avoid collisions or excessive force on delicate anatomy while still performing desired motion to accomplish an intended task.

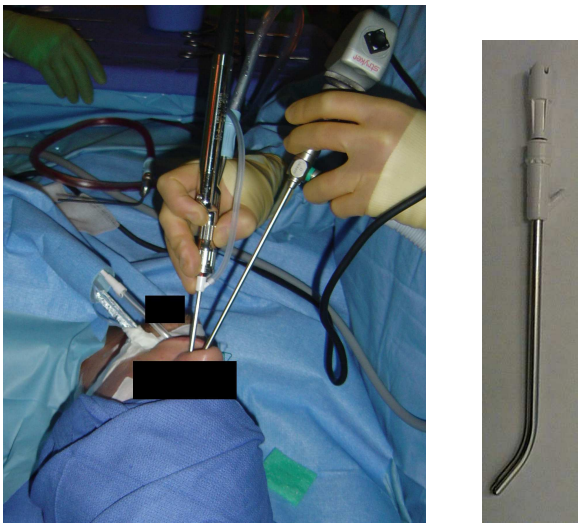

Fig. 1. An operating room scene of endoscopic sinus surgery and a typical shaver with a bent tip portion for endoscopic sinus surgery.

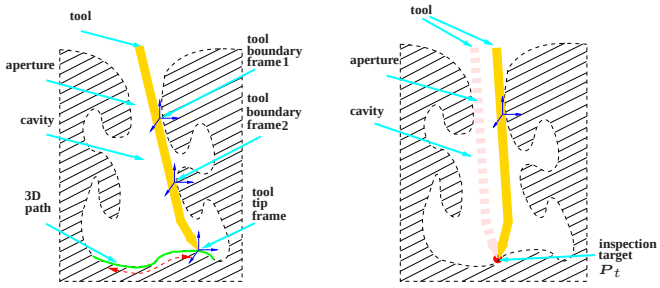

Fig. 2. Geometric relation for spatial motion constraints. The number of tool shaft boundary constraints changes based on the relative position between the tool and the anatomy. (left) In the path-following task, the tool tip is required to move along the given 3d path. (right) In the target-inspection task, the tool tip is kept staying on the target, the orientation of the tool shaft is changed to access the target from different angles.

The difficulty of precise surgical manipulation with endoscopic instruments invokes the application of robotic assistants. In robotic assisted procedures, the surgeons' ability can be augmented by techniques such as virtual fixtures (VF). Virtual fixtures are algorithms generating motion constraints for a robotic manipulator perform a task by limiting its movement into restricted regions [1], [2], [3], [4], [5] and/or influencing its movement along desired paths [6], [7], [8]. Virtual fixtures

have been discussed previously in the literature for both telerobots [9], [10], [6], [11] and cooperative robots [7], [12], [13], [14], [15].

An important case of virtual fixtures for surgical application is forbidden regions, where the surgical tool is restricted to certain regions in the workspace. Davies *et al.* [16] set active constraints to constrain the robot to cut the femur and tibia within a permitted region for prosthetic knee surgery. Park *et al.* [17] developed sensor-mediated virtual fixtures that constrain the robot's motion or create haptic feedback directing the surgeon to move the surgical instruments in a desired direction. Their work applied a virtual wall based on the location of the internal mammary artery obtained from a preoperative CT scan to guide a surgeon's instrument during teleoperated coronary bypass.

Bettini *et al.* [7] focused on the techniques for guidance virtual fixtures. They used vision information to assist generation of virtual fixtures. They examined the hard and soft virtual fixtures and discussed the application to vitreoretinal surgery. Marayong *et al.* [18] demonstrated motion constraints with varying compliance that were described for the general spatial case. These work used admittance control laws to implement virtual fixtures.

Troccaz *et al.* [13], [15], [19] developed a passive arm with dynamic constraints (PADyc) for pericardial puncture in cardiac surgery. Their system implements virtual fixtures to constrain surgical tools along desired paths or away from some regions by using electrical motors to choose a clutching system for freewheels. The principal advantage of such an approach is that the robot is entirely dependent on the surgeon to provide motive force, which some authors argue may have some safety advantages if one is concerned about the robot "running away". Limitations include mechanical complexity and loss of the robot's ability to actively assist in surgical procedures.

In endoscopic sinus surgery, Wurm *et al.* [20] developed a robotic system for fully automated paranasal sinus surgery. This system uses pre-operative CT to direct the robot's autonomous motion. It allows the remote control by joystick as well. Lueth's group [21], [22], [23] presented a mechatronic system for functional endoscopic sinus surgery. Preoperatively they planned a safe working space based on a 3D model from CT data. Intra-operatively, the shaver is automatically turned on/off depending on the position of the shaver tip. Within the safe area the shaver reacts to signals from the surgeon. If the tip of the shaver moves outside the predefined working space, the shaver's automatic drive control is interrupted by an electrical pulse. This navigation based system has only concerned the position of the shaver tip itself.

Path planning and motion control is a well discussed area with a wide variety of proposed optimality criteria [24], [25], [26], [27]. Some work has used pseudo-inverse based gradient projection techniques [28], [29], [30] or extended Jacobian techniques [31], [32] to implement constrained control of kinematically redundant robots. In [33], [34], the authors used neural networks for constrained robot motion control. Funda *et al.* [35] presented an optimal motion control method to control both redundant and deficient robotic systems in constrained working volumes.

Our purpose is to derive spatial motion constraints from a very complex environment and enable the robotic system to provide assistance for dexterous procedures in that complex geometric environment in real time. There has been prior work on model-based off-line planning for robotically assisted surgery. For example, Adhami and Coste-Maniere *et al.* [36], [37] developed graphical simulation system and algorithmic approaches to port placement for endoscopic coronary artery bypass using the daVinci System. Similarly, Cannon *et al.* [38] have presented a computer-based algorithm using preoperative images to provide the surgeon with a list of feasible port triplet for coronary artery bypass grafting. Latombe's group [39], [40] used randomized techniques to guess a promising initial set of beams for robotic radiosurgery system. Sim *et al.* [41] proposed a collision free pre-planned path for the instruments in neurosurgery.

In this paper, we present an on-line collision avoidance method for real time interactive control of a surgical robot in geometrically complex environments such as the sinus cavities. We extend Funda's work [35] to generate virtual fixtures for real-time obstacle avoidance and simultaneously assist the surgeon to perform desired tool motion to accomplish intended tasks. In our work, 3D anatomic constraints are automatically generated from 3D medical images in real time. Our new potential collision searching method based on a covariance tree derived from a patient-specific anatomic model enables real-time pertinent anatomic constraints generation. We are able to follow a complex path inside a human skull phantom represented by a surface model composed of 99,000 vertices and 182,000 triangles in real time.

The remainder of the paper is organized as following. Section II provides a brief summary of our motivative clinical example. In section III, we first describe the virtual fixture generation system and the constrained control algorithm we employed for generating virtual fixtures. Thereafter we explain how to detect spatial motion constraints from complex 3D anatomy in real time. In section IV, we analyze and detail the implementation of two application tasks: path following and endoscopic inspection of a target from different orientations. In section V, we report the experiment results of two application tasks. We present performance comparisons on our path-following task using our virtual fixtures with both teleoperation and cooperative control as well as for unassisted freehand manipulation. In section VI, we conclude our work and discuss possible future extension.

II. Motivating clinical application:endoscopic sinus surgery

The paranasal sinuses represent a series of pneumatized chambers surrounding the nasal cavities. Sinusitis is inflammation of the paranasal sinuses from bacterial, fungal, viral,

allergic or autoimmune issues. Sinusitis is one of the most common health care problems in the United States affecting approximately 31 million Americans annually. [42] Chronic sinusitis, a subtype of sinusitis where signs and symptoms of inflammation persist for greater than twelve weeks, is known to significantly impact quality of life and is associated with substantial functional and emotional impairment [43]. Benninger *et al.* [44] estimates that chronic sinusitis results in 18 to 22 million physician office visits in the United States annually, and it is known that all forms of sinusitis result in significant health care expenditures.

Chronic sinusitis is typically treated using a combination of anti-inflammatory and antimicrobial agents. If these agents fail and there is radiographic evidence of disease on Computed Tomography imaging of the paranasal sinuses then surgery is warranted. Surgical therapy includes opening and enlarging blocked ostia, removing entirety of sinus air cells or marsupializing. Currently, a minimally invasive approach to the paranasal sinuses called functional endoscopic sinus surgery, FESS, is the standard of care for treating medically recalcitrant sinus disease. Here a series of instruments under endoscopic guidance are used to systematically remove diseased tissue and bone to enhance the drainage pathways of diseased sinuses.

Endoscopic sinus surgery is associated with severe and sometimes catastrophic complications. This is because the brain, eye, optic nerve, other cranial nerves, and the carotid artery are all within millimeters of the surgical field. If the surgeon becomes disoriented or confused, injury to one of these important structures is possible. A number of important technologies have been developed to minimize the chances of these injuries such as image guided surgery, intra-operative imaging, and steady hand motion stabilization; however, a strong understanding of surgical anatomy and a systematic and methodical surgical technique remain the cornerstone of safe surgical practice. The proximity of these structures to the nasal passage; however, allows for the potential of minimally invasive approaches to these extranasal structures if they are involved with disease. This has lead to a developing field known as endoscopic anterior skull based surgery. Here the confines of the nose and paranasal sinuses are violated using endoscopic sinus techniques to address disease beyond the boarders of the nose. These surgeries are technically challenging, even for experienced sinus surgeons, and often involve delicate repetitive motions such as using high speed drills to systematically remove the protective bony covering surrounding a nerve embedded deep within the skull. It is thought that steady hand motion stabilization, robotic assisted surgery and spatial motion constraint will play a major role in advancing this field and disseminating this approach to the general population. The accuracy of the constraining procedure obviously will depend on the surgical approach; however, for a prototypical procedure of a transnasal endoscopic approach to the sphenoid sinus, to address a pituitary tumor for example, or a maxillary antrostomy, to cure an acute maxillary sinusitis, the accuracy should be on the order of a millimeter. This is based on the observation that a maxillary sinus's ostia is about 2 millimeters in size [45].

