Spatial Motion Constraints: Theory and Demonstrations for Robot Guidance Using Virtual Fixtures

Panadda Marayong[1], Ming Li[2],
Allison M. Okamura[1], and Gregory D. Hager[2]
1. Department of Mechanical Engineering
2. Department of Computer Science
Engineering Research Center for Computer-Integrated
Surgical Systems and Technology
The Johns Hopkins University
Baltimore, MD 21218
{*panadda, aokamura*}*@jhu.edu*, {*liming, hager*}*@cs.jhu.edu*

Abstract - **In this article, we describe and demonstrate control algorithms for general motion constraints. These constraints are designed to enhance the accuracy and speed of a user manipulating in an environment with the assistance of a cooperative or telerobotic system. Our method uses a basis of preferred directions, created off-line or in real-time using sensor data, to generate virtual fixtures that may constrain the user to a curve, surface, orientation, etc. in space. Open loop virtual fixtures seek only to maintain user motion along preferred directions, whereas closed loop fixtures additionally guide the user toward a point, line, or surface. This article demonstrates and compares the effects of open and closed loop fixtures in both autonomous and human-machine cases.**

I. INTRODUCTION

Our group in the Engineering Research Center for Computer Integrated Surgical Systems (CISST ERC) has been working to create surgical systems that improve both the speed and precision of medical interventions. Our goal is to design mechanisms that selectively provide cooperative assistance to a surgeon, while allowing the surgeon to retain ultimate control of the procedure.

In recent work, we have focused on developing assistance methods for microsurgery. Here, the challenges of small physical scale accentuate the need for dexterity enhancement, but the unstructured nature of the tasks dictates that a human be directly "in the loop." For example, retinal vein cannulation [15] involves the insertion of a needle of approximately 20-50 microns in diameter into the lumen of a retinal vein (typically 100 microns in diameter or less, approximately the diameter of a human hair). At these scales, tactile feedback is practically non-existent, and depth perception is limited to what can be seen through a stereo surgical microscope. In short, such a procedure is at the limit of what is humanly possible in conventional surgical practice.

Given the scale of operation, the most obvious need is to increase the precision of human motion, ideally without slowing or limiting the surgeon. In recent work [1], [2], [5], [6], [7], [9], we have begun to develop assistance methods that are based on manipulating the apparent compliance of tools simultaneously held by both a surgeon and a robot. Intuitively, if a tool is extremely stiff, then it is easier to achieve high precision of motion, and to remove tremor. Conversely, a low stiffness makes it possible to perform large-scale "transport" motions. The constraints developed in this work are a specific type of virtual fixture, which we term "guidance" virtual fixtures. Other virtual fixture implementations, often called "forbidden region virtual fixtures," are described in [8], [10], [11], [12].

In this paper, we provide a demonstration of motion constraints with varying compliance that were described for the general spatial case in [3]. First, we review our algorithm for generating spatial motion constraints. We then describe implementation and experiments with two constraint types: rotational and translational virtual fixtures. For each constraint type, we explore the effects on performance of adding a closed loop term to control algorithm. Finally, we present the sensing requirements for generating n-dimensional constraints.

II. DESCRIBING SPATIAL MOTION CONSTRAINTS

Our work has been motivated by the JHU Steady Hand Robot, and, in particular, an assistance paradigm of direct manipulation (Figure 1). Briefly, the JHU Steady Hand robot is a 7 DOF robot equipped with a force sensing handle at the endpoint. Tools are mounted at the endpoint, and "manipulated" by an operator holding the force handle. The robot responds to the applied force, thus implementing a means of direct control for the operator. The robot has been designed to provide micron-scale accuracy, and
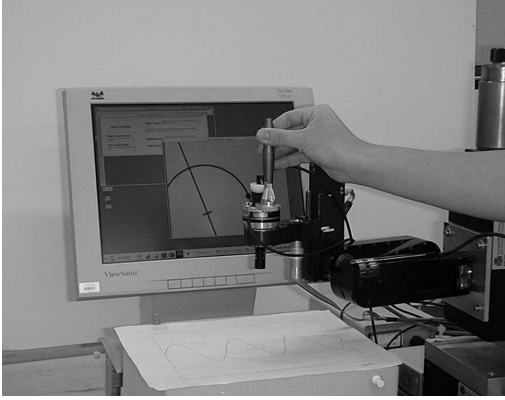
Fig. 1. The experimental setup for the Steady Hand Robot using virtual fixtures to assist in a path following task.

to be ergonomically appropriate for minimally invasive microsurgical tasks [14]. Our algorithms are developed to work with the admittance control structure of this robot.

