Visual Servoing

Luigi Villani

Dipartimento di Informatica e Sistemistica
Università di Napoli Federico II
via Claudio 21, 80125 Napoli (NA), Italy
luigi.villani@unina.it
http://wpage.unina.it/lvillani

Abstract

These notes are extracted from the book *Robotics – Modelling, Plannig and Control* (B. Siciliano, L. Sciavicco, L. Villani, G. Oriolo, Springer, 2009), where all the background material can be found. More advanced material and a wider list of further readings can be found in the chapter "Visual Servoing and Visual Tracking" (F. Chaumette, S. Hutchinson, in *Springer Handbook of Robotics*, B. Siciliano, O. Khatib (Eds.), Springer, 2008).

Contents

1 Introduction **1**
1.1 Configuration of the Visual System 2
1.2 Vision Sensors 2

2 Pose Estimation **4**
2.1 Interaction Matrix 5
2.1.1 Interaction matrix of a point 6
2.1.2 Interaction matrix of a set of points 6
2.2 Algorithmic Solution 6

3 Stereo Vision **8**
3.1 3D Reconstruction from Planar Homography 8

4 The Visual Servoing Problem **9**

5 Position-based Visual Servoing **10**
5.1 PD Control with Gravity Compensation . . 11
5.2 Resolved-velocity Control 11

6 Image-based Visual Servoing **12**
6.1 PD Control with Gravity Compensation . . 12
6.2 Resolved-velocity Control 13

7 Hybrid Visual Servoing **14**

8 Bibliography **15**

1 Introduction

Vision plays a key role in a robotic system, as it can be used to obtain geometrical and qualitative information on the environment where the robot operates, without physical interaction. Such information may be employed by the control system at different levels, for the sole task planning and also for feedback control.

As an example, consider the case of a robot manipulator, equipped with a camera, which has to grasp an object using a gripper. Through vision the robot may acquire information capable of identifying the relative pose of the object with respect to the gripper. This information allows the control system to plan a trajectory leading the manipulator in an appropriate grasping configuration, computed on the basis of the pose and of the shape of the object, from which the closure of the gripper can be commanded.

The planned trajectory can be executed using a simple motion controller. In this approach, termed *look-and-move*, visual measurements are used in open loop, making the system very sensitive to uncertainties due, for instance, to poor positioning accuracy of the manipulator or to the fact that the object may have moved while the gripper reaches the grasp position.

On the other hand, in *vision-based control* or *visual servoing*, the visual measurements are fed back to the control to compute an appropriate error vector defined between the current pose of the object and the pose of the manipulator's end-effector.

A key characteristic of visual servoing, compared to motion and force control, is the fact that the controlled variables are not directly measured by the sensor, but are obtained from the measured quantities through complex elaborations, based on algorithms of *image processing* and *computational vision*.

A standard monochrome camera simply provides a two-dimensional matrix of values of light intensity. From this matrix, the so-called *image feature parameters* are to be extracted in real time. The geometric relationships between one or more two-dimensional views of a scene and the corresponding 3D space are the basis of techniques of *pose estimation* of objects in the manipulator workspace or of the end-effector with respect to the surrounding objects. In

this regard, of fundamental importance is the operation of *camera calibration*, which is necessary for calculating the *intrinsic parameters*, relating the quantities measured in the image plane to those referred to the camera frame, and the *extrinsic parameters*, relating the latter to quantities defined in a frame attached to the manipulator.

The vision-based control schemes can be divided into two categories, namely, those that realize *visual servoing in operational space*, also termed *position-based visual servoing*, and those that realize *visual servoing in the image space*, also known as *image-based visual servoing*. The main difference lies in the fact that the schemes of the first category use visual measurements to reconstruct the relative pose of the object with respect to the robot, or vice versa, while the schemes of the second category are based on the comparison of the feature parameters of the image of the object between the current and the desired pose. There are also schemes combining characteristics common to both categories, that can be classified as *hybrid visual servoing*.

Another aspect to be considered for vision-based control is the type of camera (colour or monochrome, resolution, fixed or variable focal length, CCD or CMOS technology). In this chapter, only the case of monochrome cameras with fixed focal length will be considered.

Equally important is the choice of the number of cameras composing the visual system and their location; this issue is briefly discussed in the following.

1.1 Configuration of the Visual System

A visual system may consist of only one camera, or two or more cameras. If more cameras are used to observe the same object of a scene, it is possible to retrieve information about its depth by evaluating its distance with respect to the visual system. This situation is referred to as *3D vision* or *stereo vision*, where the term *stereo* derives from the Greek and means solid. The human capability of perceiving objects in three dimensions relies on the fact that the brain receives the same images from two eyes, observing the same scene from slightly different angles.

It is clear that 3D vision can be achieved even with one camera, provided that two images of the same object, taken from two different poses, are available. If only a single image is available, the depth can be estimated on the basis of certain geometrical characteristics of the object known in advance. This means that, in many applications, monocamera systems are often preferred to multi-camera systems, because they are cheaper and easier to calibrate, although characterized by lower accuracy.

Another feature that distinguishes visual systems for robot manipulators is the placement of cameras. For monocamera systems there are two options: the *fixed configuration*, often referred to as *eye-to-hand*, where the camera is mounted in a fixed location, and the *mobile configuration*, or *eye-in-hand*, with the camera attached to the robot. For multi-camera systems, in addition to the mentioned solutions, it is also possible to consider the *hybrid configuration*, consisting of one or more cameras in eye-to-hand configuration, and one or more cameras in eye-in-hand configuration.

In the eye-to-hand configuration, the visual system observes the objects to be manipulated by a fixed pose with respect to the base frame of the manipulator. The advantage is that the camera field of view does not change during the execution of the task, implying that the accuracy of such measurements is, in principle, constant. However, in certain applications, such as assembly, it is difficult to prevent that the manipulator, moving in the camera field of view, occludes, in part or in whole, the view of the objects.

In the eye-in-hand configuration, the camera is placed on the manipulator and can be mounted both before and after the wrist. In the first case, the camera can observe the end-effector by a favourable pose and without occlusions caused by the manipulator arm; in the latter case, the camera is attached to the end-effector and typically observes only the object. In both situations, the camera field of view changes significantly during the motion and this produces a high variability in the accuracy of measurements. However, when the end-effector is close to the object, the accuracy becomes almost constant and is usually higher than that achievable with eye-to-hand cameras, with the advantage that occlusions are virtually absent.

Finally, hybrid configuration combines the benefits of the other two configurations, namely, ensures a good accuracy throughout the workspace, while avoiding the problems of occlusions.

A separate category is represented by robotic heads, which are typically equipped with a stereo vision system consisting of two cameras mounted on motorized mechanisms that allow for yaw motion, or *pan*, and pitch motion, or *tilt*, hence the name of *pan-tilt* cameras.

In this chapter, only schemes based on a single eye-in-hand camera will be considered. The extension of the algorithms to the case of a eye-to-hand camera, or to the case of multiple cameras, requires only minor modifications.

1.2 Vision Sensors

The task of a camera as a *vision sensor* is to measure the intensity of the light reflected by an object. To this end, a photosensitive element, termed *pixel* (or *photosite*), is employed, which is capable of transforming light energy into electric energy. Different types of sensors are available depending on the physical principle exploited to realize the energy transformation. The most widely used devices are CCD and CMOS sensors based on the photoelectric effect of semiconductors.

The camera is a complex system comprising several devices other than the photosensitive sensor, i.e., a *shutter*, a *lens* and *analog preprocessing electronics*. The lens is responsible for focusing the light reflected by the object on the plane where the photosensitive sensor lies, called the *image plane*.

With reference to Fig. 1, consider a frame O_c–$x_c y_c z_c$ attached to the camera, whose location with respect to the

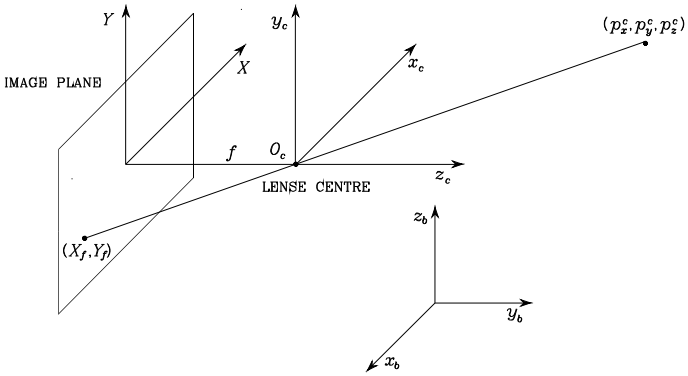

Figure 1: Perspective transformation

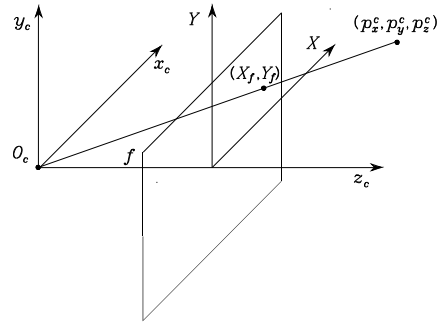

Figure 2: Frontal perspective transformation

base frame is identified by the homogeneous transformation matrix $\boldsymbol{T}_c^b$. Take a point of the object of coordinates $\boldsymbol{p}^c = [\, p_x^c \quad p_y^c \quad p_z^c \,]^T$; typically, the centroid of the object is chosen. Then, the coordinate transformation from the base frame to the camera frame is described as

$$\widetilde{\boldsymbol{p}}^c = \boldsymbol{T}_b^c \widetilde{\boldsymbol{p}}, \tag{1}$$

where $\boldsymbol{p}$ denotes the object position with respect to the base frame and homogeneous representations of vectors have been used.

A reference frame can be introduced on the image plane, whose axes X and Y are parallel to the axes x_c and y_c of the camera frame, and the origin is at the intersection of the optical axis with the image plane, termed principal point. Due to the refraction phenomenon, the point in the camera frame is transformed into a point in the image plane via the *perspective transformation*, i.e.,

$$\begin{aligned} X_f &= -\frac{f p_x^c}{p_z^c} \\ Y_f &= -\frac{f p_y^c}{p_z^c} \end{aligned}$$

where (X_f, Y_f) are the new coordinates in the frame defined on the image plane, and f is the *focal length* of the lens. Notice that these coordinates are expressed in metric units and the above transformation is singular at $p_z^c = 0$.

The presence of the minus sign in the equations of the perspective transformation is consistent with the fact that the image of an object appears upside down on the image plane of the camera. Such an effect can be avoided, for computational ease, by considering a virtual image plane positioned before the lens, in correspondence of the plane $z_c = f$ of the camera frame. In this way, the model represented in Fig. 2 is obtained, which is characterized by the *frontal* perspective transformation

$$X_f = \frac{f p_x^c}{p_z^c} \tag{2}$$

$$Y_f = \frac{f p_y^c}{p_z^c} \tag{3}$$

where, with abuse of notation, the name of the variables on the virtual plane has not been changed.

These relationships hold only in theory, since the real lenses are always affected by imperfections, which cause image quality degradation. Two types of distortions can be recognized, namely, *aberrations* and *geometric distortion*. The former can be reduced by restricting the light rays to a small central region of the lens; the effects of the latter can be compensated on the basis of a suitable model whose parameters are to be identified.

