Navigating Inner Space: 3-D Assistance for Minimally Invasive Surgery

Darius Burschka, Jason J. Corso, Maneesh Dewan, William Lau, Ming Li, Henry Lin, Panadda Marayong, Nicholas Ramey, Gregory D. Hager [1)], Brian Hoffman, David Larkin, and Christopher Hasser.[2)].

[1)]Center for Computer Integrated Surgical Systems and Technology, Johns Hopkins University, Baltimore, MD USA

[2)]Intuitive Surgical, Inc. Sunnyvale, CA, USA

***Abstract*— Since its inception about three decades ago, modern minimally invasive surgery has made huge advances in both technique and technology. However, the minimally invasive surgeon is still faced with daunting challenges in terms of visualization and hand-eye coordination.**

At the Center for Computer Integrated Surgical Systems and Technology (CISST) we have been developing a set of techniques for assisting surgeons in navigating and manipulating the three-dimensional space within the human body. In order to develop such systems, a variety of challenging visual tracking, reconstruction and registration problems must be solved. In addition, this information must be tied to methods for assistance that improve surgical accuracy and reliability but allow the surgeon to retain ultimate control of the procedure and do not prolong time in the operating room.

In this article, we present two problem areas, eye microsurgery and thoracic minimally invasive surgery, where computational vision can play a role. We then describe methods we have developed to process video images for relevant geometric information, and related control algorithms for providing interactive assistance. Finally, we present results from implemented systems.

I. Introduction

Modern minimally invasive surgery developed slowly over the twentieth century until, with the advent of rod optics and fiber-optics (shortly after World War II), the automatic insufflator (in the 1960's), and the first solid state camera (1982), it became a practical reality. Continued advancement in tools and techniques has established minimally invasive surgery as a standard of care in many areas of surgical practice. Today, the same basic concepts of remote visualization and remote manipulation of a physically removed surgical site can be found in diverse areas of surgical practice including, eye, ear, throat, and sinus surgery.

Despite its widespread use, the challenges in minimally invasive surgery are manifold. First and foremost, many procedures are performed using magnified, monocular video[1] images, thereby severely reducing the depth perception and field of view of the surgeon. At the same time, the scale of many such procedures is quite small, making it challenging to perform the required manipulations over the large fulcrum provided by the instruments which are constrained at the insertion point. Finally, the remote dexterity of the surgeon is often severely limited by the structure and design of the instruments themselves. For example, most traditional instruments do not include a remote wrist that would allow reorientation of the end-effector.

At the CISST ERC we have been developing a set of techniques for assisting surgeons in navigating and manipulating the three-dimensional space within the human body. In order to develop such systems, a variety of challenging visual tracking, reconstruction and registration problems must be solved. In addition, this information must be tied to methods for assistance that improve surgeon accuracy and reliability, allow them to retain ultimate control of the procedure, and do not significantly extend time in the operating room.

In this paper, we will review applications of computer vision and vision-based control for micro-surgery of the eye and minimally invasive thoracic surgery. Our emphasis here is to portray the broad spectrum of applications, and, by extension, the rich set of science and engineering problems they present. Much of this material is taken from our recent publications in this area, including [1, 2, 3, 4].

II. Related Work on Intra-Operative Vision and Robotics

One can generally divide the notion of image-guidance into two categories: 1) guidance based on pre-operative

[1]Although stereo imaging systems are now commercially available, they have yet to find widespread use.

images and 2) guidance based on intra-operative imaging. The former is well-established in a variety of marketed systems for surgical navigation (e.g. the StealthStation by Medtronic Inc.) and robotic interventions (e.g. the ROBODOC system by Integrated Surgical Systems). In this article, we will concentrate on the latter, with a focus on the use of video imagery for interactive surgical guidance.

Most prior work on image-guidance has concentrated on ultrasound and fluoroscopy. Abolmaesumi, Salcudean and Zhu have done work on developing a robotic assistant for ultrasound [5]. The principle goal has been to develop visual tracking and servoing techniques that compensate for patient motion in the plane of the ultrasound image. Hong et al. [6] describe a needle insertion instrument which can track a moving object based on visual servo control and tumor specific active contour models. Real-time ultrasonic image segmentation is used to drive the needle.

An algorithm for automatic needle placement using a 6-DOF robot during minimally invasive spine procedures is described by Corral et al. [7]. The placement is achieved with closed-loop position control of the needle by segmenting the needle in the x-ray images to find its position. This is then used as feedback to control the robot and move towards the desired target. Cadaver studies over twenty trials resulted in an average distance error of 1.21mm. Navab et al. [8] describe a method for image-based guidance of a surgical tool during percutaneous procedures. Visual servoing, using projective geometry and projective invariants, is used to provide precise 3D-alignment of the tool with respect to an anatomic target. This approach also estimates the required insertion depth. Experiments with a medical robot inserting a needle into a pig kidney under X-ray fluoroscopy are discussed.

In recent work, a robot vision system that automatically positions a laparoscopic surgical instrument is described by Ginhoux et al. [9]. Laser pointers are fitted on the instrument to emit optical markers on the organ. A visual servoing algorithm is used to position the surgical instrument by combining pixel coordinates of the laser spots and the estimated distance between the organ and the instrument. The system does not require knowledge of the initial perspective position of the endoscope and the instrument. Successful experiments using this system were done on living pigs. Wang and Ueker [10] describe a framework that utilizes intelligent visual modeling, recognition, and servoing methods to assist the surgeon in manipulating a laparoscope. It integrates top-down model guidance, bottom-up image analysis, and surgeon-in-the-loop monitoring. Vision algorithms are used for image analysis, modeling, and matching in a flexible, deformable environment (such as the abdominal cavity). Top-down and bottom-up activities are reconciled by robot servoing mechanisms for executing choreographed scope movements with active vision guidance. This maneuvering allows the surgeon hands-free control of the visual feedback and reduces the risk of inappropriate scope movements. This framework is described and preliminary results are given on laparoscopic image analysis for segmentation and instrument localization. Wei et al. [11] describe a visual tracking method for stereo laparoscopes. The use of color segmentation is used instead of shape analysis to correctly locate a surgical instrument in the image.

Fig. 1
The anatomy of the eye.

III. Two Applications

Our model of a surgical assistant generically includes a stereo video observation system, and a robot manipulated or guided by a surgeon. Here we describe two specific applications of this type.

A. Micro-surgery of the Eye

Age-related macular degeneration (AMD), choroidal neovascularization (CNV), branch retinal vein occlusion (BRVO), and central retinal vein occlusion (CRVO) are among the leading causes of blindness in individuals over the age of 50 [12, 13]. Current treatments involve laser-based techniques such as photodynamic therapy (PDT)

Fig. 2
The typical setup for retinal eye surgery. A surgeon observes the retina using a stereo microscope (looking through the cornea). Tools are inserted through the schlera near the cornea to access the retinal structures at the rear.

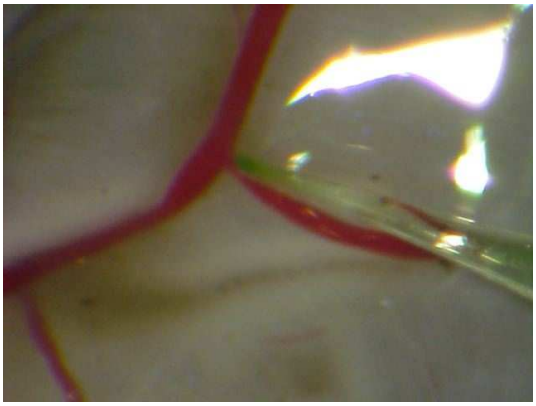

Fig. 3
A 10 μ i.d. glass pipette near a retinal blood vessel as seen through the surgical microscope at high magnification.

and panretinal laser photocoagulation (PRP). However, these treatments sometimes result in a high recurrence rate or complications leading to loss of sight [14, 15]. Recently, alternative approaches employing direct manipulation of surgical tools for local delivery of clot dissolving drug to a retinal vein (vein cannulation) or chemotherapeutic drugs to destroy a tumor have been attempted with promising results (see Figure 1 for a review of eye anatomy). However, these procedures involve manipulation within delicate vitreoretinal structures. 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 [16] 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, see Figure 3). At these scales, tactile feedback is practically non-existent, and depth perception is limited to what can be seen through a stereo surgical microscope. A typical surgical setting is shown in Figure 2.

Our work on micro-surgical assistance is motivated by the JHU Steady-Hand Robot, and, in particular, the assistance paradigm of direct manipulation it was designed for [27, 20, 28]. 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 permitting a means of direct control for the operator. The robot has been designed to provide micron-scale accuracy, and to be ergonomically appropriate for minimally invasive microsurgical tasks [28].