In this section, we describe the basic admittance control model used in our implementation, extend this control to anisotropic compliances, and finally relate anisotropic compliances to an underlying task geometry. In the remainder of this paper, transpose is denoted by $'$, scalars are written lowercase in normal face; vectors are lowercase and boldface; and matrices are normal face uppercase.

A. Virtual Fixtures as a Control Law

In what follows, we model the robot as a purely kinematic Cartesian device with tool tip position $\mathbf{x} \in SE(3)$ and a control input that is the endpoint velocity $\mathbf{v} = \dot{\mathbf{x}} \in \mathbb{R}^6$, all expressed in the robot base frame. The robot is guided by applying forces and torques $\mathbf{f} \in \mathbb{R}^6$ on the manipulator handle, likewise expressed in robot base coordinates.

In the Steady Hand paradigm, the relationship between velocity and motion is derived from an admittance control law:

$$\mathbf{v} = c\mathbf{f}, \tag{1}$$

where $c > 0$ controls the "compliance" of the robot to user input.

When using Equation (1), the manipulator is equally compliant in all directions. If we replace the scalar c with a diagonal matrix C, we can change the compliance of the manipulator in the coordinate directions [1], [2], [3]. For example, setting all but the first two diagonal entries to zero would create a system that permitted translational motion in the x-y plane only. We term this type of anisotropic compliance a *guidance virtual fixture*. In the case above, the fixture is "hard," meaning that it permits motion in a subspace of the workspace. "Soft" virtual fixtures can also be created by allowing some compliance

in the non-preferred directions (e.g., setting the third and later diagonal entries to small numbers instead of zero). Now, motion in all directions is allowed, but some directions are easier to move in than others. We refer to the motions with high compliance as *preferred* directions, and the remaining directions as *non-preferred* directions.

B. Virtual Fixtures as Geometric Constraints

While it is clearly possible to continue to extend the notion of virtual fixture purely in terms of compliances, we instead prefer to take a more geometric approach, as suggested in [1], [2]. We will develop this geometry by specifically identifying the preferred and non-preferred directions of motion at a given time point t. To this end, let us assume that we are given a $6 \times n$ time-varying matrix $D = D(t)$, $0 < n < 6$. Intuitively, D represents the instantaneous preferred directions of motion. For example, if n is 1, the preferred direction is along a curve in SE(3); if n is 2 the preferred directions span a surface; and so forth.

From D, we define two projection operators, the span and the kernel of the column space, as

$$\text{Span}(D) \equiv [D] = D(D'D)^{-1}D' \tag{2}$$
$$\text{Ker}(D) \equiv \langle D \rangle = I - [D] \tag{3}$$

This formulation assumes that D has full column rank. It will occasionally be useful to deal with cases where the rank of D is lower than the number of columns (in particular, the case when $D = 0$). For this reason, we will assume $[\cdot]$ has been implemented using the pseudo-inverse [13, pp. 142–144] and write

$$\text{Span}(D) \equiv [D] = D(D'D)^+D' \tag{4}$$
$$\text{Ker}(D) \equiv \langle D \rangle = I - [D] \tag{5}$$

where the properties of the operators $[D]$ and $\langle D \rangle$ are as described in [3].