A visual information is typically elaborated by a digital processor, and thus the measurement principle is to transform the light intensity $I(X, Y)$ of each point in the image plane into a number. It is clear that a *spatial sampling* is needed since an infinite number of points in the image plane exist, as well as a *temporal sampling* since the image can change during time. The CCD or CMOS sensors play the role of spatial samplers, while the shutter in front of the lens plays the role of the temporal sampler.

The spatial sampling unit is the pixel, and thus the coordinates (X, Y) of a point in the image plane are to be expressed in pixels, i.e., (X_I, Y_I). Due to the photosite finite dimensions, the pixel coordinates of the point are related to the coordinates in metric units through two scale factors α_x and α_y, namely,

$$X_I = \frac{\alpha_x f p_x^c}{p_z^c} + X_0 \tag{4}$$

$$Y_I = \frac{\alpha_y f p_y^c}{p_z^c} + Y_0, \tag{5}$$

where X_0 and Y_0 are the offsets which take into account the position of the origin of the pixel coordinate system with respect to the optical axis. This nonlinear transformation can be written in a linear form by resorting to the homogeneous representation of the point (x_I, y_I, z_I) via the relationships

$$\begin{aligned} X_I &= \frac{x_I}{\lambda} \\ Y_I &= \frac{y_I}{\lambda} \end{aligned}$$

where $\lambda > 0$. As a consequence, (4), (5) can be rewritten

as

$$\begin{bmatrix} x_I \\ y_I \\ \lambda \end{bmatrix} = \lambda \begin{bmatrix} X_I \\ Y_I \\ 1 \end{bmatrix} = \mathbf{\Omega\Pi} \begin{bmatrix} p_x^c \\ p_y^c \\ p_z^c \\ 1 \end{bmatrix} \tag{6}$$

where

$$\mathbf{\Omega} = \begin{bmatrix} f\alpha_x & 0 & X_0 \\ 0 & f\alpha_y & Y_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

$$\mathbf{\Pi} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}. \tag{8}$$

At this point, the overall transformation from the Cartesian space of the observed object to the *image space* of its image in pixels is characterized by composing the transformations in (1), (6) as

$$\Xi = \mathbf{\Omega\Pi}\boldsymbol{T}_b^c \tag{9}$$

which represents the so-called *camera calibration* matrix. It is worth pointing out that such a matrix contains *intrinsic parameters* $(\alpha_x, \alpha_y, X_0, Y_0, f)$ in $\mathbf{\Omega}$ depending on the sensor and lens characteristics as well as *extrinsic parameters* in $\boldsymbol{T}_c^b$ depending on the relative position and orientation of the camera with respect to the base frame. Several calibration techniques exist to identify these parameters in order to compute the transformation between the Cartesian space and the image space as accurately as possible.

If the intrinsic parameters of a camera are known, from a computationally viewpoint, it is convenient to refer to the *normalized coordinates* (X, Y), defined by the normalized perspective transformation

$$\lambda \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \mathbf{\Pi} \begin{bmatrix} p_x^c \\ p_y^c \\ p_z^c \\ 1 \end{bmatrix}. \tag{10}$$

These coordinates are defined in metrical units and coincide with the coordinates (2), (3) in the case when $f = 1$. Comparing (6) with (10) yields the invertible transformation

$$\begin{bmatrix} X_I \\ Y_I \\ 1 \end{bmatrix} = \mathbf{\Omega} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{11}$$

relating the normalized coordinates to those expressed in pixels through the matrix of intrinsic parameters.

The last step of the measurement process is to digitize the analog video signal. The special analog-to-digital converters adopted for video signal acquisition are called *frame grabbers*. By connecting the output of the camera to the frame grabber, the video waveform is sampled and quantized and the values stored in a two-dimensional memory array representing the spatial sample of the image, known as *framestore*; this array is then updated at field or frame rate.

In the case of CMOS cameras (currently available only for monochrome images), thanks to CMOS technology which allows the integration of the analog-to-digital converter in each pixel, the output of the camera is directly a two-dimensional array, whose elements can be accessed randomly. Such advantage, with respect to CCD cameras, leads to the possibility of higher frame rates if only parts of the entire frame are accessed.

The sequence of steps from image formation to image acquisition described above can be classified as a process of *low-level vision*; this includes the extraction of elementary image features, e.g., centroid and intensity discontinuities. On the other hand, a robotic system can be considered really autonomous only if procedures for emulating cognition are available, e.g., recognizing an observed object among a set of CAD models stored into a data base. In this case, the artificial vision process can be referred to as *high-level vision*.

2 Pose Estimation

Visual servoing is based on the mapping between the feature parameters of an object measured in the image plane of the camera and the operational space variables defining the relative pose of the object with respect to the camera or, equivalently, of the camera with respect to the object.

The set of feature parameters of an image defines a $(k \times 1)$ vector $\boldsymbol{s}$, termed *feature vector*. In the following, to simplify notation, normalized coordinates (X, Y) defined in (10) will be used in place of pixel coordinates (X_I, Y_I) to define the feature vector. Since only pixel coordinates can be directly measured, the normalized coordinates should be computed from pixel coordinates using the inverse of mapping (11), provided that the intrinsic parameters of the camera are known.

The feature vector $\boldsymbol{s}$ of a point is defined as

$$\boldsymbol{s} = \begin{bmatrix} X \\ Y \end{bmatrix}, \tag{12}$$

while

$$\tilde{\boldsymbol{s}} = \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

denotes its representation in homogeneous coordinates.

Consider a camera, for example an eye-in-hand camera, and a reference frame O_c–$x_c y_c z_c$ attached to the camera; consider also a reference frame O_o–$x_o y_o z_o$ attached to the object, supposed to be rigid, and let $\boldsymbol{T}_o^c$ be the homogeneous transformation matrix corresponding to the relative pose of the object with respect to the camera, defined as

$$\boldsymbol{T}_o^c = \begin{bmatrix} \boldsymbol{R}_o^c & \boldsymbol{o}_{c,o}^c \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{13}$$

with $\boldsymbol{o}_{c,o}^c = \boldsymbol{o}_o^c - \boldsymbol{o}_c^c$, where $\boldsymbol{o}_c^c$ is the position vector of the origin of the camera frame with respect to the base frame, expressed in camera frame, $\boldsymbol{o}_o^c$ is the position vector of the

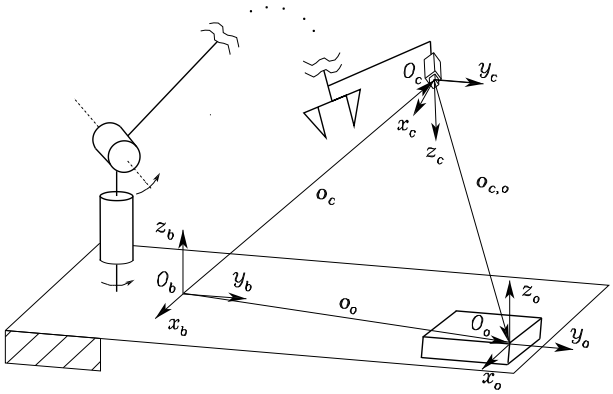

Figure 3: Reference frames for an eye-in-hand camera

origin of the object frame with respect to the base frame, expressed in the camera frame, and $\boldsymbol{R}_o^c$ is the rotation matrix of the object frame with respect to the camera frame (Fig. 3).

The problem to solve is that of computing the elements of matrix $\boldsymbol{T}_o^c$ from the measurements of object feature parameters in the camera image plane. To this end, consider n points of the object and let $\boldsymbol{r}_{o,i}^o = \boldsymbol{p}_i^o - \boldsymbol{o}_o^o$, $i = 1, \ldots, n$, denote the corresponding position vectors with respect to the object frame. These quantities are assumed to be known, for example, from a CAD model of the object. The projections of these points on the image plane have coordinates

$$\boldsymbol{s}_i = \begin{bmatrix} X_i \\ Y_i \end{bmatrix},$$

and define the feature vector

$$\boldsymbol{s} = \begin{bmatrix} \boldsymbol{s}_1 \\ \vdots \\ \boldsymbol{s}_n \end{bmatrix}. \tag{14}$$

The homogeneous coordinates of the points of the object with respect to the camera frame can be expressed as

$$\widetilde{\boldsymbol{r}}_{o,i}^c = \boldsymbol{T}_o^c \widetilde{\boldsymbol{r}}_{o,i}^o.$$

Therefore, using (10), the homogeneous coordinates of the projections of these points on the image plane are given by

$$\lambda_i \widetilde{\boldsymbol{s}}_i = \boldsymbol{\Pi} \boldsymbol{T}_o^c \widetilde{\boldsymbol{r}}_{o,i}^o, \tag{15}$$

with $\lambda_i > 0$.

Assume that n *correspondences* are available, namely n equations of the form (15) for n points of the object, whose coordinates are known both in the object frame and in the image plane. These correspondences define a system of equations to be solved for the unknown elements of matrix $\boldsymbol{T}_o^c$. Computing the solution is a difficult task because, depending on the type and on the number of correspondences, multiple solutions may exist. This problem, in photogrammetry, is known as PnP (Perspective-n-Point) problem. In particular, it can be shown that:

- P3P problem has four solutions, in the case of three non-collinear points.
- P4P and P5P problems each have at least two solutions, in the case of non-coplanar points, while the solution is unique in the case of at least four coplanar points and no triplets of collinear points.
- PnP problem, with $n \geq 6$ non-coplanar points, has only one solution.

The analytic solution to PnP problem is rather laborious. The derivation becomes simpler in some particular cases as, for example, in the case of coplanar points, but will not be considered here.

Notice that, to compute $\boldsymbol{T}_o^c$ by solving the PnP problem, it is necessary to know the object geometry but also the camera *intrinsic parameters*. The latter are required for computing normalized coordinates $\boldsymbol{s}_i$ from pixel coordinates. Moreover, if it is required to compute the object pose with respect to the base frame (as usually happens in the case of eye-to-hand cameras) or to the end-effector frame (as usually happens in the case of eye-in-hand cameras), then it is also necessary to know the camera *extrinsic parameters*. In fact, in the first case, it is

$$\boldsymbol{T}_o^b = \boldsymbol{T}_c^b \boldsymbol{T}_o^c, \tag{16}$$

where the elements of matrix $\boldsymbol{T}_c^b$ represent the extrinsic parameters of an eye-to-hand camera; on the other hand, in the case of eye-in-hand camera, it is

$$\boldsymbol{T}_o^e = \boldsymbol{T}_c^e \boldsymbol{T}_o^c, \tag{17}$$

where the extrinsic parameters matrix $\boldsymbol{T}_c^e$ characterize the pose of the camera with respect to the end-effector frame.

2.1 Interaction Matrix

If the object is in motion with respect to the camera, the feature vector $\boldsymbol{s}$ is, in general, time-varying. Therefore, it is possible to define a $(k \times 1)$ velocity vector in the image plane $\dot{\boldsymbol{s}}$.

The motion of the object with respect to the camera is characterized by the relative velocity

$$\boldsymbol{v}_{c,o}^c = \begin{bmatrix} \dot{\boldsymbol{o}}_{c,o}^c \\ \boldsymbol{R}_c^T(\boldsymbol{\omega}_o - \boldsymbol{\omega}_c) \end{bmatrix}, \tag{18}$$

where $\dot{\boldsymbol{o}}_{c,o}^c$ is the time derivative of vector $\boldsymbol{o}_{c,o}^c = \boldsymbol{R}_c^T(\boldsymbol{o}_o - \boldsymbol{o}_c)$, representing the relative position of the origin of the object frame with respect to the origin of the camera frame, while $\boldsymbol{\omega}_o$ and $\boldsymbol{\omega}_c$ are the angular velocities of the object frame and camera frame, respectively.

