University of bologna

FACULTY OF ENGINEERING - ING-IND/13

Path Planning for Image Based Visual Servoing

by

Duccio Fioravanti

Supervisors:

Prof. Benedetto Allotta

Prof. Paolo Toni

Prof. Carlo Colombo

Coordinator:

Prof. Vincenzo Parenti Castelli

A dissertation presented in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Mechanics of Machines and Mechanisms.

XX° Ph.D. cycle: academic year 2006/2007

to my family

Thanks

I'll be always grateful to Prof. Benedetto Allotta for giving me the opportunity to carry out such exciting research, for his constant help and for the complete trust placed in me during all my Philosophical Doctorate course.

A special thank to Prof Carlo Colombo for his incomparable professionalism, for his friendship and for the profuse help provided in these years of research.

I'd sincerely thanks also Prof. Poalo Toni and all my colleagues of the Mechanic of Machines Section of the Energetics Department "Sergio Stecco" of Florence University: their friendly and cooperative environment, hard to be found elsewhere, has played an essential rule in the outcomes to my activities.

A special acknowledgments also to Prof. Joris De Schutter, Prof. Herman Bruyninckx, and all the Autonomous Compliant Motion Group from the Katholieke Universiteit of Leuven (Belgium): they made possible to realize real experiments in their lab in a very constructive and friendly environment.

Thank also to Prof. Vincenzo Parenti Castelli for the continuous attention placed in following my Ph.D. activities.

Above all my greatest thanks are addressed to my family and Elena: without their unique support I'd have never been able to reach these results.

Duccio

Florence - March 2008

Contents

Chapter 1

Introduction

This work presents the greater part of the research I have carried out in Ph.D. course concerning Visual Servoing and its strictly connected disciplines like projective geometry, image processing, robotics and non-linear control. From this point of view Visual Servoing represents a many-sided research field where different subjects meet together trying to create more "intelligent" systems. Visual Servoing, on the other side, is one of the branch of the wider environment known in literature as Sensor Based Control: in this discipline the measurements provided by "sensors" make up all the necessary informations needed to drive the controlled dynamical systems to the desired configuration.
In Visual Servoing, more specifically, digital cameras play the main rule: that is why nowadays, thank to the outstanding improvements of camera hardware performances (both in terms of computational time and of image resolution), vision based control has become a challenging research topic for the international scientific community.
This thesis addresses the problem to control a robotic manipulator through one of the largely used Visual Servoing techniques: the Image Based Visual Servoing (IBVS). In Image Based Visual Servoing the robot is driven by on-line performing a feedback control loop that is closed directly in the $2D$ space of the camera sensor. The work considers the case of a monocular system with the only camera mounted on the robot end effector (eye in hand configuration). Through IBVS the system can be positioned with respect to a $3D$ fixed target by minimizing the differences between its initial view and its goal view, corresponding respectively to the initial and the goal system configurations: the robot Cartesian Motion is thus generated only by means of visual informations.
IBVS has received great attention from researchers because it ensures stability and convergence even in the presence of modelling [ECR92] and calibration errors [Esp93]. The convergence properties can be improved by using adaptive schemes devoted to estimate on line some of the extrinsic parameters (usually depths) appearing in the image control law [CA01, DFD03].
However, the execution of a positioning control task by IBVS is not straightforward because singularity problems may occur and local minima may be reached where the reached image is very close to the target one but the 3D positioning task is far from being fulfilled [Cha99]: this happens in particular for large camera displacements, when the the initial and the goal target views are noticeably different.
To overcame singularity and local minima drawbacks, maintaining the good properties of IBVS robustness with respect to modeling and camera calibration errors, an opportune image path planning can be exploited [MC02]. This work deals with the problem of gener-

ating opportune image plane trajectories for tracked points of the servoing control scheme (a trajectory is made of a path plus a time law). The generated image plane paths must be *feasible* i.e. they must be compliant with rigid body motion of the camera with respect to the object so as to avoid image jacobian singularities and local minima problems. In addition, the image planned trajectories must generate camera velocity screws which are smooth and within the allowed bounds of the robot. We will show that a scaled 3D motion planning algorithm can be devised in order to generate feasible image plane trajectories. Since the paths in the image are off-line generated it is also possible to tune the planning parameters so as to maintain the target inside the camera field of view even if, in some unfortunate cases, the feature target points would leave the camera images due to $3D$ robot motions. The thesis is structured as follow:

- Chapter II describes all the fundamental notions concerning projective geometry, camera models, and two views geometry. First the basic primitives of $\mathbb{P}^2$ and $\mathbb{P}^3$ projective spaces are presented as well as the projective transformations; then the digital image is introduced and the most important geometric models for digital cameras are addressed.

- Chapter III presents some useful relation concerning two-view Geometry: the plan-parallax equation is presented together with the Planar Pixel Homography and the corresponding Euclidean Homography between two views. Finally the Structure from Motion problem from two calibrated views is solved both for planar and for general $3D$ targets. The results of this chapter will be widely exploited in the control planning strategy.

- In Chapter IV Visual Servoing control is addressed, focusing on Image Based Visual Servoing approach with points as visual features. The chapter explains in detail also the image path planning strategy that has been developed so as to increase the control performances also in presence of large camera displacements and to avoid singularity and local minima problems.

- Chapter V presents an extension of the presented planning and control schemes to $3D$ non planar targets modeled through a pair of coaxial circles plus one point, like for instance axial symmetric objects. Moreover in this case full calibration data (fixed internal parameters) are obtained from two views, resulting in a self-calibrated approach. Calibration results are then used to recover Euclidean target structure and camera relative pose.

- Chapter VI reports several results concerning both simulation analysis and real robot experiments. To first test the validity of the proposed approach some simulations have been performed: for the axial symmetric target case, to check for the influence of noise in the path planning approach, a dedicated analysis has also been realized . Several experiments have been realized with a $6DOF$ anthropomorphic manipulator with a fire-wire camera installed on its end effector: the results demonstrate the good performances and the feasibility of the proposed approach.

- Finally in Chapter VII conclusions and future key points are discussed.

Chapter 2

Projective Geometry

Before to address the control issues typical of Image Based Visual Servoing, it is essential to master some basics notions of Projective Geometry; this subject indeed, among various items, investigates the relations between 3D entities and its image projections, building up a formalism to model images, camera sensor and 3D world both from an algebraic and a geometrical point of view. In this sense Projective Geometry represents a precious background to develop control schemes based on image informations. This chapter introduces some topics that will be useful to understand the following part of the work, focusing the attention on image-primitives, camera models and two-view geometry.

2.1 The 2D projective plane

The basic ideas of planar geometry are familiar to anyone who have studied mathematics even at elementary level. In fact, they are so much a part of our everyday experience that we take them for granted: we'll show in the sequel that what seem to be granted probably could be not. Le be a point in the plane to be represented by the couple of coordinates x, y embedded in the column vector $\boldsymbol{x} = (x, y)^T \in \mathbb{R}^{2\times 1}$: thus is common to identify the plane with $\mathbb{R}^2$. In this section we introduce the *homogeneous* notations for points, lines and conics as primitives of the plane.

2.1.1 Homogeneous representation of lines

A line in the plane can be represented by the homogeneous equation:

$$ax + by + c = 0 \tag{2.1}$$

where different choices for the a, b and c coefficients give rise to different lines. A line can thus be naturally represented by the non-minimal parameterization of the vector $(a, b, c)^T \in \mathbb{R}^{3\times 1}$. The correspondence between lines and vectors is not one to one since the lines $ax + by + c = 0$ and $(ka)x + (kb)y + (kc) = 0$ are the same, for any non zero constant k. For this reason, in Projective Geometry, two vectors that differ just for a scaling factor k are considered equivalent. From the previous statement is clear that any vector $(a, b, c)^T$ is a representative of an equivalence class.
The set of equivalence classes of vector $\in \mathbb{R}^{3\times 1} - (0, 0, 0)^T$ form the *projective space* $\mathbb{P}^{2\times 1}$. As previous pointed the vector $(0, 0, 0)^T$, which not correspond to any line, is excluded.

