Visual servo control, Part I: Basic approaches

François Chaumette, S. Hutchinson

TUTORIAL

Visual Servo Control Part I: Basic Approaches

BY FRANÇOIS CHAUMETTE AND SETH HUTCHINSON

This article is the first of a two-part series on the topic of visual servo control—using computer vision data in the servo loop to control the motion of a robot. In the present article, we describe the basic techniques that are by now well established in the field. We first give a general overview of the formulation of the visual servo control problem. We then describe the two archetypal visual servo control schemes: image-based and position-based visual servo control. Finally, we discuss performance and stability issues that pertain to these two schemes, motivating the second article in the series, in which we consider advanced techniques.

Introduction

Visual servo control refers to the use of computer vision data to control the motion of a robot. The vision data may be acquired from a camera that is mounted directly on a robot manipulator or on a mobile robot, in which case motion of the robot induces camera motion, or the camera can be fixed in the workspace so that it can observe the robot motion from a stationary configuration. Other configurations can be considered such as, for instance, several cameras mounted on pan-tilt heads observing the robot motion. The mathematical development of all these cases is similar, and in this tutorial we will focus primarily on the former, so-called eye-in-hand, case.

Visual servo control relies on techniques from image processing, computer vision, and control theory. Since it is not possible to cover all of these in depth in a single article, we will focus here primarily on issues related to control, and to those specific geometric aspects of computer vision that are uniquely relevant to the study of visual servo control. We will not specifically address issues related to feature tracking or three-dimensional (3-D) pose estimation, both of which are topics deserving of their own tutorials.

The Basic Components of Visual Servoing

The aim of all vision-based control schemes is to minimize an error $\mathbf{e}(t)$, which is typically defined by

$$\mathbf{e}(t) = \mathbf{s}(\mathbf{m}(t), \mathbf{a}) - \mathbf{s}^*. \tag{1}$$

This formulation is quite general, and it encompasses a wide variety of approaches, as we will see below. The parameters in (1) are defined as follows. The vector $\mathbf{m}(t)$ is a set of image measurements (e.g., the image coordinates of interest points or the image coordinates of the centroid of an object). These image measurements are used to compute a vector of k visual features, $\mathbf{s}(\mathbf{m}(t), \mathbf{a})$, in which $\mathbf{a}$ is a set of parameters that represent potential additional knowledge about the system (e.g., coarse camera intrinsic parameters or 3-D models of objects). The vector $\mathbf{s}^*$ contains the desired values of the features.

In Part I of the tutorial (this article), we consider the case of a fixed goal pose and a motionless target, i.e., $\mathbf{s}^*$ is constant, and changes in $\mathbf{s}$ depend only on camera motion. Further, we consider here the case of controlling the motion of a camera with six degrees of freedom (6 DOF); e.g., a camera attached to the end effector of a six degree-of-freedom arm. We will treat more general cases in Part II of the tutorial.

Visual servoing schemes mainly differ in the way that $\mathbf{s}$ is designed. In this article, we will see two very different approaches. First, we describe image-based visual servo control (IBVS), in which $\mathbf{s}$ consists of a set of features that are immediately available in the image data. Then, we describe position-based visual servo control (PBVS), in which $\mathbf{s}$ consists of a set of 3-D parameters, which must be estimated from image measurements.

Once $\mathbf{s}$ is selected, the design of the control scheme can be quite simple. Perhaps the most straightforward approach is to design a velocity controller. To do this, we require the relationship between the time variation of $\mathbf{s}$ and the camera velocity. Let the spatial velocity of the camera be denoted by $\mathbf{v}_c = (v_c, \boldsymbol{\omega}_c)$, with v_c the instantaneous linear velocity of the origin of the camera frame and $\boldsymbol{\omega}_c$ the instantaneous angular velocity of the camera frame The relationship between $\dot{\mathbf{s}}$ and $\mathbf{v}_c$ is given by

$$\dot{\mathbf{s}} = \mathbf{L}_\mathbf{s}\mathbf{v}_c, \tag{2}$$

in which $\mathbf{L}_\mathbf{s} \in \mathbb{R}^{k\times 6}$ is named the *interaction matrix* related to $\mathbf{s}$. The term *feature Jacobian* is also used somewhat interchangeably in the visual servo literature.

Using (1) and (2), we immediately obtain the relationship between camera velocity and the time variation of the error:

$$\dot{\mathbf{e}} = \mathbf{L}_\mathbf{e}\mathbf{v}_c, \tag{3}$$

where $\mathbf{L}_\mathbf{e} = \mathbf{L}_\mathbf{s}$. Considering $\mathbf{v}_c$ as the input to the robot controller, and if we would like for instance to try to ensure

1070-9932/06/$20.00©2006 IEEE

an exponential decoupled decrease of the error (i.e., $\dot{\mathbf{e}} = -\lambda \mathbf{e}$), we obtain using (3):

$$\mathbf{v}_c = -\lambda \mathbf{L}_\mathbf{e}^+ \mathbf{e}, \tag{4}$$

where $\mathbf{L}_\mathbf{e}^+ \in \mathbb{R}^{6\times k}$ is chosen as the Moore-Penrose pseudo-inverse of $\mathbf{L}_\mathbf{e}$, that is $\mathbf{L}_\mathbf{e}^+ = (\mathbf{L}_\mathbf{e}{}^\top \mathbf{L}_\mathbf{e})^{-1} \mathbf{L}_\mathbf{e}{}^\top$ when $\mathbf{L}_\mathbf{e}$ is of full rank 6. This choice allows $\|\dot{\mathbf{e}} - \lambda \mathbf{L}_\mathbf{e} \mathbf{L}_\mathbf{e}^+ \mathbf{e}\|$ and $\|\mathbf{v}_c\|$ to be minimal. When $k = 6$, if $\det \mathbf{L}_\mathbf{e} \neq 0$ it it possible to invert $\mathbf{L}_\mathbf{e}$, giving the control $\mathbf{v}_c = -\lambda \mathbf{L}_\mathbf{e}{}^{-1} \mathbf{e}$.

In real visual servo systems, it is impossible to know perfectly in practice either $\mathbf{L}_\mathbf{e}$ or $\mathbf{L}_\mathbf{e}^+$. So an approximation or an estimation of one of these two matrices must be realized. In the sequel, we denote both the pseudoinverse of the approximation of the interaction matrix and the approximation of the pseudoinverse of the interaction matrix by the symbol $\widehat{\mathbf{L}_\mathbf{e}^+}$. Using this notation, the control law is in fact:

$$\mathbf{v}_c = -\lambda \widehat{\mathbf{L}_\mathbf{e}^+} \mathbf{e}. \tag{5}$$

This is the basic design implemented by most visual servo controllers. All that remains is to fill in the details: How should $\mathbf{s}$ be chosen? What then is the form of $\mathbf{L}_\mathbf{s}$? How should we estimate $\widehat{\mathbf{L}_\mathbf{e}^+}$? What are the performance characteristics of the resulting closed-loop system? These questions are addressed in the remainder of the article.

Classical Image-Based Visual Servo

Traditional image-based control schemes [1], [2] use the image-plane coordinates of a set of points (other choices are possible, but we defer discussion of these for Part II of the tutorial) to define the set $\mathbf{s}$. The image measurements $\mathbf{m}$ are usually the pixel coordinates of the set of image points (but this is not the only possible choice), and the parameters $\mathbf{a}$ in the definition of $\mathbf{s} = \mathbf{s}(\mathbf{m}, \mathbf{a})$ in (1) are nothing but the camera intrinsic parameters to go from image measurements expressed in pixels to the features.