The equation relating $\dot{\boldsymbol{s}}$ to $\boldsymbol{v}_{c,o}^c$ is

$$\dot{\boldsymbol{s}} = \boldsymbol{J}_s(\boldsymbol{s}, \boldsymbol{T}_o^c) \boldsymbol{v}_{c,o}^c, \tag{19}$$

where $\boldsymbol{J}_s$ is a $(k \times 6)$ matrix termed *image Jacobian*. This equation is linear but $\boldsymbol{J}_s$ depends, in general, on the current value of the feature vector $\boldsymbol{s}$ and on the relative pose of the object with respect to the camera $\boldsymbol{T}_o^c$.

It is useful to consider also the mapping between the image plane velocity $\dot{\boldsymbol{s}}$, the absolute velocity of the camera frame

$$\boldsymbol{v}_c^c = \begin{bmatrix} \boldsymbol{R}_c^T \dot{\boldsymbol{o}}_c \\ \boldsymbol{R}_c^T \boldsymbol{\omega}_c, \end{bmatrix}$$

and the absolute velocity of the object frame

$$\boldsymbol{v}_o^c = \begin{bmatrix} \boldsymbol{R}_c^T \dot{\boldsymbol{o}}_o \\ \boldsymbol{R}_c^T \boldsymbol{\omega}_o \end{bmatrix}.$$

To this end, vector $\dot{\boldsymbol{o}}_{c,o}^c$ can be expressed as

$$\dot{\boldsymbol{o}}_{c,o}^c = \boldsymbol{R}_c^T(\dot{\boldsymbol{o}}_o - \dot{\boldsymbol{o}}_c) + \boldsymbol{S}(\boldsymbol{o}_{c,o}^c)\boldsymbol{R}_c^T\boldsymbol{\omega}_c,$$

which allows equality (18) to be rewritten in the compact form

$$\boldsymbol{v}_{c,o}^c = \boldsymbol{v}_o^c + \boldsymbol{\Gamma}(\boldsymbol{o}_{c,o}^c)\boldsymbol{v}_c^c, \tag{20}$$

with

$$\boldsymbol{\Gamma}(\cdot) = \begin{bmatrix} -\boldsymbol{I} & \boldsymbol{S}(\cdot) \\ \boldsymbol{O} & -\boldsymbol{I} \end{bmatrix}.$$

Therefore, Eq. (19) can be rewritten in the form

$$\dot{\boldsymbol{s}} = \boldsymbol{J}_s\boldsymbol{v}_o^c + \boldsymbol{L}_s\boldsymbol{v}_c^c, \tag{21}$$

where the $(k \times 6)$ matrix

$$\boldsymbol{L}_s = \boldsymbol{J}_s(\boldsymbol{s}, \boldsymbol{T}_o^c)\boldsymbol{\Gamma}(\boldsymbol{o}_{c,o}^c) \tag{22}$$

is termed *interaction matrix*. This matrix, in view of (21), defines the linear mapping between the absolute velocity of the camera $\boldsymbol{v}_c^c$ and the corresponding image plane velocity $\dot{\boldsymbol{s}}$, in the case that the object is fixed with respect to the base frame ($\boldsymbol{v}_o^c = \boldsymbol{0}$).

The analytic expression of the interaction matrix is, in general, simpler than that of the image Jacobian. The latter can be computed from the interaction matrix using the equation

$$\boldsymbol{J}_s(\boldsymbol{s}, \boldsymbol{T}_o^c) = \boldsymbol{L}_s\boldsymbol{\Gamma}(-\boldsymbol{o}_{c,o}^c), \tag{23}$$

obtained from (22), with $\boldsymbol{\Gamma}^{-1}(\boldsymbol{o}_{c,o}^c) = \boldsymbol{\Gamma}(-\boldsymbol{o}_{c,o}^c)$. In the following, examples of computation of interaction matrix and image Jacobian for some of the most common cases in applications are provided.

2.1.1 Interaction matrix of a point

Consider a point P of the object characterized, with respect to the camera frame, by the vector of coordinates

$$\boldsymbol{r}_c^c = \boldsymbol{R}_c^T(\boldsymbol{p} - \boldsymbol{o}_c), \tag{24}$$

where $\boldsymbol{p}$ is the position of point P with respect to the base frame. Choose vector $\boldsymbol{s}$ of normalized coordinates (12) as the feature vector of the point. In view of (10), the following expression holds:

$$\boldsymbol{s} = \boldsymbol{s}(\boldsymbol{r}_c^c), \tag{25}$$

with

$$\boldsymbol{s}(\boldsymbol{r}_c^c) = \frac{1}{z_c}\begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{bmatrix} X \\ Y \end{bmatrix}, \tag{26}$$

and $\boldsymbol{r}_c^c = [\, x_c \quad y_c \quad z_c \,]^T$. Computing the time derivative of (25) and using (26) yields

$$\dot{\boldsymbol{s}} = \frac{\partial \boldsymbol{s}(\boldsymbol{r}_c^c)}{\partial \boldsymbol{r}_c^c}\dot{\boldsymbol{r}}_c^c, \tag{27}$$

with

$$\frac{\partial \boldsymbol{s}(\boldsymbol{r}_c^c)}{\partial \boldsymbol{r}_c^c} = \frac{1}{z_c}\begin{bmatrix} 1 & 0 & -x_c/z_c \\ 0 & 1 & -y_c/z_c \end{bmatrix} = \frac{1}{z_c}\begin{bmatrix} 1 & 0 & -X \\ 0 & 1 & -Y \end{bmatrix}.$$

To compute the interaction matrix, vector $\dot{\boldsymbol{r}}_c^c$ can be obtained from the time derivative of (24) under the assumption that $\boldsymbol{p}$ is constant:

$$\dot{\boldsymbol{r}}_c^c = -\boldsymbol{R}_c^T\dot{\boldsymbol{o}}_c + \boldsymbol{S}(\boldsymbol{r}_c^c)\boldsymbol{R}_c^T\boldsymbol{\omega}_c = [\, -\boldsymbol{I} \quad \boldsymbol{S}(\boldsymbol{r}_c^c) \,]\, \boldsymbol{v}_c^c. \tag{28}$$

Combining Eqs. (27), (28), the following expression of interaction matrix of a point can be found:

$$\boldsymbol{L}_s(\boldsymbol{s}, z_c) = \begin{bmatrix} -\dfrac{1}{z_c} & 0 & \dfrac{X}{z_c} & XY & -(1+X^2) & Y \\ 0 & -\dfrac{1}{z_c} & \dfrac{Y}{z_c} & 1+Y^2 & -XY & -X \end{bmatrix}, \tag{29}$$

revealing that this matrix depends on the components of vector $\boldsymbol{s}$ and the sole component z_c of vector $\boldsymbol{r}_c^c$.

The image Jacobian of a point can be computed using (23), (29).

2.1.2 Interaction matrix of a set of points

The interaction matrix of a set of n points of the object $P_1, \ldots P_n$ can be built by considering the $(2n \times 1)$ feature vector (14). If $\boldsymbol{L}_{s_i}(\boldsymbol{s}_i, z_{c,i})$ denotes the interaction matrix corresponding to point P_i, then the interaction matrix of the set of points will be the $(2n \times 6)$ matrix

$$\boldsymbol{L}_s(\boldsymbol{s}, \boldsymbol{z}_c) = \begin{bmatrix} \boldsymbol{L}_{s_1}(\boldsymbol{s}_1, z_{c,1}) \\ \vdots \\ \boldsymbol{L}_{s_n}(\boldsymbol{s}_n, z_{c,n}) \end{bmatrix},$$

with $\boldsymbol{z}_c = [\, z_{c,1} \ldots z_{c,n} \,]^T$.

The image Jacobian of a set of points can be easily computed from the interaction matrix, using (23).

2.2 Algorithmic Solution

The interaction matrix $\boldsymbol{L}_s$ is, in general, a matrix of dimension $(k \times m)$, where k is equal to the number of feature parameters of the image and m is the dimension of velocity vector $\boldsymbol{v}_c^c$. Usually $m = 6$, but it may happen that $m < 6$, when the relative motion of the object with respect to the camera is constrained.

The image Jacobian $\boldsymbol{J}_s$ is also of dimension $(k \times m)$, being related to $\boldsymbol{L}_s$ by mapping (23). Since this mapping is invertible, the rank of $\boldsymbol{J}_s$ coincides with that of $\boldsymbol{L}_s$.

In the case that $\boldsymbol{L}_s$ is full rank, by using (19), it is possible to compute $\boldsymbol{v}_{c,o}^c$ from $\dot{\boldsymbol{s}}$.

In particular, if $k = m$, the velocity $\boldsymbol{v}_{c,o}^c$ can be obtained using the expression

$$\boldsymbol{v}_{c,o}^c = \boldsymbol{\Gamma}(\boldsymbol{o}_{c,o}^c)\boldsymbol{L}_s^{-1}\dot{\boldsymbol{s}}, \tag{30}$$

which requires the computation of the inverse of the interaction matrix.

In the case $k > m$, the interaction matrix has more rows than columns and Eq. (19) can be solved using a least-squares technique, whose solution can be written in the form

$$\boldsymbol{v}_{c,o}^c = \boldsymbol{\Gamma}(\boldsymbol{o}_{c,o}^c)(\boldsymbol{L}_s^T\boldsymbol{L}_s)^{-1}\boldsymbol{L}_s^T\dot{\boldsymbol{s}}, \tag{31}$$

where $(\boldsymbol{L}_s^T\boldsymbol{L}_s)^{-1}\boldsymbol{L}_s^T$ is the left pseudo-inverse of $\boldsymbol{L}_s$. This situation is rather frequent in applications, because it permits using interaction matrices with good condition numbers.

Finally, in the case $k < m$, the interaction matrix has more columns than rows and Eq. (19) admits infinite solutions. This implies that the number of parameters of the observed image is not sufficient to determine uniquely the relative motion of the object with respect to the camera. Hence, there exist relative motions of the object with respect to the camera (or vice versa) that do not produce variations of the image feature parameters. The velocities associated with these relative motions belong to the null subspace of $\boldsymbol{J}_s$, which has the same dimension of the null subspace of $\boldsymbol{L}_s$. If the problem is that of computing uniquely the relative pose of the object with respect to the camera from feature parameters in the image plane, this case has no interest.

