Visual Servoing

François Chaumette, Seth Hutchinson, Peter Corke

Handbook of Robotics, 2nd edition

Chapter 34: Visual servoing

François Chaumette, Seth
Hutchinson, Peter Corke

pages 841 – 866

Springer, 2016

Chapter 34

Visual Servoing

Summary

This chapter introduces visual servo control, using computer vision data in the servo loop to control the motion of a robot. We first describe the basic techniques that are by now well established in the field. We give a general overview of the formulation of the visual servo control problem, and describe the two archetypal visual servo control schemes: image-based and pose-based visual servo control. We then discuss performance and stability issues that pertain to these two schemes, motivating advanced techniques. Of the many advanced techniques that have been developed, we discuss 2.5-D, hybrid, partitioned, and switched approaches. Having covered a variety of control schemes, we deal with target tracking and controlling motion directly in the joint space and extensions to under-actuated ground and aerial robots. We conclude by describing applications of visual servoing in robotics.

Introduction

Visual servo (VS) 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 chapter 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. In the present chapter, we will deal primarily with the issues from control theory, making connections to previous chapters when appropriate.

34.1 The Basic Components of Visual Servoing

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

$$\boldsymbol{e}(t) = \boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a}) - \boldsymbol{s}^* \,. \tag{34.1}$$

This formulation is quite general, and it encompasses a wide variety approaches, as we will see below. The parameters in (34.1) are defined as follows. The vector $\boldsymbol{m}(t)$ is a set of image measurements (e.g., the image coordinates of interest points, or the parameters of a set of image lines or segments). These image measurements are used to compute a vector of k visual features, $\boldsymbol{s}(\boldsymbol{m}(t), \boldsymbol{a})$, in which $\boldsymbol{a}$ is a set of parameters that represent potential additional knowledge about the system (e.g., true or approximate camera intrinsic parameters or a model of the object to be tracked). The vector $\boldsymbol{s}^*$ contains the desired values of the features. Note that the order of the desired and

actual values in (34.1) is reversed with respect to the common convention for feedback control systems.

For now, we consider the case of a fixed goal pose and a motionless target, i. e., $\boldsymbol{s}^*$ is constant, and changes in $\boldsymbol{s}$ depend only on camera motion. Further, we consider here the case of controlling the motion of a camera with six degrees of freedom (e.g., a camera attached to the end effector of a six-degree-of-freedom arm). We will treat more general cases in later sections.

Visual servoing schemes mainly differ in the way that $\boldsymbol{s}$ is designed. In Sects. 34.2 and 34.3, we describe classical approaches, including image-based visual servo control (IBVS), in which $\boldsymbol{s}$ consists of a set of features that are immediately available in the image, and pose-based visual servo control (PBVS), in which $\boldsymbol{s}$ consists of a pose, which must be estimated from image measurements. Note that in the older visual servoing literature PBVS is named position-based, rather than pose-based, visual servoing [3, 2]. We also present in Sect. 34.4 several more-advanced methods.

Once $\boldsymbol{s}$ 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 $\boldsymbol{s}$ and the camera velocity. Let the spatial velocity of the camera be denoted by $\boldsymbol{v}_\mathrm{c} = (\boldsymbol{v}_\mathrm{c}, \omega_\mathrm{c})$ where $\boldsymbol{v}_\mathrm{c}$ is the instantaneous linear velocity of the origin of the camera frame and ω_c is the instantaneous angular velocity of the camera frame. The relationship between $\dot{\boldsymbol{s}}$ and $\boldsymbol{v}_\mathrm{c}$ is given by

$$\dot{\boldsymbol{s}} = \mathbf{L}_{\boldsymbol{s}} \boldsymbol{v}_\mathrm{c} \;, \tag{34.2}$$

in which $\mathbf{L}_{\boldsymbol{s}} \in \mathbb{R}^{k\times 6}$ is called the *interaction matrix* related to $\boldsymbol{s}$ [1]. The term *feature Jacobian* is also used somewhat interchangeably in the visual servo literature [2], but in the present chapter we will use this latter term to relate the time variation of the features to the robot's joint velocity (Sect. 34.9).

Using (34.1) and (34.2) we immediately obtain the relationship between camera velocity and the time variation of the error:

$$\dot{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{e}} \boldsymbol{v}_\mathrm{c} \;, \tag{34.3}$$

where $\mathbf{L}_{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{s}}$. Considering $\boldsymbol{v}_\mathrm{c}$ as the input to the robot controller, and if we would like, for instance, to design for an exponential and decoupled decrease of the error (i. e., $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$) then using (34.3) we obtain as controller

$$\boldsymbol{v}_\mathrm{c} = -\lambda \mathbf{L}_{\boldsymbol{e}}^{+} \boldsymbol{e} \;, \tag{34.4}$$

where $\mathbf{L}_{\boldsymbol{e}}^{+} \in \mathbb{R}^{6\times k}$ is the Moore–Penrose pseudo-inverse of $\mathbf{L}_{\boldsymbol{e}}$, that is, $\mathbf{L}_{\boldsymbol{e}}^{+} = (\mathbf{L}_{\boldsymbol{e}}^{\top}\mathbf{L}_{\boldsymbol{e}})^{-1}\mathbf{L}_{\boldsymbol{e}}^{\top}$ when $k \geq 6$ and $\mathbf{L}_{\boldsymbol{e}}$ is of full rank 6. When $k = 6$, if $\det \mathbf{L}_{\boldsymbol{e}} \neq 0$ it is possible to invert $\mathbf{L}_{\boldsymbol{e}}$, giving the control $\boldsymbol{v}_\mathrm{c} = -\lambda \mathbf{L}_{\boldsymbol{e}}^{-1} \boldsymbol{e}$. When $k \leq 6$ and $\mathbf{L}_{\boldsymbol{e}}$ is of full rank k, $\mathbf{L}_{\boldsymbol{e}}^{+}$ is given by $\mathbf{L}_{\boldsymbol{e}}^{+} = \mathbf{L}_{\boldsymbol{e}}^{\top}\left(\mathbf{L}_{\boldsymbol{e}}\mathbf{L}_{\boldsymbol{e}}^{\top}\right)^{1}$. When $\mathbf{L}_{\boldsymbol{e}}$ is not full rank, the numerical value of $\mathbf{L}_{\boldsymbol{e}}^{+}$ can be obtained from the singular value decomposition of $\mathbf{L}_{\boldsymbol{e}}$. In all cases, control scheme (34.4) allows $\|\dot{\boldsymbol{e}} - \lambda \mathbf{L}_{\boldsymbol{e}}\mathbf{L}_{\boldsymbol{e}}^{+}\boldsymbol{e}\|$ and $\|\boldsymbol{v}_\mathrm{c}\|$ to be minimal. Note that the desired behavior $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$ is obtained only when $\mathbf{L}_{\boldsymbol{e}}\mathbf{L}_{\boldsymbol{e}}^{+} = \mathbf{I}_k$, where $\mathbf{I}_k$ is the $k \times k$ identity matrix, that is, only when $\mathbf{L}_{\boldsymbol{e}}$ if of full rank k, $k \leq 6$.

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

$$\boldsymbol{v}_\mathrm{c} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \boldsymbol{e} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{s}}^{+}} (\boldsymbol{s} - \boldsymbol{s}^*) \;. \tag{34.5}$$

Closing the loop and assuming that the robot controller is able to realize perfectly $\boldsymbol{v}_\mathrm{c}$, that is inserting (34.5) into (34.3), we obtain

$$\dot{\boldsymbol{e}} = -\lambda \mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \boldsymbol{e} \;. \tag{34.6}$$

This equation characterizes the actual behavior of the closed-loop system, which is different from the desired one ($\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$) when ever $\mathbf{L}_{\boldsymbol{e}}\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \neq \mathbf{I}_k$. It is also the basis of the stability analysis of the system using Lyapunov theory.

What we have presented above is the basic design implemented by most visual servo controllers. All that remains is to fill in the details. How should $\boldsymbol{s}$ be chosen? What then is the form of $\mathbf{L}_{\boldsymbol{s}}$? How should we

estimate $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$? What are the performance characteristics of the resulting closed-loop system? These questions are addressed in the remainder of the chapter. We first describe the two basic approaches, IBVS and PBVS, whose principles were proposed more than 20 years ago [3]. We then present more-recent approaches that have improved their performance.

34.2 Image-Based Visual Servo

Traditional image-based control schemes [4, 3] use the image-plane normalised coordinates of a set of points to define the vector $\boldsymbol{s}$. The image measurements $\boldsymbol{m}$ are usually the pixel coordinates of the set of image points (although this is not the only possible choice), and the parameters $\boldsymbol{a}$ in the definition of $\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{m}, \boldsymbol{a})$ in (34.1) are nothing but the camera intrinsic parameters to go from image measurements expressed in pixels to the features.

34.2.1 The Interaction Matrix

A three-dimensional world point with coordinates $\boldsymbol{X} = (X, Y, Z)$ in the camera frame projects into the image plane of a conventional perspective camera as a two-dimensional point with normalised coordinates $\boldsymbol{x} = (x, y)$. More precisely we have

$$\begin{cases} x &= X/Z = (u - c_u)/f\alpha \\ y &= Y/Z = (v - c_v)/f \end{cases} , \tag{34.7}$$

where $\boldsymbol{m} = (u, v)$ gives the coordinates of the image point expressed in pixel units, and $\boldsymbol{a} = (c_u, c_v, f, \alpha)$ is the set of camera intrinsic parameters as defined in Chap. 032: 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. The intrinsic parameter β defined in Chap. 032 has been assumed to be 0 here. In this case, we take $\boldsymbol{s} = \boldsymbol{x} = (x, y)$, the image plane coordinates of the point. The details of the imaging geometry and perspective projection can be found in many computer vision texts, including [5, 6, 7].

Taking the time derivative of the projection equations (34.7), we obtain

$$\begin{cases} \dot{x} &= \dot{X}/Z - X\dot{Z}/Z^2 = (\dot{X} - x\dot{Z})/Z \\ \dot{y} &= \dot{Y}/Z - Y\dot{Z}/Z^2 = (\dot{Y} - y\dot{Z})/Z \end{cases} . \tag{34.8}$$

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

$$\dot{\boldsymbol{X}} = -\boldsymbol{v}_{\mathrm{c}} - \omega_{\mathrm{c}} \times \boldsymbol{X} \Leftrightarrow \begin{cases} \dot{X} = -v_x - \omega_y Z + \omega_z Y \\ \dot{Y} = -v_y - \omega_z X + \omega_x Z \\ \dot{Z} = -v_z - \omega_x Y + \omega_y X \end{cases} . \tag{34.9}$$

where $\boldsymbol{v}_{\mathrm{c}} = (v_x, v_y, v_z)$ and $\omega_{\mathrm{c}} = (\omega_x, \omega_y, \omega_z)$. Inserting (34.9) into (34.8), grouping terms, and using (34.7) we obtain

$$\begin{cases} \dot{x} = -v_x/Z + xv_z/Z + xy\omega_x - (1 + x^2)\omega_y + y\omega_z \\ \dot{y} = -v_y/Z + yv_z/Z + (1 + y^2)\omega_x - xy\omega_y - x\omega_z \end{cases} , \tag{34.10}$$

which can be written

$$\dot{\boldsymbol{x}} = \mathbf{L}_{\boldsymbol{x}} \boldsymbol{v}_{\mathrm{c}} , \tag{34.11}$$

where the interaction matrix $\mathbf{L}_{\boldsymbol{x}}$ is given by

$$\mathbf{L}_{\boldsymbol{x}} = \begin{pmatrix} -1/Z & 0 & x/Z & xy & -(1 + x^2) & y \\ 0 & -1/Z & y/Z & 1 + y^2 & -xy & -x \end{pmatrix} . \tag{34.12}$$

In the matrix $\mathbf{L}_{\boldsymbol{x}}$, 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 $\mathbf{L}_{\boldsymbol{x}}^{+}$ cannot be *directly* used in (34.4), and an estimation or an approximation $\widehat{\mathbf{L}_{\boldsymbol{x}}^{+}}$ must be used, as in (34.5). We discuss this in more detail below.

To control the six degrees of freedom, at least three points are necessary (i. e., we require $k \geq 6$). If we use the feature vector $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$, by merely stacking interaction matrices for three points we obtain

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

In this case, there will exist some configurations for which $\mathbf{L}_{\boldsymbol{x}}$ is singular [8]. Furthermore, there exist four distinct camera poses for which $\boldsymbol{e} = \mathbf{0}$, i. e., four global minima exist for the error function $\|\boldsymbol{e}\|$, and it is impossible to differentiate them [9]. For these reasons, more than three points are usually considered.

34.2.2 Approximating the Interaction Matrix

There are several choices available for constructing the estimate $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ to be used in the control law. One popular scheme is of course to choose $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}}^{+}$ if $\mathbf{L}_{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{x}}$ is known, that is if the current depth Z of each point is available [2]. In practice, these parameters must be estimated at each iteration of the control scheme. The basic IBVS methods use classical pose-estimation methods (see Chap. 032 and the beginning of Sect. 34.3). Another popular approach is to choose $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}^*}^{+}$ where $\mathbf{L}_{\boldsymbol{e}^*}$ is the value of $\mathbf{L}_{\boldsymbol{e}}$ for the desired position $\boldsymbol{e} = \boldsymbol{e}^* = 0$ [1]. In this case, $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ 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 $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = (\mathbf{L}_{\boldsymbol{e}}/2 + \mathbf{L}_{\boldsymbol{e}^*}/2)^{+}$ has been proposed [10]. Since $\mathbf{L}_{\boldsymbol{e}}$ is involved in this method, the current depth of each point also has to 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 centered in the image (see Fig. 34.1). We define $\boldsymbol{s}$ 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 accompanying videos show experimental results obtained using an Adept Viper robot arm and the ViSP library [11]. The videos all show the same task being performed from the same initial conditions, and only the control approach varies.

The results obtained by using $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}^*}^{+}$ are given in Fig. 34.2 and Video 34.1. 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).

The results obtained using $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}}^{+}$ are given in Fig. 34.3 and Video 34.2. In this case, the trajectories of the points in the image are almost straight lines, which means that the points never leave the camera's field of view. However the behavior induced in the camera frame is even less satisfactory than for the case of $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}^*}^{+}$. The large camera velocities at the beginning of the servo indicate that the condition number of $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ is high at the start of the trajectory, and the camera trajectory is far from a straight line.

The choice $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = (\mathbf{L}_{\boldsymbol{e}}/2 + \mathbf{L}_{\boldsymbol{e}^*}/2)^{+}$ provides good performance in practice. Indeed, as can be seen in Fig. 34.4, the camera velocity components do not include large oscillations, which provides a smooth trajectory in both the image and in 3-D space (see also Video 34.3).