2.1.2 Homogeneous representation of points

A point $\boldsymbol{x} = (x, y)^T$ lies on the line $\boldsymbol{l} = (a, b, c)^T$ if and only if $ax + bx + c = 0$. This can be also rewritten using the inner product as:

$$(x, y, 1)(a, b, c)^T \;=\; (x, y, 1)\boldsymbol{l} \;=\; 0 \quad ; \tag{2.2}$$

by this way the point $(x, y)^T \in \mathbb{R}^{3\times1}$ is represented by a vector $\in \mathbb{R}^{3\times1}$ by adding the final coordinate of 1. Note that for any non-zero constant k and line $\boldsymbol{l}$ we have $(kx, ky, k)\boldsymbol{l} = 0$ if and only if $(x, y, 1)\boldsymbol{l} = 0$. The set of vector (kx, ky, k) for varying values of k can thus be considered as the representation of the point $(x, y)^T \in \mathbb{R}^{3\times1}$. As with the lines also points are represented by homogeneous vectors. An arbitrary *homogeneous* vector representative of a point is of the form $\boldsymbol{x} = (x_1, x_2, x_3)^T$, representing the point $(x_1/x_3,\, x_2/x_3)^T \in \mathbb{R}^{2\times1}$ in *inhomogeneous* coordinates. Points, then, as homogeneous vector are also elements of $\mathbb{P}^{2\times1}$. To determine if a point belong to a line one has the following simple result:

Theorem 2.1.2.1. *The point $\boldsymbol{x}$ lies on the line $\boldsymbol{l}$ if and only if $\boldsymbol{x}^T\boldsymbol{l} = 0$.*

Note that 2 independent parameters are sufficient to represent both points and lines, respectively the x, y and the ratios a/b, b/c: they thus have 2 degrees of freedom.
Given the lines $\boldsymbol{l} = (a, b, c)^T$ and $\boldsymbol{l}' = (a', b', c')^T$ define the vector $\boldsymbol{x} = \boldsymbol{l} \wedge \boldsymbol{l}'$, where $\wedge$ represents the vector or cross product. From the identity $\boldsymbol{l}^T(\boldsymbol{l} \wedge \boldsymbol{l}') = \boldsymbol{l}'^T(\boldsymbol{l} \wedge \boldsymbol{l}') = 0$ is clear that $\boldsymbol{l}^T\boldsymbol{x} = \boldsymbol{l}'^T\boldsymbol{x} = 0$. Thus if $\boldsymbol{x}$ is the representation of a point, it lies both on lines $\boldsymbol{l}$ and $\boldsymbol{l}'$; this shows:

Theorem 2.1.2.2. *The intersection of two lines $\boldsymbol{l}$ and $\boldsymbol{l}'$ is the point $\boldsymbol{x} = (\boldsymbol{l} \wedge \boldsymbol{l}')$.*

With an analogous demonstration one can also verify that:

Theorem 2.1.2.3. *The line through two points $\boldsymbol{x}$ and $\boldsymbol{x}'$ is $\boldsymbol{l} = (\boldsymbol{x} \wedge \boldsymbol{x}')$.*

2.1.3 Conics

A conic is a curve described by a second-degree equation in the plane; in Euclidean geometry conics are of three main types: hyperbola, ellipse an parabola (apart from the so-called degenerate conics). Conics can be obtained by the intersection between a cone and a plane in different orientation (degenerate conics arise from plane containing the cone vertex). The inhomogeneous representation of a conic is:

$$ax^2 + bxy + cy^2 + dx + ey + f = 0 \quad . \tag{2.3}$$

By replacing $x \mapsto x_1/x_3$ and $y \mapsto x_2/x_3$ one find :

$$ax_1^2 + bx_1x_2 + cx_2^2 + dx_1x_3 + ex_2x_3 + fx_3^2 = 0 \tag{2.4}$$

or in matrix form

$$\boldsymbol{x}^T\boldsymbol{C}\boldsymbol{x} \;=\; 0 \quad , \tag{2.5}$$

where $\boldsymbol{C}$ is the following symmetric matrix:

$$\boldsymbol{C} = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix} \quad . \tag{2.6}$$

As for homogeneous representation of points and lines, only the ratios of the coefficients matrix elements are important, since multiplying $\boldsymbol{C}$ by a non zero scalar does not affect the above equation: thus $\boldsymbol{C}$ is the homogeneous representation of a conic. $\boldsymbol{C}$ has only 5 degrees of freedom that can be thought as its 6 coefficients less one for scale.

Theorem 2.1.3.1. *The line $\boldsymbol{l}$ tangent to $\boldsymbol{C}$ at a point $\boldsymbol{x}$ on $\boldsymbol{C}$ is given by $\boldsymbol{l} = \boldsymbol{C}\boldsymbol{x}$.*

The line defined by $\boldsymbol{l} = \boldsymbol{C}\boldsymbol{x}$ passes through $\boldsymbol{x}$, since $\boldsymbol{l}^T\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{C}\boldsymbol{x} = 0$. If $\boldsymbol{l}$ has one-point contact with the conic, then it is tangent and the above is proved; otherwise suppose that $\boldsymbol{l}$ meets the conic in another point $\boldsymbol{y}$. Then $\boldsymbol{y}^T\boldsymbol{C}\boldsymbol{y} = 0$ and $\boldsymbol{x}^T\boldsymbol{C}\boldsymbol{y} = \boldsymbol{l}^T\boldsymbol{y} = 0$. From this it follow that $(\boldsymbol{x} + \alpha\boldsymbol{y})^T\boldsymbol{C}(\boldsymbol{x} + \alpha\boldsymbol{y}) = 0$ for all α, which means that the whole line $\boldsymbol{l} = \boldsymbol{C}\boldsymbol{x}$ joining $\boldsymbol{x}$ and $\boldsymbol{y}$ lies on the conic $\boldsymbol{C}$, which is therefore degenerate.
A point $\boldsymbol{x}$ and a conic $\boldsymbol{C}$ define the line $\boldsymbol{l} = \boldsymbol{C}\boldsymbol{x}$: the line $\boldsymbol{l}$ is called *polar* of $\boldsymbol{x}$ with respect to $\boldsymbol{C}$, and the point $\boldsymbol{x}$ is the *pole* of $\boldsymbol{l}$ with respect to $\boldsymbol{C}$.

Theorem 2.1.3.2. *The polar line $\boldsymbol{l} = \boldsymbol{C}\boldsymbol{x}$ of the point $\boldsymbol{x}$ with respect to a conic $\boldsymbol{C}$ intersects the conic in two points. The two lines tangent to $\boldsymbol{C}$ at these points intersect at $\boldsymbol{x}$.*

To proof this theorem consider a point $\boldsymbol{y}$ on $\boldsymbol{C}$. The tangent line at $\boldsymbol{y}$ is $\boldsymbol{C}\boldsymbol{y}$, and this line contains $\boldsymbol{x}$ is $\boldsymbol{x}^T\boldsymbol{C}\boldsymbol{y} = 0$. Using the symmetry of $\boldsymbol{C}$, the condition $\boldsymbol{x}^T\boldsymbol{C}\boldsymbol{y} = (\boldsymbol{C}\boldsymbol{x})^T\boldsymbol{y} = 0$ is that the point $\boldsymbol{y}$ lies on the line $\boldsymbol{C}\boldsymbol{x}$. Thus the polar line $\boldsymbol{C}\boldsymbol{x}$ intersects the conic in the point $\boldsymbol{y}$ at which the tangent line contains $\boldsymbol{x}$. As the point $\boldsymbol{x}$ approaches the conic the tangent line become closer to collinear, and the contact point on the conic also become closer. In the limit that $\boldsymbol{x}$ lies on $\boldsymbol{C}$ we have:

Theorem 2.1.3.3. *If the point $\boldsymbol{x}$ is on $\boldsymbol{C}$ then the polar is the tangent line to the conic at $\boldsymbol{x}$.*

2.1.4 Ideal points and line at infinite

Homogeneous vector $\boldsymbol{x} = (x_1, x_2, x_3)^T$ with $x_3 \neq 0$ corresponds to finite points in the Cartesian plane represented by $\mathbb{R}^{2\times1}$. One may augment $\mathbb{R}^{2\times1}$ by adding points with last coordinate $x_3 = 0$. The resulting space is the set of all homogeneous vector, the projective plane $\mathbb{P}^{2\times1}$. The points with last coordinate $x_3 = 0$ are known as *ideal points* or *points at infinite*: all these points can be written as $(x_1, x_2, 0)^T$, with a particular point specified by the ratio $x_1 : x_2$. Note that this set lies on a single line, the *line at infinite*, defined by the vector $\boldsymbol{l}_\infty = (0, 0, 1)^T$: indeed one can verify $(0, 0, 1)(x_1, x_2, 0)^T = 0$. The line $\boldsymbol{l} = (a, b, c)^T$ intersects $\boldsymbol{l}_\infty$ in the ideal point $(b, -a, 0)^T$, as one can verify by using theorem (2.1.2.2). A line $\boldsymbol{l}' = (a, b, c')^T$ parallel to $\boldsymbol{l}$ intersects $\boldsymbol{l}_\infty$ in the same ideal point $(b, -a, 0)^T$ for any value c': this observation agrees with the usual idea that parallel line meet at infinite.
In inhomogeneous notation $(b, -a)^T$ is a vector tangent to the line and orthogonal to the line normal (a, b), thus representing the line directions. As the line direction varies, the ideal point $(b, -a, 0)^T$ varies over $\boldsymbol{l}_\infty$: for this reason $\boldsymbol{l}_\infty$ can be thought of as the set of

directions of lines in the plane.
The introduction of the points at infinite is useful to simplify the intersection properties of points and lines. In the projective space $\mathbb{P}^{2\times1}$ one may state without qualification that two distinct line meet in a single point and two distinct points define a single line. This is not true in the Euclidean geometry of $\mathbb{R}^{2\times1}$ in which parallel line form a special case.

