Human-Machine Cooperative Manipulation With Vision-Based Motion Constraints

Gregory D. Hager

Department of Computer Science
Johns Hopkins University
Baltimore, MD 21218

Abstract

This paper discusses a class of control algorithms that provide enhanced physical dexterity by imposing passive motion constraints. Such motion constraints are often referred to as virtual fixtures. It is shown that algorithms originally designed for vision-based control of manipulators can be easily converted into control algorithms that provide virtual fixtures. As a result it is possible to create advanced human-machine cooperative manipulation systems that take complete advantage of information provided by vision, yet permit the user to retain control of essential aspects of a given task.

1 Introduction

Much of "classical" robotics has focused on creating machines that are autonomous. However, such endeavors are fundamentally limited by our ability to create machines that can perceive, judge, and react to unforeseen (or sometimes foreseen!) circumstances in the world. To this day, there are still few situations, other than rote, open-loop manipulation, where robotics have even begun to compete with humans, in this regard.

At the other extreme, teleoperation tends to focus on providing a high-performance, high-fidelity operator interface to a machine. In many ways, this is a natural marriage, as the human now controls a machine that may be more accurate, reliable, or powerful than the human operator. However, while the machine is clearly amplifying some level of human skill, it does so at the cost of attenuating others. In particular, teleoperation systems do not have an underlying representation of the *intent* of the operator. As a result, it is not possible to adapt or modify the interface to enhance those skills that are most germain to the task at hand.

Our group in the Center for Computer Integrated Surgical Systems (CISST) has been working to create surgical systems that improve both the speed and precision of medical interventions. Our goal is to create mechanisms that are neither autonomous, nor purely passive. Rather, our intent is to create mechanisms that selectively provide cooperative assistance to a surgeon, while allowing the surgeon to retain ultimate control of the procedure.

In our recent work, we have focused on developing assistance methods for microsurgery. Here, the extreme challenges of physical scale accentuate the need for dexterity enhancement, but the unstructured nature of the tasks dictates that the human be directly "in the loop." For example, retinal vein cannulation [25] involves the insertion of a needle of approx. 20-50 microns in diameter into the lumen of a retinal vein (typically 100 microns in diameter or less)[1]. 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 [16, 17, 15], we have begun to develop assistant methods that are based on manipulating the apparent compliance of tools simultaneously held by both the 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.

Although they increase absolute precision, purely isotropic compliances cannot take advantage of natural task constraints to provide structured assistance. For example, when placing a needle into the lumen of a blood vessel, the natural mode of assistance would be to stabilize the needle in the lateral directions, but

[1] As a point of reference, a human hair is typically on the order of 80 microns in diameter.

permit relatively free, quasi-static positioning along the needle axis.

In this paper, we specifically focus on the use of *anisotropic* compliances as a means of assistance. In previous work we have related these anisotropic compliances to the notion of virtual fixtures [18, 22]. Virtual fixtures, like the real thing, provide a surface that confines and/or guides motion. We initially describe how virtual fixtures can be produced as a generalization of previous work in [1, 2, 20], and then turn to the problem of deriving vision-based virtual fixtures. Through example, we show how visual servoing algorithms for one camera [3] and two camera [7, 8] systems can be translated into virtual fixtures. As a result, much of the previous literature on visual servoing can be applied to the problem of human-machine cooperative manipulation.

2 Virtual Fixtures

Our work has been motivated by the JHU Steady Hand Robot, and, in particular, the assistance paradigm of direct manipulation it was designed for [15, 16, 24]. 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 to be ergonomically appropriate for minimally invasive microsurgical tasks [24].

In this section, we introduce the basic admittance control model used for the Steady Hand Robot, 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.

2.1 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 $\mathbf{v} = \dot{\mathbf{x}} \in \Re^6$, 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 a compliance $c \equiv 1/k$.

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 compliance 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 compliance 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 compliance as *preferred* directions, and the remaining directions as *non-preferred* directions.