The pose estimation problem may be cast in a form analogous to that of inverse kinematics algorithms for robot manipulators. To this end, it is necessary to represent the relative pose of the object with respect to the camera using a minimum number of coordinates, in terms of the $(m \times 1)$ vector

$$\boldsymbol{x}_{c,o} = \begin{bmatrix} \boldsymbol{o}_{c,o}^c \\ \boldsymbol{\phi}_{c,o} \end{bmatrix}, \tag{32}$$

where $\boldsymbol{o}_{c,o}^c$ characterizes the position of the origin of the object frame with respect to the camera frame and $\boldsymbol{\phi}_{c,o}$ characterizes the relative orientation. If Euler angles are used to represent orientation, then $\boldsymbol{\phi}_{c,o}$ is the vector of the angles extracted from rotation matrix $\boldsymbol{R}_o^c$ and the mapping between $\boldsymbol{v}_{c,o}^c$ and $\dot{\boldsymbol{x}}_{c,o}$ is expressed by

$$\boldsymbol{v}_{c,o}^c = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{T}(\boldsymbol{\phi}_{c,o}) \end{bmatrix} \dot{\boldsymbol{x}}_{c,o} = \boldsymbol{T}_A(\boldsymbol{\phi}_{c,o})\dot{\boldsymbol{x}}_{c,o}. \tag{33}$$

Equation (19) can be rewritten in the form

$$\dot{\boldsymbol{s}} = \boldsymbol{J}_{A_s}(\boldsymbol{s}, \boldsymbol{x}_{c,o})\dot{\boldsymbol{x}}_{c,o}, \tag{34}$$

where the matrix

$$\boldsymbol{J}_{A_s}(\boldsymbol{s}, \boldsymbol{x}_{c,o}) = \boldsymbol{L}_s\boldsymbol{\Gamma}(-\boldsymbol{o}_{c,o}^c)\boldsymbol{T}_A(\boldsymbol{\phi}_{c,o}) \tag{35}$$

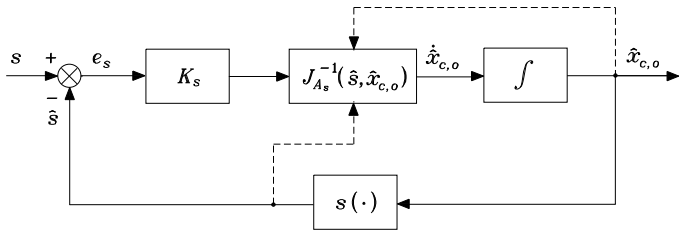

Figure 4: Pose estimation algorithm based on the inverse of image Jacobian

has a meaning analogous to that of the analytic Jacobian of a manipulator.

Equation (34) is the starting point of a numeric integration algorithm for the computation of $\boldsymbol{x}_{c,o}$, similar to inverse kinematics algorithms. Let $\widehat{\boldsymbol{x}}_{c,o}$ denote the current estimate of vector $\boldsymbol{x}_{c,o}$ and let

$$\widehat{\boldsymbol{s}} = \boldsymbol{s}(\widehat{\boldsymbol{x}}_{c,o})$$

be the corresponding vector of image feature parameters computed from the pose specified by $\widehat{\boldsymbol{x}}_{c,o}$; the objective of this algorithm is the minimization of the error

$$\boldsymbol{e}_s = \boldsymbol{s} - \widehat{\boldsymbol{s}}. \tag{36}$$

Notice that, for the purpose of numerical integration, vector $\boldsymbol{s}$ is constant while the current estimate $\widehat{\boldsymbol{s}}$ depends on the current integration time. Therefore, computing the time derivative of (36) yields

$$\dot{\boldsymbol{e}}_s = -\dot{\widehat{\boldsymbol{s}}} = -\boldsymbol{J}_{A_s}(\widehat{\boldsymbol{s}}, \widehat{\boldsymbol{x}}_{c,o})\dot{\widehat{\boldsymbol{x}}}_{c,o}. \tag{37}$$

Assumed that matrix $\boldsymbol{J}_{A_s}$ is square and nonsingular, the choice

$$\dot{\widehat{\boldsymbol{x}}}_{c,o} = \boldsymbol{J}_{A_s}^{-1}(\widehat{\boldsymbol{s}}, \widehat{\boldsymbol{x}}_{c,o})\boldsymbol{K}_s\boldsymbol{e}_s \tag{38}$$

leads to the equivalent linear system

$$\dot{\boldsymbol{e}}_s + \boldsymbol{K}_s\boldsymbol{e}_s = \boldsymbol{0}. \tag{39}$$

Therefore, if $\boldsymbol{K}_s$ is a positive definite matrix (usually diagonal), system (39) is asymptotically stable and the error tends to zero with a convergence speed that depends on the eigenvalues of matrix $\boldsymbol{K}_s$. The convergence to zero of error $\boldsymbol{e}_s$ ensures the asymptotic convergence of the estimate $\widehat{\boldsymbol{x}}_{c,o}$ to the true value $\boldsymbol{x}_{c,o}$.

The block scheme of the pose estimation algorithm is shown in Fig. 4, where $\boldsymbol{s}(\cdot)$ denotes the function computing the feature vector of the 'virtual' image corresponding to the current estimate $\widehat{\boldsymbol{x}}_{c,o}$ of the object pose with respect to the camera. This algorithm can be used as an alternative to the analytic methods for pose estimation. Obviously, the convergence properties depend on the choice of the image feature parameters and on the initial value of estimate $\widehat{\boldsymbol{x}}_{c,o}(0)$, which may produce instability problems related to the singularities of matrix $\boldsymbol{J}_{A_s}$.

Notice that, in view of (35), the singularities of matrix $\boldsymbol{J}_{A_s}$ are both the representation singularities of the orientation and those of the interaction matrix. The most critical singularities are those of the interaction matrix, since they depend on the choice of the image feature parameters.

To separate the effects of the two types of singularities, it is convenient to compute (38) in two steps, evaluating first

$$\widehat{\boldsymbol{v}}^c_{c,o} = \boldsymbol{\Gamma}(\boldsymbol{o}^c_{c,o})\boldsymbol{L}_s^{-1}\boldsymbol{K}_s\boldsymbol{e}_s, \tag{40}$$

and then

$$\dot{\widehat{\boldsymbol{x}}}_{c,o} = \boldsymbol{T}_A^{-1}(\boldsymbol{\phi}_{c,o})\widehat{\boldsymbol{v}}^c_{c,o}. \tag{41}$$

Assumed to work far from representation singularities, the problem of singularities of $\boldsymbol{L}_s$ can be overcome by using a number k of feature parameters greater than the minimum required m. This choice also allows a reduction of the effects of measurement noise. The resulting estimation algorithm requires the use of the left pseudo-inverse of $\boldsymbol{L}_s$ in place of the inverse, namely

$$\widehat{\boldsymbol{v}}^c_{c,o} = \boldsymbol{\Gamma}(\boldsymbol{o}^c_{c,o})(\boldsymbol{L}_s^T\boldsymbol{L}_s)^{-1}\boldsymbol{L}_s^T\boldsymbol{K}_s\boldsymbol{e}_s \tag{42}$$

in place of (40). The convergence of error (36) can be shown using the direct Lyapunov method based on the positive definite function

$$V(\boldsymbol{e}_s) = \frac{1}{2}\boldsymbol{e}_s^T\boldsymbol{K}_s\boldsymbol{e}_s > 0 \qquad \forall \boldsymbol{e}_s \neq \boldsymbol{0}.$$

Computing the time derivative of this function, and using (37), (35), (41), (42), yields

$$\dot{V} = -\boldsymbol{e}_s^T\boldsymbol{K}_s\boldsymbol{L}_s(\boldsymbol{L}_s^T\boldsymbol{L}_s)^{-1}\boldsymbol{L}_s^T\boldsymbol{K}_s\boldsymbol{e}_s,$$

which is negative semi-definite because $\mathcal{N}(\boldsymbol{L}_s^T) \neq \emptyset$, $\boldsymbol{L}_s^T$ being a matrix with more columns than rows. Therefore, the system is stable but not asymptotically stable. This implies that the error is bounded, but in some cases the algorithm can get stuck with $\boldsymbol{e}_s \neq \boldsymbol{0}$ and $\boldsymbol{K}_s\boldsymbol{e}_s \in \mathcal{N}(\boldsymbol{L}_s^T)$.

Notice that the pose estimation methods based on inverse Jacobian are as efficient in terms of accuracy, speed of convergence and computational load, as the initial estimate $\widehat{\boldsymbol{x}}_{c,o}(0)$ is close to the true value $\boldsymbol{x}_{c,o}$. Therefore, these methods are mainly adopted for real-time 'visual tracking' applications, where the estimate on an image taken at time $\bar{t}$ is computed assuming as initial value the estimate computed on the image taken at time $\bar{t} - T$, T being the sampling time of the image (multiple of the sampling time of the numerical integration algorithm).

3 Stereo Vision

The bidimensional image provided by a camera does not give any explicit information on depth, namely, the distance of the observed object from the camera. This information can be recovered in an indirect way from the geometric model of the object, assumed to be known.

On the other hand, the depth of a point can be directly computed in the case that two images of the same scene are available, taken from two different points of view. The two images can be obtained using two cameras, or sequentially, using a moving camera. These cases are referred to as *stereo vision*.

In the framework of stereo vision, two fundamental problems can be devised. The first is the *correspondence problem*, which consists of the identification of the points of the two images that are projections of the same point of the scene. These points are termed *conjugate* or *corresponding*. This problem is not easy to solve, and the solution is based on the existence of geometric constraints between two images of the same point, besides the fact that some details of the scene appear to be similar in the two images.

The second problem is that of *3D reconstruction* which, in general, consists of the computation of the relative pose of the cameras (calibrated and not) and thus, starting from this pose, the position in the 3D space of the points of the observed object.

An interesting case of application of 3D reconstruction occurs when the feature points of the observed object lie on the same plane. This geometric property represents a constraint between the projections of each point on the image plane of the two cameras, known as *planar homography*.

3.1 3D Reconstruction from Planar Homography

Assume that two cameras are available, with respective reference frames, denoted as 1 and 2. Moreover, let $\boldsymbol{o}^1_{1,2}$ denote the position vector and $\boldsymbol{R}^1_2$ the rotation matrix of Frame 2 with respect to Frame 1 and let $\boldsymbol{T}^1_2$ be the corresponding homogeneous transformation matrix. The coordinates of a point P expressed in the two frames are related by equation

$$\boldsymbol{p}^1 = \boldsymbol{o}^1_{1,2} + \boldsymbol{R}^1_2\boldsymbol{p}^2. \tag{43}$$

Let $\boldsymbol{n}^2$ denote the unit vector orthogonal to the plane containing the feature points and $d_2 > 0$ the distance of the plane from the origin of Frame 2. By virtue of simple geometric considerations, the following equation can be derived:

$$\frac{1}{d_2}\boldsymbol{n}^{2T}\boldsymbol{p}^2 = 1 \tag{44}$$

which defines the set of points $\boldsymbol{p}^2$ belonging to the plane. In view of the above equality, Eq. (43) can be rewritten in the form

$$\boldsymbol{p}^1 = \boldsymbol{H}\boldsymbol{p}^2, \tag{45}$$

with

$$\boldsymbol{H} = \boldsymbol{R}^1_2 + \frac{1}{d_2}\boldsymbol{o}^1_{1,2}\boldsymbol{n}^{2T}. \tag{46}$$

Let $\boldsymbol{s}_1$ and $\boldsymbol{s}_2$ be the coordinates of the projections of P on the image planes of the cameras; in view of (10) it is

$$\lambda_i\widetilde{\boldsymbol{s}}_i = \boldsymbol{\Pi}\widetilde{\boldsymbol{p}}^i = \boldsymbol{p}^i, \qquad i = 1, 2. \tag{47}$$

Replacing (47) into (45) yields

$$\widetilde{\boldsymbol{s}}_1 = \lambda\boldsymbol{H}\widetilde{\boldsymbol{s}}_2, \tag{48}$$