III. Approach

A. Assumptions

In this paper we use endoscopic sinus surgery as a motivation to discuss our spatial motion constraints generation approach for robotic surgical assistant system. The anatomy of the nasal and sinus cavity is mostly bony tissue. Currently, we assume that the anatomy is rigid, and ignore the deformation and motion during the system registration and surgical intervention. However, the basic approach is extendable to non-rigid or moving anatomy, provided that suitable imaging and real-time modeling is available.

The control algorithm we discuss in this paper is a "high level", model-based method that generates incremental joint motion or velocity set points for a low-level controller which provides stability. From the standpoint of the low-level control, these commanded motions have similar characteristics to those that would be found in a conventional teleoperated or cooperatively controlled robot, and stability has not been an issue. Our control approach readily accommodates the addition of constraints to ensure feasible, stable motions; examples would include limits on joint command, positions, velocities and accelerations.

B. Virtual Fixture Generation System Diagram

The robotic surgical assistant is a surgical CAD/CAM system [46]. Pre-operatively, the patient with radio-opaque fiducials attached on the surface is scanned with Computed Tomography (CT). A 3D surface model around the surgical area is created from the pre-operative CT images. A surgeon defines the tool-tip target trajectory or target position of interest by selecting points in the pre-operative image slices. To create a smooth trajectory, we generate a polynomial (such as a bspline) by interpolating user selected sample points.

Intra-operatively, the system registration procedure is applied to correlate the robot, patient and the pre-operative medical images. The registration procedure could be accomplished with fiducial marker based method or more sophisticated method [47], [48]. After the robot is calibrated and registered, information from pre-operative planning, which includes the planned tool-tip target path or the target position and the 3D model, is transformed into the robot coordinate frame.

In our robotic assistant system, the surgeon is in the control loop; s/he is able to control the progress of the tool. The system reads the surgeon's input, accompanied with the planned tool-tip trajectory and the current tool-tip position, generates the spatial motion constraints for the tool-tip. Meanwhile, tool-shaft boundary motion constraints are generated from the registered 3D geometry model and the current tool position. These constraints along with some other constraints, such as joints limitation of the robot, are fed into our constrained optimization control algorithm to obtain the desired joint

velocities to move the robot. Figure 3 shows the diagram of our spatial motion constraints generation system.

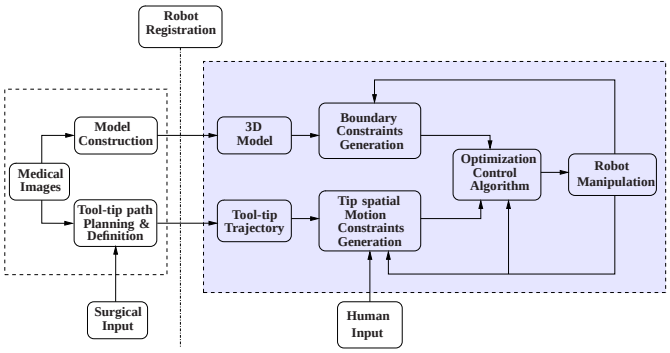

Fig. 3. Spatial motion constraints generation diagram

C. Constrained Optimization Control

In considering virtual fixtures for a surgical assistant robot, it is important to be able to place absolute bounds on the spatial motion of the different parts of the instrument as well as to specify desired nominal motions.

We define different task frames associated to different parts of the instrument. For each of the task frames, we define actual state variables $\boldsymbol{x}$ and desired state variables $\boldsymbol{x}^d$. $\boldsymbol{x} = \boldsymbol{x}(\boldsymbol{q}, \Delta\boldsymbol{q})$ is a function of joint variables $\boldsymbol{q}$ and joint incremental motion $\Delta\boldsymbol{q}$.

We formulate a constrained optimization problem to generate the constrained motion for a certain task frame. The general formulation for this problem is:

$$\begin{aligned} \Delta\boldsymbol{q}_{cmd} &= \arg\min_{\Delta\boldsymbol{q}} \; C(\boldsymbol{x}(\boldsymbol{q}, \Delta\boldsymbol{q}), \boldsymbol{x}^d) \\ s.t. \qquad & A(\boldsymbol{x}(\boldsymbol{q}, \Delta\boldsymbol{q})) \leq \boldsymbol{b} \end{aligned} \tag{1}$$

where $C(\boldsymbol{x}(\boldsymbol{q}, \Delta\boldsymbol{q}), \boldsymbol{x}^d)$ is the objective function associated with the variance between the actual state variables $\boldsymbol{x}$ and the desired state variables $\boldsymbol{x}^d$. $A(\boldsymbol{x}(\boldsymbol{q}, \Delta\boldsymbol{q})) \leq \boldsymbol{b}$ represents the constraint conditions. These constraints are used to force the solution vector $\Delta\boldsymbol{q}_{cmd}$ to satisfy certain critical requirements, such as restricting the motion of the certain part of the instrument within a strict motion envelope.

We combine the constrained motions on different task frames for generating complicated constrained motions. For an example, assume the virtual fixture for task frame $\{i\}$ is

$$\begin{aligned} \Delta\boldsymbol{q}_{cmd} &= \arg\min_{\Delta\boldsymbol{q}} \; C_i(\boldsymbol{x}_i(\boldsymbol{q}, \Delta\boldsymbol{q}), \boldsymbol{x}_i^d) \\ s.t. \qquad & A_i(\boldsymbol{x}_i(\boldsymbol{q}, \Delta\boldsymbol{q})) \leq \boldsymbol{b}_i \end{aligned} \tag{2}$$

Then the complicated virtual fixtures generated by constraining on task frames $\{i,\ (i = 1,\ ...,\ N)\}$ can be formulated as

$$\begin{aligned} \Delta\boldsymbol{q}_{cmd} &= \arg\min_{\Delta\boldsymbol{q}} \; \sum_{i=1}^{N} w_i C_i(\boldsymbol{x}_i(\boldsymbol{q}, \Delta\boldsymbol{q}), \boldsymbol{x}_i^d) \\ s.t. \qquad & A_i(\boldsymbol{x}_i(\boldsymbol{q}, \Delta\boldsymbol{q})) \leq \boldsymbol{b}_i, \\ & i = 1,\ ...,\ N \end{aligned} \tag{3}$$

where w_i giving the relative importance of minimizing the objective function error for different task frames. The combination of a weighted objective function and an additional set of task constraints allows us to exploit the geometry of a particular task space motion and effectively trade off the various performance criteria.

Surgical robots often are kinematically redundant for the purpose of providing dexterous assistance. At the same time, task constraints such as the requirement of a tool pass through a cavity restrict dexterity [35]. Indeed, some special-purpose designs for minimally invasive surgery, such as IBM-JHU LARS [49] and JHU Steady Hand robot [50], may be kinematically deficient. Other robots such as the daVinci [11] and Endorobotics [51] combine a kinematically constrained remote center of motion (RCM) mechanism with a kinematically redundant wrist. The ability to accommodate unique, special purpose mechanical designs (such as kinematically redundant or deficient) is important as well. Our formulation could easily integrate any behaviors, such as asserting joint limitation for kinematically redundant robot, incorporating haptic information, to the control strategy.

The general form of optimization problem (1) has many variants, both for the objective function and constraints. In this work we specialize (1) to produce a quadratic optimization problem with linear constraints [52]. We use linear constraints because of the efficiency and robustness of the computation. The objective function is a two-norm of incremental motion error in different task frames. Since the incremental motion is sufficiently small, $\Delta\boldsymbol{x} = J(\boldsymbol{q}) \cdot \Delta\boldsymbol{q}$ represent a good approximation to the relationship between $\Delta\boldsymbol{x}$ and $\Delta\boldsymbol{q}$. Form (1) is then rewritten as

$$\begin{aligned} \Delta\boldsymbol{q}_{cmd} &= \arg\min_{\Delta\boldsymbol{q}} \; \left\| J(\boldsymbol{q}) \cdot \Delta\boldsymbol{q} - \Delta\boldsymbol{x}^d \right\|_2^2 \\ s.t. \qquad & A \cdot J(\boldsymbol{q}) \Delta\boldsymbol{q} \leq \boldsymbol{b} \end{aligned} \tag{4}$$

$\Delta\boldsymbol{q}_{cmd}$ is used as set points for low level position/velocity control loop which guarantees stability.

Customized virtual fixtures for complicated surgical tasks can be treated as the combination of one or more of objects assigned on single or multiple task frames. The basic control loop is shown in Figure 4 and summarized as follows:

Step 0: We assume that the robot is holding the surgical instrument. The instrument and all the constraints are known in the robot coordinate frame.

Step 1: Describe a desired incremental motion of the surgical instrument $\Delta\boldsymbol{x}^d$ based upon (a) surgeon inputs, such as may be obtained from a joystick or hands-on cooperative force control; (b) an *a priori* surgical task description; (c) real-time sensor feedback, such as might be obtained from a vision sensor.

Step 2: Analyze and decompose the task into task primitives on different task frames. It 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

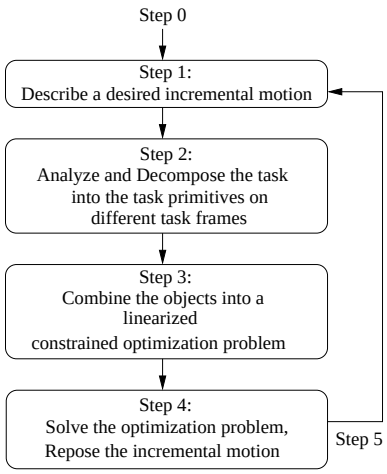

Fig. 4. Basic control loop

errors to exceed specified limits, restrict tip motion to remain within a desired envelope, etc.).

Step 3: Use the robot and the task kinematic equations to produce a new linearized optimization problem, in which the instrument motion variables and other task variables have been projected onto incremental joint variables. This problem has the general form:

$$\begin{aligned} \Delta \boldsymbol{q}_{cmd} = \arg\min_{\Delta \boldsymbol{q}} \sum_{i=1}^{N} w_i \left\| J_i(\boldsymbol{q}) \cdot \Delta \boldsymbol{q} - \Delta \boldsymbol{x}_i^d \right\|_2^2 \\ s.t. \qquad A_i \cdot J_i(\boldsymbol{q}) \Delta \boldsymbol{q} \leq \boldsymbol{b_i}, \\ i \ = \ 1, \ ..., \ N \end{aligned} \tag{5}$$

where J_i is the Jacobian matrix relating different task spaces i to the robot joint space.