The Interaction Matrix

More precisely, for a 3-D point with coordinates $\mathbf{X} = (X, Y, Z)$ in the camera frame, which projects in the image as a 2-D point with coordinates $\mathbf{x} = (x, y)$, we have:

$$\begin{cases} x &= X/Z = (u - c_u)/f\alpha \\ y &= Y/Z = (v - c_v)/f, \end{cases} \tag{6}$$

where $\mathbf{m} = (u, v)$ gives the coordinates of the image point expressed in pixel units, and $\mathbf{a} = (c_u, c_v, f, \alpha)$ is the set of camera intrinsic parameters: c_u and c_v are the coordinates of the principal point, f is the focal length, and α is the ratio of the pixel dimensions. In this case, we take $\mathbf{s} = \mathbf{x} = (x, y)$, the image plane coordinates of the point. The details of imaging geometry and perspective projection can be found in many computer vision texts, including [3], [4].

Taking the time derivative of the projection equations (6), we obtain

$$\begin{cases} \dot{x} &= \dot{X}/Z - X\dot{Z}/Z^2 &= (\dot{X} - x\dot{Z})/Z \\ \dot{y} &= \dot{Y}/Z - Y\dot{Z}/Z^2 &= (\dot{Y} - y\dot{Z})/Z. \end{cases} \tag{7}$$

We can relate the velocity of the 3-D point to the camera spatial velocity using the well-known equation

$$\dot{\mathbf{X}} = -\boldsymbol{v}_c - \omega_c \times \mathbf{X} \Leftrightarrow \begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X. \end{cases} \tag{8}$$

Injecting (8) in (7), and grouping terms we obtain

$$\begin{cases} \dot{x} = -v_x/Z + xv_z/Z + xy\omega_x - (1 + x^2)\omega_y + y\omega_z \\ \dot{y} = -v_y/Z + yv_z/Z + (1 + y^2)\omega_x - xy\omega_y - x\omega_z \end{cases} \tag{9}$$

which can be written

$$\dot{\mathbf{x}} = \mathbf{L}_\mathbf{x} \mathbf{v}_c, \tag{10}$$

where the interaction matrix $\mathbf{L}_\mathbf{x}$ related to $\mathbf{x}$ is

$$\mathbf{L}_\mathbf{x} = \begin{bmatrix} \frac{-1}{Z} & 0 & \frac{x}{Z} & xy & -(1 + x^2) & y \\ 0 & \frac{-1}{Z} & \frac{y}{Z} & 1 + y^2 & -xy & -x \end{bmatrix}. \tag{11}$$

In the matrix $\mathbf{L}_\mathbf{x}$, the value Z is the depth of the point relative to the camera frame. Therefore, any control scheme that uses this form of the interaction matrix must estimate or approximate the value of Z. Similarly, the camera intrinsic parameters are involved in the computation of x and y. Thus, $\mathbf{L}_\mathbf{x}$ cannot be *directly* used in (4), and an estimation or an approximation $\widehat{\mathbf{L}_\mathbf{x}}$ must be used. We discuss this in more detail below.

To control the 6 DOF, at least three points are necessary (i.e., we require $k \geq 6$). If we use the feature vector $\mathbf{x} = (\mathbf{x_1}, \mathbf{x_2}, \mathbf{x_3})$, by merely stacking interaction matrices for three points we obtain

$$\mathbf{L}_\mathbf{x} = \begin{bmatrix} \mathbf{L}_{\mathbf{x}_1} \\ \mathbf{L}_{\mathbf{x}_2} \\ \mathbf{L}_{\mathbf{x}_3} \end{bmatrix}.$$

In this case, there will exist some configurations for which $\mathbf{L}_\mathbf{x}$ is singular [5]. Furthermore, there exist four distinct camera poses for which $\mathbf{e} = 0$, i.e., four global minima exist, and it is impossible to differentiate them [6]. For these reasons, more than three points are usually considered.

Approximating the Interaction Matrix

There are several choices available for constructing the estimate $\widehat{\mathbf{L}_\mathbf{e}^+}$ to be used in the control law. One popular scheme is, of course, to choose $\widehat{\mathbf{L}_\mathbf{e}^+} = \mathbf{L}_\mathbf{e}^+$ if $\mathbf{L}_\mathbf{e} = \mathbf{L}_\mathbf{x}$ is known; that is, if the current depth Z of each point is available [7]. In practice, these parameters must be estimated at each iteration of the control scheme. The basic methods presented in this article use classical pose estimation methods that will be briefly presented in the next section. Another popular

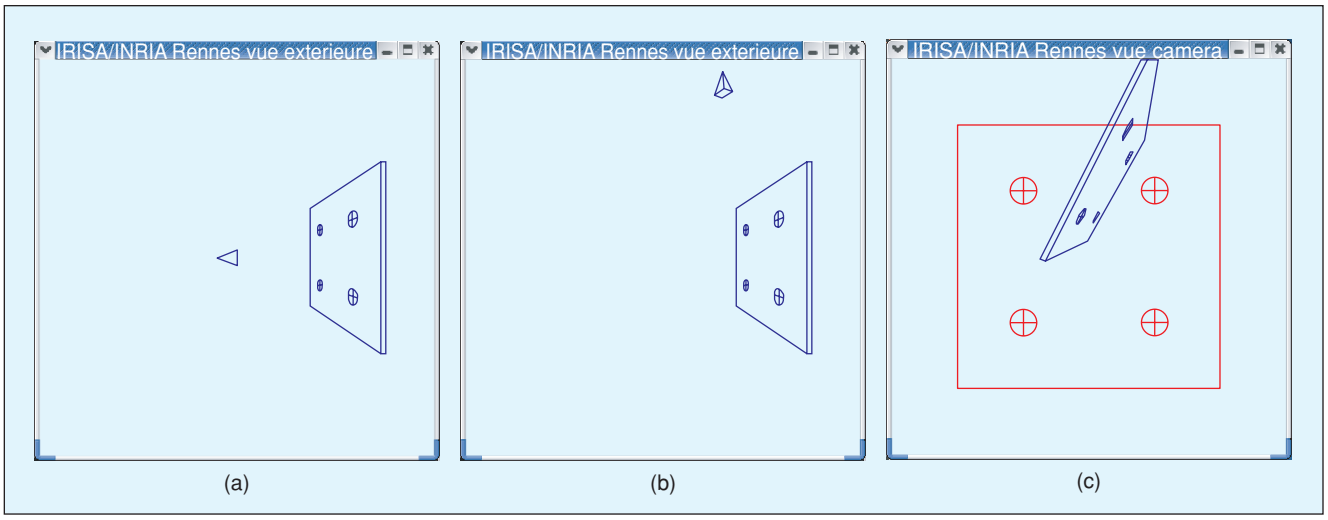

Figure 1. *An example of positioning task: (a) the desired camera pose with respect to a simple target, (b) the initial camera pose, and (c) the corresponding initial and desired image of the target.*

approach is to choose $\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$, where $\mathbf{L}_{e^*}$ is the value of $\mathbf{L}_e$ for the desired position $\mathbf{e} = \mathbf{e}^* = 0$ [8]. In this case, $\widehat{\mathbf{L}_e^+}$ is constant, and only the desired depth of each point has to be set, which means no varying 3-D parameters have to be estimated during the visual servo. Finally, the choice $\widehat{\mathbf{L}_e^+} = 1/2(\mathbf{L}_e + \mathbf{L}_{e^*})^+$ has recently been proposed in [9]. Since $\mathbf{L}_e$ is involved in this method, the current depth of each point must also be available.