34.2.3 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 Fig. 34.5 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 $\mathbf{L}_{\boldsymbol{e}}^{+}$ in the control scheme attempts to ensure an exponential decrease of the error $\boldsymbol{e}$. This 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 this 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 com-

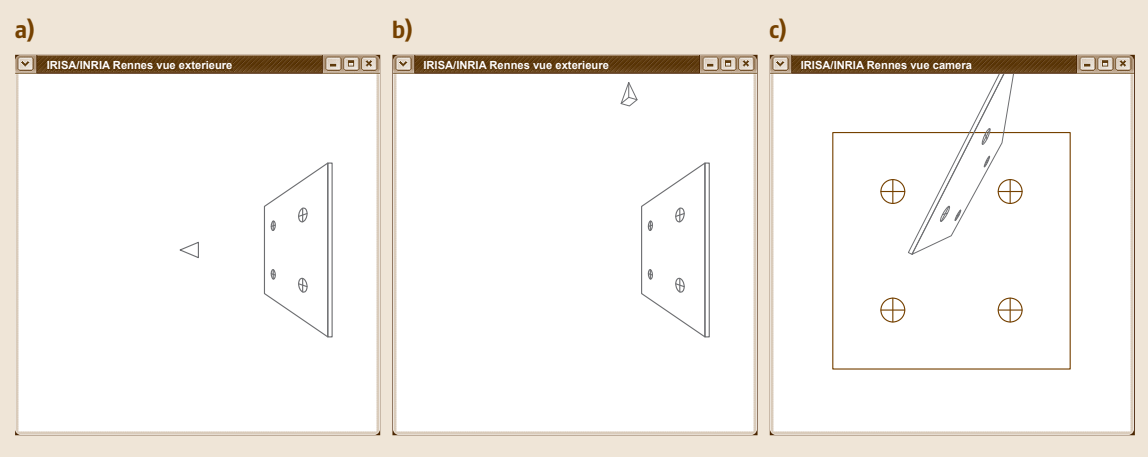

Figure 34.1: Example of positioning task: **(a)** the desired camera pose with respect to a simple target, **(b)** the initial camera pose, and **(c)** the corresponding initial and desired image of the target

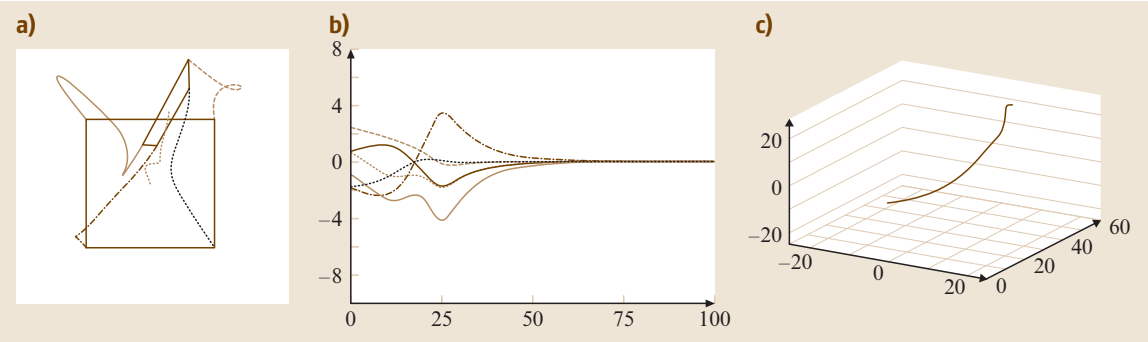

Figure 34.2: IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}^*}^{+}$: **(a)** image point trajectories including the trajectory of the center of the square, which is not used in the control scheme, **(b)** $\boldsymbol{v}_{\mathrm{c}}$ components (cm/s and deg/s) computed at each iteration of the control scheme, and **(c)** the 3-D trajectory of the camera optical center expressed in the desired camera frame $\mathcal{R}_{\mathrm{c}^*}$ (cm)

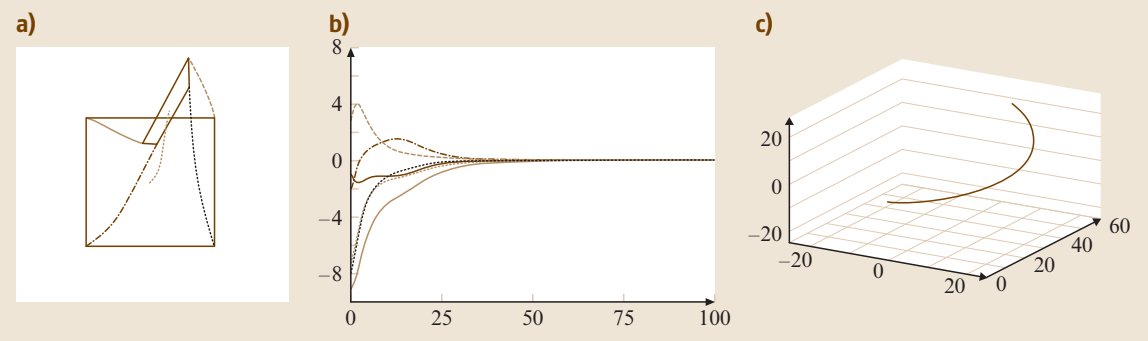

Figure 34.3: IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}}^{+}$

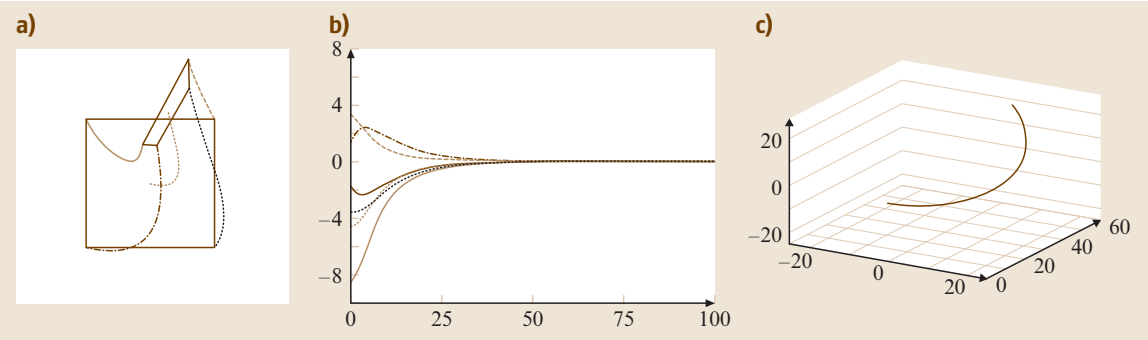

Figure 34.4: IBVS system behavior using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = (\mathbf{L}_{\boldsymbol{e}}/2 + \mathbf{L}_{\boldsymbol{e}}^{*}/2)^{+}$

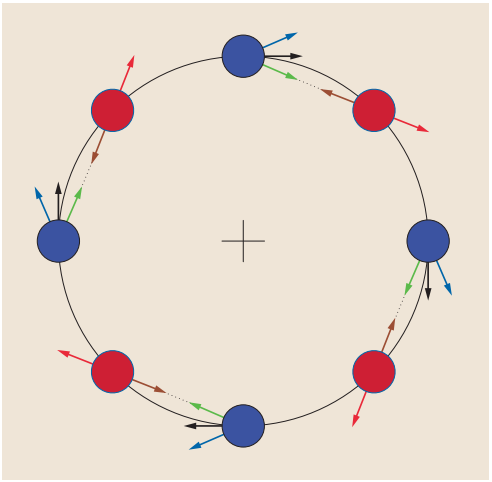

Figure 34.5: Geometrical interpretation of IBVS: going from the blue position to the red one. In *green*, image motion when $\mathbf{L}_{\boldsymbol{e}}^{+}$ is used in the control scheme; in *blue*, when $\mathbf{L}_{\boldsymbol{e}^*}^{+}$ is used; and in *black* when $\left(\mathbf{L}_{\boldsymbol{e}}/2+\mathbf{L}_{\boldsymbol{e}^*}/2\right)^{+}$ is used (see the text for more details)

posed of a rotational motion around the optical axis, but combined with a retreating translational motion along the optical axis [12]. This unexpected motion is due to the choice of the features and the form of the third and sixth columns in the interaction matrix, which induces a coupling between the features and the two degrees of freedom involved (v_z and ω_z). 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 radians where no rotational motion at all will be induced by the control scheme [13]. 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 below, these results are consistent with the local asymptotic stability results that can be obtained for IBVS.

If instead we use $\mathbf{L}_{\boldsymbol{e}^*}^{+}$ in the control scheme, the image motion generated can easily be shown to be the blue one plotted in Fig. 34.5. Indeed, if we consider the same control scheme as before but starting from $\boldsymbol{s}^*$ to reach $\boldsymbol{s}$, we obtain

$$\boldsymbol{v}_{\mathrm{c}} = -\lambda \mathbf{L}_{\boldsymbol{e}^*}^{+}(\boldsymbol{s}^* - \boldsymbol{s}) \;,$$

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

$$\boldsymbol{v}_{\mathrm{c}} = -\lambda \mathbf{L}_{\boldsymbol{e}^*}^{+}(\boldsymbol{s} - \boldsymbol{s}^*) \;,$$

and thus generates the image motion plotted in red at the red points. Transformed to 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 can be done as before for the case of 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 $\mathbf{L}_{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{e}^*}$ when $\boldsymbol{e} = \boldsymbol{e}^*$) 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 $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \left(\mathbf{L}_{\boldsymbol{e}}/2+\mathbf{L}_{\boldsymbol{e}^*}/2\right)^{+}$, it is intuitively clear that considering the mean of $\mathbf{L}_{\boldsymbol{e}}$ and $\mathbf{L}_{\boldsymbol{e}^*}$ generates the image motion plotted in black, even when the error is large. In all cases but the rotation of π radians, the camera motion is now a pure rotation around the optical axis, without any unexpected translational motion.

34.2.4 Stability Analysis

We now consider the fundamental issues related to the stability of IBVS. 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 $\mathcal{L} = \frac{1}{2}\|\boldsymbol{e}(t)\|^2$, whose derivative is given by

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

since $\dot{\boldsymbol{e}}$ is given by (34.6). The global asymptotic stability of the system is thus obtained when the following sufficient condition is satisfied:

$$\mathbf{L}_{\boldsymbol{e}}\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} > 0 \,. \tag{34.13}$$

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 $\mathbf{L}_{\boldsymbol{e}}$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ are of full rank 6, then the condition (34.13) is satisfied if the approximations involved in $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ are not too coarse.

As discussed above, for most IBVS approaches we have $k > 6$. Therefore the condition (34.13) can never be ensured since $\mathbf{L}_{\boldsymbol{e}}\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \in \mathbb{R}^{k\times k}$ is at most of rank 6, and thus $\mathbf{L}_{\boldsymbol{e}}\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ has a nontrivial null space. In this case, configurations such that $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ correspond to local minima. Reaching such a local minimum is illustrated in Fig. 34.6. As can be seen in Fig. 34.6d, each component of $\boldsymbol{e}$ has a nice exponential decrease with the same convergence speed, causing straight-line trajectories to be realized in the image, but the error reached is not exactly zero, and it is clear from Fig. 34.6c 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 $\boldsymbol{e}'$ with $\boldsymbol{e}' = \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}\boldsymbol{e}$. The time derivative of this error is given by

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

where $\boldsymbol{O} \in \mathbb{R}^{6\times 6}$ is equal to $\mathbf{0}$ when $\boldsymbol{e} = 0$, whatever the choice of $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ [14]. Using the control scheme (34.5), we obtain

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

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

$$\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}\mathbf{L}_{\boldsymbol{e}} > 0 \,, \tag{34.14}$$

where $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}\mathbf{L}_{\boldsymbol{e}} \in \mathbb{R}^{6\times 6}$. Indeed, only the linearized system $\dot{\boldsymbol{e}}' = -\lambda\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}\mathbf{L}_{\boldsymbol{e}}\boldsymbol{e}'$ has to be considered if we are interested in the local asymptotic stability [15].

Once again, if the features are chosen and the control scheme designed so that $\mathbf{L}_{\boldsymbol{e}}$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ are of full rank 6, then condition (34.14) is ensured if the approximations involved in $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$ are not too coarse.

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

Even though local asymptotic stability can be ensured when $k > 6$, we recall that global asymptotic stability cannot be ensured. For instance, as illustrated in Fig. 34.6, there may exist local minima corresponding to configurations where $\boldsymbol{e} \in \ker \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}$, which are outside of the neighborhood considered above. Determining the size of the neighborhood in which stability and the convergence are ensured is still an open issue, even if this neighborhood is surprisingly quite large in practice.

34.2.5 IBVS with a Stereo Vision System

It is straightforward to extend the IBVS approach to a multicamera system. If a stereo vision system is used, and a world point is visible in both left and right images (see Fig. 34.7), it is possible to use as visual features

$$\boldsymbol{s} = \boldsymbol{x}_{\boldsymbol{s}} = (\boldsymbol{x}_{\mathrm{l}}, \boldsymbol{x}_{\mathrm{r}}) = (x_{\mathrm{l}}, y_{\mathrm{l}}, x_{\mathrm{r}}, y_{\mathrm{r}})$$

i. e., to represent the point by just stacking in $\boldsymbol{s}$ the x and y coordinates of the observed point in the left