By decomposing the input force vector, $\mathbf{f}$, into two components

$$\mathbf{f}_D \equiv [D]\mathbf{f} \quad \text{and} \quad \mathbf{f}_\tau \equiv \mathbf{f} - \mathbf{f}_D = \langle D \rangle \mathbf{f}, \tag{6}$$

it follows that $\mathbf{f}_D \cdot \mathbf{f}_\tau = 0$ and that $\mathbf{f}_D + \mathbf{f}_\tau = \mathbf{f}$. Combining (6) and (1), we can now write

$$\mathbf{v} = c\mathbf{f} = c(\mathbf{f}_D + \mathbf{f}_\tau). \tag{7}$$

We now introducing a new compliance $c_\tau \in [0, 1]$ that attenuates the non-preferred component of the force input. With this we arrive at

$$\begin{aligned} \mathbf{v} &= c(\mathbf{f}_D + c_\tau \mathbf{f}_\tau) \\ &= c([D] + c_\tau \langle D \rangle)\mathbf{f}. \end{aligned} \tag{8}$$

Thus, the final control law is in the general form of an admittance control with a time-varying gain matrix determined by $D(t)$. By choosing c, we control the overall

compliance of the system. Choosing c_τ low imposes the additional constraint that the robot is stiffer in the non-preferred directions of motion. As noted above, we refer to the case of $c_\tau = 0$ as a *hard virtual fixture*, since it is not possible to move in any direction other than the preferred direction. All other cases will be referred to as *soft virtual fixtures*. In the case $c_\tau = 1$, we have an isotropic compliance as before.

It is also possible to choose $c_\tau > 1$ and create a virtual fixture where it is easier to move in non-preferred directions than preferred. In this case, the natural approach would be to switch the role of the preferred and non-preferred directions.

III. CHOOSING THE PREFERRED DIRECTION

The development to this point directly supports the following types of guidance:

- Motion in a subspace: suppose we are supplied with a time-varying, continuous function $D = D(t)$. Then applying (8) yields a motion constraint within that subspace.
- Motion to a target pose $\mathbf{x}_t \in SE(3)$: Suppose that we have a control law $\mathbf{u} = f(\mathbf{x}, \mathbf{x}_t)$ such that by setting $\mathbf{v} = \mathbf{u}$,

$$\lim_{t \to \infty} \mathbf{x} = \mathbf{x}_t.$$

By choosing $D = \mathbf{u}$ and applying (8), we create a virtual fixture that guides the user to the given target pose.

Based on the control law (8), the position of the tool tip is allowed to move *parallel* to the preferred direction. However, if there is an underlying desired reference trajectory or setpoint, and the tool tip is not within that reference, then it is necessary to adjust the preferred direction to move the tool tip toward it. Consider D as our preferred direction and $\mathbf{u} = f(\mathbf{x}, S)$ as the signed distance of the tool tip to the reference direction, where S is the motion objective. We define a new preferred direction as follows:

$$D_c(\mathbf{x}) = [(1 - k_d)[\mathbf{D}]\mathbf{f}/\|\mathbf{f}\| + k_d\langle\mathbf{D}\rangle\mathbf{u}] \quad 0 < k_d < 1. \tag{9}$$

Choosing the constant k_d governs how quickly the tool is moved toward the reference direction. One minor issue here is that the division by $\|\mathbf{f}\|$ is undefined when no user force is present. Anticipating the use of projection operators, we make use of a scaled version of (9):

$$D_c(\mathbf{x}) = (1 - k_d)[\mathbf{D}]\mathbf{f} + k_d\|\mathbf{f}\|\langle\mathbf{D}\rangle\mathbf{u}, \tag{10}$$

where $0 < k_d < 1$. We now apply (8) with $D = D_c$.

Using the properties of the projection operators, it can be shown that combining (10) with (8) results in a law equivalent to a pure subspace motion constraint. One potential disadvantage of this law is that when user applied

force is zero, there is no virtual fixture as there is no defined preferred direction. Thus, there is a discontinuity at the origin. However, in practice the resolution of any force sensing device is usually well below the numerical resolution of the underlying computational hardware, so the user will never experience this discontinuity.

We can now state the following informal rule to determine a virtual fixture control law:

1) A surface $S \subseteq SE(3)$ (the motion objective)
2) A control law $\mathbf{u} = f(\mathbf{x}, S)$ where by setting $\mathbf{v} = \mathbf{u}$,

$$\lim_{t \to \infty} \mathbf{x} \in S.$$

(the control law moves the tool tip into S)
3) A rule for computing preferred directions $D = D(t)$ relative to S where $\langle D\rangle\mathbf{u} = 0$ iff $\mathbf{u} = 0$ (the motion direction is consistent with the control law)

then applying the following choice of preferred direction:

$$D_g(\mathbf{x}) = (1 - k_d)[D]\mathbf{f} + k_d\|\mathbf{f}\|\langle D\rangle\mathbf{u} \quad 0 < k_d < 1 \tag{11}$$

yields a virtual fixture that controls the robot toward S and seeks to maintain user motion within that surface.