Using the Steady-Hand robot, we have developed two versions of a retinal testbed as shown in Figure 4. A preliminary testbed was first used to validate the basic concepts of vision-guided retinal surgery at a larger scale. More recently, we have developed and demonstrated an integrated visualization and assistance system operating at the micro scale. However, since quantitative performance results for this system are not yet available, results reported here are for the preliminary testbed.

For this testbed, we placed a stereo camera pair (baseline 18cm, 12mm focal length) approximately 0.6m away from the robot. A tool is attached to the end-effector of the robot where the 6 DOF force/torque sensor is located. The user applies force directly to the tool handle.

B. Robotic Minimally Invasive Surgery

Figure 5 shows a typical scene from a laparoscopic surgery (in this case, a right adrenalectomy on a porcine subject). The instruments in this image consist of one laparoscope, two devices that perform retraction and/or cutting or blunt dissection, and one that suctions out blood and other matter. These instruments, which are often a foot or more in length, are introduced through multiple incisions and held by two surgeons. Normally,

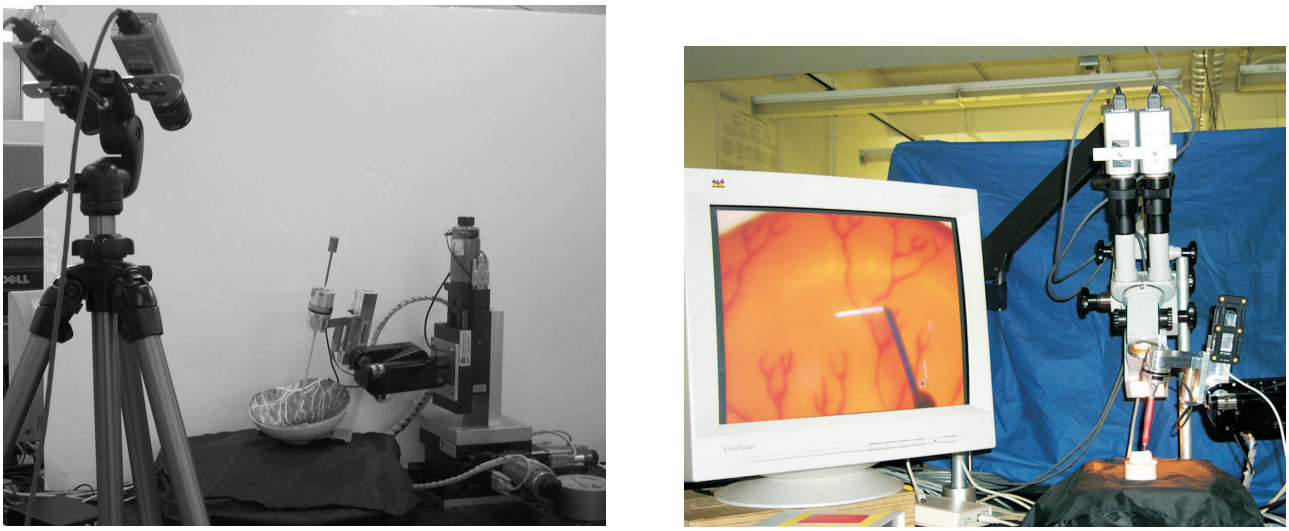

Fig. 4
Left, an initial macro-scale retinal mockup and, right, a more recent retinal workstation.

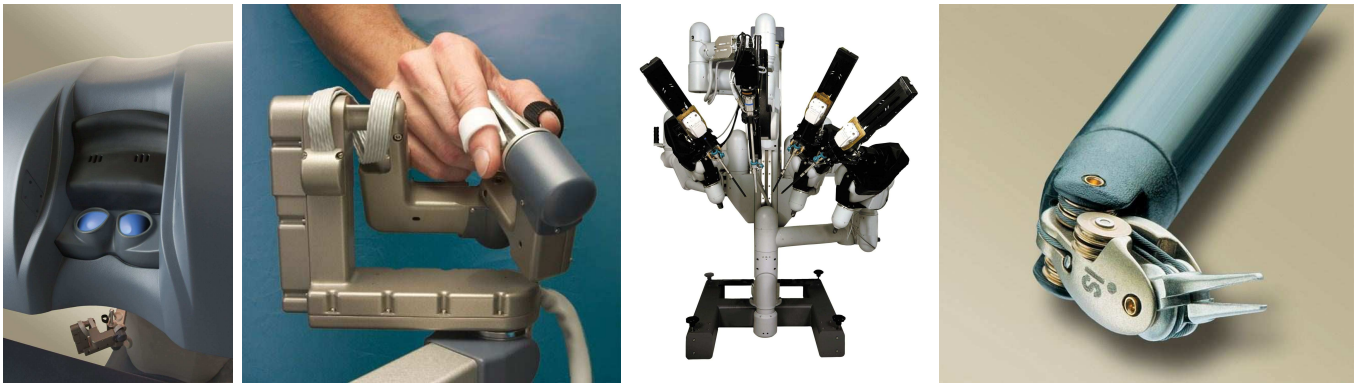

Fig. 6
The Intuitive Surgical da Vinci system. From left to right, the master console, one of the arm masters, the bed-side unit, and a close up of one of the surgical tools.

the instruments are introduced through trocars that seal the opening and permit the abdomen to be insufflated.

From this, it is easy to see the difficulties facing the minimally invasive surgeon. First, he or she is manipulating small structures (often 1 cm or less) at a relatively large distance. The view is a single monocular video sequence looking roughly along the same line as the tools, although with a somewhat arbitrary orientation. The workspace itself is confined, but most of it is not even within view. Finally, the tactile feedback from the instruments is minimal, and is combined with forces applied by the body wall at the trocar.

Robotic surgery, now primarily represented by the da Vinci^{TM} system of Intuitive Surgical, Inc., is one means of overcoming these limitations. The da Vinci, shown in Figure 6, consists of a bedside surgical robot and a master console. The bedside robot manipulates a stereo endoscope and two or three surgical instruments. The master console displays the images to the surgeon and provides two master arms for manipulating the remote tools. An innovative wrist design for the tools, together with an immersive visualization of the surgical site, makes manipulation with the da Vinci "intuitive" by turning minimally invasive surgery back into a more familiar three-dimensional hand-eye coordination problem. Figure 7 shows the left image of a stereo pair taken from the da Vinci during a suturing operation.

While the da Vinci makes surgery more "intuitive," it is important to realize that, as with the eye surgery system, the surgeon is still ultimately relying on their

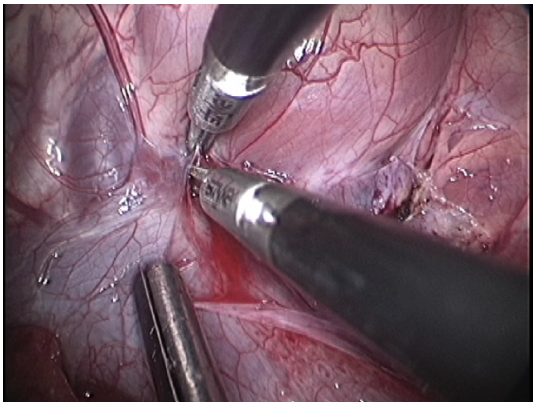

Fig. 5
Conventional minimally invasive surgery during the removal of the adrenal gland from the vena cava. Note both instruments have straight shafts with no distal dexterity and the camera view is highly oblique. The third instrument in view is a suction device.

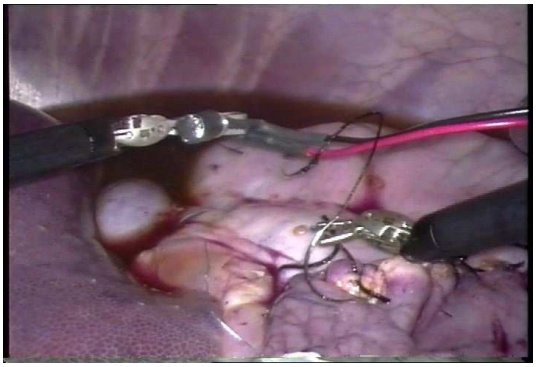

Fig. 7
The da Vinci system during a suturing operation. In comparison to conventional MIS, note the remote wrist dexterity and the more natural camera view.

intrinsic hand-eye coordination, mediated by cameras and mechanisms, to perform the task at hand.

IV. Video-guided Surgical Assistance

In this section, we first introduce the basic admittance control model used to provide interactive guidance to the surgeon. We then extend this control to anisotropic compliances, and finally we relate virtual fixtures to an underlying task geometry. We note that, although defined for an admittance style robot, in practice the same techniques can be applied to impedance style systems (such as the da Vinci master console) based on the methods of [17].

With virtual fixtures in place, we then describe the video processing used to recover the local geometry of the surgical site for both of the applications described above.

Notation: 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

In a recent series of papers [18, 19, 20, 21, 22], our group in the Engineering Research Center for Computer Integrated Surgical Systems (CISST ERC) has been steadily developing systems and related validation methods for cooperative execution of surgical tasks. The basis of these systems is 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 [23, 24, 25, 26]. Guidance virtual fixtures create anisotropic stiffness that promote motion in certain "preferred" directions, while maintaining high stiffness (and thus accuracy) in orthogonal directions.