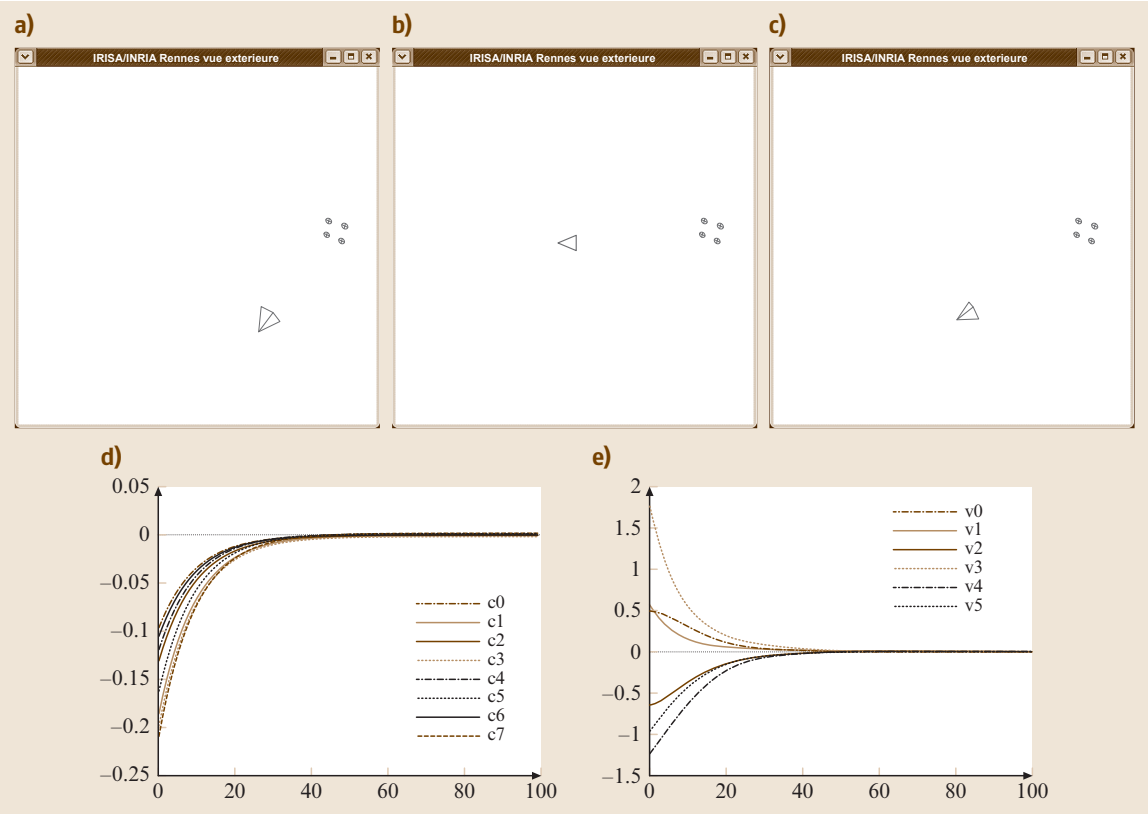

Figure 34.6: IBVS reaching a local minimum using $\boldsymbol{s} = (x_1, y_1, \ldots, x_4, y_4)$ and $\widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} = \mathbf{L}_{\boldsymbol{e}}^{+}$: **(a)** the initial configuration, **(b)** the desired one, **(c)** the configuration reached after the convergence of the control scheme, **(d)** the evolution of the error $\boldsymbol{e}$ at each iteration of the control scheme, and **(e)** the evolution of the six components of the camera velocity $\boldsymbol{v}_{\mathrm{c}}$

and right images [16]. However, care must be taken when constructing the corresponding interaction matrix since the form given in (34.11) is expressed in either the left or right camera frame. More precisely, we have

$$\begin{cases} \dot{\boldsymbol{x}}_{\mathrm{l}} &= \mathbf{L}_{\boldsymbol{x}_{\mathrm{l}}}\, \boldsymbol{v}_{\mathrm{l}} \ , \\ \dot{\boldsymbol{x}}_{\mathrm{r}} &= \mathbf{L}_{\boldsymbol{x}_{\mathrm{r}}}\, \boldsymbol{v}_{\mathrm{r}} \ , \end{cases}$$

where $\boldsymbol{v}_{\mathrm{l}}$ and $\boldsymbol{v}_{\mathrm{r}}$ are the spatial velocity of the left and right camera, respectively, and where the analytical form of $\mathbf{L}_{\boldsymbol{x}_{\mathrm{l}}}$ and $\mathbf{L}_{\boldsymbol{x}_{\mathrm{r}}}$ are given by (34.12).

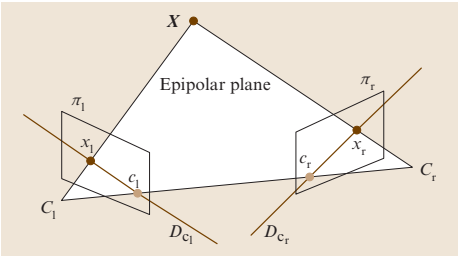

Figure 34.7: A stereo vision system

By choosing a sensor frame rigidly linked to the stereo vision system, we obtain

$$\dot{\boldsymbol{x}}_{\boldsymbol{s}} = \begin{pmatrix} \dot{\boldsymbol{x}}_{\mathrm{l}} \\ \dot{\boldsymbol{x}}_{\mathrm{r}} \end{pmatrix} = \mathbf{L}_{\boldsymbol{x}_{\boldsymbol{s}}} \boldsymbol{v}_{\boldsymbol{s}} \ ,$$

where the interaction matrix related to $\boldsymbol{x}_{\boldsymbol{s}}$ can be determined using the spatial motion transform matrix $\mathbf{V}$ defined in Chap. 002 to transform velocities expressed in the left or right cameras frames to the sensor frame. We recall that $\mathbf{V}$ is given by

$$\mathbf{V} = \begin{pmatrix} \mathbf{R} & [\boldsymbol{t}]_{\times}\mathbf{R} \\ \mathbf{0} & \mathbf{R} \end{pmatrix} \ , \tag{34.15}$$

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

$$\mathbf{L}_{\boldsymbol{x}_{\boldsymbol{s}}} = \begin{pmatrix} \mathbf{L}_{\boldsymbol{x}_{\mathrm{l}}}\, {}^{\mathrm{l}}\mathbf{V}_{\boldsymbol{s}} \\ \mathbf{L}_{\boldsymbol{x}_{\mathrm{r}}}\, {}^{\mathrm{r}}\mathbf{V}_{\boldsymbol{s}} \end{pmatrix} \ .$$

Note that $\mathbf{L}_{\boldsymbol{x}_{\boldsymbol{s}}} \in \mathbb{R}^{4\times 6}$ is always of rank 3 because of the epipolar constraint that links the perspective projection of a 3-D point in a stereo vision system (see Fig. 34.7). Another simple interpretation is that a 3-D point is represented by three independent parameters, which makes it impossible to find more than three independent parameters using any sensor observing that point.

To control the six degrees of freedom of the system, it is necessary to consider at least three points, as the rank of the interaction matrix considering only two points is 5.

Using a stereo vision 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 $\boldsymbol{s}$. Such an approach would be, strictly speaking, a position-based approach, since it would require 3-D parameters in $\boldsymbol{s}$.

34.2.6 IBVS with Cylindrical Coordinates of Image Points

In the previous sections, we have considered the Cartesian coordinates of image points. As proposed in [17] it may be useful to consider instead the cylindrical coordinates $\boldsymbol{\gamma} = (\rho, \theta)$ of the image points instead of their Cartesian coordinates $\boldsymbol{x} = (x, y)$. They are given by

$$\rho = \sqrt{x^2 + y^2} \ , \ \theta = \arctan \frac{y}{x}$$

from which we deduce

$$\dot{\rho} = (x\dot{x} + y\dot{y})/\rho \ , \ \dot{\theta} = (x\dot{y} - y\dot{x})/\rho^2 \ .$$

Using (34.11) and then substituting x by $\rho\cos\theta$ and y by $\rho\sin\theta$, we obtain immediately

$$\mathbf{L}_{\boldsymbol{\gamma}} = \begin{pmatrix} \frac{-c}{Z} & \frac{-s}{Z} & \frac{\rho}{Z} & (1+\rho^2)s & -(1+\rho^2)c & 0 \\ \frac{s}{\rho Z} & \frac{-c}{\rho Z} & 0 & \frac{c}{\rho} & \frac{s}{\rho} & -1 \end{pmatrix} \ , \tag{34.16}$$

where $c = \cos\theta$ and $s = \sin\theta$. Note that θ is not defined when the image point lies at the principal point (where $x = y = \rho = 0$). It is thus not surprising that the interaction matrix $\mathbf{L}_{\boldsymbol{\gamma}}$ is singular in that case.

If we go back to the example depicted in Fig. 34.5, the behavior obtained using cylindrical coordinates will be the expected one, that is a pure rotation around the optical axis, by using either $\mathbf{L}_{\boldsymbol{e}}^{+}$, $\mathbf{L}_{\boldsymbol{e}^*}^{+}$ or $(\mathbf{L}_{\boldsymbol{e}}/2 + \mathbf{L}_{\boldsymbol{e}^*}/2)^{+}$ in the control scheme. This is due to the form of the third and sixth columns of the interaction matrix (34.16), which leads to a decoupled system.

34.2.7 IBVS with Other Geometrical Features

In the previous sections, we have only considered image point coordinates in $\boldsymbol{s}$. Other geometrical primitives can of course be used. There are several reasons to do so. Firstly, the scene observed by the camera cannot always be described merely by a collection of points, in which case the image processing provides other types of measurements, such as a set of straight lines or the contours of an object. Secondly, richer geometric primitives may ameliorate the decoupling and linearizing issues that motivate the design of partitioned systems (see Sect. 34.4). Finally, the robotic task to be achieved may be expressed in terms of virtual linkages (or fixtures) between the camera and the observed objects [18, 19], sometimes expressed directly by constraints between primitives, such as point-to-line [20] (which means that an observed point must lie on a specified line).

It is possible to determine the interaction matrix related to the perspective projection of a large class of geometrical primitives, such as segments, straight lines, spheres, circles, and cylinders. The results are given in [1] and [18]. Recently, the analytical form of the interaction matrix related to any image moments corresponding to planar objects has been computed. This makes it possible to consider planar objects of any shape [21]. If a collection of points is measured in the image, moments can also be used [22]. In both cases, moments allow the use of intuitive geometrical features, such as the center of gravity or the orientation of an object. By selecting an adequate combination of moments, it is then possible to determine partitioned systems with good decoupling and linearizing properties [21, 22].

Note that, for all these features (geometrical primitives, moments), the depth of the primitive or of the object considered appears in the coefficients of the interaction matrix related to the translational degrees of freedom, as was the case for the image points. An estimation of this depth is thus generally necessary (see Sect. 34.6). In a few situations, for instance with a suitable normalization of moments [22], only the constant desired depth appears in the interaction matrix, which makes estimating depth unnecessary.

34.2.8 Non-perspective Cameras

The vast majority of cameras we use, and our own eyes, are characterized by a perspective projection model which closely approximate the ideal pinhole imaging model. Such cameras have a narrow field of view, typically less than one half hemisphere. For robotics it is often advantageous to have a large field of view and this can be achieved using a fisheye lens camera or a catadioptric (lens and mirror system) camera (often referred to as a panoramic camera). For these non-perspective sensors the interaction matrix of any visual feature has a different form to those discussed above, such as (34.12) and (34.16) for an image point.

Rather than determine the interaction matrix of visual features expressed in the image plane of non-perspective cameras, we can transform the images from these cameras to the view that would be seen by an ideal spherical camera. The spherical model projects world points onto a unit sphere, the intersection of the unit sphere with the ray from the world point to the centre of the sphere. Such an ideal camera has the largest possible field of view. The unified imaging model [23] shown in Figure 34.8 provides a general mechanism to project a world point to the image plane of a large class of cameras. To be precise it includes all central projection cameras and this includes perspective and some catadioptric cameras with particular mirror shapes, but in practice it is a very good approximation to non-central cameras including fisheye and general catadioptric systems. The mechanism of this unified model can also be used to reproject points from these varied image planes to a spherical camera.

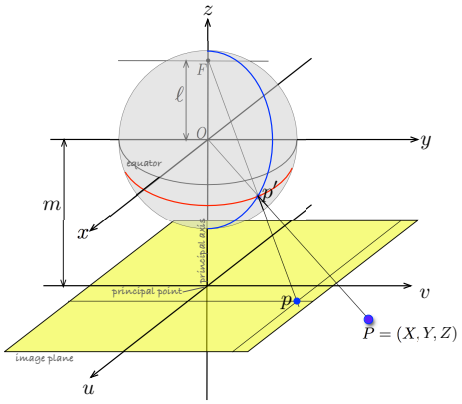

Figure 34.8: Unified imaging model of Geyer and Daniilidis (from [7] with permission)

Referring to Figure 34.8 the world point $\mathbf{P}$ is represented by the vector $\boldsymbol{X} = (X, Y, Z)$ in the camera frame, and is projected onto the surface of the unit sphere at the point $\boldsymbol{x}_s = (x_s, y_s, z_s)$ with

$$x_s = \frac{X}{R}, \quad y_s = \frac{Y}{R}, \text{ and } \quad z_s = \frac{Z}{R}$$

where $R = \sqrt{X^2 + Y^2 + Z^2}$ is the distance from the sphere centre to the world point.

The interaction matrix of $\boldsymbol{x}_s$ is derived in essentially the same manner as for the perspective camera, and it can be shown to be [24]

$$\mathbf{L}_{\boldsymbol{x}_s} = \begin{pmatrix} (x_s^2 - 1)/R & x_s y_s / R & x_s z_s / R & 0 \\ x_s y_s / R & (y_s^2 - 1)/R & y_s z_s / R & z_s \\ x_s z_s / R & y_s z_s / R & (z_s^2 - 1)/R & -y_s \\ 0 & -z_s & y_s & -x_s \\ z_s & 0 & -x_s & -y_s \end{pmatrix}. \tag{34.17}$$

Note that R can be expressed as a function of the point depth Z by using $R = Z\sqrt{1 + x_s^2 + y_s^2}$. Therefore, the general spherical model does not add any supplementary unknown in the interaction matrix.

A particular advantage of the spherical model is that for pure camera rotation the shape of an object is invariant, which eases determining visual features that are only linked to translational motions.

34.2.9 Direct Estimation

In the previous sections, we have focused on the analytical form of the interaction matrix. It is also possible to estimate its numerical value directly using either an offline learning step, or an online estimation scheme.

All the methods proposed to estimate the interaction matrix numerically rely on the observation of a variation of the features due to a known or measured camera motion. More precisely, if we measure a feature's variation $\Delta\boldsymbol{s}$ due to a camera motion $\Delta\boldsymbol{v}_c$, we have from (34.2):

$$\mathbf{L}_{\boldsymbol{s}} \Delta\boldsymbol{v}_c = \Delta\boldsymbol{s} \ ,$$

which provides k equations while we have $k \times 6$ unknown values in $\mathbf{L}_{\boldsymbol{s}}$. Using a set of N independent camera motions with $N > 6$, it is thus possible to estimate $\mathbf{L}_{\boldsymbol{s}}$ by solving

$$\mathbf{L}_{\boldsymbol{s}} \boldsymbol{A} = \boldsymbol{B} \ ,$$

where the columns of $\boldsymbol{A} \in \mathbb{R}^{6\times N}$ and $\boldsymbol{B} \in \mathbb{R}^{k\times N}$ are, respectively, formed from the set of camera motions and the set of corresponding features variations. The least-square solution is of course given by

$$\widehat{\mathbf{L}_{\boldsymbol{s}}} = \boldsymbol{B}\boldsymbol{A}^+ \ . \tag{34.18}$$

Methods based on neural networks have also been developed to estimate $\mathbf{L}_{\boldsymbol{s}}$ [25, 26]. It is also possible to estimate the numerical value of $\mathbf{L}_{\boldsymbol{s}}^+$ directly, which in practice provides a better behavior [27]. In this case, the basic relation is

$$\mathbf{L}_{\boldsymbol{s}}^+ \Delta\boldsymbol{s} = \Delta\boldsymbol{v}_c \ ,$$

which provides six equations. Using a set of N measurements, with $N > k$, we now obtain

$$\widehat{\mathbf{L}_{\boldsymbol{s}}^+} = \boldsymbol{A}\boldsymbol{B}^+ \ . \tag{34.19}$$

In the first case (34.18), the six columns of $\mathbf{L}_{\boldsymbol{s}}$ are estimated by solving six linear systems, while in the second case (34.19), the k columns of $\mathbf{L}_{\boldsymbol{s}}^+$ are estimated by solving k linear systems, which explains the difference in the results.