We illustrate the behavior of these control schemes with an example. The goal is to position the camera so that it observes a square as a centered square in the image (see Figure 1). We define $\mathbf{s}$ to include the x and y coordinates of the four points forming the square. Note that the initial camera pose has been selected far away from the desired pose, particularly with regard to the rotational motions, which are known to be the most problematic for IBVS. In the simulations presented in the following, no noise or modeling errors have been introduced in order to allow comparison of different behaviors in perfect conditions.

The results obtained by using $\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$ are given in Figure 2. Note that despite the large displacement that is required, the system converges. However, neither the behavior in the image nor the computed camera velocity components nor the 3-D trajectory of the camera present desirable properties far from the convergence (i.e., for the first 30 or so iterations).

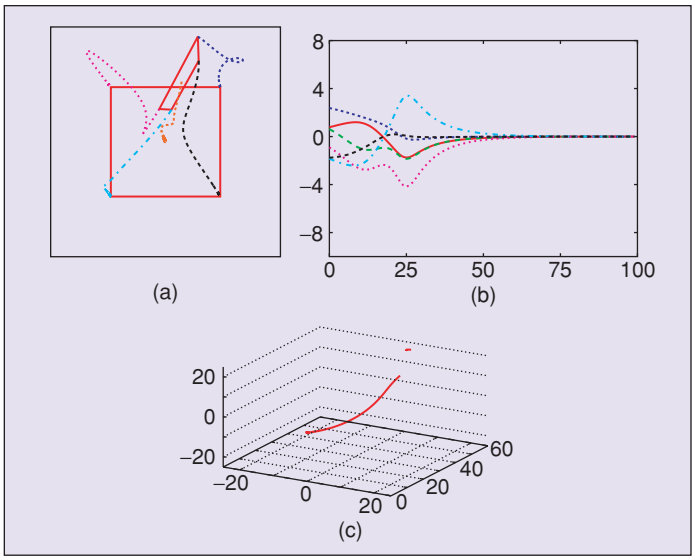

Figure 2. *The system behavior using $\mathbf{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$: (a) image points trajectories including the trajectory of the center of the square ,which is not used in the control scheme, (b) $\mathbf{v}_c$ components (cm/s and dg/s) computed at each iteration of the control scheme, and (c) 3-D trajectory of the camera optical center expressed in $\mathcal{R}_{c^*}$ (cm).*

The results obtained using $\widehat{\mathbf{L}_e^+} = \mathbf{L}_e^+$ are given in Figure 3. In this case, the trajectories of the points in the image are almost straight lines, but the behavior induced in 3-D is even less satisfactory than for the case of $\widehat{\mathbf{L}_e^+} = \mathbf{L}_{e^*}^+$. The large camera velocities at the beginning of the servo indicate that the condition number of $\widehat{\mathbf{L}_e^+}$ is high at the start of the trajectory, and the camera trajectory is far from a straight line.

The choice $\widehat{\mathbf{L}_e^+} = 1/2(\mathbf{L}_e + \mathbf{L}_{e^*})^+$ provides good performance in practice. Indeed, as can be seen in Figure 4, the camera velocity components do not include large oscillations and provide a smooth trajectory both in the image and in 3-D.

A Geometrical Interpretation of IBVS

It is quite easy to provide a geometric interpretation of the behavior of the control schemes defined above. The example illustrated in Figure 5 corresponds to a pure rotation around the optical axis from the initial configuration (shown in blue) to the desired configuration of four coplanar points parallel to the image plane (shown in red).

As explained above, using $\mathbf{L}_{\mathbf{e}}^{+}$ in the control scheme attempts to ensure an exponential decrease of the error $\mathbf{e}$. It means that when x and y image point coordinates compose this error, the points' trajectories in the image follow straight lines from their initial to their desired positions, when it is possible. This leads to the image motion plotted in green in the figure. The camera motion to realize this image motion can be easily deduced and is indeed composed of a rotational motion around the optical axis, but is combined with a retreating translational motion along the optical axis [10]. This unexpected motion is due to the choice of the features and to the coupling between the third and sixth columns in the interaction matrix. If the rotation between the initial and desired configurations is very large, this phenomenon is amplified and leads to a particular case for a rotation of π rad where no rotational motion at all will be induced by the control scheme [11]. On the other hand, when the rotation is small, this phenomenon almost disappears. To conclude, the behavior is locally satisfactory (i.e., when the error is small), but it can be unsatisfactory when the error is large. As we will see in the last part of this article, these results are consistent with the local asymptotic stability results that can be obtained for IBVS.

If instead we use $\mathbf{L}_{\mathbf{e}^*}^{+}$ in the control scheme, the image motion generated can easily be shown to be the blue one plotted in Figure 5. Indeed, if we consider the same control scheme as before but starting from $\mathbf{s}^*$ to reach $\mathbf{s}$, we obtain:

$$\mathbf{v}_c = -\lambda \mathbf{L}_{\mathbf{e}^*}^{+}(\mathbf{s}^* - \mathbf{s}),$$

which again induces straight-line trajectories from the red points to the blue ones, causing image motion plotted in brown. Going back to our problem, the control scheme computes a camera velocity that is exactly the opposite one:

$$\mathbf{v}_c = -\lambda \mathbf{L}_{\mathbf{e}^*}^{+}(\mathbf{s} - \mathbf{s}^*)$$

and thus generates the image motion plotted in red at the red points. Transformed at the blue points, the camera velocity generates the blue image motion and corresponds once again to a rotational motion around the optical axis, combined now with an unexpected forward motion along the optical axis. The same analysis as before can be done, as for large or small errors. We can add that, as soon as the error decreases significantly, both control schemes get closer and tend to the same one (since $\mathbf{L}_{\mathbf{e}} = \mathbf{L}_{\mathbf{e}^*}$ when $\mathbf{e} = \mathbf{e}^*$) with a nice behavior characterized with the image motion plotted in black and a camera motion composed of only a rotation around the optical axis when the error tends towards zero.

If we instead use $\widehat{\mathbf{L}_{\mathbf{e}}^{+}} = 1/2(\mathbf{L}_{\mathbf{e}} + \mathbf{L}_{\mathbf{e}^*})^{+}$, it is intuitively clear that considering the mean of $\mathbf{L}_{\mathbf{e}}$ and $\mathbf{L}_{\mathbf{e}^*}$ generates the image motion plotted in black, even when the error is large. In all cases but the rotation around π rad, the camera motion is now a pure rotation around the optical axis, without any unexpected translational motion.