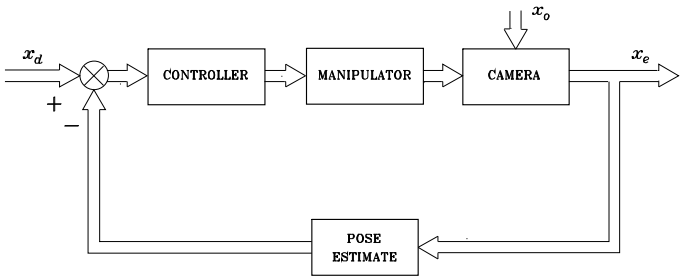

Figure 5: General block scheme of position-based visual servoing

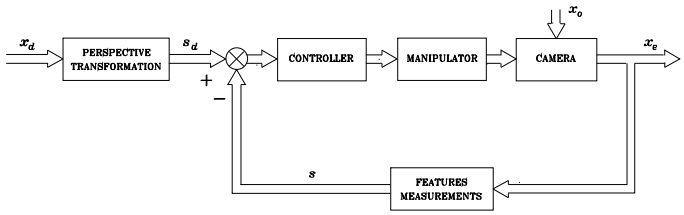

Figure 6: General block scheme of image-based visual servoing

where $\lambda = \lambda_2/\lambda_1 > 0$ is an arbitrary constant. Premultiplication of both sides of (48) by $\boldsymbol{S}(\widetilde{\boldsymbol{s}}_1)$ yields the equality

$$\boldsymbol{S}(\widetilde{\boldsymbol{s}}_1)\boldsymbol{H}\widetilde{\boldsymbol{s}}_2 = \boldsymbol{0}, \tag{49}$$

which is valid for the projections of all the points of the plane defined by (44). This equation is known a planar homography, defined by matrix $\boldsymbol{H}$.

Using a derivation similar to that adopted for solving the PnP problem, it is possible to compute numerically matrix $\zeta\boldsymbol{H}$, up to a scaling factor ζ, starting from the coordinates of n points of the plane, with $n \geq 4$.

The value of the scaling factor ζ can be computed using a numerical algorithm based on expression (46) of matrix $\boldsymbol{H}$; once $\boldsymbol{H}$ is known, it is possible to compute quantities $\boldsymbol{R}_2^1$, $\boldsymbol{o}_{1,2}^1/d_2$ and $\boldsymbol{n}^2$ in (46) — actually, it can be shown that two admissible solutions exist.

This result is of a certain relevance for visual servoing applications. For example, in the case of a camera in motion with respect to the object, if Frames 1 and 2 represent the poses of the camera in two different time instants, the computation of $\boldsymbol{H}$ with decomposition (46) can be used to evaluate the orientation displacement of the camera frame and the position displacement of the origin, the latter defined up to a scaling factor d_2. This information can be achieved without knowing the object geometry, as long as the feature points all belong to the same plane.

4 The Visual Servoing Problem

Visual measurements allow a robot to collect information on the surrounding environment. In the case of robot manipulators, such information is typically used to compute the end-effector pose with respect to an object observed by the camera. The objective of visual servoing is to ensure that the end-effector, on the basis of visual measurements elaborated in real time, reaches and keeps a (constant or time-varying) desired pose with respect to the observed object.

It is worth remarking that the direct measurements provided by the visual system are concerned with feature parameters in the image plane, while the robotic task is defined in the operational space, in terms of the relative pose of the end-effector with respect to the object. This fact naturally leads to considering two kinds of control approaches, illustrated by the block schemes of Figs. 5 and 6, namely *position-based visual servoing*, also termed *visual servoing in the operational space*, and *image-based visual servoing*, also termed *visual servoing in the image space*. In these schemes, the case of eye-in-hand camera is considered; for eye-to-hand cameras, similar schemes can be adopted.

The *position-based visual servoing* approach is conceptually similar to the operational space control. The main difference is that feedback is based on the real-time estimation of the pose of the observed object with respect to the camera using visual measurements. Estimation can be performed analytically or using iterative numerical algorithms. Its conceptual advantage regards the possibility of acting directly on operational space variables. Therefore, the control parameters can be selected on the basis of suitable specifications imposed to the time response of the end-effector motion variables, both at steady state and during the transient. The drawback of this approach is that, due to the absence of direct control of the image features, the object may exit from the camera field of view during the transient or as a consequence of planning errors; hence, the feedback loop turns out to be open due to lack of visual measurements and instability may occur.

In the *image-space visual servoing* approach, the control action is computed on the basis of the error defined as the difference between the value of the image feature parameters in the desired configuration — computed using perspective transformation or directly measured with the camera in the desired pose — and the value of the parameters measured with the camera in the current pose. The conceptual advantage of this solution regards the fact that the real-time estimate of the pose of the object with respect to the camera is not required. Moreover, since the control acts directly in the image feature parameters, it is possible to keep the object within the camera field of view during the motion. However, due to the nonlinearity of the mapping between the image feature parameters and the operational space variables, singular configurations may occur, which cause instability or saturation of the control action. Also, the end-effector trajectories cannot be easily predicted in advance and may produce collisions with obstacles or joint limits violation.

To compare the two control strategies it is also worth

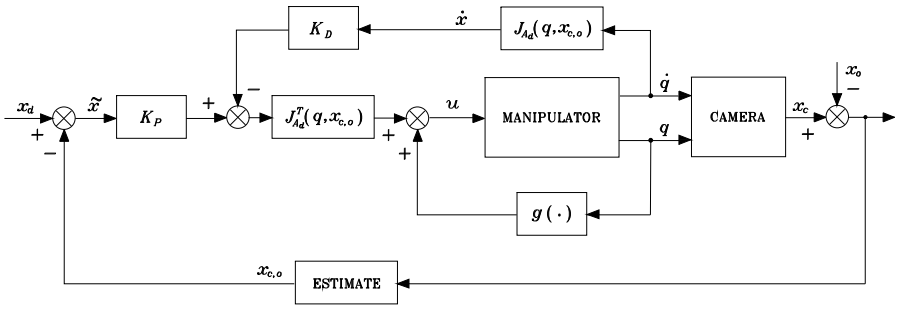

Figure 7: Block scheme of position-based visual servoing of PD type with gravity compensation

considering the operating conditions. Of particular importance is the issue of camera calibration. It is easy to understand that position-based visual servoing is more sensitive to camera calibration errors compared to image-based visual servoing. In fact, for the first approach, the presence of uncertainties on calibration parameters, both intrinsic and extrinsic, produces errors on the estimate of operational space variables that may be seen as an external disturbance acting on the feedback path of the control loop, where disturbance rejection capability is low. On the other hand, in the image-based visual servoing approach, the quantities used for the computation of the control action are directly defined in the image plane and measured in pixel units; moreover, the desired value of the feature parameters is measured using the camera. This implies that the uncertainty affecting calibration parameters can be seen as a disturbance acting on the forward path of the control loop, where disturbance rejection capability is high.

A further aspect to analyze concerns knowledge of the geometric model of the object. It is evident that, for position-based visual servoing, the object geometry must be known if only one camera is used, because it is necessary for pose estimation, while it may be unknown when a stereo camera system is employed. On the other hand, image-based visual servoing does not require, in principle, knowledge of the object geometry, even for mono-camera systems.

On the above premises, in the following, the main position-based and image-based visual servoing schemes are illustrated. For both approaches, the problem of regulation to a constant set-point is presented and the object is assumed to be fixed with respect to the base frame. Without loss of generality, the case of a single calibrated camera, mounted on the manipulator's end-effector, is considered (see Fig. 3); moreover, the end-effector frame is chosen so as to coincide with the camera frame.

5 Position-based Visual Servoing

In position-based visual servoing schemes, visual measurements are used to estimate in real time the homogeneous transformation matrix $\boldsymbol{T}_o^c$, representing the relative pose of the object frame with respect to the camera frame. From matrix $\boldsymbol{T}_o^c$, the $(m \times 1)$ vector of independent coordinates $\boldsymbol{x}_{c,o}$, defined in (32), can be extracted.

Assumed that the object is fixed with respect to the base frame, the position-based visual servoing problem can be formulated by imposing a desired value to the relative pose of the object frame with respect to the camera frame. This quantity can be specified in terms of homogeneous transformation matrix $\boldsymbol{T}_o^d$, where superscript d denotes the desired pose of the camera frame. From this matrix, the $(m \times 1)$ operational space vector $\boldsymbol{x}_{d,o}$ can be extracted.

Matrices $\boldsymbol{T}_o^c$ and $\boldsymbol{T}_o^d$ can be used to obtain the homogeneous transformation matrix

$$\boldsymbol{T}_c^d = \boldsymbol{T}_o^d (\boldsymbol{T}_o^c)^{-1} = \begin{bmatrix} \boldsymbol{R}_c^d & \boldsymbol{o}_{d,c}^d \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \tag{50}$$

expressing the position and orientation displacement of the camera frame in the current pose with respect to the desired pose. From this matrix, a suitable error vector in the operational space can be computed, defined as

$$\tilde{\boldsymbol{x}} = -\begin{bmatrix} \boldsymbol{o}_{d,c}^d \\ \boldsymbol{\phi}_{d,c} \end{bmatrix}, \tag{51}$$

where $\boldsymbol{\phi}_{d,c}$ is the vector of the Euler angles extracted from rotation matrix $\boldsymbol{R}_c^d$. Vector $\tilde{\boldsymbol{x}}$ does not depend on the object pose and represents the error between the desired pose and the current pose of the camera frame. It is worth observing that this vector does not coincide with the difference between $\boldsymbol{x}_{d,o}$ and $\boldsymbol{x}_{c,o}$, but it can be computed from the corresponding homogeneous transformation matrices, using (50), (51).

The control has to be designed so that the operational space error $\tilde{\boldsymbol{x}}$ tends to zero asymptotically.

Notice that, for the choice of the set point $\boldsymbol{x}_{d,o}$, the knowledge of the object pose is not required. However, the control objective can be satisfied provided that the desired pose of the camera frame with respect to the base frame, corresponding to the homogeneous transformation matrix

$$\boldsymbol{T}_d = \boldsymbol{T}_c (\boldsymbol{T}_c^d)^{-1} = \begin{bmatrix} \boldsymbol{R}_d & \boldsymbol{o}_d \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \tag{52}$$

belongs to the dexterous workspace of the manipulator. If the object is fixed with respect to the base frame, this matrix is constant.

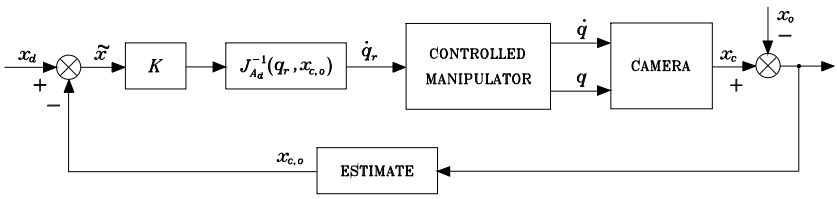

Figure 8: Block scheme of resolved-velocity position-based visual servoing

5.1 PD Control with Gravity Compensation

Position-based visual servoing can be implemented using PD control with gravity compensation, suitably modified with respect to that used for motion control.