2.2 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

$$\text{Span}(D) \equiv [D] = D(D'D)^{-1}D' \tag{3}$$

$$\text{Ker}(D) \equiv \langle D \rangle = I - [D] \tag{4}$$

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 [23, pp. 142–144] and write

$$\text{Span}(D) \equiv [D] = D(D'D)^{+}D' \tag{5}$$

$$\text{Ker}(D) \quad \equiv \quad \langle D \rangle = I - [D] \tag{6}$$

The following properties hold for these operators [23]:

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

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

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

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

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

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

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.

2.3 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 (9) 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 $\mathbf{v} = \mathbf{u}$,

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

  Then by choosing $D = \mathbf{u}$ and applying (9), we create a virtual fixture that guides the user to the given target pose.

These two tasks are, in some sense, at the extremes of guidance. In one case, there is no specific objective to attain; we are merely constraining motion. In the second, the pose of the manipulator is completely constrained by the objective. What of tasks that fall between these two extremes?

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. For the moment, let us neglect manipulator orientation and consider the problem when controlling just the spatial position of the endpoint. We define the surface as $P(\mathbf{p}) = \mathbf{n} \cdot \mathbf{p} = 0$ where $\mathbf{n}$ is a unit vector expressed in robot base coordinates.

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 from the plane, we define a new preferred direction as follows:

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

The geometry of (10) is as follows. The idea is first produce 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. 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. 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.

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 $\mathbf{v} = \mathbf{u}$ would drive the manipulator into the plane. This is, of course, exactly what appears in the second term of (11). 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 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:

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

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 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 fixturing rule can be applied.

3 Vision-Based Virtual Fixtures

Now, we turn to the problem of providing assistance, where the objective defining the virtual fixture is observed by one or more cameras. To simplify the presentation, in what follows we assume that we have calibrated the camera internal parameters and can therefore work in image normalized coordinates.

3.1 Controlling the Viewer: Pure Translation

Let us start with a well-studied problem. We have a camera fixed to the endpoint of the manipulator, and the camera observes a fixed, static environment. Our goal is to control the motion of the end-effector by defining a motion for the camera itself based on information extracted from the camera image.

To keep things simple, consider first the case of pure translation ($\mathbf{v} \in \Re^3$) where the camera is aligned

with the robot base frame. In this case, the relationship between the motion of the camera and the image motion of a fixed point in space is given by the well-known image Jacobian relationship [12]:

$$\dot{\mathbf{h}} = J\mathbf{v} \tag{15}$$

where $\mathbf{h} = (u, v)' \in \Re^2$ is the image location of a feature point, and J is 2×3.

It is again well-known [3, 12] that the rows of J span the (two-dimensional) space of motions that create feature motion in the image, and therefore $\langle J' \rangle$ is the (one-dimensional) space of motions that leave the point fixed in the image. Consider, thus, creating a virtual fixture by defining

$$D = J' \tag{16}$$

in (9). From the discussion above, it should be clear that this will create a virtual fixture that prefers motion in any direction except along the viewing direction. While it would seem we are done at this point, there is one minor issue: the image Jacobian depends on the depth of the estimated point. However, if we consider the form of the Jacobian in this case, we see it can be written thus:

$$J = \begin{bmatrix} \frac{1}{z} & 0 & \frac{-u}{z} \\ 0 & \frac{1}{z} & \frac{-v}{z} \end{bmatrix} = \frac{1}{z} \begin{bmatrix} 1 & 0 & -u \\ 0 & 1 & -v \end{bmatrix} \tag{17}$$

As such, we see that the term involving z is a scale factor and, as noted earlier, our projection operators are invariant over scaling of their argument. Thus, we have our first result:

Image plane translation: If we restrict $\mathbf{v}$ to be pure translation and choose an image location $\mathbf{h} = (u, v)'$, then implementing (9) using

1. $D = J'$ creates a virtual fixture that prefers motion in the plane normal to the viewing direction defined by $\mathbf{h}$ and the camera optical center
2. $D = \langle J' \rangle$ creates a virtual fixture that prefers motion along the viewing direction defined by $\mathbf{h}$ and the camera optical center.

As a special case, choosing $u = v = 0$ yields a virtual fixture parallel to the image plane, which, as is obvious from (17), is the camera x-y plane.

It is important to note that the image plane virtual fixtures defined above can be implemented both with and without feedback. That is, if we simply choose a fixed image location (e.g. the origin), then the camera will always move in a plane orthogonal to the line of sight through the chosen location. On the other hand, if we choose to track a feature over time, then the motion will always be orthogonal to the line of sight to that feature.