1) 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 endpoint velocity $\mathbf{v} \in \Re^6$, all expressed in the robot base frame. The robot is guided by applying forces and torques $\mathbf{f} \in \Re^6$ on the manipulator handle, likewise expressed in robot base coordinates.

In the Steady-Hand paradigm, the relationship between velocity and motion is derived by considering a "virtual contact" between the robot tool tip and the environment. In most cases, this contact is modeled by a linear viscous friction law

$$k\mathbf{v} = \mathbf{f} \tag{1}$$

or equivalently

$$\mathbf{v} = \frac{1}{k}\mathbf{f} \tag{2}$$

where $k > 0$ controls the stiffness of the contact. In what follows, it will be more convenient to talk in terms of an admittance gain $c \equiv 1/k$.

When using (2), the effect is that the manipulator is equally compliant in all directions. Suppose we now replace the single constant c with a diagonal matrix C. Making use of C in (2) gives us the freedom to change the admittance of the manipulator in the coordinate directions. For example, setting all but the first two diagonal entries to zero would create a system that permitted motion only in the x-y plane. It is this type of anisotropic gain matrix that we term a virtual fixture. In the case above, the fixture is "hard," meaning it permits

motion in a subspace of the workspace. If we instead set the first two entries to a large value, and the remaining entries to a small one, the fixture becomes "soft." Now, motion in all directions is allowed, but some directions are easier to move in than others. We refer to the motions with high admittance as *preferred* directions, and the remaining directions as *non-preferred* directions.

2) Virtual Fixtures as Geometric Constraints: While it is clearly possible to continue to extend the notion of virtual fixture purely in terms of admittances, we instead prefer to take a more geometric approach, as suggested in [29, 30]. 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

$$\begin{aligned} \text{Span}(D) &\equiv [D] = D(D'D)^{-1}D' && (3) \\ \text{Ker}(D) &\equiv \langle D \rangle = I - [D] && (4) \end{aligned}$$

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 [31, pp. 142–144] and write

$$\begin{aligned} \text{Span}(D) &\equiv [D] = D(D'D)^{+}D' && (5) \\ \text{Ker}(D) &\equiv \langle D \rangle = I - [D] && (6) \end{aligned}$$

The following properties hold for these operators [31]:

1) symmetry: $[D] = [D]'$
2) idempotence: $[D] = [D][D]$
3) scale invariance: $[D] = [kD]$
4) orthogonality: $\langle D \rangle' [D] = 0$
5) completeness: $\text{rank}(\alpha \langle D \rangle + \beta [D]) = n$ where D is $n \times m$ and $\alpha, \beta \neq 0$
6) equivalence of projection: $[\langle D \rangle f] f = \langle D \rangle f$

The above statements remain true if we exchange $\langle D \rangle$ and $[D]$. Finally, it is useful to note the following equivalences:

- $[[D]] = [D]$
- $\langle \langle D \rangle \rangle = [D]$
- $[\langle D \rangle] = \langle [D] \rangle = \langle D \rangle$

Returning to our original problem, consider now decomposing the input force vector, $\mathbf{f}$, into two components

$$\mathbf{f}_D \equiv [D]\mathbf{f} \quad \text{and} \quad \mathbf{f}_\tau \equiv \mathbf{f} - \mathbf{f}_D = \langle D \rangle \mathbf{f} \qquad (7)$$

It follows directly from property 4 that $\mathbf{f}_D \cdot \mathbf{f}_\tau = 0$ and from property 5 that $\mathbf{f}_D + \mathbf{f}_\tau = \mathbf{f}$. Combining (7) and (2), we can now write

$$\mathbf{v} = c\mathbf{f} = c(\mathbf{f}_D + \mathbf{f}_\tau) \qquad (8)$$

Let us now introduce a new admittance gain $c_\tau \in [0,1]$ that attenuates the non-preferred component of the force input. With this we arrive at

$$\begin{aligned} \mathbf{v} &= c(\mathbf{f}_D + c_\tau \mathbf{f}_\tau) \\ &= c([D] + c_\tau \langle D \rangle)\mathbf{f} && (9) \end{aligned}$$

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 admittance 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 gain matrix 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.

3) Virtual Fixtures as Closed Loop Control Laws: Note that, to this point, the notion of virtual fixtures supports motion in a local tangent space of an underlying surface or manifold. Now, we consider how to ensure that the tool tip moves on a *specific* manifold or surface. To study this problem, let us take a simple, yet illustrative case: the case of maintaining the tool tip within a plane through the origin. The surface is defined as $P(\mathbf{p}) = \mathbf{n} \cdot \mathbf{p} = 0$ where $\mathbf{n}$ is a unit vector expressed in robot base coordinates. For simplicity, consider the problem when controlling just the spatial position of the endpoint.

Based on our previous observations, if the goal was to allow motion *parallel* to this plane, then, noting that $\mathbf{n}$ is a non-preferred direction in this case, we would define $D = \langle \mathbf{n} \rangle$ and apply (9). However, if the tool tip is not in the plane, then it is necessary to adjust the preferred direction to move the tool tip toward it. Noting that $P(\mathbf{x})$ is the (signed) distance of $\mathbf{x}$ from the plane, we define a new preferred direction as follows:

$$D_c(\mathbf{x}) = (1 - k_d)\langle \mathbf{n} \rangle \mathbf{f} / \|\mathbf{f}\| + k_d [\mathbf{n}] \mathbf{x}, \quad 0 < k_d < 1. \quad (10)$$

The geometry of (10) is as follows. The idea is to first compute the projection of the applied force onto the nominal set of preferred directions, in this case $\langle \mathbf{n} \rangle$.

At the same time, the location of the tool tip is projected onto the plane normal vector, producing a setpoint error vector. The convex combination of the two vectors yields a resultant vector that will return the tool tip to the plane. Choosing the constant k_d governs how quickly the tool is moved toward the plane. 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 (10) that does not suffer this problem:

$$D_c(\mathbf{x}) = (1-k_d)\langle \mathbf{n} \rangle \mathbf{f} + k_d \|\mathbf{f}\| [\mathbf{n}]\mathbf{x}, \quad 0 < k_d < 1. \quad (11)$$

We now apply (9) with $D = D_c$.

Noting that the second term could also be written

$$\|\mathbf{f}\| k_d P(\mathbf{x})\mathbf{n},$$

it is easy to see that, when the tool tip lies in the plane, the second term vanishes. In this case, it is not hard to show, using the properties of the projection operators, that combining (11) with (9) results in a law equivalent to a pure subspace motion constraint.

With this example in place, it is not hard to see its generalization to a broader set of control laws. We first note that another way of expressing this example would be to posit a control law of the form:

$$\mathbf{u} = -(\mathbf{n} \cdot \mathbf{x})\mathbf{n} = -[\mathbf{n}]\mathbf{x} \quad (12)$$

and to note that assigning $D_c = \mathbf{u}$ would drive the manipulator into the plane. This is, of course, exactly what appears in the second term of (11). This introduces another implementation of virtual fixture control law where the motion is now restricted toward a target pose.

One potential disadvantage of the law described in (11) 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. Another special case is introduced when the force input is very near the null-space of the preferred directions, i.e. $\langle \mathbf{n} \rangle \mathbf{f} \approx 0$. In this case, $D_c(\mathbf{x})$, will be defined in the non-preferred direction. As a result, the user can drive the tool away from the preferred subspace. This is easily avoided by checking the direction of the force input relative to the signed distance $\mathbf{u}$. If the force vector points in the opposite direction, the motion is treated as non-preferred and attenuated by c_τ. A particular case of this is "targeting mode" where the preferred direction is defined as $\mathbf{u}$. The control law in (11) will allow motion both away and toward the target position. The gain switch described above ensures that motion toward the target is preferred to the motion away.

If we now generalize this idea, we can state the following informal rule.

General Virtual Fixture Rule: Given:

1) A surface $S \subseteq SE(3)$ (the motion objective)
2) A signed error vector $\mathbf{u} = f(\mathbf{x}, S)$
3) A rule for computing preferred directions $D = D(t)$ relative to S where $\langle D \rangle \mathbf{u} = 0$ iff $\mathbf{u} = 0$

then applying the following choice of preferred direction:

$$D_g(\mathbf{x}) = (1-k_d)[D]\mathbf{f} + k_d \|\mathbf{f}\| \langle D \rangle \mathbf{u} \quad 0 < k_d < 1. \quad (13)$$

yields a virtual fixture that prefers motion 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 given previously.