IBVS with a Stereovision System

It is straightforward to extend the IBVS approach to a multicamera system. If a stereovision system is used, and a 3-D point is visible in both left and right images (see Figure 6), it is possible to use as visual features

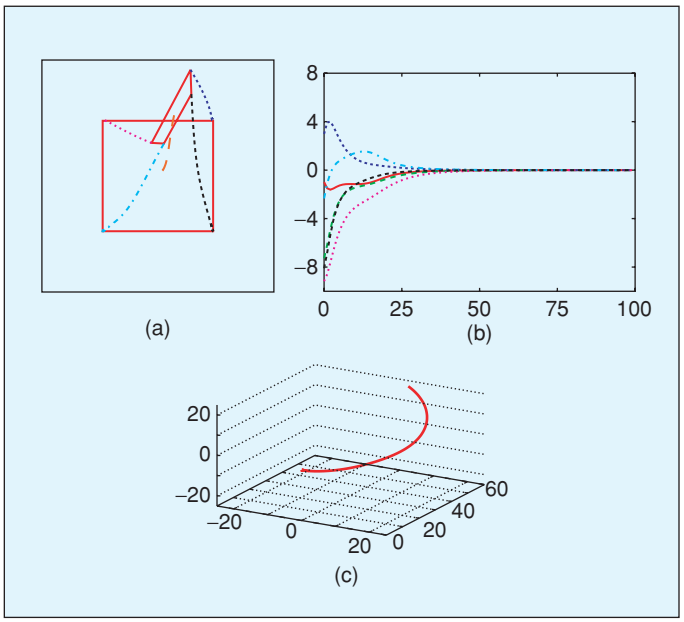

Figure 3. *The system behavior using* $\mathbf{s} = (x_1, y_1, \ldots, x_4, y_4)$ *and* $\widehat{\mathbf{L}_{\mathbf{e}}^{+}} = \mathbf{L}_{\mathbf{e}}^{+}$.

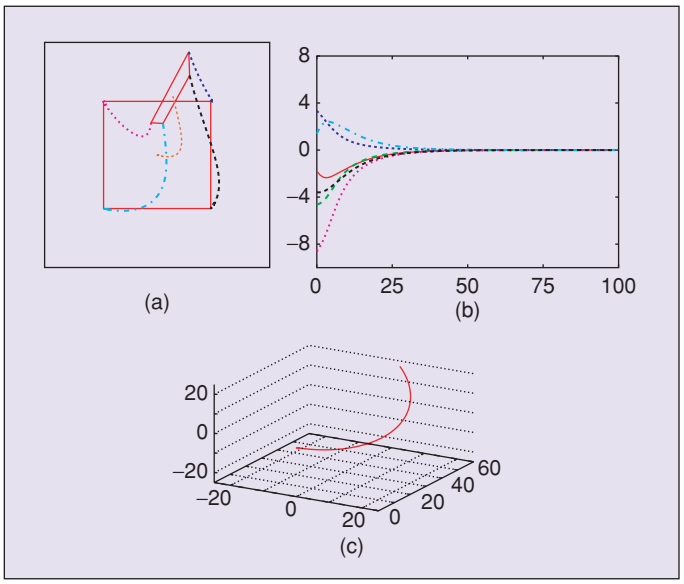

Figure 4. *The system behavior using* $\mathbf{s} = (x_1, y_1, \ldots, x_4, y_4)$ *and* $\widehat{\mathbf{L}_{\mathbf{e}}^{+}} = \frac{1}{2}(\mathbf{L}_{\mathbf{e}} + \mathbf{L}_{\mathbf{e}}^{*})^{+}$.

$$\mathbf{s} = \mathbf{x}_s = (\mathbf{x}_l, \mathbf{x}_r) = (x_l, y_l, x_r, y_r),$$

i.e., to represent the point by just stacking in $\mathbf{s}$ the x and y coordinates of the observed point in the left and right images [12]. However, care must be taken when constructing the corresponding interaction matrix since the form given in (10) is expressed in either the left or right camera frame. More precisely, we have:

$$\begin{cases} \dot{\mathbf{x}}_l &= \mathbf{L}_{\mathbf{x}_l}\, \mathbf{v}_l \\ \dot{\mathbf{x}}_r &= \mathbf{L}_{\mathbf{x}_r}\, \mathbf{v}_r, \end{cases}$$

where $\mathbf{v}_l$ and $\mathbf{v}_r$ are the spatial velocity of the left and right camera respectively, and where the analytical form of $\mathbf{L}_{\mathbf{x}_l}$ and $\mathbf{L}_{\mathbf{x}_r}$ are given by (11).

By choosing a sensor frame rigidly linked to the stereovision system, we obtain:

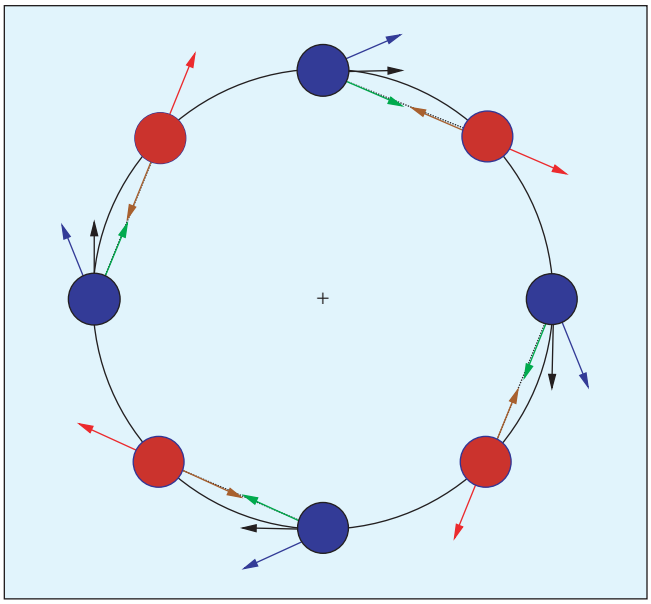

Figure 5. *A geometrical interpretation of IBVS.*

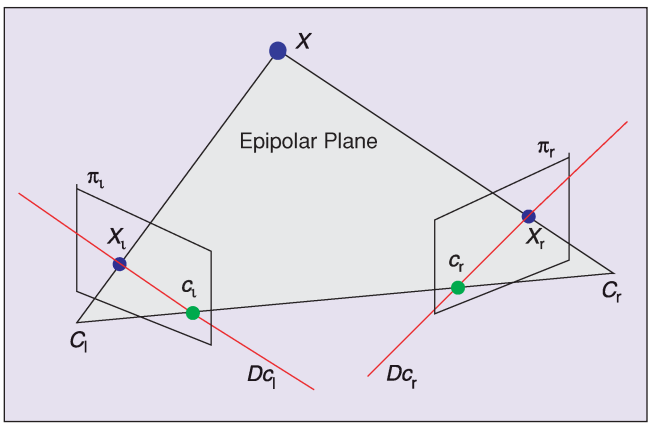

Figure 6. *A stereovision system.*

$$\dot{\mathbf{x}}_s = \begin{bmatrix} \dot{\mathbf{x}}_l \\ \dot{\mathbf{x}}_r \end{bmatrix} = \mathbf{L}_{\mathbf{x}_s}\, \mathbf{v}_s,$$

where the interaction matrix related to $\mathbf{x}_s$ can be determined using the spatial motion transform matrix $\mathbf{V}$ to transform velocities expressed in the left or right cameras frames to the sensor frame. $\mathbf{V}$ is given by [13]

$$\mathbf{V} = \begin{bmatrix} \mathbf{R} & [\mathbf{t}]_\times \mathbf{R} \\ \mathbf{0} & \mathbf{R} \end{bmatrix}, \tag{12}$$

where $[\mathbf{t}]_\times$ is the skew symmetric matrix associated to the vector $\mathbf{t}$ and where $(\mathbf{R}, \mathbf{t}) \in SE(3)$ is the rigid body transformation from camera to sensor frame. The numerical values for these matrices are directly obtained from the calibration step of the stereovision system. Using this equation, we obtain

$$\mathbf{L}_{\mathbf{x}_s} = \begin{bmatrix} \mathbf{L}_{\mathbf{x}_l}\, {}^l\mathbf{V}_s \\ \mathbf{L}_{\mathbf{x}_r}\, {}^r\mathbf{V}_s \end{bmatrix}.$$

Note that $\mathbf{L}_{\mathbf{x}_s} \in \mathbb{R}^{4\times 6}$ is always of rank 3 because of the epipolar constraint that links the perspective projection of a 3-D point in a stereovision system (see Figure 6). Another simple interpretation is that a 3-D point is represented by three independent parameters, making it impossible to find more than three independent parameters using any sensor observing that point.