The other possibility, with a single feature, is to maintain the visual cue at a specific image location. For example, suppose the goal is to center an observed point $\mathbf{h}$ in the image. Since our objective is at the origin, we can define a control law of the form

$$\mathbf{u} = -J'\mathbf{h} \tag{18}$$

where J is evaluated at the origin. It is possible to show this law will converge for any feature starting and remaining in the image [14]. Furthermore, the preferred directions of motion in this case are $\langle J' \rangle$ and so it follows that $\langle \langle J' \rangle \rangle \mathbf{u} = -[J']J'\mathbf{h} = 0$ only when $\mathbf{h} = 0$. Since $\mathbf{u} \neq 0$ when $\mathbf{h} \neq 0$, we can apply the general virtual fixturing rule.

At this point, we can state a useful specialization for the rest of this paper:

General Vision-Based Virtual Fixtures Suppose we are supplied with an error term $\mathbf{e} = \mathbf{e}(\mathbf{x})$. Let $S = \{\mathbf{x} | \mathbf{e}(\mathbf{x}) = 0\}$, let $J = \partial \mathbf{e} / \partial \mathbf{x}$, and define $\mathbf{u} = WJ'\mathbf{e}$ where W is a symmetric, positive definite matrix of appropriate dimension (e.g. $W = (J'J)^+$). Then the general virtual fixture rule can be applied with preferred directions $\langle J' \rangle$ provided $\mathbf{u}$ so computed converges to S.

There is an interesting variation on this. Suppose we choose *no* preferred direction of motion (i.e. $D = 0$.) In this case, the first term of (13) disappears and the preferred direction in (9) is simply $\mathbf{u}$. Thus, the result is a virtual fixture that moves the robot to a target position (compare with the rules at the beginning of Section 2.3) and then becomes isotropic. Note, however, that by definition $\mathbf{u}$ is always orthogonal to the line of sight, so the camera prefers to maintain a constant distance to the point during motion.

To press home these points, consider a final problem: to place a specific image location on an observed line, and to facilitate motion along the line. Following the development in [8], suppose we observe a fixed line $\mathbf{l} \in \Re^3$, where the three components of $\mathbf{l}$ can be thought of as the normal vector to the line in the image, and the distance from the origin to the line. This vector is also parallel to the normal vector to the plane formed by the optical axis and the line as it appears in the image plane. We also furnish a distinguished image location $\hat{\mathbf{h}} \in \Re^3$, expressed in homogeneous coordinates. We can then define $\mathbf{e} = \hat{\mathbf{h}} \cdot \mathbf{l}$, to be the image-plane distance between the point and the line.

First, we note that the image Jacobian (relative to $\mathbf{e}$) is now simply

$$L = \mathbf{l}' \begin{bmatrix} J \\ 0 \end{bmatrix} \in \Re^3$$

(note that the z dependence in L is once again a non-issue). As we would now expect, L represents *non-preferred* directions of motion, as it spans the space of motions that change the distance from the point to the line. As a result, choosing preferred directions as $\langle L' \rangle$ in (9) would prefer camera motion within the plane encoded by $\mathbf{l}$.

In order to actually place the designated point onto the line, we note that the control law

$$\mathbf{u} = L'\mathbf{e} \tag{19}$$

will move the feature point $\hat{\mathbf{h}}$ to the observed line [8]. Hence, we apply (13) using $D = \langle L' \rangle$ and $\mathbf{u}$ as defined above — in short, another application of the general vision-based virtual fixture rule.