Estimating the interaction matrix online can be viewed as an optimization problem, and consequently a number of researchers have investigated approaches

that derive from optimization methods. These methods typically discretize the system equation (34.2), and use an iterative updating scheme to refine the estimate of $\widehat{\mathbf{L}_{\boldsymbol{s}}}$ at each stage. One such online and iterative formulation uses the Broyden update rule given by [28, 29]:

$$\widehat{\mathbf{L}_{\boldsymbol{s}}}(t+1) = \widehat{\mathbf{L}_{\boldsymbol{s}}}(t) + \frac{\alpha}{\Delta \boldsymbol{v}_{\mathrm{c}}^{\top} \Delta \boldsymbol{v}_{\mathrm{c}}} \left[\Delta \boldsymbol{x} - \widehat{\mathbf{L}_{\boldsymbol{s}}}(t) \Delta \boldsymbol{v}_{\mathrm{c}}\right] \Delta \boldsymbol{v}_{\mathrm{c}}^{\top} \ ,$$

where α defines the update speed. This method has been generalized to the case of moving objects in [30].

The main interest of using such numerical estimations in the control scheme is that it avoids all the modeling and calibration steps. It is particularly useful when using features whose interaction matrix is not available in analytical form. For instance, in [31], the main eigenvalues of the principal component analysis of an image have been considered in a visual servoing scheme. The drawback of these methods is that no theoretical stability and robustness analysis can be made.

34.3 Pose-Based Visual Servo

Pose-based control schemes (PBVS) [3, 32, 33] use the pose of the camera with respect to some reference coordinate frame to define $\boldsymbol{s}$. Computing this 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 classic computer vision problem is called the 3-D localization problem. While this problem is beyond the scope of the present chapter, many solutions have been presented in the literature [34, 35] and its basic principles are recalled in Chap. 032.

It is then typical to define $\boldsymbol{s}$ in terms of the parameterization used to represent the camera pose. Note that the parameters $\boldsymbol{a}$ involved in the definition (34.1) of $\boldsymbol{s}$ 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 $\mathcal{F}_{\mathrm{c}}$, the desired camera frame $\mathcal{F}_{\mathrm{c}^*}$, and a reference frame $\mathcal{F}_{\mathrm{o}}$ 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 ${}^{\mathrm{c}}\boldsymbol{t}_{\mathrm{o}}$ and ${}^{\mathrm{c}^*}\boldsymbol{t}_{\mathrm{o}}$ 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 $\mathbf{R} = {}^{\mathrm{c}^*}\mathbf{R}_{\mathrm{c}}$ be the rotation matrix that gives the orientation of the current camera frame relative to the desired frame.

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

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

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} -\mathbf{I}_3 & [{}^{\mathrm{c}}\boldsymbol{t}_{\mathrm{o}}]_{\times} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}} \end{pmatrix} \ , \tag{34.20}$$

in which $\mathbf{I}_3$ is the 3×3 identity matrix and $\mathbf{L}_{\theta\boldsymbol{u}}$ is given by [36]

$$\mathbf{L}_{\theta\boldsymbol{u}} = \mathbf{I}_3 - \frac{\theta}{2}[\boldsymbol{u}]_{\times} + \left(1 - \frac{\operatorname{sinc}\theta}{\operatorname{sinc}^2 \frac{\theta}{2}}\right)[\boldsymbol{u}]_{\times}^2 \ , \tag{34.21}$$

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

Following the development in Sect. 34.1, we obtain the control scheme

$$\boldsymbol{v}_{\mathrm{c}} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} \boldsymbol{e}$$

since the dimension k of $\boldsymbol{s}$ is six, that is, the number of camera degrees of freedom. By setting

$$\widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} = \begin{pmatrix} -\mathbf{I}_3 & [{}^{\mathrm{c}}\boldsymbol{t}_{\mathrm{o}}]_{\times} \mathbf{L}_{\theta\boldsymbol{u}}^{-1} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}}^{-1} \end{pmatrix} \ , \tag{34.22}$$

we obtain after simple developments:

$$\begin{cases} \boldsymbol{v}_{\mathrm{c}} &= -\lambda\left[{}^{c^*}\boldsymbol{t}_{\mathrm{o}} - {}^{c}\boldsymbol{t}_{\mathrm{o}}\right] + \left[{}^{\mathrm{c}}\boldsymbol{t}_{\mathrm{o}}\right]_{\times} \theta\boldsymbol{u} \\ \omega_{\mathrm{c}} &= -\lambda\theta\boldsymbol{u} \end{cases} . \tag{34.23}$$

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

Ideally, that is, if the pose parameters are perfectly estimated, the behavior of $\boldsymbol{e}$ will be the expected one

($\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$). The choice of $\boldsymbol{e}$ causes the rotational motion to follow a geodesic with an exponential decreasing speed and causes the translational parameters involved in $\boldsymbol{s}$ to decrease at the same speed. This explains the nice exponential decrease of the camera velocity components in Fig. 34.9. Furthermore, the trajectory in the image of the origin of the object frame follows a 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 (see Video 34.4).

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

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}} \end{pmatrix} . \tag{34.24}$$

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

$$\begin{cases} \boldsymbol{v}_{\mathrm{c}} &= -\lambda \mathbf{R}^{\top}\, {}^{c^*}\boldsymbol{t}_{\mathrm{c}} \\ \omega_{\mathrm{c}} &= -\lambda\, \theta\boldsymbol{u} \end{cases} . \tag{34.25}$$

In this case, as can be seen in Fig. 34.10 and in Video 34.5, if the pose parameters involved in (34.25) are estimated perfectly, the camera trajectory is a straight line, while the image trajectories are less satisfactory than before. Some particular configurations can be found which will lead to some points leaving the camera field of view during the robot motion.

The stability properties of PBVS seem quite attractive. Since $\mathbf{L}_{\theta\boldsymbol{u}}$ given in (34.21) is nonsingular when $\theta \neq 2k\pi$, $\forall k \in Z^*$, we obtain from (34.13) the global asymptotic stability of the system since $\mathbf{L}_{\boldsymbol{e}}\widehat{\mathbf{L}_{\boldsymbol{e}}^{-1}} = \mathbf{I}_6$, under the strong hypothesis that all the pose parameters are perfect. This is true for both methods presented above, since the interaction matrices given in (34.20) and (34.24) are full rank when $\mathbf{L}_{\theta\boldsymbol{u}}$ 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 method presented in Sect. 34.3 (the analysis for the second method is analogous), the interaction matrix given in (34.20) 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 [13]. The true positivity condition (34.13) should in fact be written:

$$\mathbf{L}_{\widehat{\boldsymbol{e}}}\widehat{\mathbf{L}_{\widehat{\boldsymbol{e}}}^{-1}} > 0 , \tag{34.26}$$

where $\widehat{\mathbf{L}_{\widehat{\boldsymbol{e}}}^{-1}}$ is given by (34.22) but where $\mathbf{L}_{\widehat{\boldsymbol{e}}}$ is unknown, and not given by (34.20). Indeed, even small errors in computing the position of points in the image can lead to pose errors which will significantly impact the accuracy and the stability of the system (see Fig. 34.11).

34.4 Advanced Approaches

34.4.1 Hybrid VS

Suppose we have access to a control law for ω_{c}, such as the one used in PBVS [see (34.23) or (34.25)]:

$$\omega_{\mathrm{c}} = -\lambda\, \theta\boldsymbol{u} . \tag{34.27}$$

How could we use this in conjunction with IBVS?

Considering a feature vector $\boldsymbol{s}_{\mathrm{t}}$ and an error $\boldsymbol{e}_{\mathrm{t}}$ devoted to control the translational degrees of freedom, we can partition the interaction matrix as follows

$$\begin{aligned} \dot{\boldsymbol{s}}_{\mathrm{t}} &= \mathbf{L}_{\boldsymbol{s}_{\mathrm{t}}}\boldsymbol{v}_{\mathrm{c}} \\ &= \begin{pmatrix} \mathbf{L}_{\boldsymbol{v}} & \mathbf{L}_{\omega} \end{pmatrix} \begin{pmatrix} \boldsymbol{v}_{\mathrm{c}} \\ \omega_{\mathrm{c}} \end{pmatrix} \\ &= \mathbf{L}_{\boldsymbol{v}}\boldsymbol{v}_{\mathrm{c}} + \mathbf{L}_{\omega}\omega_{\mathrm{c}} . \end{aligned}$$

Now, setting $\dot{\boldsymbol{e}}_{\mathrm{t}} = -\lambda \boldsymbol{e}_{\mathrm{t}}$, we can solve for the desired translational control input as

$$\begin{aligned} -\lambda\boldsymbol{e}_{\mathrm{t}} = \dot{\boldsymbol{e}}_{\mathrm{t}} = \dot{\boldsymbol{s}}_{\mathrm{t}} = \mathbf{L}_{\boldsymbol{v}}\boldsymbol{v}_{\mathrm{c}} + \mathbf{L}_{\omega}\omega_{\mathrm{c}} , \\ \Rightarrow \boldsymbol{v}_{\mathrm{c}} = -\mathbf{L}_{\boldsymbol{v}}^{+}(\lambda\boldsymbol{e}_{\mathrm{t}} + \mathbf{L}_{\omega}\omega_{\mathrm{c}}) . \end{aligned} \tag{34.28}$$

We can think of the quantity $(\lambda\boldsymbol{e}_{\mathrm{t}} + \mathbf{L}_{\omega}\omega_{\mathrm{c}})$ as a modified error term, one that combines the original error with the error that would be induced by the rotational motion due to ω_{c}. The translational control input $\boldsymbol{v}_{\mathrm{c}} = -\mathbf{L}_{\boldsymbol{v}}^{+}(\lambda\boldsymbol{e}_{\mathrm{t}} + \mathbf{L}_{\omega}\omega_{\mathrm{c}})$ will drive this error to zero. The method known as 2.5-D visual

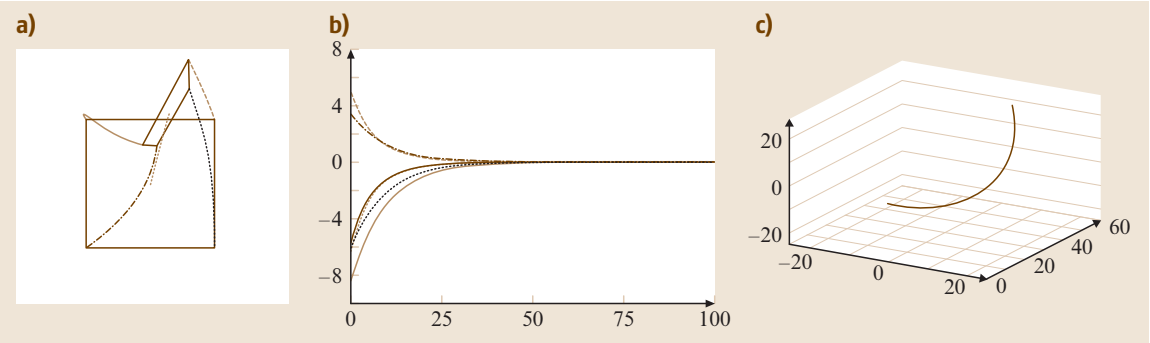

Figure 34.9: PBVS system behavior using $\boldsymbol{s} = ({}^{\mathrm{c}}\boldsymbol{t}_{\mathrm{o}}, \theta\boldsymbol{u})$

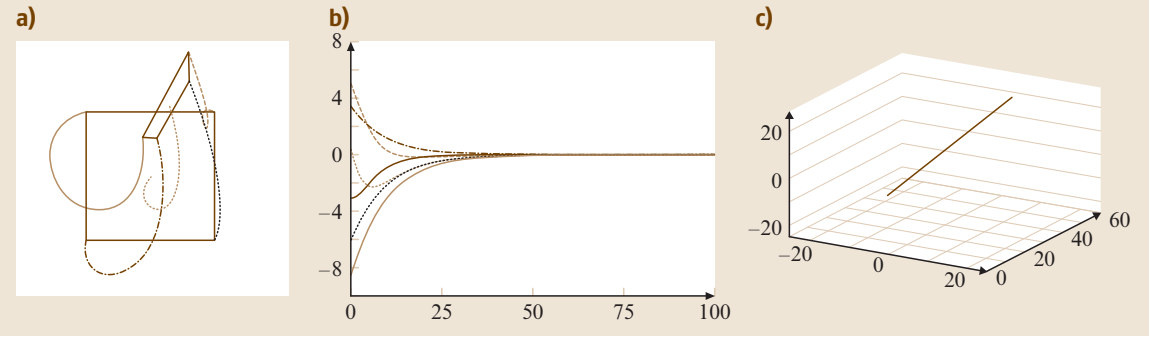

Figure 34.10: PBVS system behavior using $\boldsymbol{s} = ({}^{\mathrm{c}^*}\boldsymbol{t}_{\mathrm{c}}, \theta\boldsymbol{u})$

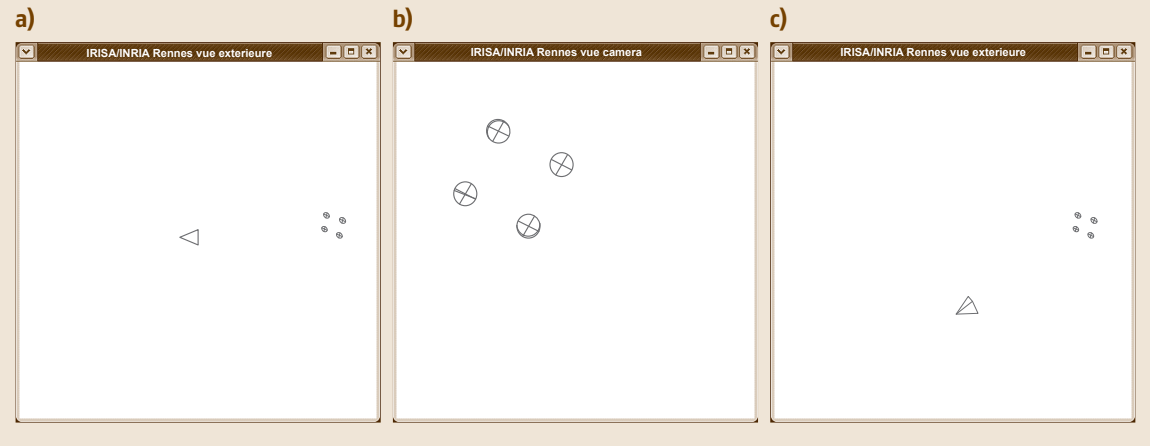

Figure 34.11: Two different camera poses **(a, c)** that provide almost the same image of four coplanar points shown overlaid in **(b)**

servo [36] was the first to exploit such a partitioning in combining IBVS and PBVS. More precisely, in [36], $\boldsymbol{s}_{\mathrm{t}}$ has been selected as the coordinates of an image point, and the logarithm of its depth, so that $\mathbf{L}_{\boldsymbol{v}}$ is a triangular always invertible matrix. More precisely, we have $\boldsymbol{s}_{\mathrm{t}} = (\boldsymbol{x}, \log Z)$, $\boldsymbol{s}_{\mathrm{t}}^* = (\boldsymbol{x}^*, \log Z^*)$, $\boldsymbol{e}_{\mathrm{t}} = (\boldsymbol{x} - \boldsymbol{x}^*, \log \rho_Z)$ where $\rho_Z = Z/Z^*$, and

$$\mathbf{L}_{\boldsymbol{v}} = \frac{1}{Z^* \rho_Z} \begin{pmatrix} -1 & 0 & x \\ 0 & -1 & y \\ 0 & 0 & -1 \end{pmatrix}$$

$$\mathbf{L}_{\boldsymbol{\omega}} = \begin{pmatrix} xy & -(1+x^2) & y \\ 1+y^2 & -xy & -x \\ -y & x & 0 \end{pmatrix} .$$

Note that the ratio ρ_Z can be obtained directly from the partial pose estimation algorithm that will be described in Sect. 34.6.