2.1.4.1 The circular Points

The *circular points* also called *absolute points* are a pair of complex conjugate points on $\boldsymbol{l}_\infty$ denoted as $\boldsymbol{I}$, $\boldsymbol{J}$, with the following canonical coordinates

$$\boldsymbol{I} = \begin{pmatrix} 1 \\ i \\ 0 \end{pmatrix} \qquad \boldsymbol{J} = \begin{pmatrix} 1 \\ -i \\ 0 \end{pmatrix} \quad . \tag{2.7}$$

The name "circular points" arises because every circle intersects $\boldsymbol{l}_\infty$ at the circular points. To verify it consider the homogeneous representation of a circle, derived from equation (2.4) setting $a = c$ and $b = 0$:

$$x_1^2 + x_2^2 + dx_1x_3 + ex_2x_3 + fx_3^2 = 0 \quad ,$$

where a has been set to unity. This conic intersects $\boldsymbol{l}_\infty$ in the ideal points for which $x_3 = 0$, namely

$$x_1^2 + x_2^2 = 0$$

with solution $\boldsymbol{I} = (1, i, 0)^T$ and $\boldsymbol{J} = (1, -i, 0)^T$: thus any circle intersects $\boldsymbol{l}_\infty$the circular points.

2.1.5 A model for the projective plane

The projective plane $\mathbb{P}^{2\times1}$ can be though as the set of rays in $\mathbb{R}^{3\times1}$ as shown in figure (2.1). The set of all vectors $k(x_1, x_2, x_3)^T$ as k varies forms a ray through the origin: this ray may be thought of as representing as a single point of $\mathbb{P}^{2\times1}$. The I In this model, the

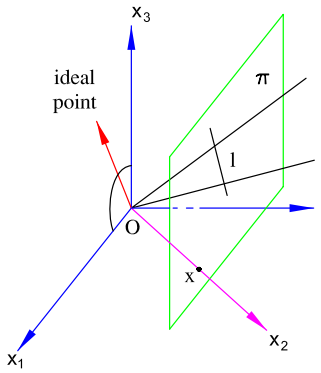


Figure 2.1: A model fro the projective plane $\mathbb{P}^{2\times1}$.

lines of $\mathbb{P}^{2\times1}$ are planes passing through the origin. One verifies that two non-identical

rays lies on exactly one plane, and two planes intersect in one ray in analogy respectively with points and lines of $\mathbb{P}^{2\times1}$. Points and lines may be obtained by intersecting this set of rays and planes with the plane $x_3 = 1$.The rays representing ideal points are parallel to the plane $x_3 = 1$ and form the plane representing $\boldsymbol{l}_\infty$.

2.1.6 Projective Transformations

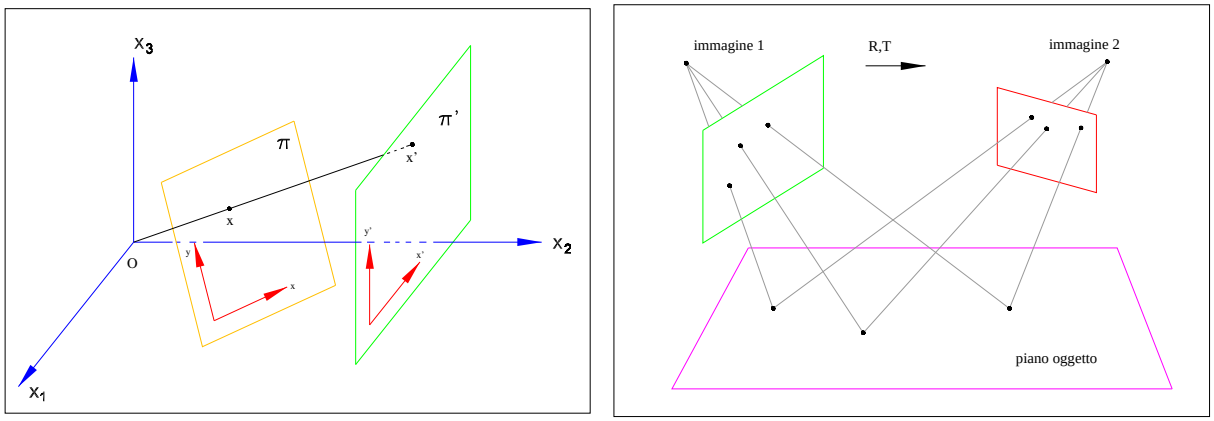
A transformation of the projective plane is named *projectivity* and represents a mapping from points in $\mathbb{P}^{2\times1}$ to points in $\mathbb{P}^{2\times1}$ that maps lines to lines (see figure 2.2).

Definition 1. *A **projectivity** is an invertible mapping h from $\mathbb{P}^{2\times1}$ to itself such that three points $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ lie on the same line if and only if $g(\boldsymbol{x}_1)$, $g(\boldsymbol{x}_2)$ and $g(\boldsymbol{x}_3)$ do.*

Projectivity form a group since both the inverse and the composition of projectivities are projectivities. A projectivity can be also named as *collineation* or *homography* (the terms are synonymous). In the previous definition a projectivity is defined in terms of a coordinate-free geometry.An equivalent algebraic definitions of projectivity is possible thanks to the following results:

Theorem 2.1.6.1. *A mapping $g : \mathbb{P}^{2\times1} \rightarrow \mathbb{P}^{2\times1}$ is a projectivity if and only if there exists a non-singiular 3×3 matrix $\boldsymbol{G}$ such that for any point in $\mathbb{P}^{2\times1}$, represented by a vector $\boldsymbol{x}$ it is true that $h(\boldsymbol{x}) = \boldsymbol{G}\boldsymbol{x}$.*

To prove this theorem let be $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$ lie on a line $\boldsymbol{l}$. Thus $\boldsymbol{l}^T\boldsymbol{x}_i = 0$ for $i = 1, \dots, 3$. Let $\boldsymbol{G}$ be a non singular matrix $\in \mathbb{R}^{3\times3}$. It is easy to verify that $\boldsymbol{l}^T\boldsymbol{G}^{-1}\boldsymbol{G}\boldsymbol{x}_i = 0$. Thus the points $\boldsymbol{G}\boldsymbol{x}_i$ all lies on the line $\boldsymbol{G}^{-T}\boldsymbol{l}$, and collinearity is preserved by the transformation. The converse is much harder to prove, namely that each projectivity arise in this way. A projectivity can thus be considered as a linear mapping of homogeneous coordi-



a) b)

Figure 2.2: a) The two images generated by the rays of a central projection model are linked by a projectivity. b) The images generated by the combination of two central projection models via a plane are linked by a projectivity.

nates. Thanks to the theorem (2.1.6.1) it is possible to define a projective transformation as follows.

Definition 2. *A planar* ***projective transformation*** *is a linear transformation on homogeneous* 3*-vectors represented by a non singular* 3×3 *matrix:*

$$\begin{pmatrix} x_1^{'} \\ x_2^{'} \\ x_3^{'} \end{pmatrix} = \begin{bmatrix} g_{11} & g_{12} & g_{13} \\ g_{21} & g_{22} & g_{23} \\ g_{31} & g_{32} & g_{33} \end{bmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \quad , \tag{2.8}$$

that is $\boldsymbol{x}^{'} = \boldsymbol{G}\boldsymbol{x}$*.*

The matrix $\boldsymbol{G}$ defined in the previous equation may be changed by a multiplication by an arbitrary non-zero scalar factor without altering the projective transformation: $\boldsymbol{G}$ is thus an *homogeneous* matrix since, as for the representation of point and line in $\mathbb{P}^{2\times1}$, only the ratio of the matrix elements are significant. There are eight independent ratio among the nine elements of $\boldsymbol{G}$, and it follows that a projectivity has *eight degrees of freedom.*

2.1.6.1 Transformation of lines

It has been shown in the proof of theorem (2.1.6.1) that if point $\boldsymbol{x}_i$ lies on a line $\boldsymbol{l}$, the transformed point $\boldsymbol{x}_i^{'} = \boldsymbol{G}\boldsymbol{x}_i$ under an homography lies on the line $\boldsymbol{l}^{'} = \boldsymbol{G}^{-T}\boldsymbol{l}$. In this way incidence of points on lines is preserved, since $\boldsymbol{l}^{'T}\boldsymbol{x}_i^{'} = \boldsymbol{l}^T\boldsymbol{G}^{-1}\boldsymbol{G}\boldsymbol{x}_i = 0$. This gives the transformation rule for lines:

Theorem 2.1.6.2. *Under a point transformation* $\boldsymbol{x}^{'} = \boldsymbol{G}\boldsymbol{x}$*, a line transforms as* $\boldsymbol{l}^{'} = \boldsymbol{G}^{-T}\boldsymbol{l}$*.*

One may alternatively write $\boldsymbol{l}^{'T} = \boldsymbol{l}^T\boldsymbol{G}^{-1}$. Note the fundamentally different way in which points and lines transform. Points transforms according to $\boldsymbol{G}$ (namely *contravariantly*) while line according to $\boldsymbol{G}^{-1}$ (namely *covariantly*).