To provide a concrete example, consider again the problem of moving the tool tip to a plane through the origin, but let us now add the constraint that the tool z axis should be oriented along the plane normal vector. In this case, $\mathbf{n}$ is a preferred direction of motion (it encodes rotations about the z axis which we don't care about). Let $\mathbf{z}$ denote the vector pointing along the tool z axis and define a control law that is

$$\mathbf{u} = \begin{bmatrix} -(\mathbf{x} \cdot \mathbf{n})\mathbf{n} \\ \mathbf{z} \times \mathbf{n} \end{bmatrix} \quad (14)$$

It is easy to see that this law moves the robot into the plane, and also simultaneously orients the end-effector z axis to be along the normal to the plane. Now, let

$$D = D(t) = \begin{bmatrix} \langle \mathbf{n} \rangle & 0 \\ 0 & \mathbf{n} \end{bmatrix}$$

It follows that $[D]$ is a basis for translation vectors that span the plane, together with rotations about the normal to the plane. Therefore $[D]\mathbf{u} = 0$ since (14) produces translations normal to the plane, and rotations about axes that lie in the plane. Thus, the general virtual fixture rule can be applied.

B. Surface Recovery and Tracking

In order to implement virtual fixtures using video data, it is necessary to compute the position of the surgical instruments and the organ surface. To that end, we have developed a set of techniques for tracking deforming surfaces from stereo video streams. In this development, we assume a calibrated stereo system. Thus, incoming pairs

of images can be rectified to form an equivalent non-verged stereo pair. Let $L(u,v,t)$ and $R(u,v,t)$ denote the left and right rectified image pair at time t, respectively.

The disparity map $\mathcal{D}$ is a mapping from image coordinates to a scalar offset such that $L(u,v,t)$ and $R(u+\mathcal{D}(u,v),v,t)$ are the projection of the same physical point in 3D space. Given disparities, computing the 3-D surface geometry at a point is well-known [32]. In this development, we assume that the disparity map is a parametric function with parameters $\mathbf{p}$ and thus write $\mathcal{D}(\mathbf{p};u,v)$. Thus, the problem of computing disparities is reduced to a parameter estimation problem. For our purposes, it suffices to assume that this disparity map is defined on a given region A of pixel locations in the left image, where A is an enumeration of image locations, $A = \{(u_i,v_i)'\}$, $1 \le i \le N$.

In traditional region-based stereo, correspondences are computed by a search process that locates the maximum of a similarity measure defined on image regions. As we intend to perform a continuous optimization over $\mathbf{p}$, we are interested in analytical similarity measures. Candidate functions include sum of squared differences (SSD), zero-mean SSD (ZSSD), and normalized cross-correlation (NCC) to name a few.

We choose our objective to be ZSSD. In practice, zero-mean comparison measures greatly outperform their non-zero-mean counterparts [33] as they provide a measure of invariance over local brightness variations. If the average is computed using Gaussian weighting, then this difference can be viewed as an approximation to convolving with the Laplacian of a Gaussian. Indeed, such a convolution is often employed with the same goal of achieving local illumination invariance.

Let $\overline{L}(u,v,t) = L(u,v,t) - (L * M)(u,v,t)$ and $\overline{R}(u,v,t) = R(u,v,t) - (R * M)(u,v,t)$ where $*$ denotes convolution and M is an appropriate averaging filter kernel in the spatial-temporal domain. Define $d_i = \mathcal{D}(\mathbf{p};u_i,v_i)$. We can then write our chosen optimization criterion as

$$O(\mathbf{p}) = \sum_{(u_i,v_i)\in A} w_i(\overline{L}(u_i,v_i,t) - \overline{R}(u_i+d_i,v_i,t))^2 \quad (15)$$

where w_i is an optional weighting factor for location $(u_i,v_i)'$.

For compactness of notation, consider A to be fixed and write $\overline{L}(t)$ to denote the $N \times 1$ column vector $(\overline{L}(u_1,v_1,t),\overline{L}(u_2,v_2,t),\ldots\overline{L}(u_N,v_N,t))'$. Likewise, we define $\overline{R}(\mathbf{p},t) = (\overline{R}(u_1+d_1,v_1,t),\ldots\overline{R}(u_N+d_N,v_N,t))'$.

We now adopt the same method as in [34, 35, 36] and expand $\overline{R}(\mathbf{p},t)$ in a Taylor series about a nominal value of $\mathbf{p}$. In this case, we have

$$\begin{aligned} O(\triangle\mathbf{p}) &= \|(\overline{L}(t) - \overline{R}(\mathbf{p}+\triangle\mathbf{p},t))W^{1/2}\|^2 \\ &\approx \|(\overline{L}(t) - \overline{R}(\mathbf{p},t) - J(\mathbf{p},t)\triangle\mathbf{p})W^{1/2}\|^2 \\ &= \|(\mathbf{E}(\mathbf{p},t) - J(\mathbf{p},t)\triangle\mathbf{p})W^{1/2}\|^2 \end{aligned} \quad (16)$$

where $\mathbf{E}(\mathbf{p},t) \equiv \overline{L}(t) - \overline{R}(\mathbf{p},t)$, $J(\mathbf{p},t) = \partial\overline{R}/\partial\mathbf{p}$ is the $N \times n$ Jacobian matrix of $\overline{R}$ considered as a function of $\mathbf{p}$, and $W = \mathrm{diag}(w_1,w_2,\ldots w_N)$. Furthermore, if we define $J_{\mathcal{D}}(\mathbf{p}) = \partial\mathcal{D}/\partial\mathbf{p}$, we have

$$J(\mathbf{p},t) = \mathrm{diag}(\overline{L}_x(t))J_{\mathcal{D}}(\mathbf{p}) \quad (17)$$

where $\overline{L}_x(t)$ is the vector of spatial derivatives of $\overline{L}(t)$ taken along the rows. [2]

It immediately follows that the optimal $\triangle\mathbf{p}$ is the solution to the (over-determined) linear system

$$[J(\mathbf{p},t)^t W J(\mathbf{p},t)]\triangle\mathbf{p} = J(\mathbf{p},t)^t W \mathbf{E}(\mathbf{p},t) \quad (18)$$

In the case that the disparity function is linear in parameters, $J_{\mathcal{D}}$ is a constant matrix and J varies only due to time variation of the gradients on the image surface.

At this point, the complete surface tracking algorithm can now be written as follows:

1) Acquire a pair of stereo images and rectify them.
2) Convolve both images with an averaging filter and subtract the result.
3) Compute spatial x derivatives in the zero-mean left image.
4) Warp the right image by a nominal disparity map (e.g. that computed in the previous step) and subtract from the zero mean left image.
5) Solve (18).

The final two steps may be iterated if desired to achieve higher precision. The entire procedure may also be repeated at multiple scales to improve convergence, if desired. In practice we have not found this to be necessary.

A general example of a linear in parameters model is a B-spline. Consider a set of scanline locations α and row locations β, such that $(\alpha,\beta) \in A$. With m parameters per scanline and n parameters for row locations, a pth by qth degree tensor B-spline is a disparity function of the form

$$\mathcal{D}(\mathbf{p};\alpha,\beta) = \sum_{i=0}^{m}\sum_{j=0}^{n} N_{i,p}(\alpha)\, N_{j,q}(\beta)\, \mathbf{p}_{i,j} \quad (19)$$

[2] Here, we should in fact use the spatial derivatives of the right image after warping or a linear combination of left and right image derivatives. However in practice using just left image derivatives works well and avoids the need to recompute image derivatives if iterative warping is used.

To place this in the framework above, let k denote an indexing linear enumeration of the mn evaluated basis functions, and define $B_{i,k} = N_{k,p}(\alpha_i) * N_{k,q}(\beta_i)$ for all $(\alpha_i, \beta_i) \in A$. It immediately follows that we can create the $N \times mn$ matrix $\mathcal{B}$

$$\mathcal{B} \equiv \begin{bmatrix} B_{1,1}, B_{1,2} .... B_{1,mn} \\ B_{2,1}, B_{2,2} .... B_{2,mn} \\ \vdots \\ B_{N,1}, B_{N,2} .... B_{N,mn} \end{bmatrix}$$

and write

$$\mathcal{D}(\mathbf{p}) = \mathcal{B}\mathbf{p} \tag{20}$$

It follows that the formulation above applies directly with $J_{\mathcal{D}} = \mathcal{B}$. By applying the tracking algorithm to a B-spline disparity surface, we are able to tracking a smooth, deformable surface in real-time. Furthermore this surface is easily queried for the geometrical information necessary to perform vision-based control.

C. Tool Tracking

In addition to tracking the tissue surface, it is necessary to know the position of the surgical tool relative to those surfaces. Although it is in principle possible to compute such relationships using robot kinematic information, we have found that internal kinematics are not generally accurate enough for our purposes. This is particularly true when the surgical tools apply force at the insertion point, or to the tissue surfaces. Here we describe the two different tracking methods used for eye surgery and thoracic surgery.