If we come back to the usual global representation of visual servo control schemes, we have $\boldsymbol{e} = (\boldsymbol{e}_{\mathrm{t}}, \theta\boldsymbol{u})$ and $\mathbf{L}_{\boldsymbol{e}}$ given by

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} \mathbf{L}_{\boldsymbol{v}} & \mathbf{L}_{\boldsymbol{\omega}} \\ \mathbf{0} & \mathbf{L}_{\theta\boldsymbol{u}} \end{pmatrix} ,$$

from which we immediately obtain the control law (34.27) and (34.28) by applying (34.5).

The behavior obtained using this choice for $\boldsymbol{s}_{\mathrm{t}}$ is shown in Fig. 34.12 and Video 34.6. Here, the point that has been considered in $\boldsymbol{s}_{\mathrm{t}}$ is the center of gravity $\boldsymbol{x}_{\mathrm{g}}$ of the target. We note the image trajectory of that point, which is a straight line as expected, and the nice decreasing of the camera velocity components, which makes this scheme very similar to the first PBVS one.

As for stability, it is clear that this scheme is globally asymptotically stable in perfect conditions. Furthermore, thanks to the triangular form of the interaction matrix $\mathbf{L}_{\boldsymbol{e}}$, it is possible to analyze the stability of this scheme in the presence of calibration errors using the partial pose-estimation algorithm that will be described in Sect. 34.6 [37]. Finally, the only unknown constant parameter involved in this scheme, that is Z^*, can be estimated online using adaptive techniques [38].

Other hybrid schemes can be designed. For instance, in [39], the third component of $\boldsymbol{s}_{\mathrm{t}}$ is different and has been selected so that all the target points remain in the camera field of view as far as possible. Another example has been proposed in [40]. In that case, $\boldsymbol{s}$ is selected as $\boldsymbol{s} = ({}^{\mathrm{c}^*}\boldsymbol{t}_{\mathrm{c}}, \boldsymbol{x}_{\mathrm{g}}, \theta u_z)$ which provides with a block-triangular interaction matrix of the form:

$$\mathbf{L}_{\boldsymbol{e}} = \begin{pmatrix} \mathbf{R} & \mathbf{0} \\ \mathbf{L}'_{\boldsymbol{v}} & \mathbf{L}'_{\boldsymbol{\omega}} \end{pmatrix}$$

where $\mathbf{L}'_{\boldsymbol{v}}$ and $\mathbf{L}'_{\boldsymbol{\omega}}$ can easily be computed. This scheme is such that, under perfect conditions, the camera trajectory is a straight line (since ${}^{\mathrm{c}^*}\boldsymbol{t}_{\mathrm{c}}$ is a part of $\boldsymbol{s}$), and the image trajectory of the center of gravity of the object is also a straight line (since $\boldsymbol{x}_{\mathrm{g}}$ is also a part of $\boldsymbol{s}$). The translational camera degrees of freedom are devoted to realize the 3-D straight line, while the rotational camera degrees of freedom are devoted to realize the 2-D straight line and also compensate the 2-D motion of $\boldsymbol{x}_{\mathrm{g}}$ due to the translational motion. As can be seen in Fig. 34.13 and Video 34.7, this scheme is particularly satisfactory in practice.

Finally, it is possible to combine 2-D and 3-D features in different ways. For instance, in [41], it has been proposed to use in $\boldsymbol{s}$ the 2-D homogeneous coordinates of a set of image points expressed in pixels multiplied by their corresponding depth: $\boldsymbol{s} = (u_1 Z_1, v_1 Z_1, Z_1, \cdots, u_n Z_n, v_n Z_n, Z_n)$. As for classical IBVS, we obtain in this case a set of redundant features, since at least three points have to be used to control the six camera degrees of freedom (here $k \geq 9$). However, it has been demonstrated in [42] that this selection of redundant features is free of attractive local minima.

34.4.2 Partitioned VS

The hybrid visual servo schemes described above have been designed to decouple the rotational motions from the translational ones by selecting adequate visual features defined in part in 2-D, and in part in 3-D (which is why they have been called 2.5-D visual servoing). This work has inspired some researchers

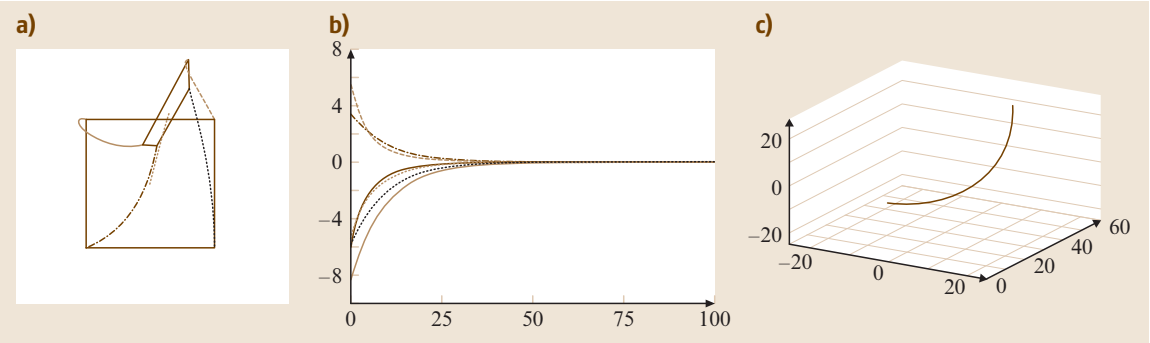

Figure 34.12: 2.5D VS system behavior using $\boldsymbol{s} = (\mathbf{x}_{\mathrm{g}}, \log(Z_{\mathrm{g}}), \theta\boldsymbol{u})$

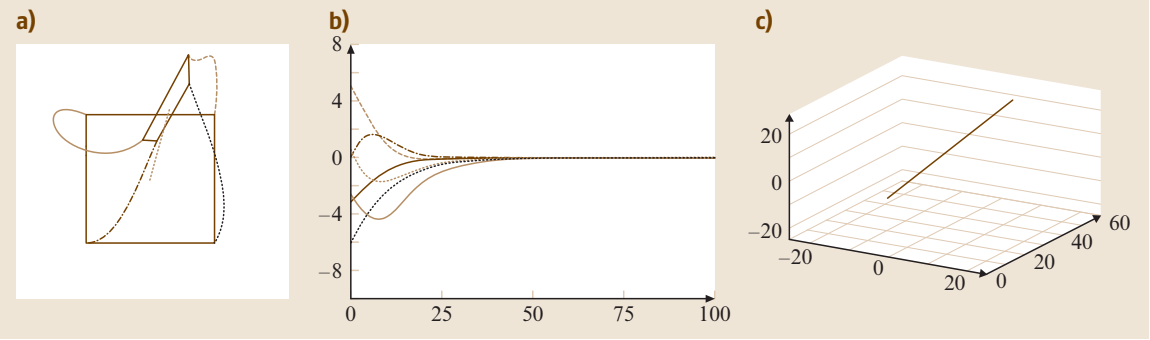

Figure 34.13: 2.5D VS system behavior using $\boldsymbol{s} = ({}^{\mathrm{c}^*}\mathbf{t}_{\mathrm{c}}, \mathbf{x}_{\mathrm{g}}, \theta u_z)$

to find features that exhibit similar decoupling properties but using only features expressed directly in the image. More precisely, the goal is to find six features such that each is related to only one degree of freedom (in which case the interaction matrix is a diagonal matrix). The Grail is to find a diagonal interaction matrix whose elements are constant, as near as possible to the identity matrix, leading to a pure, direct, and simple linear control problem.

The first work in this area partitioned the interaction matrix to isolate motion related to the optic axis [12]. Indeed, whatever the choice of $\boldsymbol{s}$, we have

$$\begin{aligned}\dot{\boldsymbol{s}} &= \mathbf{L}_{\boldsymbol{s}}\boldsymbol{v}_{\mathrm{c}} \\ &= \mathbf{L}_{xy}\boldsymbol{v}_{xy} + \mathbf{L}_z\boldsymbol{v}_z \\ &= \dot{\boldsymbol{s}}_{xy} + \dot{\boldsymbol{s}}_z\end{aligned}$$

in which $\mathbf{L}_{xy}$ includes the first, second, fourth, and fifth columns of $\mathbf{L}_{\boldsymbol{s}}$, and $\mathbf{L}_z$ includes the third and sixth columns of $\mathbf{L}_{\boldsymbol{s}}$. Similarly, $\boldsymbol{v}_{xy} = (v_x, v_y, \omega_x, \omega_y)$ and $\boldsymbol{v}_z = (v_z, \omega_z)$. Here, $\dot{\boldsymbol{s}}_z = \mathbf{L}_z\boldsymbol{v}_z$ gives the component of $\dot{\boldsymbol{s}}$ due to the camera motion along and rotation about the optic axis, while $\dot{\boldsymbol{s}}_{xy} = \mathbf{L}_{xy}\boldsymbol{v}_{xy}$ gives the component of $\dot{\boldsymbol{s}}$ due to velocity along and rotation about the camera x and y axes.

Proceeding as above, by setting $\dot{\boldsymbol{e}} = -\lambda\boldsymbol{e}$ we obtain

$$-\lambda\boldsymbol{e} = \dot{\boldsymbol{e}} = \dot{\boldsymbol{s}} = \mathbf{L}_{xy}\boldsymbol{v}_{xy} + \mathbf{L}_z\boldsymbol{v}_z,$$

which leads to

$$\boldsymbol{v}_{xy} = -\mathbf{L}_{xy}^{+}[\lambda\boldsymbol{e}(t) + \mathbf{L}_z\boldsymbol{v}_z] \; .$$

As before, we can consider $[\lambda\boldsymbol{e}(t) + \mathbf{L}_z\boldsymbol{v}_z]$ as a modified error that incorporates the original error while taking into account the error that will be induced by $\boldsymbol{v}_z$.

Given this result, all that remains is to choose $\boldsymbol{s}$ and $\boldsymbol{v}_z$. As for basic IBVS, the coordinates of a collection of image points can be used in $\boldsymbol{s}$, while two new image features can be defined to determine $\boldsymbol{v}_z$.

- Define α, with $0 \leq \alpha < 2\pi$, as the angle between the horizontal axis of the image plane and the directed line segment joining two feature points.

  It is clear that α is closely related to the rotation around the optic axis.

- Define σ^2 to be the area of the polygon defined by these points. Similarly, σ^2 is closely related to the translation along the optic axis.

Using these features, $\boldsymbol{v}_z$ has been defined in [12] as

$$\begin{cases} v_z &= \lambda_{v_z} \ln \frac{\sigma^*}{\sigma} \; , \\ \omega_z &= \lambda_{\omega_z}(\alpha^* - \alpha) \; . \end{cases}$$

34.5 Performance Optimization and Planning

In some sense, partitioned methods represent an effort to optimize system performance by assigning distinct features and controllers to individual degrees of freedom. In this way, the designer performs a sort of offline optimization when allocating controllers to degrees of freedom. It is also possible to explicitly design controllers that optimize various system performance measures. We describe a few of these in this section.

34.5.1 Optimal Control and Redundancy Framework

An example of such an approach is given in [43] and [44], in which linear quadratic Gaussian (LQG) control design is used to choose gains that minimize a linear combination of state and control inputs. This approach explicitly balances the trade-off between tracking errors (since the controller attempts to drive $\boldsymbol{s} - \boldsymbol{s}^*$ to zero) and robot motion. A similar control approach is proposed in [45] where joint limit avoidance is considered simultaneously with the positioning task.

It is also possible to formulate optimality criteria that explicitly express the observability of robot motion in the image. For example, the singular value decomposition of the interaction matrix reveals which degrees of freedom are most apparent and can thus be easily controlled, while the condition number of the interaction matrix gives a kind of global measure of the visibility of motion. This concept has been called resolvability in [46] and motion perceptibility in [47]. By selecting features and designing controllers that

maximize these measures, either along specific degrees of freedom or globally, the performance of the visual servo system can be improved.

The constraints considered to design the control scheme using the optimal control approach may be contradictory in some cases, leading the system to fail due to local minima in the objective function to be minimized. For example, it may happen that the motion produced to move away from a robot joint limit is exactly the opposite of the motion produced toward the desired pose, which results in a zero global motion. To avoid this potential problem, it is possible to use the gradient projection method, which is classical in robotics. Applying this method to visual servoing has been proposed in [1] and [19]. The approach consists of projecting the secondary constraints $\boldsymbol{e_s}$ onto the null space of the vision-based task $\boldsymbol{e}$ so that they have no effect on the regulation of $\boldsymbol{e}$ to 0:

$$\boldsymbol{e}_\mathrm{g} = \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}\boldsymbol{e} + \mathbf{P}_{\boldsymbol{e}}\,\boldsymbol{e_s}\ ,$$

where $\boldsymbol{e}_\mathrm{g}$ is the new global task considered and $\mathbf{P}_{\boldsymbol{e}} = (\mathbf{I}_6 - \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}}\widehat{\mathbf{L}_{\boldsymbol{e}}})$ is such that $\widehat{\mathbf{L}_{\boldsymbol{e}}}\mathbf{P}_{\boldsymbol{e}}\boldsymbol{e_s} = 0, \forall \boldsymbol{e_s}$. Avoiding the robot joint limits using this approach has been presented in [48]. However, when the vision-based task constrains all the camera degrees of freedom, the secondary constraints cannot be considered since, when $\widehat{\mathbf{L}_{\boldsymbol{e}}}$ is of full rank 6, we have $\mathbf{P}_{\boldsymbol{e}}\boldsymbol{e_s} = 0, \forall \boldsymbol{e_s}$. In this case, it is necessary to insert the constraints into a global objective function, such as navigation functions that are free of local minima [49, 50].

34.5.2 Switching Schemes

The partitioned methods described previously attempt to optimize performance by assigning individual controllers to specific degrees of freedom. Another way to use multiple controllers to optimize performance is to design switching schemes that select at each moment in time which controller to use based on criteria to be optimized.