3.2 Controlling the Viewer: General Case

In moving from pure translation to general robot motion, almost nothing changes from the previous section other than increases in dimensionality. In the case of full motion in $SE(3)$, the image Jacobian becomes 2×6 and has the following general form:

$$\begin{bmatrix} \frac{1}{z} & 0 & \frac{-u}{z} & \frac{-uv}{1} & \frac{1^2+u^2}{1} & -v \\ 0 & \frac{1}{z} & \frac{-v}{z} & \frac{-1^2-v^2}{1} & \frac{uv}{1} & u \end{bmatrix} \tag{20}$$

As shown in [10], the kernel of the image Jacobian given in (20), is spanned by the four vectors

$$\begin{bmatrix} u \\ v \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ 0 \\ 0 \\ u \\ v \\ 1 \end{bmatrix} \quad \begin{bmatrix} uvz \\ -(u^2+1)z \\ vz \\ -1 \\ 0 \\ u \end{bmatrix} \quad \begin{bmatrix} dz \\ 0 \\ -udz \\ uv \\ -(u^2+1)z \\ u \end{bmatrix} \tag{21}$$

where $d = (u^2 + v^2 + 1)$. As such, we can see that the kernel spans motions that include: 1) motion along the line of sight; 2) rotation about the line of sight; 3) motion on a sphere linking the point with the camera; and, linear combinations thereof. Note that 3) spans two degrees of freedom.

If we reconsider all of the cases of the previous section, we see that by using the full Jacobian, we achieve the same virtual fixtures, albeit in a larger space of allowed motions. In particular, choosing $D = \langle J \rangle$ now prefers motions on a sphere about the observed point, together with rotation about, and translation along, the line of sight. It is, however, important to note that the distance from the camera to the observed point no longer "drops out" of the system as in the case of pure translation. Therefore, distance must be estimated, for example by using adaptive schemes as outlined in [21].

At this point, we redirect the reader to [3], where it is observed that regulating the motion of the camera through invariants defined on observed features or measures thereof creates a broad family of "virtual linkages" between the camera and the world. In effect applying the constructions as laid out above to these control laws, creates a corresponding family of virtual fixtures. Likewise, more recent "hybrid" approaches to control that seek to produce more reliable converge of visual servoing methods [19] and/or control other properties of the motion of features in the image plane [4, 5] can be applied, to the extent that the properties outlined in Section 2.3 are satisfied.

3.3 More General Camera Configurations

Until now, we have only considered a single end-effector mounted camera. However, it is important to note that everything we have said above can be applied to the case of a fixed camera observing an independently moving manipulator, with suitable adjustment of the form of the image Jacobian. Furthermore, we note that, for either configuration, observing both the end-effector and the external features defining the task creates an endpoint closed-loop control law which has well-known robustness against camera calibration error [7, 8, 11, 12]. Likewise, methods for estimating the image Jacobian online [13] can, in principle, be applied practically without change.

As a final generalization, we could also add a second observing camera. It is well known [8, 12] that the relationship between control velocities and changes in observed image errors are expressed by "stacking" the individual image Jacobians for each camera, now expressed in a common coordinate system. Furthermore, the estimate of z (depth) in the Jacobian becomes trivial using triangulation from the two cameras. If we return to our list of examples, the following comments apply:

Pure Translation In the case of feature points and pure translation, the stacked Jacobian matrix spans the entire space of robot motions (except for points along the baseline of the two-camera systems), and therefore it is not possible to define interesting

virtual fixtures other than point targeting.

If we consider the case of following a line, however, then when we stack the two Jacobians as before and apply the general vision-based virtual fixture rule, we arrive at a control law that effectively creates a prismatic joint that permits motion strictly along a line in space.

General Motion For general motion, we see that the "stack" of two Jacobians for feature-point servoing creates a spherical joint: the preferred degrees of freedom are motion on a sphere about the observed point while maintaining direction to the point (two degrees of freedom) and rotation about that line of sight (one degree of freedom). This may, at first, seem counter-intuitive since the stacked Jacobian has 4 rows. However, due to the epipolar constraints of the camera, one of these constraints is redundant and thus the Jacobian spans only three degrees of freedom. If we add a second point, we further reduce the degrees of freedom by two, with the remaining allowed motion being rotation about the line defined by the two observed points. A third observed point completely determines the pose of the observed (or observing) system, and so virtual fixturing once again reduces to motion to a target pose.

In the case of placing a point on a line, the image constraints now create a constraint on two degrees of positional freedom. It is, however, still possible to rotate in any direction (with the constraint that the rotation preserved distance to the observed point) and to move along the line. Placing a second point on the line reduces this to two degrees of freedom (rotation about the line and translation along it).

For a complete categorization of image-plane constructions for two-camera systems, we refer the reader to [8, 6].