2.1.6.2 Transformation of conics

Under a point transformation $\boldsymbol{x}^{'} = \boldsymbol{G}\boldsymbol{x}$ the conic equation (2.5) becomes

$$\begin{aligned} \boldsymbol{x}^T\boldsymbol{C}\boldsymbol{x} &= \boldsymbol{x}^{'T}[\boldsymbol{G}^{-1}]^T\boldsymbol{C}\boldsymbol{G}^{-1}\boldsymbol{x}^{'} \\ &= \boldsymbol{x}^{'T}\boldsymbol{G}^{-T}\boldsymbol{C}\boldsymbol{G}^{-1}\boldsymbol{x}^{'} \quad , \end{aligned}$$

which is a quadratic form $\boldsymbol{x}^{'T}\boldsymbol{C}^{'}\boldsymbol{x}^{'}$ with $\boldsymbol{C}^{'} = \boldsymbol{G}^{-T}\boldsymbol{C}\boldsymbol{G}^{-1}$. This gives the transformation rule for a conic:

Theorem 2.1.6.3. *Under a point transformation* $\boldsymbol{x}^{'} = \boldsymbol{G}\boldsymbol{x}$*, a conic* $\boldsymbol{C}$ *transforms to* $\boldsymbol{C}^{'} = \boldsymbol{G}^{-T}\boldsymbol{C}\boldsymbol{G}^{-1}$*.*

The presence of $\boldsymbol{G}^{-1}$ in the theorem may be expressed by saying that a conic transforms *covariantly.*

2.2 The 3D projective space

Let now outline some fundamental properties and entities of the projective 3-space, named $\mathbb{P}^{3\times1}$. Most of these are a straightforward generalizations of those of the projective plane described in the previous section and will be useful for the development of the work.

2.2.1 Points

A point in 3-space is represented in homogeneous coordinate as a 4-vector. More precisely the homogeneous vector $\boldsymbol{x} = (x_1, x_2, x_3, x_4)^T$ with $x_4 \neq 0$ represents the point $(x, y, z)^T$ of $\mathbb{R}^{3\times1}$ with inhomogeneous coordinates

$$x = \frac{x_1}{x_4}\,,\ y = \frac{x_2}{x_4}\,,\ z = \frac{x_3}{x_4}\,, \quad .$$

Homogeneous points with $x_4 = 0$ represents points at infinity.

2.2.2 Planes

A plane in 3-space may be written as

$$\pi_1 x + \pi_2 y + \pi_3 z + \pi_4 = 0 \quad . \tag{2.9}$$

Clearly this equation is unaffected by multiplication by a non-zero scalar, so only the three independent ratios $\{\pi_1 : \pi_2 : \pi_3 : \pi_4\}$ of the plane coefficients are significant. It follows that a plane has 3 degrees of freedom in 3-space. The homogeneous representation of a plane is the 4-vector $\boldsymbol{\pi} = (\pi_1, \pi_2, \pi_3, \pi_4)^T$. The first 3 components of $\boldsymbol{\pi}$ correspond to the plane normal $\boldsymbol{n}$ wile $(\pi_4/||\boldsymbol{n}||)$ is the signed distance of the plane from the origin.
Rearranging equation (2.9) in homogeneous form by substituting $x \mapsto x_1/x_4$, $y \mapsto x_2/x_4$ and $z \mapsto x_3/x_4$, we have the following expression:

$$\pi_1 x_1 + \pi_2 x_2 + \pi_3 x_3 + \pi_4 x_4 = 0$$

ore more briefly

$$\boldsymbol{\pi}^T \boldsymbol{x} = 0 \tag{2.10}$$

which express that the point $\boldsymbol{x}$ belong to the plane $\boldsymbol{\pi}$.

2.2.3 The Plane at infinite

In the geometry of projective 3-space $\mathbb{P}^{3\times1}$ the corresponding entity of the line at infinite $\boldsymbol{l}_\infty$, presented in subsection (2.1.4), is the *plane at infinite* $\boldsymbol{\pi}_\infty$.
The plane at infinite has its canonical position expressed by $\boldsymbol{\pi}_\infty = (0, 0, 0, 1)^T$ in the affine 3-space. $\boldsymbol{\pi}_\infty$ contains all the points at infinite of $\mathbb{P}^{3\times1}$ (all the directions) of coordinates $\boldsymbol{d}_\infty = (x_1, x_2, x_3, 0)^T$. $\boldsymbol{\pi}_\infty$ enables the identification of affine properties such as parallelism from a 3D perspective reconstruction. We have indeed that:

- Two planes are parallel if, and only if, their line of intersection is on $\boldsymbol{\pi}_\infty$.
- A line is parallel to another line, or to a plane, if their point of intersection is on $\boldsymbol{\pi}_\infty$.

We then have that in $\mathbb{P}^{3\times1}$ any pair of planes intersect in a line, with parallel planes intersecting in a line belonging to the plane at infinite.

2.2.4 The absolute conic

The corresponding entities of circular points $\boldsymbol{I}$ and $\boldsymbol{I}$, defined in subsection (2.1.4), in the projective geometry of $\mathbb{P}^{3\times1}$ is the *absolute conic* $\boldsymbol{\Omega}_\infty$. The absolute conic is a degenerate point conic on plane at infinite $\boldsymbol{\pi}_\infty$ defined by the following equations:

$$\begin{cases} x_1^2 + x_2^2 + x_3^2 = 0 \\ x_4 = 0 \end{cases} . \tag{2.11}$$

Note that two equation are necessary to define $\boldsymbol{\Omega}_\infty$.
For directions on $\boldsymbol{\pi}_\infty$ (that are points with $x_4 = 0$) the defining equation can be written

$$(x_1, x_2, x_3)\boldsymbol{I}(x_1, x_2, x_3)^T = 0$$

so that $\boldsymbol{\Omega}_\infty$ correspond to a conic $\boldsymbol{C}$ with matrix $\boldsymbol{C} = \boldsymbol{I}$. $\boldsymbol{\Omega}_\infty$ is thus a conic of purely imaginary points on $\boldsymbol{\pi}_\infty$. Even though $\boldsymbol{\Omega}_\infty$ does not have any real points, it shares properties of any conic such as that a line intersects a conic in two points; the pole-polar relationship and etc. Here are few particular properties of $\boldsymbol{\Omega}_\infty$:

- All circles intersect $\boldsymbol{\Omega}_\infty$ in two points. Suppose that the support plane of the circle is $\boldsymbol{\pi}$. Then $\boldsymbol{\pi}$. intersect $\boldsymbol{\pi}_\infty$ in a line, and this line intersects $\boldsymbol{\Omega}_\infty$ into two points. These two points are the circular points of $\boldsymbol{\pi}$.
- All spheres intersects $\boldsymbol{\pi}_\infty$ in $\boldsymbol{\Omega}_\infty$.

2.3 Image and Camera Devices

This section presents the image concept and the basic working principle of the most commonly used digital camera sensor: these themes are of great importance dealing with vision based control.

2.3.1 Image and Picture

A practical definition of image can be the following:

Definition 3. *An Image is a measured physical quantity as function of position.*

Radiation intensity (X-ray, UV, visible and IR), acoustical wave and surface depth measurements as function or the position are common image examples. Among different image types *Two-dimensional Images* I_2 represent a measured physical quantity (i.e. the light brightness) in a compact region Ω_2 of a two dimensional space while *Three-dimensional Images* I_3 (i.e. the computerized tomography or the magnetic resonance imaging) map a compact portion Ω_3 of a three dimensional space in the measurement space. From a closer analytical point of view, two and three dimensional images can be respectively defined as follows:

$$\begin{aligned} I_2 &: \Omega_2 \subset \mathbb{R}^2 \to \mathbb{R}_+; \qquad (x, y) \mapsto I_2(x, y) \\ I_3 &: \Omega_3 \subset \mathbb{R}^3 \to \mathbb{R}_+; \qquad (x, y, z) \mapsto I_3(x, y, z) \quad , \end{aligned} \tag{2.12}$$

where we have considered the image measurement range $\mathbb{R}_+$ mono-dimensional. For instance, in case of a camera, Ω_2 is planar, rectangular region occupied by the photographic medium or by the CCD sensor. In case of a digital image, both the domain ω_2 and the range $\mathbb{R}_+$ are discretized. For instance, $\Omega_2 = [1, 640] \times [1, 480] \subset \mathbb{Z}^2$ and $\mathbb{R}_+$ is approximated by an interval of integers $[0, 255] \subset \mathbb{Z}_+$. The graphical representation of the analytical function I, representing the image, do not seem very indicative of the properties of the scene it embeds. A different representation of the same image that is better suited for interpretation by the human visual system is obtained by generating a *picture* (i.e. by plotting a photos from image I).

Definition 4. *A picture can be thought of as a scene different from the true one that produces on the imaging sensor (the eye in this case) the same image as the true one.*

In this sense pictures are "controlled illusions": they are scenes different from the true one (they are flat) that produce in the eye the same image as the original scenes (but they contain exactly the same informations of the original images).
In order to describe image formation process through a two-dimensional camera sensor, we must specify the value of $I_2(x, y)$ at each point (x, y) in Ω_2. Such a value $I_2(x, y)$ is typically called *image intensity* or *brightness*, or more formally *irradiance*. It has the units of power per unit area (W/m^2) and describes the energy falling onto a small patch of the imaging sensor. The irradiance at a point of coordinates (x, y) is obtained by integrating power both in time (i.e., the shutter interval in a camera or the integration time in a CCD array) and in a region of space. The region of space that contributes to the irradiance at (x, y) depends upon the shape of the object (surface) of interest, the optics of the imaging device, and it is by no means trivial to determine.