Step 4: Use known numerical methods, such as LSI described in [53], to compute the set of incremental joint motions $\Delta \boldsymbol{q}$, and use these results to move the robot. Form (5) is a linear least squares problem, its solution reaches the global minimum.

Step 5: Go back to Step 1.

The basic loop can be terminated by predefined conditions or the signal from the operator.

D. Real-time Constraints Detection from Registered Model

The geometry of the instrument working area such as in endoscopic sinus surgery is complicated. Fast potential collision detection between the tool and anatomy is required for a real-time robot control. This potential collision is defined as the closest point pair $\boldsymbol{P}_b$ and $\boldsymbol{P}_k$ on the anatomical structure and the tool, respectively. This potential collision is used to generate instrument boundary constraints. In the current work, we model the surgical tool as one or more cylinders representing the tip and the tool shaft. We use 3D triangulated surface models of anatomy to develop real-time constraints on the tool motion. Productions of such anatomic models from 3D medical images and registration of the models to the robot workspaces are well-established technique [47]. However, the models are geometrically complex; generating constraints in real time can be a challenge.

1) Covariance K-D Tree - A Structure For Interference Detection: We use a covariance tree data structure [54] 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). The traditional k-D tree structure partitions space recursively along the principal coordinate axes. In our covariance tree, each subspace is defined in the orthogonal coordinate frame of the eigenvectors centered at the center of mass of the point set, and is recursively partitioned along these local coordinate axes. 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 the surfaces, thus producing a more efficient search.

The tree is built recursively as follows. Given a set of points $\{\boldsymbol{P}_i\}$ sampled from the surface, we find the centroid and the moments

$$\begin{aligned} \bar{\boldsymbol{P}} = \tfrac{1}{N} \sum_{i=1}^{N} \boldsymbol{P}_i; \quad \boldsymbol{P}'_i = \boldsymbol{P}_i - \bar{\boldsymbol{P}}; \\ M = \sum_{i=1}^{N} \boldsymbol{P}'_i \cdot \boldsymbol{P}'^T_i \end{aligned} \tag{6}$$

We then analyze the principal components of M and determine the rotation matrix $R = \begin{bmatrix} \boldsymbol{R}_x & \boldsymbol{R}_y & \boldsymbol{R}_z \end{bmatrix}$ with columns $\boldsymbol{R}_x$ and $\boldsymbol{R}_z$ corresponding to the largest and the smallest eigenvalues of M, respectively. Next, we compute the set of points $\{\boldsymbol{P}^*_i\}$ in local frame by rotating $\{\boldsymbol{P}_i\}$: $\boldsymbol{P}^*_i = R \cdot \boldsymbol{P}_i$. Finally, we compute the bounding box of $\{\boldsymbol{P}^*_i\}$ and partition $\{\boldsymbol{P}^*_i\}$ about the cutting plane $x = 0$. Each of these sets is thus recursively subdivided until only a small number of points remain in each box.

We fit a circumsphere about each triangle. We then construct a covariance tree of the centers of each circumsphere. The use of circumspheres about triangles is necessary to ensure the correctness of the algorithm, which is intended to find the closest point on a triangulated surface. It is possible for the vertices of a triangle to be long distances from a target point while there is a point on the edge or interior of the triangle to be quite close. Although it is theoretically possible for an extremely long triangle to degrade the efficiency of our data structure, as practical matter this does not occur with the image-based anatomical models which we use.

At each node (i.e. for each bounding box) of the tree, we note the maximum radius $r_{\max}$ of all circumspheres associated with that node. We compute the corners of an expanded bounding box

$$\begin{aligned} \boldsymbol{P}^*_{\min} = -r_{\max} + \min_i \boldsymbol{P}^*_i; \\ \boldsymbol{P}^*_{\max} = \ r_{\max} + \max_i \boldsymbol{P}^*_i \end{aligned} \tag{7}$$

where the $\min$ and the $\max$ operations are performed element-wise.

2) Potential Collision Detection Using K-D Tree: The tree searching for collision detection proceeds as follows. Given a line segment $\overline{\boldsymbol{P}_1\boldsymbol{P}_2}$ between two points $\boldsymbol{P}_1$ and $\boldsymbol{P}_2$, we wish to find all possible surface patches that may be closer than some threshold distance ϵ_{thresh} to the line segment. At each level of the tree, we first transform the line segment to the local coordinate frame

$$\begin{array}{l} \boldsymbol{P}_1^* = R^{-1} \cdot (\boldsymbol{P}_1 - \bar{\boldsymbol{P}}); \\ \boldsymbol{P}_2^* = R^{-1} \cdot (\boldsymbol{P}_2 - \bar{\boldsymbol{P}}) \end{array} \tag{8}$$

If we are at a terminal node of the tree, then for each triangle t in the node, we compute the point pair $\boldsymbol{P}_{b\_t}^*$ and $\boldsymbol{P}_{k\_t}^*$ corresponding to the closest approach between the line segment and triangle, where $\boldsymbol{P}_{b\_t}^*$ is on the triangle and $\boldsymbol{P}_{k\_t}^*$ is on the line segment. We retain any such pairs with $\|\boldsymbol{P}_{b\_t}^* - \boldsymbol{P}_{k\_t}^*\|_2 < \epsilon_{thresh}$.

If we are at a non-terminal node, we first check whether its bounding box is penetrated by the line segment or not. We recursively search the left and right sub-trees, if its bounding box is penetrated by the line segment. Otherwise, we compute the point of closest approach $\boldsymbol{P}_{b\_node}^*$ and the corresponding point $\boldsymbol{P}_{k\_node}^*$ on the line segment. If the distance between them is small, such as $\|\boldsymbol{P}_{b\_node}^* - \boldsymbol{P}_{k\_node}^*\|_2 < \epsilon_{thresh}$, we recursively search the left and right sub-trees for points of closest approach to $\overline{\boldsymbol{P}_1^*\boldsymbol{P}_2^*}$.

When the search is completed, we transform all point pairs back into the world coordinate frame. If there are no point pairs closer than ϵ_{thresh}, then we claim that there is no boundary constraints.

One difficulty with this approach is that it tends to produce many point pairs from the same patch of an anatomy that are almost equivalent, thus producing an excessive number of motion control constraints. Therefore, in practice we modify our search as follows: for any non-terminal node whose bounding box is not penetrated by the line segment, if 1) $\|\boldsymbol{P}_{b\_node}^* - \boldsymbol{P}_{k\_node}^*\|_2 < \epsilon_{thresh}$ and 2) the bounding box of the node is also very flat (i.e., with $\boldsymbol{P}_{\max}^*(z) - \boldsymbol{P}_{\min}^*(z)$ less than a specified value), we simply return the point pair $\boldsymbol{P}_{b\_node}^*$ and $\boldsymbol{P}_{k\_node}^*$ without further recursion.

IV. Application Task Analysis

We analyze two application tasks which are useful in endoscopic sinus intervention procedure. The "path-following" task is to guide a surgical tool following a pre-defined trajectory. The "target-inspection" task is to move a high-degree endoscope to observe a target from different angles. In both tasks, the tool works in a complicated geometric environment, collision avoidance between the tool shaft and anatomy is required.

Our constrained robot motion control bases on two important components: tool-tip motion constraint generation and the tool shaft boundary constraint generation. These generated constraints along with some other constraints, such as joints limitation of the robot, are fed into the constrained quadratic optimization algorithm to obtain the desired robot motion. The tool-tip motion constraints are task related. This component is responsible for relating the defined surgical task (such as, moving a surgical tool along a desired path or observe a target position on the patient's anatomy) with user inputs, which may be expressed either by hands-on cooperative control or through conventional teleoperation control.

A. Path-following in Complicated Geometric Environment

1) Tip Motion Constraints Definition: There are different ways to implement tip motion constraints [52], [8]. We use the method described in [18] to implement tip motion constraints and integrate it in our path following task. For the detail of this admittance law based method, the reader could refer to [18]. We abbreviate it below.

We model the tool tip as a Cartesian device with the position $\boldsymbol{x}_t \in SE(3)$ and velocity $\boldsymbol{v}_t = \dot{\boldsymbol{x}} \in \Re^3$ all expressed in the robot base frame. Our desired three-dimensional curve, which serves as the reference direction, is described as a bspline polynomial. At each control loop, the robot encoders read out the current tool tip position. We search the closest point on the bspline curve to the current tool tip position, and compute the tangent direction of the bspline curve on that point.

Assume the vector $\boldsymbol{t} = \begin{bmatrix} t_x & t_y & t_z \end{bmatrix}^T$ represents the tangent to the curve, and $\boldsymbol{n} = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T$ represents the vector from the current tool tip position to the closest point on the curve, we have the reference direction of motion D and the signed distance from the tool tip to the task motion target $\boldsymbol{u}$ as

$$D = \begin{bmatrix} \boldsymbol{t} & \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} = \begin{bmatrix} t_x & 0 & 0 \\ t_y & 0 & 0 \\ t_z & 0 & 0 \end{bmatrix},$$
$$\boldsymbol{u} = \boldsymbol{n} = \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}^T.$$

The user's input obtained from a force sensor or from a master robot is transformed to produces $\boldsymbol{f} \in \Re^3$ in the robot base frame. Following [18], we define two projection operation the span and the kernel of the column space as

$$\begin{array}{l} span\,(D) \equiv [D] = D\left(D^T D\right)^+ D^T; \\ kernel\,(D) \equiv \langle D \rangle = I - [D]\,. \end{array} \tag{9}$$

The definition and the properties of these two operators are described in [55]. We decompose $\boldsymbol{f}$ into two components: $[D]\,\boldsymbol{f}$ along the preferred direction and $\langle D \rangle\,\boldsymbol{f}$ perpendicular to the preferred direction. We apply an admittance ratio k_τ $(0 \leq k_\tau \leq 1)$ to attenuate the non-preferred component of the force input. Then we compute the desired tool tip velocity $\boldsymbol{v}_t^d$ by an admittance control law:

$$\boldsymbol{v}_t^d = k([D_c] + k_\tau \langle D_c \rangle)\boldsymbol{f} \tag{10}$$

where k is the admittance gain. D_c is the new preferred direction, which adjusts the tool tip toward and follows the original preferred direction,

$$D_c = (1 - k_d)[D]\boldsymbol{f} + k_d \|\boldsymbol{f}\|_2 \langle D \rangle \boldsymbol{u} \tag{11}$$

where k_d $(0 \leq k_d \leq 1)$ is a blending coefficient with which governs how quickly the tool tip is moved toward the reference direction.