A simple switching controller can be designed using an IBVS and a PBVS controller as follows [51]. Let the system begin by using the IBVS controller. Consider the Lyapunov function for the PBVS controller given by $\mathcal{L}_\mathrm{P} = \frac{1}{2}\|\boldsymbol{e}_\mathrm{P}(t)\|^2$, with $\boldsymbol{e}_\mathrm{P}(t) = ({}^{\mathrm{c}}\boldsymbol{t}_\mathrm{o} - {}^{\mathrm{c}^*}\boldsymbol{t}_\mathrm{o}, \theta\boldsymbol{u})$. If at any time the value of this Lyapunov function exceeds a threshold γ_P, the system switches to the PBVS controller. While using the PBVS controller, if at any time the value of the Lyapunov function $\mathcal{L}_\mathrm{I}$ for the IBVS controller exceeds a threshold, $\mathcal{L}_\mathrm{I} = \frac{1}{2}\|\boldsymbol{e}_\mathrm{I}(t)\|^2 > \gamma_\mathrm{I}$, the system switches to the IBVS controller. With this scheme, when the Lyapunov function for a particular controller exceeds a threshold, that controller is invoked, which in turn reduces the value of the corresponding Lyapunov function. If the switching thresholds are selected appropriately, the system is able to exploit the relative advantages of IBVS and PBVS, while avoiding their shortcomings.

An example of such a system is shown in Fig. 34.14 for the case of a rotation by 160° about the optical axis. Note that the system begins in IBVS mode and the features initially move on straight lines toward their goal positions in the image. However, as the camera retreats, the system switches to PBVS, which allows the camera to reach its desired position by combining a rotational motion around its optic axis and a forward translational motion, producing the circular trajectories observed in the image.

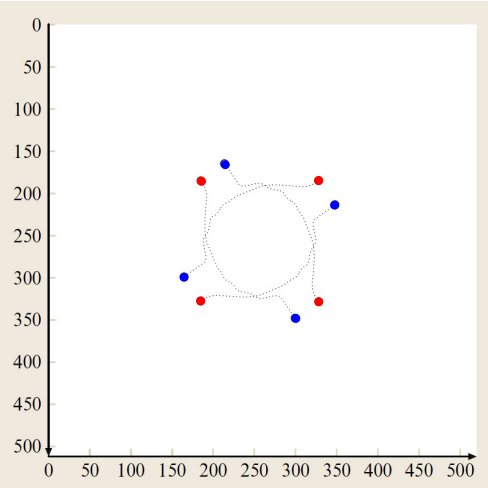

Figure 34.14: Image feature trajectories for a rotation of 160° about the optical axis using a switched control scheme (initial point positions in *blue*, and desired points position in *red*)

Other examples of temporal switching schemes can be found, such as the one developed in [52], to ensure the visibility of the target observed.

34.5.3 Feature Trajectory Planning

It is also possible to treat the optimization problem offline, during a planning stage, if we have sufficient knowledge of the system and world. In this case, several constraints can be taken into account simultaneously, such as obstacle avoidance [53], joint limit and occlusions avoidance, and ensuring the visibility of the target [54]. The feature trajectories $\boldsymbol{s}^*(t)$ that allow the camera to reach its desired pose while ensuring that the constraints are satisfied are determined using path planning techniques, such as the well-known potential field approach [54] or linear matrix inequality optimizations [55].

Coupling path planning with trajectory following also allows the robustness of the visual servo with respect to modeling errors to be significantly improved. Indeed, modeling errors may have large effects when the error $\boldsymbol{s} - \boldsymbol{s}^*$ is large, but have little effect when $\boldsymbol{s} - \boldsymbol{s}^*$ is small. Once the desired features trajectories $\boldsymbol{s}^*(t)$ such that $\boldsymbol{s}^*(0) = \boldsymbol{s}(0)$ have been designed during the planning stage, it is easy to adapt the control scheme to take into account the fact that $\boldsymbol{s}^*$ is varying, and to make the error $\boldsymbol{s} - \boldsymbol{s}^*$ remain small. More precisely, we now have

$$\dot{\boldsymbol{e}} = \dot{\boldsymbol{s}} - \dot{\boldsymbol{s}}^* = \mathbf{L}_{\boldsymbol{e}} \boldsymbol{v}_{\mathrm{c}} - \dot{\boldsymbol{s}}^* \ ,$$

from which we deduce, by selecting as usual $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$ as the desired behavior,

$$\boldsymbol{v}_{\mathrm{c}} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \boldsymbol{e} + \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \dot{\boldsymbol{s}}^* \ .$$

The new second term of this control law anticipates the variation of $\boldsymbol{s}^*$, removing the tracking error it would produce. We will see in the Sect. 34.8 that the form of the control law is similar when tracking of a moving target is considered.

34.6 Estimation of 3-D Parameters

All the control schemes described in the previous sections use 3-D parameters that are not directly available from the image measurements. As for IBVS, we recall that the range of the object with respect to the camera appears in the coefficients of the interaction matrix related to the translational degrees of freedom. Noticeable exceptions are the schemes based on a numerical estimation of $\mathbf{L}_{\boldsymbol{e}}$ or of $\mathbf{L}_{\boldsymbol{e}}^{+}$ (see Sect. 34.2.9). Another exception is the IBVS scheme that uses the constant matrix $\widehat{\mathbf{L}_{\boldsymbol{e}^*}^{+}}$ in the control scheme, in which only the depth for the desired pose is required, which is not so difficult to obtain in practice. As for PBVS and hybrid schemes that combine 2-D and 3-D data in $\boldsymbol{e}$, 3-D parameters appear both in the error $\boldsymbol{e}$ and in the interaction matrix. A correct estimation of the 3-D parameters involved is thus important for IBVS since they will have an effect on the camera motion during the task execution (they appear in the stability conditions (34.13) and (34.14)), while a correct estimation is crucial in PBVS and hybrid schemes since they will have also an effect on the accuracy of the pose reached after convergence.

If a calibrated stereo vision system is used, all 3-D parameters can be easily determined by triangulation, as mentioned in Sect. 34.2.5 and described in Chap. 032. Similarly, if a 3-D model of the object is known, all 3-D parameters can be computed from a pose-estimation algorithm. However, we recall that such an estimation can be quite unstable due to image noise (see Sect. 34.3). As already said, IBVS does not require full pose estimation, simply the range of the object with respect to the camera. When image points are involved in $\boldsymbol{s}$, the range is expressed as the scalar depth Z or distance R of the corresponding world points, which appears in the interaction matrix (34.12), (34.16) and (34.17). This can be considered as a parameter estimation problem, that is, estimating the 3-D parameters from knowledge of the analytical form of the interaction matrix, and measurements of camera motion and visual feature position and velocity [56, 57, 58].

It is also possible to estimate 3-D parameters by using the epipolar geometry that relates the images of the same scene observed from different viewpoints. Indeed, in visual servoing, two images are generally available: the current one and the desired one. Given a set of matches between the image measurements in the current image and in the desired one, the fundamental matrix, or the essential matrix if the camera is calibrated, can be recovered [6], and then used in visual servoing [59]. Indeed, from the essential matrix, the rotation and the translation up to a scalar factor between the two views can be estimated. However, near the convergence of the visual servo, that is, when the current and desired images are similar, the epipolar geometry becomes degenerate and it is not possible to estimate accurately the partial pose between the two views. For this reason, using homography is generally preferred.

Let $\boldsymbol{x}_i$ and $\boldsymbol{x}_i^*$ denote the homogeneous image coordinates for a point in the current and desired images. Then $\boldsymbol{x}_i$ is related to $\boldsymbol{x}_i^*$ by

$$\boldsymbol{x}_i = \mathbf{H}_i \boldsymbol{x}_i^*$$

in which $\mathbf{H}_i$ is a homography matrix.

If all feature points lie on a 3-D plane, then there is a single homography matrix $\mathbf{H}$ such that $\boldsymbol{x}_i = \mathbf{H}\boldsymbol{x}_i^*$ for all i. This homography can be estimated using the position of four matched points in the desired and the current images. If all the features points do not belong to the same 3-D plane, then three points can be used to define such a plane and five supplementary points are needed to estimate $\mathbf{H}$ [60].

Once $\mathbf{H}$ is available, it can be decomposed as

$$\mathbf{H} = \mathbf{R} + \frac{\boldsymbol{t}}{d^*}\boldsymbol{n}^{*\top} , \tag{34.29}$$

in which $\mathbf{R}$ is the rotation matrix relating the orientation of the current and desired camera frames, $\boldsymbol{n}^*$ is the normal to the chosen 3-D plane expressed in the desired frame, d^* is the distance to the 3-D plane from the desired frame, and $\boldsymbol{t}$ is the translation between current and desired frames. From $\mathbf{H}$, it is thus possible to recover $\mathbf{R}$, $\boldsymbol{t}/d^*$, and $\boldsymbol{n}$. In fact, two solutions for these quantities exist [62], but it is quite easy to select the correct one using some knowledge about the desired pose. It is also possible to estimate the depth of any target point up to a common scale factor [54]. The unknown depth of each point that appears in classical IBVS can thus be expressed as a function of a single, constant parameter whatever the number of points. Similarly, the pose parameters required by PBVS can be recovered up to a scalar factor as for the translation term. The PBVS schemes described previously can thus be revisited using this approach, with the new error defined as the translation up to a scalar factor and the angle/axis parameterization of the rotation. This approach has also been used for the hybrid visual servoing schemes described in Sect. 34.4.1. In that case, using such homography estimation, it has been possible to analyze the stability of hybrid visual servoing schemes in the presence of calibration errors [36]. Finally, it is also possible to directly use the homography in the control scheme, avoiding thus its decomposition as a partial pose [61].

34.7 Determining s^* and matching issues

All visual servo methods require knowledge of the desired feature values $\boldsymbol{s}^*$ which implicitly define constraints on the desired camera or robot pose with respect to the target. Three common approaches are employed. The first is when the task is directly specified as a desired value of some features to be reached. This is the case for instance when a target has to be centered in the image. This is also the case when the task is specified as a particular pose to reach and a PBVS is chosen. In that case however, the camera will reach its desired pose only if the camera is perfectly calibrated. Indeed, a coarse camera calibration will induce a biased estimation of the pose, which will make the final pose different from the desired one.

For IBVS and hybrid schemes, the second approach is to use knowledge of the object and camera projection models to compute $\boldsymbol{s}^*$ for the desired relative pose. Once again, the accuracy of the system directly depends on the camera calibration, since the camera intrinsic parameters are involved to compute $\boldsymbol{s}^*$.

Finally, the third approach is to simply record the feature values $\boldsymbol{s}^*$ when the camera or robot has the desired pose with respect to the target. This is usually done during an off-line teaching step. When it is possible in practice, this approach is very efficient since the positioning accuracy does not depend anymore on the camera calibration. This is still true for PBVS since, even if the pose estimated from the desired image is biased due to calibration errors, the same biased posed will be estimated once the robot will have converged so that the final image acquired by the camera will be the desired one.

So far we have not commented on the matching issues involved in visual servo methods. Two cases can be differentiated in function of the nature of the components of $\boldsymbol{s}$. When some components of $\boldsymbol{s}$ come from a pose estimation, it is necessary to match the measurements in the image (usually some image points) to the model of the object (usually some world points). Incorrect association will lead to an erroneous estimation of the pose. In all other cases, the calculation of the error vector $\boldsymbol{e}$ defined in (34.1) necessitates a matching between the measurements $\boldsymbol{m}(t)$ in the current image and $\boldsymbol{m}^*$ in the desired image. For instance, in all IBVS examples presented above, the camera observes four image points and we need to determine which of the four desired points to associate with each observed point so that the visual features $\boldsymbol{s}$ are correctly associated to their desired value $\boldsymbol{s}^*$. Incorrect association will lead to incorrect final camera pose and possibly a configuration which can not be achieved from any real camera pose. If we want to use the epipolar geometry or estimate a homography matrix, a similar matching process is necessary (see Sect. 34.6).

This matching process is a classical computer vision problem. Note that it may be particularly difficult for the very first image, especially when the robot displacement to achieve is large, which generally implies large disparities between the initial and desired images. Once the association has been correctly performed for the very first image, the matching is greatly simplified since it transforms to a visual tracking problem where the results obtained for the previous image can be used as initialization for the current one.

34.8 Target Tracking

We now consider the case of a moving target and a constant desired value $\boldsymbol{s}^*$ for the features, the generalization to varying desired features $\boldsymbol{s}^*(t)$ being immediate. The time variation of the error is now given by

$$\dot{\boldsymbol{e}} = \mathbf{L}_{\boldsymbol{e}} \boldsymbol{v}_{\mathrm{c}} + \frac{\partial \boldsymbol{e}}{\partial t} \,, \tag{34.30}$$

where the term $\frac{\partial \boldsymbol{e}}{\partial t}$ expresses the time variation of $\boldsymbol{e}$ due to the generally unknown target motion. If the control law is still designed to try to ensure an exponential decoupled decrease of $\boldsymbol{e}$ (that is, once again $\dot{\boldsymbol{e}} = -\lambda \boldsymbol{e}$), we now obtain using (34.30):

$$\boldsymbol{v}_{\mathrm{c}} = -\lambda \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \boldsymbol{e} - \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} \,, \tag{34.31}$$

where $\widehat{\frac{\partial \boldsymbol{e}}{\partial t}}$ is an estimation or an approximation of $\frac{\partial \boldsymbol{e}}{\partial t}$. This term must be introduced into the control law to compensate for the target motion.

Closing the loop, that is, inserting (34.31) into (34.30), we obtain

$$\dot{\boldsymbol{e}} = -\lambda \mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \boldsymbol{e} - \mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} + \frac{\partial \boldsymbol{e}}{\partial t} \,. \tag{34.32}$$

Even if $\mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} > 0$, the error will converge to zero only if the estimation of $\widehat{\frac{\partial \boldsymbol{e}}{\partial t}}$ is sufficiently accurate so that

$$\mathbf{L}_{\boldsymbol{e}} \widehat{\mathbf{L}_{\boldsymbol{e}}^{+}} \widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \frac{\partial \boldsymbol{e}}{\partial t} \,, \tag{34.33}$$

otherwise tracking errors will be observed. Indeed, by just solving the scalar differential equation $\dot{e} = -\lambda e + b$, which is a simplification of (34.32), we obtain $e(t) = e(0)\exp(-\lambda t) + b/\lambda$, which converges towards b/λ. On one hand, setting a high gain λ will reduce the tracking error, but on the other hand, setting the gain too high can make the system unstable. It is thus necessary to make b as small as possible.