1) Eye Surgery: In eye surgery, the tool is a very simple mono-colored cylinder. Thus, the tracking algorithm fundamentally involves segmenting the tool and then computing the configuration of the tool from the segmented image as outlined below and as shown in Figure 8. We note this is a straightforward variation on the XVision line tracking algorithm [36].

We maintain the color statistics of the tool in order to segment it from the background. The color statistics of the tool are assumed to follow a Gaussian density model. Initially, the tool is selected from the first frame and the color mean and covariance of both the tool and the background is generated. We use the Mahalanobis distance (r) to segment the tool from the image. A particular pixel is considered part of the tool if the color value at that pixel has $r \leq 3$.

In order to track the tool robustly over long periods of time, it is necessary to update the color statistics of the tool. As the tool is being tracked in subsequent frames,

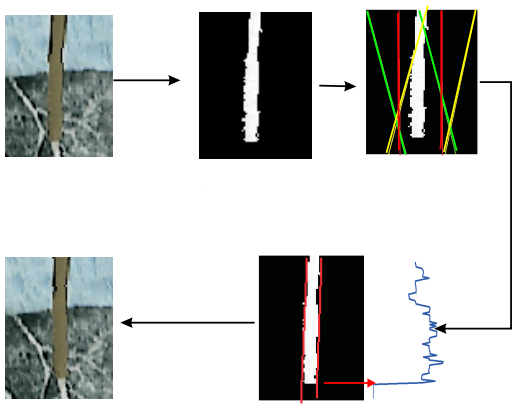

Fig. 8

Initially the warped input image is color segmented. Then the tool center is computed by summing along the y direction. Next, the tool orientation is obtained by interpolating the response at 3 different angles. The tool tip is finally computed by summing along x perpendicular to the tool orientation. The motion parameters are updated by matching with the reference image to obtain the updated warped image.

the pixels associated with the tool are used to update the color mean adaptively based on the following update rule

$$\mu_{t+1} = \alpha\mu_{t-1} + (1-\alpha)\mu_t \tag{21}$$

where α is a design parameter and $(\mu)_t$ is the color mean at time step t. In the current implementation, the covariance matrix is not updated. To decide which pixels belong to the tool, we maintain distributions for both foreground and background pixels. A pixel is deemed to belong to the tool if the log likelihood value for the tool density is higher than that of the background. Only pixels passing the likelihood test are used for updating the mean color value of the tool.

Given prior segmentation, the localization step involves finding the location of the tool tip and the orientation of the tool shaft in both camera images. It is assumed that the tool undergoes primarily image-plane rotation and translation. Thus the tool motion can be represented by three parameters, namely image-plane rotation angle θ and the two components of the translation vector $\mathbf{u}$. At every time step, using the parameters at the last time step, a warped image of fixed size is obtained from the input image. The warped image is then color segmented as described above and the tool is localized in the warped

image with the following three sub-steps:

- The x location of tool center is computed by convolving the warped and segmented image with a box filter at least as wide as the tool width along the x-direction. The resultant image is then summed along the y-direction to obtain a 1D signal, the maximum of which gives the x location of the tool center.
- The tool orientation is obtained from the algorithm similar to the one used in [37] to compute the edge orientation. The filter response is summed along shallow diagonals, which approximates small changes in tool orientation and then interpolating the resulting responses to obtain the maximum response. The result of the interpolation yields the orientation of the tool in the warped image.
- Once the tool orientation and center of the tool in the x direction is known, the tool tip is located by summing the segmented image perpendicular to the computed tool orientation. This results in an obvious step edge in the resulting 1D signal; the location of this step is the end of the tool.

In the initial step a reference template and reference tip location are stored. At every time step, a few iterations of the tool localization are performed. After every iteration, the parameters are updated by matching the warped image and tip location with the reference values. The tip location and orientation are computed independently in the left and right rectified images and if the difference in y location of the tip in both images is less than a certain threshold, the tip is reconstructed to obtain the 3D location. Likewise, the tool orientation is used to obtain a vector describing the tool axis.

The combined information of the tool and the background surface constitute a complete model of the surgical field for this application.

2) Da Vinci Tool Tracking: The da Vinci robot provides a much more challenging tracking problem due to the articulated end-effectors and more dynamic motions. At the same time, the size and kinematic complexity of the robot means that the tool position information supplied by the internal encoders is relatively inaccurate. Further kinematic errors occur when the tool is applying force to a surface. Indeed, one of the reasons for tracking the tools is to provide a accurate image positions for graphical displays which are to follow the tools; such displays are not possible using pure kinematic information.

Our visual tracking system operates directly in the (stereo) endoscopic camera view used by the surgeon. In addition, we make use of the internal API which provides estimated position and velocity of the robot end-effector based on the internal encoder information.

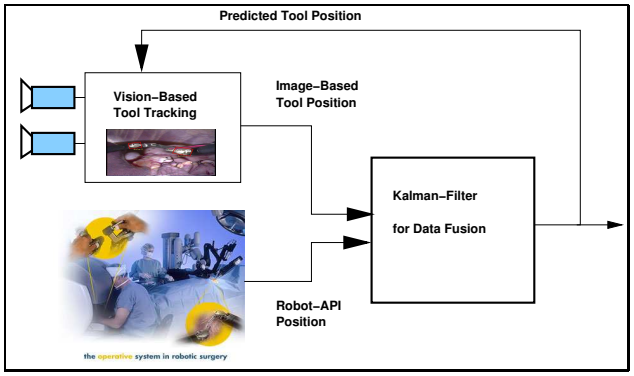

Fig. 9

The kinematics data from the robot and the visual tracker output from the stereo endoscope are fused to obtain reliable information about the current position of the tool.

This information is fused using a Kalman filter as shown in Figure 9.

We have chosen to track the tool directly in the image space of the single images using the stereo constraints between the two views to improve the tracker robustness.

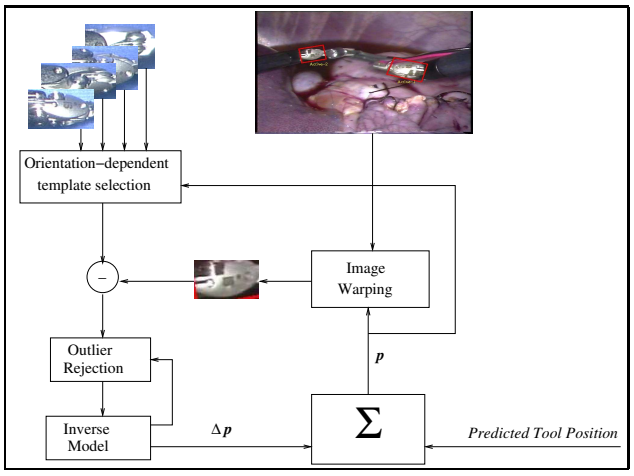

Fig. 10

Data flow in the da Vinci tool tracking system.

At the image level, the tool tracker uses a standard SSD (Sum-of-Square-Distances) approach [36] extended to use multiple templates(Figure 10). A candidate region in the current image is selected and warped based on the current pose estimate provided by the Kalman filter, and then the current pose is updated by template matching. If a sufficient percentage of the pixels ε_{th} are matched in the current view then the tracker estimate is incorporated in the Kalman update step, otherwise just the Kalman prediction is used at the next time step. The value ε_{th} is dependent on the shape and texture of the tool. The

value is estimated for each tool in a test phase that is necessary just once for a new tool design.

The multiple templates model the fact that the tool has different appearances from different views. Each template is keyed to a specific range of tool orientations in the two out of plane rotations (all other appearance changes are be compensated for by image warping). We switch between the different templates depending on the current orientation of the tool predicted by the Kalman Filter module. It is also possible to update the templates over time, thus providing for changes in tool appearance due to stains and tissue deposits on the tool.

The *Outlier Rejection* module depicted in Figure 10 is responsible to dealing with occlusions, minor deposits and stains on the tool and illumination problems. Since the light source is a part of the endoscopic system, we have to deal with significant specularity on the metallic body of the tool which cannot and should not be matched to the template.

V. Experimental Results

The methods described above have been implemented and are a part of our continuing development and testing efforts. Below, we illustrate the results we have achieved to date.

A. Minimally Invasive Surgery Results

At the moment, we do not have access to the da Vinci control system to implement a complete closed loop system with virtual fixtures. However, we have implemented and validated both the tool tracking and surface reconstruction.

In the case of tool tracking, we have developed a graphical overlay system that shows the location and orientation of the tools in the master console of the surgical robot system. Forces measured in the tool shaft can be visualized in the master console to the surgeon as shown in Figure 11.

Additional detailed information about the tool status and orientation can be displayed that is not available or immediately apparent from the image data itself due to occlusions or small scale as shown in Figure 12. The tracking helps to register the tool position correctly in the image plane, while the information from the robot kinematics provides details about the tool state, which may be not visible in the actual image.