Note that a sufficient condition for condition 3 above to be true is that, for all pairs $\mathbf{u} = \mathbf{u}(t)$ and $D = D(t)$, $[D]\mathbf{u} = 0$. This follows directly from the properties of projection operators described in [3].

IV. ROTATIONAL VIRTUAL FIXTURE IMPLEMENTATION

We now present an illustrative case of virtual fixture implementation and the effect of control parameters (e.g., servo gain) on system performance. In this example, a predefined spatial virtual fixture is used to guide the robot. We applied the virtual fixture control law using the Remote Center Motion (RCM) module of the JHU Steady-Hand robot [14], which rotates the end-effector of the robot about a fixed point in the workspace (RCM point). The goal is to rotate the tool about the robot z axis with a fixed angle, by simultaneously rotating about the robot x and y axes. This creates a cone-shaped motion with the tip located at the RCM point. In both open loop and closed loop cases, the compliance (Equation (1)) is set to zero. For a pure rotational motion, our preferred direction can be defined in the Cartesian space by a 6 x 1 time-varying matrix D in the robot base frame as

$$D = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}' \tag{12}$$

The orientation of the tool (obtained from the robot encoders) is expressed in the joint coordinates of the robot as θ_1 and θ_2 for rotation about the x and y axes, respectively. Thus, the *preferred* direction can then be expressed in the tool frame as
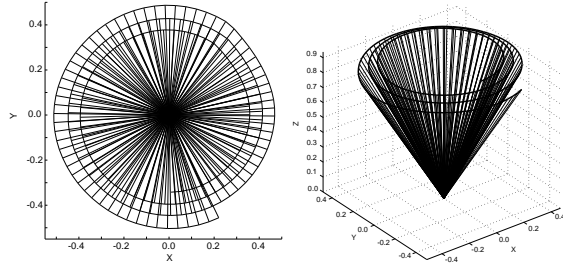
Fig. 2. Position of robot tool tip with open loop virtual fixtures for *autonomous* manipulation: top view (left) and side view (right). The units are normalized for a tool length of 1.
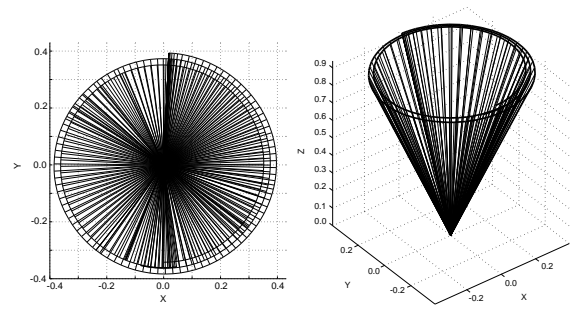

Fig. 3. Position of robot tool tip with open loop virtual fixtures for *cooperative* manipulation: top view (left) and side view (right). The units are normalized for a tool length of 1.

$$D_t = \begin{bmatrix} T_{xyz} & 0 \\ 0 & R_{xyz} \end{bmatrix} \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}', \quad (13)$$

where T_{xyz} and R_{xyz} are 3×3 matrices that describe the translational and rotational components of the inverse kinematics of the robot, respectively. If the robot joint coordinates are

$$\mathbf{q} = \begin{bmatrix} x & y & z & \theta_1 & \theta_2 & \theta_3 \end{bmatrix}', \quad (14)$$

the Steady Hand Robot kinematics yield $T_{xyz} = I_{3\times3}$, since there is no translation, and

$$R_{xyz} = \begin{bmatrix} c(\theta_2) & s(\theta_1)s(\theta_2) & -c(\theta_1)s(\theta_2) \\ 0 & c(\theta_1) & s(\theta_1) \\ s(\theta_2) & -s(\theta_1)c(\theta_2) & c(\theta_1)c(\theta_2) \end{bmatrix}, \quad (15)$$