Computing the time derivative of (51), for the position part, gives

$$\dot{\boldsymbol{o}}_{d,c}^d = \dot{\boldsymbol{o}}_c^d - \dot{\boldsymbol{o}}_d^d = \boldsymbol{R}_d^T \dot{\boldsymbol{o}}_c,$$

while, for the orientation part, it gives

$$\dot{\boldsymbol{\phi}}_{d,c} = \boldsymbol{T}^{-1}(\boldsymbol{\phi}_{d,c})\boldsymbol{\omega}_{d,c}^d = \boldsymbol{T}^{-1}(\boldsymbol{\phi}_{d,c})\boldsymbol{R}_d^T \boldsymbol{\omega}_c.$$

To compute the above expressions, equalities $\dot{\boldsymbol{o}}_d^d = \boldsymbol{0}$ and $\boldsymbol{\omega}_d^d = \boldsymbol{0}$ have been taken into account, observing that $\boldsymbol{o}_d$ and $\boldsymbol{R}_d$ are constant. Therefore, $\dot{\tilde{\boldsymbol{x}}}$ has the expression

$$\dot{\tilde{\boldsymbol{x}}} = -\boldsymbol{T}_A^{-1}(\boldsymbol{\phi}_{d,c}) \begin{bmatrix} \boldsymbol{R}_d^T & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}_d^T \end{bmatrix} \boldsymbol{v}_c. \tag{53}$$

Since the end-effector frame and the camera frame coincide, the following equality holds:

$$\dot{\tilde{\boldsymbol{x}}} = -\boldsymbol{J}_{A_d}(\boldsymbol{q}, \tilde{\boldsymbol{x}})\dot{\boldsymbol{q}}, \tag{54}$$

where the matrix

$$\boldsymbol{J}_{A_d}(\boldsymbol{q}, \tilde{\boldsymbol{x}}) = \boldsymbol{T}_A^{-1}(\boldsymbol{\phi}_{d,c}) \begin{bmatrix} \boldsymbol{R}_d^T & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}_d^T \end{bmatrix} \boldsymbol{J}(\boldsymbol{q}) \tag{55}$$

has the meaning of analytic Jacobian of the manipulator in the operational space.

Position-based visual servoing of PD type with gravity compensation has the expression

$$\boldsymbol{u} = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}_{A_d}^T(\boldsymbol{q}, \tilde{\boldsymbol{x}})(\boldsymbol{K}_P\tilde{\boldsymbol{x}} - \boldsymbol{K}_D\boldsymbol{J}_{A_d}(\boldsymbol{q}, \tilde{\boldsymbol{x}})\dot{\boldsymbol{q}}). \tag{56}$$

The asymptotic stability of the equilibrium pose corresponding to $\tilde{\boldsymbol{x}} = \boldsymbol{0}$, under the assumption of symmetric and positive definite matrices $\boldsymbol{K}_P$, $\boldsymbol{K}_D$, can be proven using the Lyapunov function

$$\boldsymbol{V}(\dot{\boldsymbol{q}}, \tilde{\boldsymbol{x}}) = \frac{1}{2}\dot{\boldsymbol{q}}^T\boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{1}{2}\tilde{\boldsymbol{x}}^T\boldsymbol{K}_P\tilde{\boldsymbol{x}} > 0 \qquad \forall \dot{\boldsymbol{q}}, \tilde{\boldsymbol{x}} \neq \boldsymbol{0}.$$

Notice that, for the computation of control law (56), the estimation of $\boldsymbol{x}_{c,o}$ and the measurements of $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$ are required. Moreover, the derivative term can also be chosen as $-\boldsymbol{K}_D\dot{\boldsymbol{q}}$.

The block scheme of position-based visual servoing of PD type with gravity compensation is shown in Fig. 7. Notice that the sum block computing error $\tilde{\boldsymbol{x}}$ and that computing the output of the controlled system have a purely conceptual meaning and do not correspond to algebraic sums.

5.2 Resolved-velocity Control

The information deriving from visual measurements is computed at a frequency lower or equal to the camera frame rate. This quantity, especially for CCD cameras, is at least of one order of magnitude lower than the typical frequencies used for motion control of robot manipulators. As a consequence, in the digital implementation of control law (56), to preserve stability of the closed-loop system, the control gains must be set to values much lower than those typically used for motion control; therefore, the performance of the closed-loop system in terms of speed of convergence and disturbance rejection capability turns out to be poor.

This problem can be avoided assuming that the manipulator is equipped with a high-gain motion controller in the joint space or in the operational space. Neglecting the effects on the tracking errors deriving from manipulator dynamics and disturbances, the controlled manipulator can be considered as an ideal positioning device. This implies that, in the case of joint space motion control, the following equality holds:

$$\boldsymbol{q}(t) \approx \boldsymbol{q}_r(t), \tag{57}$$

$\boldsymbol{q}_r(t)$ being the imposed reference trajectory for the joint variables.

Therefore, visual servoing can be achieved by computing the trajectory $\boldsymbol{q}_r(t)$ on the basis of visual measurements, so that the operational space tracking error (51) goes asymptotically to zero.

To this end, Eq. (54) suggests the following choice for the joint space reference velocity:

$$\dot{\boldsymbol{q}}_r = \boldsymbol{J}_{A_d}^{-1}(\boldsymbol{q}_r, \tilde{\boldsymbol{x}})\boldsymbol{K}\tilde{\boldsymbol{x}} \tag{58}$$

which, replaced in (54), by virtue of equality (57), yields the linear equation

$$\dot{\tilde{\boldsymbol{x}}} + \boldsymbol{K}\tilde{\boldsymbol{x}} = \boldsymbol{0}. \tag{59}$$

This equality, for a positive definite matrix $\boldsymbol{K}$, implies that the operational space error tends to zero asymptotically with a convergence of exponential type and speed depending on the eigenvalues of matrix $\boldsymbol{K}$; the larger the eigenvalues, the faster the convergence.

The above scheme is termed *resolved-velocity control* in the operational space, because it is based on the computation of velocity $\dot{\boldsymbol{q}}_r$ from the operational space error. Trajectory $\boldsymbol{q}_r(t)$ is computed from (58) via a simple integration.

The block scheme of resolved-velocity position-based visual servoing is reported in Fig. 8. Also in this case, the

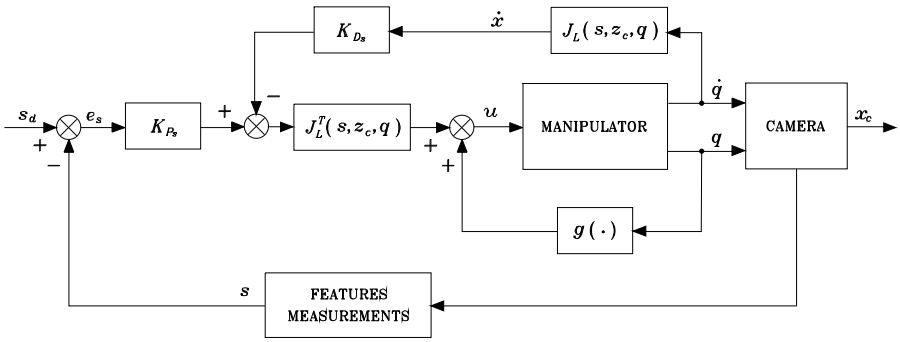

Figure 9: Block scheme of image-based visual servoing of PD type with gravity compensation

sum block computing the error $\widetilde{\boldsymbol{x}}$ and that computing the output of the scheme have a purely conceptual meaning and do not correspond to algebraic sums.

Notice that the choice of $\boldsymbol{K}$ influences the transient behaviour of the trajectory of the camera frame, which is the solution to the differential equation (59). If $\boldsymbol{K}$ is a diagonal matrix with the same gains for the positional part, the origin of the camera frame follows the line segment connecting the initial position to the desired position. On the other hand, the orientation trajectory depends on the particular choice of Euler angles and, more in general, of the orientation error. The possibility of knowing in advance the trajectory of the camera is important because, during the motion, the object may exit from the camera field of view, making visual measurements unavailable.

6 Image-based Visual Servoing

If the object is fixed with respect to the base frame, image-based visual servoing can be formulated by stipulating that the vector of the object feature parameters has a desired constant value $\boldsymbol{s}_d$ corresponding to the desired pose of the camera. Therefore, it is implicitly assumed that a desired pose $\boldsymbol{x}_{d,o}$ exists so that the camera pose belongs to the dexterous workspace of the manipulator and

$$\boldsymbol{s}_d = \boldsymbol{s}(\boldsymbol{x}_{d,o}). \tag{60}$$

Moreover, $\boldsymbol{x}_{d,o}$ is supposed to be unique. To this end, the feature parameters can be chosen as the coordinates of n points of the object, with $n \geq 4$ for coplanar points (and no triplets of collinear points) or $n \geq 6$ in case of non-coplanar points. Notice that, if the operational space dimension is $m < 6$, as for the case of SCARA manipulator, a reduced number of points can be used.

The interaction matrix $\boldsymbol{L}_s(\boldsymbol{s}, \boldsymbol{z}_c)$ depends on variables $\boldsymbol{s}$ and $\boldsymbol{z}_c$ with $\boldsymbol{z}_c = [\, z_{c,1} \ldots z_{c,n}]^T$, $z_{c,i}$ being the third coordinate of the generic feature point of the object.

It is worth noticing that the task is assigned directly in terms of feature vector $\boldsymbol{s}_d$, while pose $\boldsymbol{x}_{d,o}$ does not need to be known. In fact, $\boldsymbol{s}_d$ can be computed by measuring the feature parameters when the object is in the desired pose with respect to the camera.

The control law must be designed so as to guarantee that the image space error

$$\boldsymbol{e}_s = \boldsymbol{s}_d - \boldsymbol{s} \tag{61}$$

tends asymptotically to zero.

6.1 PD Control with Gravity Compensation

Image-based visual servoing can be implemented using a PD control with gravity compensation defined on the basis of the image space error.

To this end, consider the following positive definite quadratic form as Lypapunov function candidate:

$$\boldsymbol{V}(\dot{\boldsymbol{q}}, \boldsymbol{e}_s) = \frac{1}{2}\dot{\boldsymbol{q}}^T \boldsymbol{B}(\boldsymbol{q})\dot{\boldsymbol{q}} + \frac{1}{2}\boldsymbol{e}_s^T \boldsymbol{K}_{Ps}\boldsymbol{e}_s > 0 \qquad \forall \dot{\boldsymbol{q}}, \boldsymbol{e}_s \neq \boldsymbol{0}, \tag{62}$$

with $\boldsymbol{K}_{Ps}$ symmetric and positive definite $(k \times k)$ matrix.

Computing the time derivative of (62) and taking into account the expression of the joint space dynamic model of the manipulator and the skew-symmetry of $\dot{\boldsymbol{B}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ yields

$$\dot{\boldsymbol{V}} = -\dot{\boldsymbol{q}}^T \boldsymbol{F}\dot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^T(\boldsymbol{u} - \boldsymbol{g}(\boldsymbol{q})) + \dot{\boldsymbol{e}}_s^T \boldsymbol{K}_{Ps}\boldsymbol{e}_s. \tag{63}$$

Since $\dot{\boldsymbol{s}}_d = \boldsymbol{0}$ and the object is fixed with respect to the base frame, the following equality holds:

$$\dot{\boldsymbol{e}}_s = -\dot{\boldsymbol{s}} = -\boldsymbol{J}_L(\boldsymbol{s}, \boldsymbol{z}_c, \boldsymbol{q})\dot{\boldsymbol{q}}, \tag{64}$$

where

$$\boldsymbol{J}_L(\boldsymbol{s}, \boldsymbol{z}_c, \boldsymbol{q}) = \boldsymbol{L}_s(\boldsymbol{s}, \boldsymbol{z}_c) \begin{bmatrix} \boldsymbol{R}_c^T & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}_c^T \end{bmatrix} \boldsymbol{J}(\boldsymbol{q}), \tag{65}$$

the camera frame and the end-effector frame being coincident.