2.3.2 Digital Image

As mentioned above, to treat an image with a computer we need to convert the continuous image I_2 of equation (2.12) in its digital representation: digital cameras perform this operation leading to the formation of the digital image:

Definition 5. *A digital image, whatever its type, is represented by a matrix of numbers.*

The exact correspondence between the digital image and physical $3D$ world is determined by the acquisition process that depends on the adopted camera sensor: all the informations that can be retrieved from a digital image (such as shapes, distances and objects identifications) have to be extracted from the matrix of number encoding them. Each entry of the matrix representing the digital image is called image *pixel*. Larger the size of the number matrix, bigger will be the *image resolution* (corresponding to the total pixel number) and the memory requested to store the digital image in a computer.
Digital images can be subdivided in two main groups: *mono-channel images* and *multi-channel images*. Among the mono-channel images the most used types in computer vision are:

- the *Binary image*, also known as *bilevel image*, where each pixel can assume only two discrete values, 0 and 1, corresponding respectively to black and white color levels, thus requiring a very limited memory space to be stored;
- the *Grayscale image*, also known as *intensity image*, where each pixel can assume an integer value ranging from 1 to 256 (or from 0 to 255 adopting the zero convention)

representing as the name suggests a scale of grays from black (1) to white (256): in such image each pixel occupies $8 - bits$ of memory (generally speaking it is called indeed 8-bit images);

- the *Indexed images* where each pixel assumes an integer value that is codified in a color by a predefined *colormap* (the pixel value is a pointer to a precise element of the colormap).

In multi-channel images instead each pixel is associated with more than one value: such images are thus represented by a multidimensional matrix where the first two dimension (row and column indexes) represent the pixel position and the other dimensions (remaining indexes) represents the image channels. Among the multichannel images we recall:

- the *Truecolor image*, also known as *RGB image*, that are three channels images (thus coded by a matrix $\in \mathbb{Z}^{m \times n \times 3}$) where each pixel is associated with three integers numbers all ranging from 1 to 256 respectively corresponding to the three primary color level of the red green and blue (also different colorimetric spaces can be used like HSV lett. hue-saturation-value);

- the *Multispectral Images* where each pixel is associated with more than three channel (i.e. R,G,B channel plus infra-red IR and ultraviolet UV channels) largely used in photogrammetry and in satellite images (where also 16 channels per pixel can be reached!.

Usually a digital image file, further on embedding the pixel matrix, contains also some general data (such as the capture time and date of the image) called *metadata*.
In this thesis we'll be interested more in grayscale and truecolor images that are the standard output of digital camera sensor adopted in visual servoing systems.

2.3.3 CCD and CMOS cameras

A Digital camera camera is a sensor mapping the continuous $3D$ space in a time and space sampled $2D$ image. This sensor is a complex system embedding various devices; from a simplified point of view it can be thought as formed by three main components:

- a *lens* (camera optical system) used to "direct" light (that is to control the change in the direction of light propagation). This can be performed by means of diffraction, refraction an reflection. By the optical system indeed light rays coming from $3D$ space are directed to converge in a *focus*, the *optical center*;

- a *sensor* placed near the camera optical center measuring the light intensity on its surface and storing it in some memory registers; to measure the light reflected from the $3D$ scene, the sensor is formed by matrix of small photosensitive elements, the *sensor pixels*, able to transform the light radiation power in electrical energy;

- a electronic device, called *frame grabber*, used to build the digital images (sensor *video stream*) and to send it with the desired time step (the camera *frame rate*) to the computer.

Today several types of sensors are available: they differs in the physical principle used to transform light radiation power.The most commonly used are the CCD and CMOS sensors: both two exploit the photoelectric effect of semiconductors. Now it is clear that the CCD and CMOS sensor perform a spatial image sampling while the frame grabber behaves as time sampler.

2.3.3.1 CCD

A CCD (Charge Coupled Device) sensor is composed by a rectangular pixel matrix. The total pixel number determine the *image resolution*. By means of the photoelectric effect, when the light photon hits the semiconductor, some free electrons are created so that each pixel stores an electric charge depending on the time integral of the light intensity incident in the pixel itself. Such electric charge is passed by a shift device (like an analog shift register) to an output amplifier while, at the same time, the pixel runs down. The output signal is further on processed to generate the final video stream. CCD sensor are the commonly used both in professional camera and in low cost webcam.
A drawback of CCD sensor is the *overflow effect*: this effect arises when the electrons from a saturated pixel overflow on the neighbouring pixels thus falsifying their color levels: this effect provokes a blurring in the overflow zone of the image. The key parameters to describe a CCD sensor are the following:

- *size* with an order of magnitude of about $1cm$;
- *resolution* usually of (320×240) or (640×480) pixels for the cheaper cameras; for expensive digital cameras higher resolution are reached (i.e. 2048×1536);
- *pixel size* with an order of magnitude of about $1\mu m$
- *pixel capacity* having an order of magnitude of about 700 electrons for a μm^2;
- *filling factor*. It is the ratio between the active area of the sensor (occupied by pixels) and the overall area;
- *absolute sensitivity* defined as the smallest detectable light intensity level higher than the background noise;
- *relative sensitivity* defined as the smallest detectable increment of light intensity.

As any real sensor also CCD is subjected to various sources of noise:

- the *photon noise* due to the particle nature of light; photon noise is always present and can be described as a white noise;
- the *quantization noise* due to the discretization of the continuous light intensity signal;
- the *thermic noise*; this noise source becomes relevant at the increasing of the CCD working temperature.

2.3.3.2 CMOS

A CMOS sensor (Complementary Metal Oxide Semiconductor) is composed by a rectangular matrix of photodiodes (the CMOS pixels defining image resolution). The junction of each photodiode is preloaded and is unloaded every time it is hit by photons. An integrated amplifier in each pixel is able to transform the photon charge in a precise current or voltage level. The main difference with the CCD sensor is that CMOS pixels don't perform time integration: after their activation indeed CMOS pixels measure the photons quantity of and not the photon volume. By this way CMOS sensor solves the overflow effect typical of CCD. However CMOS sensor are more expensive of CCD and for this reason are not so widely used.

2.4 A Hierarchy of Camera Models

This section introduces the most used mathematical models of image formation process through a camera sensor. Since cameras are the key sensors of a visual servoing system, to develop suitable geometrical model for image projection, according to both system specifications and image sensor features, is essential for the designing strategy of the control architecture. The level of abstraction and complexity in modeling image formation must trade off physical constraints and mathematical simplicity in order to result in a manageable camera model, one that can be inverted with reasonable effort. The section describes a hierarchy of geometrical camera models useful for vision systems.

2.4.1 The thin lens model

The simplest possible model of light propagation through a lens is the one of the *thin lens*: it is defined by an axis, called the *optical axis*, and a plane perpendicular to the axis, called *focal plane*, with a circular aperture centered in the *optical center* (the intersection of the focal plan with the optical axis), as shown in figure (2.3). The thin lens has two parameters: its focal length f and its diameter d. Its propagation model for the light is characterized by two properties:

- all rays entering the aperture parallel to the optical axis intersect on the optical axis at a distance f from the optical center; the point of intersection is called *focus* of the lens;
- all rays passing through the optical center remain undeflected.

Consider a point $\boldsymbol{X} \in \mathbb{R}^{3\times1}$ not too far from the optical axis at a distance Z from the focal plane. Now direct two rays from the point $\boldsymbol{X}$: one parallel to the optical axis and one through the optical center. The first one intersects the optical axis at the focus while the second remains undeflected. Call $\boldsymbol{x}$ the point where the two rays intersect and let be z its distance from the focal plane. By decomposing any other ray from $\boldsymbol{P}$ into a component ray parallel to optical axis and one through the optical center, we can easily see that all rays from $\boldsymbol{X}$ intersect at $\boldsymbol{x}$ in the opposite side of the lens. Let be respectively H and h the distances of $\boldsymbol{X}$ and $\boldsymbol{x}$ from the optical axis; exploiting the properties of similar

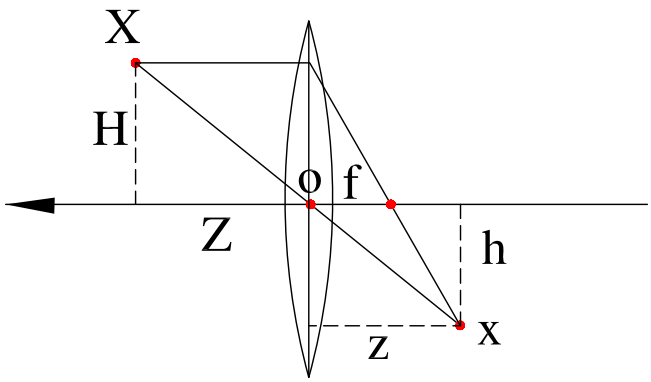


Figure 2.3: The thin lens model.

triangles we have:

$$\begin{cases} \frac{H+h}{Z+z} = \frac{h}{z} \\ \\ \frac{H}{h} = \frac{f}{z-f} \end{cases} . \tag{2.13}$$

Injecting the second equation of (2.13) in the first by making $\frac{H}{h}$ explicit, it is possible to obtain the *fundamental equation of the thin lens*:

$$\frac{1}{Z} + \frac{1}{z} = \frac{1}{f} . \tag{2.14}$$

The point $\boldsymbol{x}$ will be called the projection or simply the "image" of $\boldsymbol{X}$: be careful to don't confuse $\boldsymbol{x}$ with the irradiance $I(\boldsymbol{x})$ defined before.