2) Task Implementation: At each time step, the goal is to compute incremental joint motions $\Delta \boldsymbol{q}$, which then are fed to the low-level position servos. We compute the desired tool tip velocity using the admittance law described above, and convert this to an incremental 6-DoF tool motion $\Delta \boldsymbol{P}_t^d$:

$$\Delta \boldsymbol{P}_t^d = [\ (\boldsymbol{v}_t^d)_x \cdot \Delta t \quad (\boldsymbol{v}_t^d)_y \cdot \Delta t \quad (\boldsymbol{v}_t^d)_z \cdot \Delta t \quad 0 \quad 0 \quad 0\ ]^T \tag{12}$$

where Δt is the sample interval.

We identify three classes of requirements: in the tool tip frame, the tool boundary frame(s), and in the joint spaces. For each, we define an objective function ζ to be minimized and a set of linearized constraints.

a) Tool tip motion:: We require an incremental tool tip motion $\Delta \boldsymbol{P}_t$ to be as close as possible to some desired value $\Delta \boldsymbol{P}_t^d$. We express this as:

$$\zeta_t = \left\| \Delta \boldsymbol{P}_t - \Delta \boldsymbol{P}_t^d \right\|_2^2 \tag{13}$$

In order to enforce tip motion along that preferred direction, we approximate the direction of $\Delta \boldsymbol{P}_t$ to be same as $\Delta \boldsymbol{P}_t^d$

$$\Delta \boldsymbol{P}_t^{d\ T} \cdot \Delta \boldsymbol{P}_t \geq (1 - \epsilon_t) \left\| \Delta \boldsymbol{P}_t^d \right\|_2^2 \tag{14}$$

where ϵ_t is a small positive number. We relate the tool tip frame motion to the joint motion via the Jacobian $\Delta \boldsymbol{P}_t = J_t(\boldsymbol{q}) \Delta \boldsymbol{q}$. We rewrite (13) and (14) as

$$\begin{array}{ll} \zeta_t = \left\| W_t \cdot (J_t(\boldsymbol{q}) \Delta \boldsymbol{q} - \Delta \boldsymbol{P}_t^d) \right\|_2^2, \\ s.t. \qquad A_t \cdot J_t(\boldsymbol{q}) \Delta \boldsymbol{q} \leq \boldsymbol{b}_t \end{array} \tag{15}$$

where $A_t = -\Delta \boldsymbol{P}_t^{d\ T}$ and $\boldsymbol{b}_t = -(1 - \epsilon_t) \left\| \Delta \boldsymbol{P}_t^d \right\|_2^2$. $W_t = diag\{w_t\}$ denotes a diagonal matrix of weighting factors specifying the relative importance of each component of $\Delta \boldsymbol{P}_t$. Since we want to track the path tightly, we set the translational components of w_t to a fairly high value and let the rotational components to a low value.

b) Boundary constraints:: We want to ensure that the instrument itself 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

$$\boldsymbol{n}_b^{\ T} \cdot (\boldsymbol{P}_k + \Delta \boldsymbol{P}_k - \boldsymbol{P}_b) \geq \epsilon_{bnd} \tag{16}$$

where $\boldsymbol{P}_b$ and $\boldsymbol{P}_k$ are the position of the potential collision point pair on the surface and tool, respectively. $\boldsymbol{n}_b$ is the normalized normal vector at the contact point on the surface, and ϵ_{bnd} is a small positive number. The constraint described in equation (16) indicates that the angle between $(\boldsymbol{P}_k + \Delta \boldsymbol{P}_k - \boldsymbol{P}_b)$ and $\boldsymbol{n}_b$ is less than $90\deg$. Indeed, this constraint prevents the tool shaft from penetrating a plane. We can also define an objective function $\zeta_k = \|W_k \cdot \Delta \boldsymbol{P}_k\|_2^2$ expressing the desirability of minimizing extraneous motion of the tool near the boundary. We again rewrite these formulae in terms of $\Delta \boldsymbol{q}$.

$$\begin{array}{ll} \zeta_k = \|W_k \cdot \Delta \boldsymbol{P}_k\|_2^2, \\ s.t. \quad A_k \cdot J_k(\boldsymbol{q}) \Delta \boldsymbol{q} \leq \boldsymbol{b}_k \end{array} \tag{17}$$

where $A_k = [\ -\boldsymbol{n}_b \quad 0 \quad 0 \quad 0\ ]^T$ and $\boldsymbol{b}_k = -\epsilon_{bnd} + \boldsymbol{n}_b^{\ T} \cdot (\boldsymbol{P}_k - \boldsymbol{P}_b)$.

We can use very low values for w_k and rely mainly on the inequality constraints. An alternative approach is leaving the ζ_k term out of the optimization altogether. The number of constraints in this class changes dynamically and depends on how many closest-point pairs we generate based on the relative position of the tool and the geometric environment.

c) 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

$$\boldsymbol{q}_{\min} - \boldsymbol{q} \leq \Delta \boldsymbol{q} \leq \boldsymbol{q}_{\max} - \boldsymbol{q} \tag{18}$$

where $\boldsymbol{q}$ is the vector of the current values of the joint variables, and $\boldsymbol{q}_{\min}$ and $\boldsymbol{q}_{\max}$ 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. These can be rewritten in the form

$$\begin{array}{ll} \zeta_j = \|W_j \cdot \Delta \boldsymbol{q}\|_2^2, \\ s.t. \quad A_j \cdot \Delta \boldsymbol{q} \leq \boldsymbol{b}_j \end{array} \tag{19}$$

where

$$A_j = \begin{bmatrix} -I \\ I \end{bmatrix}, \boldsymbol{b}_j = \begin{bmatrix} \boldsymbol{q} - \boldsymbol{q}_{\min} \\ \boldsymbol{q}_{\max} - \boldsymbol{q} \end{bmatrix}$$

Again, we set w_j to low values for all of the joints and simply enforce the inequality constraints.

d) Putting it together:: We combine all of the task constraints and objective functions, and then obtain the overall optimization problem:

$$\underset{\Delta \boldsymbol{q}}{\arg\min} \left\| \begin{bmatrix} W_t & & \\ & W_k & \\ & & W_j \end{bmatrix} \cdot \left(\begin{bmatrix} J_t(\boldsymbol{q}) \\ J_k(\boldsymbol{q}) \\ I \end{bmatrix} \Delta \boldsymbol{q} - \begin{bmatrix} \Delta \boldsymbol{P}_t^d \\ 0 \\ 0 \end{bmatrix} \right) \right\|_2^2 \tag{20}$$

subject to

$$\begin{bmatrix} A_t & & \\ & A_k & \\ & & A_j \end{bmatrix} \cdot \begin{bmatrix} J_t(\boldsymbol{q}) \\ J_k(\boldsymbol{q}) \\ I \end{bmatrix} (\Delta \boldsymbol{q}) \leq \begin{bmatrix} \boldsymbol{b}_t \\ \boldsymbol{b}_k \\ \boldsymbol{b}_j \end{bmatrix} \tag{21}$$

B. Target-inspection in Complicated Geometric Environment

Another useful procedure in endoscopic sinus surgery is to manipulate an endoscope to observe a target anatomy from different view. Our target-inspection task is then defined as keeping a virtual position $\boldsymbol{P}_v$, which is along the endoscope optical axis and $r\,mm$ away from the lens, staying on a given target position $\boldsymbol{P}_t$, while approaching the target position from different angles based on a user's input.

1) Task Implementation:

a) Tip constraints:: Once $\boldsymbol{P}_v$ reaches on the target position $\boldsymbol{P}_t$, we require the virtual point $\boldsymbol{P}_v$ to stay on $\boldsymbol{P}_t$:

$$\|\boldsymbol{P}_v - \boldsymbol{P}_t\| \leq \epsilon_t \tag{22}$$

where ϵ_t is a small positive number. We further approximate (22) using method described in [52] as

$$A_t J_t(\boldsymbol{q})\Delta\boldsymbol{q} \leq \boldsymbol{b}_t$$

$$A_t = \begin{bmatrix} \cos\alpha_1\cos\beta_1 & \cos\alpha_1\sin\beta_1 & \sin\alpha_1 & 0 & 0 & 0 \\ & \vdots & & & & \\ \cos\alpha_n\cos\beta_m & \cos\alpha_n\sin\beta_m & \sin\alpha_n & 0 & 0 & 0 \end{bmatrix},$$

$$\boldsymbol{b}_t = \begin{bmatrix} \epsilon_t \\ \vdots \\ \epsilon_t \end{bmatrix} - A_t \cdot \begin{bmatrix} \boldsymbol{P}_v - \boldsymbol{P}_t \\ \boldsymbol{0} \end{bmatrix}. \tag{23}$$

where $\alpha_i = \frac{i2\pi}{n}$, $\beta_j = \frac{j2\pi}{m}$. ϵ_t is the tool tip distance error tolerance.