where $s(\cdot)$ and $c(\cdot)$ denote $\sin(\cdot)$ and $\cos(\cdot)$, respectively. θ_1 and θ_2 can be represented by a parametric function of γ and α ($\theta = f(\alpha, \gamma)$), where γ is a pre-defined initial angle of rotation about the x-axis and α is the initial angle of rotation about the z-axis in Cartesian coordinates. In this case, α has the value between 0 to 2π. The rotation in Cartesian coordinates is represented using $x - y - z$ Euler angles.

A. Open Loop Virtual Fixtures

For open loop virtual fixtures, where the error compensation term in (11) is zero, the preferred direction has the same form as D_t. We implemented the control law by running the robot both manually and autonomously. In the autonomous mode, the input force was computed by simply scaling the preferred direction of motion. The result is to simulate force applied by a "perfect" user intending to follow the path. Figures 2 and 3 show the position of the tool with respect to the center point of motion for the cases of autonomous manipulation and cooperative manipulation, respectively.

It is evident from the figures that without any error servo, the position of the tool tip gradually spirals out from the reference path. The residual error itself results in

a gradual deviation of the tool even without any external parameter such as force provided by a user. Cooperative manipulation offers a more accurate result as shown in Figure 3. The user acts as an additional error servo to lessen the tool deviation.

B. Closed Loop Virtual Fixtures

We now introduce an error compensation term to guide the tool tip toward the reference surface. Let $\mathbf{n}$ denote a 3×1 vector pointing along the preferred rotational motion along the axis of the tool and $\mathbf{z}$ denote a 3×1 vector pointing along the axis of the tool at any point in time, both expressed in the tool frame. However, $\mathbf{z}$ is determined from the values of θ_1 and θ_2. Since these joint coordinates are read from the robot encoders, this will inherently cause some inaccuracy in positioning, as is evident in the open loop case. The control command for closed loop control is

$$\mathbf{u} = \begin{bmatrix} 0 \\ \mathbf{z} \times \mathbf{n} \end{bmatrix} \quad (16)$$

The new preferred direction, $D_g(\mathbf{x})$, can be calculated from (11). The position of the tool for autonomous and cooperative manipulation are shown in Figures 4 and 5, respectively.

It is clear from the figures that the closed loop control offers much more accurate guidance. A major issue in closed loop virtual fixturing is the selection of a proper gain parameter (k_d). In the experiment, we used the values of 0.5 and 0.08 in the control laws for the autonomous and cooperative cases, respectively. The selection of k_d in this experiment was chosen through trial-and-error. When the gain is too low, the tool tip converges to the reference path slowly. In contrast, making it too high results in instability. The range of gains that offers the best guidance is much smaller in the autonomous robot than in the cooperative system. Because the value of k_d with the best performance is much larger for the autonomous manipulation case, the
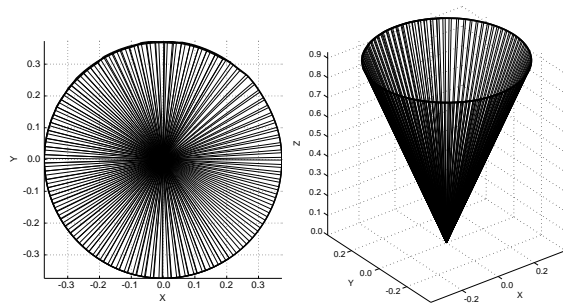
Fig. 4. Position of robot tool tip with closed loop virtual fixtures for *autonomous* manipulation: top view (left) and side view (right). The units are normalized for a tool length of 1.
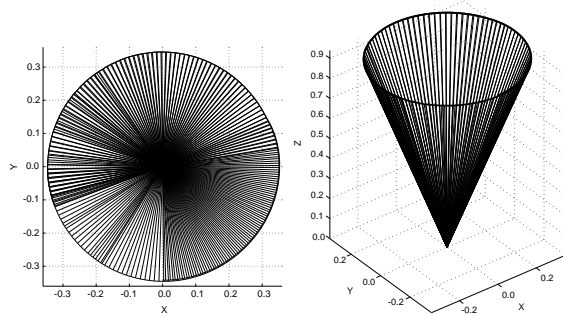

Fig. 5. Position of robot tool tip with closed loop virtual fixtures for *cooperative* manipulation: top view (left) and side view (right). The units are normalized for a tool length of 1.

actions of the user during cooperative manipulation have a significant effect on system performance.