2.4.2 Basic pinhole camera

If we let the aperture of a thin lens decrease to zero, all rays are forced to go through the optical center $\boldsymbol{o}$, and therefore remain undeflected. In such a configuration only the points that contribute to the irradiance at image point $\boldsymbol{x} = [x\, y]^T$ are on a line through the center $\boldsymbol{o}$ of the lens. Let be $\boldsymbol{X} = [XYZ]^T$ a $3D$ point relative to a reference frame $\mathcal{C}$, centered at the optical center $\boldsymbol{o}$, with its z-axis being the lens optical axis pointing to the scene, defined as *camera frame*. It is immediate to see from triangles similitude that the coordinates of $\boldsymbol{X}$ and of its corresponding image $\boldsymbol{x}$ on the *image plane* are related by the so called *central perspective projection*:

$$x = -f\frac{X}{Z} , \qquad y = -f\frac{Y}{Z} , \tag{2.15}$$

where f is referred as *focal length*. Simply we write that the projection is a map η:

$$\eta : \mathbb{R}^{3\times 1} \to \mathbb{R}^{2\times 1}; \qquad \boldsymbol{X} \mapsto \boldsymbol{x} . \tag{2.16}$$

We also often write $\boldsymbol{x} = \eta(\boldsymbol{X})$. Notice that any other point on the same ray through $\boldsymbol{o}$ and $\boldsymbol{X}$ projects onto the same image point $\boldsymbol{x} = [x\, y]^T$. This imaging model is called *pinhole camera model*. It is an idealization of the thin lens model since, when lens aperture

decreases, diffractions effect become dominant and so the purely refractive thin lens model do not hold. Furthermore as lens aperture decreases to zero, the energy going through the lens also becomes zero. Although it is possible to actually build devices that approximate pinhole cameras, this model is just an simplification of a well-focused imaging system. The negative sign present in equation (2.15) make the image of an object to appear upside down on the image plane (also called retina). To eliminate this effect, we can just flip the image: $(x, y) \mapsto (-x, -y)$. This correspond to place the image plane ($z = -f$) between the optical center and the scene ($z = f$) obtaining a more convenient "frontal" pinhole camera model. In this case the relation between a point $\boldsymbol{X}$ and its image $\boldsymbol{x}$ becomes:

$$x = f\frac{X}{Z}\,, \qquad y = f\frac{Y}{Z}\,, \tag{2.17}$$

In practice, the size of the image plane is limited, hence not every point $\boldsymbol{X}$ in the space will generate an image inside this plane. We define *field of view* (FOV) the angle subtended by the spatial size of the sensor as seen from the optical center. If $2r$ is the larger side of the sensor (the base of the CCD) then the field of view is $\theta = 2\arctan(r/f)$. Since in a classic camera a flat plane is used as the image plane, then θ is always less than 180°. If we use homogeneous coordinates to express image point $\boldsymbol{x} = [x, y, 1]^T$, the basic pinhole projection model can be expressed by the composition of two linear transformation with scaling $1/Z$ as follows:

$$\boldsymbol{x} = \eta(\gamma(\boldsymbol{X}))\,\tfrac{1}{Z} \;\Rightarrow\; \boldsymbol{x} = \boldsymbol{K}_f\,(\boldsymbol{K}_I\,\boldsymbol{X})\,\tfrac{1}{Z} = \boldsymbol{K}\boldsymbol{X}\,\tfrac{1}{Z} = \boldsymbol{K}\,\tilde{\boldsymbol{m}}$$

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tfrac{1}{Z}\,. \tag{2.18}$$

The vector $\tilde{\boldsymbol{m}}$ inside equation (2.18) is called **normalized image point**:

$$\tilde{\boldsymbol{m}} \;=\; \begin{bmatrix} X/Z \\ Y/Z \\ 1 \end{bmatrix} \;=\; \frac{1}{Z}\,\boldsymbol{X}\;; \tag{2.19}$$

it represents the image of $\boldsymbol{X}$ observed by an *ideal pinhole camera* with $f = 1$ applying the map γ (corresponding to the identity 3×3 matrix $\boldsymbol{K}_I$), while $\boldsymbol{K}_f = diag(f, f, 1)$, applies the map η and embeds the focal length of the actual camera.
The matrix $\boldsymbol{K} = \boldsymbol{K}_f\,\boldsymbol{K}_I$, defined in equation (2.18), is called *intrinsic camera calibration matrix* containing all characteristic camera own parameters. The pinhole model is thus defined by a single parameter: the focal length f (a time varying f models a zooming camera).
The principal point, the image plane, the camera frame etc. are ideal geometric entities modeling the camera projection mapping: they don't refer directly to any physical camera components.

2.4.3 Full CCD perspective camera

The basic pinhole model is specified relative to a very particular frame $\mathcal{C}$ (the camera or canonical retina frame) as defined in (2.4.2). In practice when one captures images with a digital camera the measurements are obtained in terms of pixels (i, j), with the origin of the image coordinate typically in the upper-left corner of the image. In order to render usable the model of equation (2.18), we need to specify the relationship between the image plane of the pinhole model(retinal plane) and the pixel array.
The first step consists in specifying the units along the $x-$ and $y-$ axes in the retinal plane: if the pinhole image (x, y) is specified in terms of metric units (e.g. millimeters), and (x_s, y_s) are scaled version corresponding to pixel coordinates, then the transformation can be described by the scaling matrix:

$$\begin{bmatrix} x_s \\ y_s \end{bmatrix} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.20}$$

that depends on the pixel size (in metric units) along the x and y directions. When $s_x = s_y$ each pixel is square. In general they can be different and the pixel is rectangular. However x_s and y_s are still specified relative to the *principal point* (defined as the intersection of $\mathcal{C}$ $z-$ axis and the retina plane), whereas pixel index (i, j) is conventionally specified relative to the upper left corner, and is indicated by positive numbers. Therefore we need to translate the origin of image plane coordinate system to this corner:

$$\begin{cases} p_x = x_s + o_x \\ p_y = y_s + o_y \ , \end{cases} \tag{2.21}$$

where (o_x, o_y) ore the pixel coordinates of the principal point relative to the upper left centered pixel image coordinate system. So the actual image pixel coordinates are given in homogeneous coordinates by the vector $\boldsymbol{p} = [p_x, p_y, 1]^T$ instead of the pinhole image coordinates $\boldsymbol{x} = [x, y, 1]^T$. Considering the above steps of coordinate transformation can be written as:

$$\boldsymbol{p} = \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \ , \tag{2.22}$$

with p_x and p_y actual image coordinates in pixels. In the case the pixel sides are not perpendicular a more general form for the scaling matrix can be considered:

$$\begin{bmatrix} s_x & s_\theta \\ 0 & s_y \end{bmatrix} \quad \in \mathbb{R}^{2\times 2} \ ; \tag{2.23}$$

s_θ is called *skew factor* and is proportional to $\cot(\theta)$ where θ is the angle between pixel sides. The transformation matrix in (2.22) takes the general form:

$$\boldsymbol{K}_s = \begin{bmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \quad \in \mathbb{R}^{3\times 3} \ . \tag{2.24}$$

In many practical applications θ is very close to 90°, and hence it is common to assume that $s_\theta = 0$. Now, combining the scaling of equation (2.22) with the pinhole model of equation (2.18), we obtain the *full perspective CCD camera model*:

$$\boldsymbol{p} = \sigma(\,\eta(\,\gamma(\,\boldsymbol{X}\,)\,)\,)\,\tfrac{1}{Z} \;\Rightarrow\; \boldsymbol{p} = \boldsymbol{K}_s(\boldsymbol{K}_f\;(\boldsymbol{K}_I\,\boldsymbol{X}))\,\tfrac{1}{Z} = \boldsymbol{K}\boldsymbol{X}\,\tfrac{1}{Z} = \boldsymbol{K}_s\,\boldsymbol{K}_f\,\tilde{\boldsymbol{m}}$$

$$\begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tfrac{1}{Z}\,. \tag{2.25}$$

Notice that in this case the intrinsic camera calibration matrix $\boldsymbol{K}$ results:

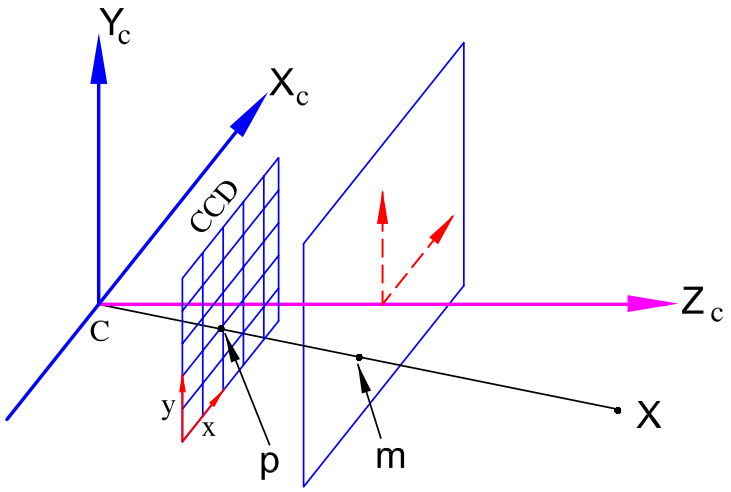


Figure 2.4: The CCD camera model.

$$\boldsymbol{K} \;=\; \begin{bmatrix} f\,s_x & f\,s_\theta & o_x \\ 0 & f\,s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}\,, \tag{2.26}$$

while the relation between a normalized point $\tilde{\boldsymbol{m}}$ and a pixel point $\boldsymbol{p}$ results:

$$\boldsymbol{p} \;=\; \boldsymbol{K}\tilde{\boldsymbol{m}}\,. \tag{2.27}$$

The entries of matrix $\boldsymbol{K}$ have the following geometric interpretation (see figure 2.4):

- o_x and o_y are the principal point pixel coordinates:
- $f\,s_x \;=\; \alpha_x$ is the focal length expressed in pixel horizontal unit;
- $f\,s_x \;=\; \alpha_y$ is the focal length expressed in pixel vertical unit;
- α_x/α_y is the *aspect ratio* σ;

- $f\, s_\theta$ is the skew parameter of the pixel often close to zero.