We require our tool rotational motion proportional to the user's input $\boldsymbol{\tau}$, then we set the cost function as

$$\zeta_t = \|W_t \cdot (J_t(\boldsymbol{q})\Delta\boldsymbol{q} - k \cdot \boldsymbol{\tau})\|_2^2$$

b) Task constraints:: The other constraints, such as tool shaft boundary constraints and joint limitation constraints, are the same as we described in the "path-following" task. We combine all of the task constraints and objective functions, and then obtain the overall optimization problem, which is:

$$\arg\min_{\Delta\boldsymbol{q}} \left\| \begin{bmatrix} W_t & & \\ & W_k & \\ & & W_j \end{bmatrix} \cdot \left(\begin{bmatrix} J_t(\boldsymbol{q}) \\ J_k(\boldsymbol{q}) \\ I \end{bmatrix} \Delta\boldsymbol{q} - \begin{bmatrix} k\cdot\boldsymbol{\tau} \\ 0 \\ 0 \end{bmatrix} \right) \right\|_2^2 \tag{24}$$

subject to

$$\begin{bmatrix} A_t & & \\ & A_k & \\ & & A_j \end{bmatrix} \cdot \begin{bmatrix} J_t(\boldsymbol{q}) \\ J_k(\boldsymbol{q}) \\ I \end{bmatrix} \Delta\boldsymbol{q} \leq \begin{bmatrix} \boldsymbol{b}_t \\ \boldsymbol{b}_k \\ \boldsymbol{b}_j \end{bmatrix} \tag{25}$$

V. Experiments and Results

A. Experimental System Setup

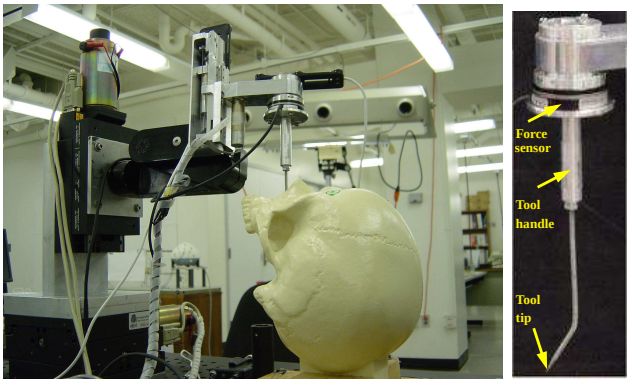

Fig. 5. (left) experimental setups, (right) the tool with a bent tip portion

Our current implementation used the Johns Hopkins University Steady Hand robot (JHU SHR) [50]. The JHU SHR is an admittance-controlled non-backdrivable 7-DoF robot with a remote-center-of-motion (RCM) kinematic structure. The robot is ergonomically appropriate for minimally invasive microsurgical tasks and provides $10\mu m$ scale, tremor free motion.

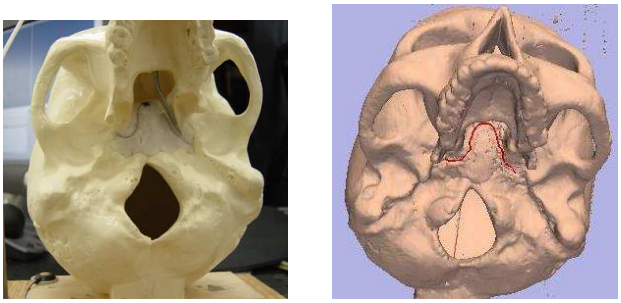

Fig. 6. (left) phantom skull and wire serving as path, (right) reconstructed 3D skull model and path

The experimental setup is shown in Figure 5. A commercial 6-DOF force sensor (Nano43, ATI Industrial Automation, North Carolina, USA) was mounted to the robots tool holder. The handle of the tool was mounted co-axially with the sensor, and the operator manipulates the tool by exerting forces on the handle. In Steady Hand collaborative control, forces exerted by the operator on the handle are measured and equation (10) is used to compute the desired motion of the robot. In the absence of other constraints, the robots' motions simply comply with the operator's hand forces. This feels to the operator like he or she is manipulating the tool in a viscous medium.

A tool with a bent tip was attached to the end-effector of the Steady Hand robot. The purpose of the bent part is 1) to simulate the shape of the sinus surgery instrument; 2) to simulate the angular view of a high degree endoscope. In the experiments, we constrained motion to avoid collisions with any portion of the tool shaft (i.e., either with the "straight" or "bent" portions). Motion constraints for the tool tip vary with the specific experiment, as discussed elsewhere.

Our results were based on experiments with a phantom skull. Five small spherical fiducials were attached to the skull, which was then CT scanned. 3D-SLICER's [56] built-in segmentation functionality was used to threshold, segment and render the skull CT dataset to create a triangulated model of the skull surface. Figure 6 shows the phantom and the constructed 3D model.

An optical tracking system Optotrak®3020 (Northern Digital, Inc. Waterloo, Canada) was employed for the robot calibration, system registration and the validation for experimental results. A 2GHz Pentium IV IBM PC-compatible machine running Windows 2000 was used for robot control and registration. The basic control loop (shown in Figure 4) was operated at a rate around 30Hz. PID controller for servoing the robot position runs on-board at a much higher rate (around 1000Hz).

B. System Registration Accuracy

The positions of the fiducials in CT coordinate frame were determined by a fiducial finding algorithm. The algorithm

marches through consecutive CT slices gathering pixels based on a threshold input and collecting adjacent hyperdense pixels into consolidated objects representing fiducials. The fiducials positions in Optotrak coordinate frame were gathered by calibrated pointers. We computed the transformation from the skull to the pre-operative CT coordinate frame by least square fitting method [57]. To evaluate the registration accuracy, we computed the residual error across the fiducials. The average residual registration error measured across the five fiducials upon five trails is $0.473 \pm 0.07\, mm$.

C. Surface Model Representation

We used a triangulated surface mesh to represent the phantom skull. In our experiment, we only used the nose and sinus portion of the resulting skull model. There are about 99,000 vertices and 182,000 triangles in this surface model. The average area of the triangles is around $0.5\ mm^2$. The average length of the triangle sides is around $0.9\ mm$. The time used for building a tree from this surface model is around 60 seconds.

There is some trade-off between mesh resolution, model fidelity, and computational efficiency, although the hierarchy nature of our covariance tree data structure limits this sensitivity. It would be a straightforward matter to vary the geometric tightness of collision constraints simply by stopping the tree search at a coarse level of detail.

D. Results of Target-Inspection

We marked a position at the bottom of the phantom skull as the target. This target position with respect to CT space was determined using an Optotrak pointer. We used the bent tip tool (shown in Figure 5) to represent a high degree endoscope. The tool was inserted through the phantom nasal cavity. The tool tip represented the virtual position $\boldsymbol{P}_v$.

A user held the handle to manipulate the tool. The robot read the applied force $\boldsymbol{f} \in \Re^3$ and provided assistance to avoid collisions between the tool shaft and the phantom, meanwhile maintain the tool tip at the given position.

We set distance error tolerance ϵ_t in (23) as $0.1\ mm$ to generate stiff tip motion constraints. k in equation (24) was set as 0.01. We used both the robot encoders and the Optotrak to record the tool tip position. The average distance of the tool tip $\boldsymbol{P}_v$ to target $\boldsymbol{P}_t$ for ten trials is $0.02 \pm 0.02\, mm$ measured by robot encoders, while the average distance measured by the Optotrak is $0.34 \pm 0.15\, mm$. The maximum distance from $\boldsymbol{P}_v$ to $\boldsymbol{P}_t$ is $0.08\ mm$ and $0.81\ mm$ measured by the robot encoder and the Optotrak respectively. Figure 7 shows the tool tip position measured by the robot encoders and the Optotrak. During the user operation, the tool shaft did not collide with the plastic skull. Figure 8 shows the relative position of the tool and the plastic skull from two different views.

E. Results of path-following simulation

A thin wire attached inside the nasal cavity of the plastic skull served as the target path. The target path with respect to

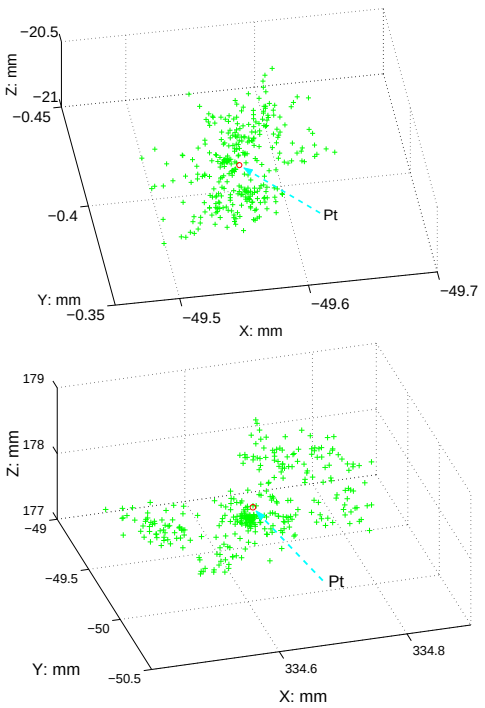

Fig. 7. The tool tip position measured by (top) the robot encoders and (bottom) the Optotrak in a target-inspection task. The green '+'s are the measured positions, the red 'o' is the target position.

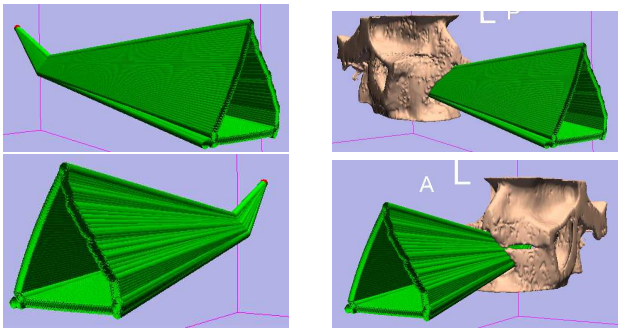

Fig. 8. Trajectories of the tool during a target-inspection trial from two different views. (left) the swept volume of the tool path, (right) the relative relation between the tool and the nasal part of the plastic skull

CT space was defined by tracing the wire with an Optotrak pointer. A 5-th degree bspline curve, which interpolated the gathered sample points, represented the target path. This path was transformed to robot coordinate frame after registration.

The tool was inserted through the phantom skull nasal cavity. In the path-following experiment, a user was asked to manipulate the tool to move along the wire as close as possible while avoiding the collision between the tool-shaft and the phantom.

We first applied force on the handle and performed computer simulation to check the feasibility of motion for path-following task given the constraints and robot kinematics. We chose the admittance ratio k_τ in equation (10) as 0 to enforce the tool-tip motion only along the preferred direction.

The control gain k_d in equation (11) was set as 0.2. The components of w_k in (17) and w_j in (19) were set as 0.01.

Figure 9 shows tool trajectories and the relative position of the tool with the phantom skull model. The result of our simulation suggests that the robot can exactly track the path while avoiding the collision between the tool-shaft and the skull.