V. TRANSLATIONAL VIRTUAL FIXTURES IMPLEMENTATION

In this section, we demonstrate a control scheme for vision-based virtual fixtures. For a two-dimensional case (following a path in the x-y plane), the preferred direction is expressed as a 6 x 1 time-varying matrix D with terms corresponding to the motion along the x and y axes. The control, **u** is a signed vector describing the error from the current tool tip position to the closest point on the reference path determined by the camera.

In the experiment, visual information was provided by a CCD camera with a lens of 12mm focal length, mounted on the end-effector of the Steady-Hand robot and elevated 11 cm from the plane of the reference path. The experiment used 640×480 pixel images where 1 pixel equals 0.002 mm^2. Only a portion of the path can be seen by the user on the computer monitor, as shown in Figure 1. The XVision system [4] was used to track the tangent to the path at a point closest to the user. The position of the path and the end-effector were displayed in real time at 30Hz during task execution. The path was a sine curve

c_τ	Avg. Time [Sec]	Std. Dev.	Avg. Error [Pixels]	Std. Dev.
0	12.771	0.344	1508.7	211.977
0.3	12.221	0.106	288.090	66.945
0.6	14.091	0.809	288.840	43.267

printed in black on white paper with 35mm amplitude, 70mm wavelength, and line thickness of 0.39mm. The metrics of error and execution time, taken from a start point to an end point on the path, were recorded to evaluate system performance.

The control laws used in these experiments are the same as in the rotational virtual fixture case, although D is now defined by the tangent to the reference path, as determined by the XVision system.

$$D = \begin{bmatrix} t_x & t_y & 0 & 0 & 0 & 0 \end{bmatrix}', \qquad (17)$$

where t_x and t_y are the components of the vector tangent to the reference path, at a point on the path closest to the robot tool, in the x and y directions, respectively. Three compliance values (c_τ in (8)) of 0, 0.3, and 0.6, were used in this case to represent "complete", "hard", and "soft" virtual fixture guidance, respectively. These experiments were performed in cooperative manipulation mode only.

A. Open Loop Virtual Fixtures

For open loop virtual fixtures, the error information provided by the camera is ignored. The average time and average error taken three trials of experiment from an experienced user is shown in Table I.

From the results, it can be seen that compliance variation does not have much effect on the performance improvement without any error compensation. In case of the "complete" guidance, mechanical effects such as the dynamics of the robot eventually deviate the user from the path. Since the virtual fixture allows motion parallel to the reference path, having c_τ equal to zero, in fact, makes it more difficult for the user to return back to the path again. This explains the high error value when $c_\tau=0$ in Table I.

B. Closed Loop Virtual Fixtures

With error compensation to guide the tool tip back to the reference path, the error is reduced significantly, as shown by the experimental results in Table II. In this experiment, the control gain, k_d is fixed at 0.08.

The effect of compliance tuning on error reduction, is significant in this case. With "complete" guidance, error and execution time are kept at the minimum.

c_τ	Avg. Time [Sec]	Std. Dev.	Avg. Error [Pixels]	Std. Dev.
0	12.023	0.069	59.069	4.320
0.3	12.056	0.069	89.774	13.668
0.6	13.035	1.124	176.550	29.669

VI. CONCLUSION

In this paper, we have outlined a broad approach to compliant guidance virtual fixtures, and have applied the algorithms to two specific cases of assistance. Our earlier work suggests that such virtual fixtures can be a useful aid in fine manipulation.

The work reported in this article is the current status of an ongoing effort to develop a broad "library" of sensor-guided and geometric assistance modes. In particular, in our earlier paper [3], we developed a number of vision-based virtual fixturing methods for both one and two camera systems. The second demonstration cited in this article is, in practice, a simple case of these more general developments. The implementation of those methods, for both one and two camera systems, is currently underway.