The tool tracking operates in real-time (30 frames/s) localizing the reference template in the current camera view. The motion trajectory reported by the robot often has a significant rotational and translational error that does not allow a correct overlay of the annotation data in the image. Figure 13 shows the results from a trajectory in an scenario depicted in the left image. The system switched the reference templates during the operation at the points marked with circles along the brighter trajectory. The solid line depicts the position of the center of the chosen template in the image. The dashed line represents regions during the motion, where the tracking reported an insufficient matching quality between the template data and the actual image data and the corrected kinematics data was reported to the user instead.

The sequence consisted of a fast motion to test the capabilities of the tracking system. The motion was 2-3 times faster than the typical tool motion during surgical procedures. The reader can see that the system could rely

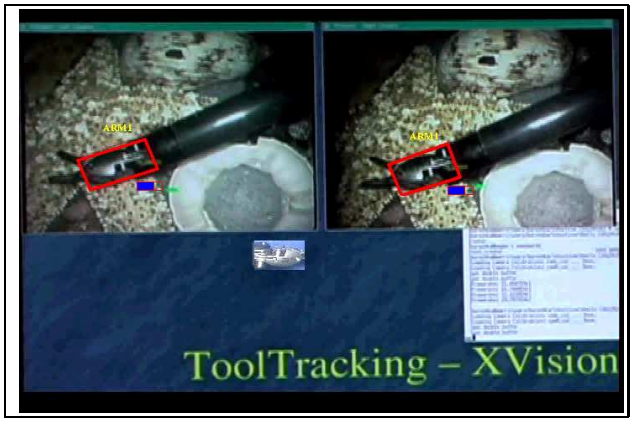

Fig. 11

Template-based tracking result of the daVinci tool: the stereo image and the selected template underneath; additional information about forces and passive and active arms can be displayed.

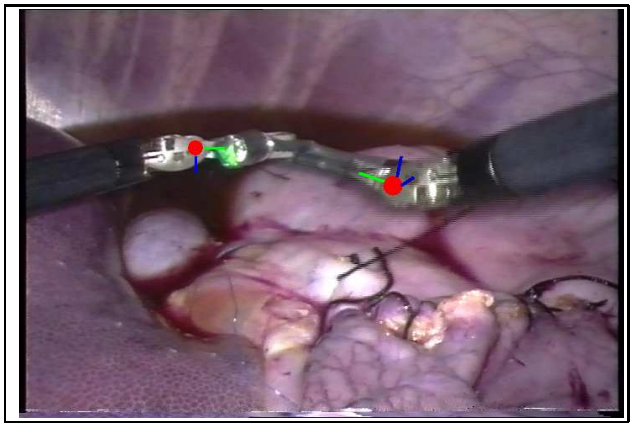

Fig. 12

The tracking result is used for fine registration of the tool position to the camera image. Combination of tracking information with kinematics data allows powerful display possibilities that simplify the tool navigation.

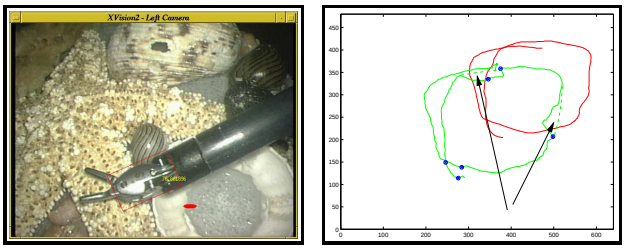

Fig. 13

The re-projection of the tool position based on the kinematics data may have a significant rotational and translational error: (left) tool position reported by the robot is shown as a small ellipse; (right) comparison of the tool trajectory reported by the robot (upper trajectory) to the trajectory reported by tracking (lower trajectory).

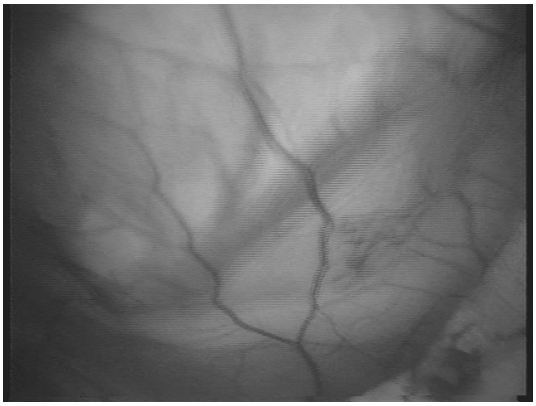

Fig. 14

One image from the stereo sequence used to track the surface of a porcine heart.

on the visual tracking most of the time in this sequence. The tracking only reported incorrect data in two regions, where the specularity on the tool did not leave any sufficient region on the tool to be matched correctly. In this case, the system switched to the corrected kinematics data that became available after the initial registration error got corrected.

This shows the importance of the fusion of the different sources of information. The robot provides reliable data that may have an initial registration error and an error due to forces applied on the tool. These errors can be corrected by the visual tracking. The important fact here is that the tracker can recognize correctly when it gets the wrong result. In our case, the measure for it is the number of correctly matched pixels to the number of pixels in the template, which needs to be above the threshold ε_{th} as we discussed in section IV-C.2.

We have tested the deformable surface tracking on two animal models. In the first, we used an anesthetized Wistar rat. Images of the rat's chest's movement were acquired by a stereo microscope (Zeiss OPMI1-H) mounted with two CCD cameras (SONY XC77). The rat's fur provided a natural texture. An eight second sequence was processed offline by our Matlab implementation. In Figure 15 we graph the respiration (75 breaths per minute) of the rat which was computed by recording a fixed point on the tracked surface.

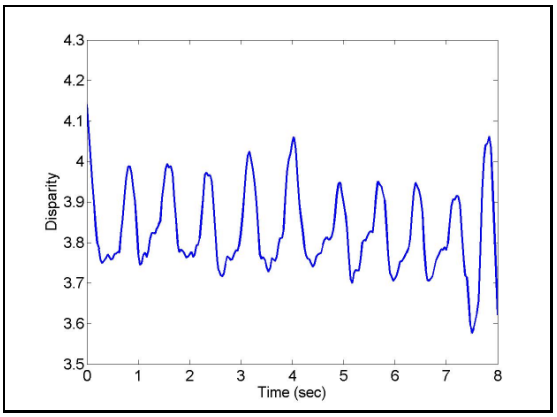

Fig. 15

Recovered disparity values from rat respiration.

In a second experiment, a cross-bred domestic pig (weight, 19.5 kg) was anesthetized with telazol-ketamine-xylazine (TKX, 4.4 mg T/kg, 2.2 mg K/kg, and 2.2 mg X/kg) and mechanically ventilated with a mixture of isoflurane (2%) and oxygen. Heart rate was continuously monitored by a pulse oximeter (SurgiVet, Waukesha, WI). The da Vinci tele-manipulation system (Intuitive Surgical, Sunnyville, CA) was used for endoscopic visualization. Three small incisions were made on the chest to facilitate the insertion of a zero-degree endoscope and other surgical tools. The pericardium was opened and video sequences of the beating heart from the left and right cameras were recorded at 30 frames/sec. The recording lasted approximately two minutes.

The system captured both the beating of the heart and the respiration of the subject (Figure 16). The results are consistent with the other measurements we took during the surgery. In Figure 16, the blue line is a plot of the motion of a fixed point on the surface. The respiration (red-dotted line) is computed using Savitzy-Golay Filtering. For a more detailed discussion of the experiment please see [38].

B. *Eye Testbed Results*

We have validated the preliminary eye testbed on two types of surfaces: a slanted plane and a curved surface.

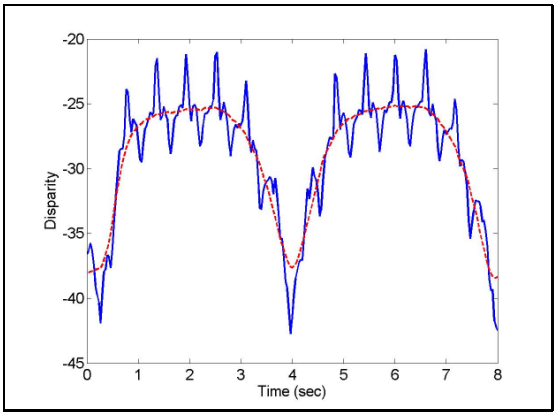

Fig. 16
Graph of pig respiration and heart beat.

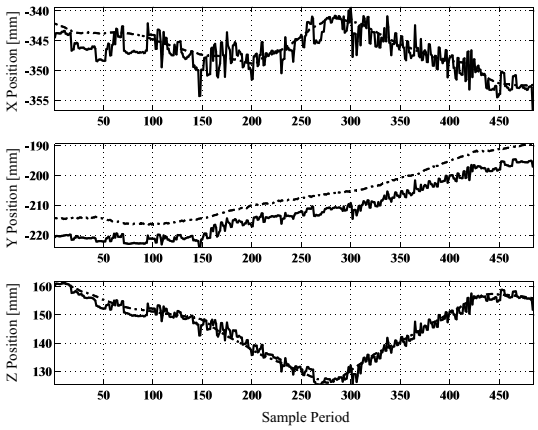

Fig. 17
XYZ positions of the tool tip estimated by the tool tracker (solid line) and estimated by the Optotrak (dash line). Calibration error introduced a constant offset of 5 mm in the transformation as shown in the Y position plot.