Of course, if the system is known to be such that $\frac{\partial \boldsymbol{e}}{\partial t} = 0$ (that is, the camera observes a motionless object, as described in Sect. 34.1), no tracking error will appear with the most simple estimation given by $\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = 0$. Otherwise, a classical method in automatic

control to cancel tracking errors consists of compensating the target motion through an integral term in the control law. In this case, we have

$$\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \mu \sum_j \boldsymbol{e}(j) \ ,$$

where μ is the integral gain that has to be tuned. This scheme allows the tracking errors to be canceled only if the target has a constant velocity. Other methods, based on feedforward control, estimate the term $\widehat{\frac{\partial \boldsymbol{e}}{\partial t}}$ directly through the image measurements and the camera velocity, when it is available. Indeed, from (34.30), we obtain

$$\widehat{\frac{\partial \boldsymbol{e}}{\partial t}} = \widehat{\dot{\boldsymbol{e}}} - \widehat{\mathbf{L}_{\boldsymbol{e}}} \widehat{\boldsymbol{v}_{\mathrm{c}}} \ ,$$

where $\widehat{\dot{\boldsymbol{e}}}$ can, for instance, be obtained as $\widehat{\dot{\boldsymbol{e}}}(t) = [\boldsymbol{e}(t) - \boldsymbol{e}(t - \Delta t)]/\Delta t$, Δt being the duration of the control loop. A Kalman filter [63] or more-elaborate filtering methods [64] can then be used to improve the estimated values obtained. If some knowledge about the target velocity or the target trajectory is available, it can of course be used to smooth or predict the motion [65, 66, 67]. For instance, in [68], the periodic motion of the heart and breathing are compensated for an application of visual servoing in medical robotics. Finally, other methods have been developed to remove the perturbations induced by the target motion as fast as possible [43], using for instance predictive controllers [69].

34.9 Eye-in-Hand and Eye-to-Hand Systems Controlled in the Joint Space

In the previous sections, we have considered the six components of the camera velocity as the input of the robot controller. As soon as the robot is not able to realize this motion, for instance, because it has fewer than six degrees of freedom, the control scheme must be expressed in the joint space. In this section, we describe how this can be done, and in the process develop a formulation for eye-to-hand systems.

In the joint space, the system equations for both the eye-to-hand and eye-in-hand configurations have the same form:

$$\dot{\boldsymbol{s}} = \mathbf{J}_{\boldsymbol{s}} \, \dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{e}}{\partial t} \ . \tag{34.34}$$

Here, $\mathbf{J}_{\boldsymbol{s}} \in \mathbb{R}^{k \times n}$ is the feature Jacobian matrix, which can be linked to the interaction matrix, and n is the number of robot joints.

For an eye-in-hand system (Fig. 34.15a), $\frac{\partial \boldsymbol{e}}{\partial t}$ is the time variation of $\boldsymbol{s}$ due to a potential object motion, and $\mathbf{J}_{\boldsymbol{s}}$ is given by

$$\mathbf{J}_{\boldsymbol{s}} = \mathbf{L}_{\boldsymbol{s}} \, {}^{\mathrm{c}}\boldsymbol{X}_N \mathbf{J}(\boldsymbol{q}) \ , \tag{34.35}$$

where

- ${}^{\mathrm{c}}\boldsymbol{X}_N$ is the spatial motion transform matrix (as defined in Chap. 002 and recalled in (34.15)) from the vision sensor frame to the end-effector frame. It is usually a constant matrix (as long as the vision sensor is rigidly attached to the end-effector). Thanks to the robustness of closed-loop control schemes, a coarse approximation of this transform matrix is sufficient in visual servoing. If needed, an accurate estimation is possible through classical hand–eye calibration methods [70].
- $\mathbf{J}(\boldsymbol{q})$ is the robot Jacobian expressed in the end-effector frame (as defined in Chap. 002)

For an eye-to-hand system (Fig. 34.15b), $\frac{\partial \boldsymbol{e}}{\partial t}$ is now the time variation of $\boldsymbol{s}$ due to a potential vision sensor motion and $\mathbf{J}_{\boldsymbol{s}}$ can be expressed as:

$$\mathbf{J}_{\boldsymbol{s}} = -\mathbf{L}_{\boldsymbol{s}} \, {}^{\mathrm{c}}\boldsymbol{X}_N \, {}^{N}\mathbf{J}(\boldsymbol{q}) \ , \tag{34.36}$$

$$= -\mathbf{L}_{\boldsymbol{s}} \, {}^{\mathrm{c}}\boldsymbol{X}_0 \, {}^{0}\mathbf{J}(\boldsymbol{q}) \ . \tag{34.37}$$

In (34.36), the classical robot Jacobian ${}^{N}\mathbf{J}(\boldsymbol{q})$ expressed in the end-effector frame is used but the spatial motion transform matrix ${}^{\mathrm{c}}\boldsymbol{X}_N$ from the vision sensor frame to the end-effector frame changes all along the robot motion, and it has to be estimated at each iteration of the control scheme, usually using pose-estimation methods.

In (34.37), the robot Jacobian ${}^{0}\mathbf{J}(\boldsymbol{q})$ is expressed in the robot reference frame, and the spatial motion

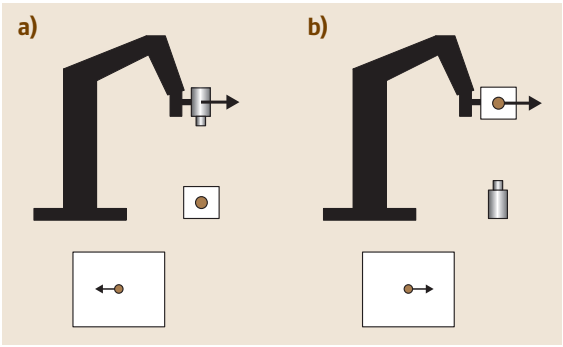

Figure 34.15: **(a)** Eye-in-hand system, **(b)** eye-to-hand system: system schematic (*top*) and opposite image motion produced by the same robot motion (*bottom*)

transform matrix ${}^{\mathrm{c}}\boldsymbol{X}_0$ from the vision sensor frame to that reference frame is constant as long as the camera does not move. In this case, which is convenient in practice, a coarse approximation of ${}^{\mathrm{c}}\boldsymbol{X}_0$ is usually sufficient.

Once the modeling step is finished, it is quite easy to follow the procedure that has been used above to design a control scheme expressed in the joint space, and to determine the sufficient condition to ensure the stability of the control scheme. We obtain, considering again $\boldsymbol{e} = \boldsymbol{s} - \boldsymbol{s}^*$, and an exponential decoupled decrease of $\boldsymbol{e}$:

$$\dot{\boldsymbol{q}} = -\lambda \widehat{\mathbf{J}_{\boldsymbol{e}}^{+}} \boldsymbol{e} - \widehat{\mathbf{J}_{\boldsymbol{e}}^{+}} \frac{\widehat{\partial \boldsymbol{e}}}{\partial t} \; . \tag{34.38}$$

If $k = n$, considering as in Sect. 34.1 the Lyapunov function $\mathcal{L} = \frac{1}{2}\|\boldsymbol{e}(t)\|^2$, a sufficient condition to ensure the global asymptotic stability is given by

$$\mathbf{J}_{\boldsymbol{e}} \widehat{\mathbf{J}_{\boldsymbol{e}}^{+}} > 0 \; . \tag{34.39}$$

If $k > n$, we obtain similarly to Sect. 34.1

$$\widehat{\mathbf{J}_{\boldsymbol{e}}^{+}} \mathbf{J}_{\boldsymbol{e}} > 0 \tag{34.40}$$

to ensure the local asymptotic stability of the system. Note that the actual extrinsic camera parameters appear in $\mathbf{J}_{\boldsymbol{e}}$ while the estimated ones are used in $\widehat{\mathbf{J}_{\boldsymbol{e}}^{+}}$. It is thus possible to analyze the robustness of the control scheme with respect to the camera extrinsic parameters. It is also possible to estimate directly the numerical value of $\mathbf{J}_{\boldsymbol{e}}$ or $\mathbf{J}_{\boldsymbol{e}}^{+}$ using the methods described in Sect. 34.2.9.

Finally, to remove tracking errors, we have to ensure that

$$\mathbf{J}_{\boldsymbol{e}} \widehat{\mathbf{J}_{\boldsymbol{e}}^{+}} \frac{\widehat{\partial \boldsymbol{e}}}{\partial t} = \frac{\partial \boldsymbol{e}}{\partial t} \; .$$

Let us note that, even if the robot has six degrees of freedom, it is generally not equivalent to first compute $\boldsymbol{v}_{\mathrm{c}}$ using (34.5) and then deduce $\dot{\boldsymbol{q}}$ using the robot inverse Jacobian, and to compute directly $\dot{\boldsymbol{q}}$ using (34.38). Indeed, it may occur that the robot Jacobian $\mathbf{J}(\boldsymbol{q})$ is singular while the feature Jacobian $\mathbf{J}_{\boldsymbol{s}}$ is not (that may occur when $k < n$). Furthermore, the properties of the pseudo-inverse ensure that using (34.5), $\|\boldsymbol{v}_{\mathrm{c}}\|$ is minimal while using (34.38), $\|\dot{\boldsymbol{q}}\|$ is minimal. As soon as $\mathbf{J}_{\boldsymbol{e}}^{+} \neq \mathbf{J}^{+}(\boldsymbol{q})^{N} \boldsymbol{X}_{\mathrm{c}} \mathbf{L}_{\boldsymbol{e}}^{+}$, the control schemes will be different and will induce different robot trajectories. The choice of the state space is thus important.

34.10 Under actuated robots

Many useful robots are under actuated, that is, they cannot move instantaneously in all directions because of the number or configuration of their actuators (see Chap. 052). For instance, a quadrotor flying robot is under actuated since it has only four actuators while the dimension of its configuration space is six. Many other useful robots are subject to non-holonomic constraints (see Chap. 049), leading to similar motion inability. A car for instance has only two degrees of freedom (for velocity and steering) while the dimension of its configuration space is three (position and orientation in the ground plane).

A consequence of under actuation is that time varying manoeuvres may be required in order to achieve particular goal states. For example, if a quadrotor has to move forward, it must first change its attitude, pitching down, so that a component of its thrust vector is able to accelerates the vehicle forward. For a visual servo system this can be problematic since the

initial attitude change will affect the value of the visual features even before the vehicle has moved. In fact the attitude change increases the error and this is ultimately destabilising.

A common and expedient solution [71] is to *derotate* the image, that is, to use information from a non-vision attitude sensor such as an IMU (see Chap. 029) to correct the feature coordinates as if they had been viewed by a virtual camera whose optical axis has a constant orientation in space, typically, straight down. As discussed in Sect. 34.2.8, the spherical projection model is well suited for such image transformation, thanks to its invariance properties with respect to rotational motion.

For a non-holonomic vehicle, there are several solutions. In the non-general case where the vehicle is able to follow a smooth path from its initial to goal configuration, a controller based on visually estimated relative pose (range, heading angle and lateral offset) can be used (see Chap. 049), following a PBVS strategy. Particular controllers based on the epipolar geometry or the trifocal tensor have also been designed [72, 73]. For the more general case, switching control laws can be used. If possible in practice, another common solution is to add a controlled DOF between the vehicle and the camera in order to bypass the non-holonomic constraint through redundancy. It is thus possible to control the full camera configuration, but not the robot one.

34.11 Applications

Applications of visual servoing in robotics are numerous. It can be used as soon as a vision sensor is available and a task is assigned to a dynamic system to control its motion. A non exhaustive list of examples are (see Figure 34.16):

- the control of a pan-tilt-zoom camera for target tracking;
- grasping using a robot arm;
- locomotion and dextrous manipulation with a humanoid robot;
- micro or nano manipulation of MEMS or biological cells;
- pipe inspection by an underwater autonomous vehicle;
- autonomous navigation of a mobile robot in indoor or outdoor environment;
- aircraft landing;
- autonomous satellite rendezvous;
- biopsy using ultrasound probes or heart motion compensation in medical robotics;
- virtual cinematography in animation.

34.12 Conclusions

In this chapter we have only considered velocity controllers, which are convenient for most classical robot arms. However, the dynamics of the robot must of course be taken into account for high-speed tasks. Features related to the image motion [74], image intensity [75], or coming from other vision sensors (RGB-D sensors, ultrasonic probes [76], etc.) necessitate reconsideration of the modeling issues to select adequate visual features. Finally, fusing visual features with data coming from other sensors (force sensor, proximity sensors, etc.) at the level of the control scheme will allow new research topics to be addressed. The end of fruitful research in the field of visual servo is thus nowhere yet in sight.

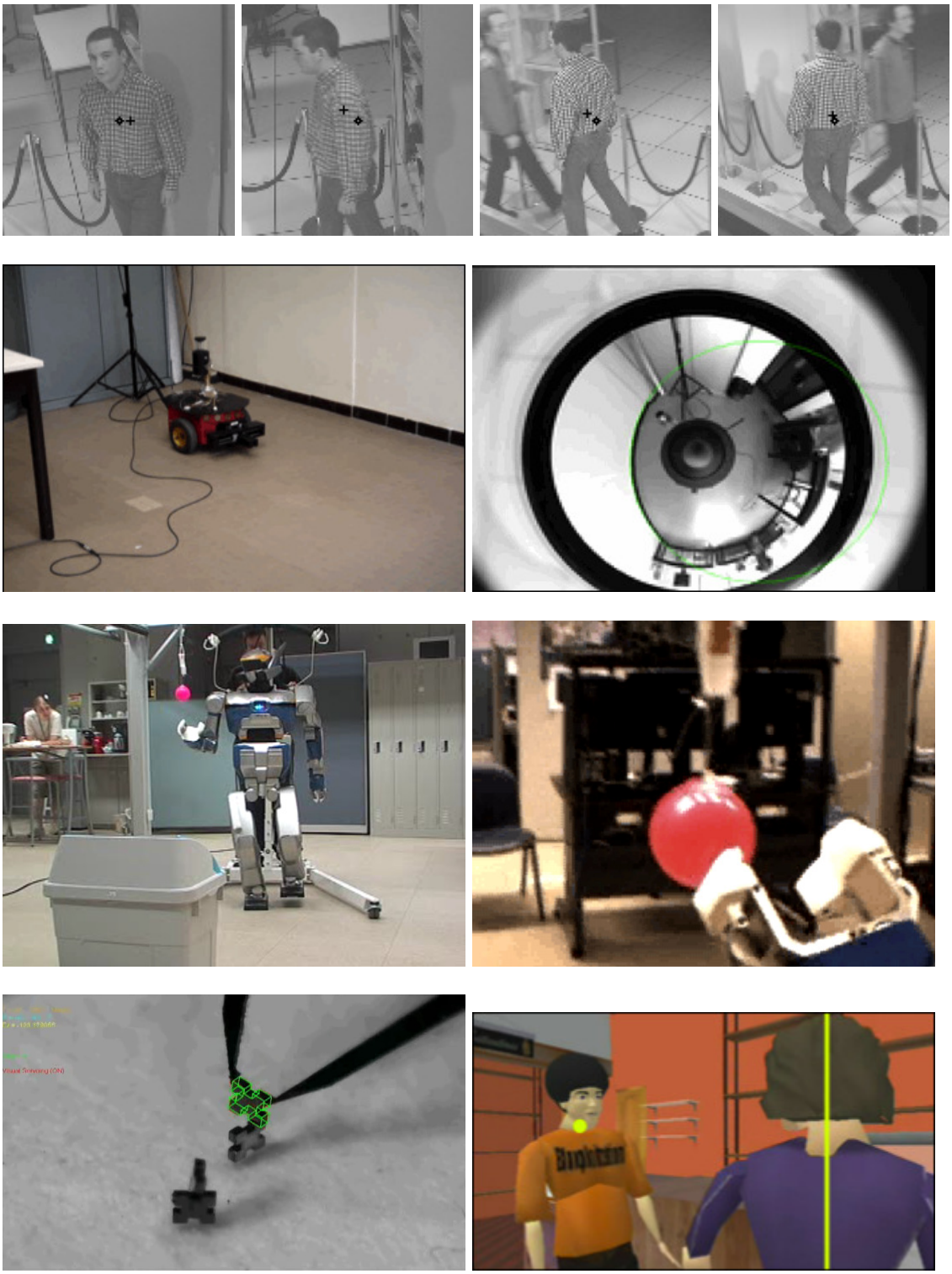

Figure 34.16: Few applications of visual servoing: gaze control for target tracking, navigation of a mobile robot to follow a wall using an omnidirectional vision sensor, grasping a ball with a humanoid robot, assembly of MEMS and film of a dialogue within the constraints of a script in animation.