Therefore, with the choice

$$\boldsymbol{u} = \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}_L^T(\boldsymbol{s}, \boldsymbol{z}_c, \boldsymbol{q})\left(\boldsymbol{K}_{Ps}\boldsymbol{e}_s - \boldsymbol{K}_{Ds}\boldsymbol{J}_L(\boldsymbol{s}, \boldsymbol{z}_c, \boldsymbol{q})\dot{\boldsymbol{q}}\right), \tag{66}$$

where $\boldsymbol{K}_{Ds}$ is a symmetric and positive definite $(k \times k)$ matrix, Eq. (63) becomes

$$\dot{V} = -\dot{\boldsymbol{q}}^T \boldsymbol{F}\dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}^T \boldsymbol{J}_L^T \boldsymbol{K}_{Ds}\boldsymbol{J}_L\dot{\boldsymbol{q}}. \tag{67}$$

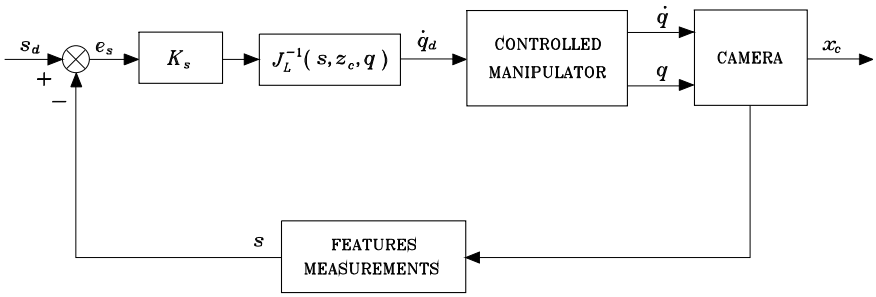

Figure 10: Block scheme of resolved-velocity image-based visual servoing

Control law (66) includes a nonlinear compensation action of gravitational forces in the joint space and a linear PD action in the image space. The last term, in view of (64), corresponds to a derivative action in the image space and has been added to increase damping. The resulting block scheme is reported in Fig. 9.

The direct measurement of $\dot{\boldsymbol{s}}$ would permit the computation of the derivative term as $-\boldsymbol{K}_{Ds}\dot{\boldsymbol{s}}$; this measurement, however, is not available. As an alternative, the derivative term can simply be set as $-\boldsymbol{K}_D\dot{\boldsymbol{q}}$, with $\boldsymbol{K}_D$ symmetric and positive definite $(n \times n)$ matrix.

Equation (67) reveals that, for all trajectories of the system, the Lyapunov function decreases until $\dot{\boldsymbol{q}} \neq \boldsymbol{0}$. Therefore the system reaches an equilibrium state, characterized by

$$\boldsymbol{J}_L^T(\boldsymbol{s}, \boldsymbol{z}_c, \boldsymbol{q})\boldsymbol{K}_{Ps}\boldsymbol{e}_s = \boldsymbol{0}. \tag{68}$$

Equations (68), (65) show that, if the interaction matrix and the geometric Jacobian of the manipulator are full rank, then $\boldsymbol{e}_s = \boldsymbol{0}$, which is the sought result.

Notice that control law (66) requires not only the measurement of $\boldsymbol{s}$ but also the computation of vector $\boldsymbol{z}_c$ which, in the image-based visual servoing philosophy, should be avoided. In some applications $\boldsymbol{z}_c$ is known with good approximation, as in the case that the relative motion of the camera with respect to the object belongs to a plane. Alternatively, estimated or constant values can be used for $\boldsymbol{z}_c$, as the value in the initial configuration or that in the desired configuration. This is equivalent to using an estimate $\widehat{\boldsymbol{L}}_s$ of the interaction matrix. In such cases, however, the stability proof becomes much more complex.

6.2 Resolved-velocity Control

The concept of resolved-velocity control can easily be extended to the image space. In such a case, Eq. (64) suggests the following choice of the reference velocity in joint space

$$\dot{\boldsymbol{q}}_r = \boldsymbol{J}_L^{-1}(\boldsymbol{s}, \boldsymbol{z}_c, \boldsymbol{q}_r)\boldsymbol{K}_s\boldsymbol{e}_s, \tag{69}$$

under the assumption of invertible matrix $\boldsymbol{J}_L$. This control law, replaced in (64), yields the linear equation

$$\dot{\boldsymbol{e}}_s + \boldsymbol{K}_s\boldsymbol{e}_s = \boldsymbol{0}. \tag{70}$$

Therefore, if $\boldsymbol{K}_s$ is a positive definite matrix, Eq. (70) is asymptotically stable and error $\boldsymbol{e}_s$ tends asymptotically to zero with convergence of exponential type and speed depending on the eigenvalues of matrix $\boldsymbol{K}_s$. The convergence to zero of the image space error $\boldsymbol{e}_s$ ensures the asymptotic convergence of $\boldsymbol{x}_{c,o}$ to the desired pose $\boldsymbol{x}_{d,o}$.

The block scheme of resolved-velocity image-based visual servoing is shown in Fig. 10.

Notice that this control scheme requires the computation of the inverse of matrix $\boldsymbol{J}_L$; therefore it is affected by problems related to the singularities of this matrix which, in view of (65), are both those of the geometric Jacobian and those of the interaction matrix. The most critical singularities are those of the interaction matrix, since they depend on the choice of the image feature parameters.

Therefore, it is convenient to compute control law (69) in two steps. The first step is the computation of vector

$$\boldsymbol{v}_r^c = \boldsymbol{L}_s^{-1}(\boldsymbol{s}, \boldsymbol{z}_c)\boldsymbol{K}_s\boldsymbol{e}_s. \tag{71}$$

The second step is the computation of the joint space reference velocity using the relationship

$$\dot{\boldsymbol{q}}_r = \boldsymbol{J}^{-1}(\boldsymbol{q}) \begin{bmatrix} \boldsymbol{R}_c & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{R}_c \end{bmatrix} \boldsymbol{v}_r^c. \tag{72}$$

Far from the kinematic singularities of the manipulator, the problem of the singularities of the interaction matrix can be overcome by using a number k of feature parameters greater than the minimum required m, similarly to the case considered in Sect. 2.2. The control law can be modified by using the left pseudo-inverse of interaction matrix $\boldsymbol{L}_s$ in place of the inverse, namely

$$\boldsymbol{v}_r^c = (\boldsymbol{L}_s^T\boldsymbol{L}_s)^{-1}\boldsymbol{L}_s^T\boldsymbol{K}_s\boldsymbol{e}_s \tag{73}$$

in place of (71). Stability of the closed-loop system with control law (72), (73) can be shown using the direct Lyapunov method based on the positive definite function

$$V(\boldsymbol{e}_s) = \frac{1}{2}\boldsymbol{e}_s^T\boldsymbol{K}_s\boldsymbol{e}_s > 0 \qquad \forall \boldsymbol{e}_s \neq \boldsymbol{0}.$$

Computing the time derivative of this function and taking into account (64), (65), (72), (73), yields

$$\dot{V} = -\boldsymbol{e}_s^T\boldsymbol{K}_s\boldsymbol{L}_s(\boldsymbol{L}_s^T\boldsymbol{L}_s)^{-1}\boldsymbol{L}_s^T\boldsymbol{K}_s\boldsymbol{e}_s$$

which is negative semi-definite because $\mathcal{N}(\boldsymbol{L}_s^T) \neq \emptyset$, $\boldsymbol{L}_s^T$ being a matrix with more columns than rows. Therefore,

the closed-loop system is stable but not asymptotically stable. This implies that the error is bounded, but in some cases the system may reach an equilibrium with $\boldsymbol{e}_s \neq \boldsymbol{0}$ and $\boldsymbol{K}_s\boldsymbol{e}_s \in \mathcal{N}(\boldsymbol{L}_s^T)$.

Another problem connected with the implementation of control law (71) or (73) and (72) depends on the fact that the computation of interaction matrix $\boldsymbol{L}_s$ requires knowledge of $\boldsymbol{z}_c$. Similar to Sect. 6.1, this problem can be solved by using an estimate of matrix $\widehat{\boldsymbol{L}}_s^{-1}$ (or of its pseudo-inverse). In this case, the Lypapunov method can be used to prove that the control scheme remains stable provided that matrix $\boldsymbol{L}_s\widehat{\boldsymbol{L}}_s^{-1}$ is positive definite. Notice that $\boldsymbol{z}_c$ is the only information depending on object geometry. Therefore, it can also be seen that image-based visual servoing, in the case that only one camera is used, does not require exact knowledge of object geometry.

The choice of the elements of matrix $\boldsymbol{K}_s$ influences the trajectories of the feature parameters, which are solution to differential equation (70). In the case of feature points, if a diagonal matrix $\boldsymbol{K}_s$ with equal elements is set, the projections of these points on the image plane will follow line segments. The corresponding camera motion, however, cannot be easily predicted, because of the nonlinearity of the mapping between image plane variables and operational space variables.

7 Hybrid Visual Servoing

An approach which combines the benefits of position-based and image-based visual servoing is *hybrid visual servoing*. The name stems from the fact that the control error is defined in the operational space for some components and in the image space for the others. This implies that a desired motion can be specified, at least partially, in the operational space so that the camera trajectory during visual servoing can be predicted in advance for some components. On the other hand, the presence of error components in the image space helps keep the image features in the camera field of view, which is a difficult task in position-based approaches.

Hybrid visual servoing requires the estimation of some operational space variables. Assume that the object has a planar surface where at least four feature points, and no triplets of collinear points, can be selected. Using the coordinates of these points in the camera image plane, both in the current and in the desired pose of the camera frame, it is possible to compute the planar homography $\boldsymbol{H}$ as described in Sect. 3.1. Notice that, for this computation, knowledge of the current and the desired camera pose is not required, provided that the feature vectors $\boldsymbol{s}$ and $\boldsymbol{s}_d$ are known.

In view of (46), assuming that Frame 1 coincides with the camera frame in the current pose and Frame 2 coincides with the camera frame in the desired pose, the following equality holds:

$$\boldsymbol{H} = \boldsymbol{R}_d^c + \frac{1}{d_d}\boldsymbol{o}_{c,d}^c\boldsymbol{n}^{dT},$$

where $\boldsymbol{R}_d^c$ is the rotation matrix between the desired orientation and the current orientation of the camera frame, $\boldsymbol{o}_{c,d}^c$ is the position vector of the origin of the camera frame in the desired pose with respect to the current pose, $\boldsymbol{n}^d$ is the unit vector normal to the plane containing the feature points, and d_d is the distance between this plane and the origin of the camera frame in the desired pose. The quantities $\boldsymbol{R}_d^c$, $\boldsymbol{n}^d$, $(1/d_d)\boldsymbol{o}_{c,d}^c$, in the current camera pose, can be computed at each sampling time from matrix $\boldsymbol{H}$.