When $\boldsymbol{K}$ is known, we say that the system is *calibrated*: by inversion of equation (2.27) is thus possible to find the normalized point coordinates from the respective pixel ones. Furthermore it is easy to see that a normalized image point $\tilde{\boldsymbol{m}} = [m_x, m_y, 1]^T$ represents also the direction of the ray through the camera optical center and the $3D$ point $\boldsymbol{X}$ expressed in $\mathcal{C}$: calibration makes thus possible to know the direction of $3D$ ray from the camera optical center.

2.4.4 CCD camera with lens distortion

In addition to the affine mapping defined by intrinsic calibration matrix $\boldsymbol{K}$, if a cheap camera with a wild field of view is used (like commercial web-cams), one can often observe significant distortion in the images: $3-D$ straight lines appear no more straight in the image but result noticeably curved. To overcome this problem, distortion has to be properly modeled in order to compensate for it. In this thesis we will consider the distortion model first introduced by Brown in 1966 and called "*Plumb Bob*" model ("*radial polynomial*" plus "*thin prism*" models). This model considers two distortion sources: the *radial distortion* defined by three coefficients and the *tangential distortion* defined by two coefficients: while the first arises for cameras with wide FOV, the second is due to "decentering", or imperfect centering of the lens components and other manufacturing defects in a compound lens. Plumb Bob distortion model consists in a *non linear transformation* between the normalized image point $\boldsymbol{m} = [m_x,\, m_y]^T = [X/Z,\, Y/Z]^T$ and the respective *distorted normalized image point* $\boldsymbol{m}_d = [m_{dx},\, m_{dy}]^T$ (expressed in non homogeneous coordinates); it is defined as follows:

$$\boldsymbol{m}_d \;=\; k_r\, \boldsymbol{m} \,+\, \boldsymbol{d}_t \quad , \tag{2.28}$$

where:

- $\boldsymbol{d}_t$ is the *tangential distortion vector*;
- k_r is the *radial distortion factor* (scalar).

The scalar k_r and the vector $\boldsymbol{d}_t$ are defined in the following forms:

$$\begin{cases} k_r &= 1 \,+\, c_1\, r^2 \,+\, c_2\, r^4 \,+\, c_3\, r^6 \\ \boldsymbol{d}_t &= \begin{bmatrix} 2\, p_1\, m_x\, m_y \,+\, p_2\, (r^2 \,+\, 2\, m_x^2) \\ 2\, p_2\, m_x\, m_y \,+\, p_1\, (r^2 \,+\, 2\, m_y^2) \end{bmatrix} \;\in\; \mathbb{R}^{2\times 1}\,; \end{cases} \tag{2.29}$$

$r^2 = (m_x^2 + m_y^2)$ is the distance of the normalized image point from the principal point, while (c_1, c_2, c_3) and (p_1, p_2) are respectively the radial and tangential distortion coefficients. Once defined $\boldsymbol{m}_d$, to find the corresponding image point in pixel coordinates $\boldsymbol{p}$, the CCD projective model is finally applied:

$$\boldsymbol{p} = \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} \;=\; \begin{bmatrix} f\, s_x & f\, s_\theta & o_x \\ 0 & f\, s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_{dx} \\ m_{dy} \\ 1 \end{bmatrix} \tag{2.30}$$

The projection of a $3D$ point $\boldsymbol{X}$ on the image pixel point $\boldsymbol{p}$ by the CCD perspective camera with Plumb Bob distortion model, can be thus decomposed into four concatenate mappings:

$$\boldsymbol{p} = \sigma(\,\eta(\,\delta(\,\gamma(\,\boldsymbol{X}\frac{1}{Z}\,)\,)\,)\,) \;\Rightarrow\; \boldsymbol{p} = \boldsymbol{K}_s\,\boldsymbol{K}_f\,(\delta\,(\boldsymbol{K}_I\,\boldsymbol{X}\frac{1}{Z})) \;=\; \boldsymbol{K}_s\,\boldsymbol{K}_f\,\delta(\boldsymbol{m}) \;=\; \boldsymbol{K}\,\tilde{\boldsymbol{m}}_d\;. \tag{2.31}$$

Notice that in this case the mapping δ represented by equation (2.28) is no more linear like σ, η and γ and is moreover described by non-homogeneous coordinates.

2.4.4.1 Undistort an image

Supposing to deal with a calibrated system, that is to know both $\boldsymbol{K}$ and the five distortion coefficients ($c1$, $c2$, $c3$, $p1$, $p2$), if we want to find the corresponding normalized image point $\boldsymbol{m}$ starting from the pixel image point $\boldsymbol{p}$, we have execute two steps:

- compute the distorted normalized point from the corresponding pixel one as $\tilde{\boldsymbol{m}}_d \;=\; \boldsymbol{K}^{-1}\,\boldsymbol{p}$;
- *compensate for the distortion* determining $\boldsymbol{m} \;=\; \delta^{-1}(\boldsymbol{m_d})$ in equation (2.31).

The second step has been solved at Oulu University by using iterative procedure realizing the inversion of (2.28) summarized as follows:

$$\begin{cases} \textit{initialization} & \rightarrow \quad \boldsymbol{m}^1 \;=\; \boldsymbol{m}_d \\ \textit{iteration} & \rightarrow \quad \boldsymbol{m}^{i+1} = \frac{\boldsymbol{m}_d - \boldsymbol{d}_t(\boldsymbol{m}^i)}{k_r\,(\boldsymbol{m}^i)} \\ \textit{stop condition} & \rightarrow \quad (\boldsymbol{m}^{i+1} \;-\; \boldsymbol{m}^i) \;=\; \mu < \epsilon \end{cases} \tag{2.32}$$

As clear from (2.32), the unknown point $\boldsymbol{m}$ (the cycle iterating variable) is initialized with $\boldsymbol{m}_d$. The iterations stop when μ is less than a certain accuracy threshold ϵ: usually the procedure converges for $i < 10$.

2.4.5 Extrinsic camera parameters

Usually $3D$ point coordinates $\boldsymbol{X}$ are not directly available with respect to the camera frame $\mathcal{F}_c$ but are only known with respect to a *world frame* defined as $\mathcal{F}_o$: let be such point world based coordinates $\boldsymbol{X}_o \;=\; [X_o, Y_o, Z_o, 1]^T$ in homogeneous coordinates.
The rotation matrix $\boldsymbol{R}_o^c \in SO(3)$ and the translation vector $\boldsymbol{t}_o^c \in \mathbb{R}^{3\times 1}$ defining the rigid-body motion between the camera frame and the world frame, are known as *extrinsic calibration parameters*; $\boldsymbol{X}$ and $\boldsymbol{X}_o$ are linked by this equation:

$$\boldsymbol{X} \;=\; \left[\; \boldsymbol{R}_o^c \quad \boldsymbol{t}_o^c\,. \;\right] \boldsymbol{X}_o \tag{2.33}$$

The overall model of image formation between a $3D$ point $\boldsymbol{X}_o$ and its pixel image $\boldsymbol{p}$, neglecting distortion effects, can thus be modeled by joining equations (2.25) and (2.33):

$$Z\boldsymbol{p} = \boldsymbol{K}_s \boldsymbol{K}_f \boldsymbol{K}_I \left[\begin{array}{cc} \boldsymbol{R}_o^c & \boldsymbol{t}_o^c \end{array} \right] \boldsymbol{X}_o = \boldsymbol{K} \left[\begin{array}{cc} \boldsymbol{R}_o^c & \boldsymbol{t}_o^c \end{array} \right] \boldsymbol{X}_o = \left[\begin{array}{cc} \boldsymbol{K}\boldsymbol{R}_o^c & \boldsymbol{K}\boldsymbol{t}_o^c \end{array} \right] \boldsymbol{X}_o \,,$$

$$Z \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & s_\theta & o_x \\ 0 & s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_o^c & \boldsymbol{t}_o^c \\ & \end{bmatrix} \begin{bmatrix} X_o \\ Y_o \\ Z_o \\ 1 \end{bmatrix} . \tag{2.34}$$

The matrix $\left[\begin{array}{cc} \boldsymbol{K}\boldsymbol{R}_o^c & \boldsymbol{K}\boldsymbol{t}_o^c \end{array} \right] \in \mathbb{R}^{3\times 4}$ is called *camera projection matrix* and embeds both internal and external calibration parameters. This model is the most appropriate for medium and expensive professional firewire cameras: on the contrary for cheaper cameras like webcams distortion effects in image formation can't be neglected.
There exists several ways to retrieve the camera projection matrix and then the intrinsic matrix for a certain camera, assuming a precise geometric model among the ones described above. These techniques are known in literature as *camera calibration techniques*. Most of this calibration strategy use the image correspondences obtained from $3D$ textured "calibration grid" or just a planar chessboard and the corresponding known $3D$ grid model [MSKS03], [HZ03]. Other approach, on the contrary, called *self-calibration techniques* exploit the special target structure to retrieve the camera intrinsics [CDBP05] as we will see in chapter (5).