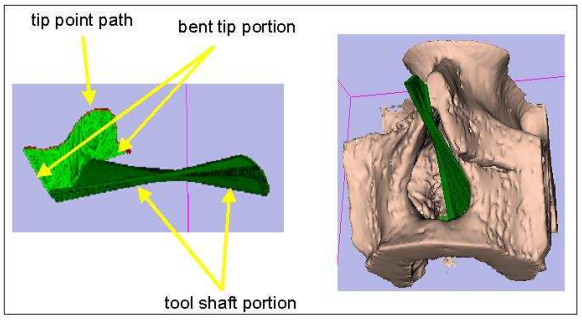

Fig. 9. Trajectories of the tool in a path-following simulation. (left) the swept volume of the tool path, (right) the relative position between the tool and the nasal part of the phantom skull model

In our experiment, the size of $\Delta \boldsymbol{q}$ in (20) and (21) was 7×1. The size of $A \cdot J$ in (21) varied from 20×7 to 39×7; the size of $W \cdot J$ in (20) varied from 13×7 to 37×7 according to the number of boundary constraints. With our 2GHz Pentium IV PC, the average time in each control interval for the boundary search and optimization problem solution is $6\ ms$.

According to [53], LSI problem is eventually reduced to NNLS problem. NNLS is solved iteratively and the algorithm always converges in a finite number of iteration. Based on the implementation of [53], NNLS returns with a feasible (approximated) solution if it fails to converge after $3 \times n$ (n is the number of decision variable, which is 7 in our case) iterations. In our experiments, the numbers of iteration of NNLS were less than 21. In other words, we obtained the exact solution, instead of approximated solution to the constructed optimization problem.

F. Performance comparison of cooperative and teleoperative control in path-following task

The main purpose of the experiments is to evaluate the improvement of users' performance of manipulating instrument with robot guidance using virtual fixtures derived from complex geometry compared to free-hand instrument manipulation. Our constrained control method works for both hands-on cooperative control operation and more traditional master-slave teleoperation. We evaluated users' performance for path-following task with both operation modes, and free hand as well. 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.

1) Protocol: The experimental system setup is described in section V-A and V-E. A tool was inserted through the phantom skull nasal cavity. The length of the tool was around $15\ mm$ and the weight was around $0.5kg$. It was based closely on real-life practice. The experiments required subjects to manipulate a tool to move along the wire attached to the bottom of phantom nasal cavity as close as possible while avoiding the collision between the tool-shaft and the phantom. We tested the user's performance in three modes.

- **Freehand mode:** The user held the tool and manipulated it without any assistance. No robot was involved.
- **Steady Hand robot guidance in hands-on cooperative operation mode:** We attached the tool handle to the force sensor. Both of the tool handle and the force sensor were mounted on the SHR end-effector. The user held the tool handle to manipulate the tool. The robot read the applied force $\boldsymbol{f} \in \Re^3$, provided assistance to avoid collisions between the tool shaft and the phantom and move the tool tip along the desired path as well.
- **Steady Hand robot guidance in remote teleoperation mode:** We employed a SpaceBall™ mouse (3D Connexion, Germany) to implement a simple teleoperator interface. The SpaceBall™ is a 6-DoF (three translational components and three rotational components) force sensor commonly used as a joy stick or mouse in computer graphics or gaming application. The user controlled the tool tip motion through the SpaceBall three translational components, $\boldsymbol{\tau} \in \Re^3$. The translational components were aligned with the robot base frame. The robot read the output of the SpaceBall $\boldsymbol{\tau} \in \Re^3$, provided assistance to avoid collisions and move the tool tip along the desired path. The desired tool tip velocity $\boldsymbol{v}_t^d$ then was determined by replace $\boldsymbol{f}$ by $\boldsymbol{\tau}$ in (10) and (11).

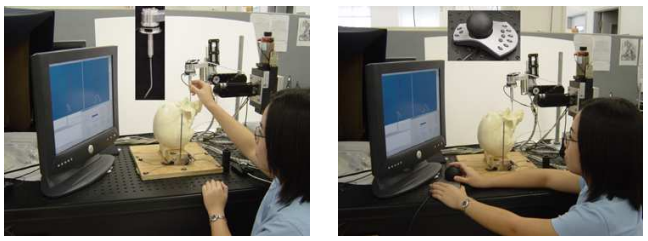

Fig. 10. (left) Hands-on cooperative control, and (right) Teleoperative control. In hands-on cooperative control, a force sensor and tool are mounted on the robot end-effector. Tool is "manipulated" by an operator holding a handle attached to the force sensor. The robot read the applied force and provided assistance. In remote teleoperative control we use 3D mouse to tele- control the robot.

A 3D visualization interface provided the user with visual information about 3D geometry, the tool position, orientation and the reference target path around the working site.

Ten subjects participated in the experiment. These subjects were varied in experience with the Steady Hand robot from novice to expert. The subjects were asked to perform five trials for each of three different modes. Between every two trials,

the subjects were given one or two minutes for rest.

An Optotrak rigid body was fixed to the tool. 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 average error is defined as the total error divided by the number of samples throughout the task. 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.

We compared the performance of different operation 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 assumed that there was no other external error: error was 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 measured the error on the system level.

2) 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 and trajectories of tip during task execution of the two Steady Hand robot guidance modes in robot context are shown in Figure 11 and 12. Although the error in hands-on cooperative mode ($0.204 \pm 0.01\,mm$) is slightly better than the error performance in teleoperative mode ($0.219 \pm 0.02mm$), there is no significant difference between two modes (paired t-test, $p = 0.31$) as shown in Table I. However the execution time in hands-on cooperative mode ($19.00 \pm 2.31\ s$) is significantly better than in teleoperative mode ($24.17 \pm 4.14\ s$). All subjects moved faster in hands-on cooperative mode.

TABLE I

ERROR AND TIME IN HANDS-ON COOPERATIVE MODE AND TELEOPERATIVE MODE FOR PATH-FOLLOWING TASK MEASURED BY THE ROBOT

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

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 Figure 13 and 14 show that commanded joint velocities change more smoothly in hands-on cooperative mode than in our simple teleoperative mode. The users SpaceBall input that

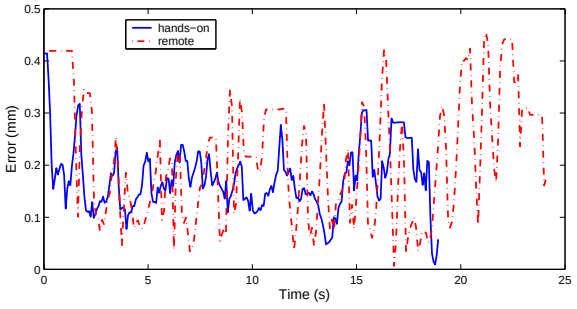

Fig. 11. Error profiles of a path-following task measured by the robot. In the robot context, there is no significant difference between two robot-assisted modes.

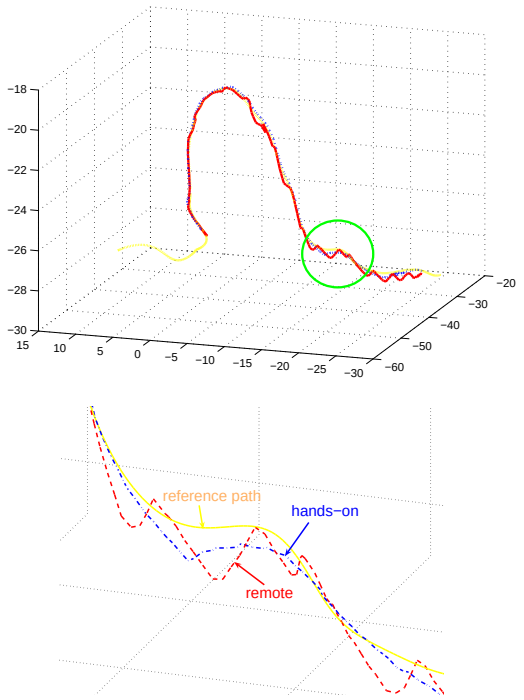

Fig. 12. (top) Trajectory of the tool tip in "path-following" task measured by the robot. (bottom) close-up of the highlighted part.

was used to calculate the joint velocities was not as smooth as the force input. The teleoperative control via SpaceBall is harder than hands-on cooperative control. The subjects also commented that the task was 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 remote teleoperative mode, as well as the free hand mode in the Optotrak tracking coordinate system. As shown in Table II, errors in both robot-assisted modes ($0.99 \pm 0.14\,mm$ in hands-on cooperative mode, $0.72 \pm 0.11\,mm$ in remote teleoperative mode) are significantly better than in free hand mode ($2.47 \pm 0.98\,mm$). Similarly, the execution times in robot-assisted modes ($19.00 \pm 2.31\,s$ in hands-on cooperative mode, $24.17 \pm 4.14\,s$ in remote teleoperative

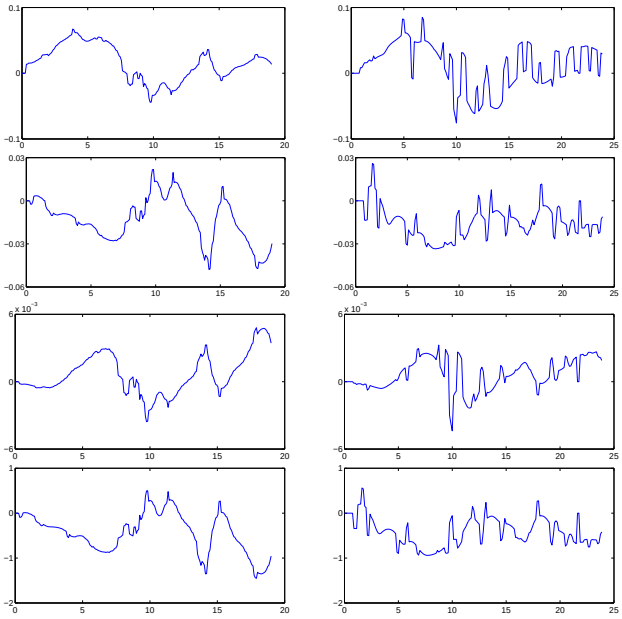

Fig. 13. Commanded velocity profiles of prismatic joints of the Steady Hand robot (joint No. 1,2,3,6) in path-following task (left) hands-on mode (right) remote mode. The labels of y-axis are Velocity(mm/s) and labels of x-axis are Time(s).