To control the 6 DOF of the system, it is necessary to consider at least three points, the rank of the interaction matrix by considering only two points being equal to 5.

Using a stereovision system, since the 3-D coordinates of any point observed in both images can be easily estimated by a simple triangulation process it is possible and quite natural to use these 3-D coordinates in the features set $\mathbf{s}$. Such an approach would be, strictly speaking, a position-based approach, since it would require 3-D parameters in $\mathbf{s}$.

Position-Based Visual Servo

Position-based control schemes (PBVS) [2], [14], [15] use the pose of the camera with respect to some reference coordinate frame to define $\mathbf{s}$. Computing that pose from a set of measurements in one image necessitates the camera intrinsic parameters and the 3-D model of the object observed to be known. This classical computer vision problem is called the *3-D localization problem*. While this problem is beyond the scope of the present tutorial, many solutions have been presented in the literature (see, e.g., [16] and [17]).

It is then typical to define $\mathbf{s}$ in terms of the parameterization used to represent the camera pose. Note that the parameters $\mathbf{a}$ involved in the definition (1) of $\mathbf{s}$ are now the camera intrinsic parameters and the 3-D model of the object.

It is convenient to consider three coordinate frames: the current camera frame $\mathcal{F}_c$, the desired camera frame $\mathcal{F}_{c^*}$, and a reference frame $\mathcal{F}_o$ attached to the object. We adopt here the

standard notation of using a leading superscript to denote the frame with respect to which a set of coordinates is defined. Thus, the coordinate vectors ${}^{c}\mathbf{t}_o$ and ${}^{c^*}\mathbf{t}_o$ give the coordinates of the origin of the object frame expressed relative to the current camera frame and relative to the desired camera frame, respectively. Furthermore, let $\mathbf{R} = {}^{c^*}\mathbf{R}_c$ be the rotation matrix that gives the orientation of the current camera frame relative to the desired frame.

We can define $\mathbf{s}$ to be $(\mathbf{t}, \theta\mathbf{u})$, in which $\mathbf{t}$ is a translation vector, and $\theta\mathbf{u}$ gives the angle/axis parameterization for the rotation. We now discuss two choices for $\mathbf{t}$ and give the corresponding control laws.

If $\mathbf{t}$ is defined relative to the object frame $\mathcal{F}_o$, we obtain $\mathbf{s} = ({}^{c}\mathbf{t}_o, \theta\mathbf{u})$, $\mathbf{s}^* = ({}^{c^*}\mathbf{t}_o, \mathbf{0})$, and $\mathbf{e} = ({}^{c}\mathbf{t}_o - {}^{c^*}\mathbf{t}_o, \theta\mathbf{u})$. In this case, the interaction matrix related to $\mathbf{e}$ is given by

$$\mathbf{L}_\mathbf{e} = \begin{bmatrix} -\mathbf{I}_3 & [{}^{c}\mathbf{t}_o]_\times \\ \mathbf{0} & \mathbf{L}_{\theta\mathbf{u}} \end{bmatrix}, \tag{13}$$

in which $\mathbf{I}_3$ is the 3×3 identity matrix and $\mathbf{L}_{\theta\mathbf{u}}$ is given by [18]:

$$\mathbf{L}_{\theta\mathbf{u}} = \mathbf{I}_3 - \frac{\theta}{2}[\mathbf{u}]_\times + \left(1 - \frac{\operatorname{sinc}\theta}{\operatorname{sinc}^2\frac{\theta}{2}}\right)[\mathbf{u}]_\times^2, \tag{14}$$

where $\operatorname{sinc} x$ is the sinus cardinal defined such that $x \operatorname{sinc} x = \sin x$ and $\operatorname{sinc} 0 = 1$.

Following the developments presented at the beginning of the article, we obtain the control scheme

$$\mathbf{v}_c = -\lambda\widehat{\mathbf{L}_\mathbf{e}^{-1}}\mathbf{e},$$

since the dimension k of $\mathbf{s}$ is 6, which is the number of camera degrees of freedom. By setting:

$$\widehat{\mathbf{L}_\mathbf{e}^{-1}} = \begin{bmatrix} -\mathbf{I}_3 & [{}^{c}\mathbf{t}_o]_\times \mathbf{L}_{\theta\mathbf{u}}^{-1} \\ \mathbf{0} & \mathbf{L}_{\theta\mathbf{u}}^{-1} \end{bmatrix}, \tag{15}$$

we obtain after simple developments:

$$\begin{cases} \boldsymbol{v}_c &= -\lambda(({}^{c^*}\mathbf{t}_o - {}^{c}\mathbf{t}_o) + [{}^{c}\mathbf{t}_o]_\times \theta\mathbf{u}) \\ \boldsymbol{\omega}_c &= -\lambda\theta\mathbf{u}, \end{cases} \tag{16}$$

since $\mathbf{L}_{\theta\mathbf{u}}$ is such that $\mathbf{L}_{\theta\mathbf{u}}^{-1}\theta\mathbf{u} = \theta\mathbf{u}$.

This PBVS scheme causes the rotational motion to follow a geodesic with an exponential decreasing speed and so that the translational parameters involved in $\mathbf{s}$ decrease with the same speed. This explains the nice exponential decrease of the camera velocity components in Figure 7. Furthermore, the trajectory in the image of the origin of the object frame follows a pure straight line (here the center of the four points has been selected as this origin). On the other hand, the camera trajectory does not follow a straight line.

Another PBVS scheme can be designed by using $\mathbf{s} = ({}^{c^*}\mathbf{t}_c, \theta\mathbf{u})$. In that case, we have $\mathbf{s}^* = \mathbf{0}$, $\mathbf{e} = \mathbf{s}$, and

$$\mathbf{L}_\mathbf{e} = \begin{bmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_{\theta\mathbf{u}} \end{bmatrix}. \tag{17}$$

Note the decoupling between translational and rotational motions, which allows us to obtain a simple control scheme:

$$\begin{cases} \boldsymbol{v}_c &= -\lambda\mathbf{R}^\top {}^{c^*}\mathbf{t}_c \\ \boldsymbol{\omega}_c &= -\lambda\theta\mathbf{u}. \end{cases} \tag{18}$$

In this case, as can be seen in Figure 8, if the pose parameters involved in (18) are perfectly estimated, the camera trajectory is a pure straight line, while the image trajectories are less satisfactory than before. Some particular configurations can even be found so that some points leave the camera field of view.