Bibliography

[1] B. Espiau, F. Chaumette, P. Rives: A new approach to visual servoing in robotics, IEEE Trans. Robot. Autom. **8**, 313–326 (1992)

[2] S. Hutchinson, G. Hager, P. Corke: A tutorial on visual servo control, IEEE Trans. Robot. Autom. **12**, 651–670 (1996)

[3] L. Weiss, A. Sanderson, C. Neuman: Dynamic sensor-based control of robots with visual feedback, IEEE J. Robot. Autom. **3**, 404–417 (1987)

[4] J. Feddema, O. Mitchell: Vision-guided servoing with feature-based trajectory generation, IEEE Trans. Robot. Autom. **5**, 691–700 (1989)

[5] D. Forsyth, J. Ponce: *Computer Vision: A Modern Approach* (Prentice Hall, Upper Saddle River 2003)

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

[7] P. Corke: *Robotics, Vision & Control: Fundamental Algorithms in MATLAB* (Springer, 2011)

[8] H. Michel, P. Rives: Singularities in the determination of the situation of a robot effector from the erspective view of three points. Tech. Rep. 1850, INRIA Res. Rep. (1993)

[9] M. Fischler, R. Bolles: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, Commun. ACM **24**, 381–395 (1981)

[10] E. Malis: Improving vision-based control using efficient second-order minimization techniques, IEEE Int. Conf. Robot. Autom. (New Orleans 2004) pp. 1843–1848

[11] E. Marchand, F. Spindler, F. Chaumette: ViSP for visual servoing: a generic software platform with a wide class of robot control skills, IEEE Robot. Autom. Mag. **12**(4), 40–52 (2005)

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

[13] F. Chaumette: Potential problems of stability and convergence in image-based and position-based visual servoing. In: *The Confluence of Vision and Control*, LNCIS Series, Vol. 237, ed. by D. Kriegman, G. Hager, S. Morse (Springer, Heidelberg 1998) pp. 66–78

[14] E. Malis: Visual servoing invariant to changes in camera intrinsic parameters, IEEE Trans. Robot. Autom. **20**, 72–81 (2004)

[15] A. Isidori: *Nonlinear Control Systems*, 3^{rd} edn. (Springer Berlin, Heidelberg 1995)

[16] G. Hager, W. Chang, A. Morse: Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination, IEEE Contr. Syst. Mag. **15**, 30–39 (1995)

[17] M. Iwatsuki, N. Okiyama: A new formulation of visual servoing based on cylindrical coordinate system, IEEE Trans. Robot. Autom. **21**, 266–273 (2005)

[18] F. Chaumette, P. Rives, B. Espiau: Classification and realization of the different vision-based tasks. In: *Visual Servoing*, Robot. Autom. Syst.,

Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 199–228

[19] A. Castano, S. Hutchinson: Visual compliance: task directed visual servo control, IEEE Trans. Robot. Autom. **10**, 334–342 (1994)

[20] G. Hager: A modular system for robust positioning using feedback from stereo vision, IEEE Trans. Robot. Autom. **13**, 582–595 (1997)

[21] F. Chaumette: Image moments: a general and useful set of features for visual servoing, IEEE Trans. Robot. Autom. **20**, 713–723 (2004)

[22] O. Tahri, F. Chaumette: Point-based and region-based image moments for visual servoing of planar objects, IEEE Trans. Robot. **21**, 1116–1127 (2005)

[23] C. Geyer, K. Daniilidis: Catadioptric projective geometry, Int. J. Comput. Vision. **45**(3), 223–243 (2001)

[24] T. Hamel, R. Mahony: Visual servoing of an under-actuated dynamic rigid-body system: An image-based approach, IEEE Trans Robot.. **18**(2), 187–198 (2002)

[25] I. Suh: Visual servoing of robot manipulators by fuzzy membership function based neural networks. In: *Visual Servoing*, Robotics and Automated Systems, Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 285–315

[26] G. Wells, C. Venaille, C. Torras: Vision-based robot positioning using neural networks, Image Vision Comput. **14**, 75–732 (1996)

[27] J.T. Lapresté, F. Jurie, F. Chaumette: An efficient method to compute the inverse jacobian matrix in visual servoing, IEEE Int. Conf. Robot. Autom. (New Orleans 2004) pp. 727–732

[28] K. Hosada, M. Asada: Versatile visual servoing without knowledge of true jacobian, IEEE/RSJ Int. Conf. Intell. Robots Syst. (Munchen 1994) pp. 186–193

[29] M. Jägersand, O. Fuentes, R. Nelson: Experimental evaluation of uncalibrated visual servoing for precision manipulation, IEEE Int. Conf. Robot. Autom. (Albuquerque 1997) pp. 2874–2880

[30] J. Piepmeier, G.M. Murray, H. Lipkin: Uncalibrated dynamic visual servoing, IEEE Trans. Robot. Autom. **20**, 143–147 (2004)

[31] K. Deguchi: Direct interpretation of dynamic images and camera motion for visual servoing without image feature correspondence, J. Robot. Mechatron. **9**(2), 104–110 (1997)

[32] W. Wilson, C. Hulls, G. Bell: Relative end-effector control using cartesian position based visual servoing, IEEE Trans. Robot. Autom. **12**, 684–696 (1996)

[33] B. Thuilot, P. Martinet, L. Cordesses, J. Gallice: Position based visual servoing: Keeping the object in the field of vision, IEEE Int. Conf. Robot. Autom. (Washington 2002) pp. 1624–1629

[34] D. Dementhon, L. Davis: Model-based object pose in 25 lines of code, Int. J. Comput. Vision **15**, 123–141 (1995)

[35] D. Lowe: Three-dimensional object recognition from single two-dimensional images, Artif. Intell. **31**(3), 355–395 (1987)

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

[37] E. Malis, F. Chaumette: Theoretical improvements in the stability analysis of a new class of model-free visual servoing methods, IEEE Trans. Robot. Autom. **18**, 176–186 (2002)

[38] J. Chen, D. Dawson, W. Dixon, A. Behal: Adaptive homography-based visual servo tracking for fixed camera-in-hand configurations, IEEE Trans. Contr. Syst. Technol. **13**, 814–825 (2005)

[39] G. Morel, T. Leibezeit, J. Szewczyk, S. Boudet, J. Pot: Explicit incorporation of 2-D constraints

in vision-based control of robot manipulators, Int. Symp. Exp. Robot. **250**, 99–108 (2000), LNCIS Series

[40] F. Chaumette, E. Malis: 2 1/2 D visual servoing: a possible solution to improve image-based and position-based visual servoings, IEEE Int. Conf. Robot. Autom. (San Fransisco 2000) pp. 630–635

[41] E. Cervera, A.D. Pobil, F. Berry, P. Martinet: Improving image-based visual servoing with three-dimensional features, Int. J. Robot. Res. **22**, 821–840 (2004)

[42] F. Schramm, G. Morel, A. Micaelli, A. Lottin: Extended 2-D visual servoing, IEEE Int. Conf. Robot. Autom. (New Orleans 2004) pp. 267–273

[43] N. Papanikolopoulos, P. Khosla, T. Kanade: Visual tracking of a moving target by a camera mounted on a robot: A combination of vision and control, IEEE Trans. Robot. Autom. **9**, 14–35 (1993)

[44] K. Hashimoto, H. Kimura: LQ optimal and nonlinear approaches to visual servoing. In: *Visual Servoing*, Robot. Autom. Syst., Vol. 7, ed. by K. Hashimoto (World Scientific, Singapore 1993) pp. 165–198

[45] B. Nelson, P. Khosla: Strategies for increasing the tracking region of an eye-in-hand system by singularity and joint limit avoidance, Int. J. Robot. Res. **14**, 225–269 (1995)

[46] B. Nelson, P. Khosla: Force and vision resolvability for assimilating disparate sensory feedback, IEEE Trans. Robot. Autom. **12**, 714–731 (1996)

[47] R. Sharma, S. Hutchinson: Motion perceptibility and its application to active vision-based servo control, IEEE Trans. Robot. Autom. **13**, 607–617 (1997)

[48] E. Marchand, F. Chaumette, A. Rizzo: Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing, IEEE/RSJ Int. Conf. Intell. Robots Syst. (Osaka 1996) pp. 1083–1090

[49] E. Marchand, G. Hager: Dynamic sensor planning in visual servoing, IEEE Int. Conf. Robot. Autom. (Leuven 1998) pp. 1988–1993

[50] N. Cowan, J. Weingarten, D. Koditschek: Visual servoing via navigation functions, IEEE Trans. Robot. Autom. **18**, 521–533 (2002)

[51] N. Gans, S. Hutchinson: An asymptotically stable switched system visual controller for eye in hand robots, IEEE/RSJ Int. Conf. Intell. Robots Syst. (Las Vegas 2003) pp. 735–742

[52] G. Chesi, K. Hashimoto, D. Prattichizio, A. Vicino: Keeping features in the field of view in eye-in-hand visual servoing: a switching approach, IEEE Trans. Robot. Autom. **20**, 908–913 (2004)

[53] K. Hosoda, K. Sakamato, M. Asada: Trajectory generation for obstacle avoidance of uncalibrated stereo visual servoing without 3-D reconstruction, IEEE/RSJ Int. Conf. Intell. Robots Syst., Vol. 3 (Pittsburgh 1995) pp. 29–34

[54] Y. Mezouar, F. Chaumette: Path planning for robust image-based control, IEEE Trans. Robot. Autom. **18**, 534–549 (2002)

[55] G. Chesi: Visual servoing path-planning via homogeneous forms and LMI optimizations, IEEE Trans. Robot. **25**(2), 281–291 (2009)

[56] L. Matthies, T. Kanade, R. Szeliski: Kalman filter-based algorithms for estimating depth from image sequences, Int. J. Comput. Vision **3**(3), 209–238 (1989)

[57] C. E. Smith, N. Papanikolopoulos: Computation of shape through controlled active exploration, IEEE Int. Conf. Robot. Autom., (San Diego 1994) pp. 2516–2521

[58] A. De Luca, G. Oriolo, P. Robuffo Giordano: Feature depth observation for image-based visual servoing: Theory and experiments, Int. J. Rob. Res. **27**(10), 1093–1116 (2008)

[59] R. Basri, E. Rivlin, I. Shimshoni: Visual homing: Surfing on the epipoles, Int. J. Comput. Vision **33**, 117–137 (1999)

[60] E. Malis, F. Chaumette, S. Boudet: 2 1/2 D visual servoing with respect to unknown objects through a new estimation scheme of camera displacement, Int. J. Comput. Vision **37**, 79–97 (2000)

[61] G. Silveira, E. Malis: Direct visual servoing: Vision-based estimation and control using only nonmetric information, IEEE Trans. Robot. **28**(4), 974–980 (2012)

[62] O. Faugeras: *Three-Dimensional Computer Vision: a Geometric Viewpoint* (MIT Press, Cambridge 1993)

[63] P. Corke, M. Goods: Controller design for high performance visual servoing, 12th World Congress IFAC'93 (Sydney 1993) pp. 395–398

[64] F. Bensalah, F. Chaumette: Compensation of abrupt motion changes in target tracking by visual servoing, IEEE/RSJ Int. Conf. Intell. Robots Syst. (Pittsburgh 1995) pp. 181–187

[65] P. Allen, B. Yoshimi, A. Timcenko, P. Michelman: Automated tracking and grasping of a moving object with a robotic hand-eye system, IEEE Trans. Robot. Autom. **9**, 152–165 (1993)

[66] K. Hashimoto, H. Kimura: Visual servoing with non linear observer, IEEE Int. Conf. Robot. Autom. (Nagoya 1995) pp. 484–489

[67] A. Rizzi, D. Koditschek: An active visual estimator for dexterous manipulation, IEEE Trans. Robot. Autom. **12**, 697–713 (1996)

[68] R. Ginhoux, J. Gangloff, M. de Mathelin, L. Soler, M.A. Sanchez, J. Marescaux: Active filtering of physiological motion in robotized surgery using predictive control, IEEE Trans. Robot. **21**, 67–79 (2005)

[69] J. Gangloff, M. de Mathelin: Visual servoing of a 6 dof manipulator for unknown 3-D profile following, IEEE Trans. Robot. Autom. **18**, 511–520 (2002)

[70] R. Tsai, R. Lenz: A new technique for fully autonomous efficient 3-D robotics hand-eye calibration, IEEE Trans. Robot. Autom. **5**, 345–358 (1989)

[71] N. Guenard, T. Hamel, R. Mahony: A practical visual servo control for an unmanned aerial vehicle, IEEE Trans. Robot. **24**(2), 331–340 (2008)

[72] G.L. Mariottini, G. Oriolo, D. Prattichizo Image-based visual servoing for nonholonomic mobile robots using epipolar geometry, IEEE Trans. Robot. **23**(1), 87–100 (2007)

[73] G. Lopez-Nicolas, J.J. Guerrero, C. Sagues: Visual control through the trifocal tensor for nonholonomic robots Robot. Auton. Syst. **58**(2) 216–226 (2010)

[74] A. Crétual, F. Chaumette: Visual servoing based on image motion, Int. J. Robot. Res. **20**(11), 857–877 (2001)

[75] C. Collewet, E.Marchand: Photometric visual servoing, IEEE Trans. Robot. **27**(4), 828–834 (2011)

[76] R. Mebarki, A. Krupa, F. Chaumette: 2D ultrasound probe complete guidance by visual servoing using image moments, IEEE Trans. Robot. **26**(2), 296–306 (2010)