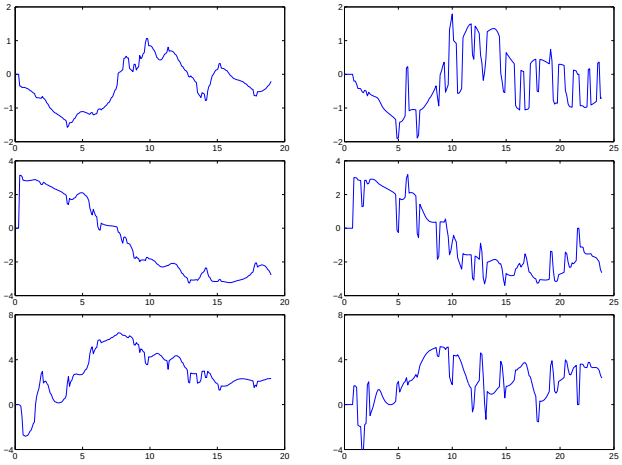

Fig. 14. Commanded velocity profiles of revolute joints of the Steady Hand robot (joint No. 4,5,7) in path-following task (left) hands-on mode (right) remote mode. The labels of y-axis are Velocity(deg/s) and labels of x-axis are Time(s).

mode) are better than free hand mode ($27.56 \pm 8.82\ s$). The error profiles and the trajectories of tip of three modes during task execution are shown in Figure 15 and Figure 16.

c) Discussion: As might be expected, the error of both robot-assisted modes in Optotrak context is much larger than in robot context. In the former case, in addition to the control algorithm and robot controller, the robot calibration and system registration errors are other two main sources of the tip motion error. Moreover, the accuracy of the Optotrak tracking system ($< 0.2\ mm$ in space) also contributes to the overall error.

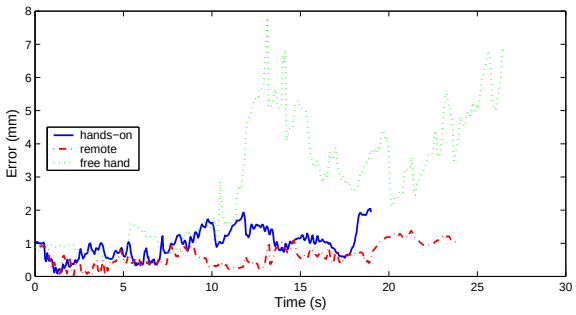

Fig. 15. Error profiles of the "path-following" task measured by the Optotrak.

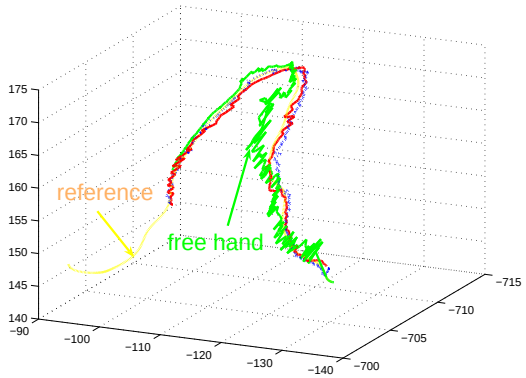

Fig. 16. Trajectory of the tool tip in "path-following" task measured by the Optotrak. Errors in both robot-assisted modes are significantly better than in free hand mode.

Although hands-on cooperative mode shows almost same error as teleoperative mode when the robot is used to measure its own performance, the relative accuracy of the teleoperative mode is better than that of the hands-on cooperative mode 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 harder than was necessary to cause the desired motions. In teleoperation mode, of course, the users exerted no forces on the tool.

As noted below, we are currently adapting our constrained control paradigm to full 6-DOF master-slave systems such as the daVinci, although the daVinci would not necessarily be a particularly appropriate robot for sinus surgery. Although we have not done an exhaustive comparative study, our anecdotal experience leads us to believe that the major effect of changing master devices is likely to be on the time, rather than on the accuracy of path performance. The choice of hands-on guiding *vs.* teleoperation, in any case, is likely to be dictated by other ergonomic and economic considerations which will vary with

TABLE II

ERROR AND TIME IN THREE DIFFERENT MODE FOR "PATH-FOLLOWING" TASK MEASURED BY OPTOTRAK

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

clinical application.

As mentioned, the visualization interface only provides the user geometric information around the working site (the area around the tool tip). There is no further feedback other than user's awareness of the relative position between the tool shaft and phantom. The reader may notice that the position error shown in Figure 15 is much larger in the second half of the task in the free hand mode. That is because the relative position of the second part of the path with respect to the cavity is complicated. With the free hand mode, the subjects needed to tilt and rotate the tool simultaneously to keep the tool tip on the path and manage to avoid collision based on his awareness.

VI. CONCLUSION AND FUTURE WORK

We have developed a real time task-based control method of surgical robot in a precise interactive minimally invasive surgical environment equally applicable to teleoperation and hands-on cooperative control. Robot-guidance (both hands-on cooperative and remote teleoperative control) employing 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 experimental results show significant improvement in both accuracy and execution time, compared to free hand instrument manipulation. The results suggest that the constrained optimization robot control can release the surgeon's tension on avoiding collision of the instrument to the anatomic structure during precision instrument manipulation in minimally invasive surgical procedure.

We have compared the performance of hands-on cooperative operation and teleoperation to control a tool manipulated in a complicated working volume. The performance comparison 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 is identical to that of teleoperative mode. The contact between the user and the tool that are mounted on the robot end-effector introduces perturbations into the system for hands-on cooperative control.

The performance comparison experiments reported here are primarily intended to demonstrate the performance of our constrained control method in both standard teleoperation and hands-on cooperative control paradigms. They are not designed as definitive experiments comparing these two paradigms in general. 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.

As discussed above, the primary focus of this paper has been development of techniques for controlling the motion of teleoperated and cooperatively controlled surgical robots in the presence of complex geometric constraints associated with patient anatomy, using endoscopic sinus surgery as a motivating application.

Further work will be required before this work can be applied clinically. In addition to the obvious progression through animal model and cadaver testing and development of customized instruments for the robot to hold, important further considerations are: extension of our methods to work with constraints associated with soft tissues, registration between robot, patient, and image coordinates; and accommodating patient motion.

The current experiment used rigid models derived from CT images of a plastic skull. Clinical application would require CT or MRI images of an actual patient, including soft tissues. Usually CT scans are used to plan and to perform sinus surgery. An MRI scan is only used to understand what type of disease process is going on, i.e., to form a differential diagnosis. Our entire paradigm for performing safe surgery is based on understanding the bony anatomy, because it is so important for identifying surgical landmarks. For sinus surgery the bony anatomy is what is altered; the exception will be polyp disease which is beyond the scope of this paper. As for the remaining soft tissue structures, they would be amenable to our approach. In treatment planning, the surgeon would need to determine how much impingement into soft tissue is permitted at different places and could also specify other constraints on permissible tool motion. Especially where soft tissue is involved, it is desirable to provide some compliance in the corresponding constraints. Similarly, some compliance and adaptation is important to permit the surgeon to accommodate registration uncertainty. To address these issues, we have been exploring extensions to our formulation, using nonlinear constraints and objective functions to implement "soft" virtual fixtures. [58].

Registration and patient motion accommodation could be performed in a manner similar to existing surgical navigation systems [21], [23], [59], [60] possibly with some additional physical supports to prevent very fast head motions. We have also been investigating alternative direct registration methods from endoscopic video [48].

Finally, clinical application of this work will require development of a clinically qualified version of our robot and instrumentation to replace the engineering prototype reported here. Although these considerations are beyond the scope of this paper, we note that our basic control methods are independent of the particular robot used. For example, in other work, we have applied our approach to control of a modified

daVinci robot, as well as an experimental "snake" robot controlled from a modified daVinci master controller [61], rather than the simple 3D joystick used in the experiments reported here. Most probably, we will develop an appropriate industry collaboration to develop a clinically deployed version, although we are also considering appropriate next steps in this direction to be taken internally.

ACKNOWLEDGMENT

The authors gratefully acknowledge the advice and collaboration of Dr. Allison Okamura, Dr. Gregory Hager, Ankur Kapoor, Gregory Fischer, Panadda Marayong and all the experimental subjects for these experiments.

REFERENCES

[1] L. B. Rosenberg, "Virtual fixtures," *PhD Thesis, Department of Mech. Eng., Stanford University*, 1994.

[2] F. Lai and R. D. Howe, "Evaluating control modes for constrained robotic surgery," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 603–609, 2000.

[3] M. A. Peshkin, J. E. Colgate, W. Wannasuphoprasit, C. A. Moore, R. B. Gillespie, and P. Akella, "Cobot architecture," *IEEE Transactions on Robotics and Automation*, vol. 17, no. 4, pp. 377–390, 2001.

[4] S. Payandeh and Z. Stanisic, "On application of virtual fixture as an aid for telemanipulation and training," *10th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, pp. 18–23, 2002.

[5] Z. Stanisic, S. Payandeh, and E. Jackson, "Virtual fixture as an aid for teleoperation," *Proc. 9th Canadian Aeronautic and Space Institute Conf*, 1996.

[6] J. J. Abbott, G. D. Hager, and A. M. Okamura, "Steady-hand teleoperation with virtual fixtures," *12th IEEE Int. Workshop on Robot and Human Interactive Communication(RO-MAN 2003)*, pp. 145–151, 2003.

[7] A. Bettini, P. Marayong, S. Lang, A. M. Okamura, and G. D. Hager, "Vision assisted control for manipulation using virtual fixtures," *IEEE Transactions on Robotics*, vol. 20, no. 6, pp. 953–966, 2004.

[8] A. Bettini, S. Lang, A. M. Okamura, and G. D. Hager, "Vision assisted control for manipulation using virtual fixtures," *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 1171–1176, 2001.

[9] K. Kosuge, K. Takeo, and T. Fukuda, "Unified approach for teleoeration of vitural and real environment - manipulation based on reference dynamics," *Proc. IEEE Int. Conf. Systems, Man and Cybernetics*, pp. 938–943, 1995.

[10] S. E. Everett and R. V. Dubey, "Vision-based end-effector alignment assistance for teleoperation," *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 543–549, 1999.