Stability Analysis

In this section, we consider the fundamental issues related to the stability of the controllers. To assess the stability of the closed-loop visual servo systems, we will use Lyapunov analysis. In particular, consider the candidate Lyapunov function defined by the squared error norm $\mathcal{L} = 1/2\|\mathbf{e}(t)\|^2$, whose derivative is given by

$$\begin{aligned} \dot{\mathcal{L}} &= \mathbf{e}^\top\dot{\mathbf{e}} \\ &= -\lambda\mathbf{e}^\top\mathbf{L}_\mathbf{e}\widehat{\mathbf{L}_\mathbf{e}^{+}}\mathbf{e}. \end{aligned}$$

The global asymptotic stability of the system is thus obtained when the following sufficient condition is ensured:

$$\mathbf{L}_\mathbf{e}\widehat{\mathbf{L}_\mathbf{e}^{+}} > 0. \tag{19}$$

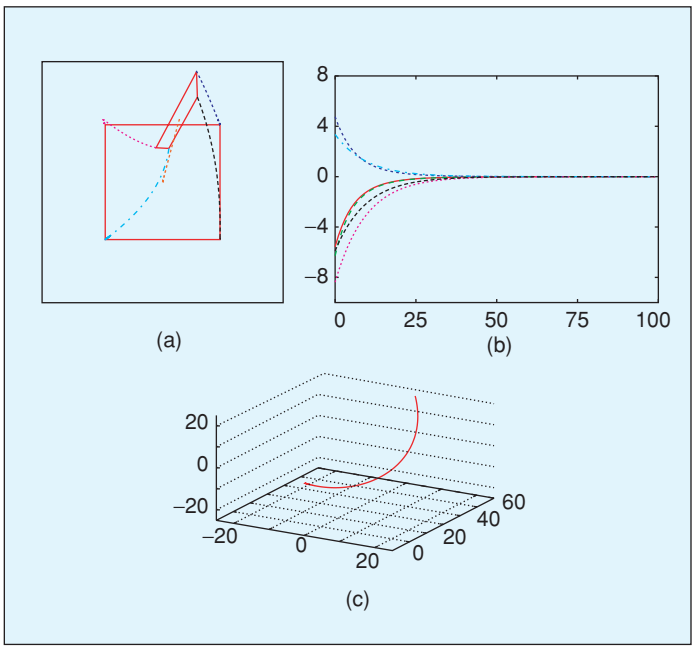

Figure 7. *The system behavior using* $\mathbf{s} = ({}^{c}\mathbf{t}_0, \theta\mathbf{u})$.

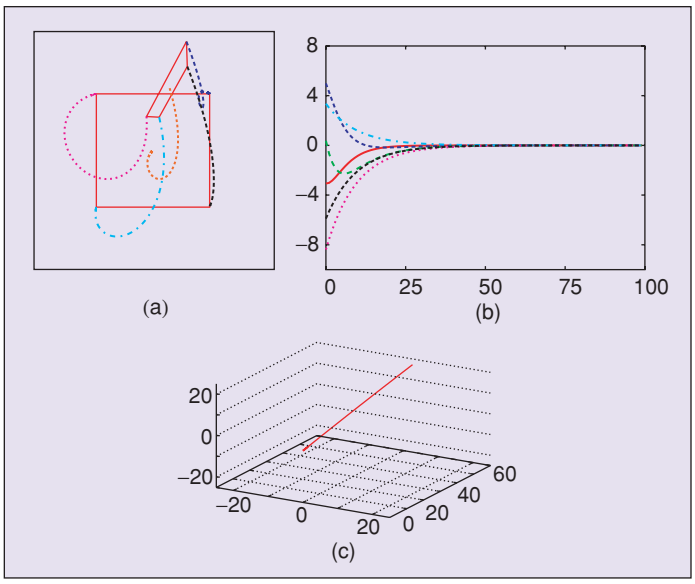

Figure 8. *The system behavior using* $\mathbf{s} = ({}^{c^*}\mathbf{t}_c, \theta\mathbf{u})$.

If the number of features is equal to the number of camera degrees of freedom (i.e., $k = 6$), and if the features are chosen and the control scheme designed so that $\mathbf{L}_\mathbf{e}$ and $\widehat{\mathbf{L}_\mathbf{e}^+}$ are of full rank 6, then condition (19) is ensured if the approximations involved in $\widehat{\mathbf{L}_\mathbf{e}^+}$ are not too coarse.

We now consider the particularization of this to the specific cases of IBVS and PBVS.

Stability Analysis of IBVS

As discussed previously, for most IBVS approaches we have $k > 6$. Therefore, condition (19) can never be ensured since $\mathbf{L}_\mathbf{e}\widehat{\mathbf{L}_\mathbf{e}^+} \in \mathbb{R}^{k\times k}$ is at most of rank 6; thus, $\mathbf{L}_\mathbf{e}\widehat{\mathbf{L}_\mathbf{e}^+}$ has a nontrivial null space. In this case, configurations such that $\mathbf{e} \in \mathrm{Ker}\widehat{\mathbf{L}_\mathbf{e}^+}$ correspond to local minima. Reaching such a local minimum is illustrated in Figure 9. As can be seen in Figure 9(d), each component of $\mathbf{e}$ has a nice exponential decrease with the same convergence speed, causing straight-line trajectories to be realized in