4 Preliminary Tests

A preliminary version of the algorithms described above were implemented on the JHU Steady Hand Robot system (SHR) [16]. Here, we briefly describe the setup, the results, and its relationship to the more general framework given above. More details can be found in [1].

The robot was equipped with a vision sensor rigidly attached to the force-sensing handle on the end effector. We chose to execute two-dimensional tasks parallel to the image plane, which was in turn arranged to be parallel to two of the base stages of the robot. We performed experiments using a CCD camera at the macro scale and a grin lens endoscope at the micro scale. The vision sensor always viewed

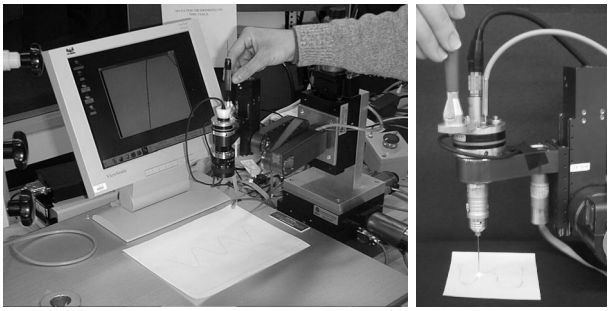

Figure 1: *The experimental setup of the steady hand robot using virtual fixtures to assist in path following and positioning tasks: left, macro scale; right, micro scale.*

the task plane, allowing reading of the motion references and real-time display of task execution (Figure 1). On-screen display of the task execution is useful for operators at the macro scale, and essential at the micro scale, as it would be impossible to complete the task using the naked eye.

The path was furnished to both the system and the user by printing a sine curve (35mm amplitude, 70mm wavelength, and 0.54mm width) on the task plane (in black on white paper). At micro scale, it was not possible to print a sufficiently smooth curve, so we instead embedded a wavy human hair (about 80 μm diameter) in glue on a yellow sheet of paper. In the macro case, the camera was positioned 200mm from the paper, yielding a pixel footprint of 0.066mm on the working surface. In the micro case, the endoscope was about 150 μm above the working surface, yielding a pixel footprint of about 1 μm (Figure 2).

The center of the image was graphically marked, and users were instructed to perform path following tasks relative to this mark. The sensor on the handle was used to record user commands. The force sensor resolution is $12.5mN$ and force values are expressed as multiples of this base unit.

Visual tracking (XVision system [9]) was used to measure (in real time) the local position and tangent direction, $\mathbf{d}$, to the path. Subpixel interpolation was used to increase the precision of these measurements. The vision and control subsystems executed on two different PCs, and the data exchange was realized over a local network. The control system operated at 100 Hz, using the most recent available data from the vision system and handle force sensor.

In terms of our previous formulation, the marked image location at the image center means that $\mathbf{x} = 0$ Further, the workspace is a plane, ($\mathbf{x} \in \Re^2$). The preferred direction is given by the tangent measurements from the tracking algorithm. Implicitly, the

control law used to position the manipulator on the line is

$$\mathbf{u} = \mathbf{s} - \mathbf{x} = \mathbf{s} \tag{22}$$

where $\mathbf{s} \in \Re^2$ is the current location of the visual tracker in the image. Further, $\mathbf{s}$ was constrained to lie along the line through the marked image location, normal to $\mathbf{d}$. Choosing $\mathbf{d}$ as the preferred direction of motion, we see the conditions of the general virtual fixturing rule are satisfied.

This class of virtual fixtures has been tested at both macro and micro scales. Results for a specific user and a wide class of compliances and situations can be found in [1, 2]. Tests for a larger class of users can be found in [20].

5 Conclusion

In this paper, we have outlined a broad theory of compliant virtual fixtures, and have applied that theory to the specific case of vision-guided assistance. Our earlier work suggests that such virtual fixtures can be a useful aid to dextrous manipulation.

In many ways, this paper is intended to point toward interesting further directions to be explored. First and foremost, we have begun to develop a general means for translating control algorithms into virtual fixtures. However, the treatment to this point has not been sufficiently formal to determine when such a translation is possible. In particular, we have not described how to exhibit a set of preferred directions that are consistent with a control input. Further, we have not offered a formal definition of guidance with virtual fixtures that would permit a general theoretical statement of an equivalence between active control and passive virtual fixturing. These remain interesting open problems.