[11] G. S. Guthart and J. K. Salisbury, "The intuitive telesurgery system: Overview and application," *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 618–621, 2000.

[12] P. Marayong, A. Bettini, and A. M. Allison, "Effect of virtual fixture compliance on human-machine cooperative manipulation," *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 1089–1095, 2002.

[13] J. Troccaz and Y. Delnondedieu, "Semi-active guiding systems in surgery. a two-dof prototype of the passive arm with dynamic constraints(padyc)," *Mechatronics*, vol. 6, no. 4, pp. 399–421, 1996.

[14] J. Troccaz, M. A. Peshkin, and B. L. Davies, "The use of localizers, robots and synergistic devices in cas," *Proc. 1st Joint Conf. CVRMed and MRCAS*, pp. 727–736, 1997.

[15] J. Troccaz and Y. Delnondedieu, "Synergistic robots for surgery: an algorithmic view of the approach," *Proc. Workshop on Algorithmic Foundations of Robotics*, 1998.

[16] B. L. Davies, S. J. Harris, W. J. Lin, R. D. Hibberd, R. Middleton, and J. C. Cobb, "Active compliance in robotic surgery - the use of force control as a dynamic constraint," *Proc. Inst. Mech. Eng. H*, vol. 211, no. 4, pp. 285–292, 1997.

[17] S. Park, R. D. Howe, and D. F. Torchiana, "Virtual fixtures for robotic cardiac surgery," *Proc. Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pp. 1419–1420, 2001.

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

[19] O. Schneider and J. Troccaz, "A six-degree-of-freedom passive arm with dynamic constraints (padyc) for cardiac surgery application: preliminary experiments," *Computer Aided Surgery*, vol. 6, pp. 340–351, 2001.

[20] J. Wurm, H. Steinhart, K. Bumm, M. Vogele, C. Nimsky, and H. Iro, "A novel robot system for fully automated paranasal sinus surgery," *International Congress Series*, vol. 1256, pp. 633–638, 2003.

[21] G. Strauss, K. Koulechov, R. Richter, A. Dietz, C. Trantakis, and T. Lueth, "Navigated control in functional endoscopic sinus surgery," *Int. J Medical Robotics and Computer Assisted Surgery*, vol. 1(3), pp. 31–41, 2005.

[22] K. Koulechov, G. Strauss, R. Richter, C. Trantakis, and T. C. Lueth, "Mechatronical assistance for paranasal sinus surgery," *Proc. Computer Assisted Radiology and Surgery (CARS)*, pp. 636–641, 2005.

[23] K. Koulechov, G. Strauss, A. Dietz, M. Strauss, M. Hofer, and T. C. Lueth, "Fess control: Realization and evaluation of navigated control for functional endoscopic sinus surgery," *Computer Aided Surgery*, vol. 11(3), pp. 147–159, 2006.

[24] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, "Task-priority based redundancy control of robot manipulator," *International Journal of Robotics Research*, vol. 6, no. 2, pp. 3–15, 1987.

[25] C. Klein and B. E. Blaho, "Dexterity measures for the design and control of kinematically redundant manipulators," *International Journal of Robotics Research*, vol. 2, no. 6, pp. 72–83, 1987.

[26] J. M. Hollerbach and K. C. Suh, "Redundancy resolution of manipulators through torque optimization," *IEEE Transactions of Robotics and Automation*, vol. 3, no. 4, pp. 308–316, 1987.

[27] R. J. Spiteri, D. K. Pai, and U. M. Ascher, "Programming and control of robots by means of differential algebraic inequalities," *International Journal of Robotics Research*, vol. 16, no. 2, pp. 135–145, 2000.

[28] D. E. Whitney, "Resolved motion rate control of manipulators and human prostheses," *IEEE Transation on Man-Machine System*, vol. 10, no. 2, pp. 47–53, 1969.

[29] A. A. Maciejewski and C. A. Klein, "Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments," *International Journal of Robotics Research*, vol. 4, no. 3, pp. 109–117, 1985.

[30] A. A. Mohamed and C. Chevallereau, "Resolution of robot redundancy in the cartesian space by criteria optimization," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 646–651, 1993.

[31] H. Seraji, M. K. Long, and T. S. Lee, "Motion control of 7-dof arms - the configuration control approach," *IEEE Transactions on Robots and Automation*, vol. 9, no. 2, pp. 125–139, 1993.

[32] P. Chiacchio, S. Chiaverini, L. Sciavicco, and B. Siciliano, "Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy," *International Journal of Robotics Research*, vol. 10, no. 4, pp. 410–425, 1991.

[33] Y. H. Kim and F. L. Lewis, "Neural network output feedback control of robot manipulators," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 2, pp. 301–309, 1999.

[34] S. S. Ge, C. C. Hang, and L. C. Woon, "Adaptive neural network control of robot manipulators in task space," *IEEE Transactions on Industrial Electronics*, vol. 44, no. 6, pp. 746–752, 1997.

[35] J. Funda, R. H. Taylor, B. Eldridge, S. Gomory, and K. G. Gruben, "Constrained cartesian motion control for teleoperated surgical robots," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, pp. 453–465, 1996.

[36] L. Adhami, E. Coste-Maniere, and J. D. Boissonnat, "Planning and simulation of robotically assisted minimal invasive surgery," *Proc. Medical Image Computing and Computer Assisted Intervention (MICCAI)*, pp. 624–633, 2000.

[37] E. Costse-Maniere, L. Adhami, R. Severac-Bastide, A. Lobontiu, J. Salsibury, J. D. Boissonnat, N. Swarup, G. Guthart, E. Mousseaux, and A. Carpentier, "Optimized port placement for the totally endoscopic coronary artery bypass grafting using the davinci robotic system," *Proc. Int. Symp. Experimental Robotics*, pp. 199–208, 2000.

[38] J. Cannon, J. Stoll, S. Selha, P. Dupont, R. D. Howe, and D. Torchiana, "Port placement planning in robot-assisted coronary artery bypass," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 5, pp. 912–917, 2003.

[39] A. Schweikard, R. Tombropoulos, L. E. Kavraki, J. R. Adler, and J. C. Latombe, "Treatment planning for a radiosurgical system with general kinematics," *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 1720–1727, 1994.
[40] R. Z. Tombropoulos, J. R. Adler, and J. C. Latombe, "Carabeamer: a treatment planner for a robotic radiosurgical system with general kinematics," *Medical Imaging Analysis*, vol. 3, no. 3, pp. 237–264, 1999.
[41] C. Sim, M. Teo, W. Hg, C. Yap, Y. Loh, T. Yeo, S. Bai, and C. Lo, "Hexapod intervention planning for a robotic skull-base surgery system," *Proc. Medical Image Computing and Computer Assisted Intervention (MICCAI)*, 2001.
[42] "International rhinosinusitis advisory board. infectious rhinosinusitis in adults: classification, etiology and management." pp. 5–22, 1997.
[43] B. Senior, C. Glaze, and M. Benninger, "Use of the rhinosinusitis disability index in rhinolic disease," *American Journal of Rhinology*, vol. 15, pp. 15–20, 2001.
[44] M. S. Benninger, S. Holzer, and J. Lau, "Diagnosis and treatment of uncomplicated acute bacterial rhinosinusitis: Summary of the agency for health care policy and research evidence-based report," *Otolaryngol head Neck Surg*, vol. 122, pp. 1–7, 2000.
[45] R. Aust and B. Drettner, "The functional size of the human maxillary ostium in vivo." *Acta Otolaryngol*, vol. 78, pp. 432–435, 1975.
[46] R. H. Taylor and D. Stoianovici, "Medical robotics in computer-integrated surgery," *IEEE Transactions on robotics and sutomation*, vol. 19, no. 5, pp. 765–781, 2003.
[47] S. Lavallee, "Registration for computer-integrated surgery: methodology, state of the art," *Computer-Integrated Surgery Technology and Clinical Applications*, pp. 77–97, 1996.
[48] D. Burschka, M. Li, M. Ishii, R. H. Taylor, and G. D. Hager, "Scale-invariant registration of monocular endoscopic images to ct-scans for sinus surgery," *Medical Image Analysis*, vol. 9(5), pp. 413–426, 2005.
[49] R. H. Taylor, J. Funda, B. Eldridge, K. Gruben, D. LaRose, S. Gomory, M. Talamini, L. R. Kavoussi, and J. Anderson, "A telerobotic assistant for laparoscopic surgery," *IEEE Eng. Med. Biol. Mag.*, vol. 14, pp. 279–287, 1995.
[50] R. H. Taylor, P. Jensen, L. L. Whitcomb, A. Barnes, R. Kumar, D. Stoianovici, P. Gupta, Z. Wang, E. deJuan, and L. Kavoussi, "A steady-hand robotic system for microsurgical augmentation," *International Journal of Robotics Research*, vol. 18, no. 12, pp. 1201–1210, 1999.
[51] M. Cavasoglu, F. Tendick, M. Cohn, and S. Sastry, "A laparoscopic telesurgical workstation," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 4, pp. 728–739, 1999.
[52] M. Li, A. Kapoor, and R. H. Taylor, "A constrained optimization approach to virtual fixtures," *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pp. 2924–2929, 2005.
[53] C. Lawson and R. Hanson, "Solving least squares problems," *Englewood Cliffs, NJ: Prentice-Hall*, 1974.
[54] J. J. Williams, R. Taylor, and 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.
[55] G. D. Hager, "Vision-based motion constraints," *IEEE International Workshop on Intelligent Robots and Systems, Workshop on Visual Servoing*, 2002.
[56] `http://www.slicer.org`.
[57] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3-d point sets," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9(5), pp. 698–700, 2000.
[58] A. Kapoor, M. Li, and R. H. Taylor, "Constrained control for surgical assistant robots," *Proc. IEEE Int. Conf. on Robotics and Automation*, pp. 231–236, 2006.
[59] `http://www.stealthstation.com`.
[60] `http://www.brainlab.com`.
[61] A. Kapoor, K. Xu, W. Wei, N. Simann, and R. H. Taylor, "Telemanipulation of snake-like robots for minimally invasive surgery of the upper airway," in *Medical Image Computing and Computer Assisted Intervention (MICCAI), Medical Robotics Workshop*, Copenhagen, Denmark, October 2006, (to appear).