The latter is a simulated human retina consisting of a concave surface of approximately 15 cm in diameter covered with an enlarged gray-scale printout of a human retina.

During the manipulation, the user has a direct view of the surface. The user is allowed to change the state of manipulation (Free motion, Tool insertion/extraction, Tool alignment, Targeting, and Surface following) from a GUI. Each state corresponds to a different vision-based virtual fixture. To test the effectiveness of the virtual fixtures, only a hard constraint ($c_\tau = 0$) is implemented. The threshold for the surface tracking task was a constant 3mm offset from the surface. We note this corresponds roughly to one pixel of disparity resolution in the stereo cameras.

Robot-to-camera and stereo calibration is an essential step in the experimental setup. As currently implemented, the resolution and accuracy of the calibration determine the efficacy of the virtual fixtures implementation. The 3D coordinates of the surface and the tool pose are computed in the camera coordinate frame by the visual system. The preferred directions and errors are hence first calculated in the camera frame and then converted to the robot frame where the rest of the virtual fixture control law is implemented to obtain the desired Cartesian tip velocity.

In the preliminary setup, the robot fixed-frame is computed in the Optotrak frame of reference by first rotating the robot about the fixed RCM points in both X and Y axes and then translating it along the three coordinate axes with a rigid body placed at the last stage of the joint. In order to obtain the robot-to-camera transformation, we need the transformation between the Optotrak and the camera is which computed using the Optotrak LED markers rigidly attached to a planar rigid body. Once calibrated, the centers of the LED markers are estimated in the left and right rectified images and then triangulated to compute the 3D positions in the camera coordinate frame. These points are also observed in the Optotrak coordinate system. Finally, using standard least square fitting techniques the rigid body transformation between the Optotrak and the camera is obtained. All transformations are computed with reference to an independent fixed rigid body to avoid any noise from the motion of the Optotrak. As stated earlier, once these transformations are known, the transformation between the robot and the camera can be computed easily.

With the least squares technique used to compute rigid body transformations, the average error for the transformation was approximately 2 ± 1.5mm. We believe that the significant error is due to difficulty of segmenting the center of LED markers in the images. In the more recent testbed, the Optotrak has been eliminated in favor of a direct robot/video calibration method. Although clearly more accurate, we do not yet have quantitative results on this calibration approach.

The accuracy of the tool tracker is validated with the Optotrak. Using the transformation between the Optotrak and the camera already computed, the tracking error can be computed in a straightforward manner. Figure 17 shows the comparison of the XYZ positions obtained by the Optotrak and the tracker. Optotrak data was converted to the camera frame by the corresponding transformation described earlier. Note here that there is a constant offset of approximately 5mm in the Y coordinates. We believe that this constant offset is due to an error in the Camera-Optotrack calibration because of

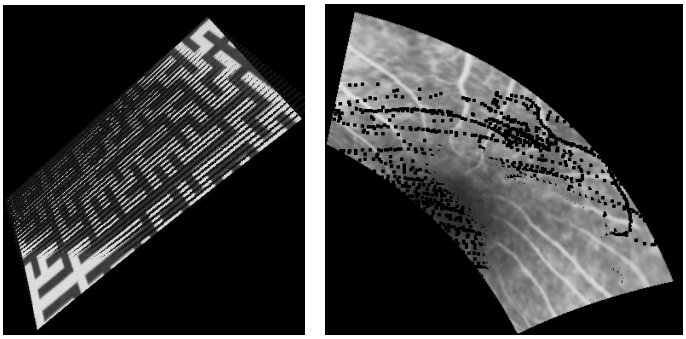

Fig. 18

Textured surface reconstruction overlayed with the traced data (black dots) obtained from the Optotrak. (Left) Slanted plane and (Right) Portion of the eye phantom

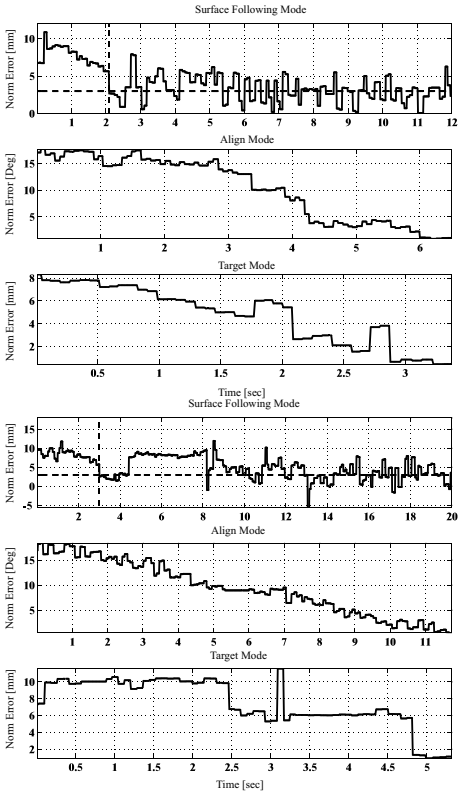

Fig. 19

The magnitude of error between the tool tip position and its intersection on a plane (left) and a concave surface (right).

the difficulty of segmenting LED centers as mentioned earlier. The tool tracking has about 1 pixel error when the offset is accounted for. The surface reconstruction accuracy can be similarly estimated by reconstructing the surface in the Optotrak using a pointed rigid body calibrated at the tip. Figure 18 shows the overlay of the data points obtained from the Optotrak (black dots) on the reconstructed surfaces(with texture). The error in the plane reconstruction (left) is quite low. In the case of the concave surface (right) though, the reconstruction from the camera shows some errors especially toward the boundary where the neighborhood information is sparse. The reconstruction covers an area of approximately 6.5cm x 11cm and 6.5cm x 17cm for the plane and the concave surface, respectively.

Figure 19 shows the magnitude of the error over time of manipulation during surface following, alignment, and targeting tasks with hard virtual fixtures. Note here that the small steps shown on the plot are the result of update rate difference between the cameras and the robot. Sources of noise in the magnitude plot may be due to inaccuracy of the surface reconstruction, error in the estimation of tool tip position, error in calibration, and gain tuning of the virtual fixture control law. In surface following mode, we use 3mm as the desired offset above the actual surfaces. The average magnitude of error was approximately $3 \pm 2mm$. The dashed vertical and horizontal lines indicate the time when the tool reached the surface and the average value (at 3mm), respectively. The decay of the error as the tool approached a specified orientation (Align mode) and target (Target mode) can be clearly seen. In Align and Target modes, the virtual fixture control law allows the movement along the direction of the error vector (forward and backward motion toward the desired orientation/target). Therefore, the user is allowed to move toward or away from the desired pose, as the plots show, in different portions of the targeting task. However, gain tuning is used to minimize the motion in the direction away from the desired pose. Similar results can be seen for the concave surface. We also performed the surface following task in free motion. The performances with virtual fixtures guidance in the surface following and targeting tasks do not show significant improvement over free motion especially with the level of noise and the resolution of the system. However, performance with virtual fixtures guidance naturally surpasses free motion in the alignment task.

The accuracy of the calibration and the limitation of the system are important factors affecting the system performance. As noted previously, with the macro-scale setup, the resolution we can obtain is approximately 3 mm/pix. Table I provides an estimate of the resolution of the current setup (Sony X700) along with an endoscope and a microscope system. Based on our implementation in the current setup and some preliminary experiments, we believe that it is extensible to the high resolution

systems to perform tasks at a 50 micro-scale. In order to accomplish this we will need to operate at higher magnification and/or compute sub-pixel accuracy in tracking and reconstruction. Thus we would like to incorporate fine motion tracking techniques like SSD and kernel-based methods to obtain sub-pixel accuracy for tool tracking.

One of the limitations of the system is the slow update rate of the cameras (10-15Hz) with respect to the robot (100Hz). To deal with this, we would like to explore the possibility of using estimation techniques like Kalman filtering and obtain much smoother tool positions.

VI. Conclusion

When a surgeon is operating on a patient, he or she carries extensive knowledge of human anatomy which is combined with available pre-operative and intra-operative information to perform an intervention. If we are to develop successful assistance methods for computer-integrated surgery systems, they must have similar capabilities.

In this paper, we have briefly discussed some approaches that can be used to acquire three-dimensional geometric information of the surgical field. These models can be used for dexterity and visualization enhancement for surgical assistance. There are several areas for advancement and improvement, many of which we are currently working on. In the case of minimally invasive surgery, our colleagues are developing a new control system around the da Vinci hardware. With this control system, we will be able to implement both active (motion stabilized) and passive (virtual fixture guidance) control algorithms. Indeed, one interesting avenue for future work is the combination of both of these in a single unified framework. Likewise, in eye surgery work, the implementation is under way on the micro-scale testbed, including improvements in tool tracking in order to achieve the necessary levels of precision.