On the practical side all of our experiments with vision-guided virtual fixtures have been within a very specific setup. Numerous issues must be solved before a robust, general implementation of vision-guided virtual fixtures can be achieved. For example, in our previous work, 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.

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. Initial work on the latter problem will be reported in a forthcoming paper.

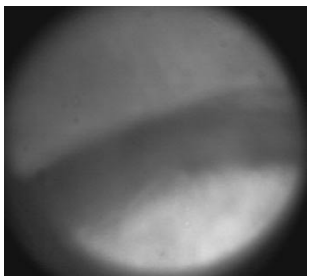

Figure 2: *Endoscope image of the 80 μm-diameter human hair used as the path in micro scale experiments.*

Acknowledgments

We would like to acknowledge the contributions of Pannada Marayong and Allison Okamura to this work. This material is based upon work supported by the National Science Foundation under Grant Nos. IIS-0099770 and EEC-9731478. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

References

[1] 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*, pages 1171–1176, 2001.

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

[3] F. Chaumette, P. Rives, and B. Espiau. Classification and realization of the different vision-based tasks. In K. Hashimoto, editor, *Visual Servoing*, pages 199–228. World Scientific, 1994.

[4] P. Corke and S. Hutchinson. A new partitioned approach to image-based visual servo control. *IEEE Trans. Robot. Autom.*, 17(4):507–515, 2001.

[5] N.J. Cowan, J. Weingarten, and D.E. Koditschek. Vision-based control via navigation functions. *IEEE Trans. Robot. Autom.*, 2002. To Appear.

[6] Z. Dodds. *Task Specification Languages for Uncalibrated Visual Servoing.* PhD thesis, Yale University, 2000.

[7] Z. Dodds, G. Hager, A.S. Morse, and J.P. Hespanha. Task specification and monitoring for uncalibrated hand/eye coordination. In *Proc. IEEE Int. Conference Rob. Automat.*, pages 1607–1613, May 1999.

[8] G. D. Hager. A modular system for robust hand-eye coordination. *IEEE Trans. Robot. Automat*, 13(4):582–595, 1997.

[9] G. D. Hager and K. Toyama. The "XVision" system: A general purpose substrate for real-time vision applications. *Comp. Vision, Image Understanding.*, 69(1):23–27, January 1998.

[10] R. M. Haralick and L. G. Shapiro. *Computer and Robot Vision*. Addison Wesley, 1993.

[11] J. Hespanha, Z. Dodds, G. D. Hager, and A. S. Morse. What tasks can be performed with an uncalibrated stereo vision system? *International Journal of Computer Vision*, 35(1):65–85, 1999.

[12] S. Hutchinson, G. D. Hager, and P. Corke. A tutorial introduction to visual servo control. *IEEE Trans. Robot. Automat*, 12(5):651–670, 1996.

[13] M. Jägersand. *On-line Estimation of Visual-Motor Models for Robot Control and Visual Simulation*. PhD thesis, University of Rochester, 1997.

[14] D. Kim, A. Rizzi, G. D. Hager, and D. Koditschek. A "robust" convergent visual servoing system. In *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume I, pages 348–353. IEEE Computer Society Press, 1995.

[15] 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 *Proceedings of Medical Image Computing and Computer Assisted Intervention*, volume 1679 of *Lecture Notes in Computer Science*, pages 1108–1115. Springer-Verlag, 1999.

[16] 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.

[17] 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.

[18] 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.

[19] E. Malis, F. Chaumette, and S. Boudet. 2 1/2 D visual servoing. *IEEE Trans. Robot. Autom.*, 15(2):238–250, 1999.

[20] 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*, page To Appear, 2001.

[21] N. P. Papanikolopoulos and P. K. Khosla. Adaptive Robot Visual Tracking: Theory and Experiments. *IEEE Trans. on Automatic Control*, 38(3):429–445, 1993.

[22] L. Rosenberg. *Virtual Fixtures*. PhD thesis, Dept. of Mechanical Engineering, Stanford University, 1994.

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

[24] 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.

[25] 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.