Adopting a resolved-velocity approach, the control objective consists of computing the reference absolute velocity of the camera frame

$$\boldsymbol{v}_r^c = \begin{bmatrix} \boldsymbol{\nu}_r^c \\ \boldsymbol{\omega}_r^c \end{bmatrix}$$

from a suitably defined error vector.

To this end, the orientation error between the desired and the current camera pose can be computed from matrix $\boldsymbol{R}_d^c$, as for position-based visual servoing. If $\boldsymbol{\phi}_{c,d}$ denotes the vector of the Euler angles extracted from $\boldsymbol{R}_d^c$, the control vector $\boldsymbol{\omega}_r^c$ can be chosen as

$$\boldsymbol{\omega}_r^c = -\boldsymbol{T}(\boldsymbol{\phi}_{c,d})\boldsymbol{K}_o\boldsymbol{\phi}_{c,d}, \tag{74}$$

where $\boldsymbol{K}_o$ is a (3×3) matrix. With this choice, the equation of the orientation error has the form

$$\dot{\boldsymbol{\phi}}_{c,d} + \boldsymbol{K}_o\boldsymbol{\phi}_{c,d} = \boldsymbol{0}. \tag{75}$$

Equation (75), if $\boldsymbol{K}_o$ is a symmetric and positive definite matrix, implies that the orientation error tends to zero asymptotically with convergence of exponential type and speed depending on the eigenvalues of matrix $\boldsymbol{K}_o$.

The control vector $\boldsymbol{\nu}_r^c$ should be selected so that the positional part of the error between the desired and the current camera pose converges to zero. The position error could be defined as the difference of the coordinates of a point of the object in the desired camera frame $\boldsymbol{r}_d^c = [\,x_d \quad y_d \quad z_d\,]^T$ and those in the current camera frame $\boldsymbol{r}_c^c = [\,x_c \quad y_c \quad z_c\,]^T$, namely $\boldsymbol{r}_d^c - \boldsymbol{r}_c^c$. These coordinates, however, cannot be directly measured, unlike the corresponding coordinates in the image plane, defining the feature vectors $\boldsymbol{s}_{p,d} = [\,X_d \quad Y_d\,]^T = [\,x_d/z_d \quad y_d/z_d\,]^T$ and $\boldsymbol{s}_p = [\,X \quad Y\,]^T = [\,x_c/z_c \quad y_c/z_c\,]^T$.

The information deriving from the computation of homography $\boldsymbol{H}$ can be used to rewrite the ratio

$$\rho_z = z_c/z_d$$

in terms of known or measurable quantities in the form

$$\rho_z = \frac{d_c}{d_d}\frac{\boldsymbol{n}^{dT}\widetilde{\boldsymbol{s}}_{p,d}}{\boldsymbol{n}^{cT}\widetilde{\boldsymbol{s}}_p} \tag{76}$$

with

$$\frac{d_c}{d_d} = 1 + \boldsymbol{n}^{cT}\frac{\boldsymbol{o}_{c,d}^c}{d_d} = \det(\boldsymbol{H}), \tag{77}$$

and $\boldsymbol{n}^c = \boldsymbol{R}_d^c\boldsymbol{n}^d$, where vectors $\widetilde{\boldsymbol{s}}_p$ and $\widetilde{\boldsymbol{s}}_{p,d}$ denote the representations in homogeneous coordinates of $\boldsymbol{s}_p$ and $\boldsymbol{s}_{p,d}$, respectively.

The position error, expressed in terms of known or measurable quantities, can be defined as

$$\boldsymbol{e}_p(\boldsymbol{r}_d^c, \boldsymbol{r}_c^c) = \begin{bmatrix} X_d - X \\ Y_d - Y \\ \ln\rho_z \end{bmatrix}.$$

Notice that, in view of (10), convergence to zero of $\boldsymbol{e}_p$ implies convergence to zero of $\boldsymbol{r}_d^c - \boldsymbol{r}_c^c$ and vice versa.

Computing the time derivative of $\boldsymbol{e}_p$ yields

$$\dot{\boldsymbol{e}}_p = \frac{\partial \boldsymbol{e}_p(\boldsymbol{r}_c^c)}{\partial \boldsymbol{r}_c^c}\dot{\boldsymbol{r}}_c^c,$$

$\boldsymbol{r}_d^c$ being constant. By taking into account (28) and the decomposition

$$\boldsymbol{v}_c^c = \begin{bmatrix} \boldsymbol{\nu}_c^c \\ \boldsymbol{\omega}_c^c \end{bmatrix}$$

with $\boldsymbol{\nu}_c^c = \boldsymbol{R}_c^T\dot{\boldsymbol{o}}_c$, the above expression can be rewritten in the form

$$\dot{\boldsymbol{e}}_p = -\boldsymbol{J}_p\boldsymbol{\nu}_c^c - \boldsymbol{J}_o\boldsymbol{\omega}_c^c, x \tag{78}$$

with

$$\boldsymbol{J}_p = \frac{1}{z_d\rho_z}\begin{bmatrix} -1 & 0 & X \\ 0 & -1 & Y \\ 0 & 0 & -1 \end{bmatrix}$$

and

$$\boldsymbol{J}_o = \begin{bmatrix} XY & -1-X^2 & Y \\ 1+Y^2 & -XY & -X \\ -Y & X & 0 \end{bmatrix}.$$

Equation (78) suggests the following choice of control vector $\boldsymbol{\nu}_r^c$

$$\boldsymbol{\nu}_r^c = \boldsymbol{J}_p^{-1}(\boldsymbol{K}_p\boldsymbol{e}_p - \boldsymbol{J}_o\boldsymbol{\omega}_r^c), \tag{79}$$

$\boldsymbol{J}_p$ being a nonsingular matrix.

Notice that, for the computation of $\boldsymbol{J}_p^{-1}$, knowledge of the constant quantity z_d is required.

If z_d is known, control law (79), in view of assumptions $\dot{\boldsymbol{o}}_c^c \approx \boldsymbol{\nu}_r^c$ and $\boldsymbol{\omega}_c^c \approx \boldsymbol{\omega}_r^c$, yields the following error equation:

$$\dot{\boldsymbol{e}}_p + \boldsymbol{K}_p\boldsymbol{e}_p = \boldsymbol{0},$$

which implies the exponential convergence of $\boldsymbol{e}_p$ to zero, provided that $\boldsymbol{K}_p$ is a positive definite matrix.

If z_d is not known, an estimate $\widehat{z}_d$ can be adopted. Therefore, in control law (79), matrix $\boldsymbol{J}_p^{-1}$ can be replaced by an estimate $\widehat{\boldsymbol{J}}_p^{-1}$. In view of the equality

$$\widehat{\boldsymbol{J}}_p^{-1} = \frac{\widehat{z}_d}{z_d}\boldsymbol{J}_p^{-1},$$

the following error equation is obtained:

$$\dot{\boldsymbol{e}}_p + \frac{\widehat{z}_d}{z_d}\boldsymbol{K}_p\boldsymbol{e}_p = \left(1 - \frac{\widehat{z}_d}{z_d}\right)\boldsymbol{J}_o\boldsymbol{\omega}_r^c.$$

This equation shows that the use of an estimate $\widehat{z}_d$ in place of the true value z_d implies a simple gain scaling in the error equation, and asymptotic stability is preserved. Moreover, due to the presence in the right-hand side of the above equation of a term depending on $\boldsymbol{\omega}_r^c$, the time history of $\boldsymbol{e}_p$ is influenced by the orientation error, which evolves according to (75).

The method illustrated above is only one of the possible visual servoing approaches based on the computation of the planar homography and on its decomposition. It is worth pointing out that knowledge of $(1/d_d)\boldsymbol{o}_{c,d}^c$ and $\boldsymbol{R}_d^c$ allows the computation of the operational space error (51), up to a scaling factor for the positional part; therefore, it is also possible to use the position-based visual servoing schemes presented in Sect. 5. On the other hand, in hybrid visual servoing approaches, different choices are possible for the error components depending on feature parameters as well as for those depending on the operational space variables.

8 Bibliography

The literature dealing with computational vision is quite extensive and various. Image processing is treated, e.g., in [1], while geometrical issues are considered in [2] and [3]. The concept of interaction matrix was originally proposed in [4] under a different name, while the actual name was introduced in [9]. The pose estimation algorithm based on the inverse of the image Jacobian is also termed virtual visual servoing [10]. Position-based visual servoing was proposed in [4] and, more recently, in [5] and [6]. Several papers dealing with image-based visual servoing can be found, starting from early works of [4] and [7]. A rigorous stability analysis is reported in [8] for PD control with gravity compensation and in [9] for resolved-velocity control. Hybrid visual servoing, presented in [11], is only one of advanced control schemes based on decoupling of the camera DOFs, e.g., the partitioned approach [12] and the control based on image moments [13]. Finally, visual servoing based on stereo vision is considered in [14]. An interesting review of the state of the art of visual servoing until mid 1990s is presented in [15], while a more recent review can be found in [16].

References

[1] R.M. Haralick, L.G. Shapiro, *Computer and Robot Vision*, vols. 1 & 2, Addison-Wesley, Reading, MA, 1993.

[2] O. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*, MIT Press, Boston, MA, 1993.

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

[4] L. Weiss, A. Sanderson, C. Neuman, "Dynamic sensor-based control of robots with visual feedback," *IEEE Journal of Robotics and Automation*, vol. 3, pp. 404–417, 1987.

[5] W. Wilson, C. Hulls, G. Bell, "Relative end-effector control using Cartesian position based visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 684–696, 1996.

[6] V. Lippiello, B. Siciliano, L. Villani, "Position-based visual servoing in industrial multirobot cells using a hybrid camera configuration," *IEEE Transactions on Robotics*, vol. 23, pp. 73–86, 2007.

[7] J. Feddema, O. Mitchell, "Vision-guided servoing with feature-based trajectory generation," *IEEE Transactions on Robotics and Automation*, vol. 5, pp. 691–700, 1989.

[8] R. Kelly, R. Carelli, O. Nasisi, B. Kuchen, F. Reyes, "Stable visual servoing of camera-in-hand robotic systems," *IEEE/ASME Transactions on Mechatronics*, vol. 5, pp. 39–48, 2000.

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

[10] A.I. Comport, E. Marchand, M. Pressigout, F. Chaumette, "Real-time markerless tracking for augmented reality: The virtual visual servoing framework," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, pp. 615–628, 2006.

[11] E. Malis, F. Chaumette, S. Boudet, "2-1/2D visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 15, pp. 238–250, 1999.

[12] P. Corke, S. Hutchinson, "A new partitioned approach to image-based visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 17, pp. 507-515, 2001.

[13] F. Chaumette, "Image moments: A general and useful set of features for visual servoing," *IEEE Transactions on Robotics and Automation*, vol. 21, pp. 1116-1127, 2005.

[14] G. Hager, W. Chang, A. Morse, "Robot feedback control based on stereo vision: Towards calibration-free hand-eye coordination," *IEEE Control Systems Magazine*, vol. 15, no. 1, pp. 30–39, 1995.

[15] S. Hutchinson, G. Hager, P. Corke, "A tutorial on visual servo control," *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 651–670, 1996.

[16] F. Chaumette, S. Hutchinson, "Visual servo control. Part I: Basic approaches," *IEEE Robotics and Automation Magazine*, vol. 13, no. 4, pp. 82–90, 2006.