On the practical side, all of our experiments with virtual fixtures have been within a very specific setup. Numerous issues must be solved before a robust, general implementation of virtual fixtures can be achieved. For example, gain shaping was essential to maintain stability. Similarly, there needs to be careful gain shaping to accommodate the differing scales of forces and torques. More importantly, the ergonomics of this wider class of guidance modes remains to be explored.

On the theoretical side, we have not yet considered two important questions. First, we have suggested a rule for translating a closed loop control algorithm into a virtual fixture, but we do not have a formal proof of conditions under which that rule is valid. Second, we have not yet conducted a thorough stability analysis of these algorithms in order to ground choices such as loop gains.

Finally, it is important to point out that most virtual fixtures apply in a very limited task context. Thus, it is important to consider how to combine guidance modes in parallel (e.g. a force-based guidance mode along a needle axis combined with a vision-based virtual fixture to position the needle and a position-based alignment fixture), and to sequence them.

VII. ACKNOWLEDGMENTS

VIII. REFERENCES

[1] A. Bettini, S. Lang, A.M. Okamura, and G. Hager. Vision assisted control for manipulation using virtual fixtures. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1171–1176, 2001.

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

[3] G.D. Hager. Vision-based motion constraints. *IEEE International Workshop on Intelligent Robots and Systems, Workshop on Visual Servoing*, 2002.

[4] G.D. Hager and K. Toyama. The x-vision system: A general-purpose substrate for real-time vision applications. *Computer Vision Image Understanding*, 69(1):15–33, 1998.

[5] R. Kumar, T.M. Goradia, A. Barnes, P. Jensen, L.M. Auer L.L. Whitcomb, D. Stoianovici, and R.H. Taylor. Performance of robotic augmentation in microsurgery-scale motions. In *Proc. of Medical Image Computing and Computer Assisted Intervention*, volume 1679 of *Lecture Notes in Computer Science*, pages 1108–1115. Springer-Verlag, 1999.

[6] R. Kumar, G.D. Hager, P. Jensen, and R.H. Taylor. An augmentation system for fine manipulation. In *Proc. Medical Image Computing and Computer Assisted Intervention*, pages 956–965. Springer-Verlag, 2000.

[7] R. Kumar, P. Jensen, and R.H. Taylor. Experiments with a steady hand robot in constrained compliant motion and path following. *IEEE RO-MAN*, pages 92–97, 1999.

[8] F. Lai and R.D. Howe. Evaluating control modes for constrained robotic surgery. In *IEEE International Conference on Robotics and Automation*, pages 603–609, 2000.

[9] P. Marayong, A. Bettini, and A.M. Okamura. Effect of virtual fixture compliance on human-machine cooperative manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1089–1095, 2002.

[10] S. Payandeh and Z. Stanisic. On application of virtual fixtures as an aid for telemanipulation and training. *Proc. 10th Symposium On Haptic Interfaces For Virtual Environments and Teleoperator Systems*, pages 18–23, 2002.

[11] M.A. Peshkin, J.E. Colgate, W. Wannasuphoprasit, C.A. Moore, B.R. Gillespie, and P. Akella. Cobot architecture. *IEEE Transactions on Robotics and Automation*, 17(4):377–390, 2001.

[12] L. Rosenberg. Virtual fixtures: perceptual tools for telerobotic manipulation. *Proc. IEEE Virtual Reality International Sympsoium*, pages 76–82, 1993.

[13] G. Strang, editor. *Linear Algebra and its Applications*. Academic Press, New York, 1980.

[14] R. Taylor, P. Jensen, L. Whitcomb, A. Barnes, R. Kumar, D. Stoianovici, P. Gupta, Z.X. Wang, E. deJuan, and L. Kavoussi. Steady-hand robotic system for microsurgical augmentation. *The International Journal of Robotics Research*, 18(12):1201–1210, 1999.

[15] J.N. Weiss. Injection of tissue plasminogen activator into a branch retinal vein in eyes with central retinal vein occlusion. *Opthalmology*, 108(12):2249–2257, 2001.