More generally, continued work on robotic assistance for surgery offers the promise of improved safety, reliability and, ultimately, better surgical outcomes. However many problems remain to be solved. First and foremost, the development of provably reliable and safe methods of visual tracking and vision-based structure and motion estimation are essential. One path toward safety and reliability is to incorporate the same anatomical knowledge available to the surgeon into the algorithms. At the same time, fusing geometric information across time and across modality will be essential. During most surgeries, pre-operative information and statistical atlases can provide strong constraints on the vision problem. Finally, we believe significant advances are possible by engineering today's medical instruments to be more appropriate to robotically assisted surgery.

Acknowledgments

The authors would like to acknowledge Intuitive Surgical, Inc. and especially Rajesh Kumar and Giuseppe Prisco from Intuitive Surgical for their support in this research. This material is based upon work supported by the National Science Foundation under Grant Nos. IIS-0099770 and EEC-9731478.

References

[1] G. Hager, "Vision-based motion constraints," *IEEE/RSJ IROS, Workshop on Visual Servoing*, 2002.

[2] D. Burschka, M. Li, R. Taylor, and G. Hager, "Scale-Invariant Registration of Monocular Endoscope Images to CT-Scans For Sinus Surgery," in *Proceedings of Seventh International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2004, (to appear).

[3] D. Burschka and G. D. Hager, "Scale-Invariant Registration of Monocular Stereo Images to 3D Surface Models," in *Proc. of IROS*, 2004, to appear.

[4] N. A. Ramey, J. J. Corso, W. W. Lau, D. Burschka, and G. D. Hager, "Real Time 3D Surface Tracking and Its Applications," in *Proceedings of Workshop on Real-time 3D Sensors and Their Use (at CVPR 2004)*, 2004.

[5] P. Abolmaesumi, S. Salcudean, and W. Zhu, "Visual servoing for robot-assisted diagnostic ultrasound," in *IEEE World Congress on Medical Physics and Biomedical Engineering*, 2000.

[6] J. Hong, M. Hasizume, K. Konishi, and N. Hata, "Ultrasound guided motion adaptive instrument for percutaneous needle insertion therapy," in *6th International Conference on Biomedical Engineering and Rehabilitation Engineering*, 2002, pp. 225–227.

[7] G. Corral, L. Ibanez, C. Nguyen, D. Stoianovici, N. Navab, and K. Cleary, "Robot control by fluroscopic guidance for minimally invasive spine procedures," in *Computer Assisted Radiology and Surgery*, 2004.

[8] N. Navab, B. Bascle, M. Loser, B. Geiger, and R. Taylor, "Visual servoing for automatic and uncalibrated needle placement for percutaneous procedures," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2000, pp. 2327–2334.

[9] R.Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M. M. A. Sanchez, and J. Marescaux, "Beating heart tracking in robotic surgery using 500 hz visual servoing, model predictive control and an adaptive observer," in *Proceedings of the 2004 International Conference on Robotics and Automation*, 2004, pp. 274–279.

[10] Y. F. Wang, D. R. Uecker, and Y. Wang, "A new framework for vision-enabled and robotically-assisted minimally invasive surgery," *Computerized Medical Imaging and Graphics*, vol. 22, no. 6, pp. 429–437, 1998.

[11] G. Wei, K. Arbter, and G. Hirzinger, "Real-time visual servoing for laparoscopic surgery," *IEEE Engineering in Medicine and Biology*, vol. 16, no. 1, pp. 40–45, 1997.

[12] N. Bressler, S. Bressler, and S. Fine, "Age-related macular degeneration," *Surv. Ophthalmol*, vol. 24, no. 9, pp. 375–413, 1998.

[13] I. Scott, "Vitreoretinal surgery for complications of branch retinal vein occlusion," *Curr Opin Ophthalmol*, vol. 13, pp. 161–6, 2002.

TABLE I

System parameters.

Systems	Disparity Range [Pixel]	Resolution [mm/pix]
Sony X700 with 12mm lens	275-300	3.07-2.58
Endoscope (Olympus OTV-3D2)	76-86	0.49-0.389
Microscope (Zeiss OPMI-H)	520-590	0.36-0.27

[14] B. Group, "Argon laser scatter photocoagulation for prevention of neovascularization and vitreous hemorrhange in branch vein occlusion. a randomized clinical trial," *Arch Ophthalmol*, vol. 104, pp. 34–41, 1986.

[15] M. Group, "Argon laser photocoagulation for neovascular maculopathy. 5 yrs results from randomized clinical trial," *Arch Ophthalmol*, vol. 109, pp. 1109–14, 1991.

[16] J. Weiss, "Injection of tissue plasminogen activator into a branch retinal vein in eyes with central retinal vein occlusion," *Opthalmology*, vol. 108, no. 12, pp. 2249–2257, 2001.

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

[18] A. Bettini, S. Lang, A. Okamura, and G. Hager, "Vision assisted control for manipulation using virtual fixtures," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2001, pp. 1171–1176. [Online]. Available: /CIRL/publications/pdf/Bettini-IROS01.pdf

[19] ——, "Vision assisted control for manipulation using virtual fixtures: Experiments at macro and micro scales," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2002, pp. 3354–3361. [Online]. Available: /CIRL/publications/pdf/FixtureICRA02.pdf

[20] R. Kumar, G. Hager, P. Jensen, and R. Taylor, "An augmentation system for fine manipulation," in *MICCAI*. Springer-Verlag, 2000, pp. 956–965.

[21] P. Marayong, M. Li, A. Okamura, and G. Hager, "Spatial motion constraints: Theory and demonstrations for robot guidance using virtual fixtures," *IEEE ICRA*, pp. 1954–1959, 2003.

[22] M. Dewan, P. Marayong, A. Okamura, and G. D. Hager, "Vision-Based Assistance for Ophthalmic Micro-Surgery," in *Proceedings of Seventh International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2004, (to appear).

[23] F. Lai and R. Howe, "Evaluating control modes for constrained robotic surgery," in *IEEE ICRA*, 2000, pp. 603–609.

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

[25] M. Peshkin, J. Colgate, W. Wannasuphoprasit, C. Moore, B. Gillespie, and P. Akella, "Cobot architecture," *IEEE TRA*, vol. 17, no. 4, pp. 377–390, 2001.

[26] L. Rosenberg, "Virtual fixtures: perceptual tools for telerobotic manipulation," *IEEE Virtual Reality International Sympsoium*, pp. 76–82, 1993.

[27] R. Kumar, T. Goradia, A. Barnes, P. Jensen, L. A. L.L. Whitcomb, D. Stoianovici, and R. Taylor, "Performance of robotic augmentation in microsurgery-scale motions," in *MICCAI*, ser. Lecture Notes in Computer Science. Springer-Verlag, 1999, vol. 1679, pp. 1108–1115.

[28] R. Taylor, P. Jensen, L. Whitcomb, A. Barnes, R. Kumar, D. Stoianovici, P. Gupta, Z. Wang, E. deJuan, and L. Kavoussi, "Steady-hand robotic system for microsurgical augmentation," *The International Journal of Robotics Research*, vol. 18, no. 12, pp. 1201–1210, 1999.

[29] A. Bettini, S. Lang, A. Okamura, and G. Hager, "Vision assisted control for manipulation using virtual fixtures," in *IEEE/RSJ IROS*, 2001, pp. 1171–1176.

[30] ——, "Vision assisted control for manipulation using virtual fixtures: Experiments at macro and micro scales," in *IEEE ICRA*, 2002, pp. 3354–3361.

[31] G. Strang, Ed., *Linear Algebra and its Applications*. New York: Academic Press, 1980.

[32] E. Trucco and A. Verri, *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.

[33] J. Banks, M. Bennamoun, K. Kubik, and P. Corke, "Evaluation of new and existing confidence measures for stereo matching," in *Proc. of the Image & Vision Computing NZ conference (IVCNZ98)*, 1998.

[34] B. Lucas and T. Kanade, "An iterative image registratoin technique with an application to stereo vision," in *Proceedings DARPA Image Understanding Workshop*, 1981.

[35] D. Keren, S. Peleg, and R. Brada, "Image sequence enhancement using sub-pixel displacements," in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1988.

[36] G. Hager and P. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, vol. 20, no. 10, pp. 1125–1139, 1998.

[37] G. D. Hager and K. Toyama, "The "XVision" system: A general purpose substrate for real-time vision applications," *CVIU*, vol. 69, no. 1, pp. 23–27, January 1998.

[38] W. W. Lau, N. A. Ramey, J. J. Corso, N. Thakor, and G. D. Hager, "Stereo-Based Endoscopic Tracking of Cardiac Surface Deformation," in *Proceedings of Seventh International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2004, (to appear).