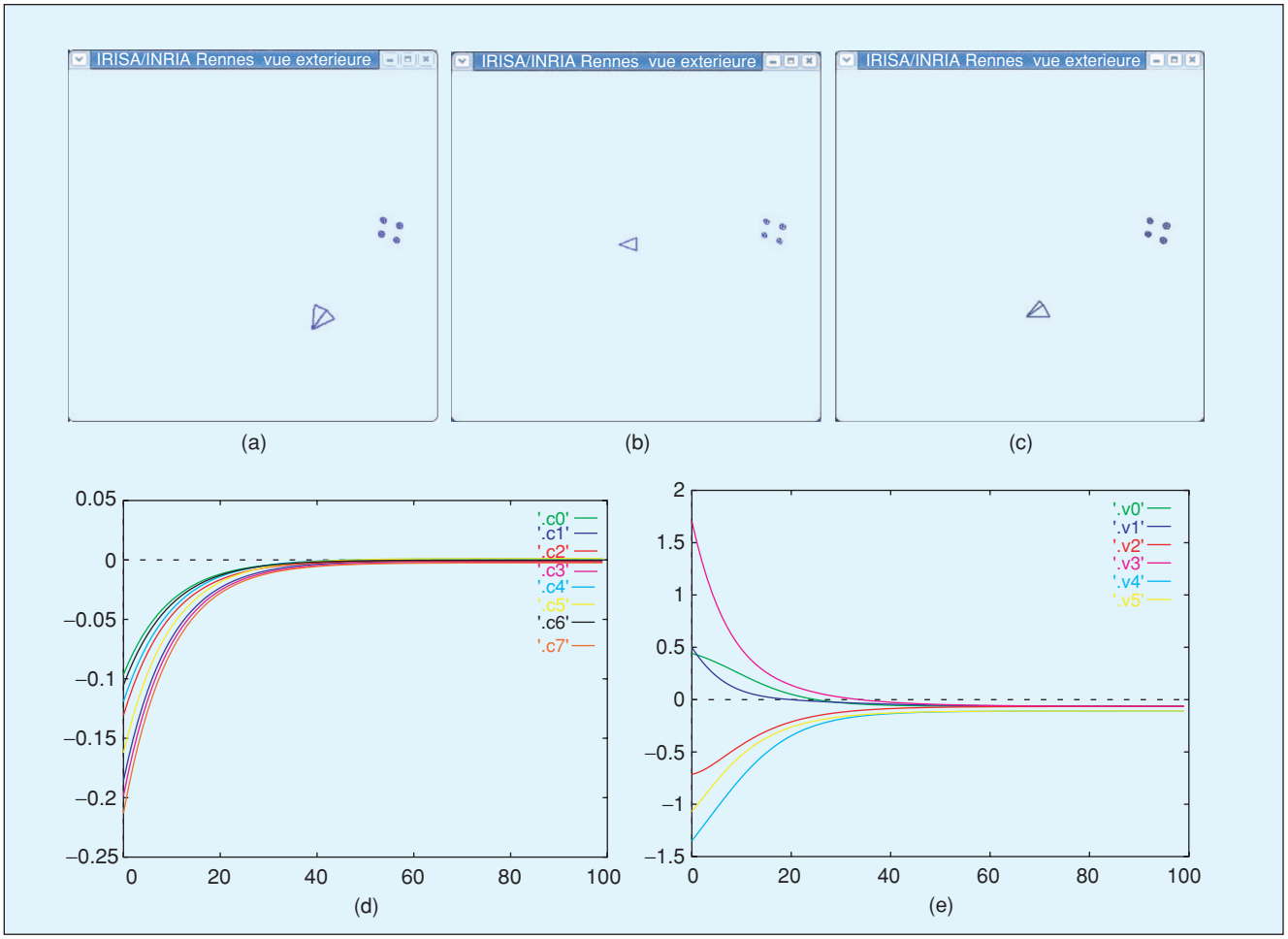

Figure 9. *Reaching a local minimum using* $\mathbf{s} = (x_1, y_1, \ldots, x_4, y_4)$ *and* $\widehat{\mathbf{L}_\mathbf{e}^+} = \mathbf{L}_\mathbf{e}^+$*: (a) initial configuration, (b) the desired one, (c) the configuration reached after the convergence of the control scheme, (d) the evolution of the error* $\mathbf{e}$ *at each iteration of the control scheme, and (f) the evolution of the six components of the camera velocity* $\mathbf{v}_c$.

the image, but the error reached is not exactly zero, and it is clear from Figure 9(c) that the system has been attracted to a local minimum far away from the desired configuration. Thus, only local asymptotic stability can be obtained for IBVS.

To study local asymptotic stability when $k > 6$, let us first define a new error $\mathbf{e}'$ with $\mathbf{e}' = \widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{e}$. The time derivative of this error is given by

$$\begin{aligned}\dot{\mathbf{e}}' &= \widehat{\mathbf{L}_\mathbf{e}^+}\dot{\mathbf{e}} + \dot{\widehat{\mathbf{L}_\mathbf{e}^+}}\mathbf{e} \\ &= (\widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{L}_\mathbf{e} + \mathbf{O})\mathbf{v}_c,\end{aligned}$$

where $\mathbf{O} \in \mathbb{R}^{6\times 6}$ is equal to 0 when $\mathbf{e} = 0$, whatever the choice of $\widehat{\mathbf{L}_\mathbf{e}^+}$ [19]. Using the control scheme (5), we obtain:

$$\dot{\mathbf{e}}' = -\lambda(\widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{L}_\mathbf{e} + \mathbf{O})\mathbf{e}',$$

which is known to be locally asymptotically stable in a neighborhood of $\mathbf{e} = \mathbf{e}^* = 0$ if

$$\widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{L}_\mathbf{e} > 0, \tag{20}$$

where $\widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{L}_\mathbf{e} \in \mathbb{R}^{6\times 6}$. Indeed, only the linearized system $\dot{\mathbf{e}}' = -\lambda\widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{L}_\mathbf{e}\mathbf{e}'$ has to be considered if we are interested in the local asymptotic stability [20].

Once again, if the features are chosen and the control scheme designed so that $\mathbf{L}_\mathbf{e}$ and $\widehat{\mathbf{L}_\mathbf{e}^+}$ are of full rank 6, then condition (20) is ensured if the approximations involved in $\widehat{\mathbf{L}_\mathbf{e}^+}$ are not too coarse.

To end the demonstration of local asymptotic stability, we must show that there does not exist any configuration $\mathbf{e} \neq \mathbf{e}^*$ such that $\mathbf{e} \in$ Ker $\widehat{\mathbf{L}_\mathbf{e}^+}$ in a small neighborhood of $\mathbf{e}^*$ and in a small neighborhood of the corresponding pose $\mathbf{p}^*$. Such configurations correspond to local minima where $\mathbf{v}_c = 0$ and $\mathbf{e} \neq \mathbf{e}^*$. If such a pose $\mathbf{p}$ would exist, it is possible to restrict the neighborhood around $\mathbf{p}^*$ so that there exists a camera velocity $\mathbf{v}$ to reach $\mathbf{p}^*$ from $\mathbf{p}$. This camera velocity would imply a variation of the error $\dot{\mathbf{e}} = \mathbf{L}_\mathbf{e}\mathbf{v}$. However, such a variation cannot belong to Ker $\widehat{\mathbf{L}_\mathbf{e}^+}$ since $\widehat{\mathbf{L}_\mathbf{e}^+}\mathbf{L}_\mathbf{e} > 0$. Therefore, we have $\mathbf{v}_c = 0$ if and only if $\dot{\mathbf{e}} = 0$, i.e., $\mathbf{e} = \mathbf{e}^*$, in a neighborhood of $\mathbf{p}^*$.

Even though local asymptotic stability can be ensured when $k > 6$, we cannot ensure global asymptotic stability. For instance, as illustrated in Figure 9, there may exist local minima corresponding to configurations where $\mathbf{e} \in$ Ker $\widehat{\mathbf{L}_\mathbf{e}^+}$, which are outside of the neighborhood considered above. Determining the size of the neighborhood where the stability and the convergence are ensured is still an open issue, even if this neighborhood is surprisingly quite large in practice.

Stability Analysis of PBVS

The stability properties of PBVS seem quite attractive. Since $\mathbf{L}_{\theta\mathbf{u}}$ given in (14) is nonsingular when $\theta \neq 2k\pi$, we obtain from (19) the global asymptotic stability of the system since $\mathbf{L}_\mathbf{e}\widehat{\mathbf{L}_\mathbf{e}^{-1}} = \mathbf{I}_6$, under the strong hypothesis that all the pose parameters are perfect. This is true for both PBVS methods presented previously, since the interactions matrices given in (13) and (17) are full rank when $\mathbf{L}_{\theta\mathbf{u}}$ is nonsingular.

With regard to robustness, feedback is computed using *estimated* quantities that are a function of the image measurements and the system calibration parameters. For the first PBVS method (the analysis for the second method is analogous), the interaction matrix given in (13) corresponds to perfectly estimated pose parameters, while the real one is unknown since the estimated pose parameters may be biased due to calibration errors, or inaccurate and unstable due to noise [11]. The true positivity condition (19) should be in fact written:

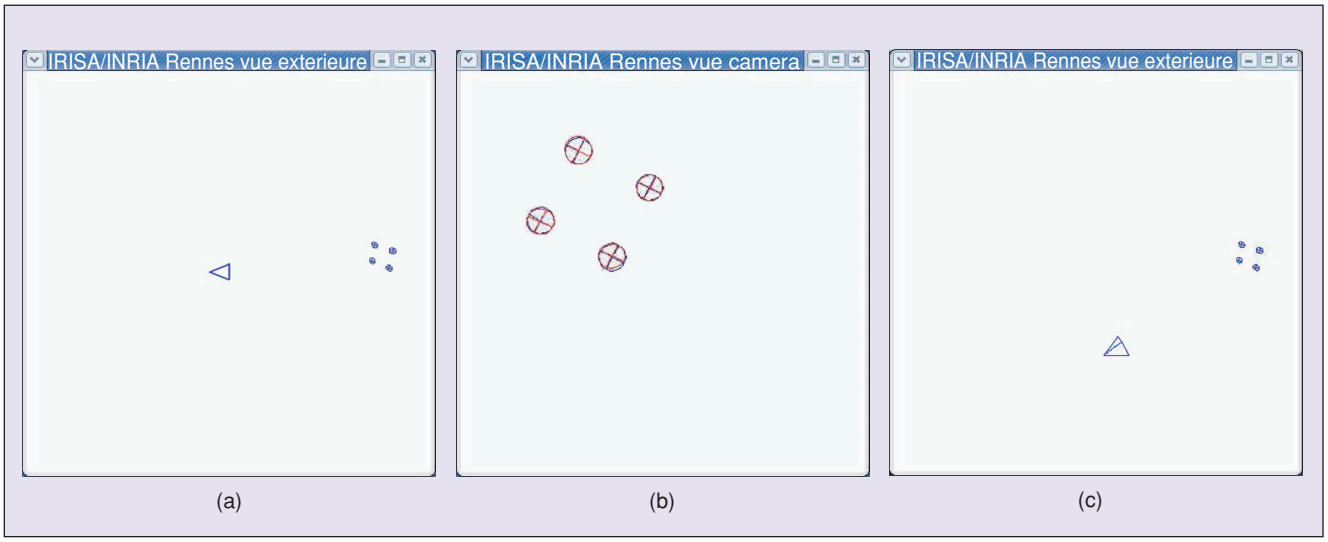

Figure 10. *Two different camera poses [(a) and (c)] that provide almost the same image of four coplanar points (b).*

$$\mathbf{L}_{\widehat{\mathbf{e}}}\widehat{\mathbf{L}_{\widehat{\mathbf{e}}}^{-1}} > 0, \tag{21}$$

where $\widehat{\mathbf{L}_{\widehat{\mathbf{e}}}^{-1}}$ is given by (15) but where $\mathbf{L}_{\widehat{\mathbf{e}}}$ is unknown and not given by (13). Indeed, even small errors in computing the points position in the image can lead to pose errors that can impact significantly the accuracy and the stability of the system (see Figure 10).

Conclusions

At this point, one might wonder which of IBVS or PBVS is the superior control strategy. As with most engineering questions, there is no definitive answer—only a set of performance tradeoffs to consider.

As for stability issues, we have seen that no strategy provides perfect properties. As for the 3-D parameters involved, a correct estimation is important in IBVS, since they appear in $\mathbf{L}_{\mathbf{e}}$ and thus in the stability condition (20). Poor estimation can thus make the system unstable, but coarse estimations will only imply perturbations in the trajectory performed by the robot to reach its desired pose and will have no effect on the accuracy of the pose reached. On the other hand, a correct estimation of the pose is crucial in PBVS, since it appears both in the error $\mathbf{e}$ to be regulated to 0 and in $\mathbf{L}_{\mathbf{e}}$. Coarse estimations will thus induce perturbations on the trajectory realized but will have also an effect on the accuracy of the pose reached after convergence.

In fact, in PBVS, the vision sensor is considered as a 3-D sensor. Since the control scheme imposes a behavior of $\mathbf{s}$, which is here expressed in the Cartesian space, it allows the camera to follow theoretically an optimal trajectory in that space but generally not in the image space. When only one image is used, even small errors in the image measurements can lead to errors in the pose that can impact significantly the accuracy of the system. The main question with this approach is thus: Can a vision sensor be considered as a 3-D sensor?

On the other hand, in IBVS, the vision sensor is considered as a two-dimensional (2-D) sensor since the features are directly expressed in the image space. That is more realistic when a monocular camera is used, and this allows IBVS to be remarkably robust to errors in calibration and image noise. However, IBVS is not without its shortcomings. When the displacement to realize is large, the camera may reach a local minimum or may cross a singularity of the interaction matrix. Furthermore, the camera motion may follow unpredictable, often suboptimal Cartesian trajectories, such as those discussed previously.

Even if the two basic approaches presented and discussed in this article give in practice satisfactory results in most cases, their respective shortcomings have led to many works and improvements. Results in computer vision about 3-D reconstruction from two views have also been applied successfully, making the knowledge of the 3-D object model unnecessary. These different approaches will form one of the main points presented in Part II of this tutorial.

References

[1] J. Feddema and O. Mitchell, "Vision-guided servoing with feature-based trajectory generation," *IEEE Trans. Robot. Automat.*, vol. 5, pp. 691–700, Oct. 1989.

[2] L. Weiss, A. Sanderson, and C. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE J. Robot. Automat.*, vol. 3, pp. 404–417, Oct. 1987.

[3] D. Forsyth and J. Ponce, *Computer Vision: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall, 2003.

[4] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3-D Vision: From Images to Geometric Models*. New York: Springer-Verlag, 2003.

[5] H. Michel and P. Rives, "Singularities in the determination of the situation of a robot effector from the perspective view of three points," INRIA Research Report, Tech. Rep. 1850, Feb. 1993.

[6] M. Fischler and R. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communicat. ACM*, vol. 24, pp. 381–395, June 1981.

[7] S. Hutchinson, G. Hager, and P. Corke, "A tutorial on visual servo control," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 651–670, Oct. 1996.

[8] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. Robotics and Automation*, vol. 8, pp. 313–326, June 1992.

[9] E. Malis, "Improving vision-based control using efficient second-order minimization techniques," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 1843–1848, Apr. 2004.

[10] P. Corke and S. Hutchinson, "A new partitioned approach to image-based visual servo control," *IEEE Trans. Robot. Automat.*, vol. 17, no. 4, pp. 507–515, 2001.

[11] F. Chaumette, "Potential problems of stability and convergence in image-based and position-based visual servoing," in *The Confluence of Vision and Control*, vol. 237, *Lecture Notes in Control and Information Sciences,* D. Kriegman, G. Hager, and S. Morse, Eds. New York: Springer-Verlag, 1998, pp. 66–78.

[12] G. Hager, W. Chang, and A. Morse, "Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination," *IEEE Control Syst. Mag.*, vol. 15, pp. 30–39, Feb. 1995.

[13] R. Paul, *Robot Manipulators: Mathematics, Programming and Control*. Cambridge, MA: MIT Press, 1982.

[14] W. Wilson, C. Hulls, and G. Bell, "Relative end-effector control using cartesian position based visual servoing," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 684–696, Oct. 1996.

[15] B. Thuilot, P. Martinet, L. Cordesses, and J. Gallice, "Position based visual servoing: Keeping the object in the field of vision," in *Proc. IEEE Int. Conf. Robot. Automat.*, pp. 1624–1629, May 2002.

[16] D. Dementhon and L. Davis, "Model-based object pose in 25 lines of code," *Int. J. Comput. Vis.*, vol. 15, pp. 123–141, June 1995.

[17] D. Lowe, "Three-dimensional object recognition from single two-dimensional images," *Artif. Intell.*, vol. 31, no. 3, pp. 355–395, 1987.

[18] E. Malis, F. Chaumette, and S. Boudet, "2-1/2D visual servoing," *IEEE Trans. Robot. Automat.*, vol. 15, pp. 238–250, Apr. 1999.

[19] E. Malis, "Visual servoing invariant to changes in camera intrinsic parameters," *IEEE Trans. Robot. Automat.*, vol. 20, pp. 72–81, Feb. 2004.

[20] A. Isidori, *Nonlinear Control Systems*, 3rd ed. New York: Springer-Verlag, 1995.

Address for Correspondence: François Chaumette, IRISA/INRIA Rennes, Campus de Beaulieu, 35 042 Rennes cedex, France. E-mail: Francois.Chaumette@irisa.fr.