Chapter 3

Geometry of two views

Now we investigate the geometric relationships that can be derived from two views of a 3D object of interest grabbed from different camera poses; such relations summarize the fundamental concepts of two view projective geometry and will be useful for the further developments of the image based control path planning strategy.

3.1 Planar Parallax Equation

Consider an initial view I_i and a final view I_f of a 3D target of interest respectively related to an initial camera frame C_i and to a final camera frame C_f: let be the target identified in both the views by several well detectable points and suppose to know the point correspondences between the views. Let be π a plane passing through the target with normal $\boldsymbol{n}_i$ expressed in C_i and define $\boldsymbol{P}$ a generic 3D target point, as shown in figure (3.1). Let be $\boldsymbol{R}_i^f$ and $\boldsymbol{t}_i^f$ the orthogonal rotation matrix and the translation vector between C_f and C_i (namely the columns of $\boldsymbol{R}_i^f$ are the unit vector of C_i expressed in C_f while $\boldsymbol{t}_i^f$ is the origin of C_i expressed in C_f).
The point $\boldsymbol{P}$ can be expressed in C_f from both its projection in C_i and the cinematic parameters defining C_i with respect to C_f as:

$$\boldsymbol{P}_f = \boldsymbol{R}_i^f \boldsymbol{P}_i + \boldsymbol{t}_i^f \tag{3.1}$$

Let us define respectively h as the signed distance between $\boldsymbol{P}$ and the plane π while let be $d_{\pi\, i}$ the distance between C_i origin and π. It is clear from the figure that the following relation holds:

$$\boldsymbol{n}_i^T \boldsymbol{P}_i \;=\; d_{\pi\, i} + h \quad \Rightarrow \quad 1 \;=\; \frac{\boldsymbol{n}_i^T \boldsymbol{P}_i - h}{d_{\pi\, i}} \tag{3.2}$$

Injecting equation(3.2) in equation(3.1) we obtain:

$$\begin{aligned} \boldsymbol{P}_f &= \boldsymbol{R}_i^f \boldsymbol{P}_i + \boldsymbol{t}_i^f \left(\tfrac{\boldsymbol{n}_i^T \boldsymbol{P}_{i\,-h}}{d_{\pi\, i}} \right) \\ \boldsymbol{P}_f &= \left[\boldsymbol{R}_i^f \boldsymbol{P}_i + \boldsymbol{t}_i^f \left(\tfrac{\boldsymbol{n}_i^T \boldsymbol{P}_i}{d_{\pi\, i}} \right) \right] - \tfrac{h}{d_{\pi\, i}} \boldsymbol{t}_i^f \quad . \end{aligned} \tag{3.3}$$

Suppose without loss of generality to have used the same camera with fixed parameters (no zoom) to grab both I_i and I_f and to use the full CCD perspective model of subsection

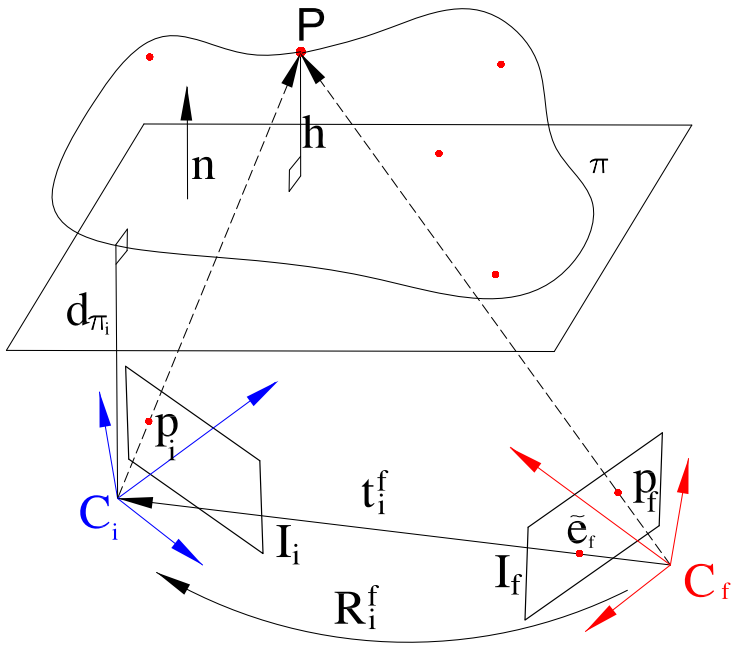


Figure 3.1: 3*D* Scene: the two cameras and the 3*D* target.

(2.4.3); camera intrinsic matrix $\boldsymbol{K}$ is also supposed to be known. As pointed in equation (2.25), we have:

$$\begin{aligned} \boldsymbol{p}_i = \tfrac{1}{Z_i}\boldsymbol{K}\boldsymbol{P}_i \quad &\Rightarrow \quad \boldsymbol{P}_i = Z_i\boldsymbol{K}^{-1}\boldsymbol{p}_i \\ \boldsymbol{p}_f = \tfrac{1}{Z_f}\boldsymbol{K}\boldsymbol{P}_f \quad &\Rightarrow \quad \boldsymbol{P}_f = Z_f\boldsymbol{K}^{-1}\boldsymbol{p}_f \quad , \end{aligned} \tag{3.4}$$

where $\boldsymbol{p}_i$ and $\boldsymbol{p}_i$ are the pixel projection of $\boldsymbol{P}$ respectively in I_i and I_f. By injecting relations (3.4) in equation (3.3), after easy algebraic manipulations we obtain:

$$\frac{Z_f}{Z_i}\boldsymbol{p}_f \;=\; \boldsymbol{K}\left[\boldsymbol{R}_i^f + \boldsymbol{t}_i^f\left(\frac{\boldsymbol{n}_i^T}{d_{\pi\,i}}\right)\right]\boldsymbol{K}^{-1}\boldsymbol{p}_i \;-\; \frac{h}{Z_i\;d_{\pi\,i}}\boldsymbol{K}\boldsymbol{t}_i^f \quad . \tag{3.5}$$

Now it is time to introduce the following entites:

$$\boldsymbol{G}_i^f \;=\; \boldsymbol{K}\left[\boldsymbol{R}_i^f + \boldsymbol{t}_i^f\left(\tfrac{\boldsymbol{n}_i^T}{d_{\pi\,i}}\right)\right]\boldsymbol{K}^{-1} \quad , \tag{3.6}$$

$$\tilde{\boldsymbol{e}}_f \;=\; \boldsymbol{K}\boldsymbol{t}_i^f \quad . \tag{3.7}$$

By using equations (3.6) and (3.7), relation (3.5) assumes a more compact form:

$$\frac{Z_f}{Z_i}\boldsymbol{p}_f \;=\; \boldsymbol{G}_i^f\boldsymbol{p}_i \;-\; \frac{h}{Z_i\;d_{\pi\,i}}\tilde{\boldsymbol{e}}_f \tag{3.8}$$

that is

$$\boldsymbol{p}_f \;\propto\; \boldsymbol{G}_i^f\boldsymbol{p}_i \;-\; \lambda_i\,\tilde{\boldsymbol{e}}_f \quad , \tag{3.9}$$

where $\lambda_i = \left(\frac{h}{Z_i\, d_{\pi\, i}}\right)$. Relation (3.9) is known as *Plane-Parallax Equation*: it states that the image of a generic $3D$ point $\boldsymbol{P}$ in the final image I_f is composed of two elements:

- the first is function of the initial image pixel point $\boldsymbol{p}_i$ mapped through the projectivity matrix $\boldsymbol{G}_i^f \in \mathbb{R}^{3\times 3}$; $\boldsymbol{G}_i^f$ is called as *pixel homography matrix* between I_i and I_f induced by the plane π;
- the second is proportional, according to λ_i, to the homogeneous unnormalized point $\tilde{\boldsymbol{e}}_f \in \mathbb{R}^{3\times 1}$. $\tilde{\boldsymbol{e}}_f$ is known as *epipole* of I_f (expressed in homogeneous coordinates) and represents the image of C_i origin in I_f.

3.2 Plane induced Homography

Notice that if the $3D$ point $\boldsymbol{P} \in \pi$ (that is $h = 0$) or if the translation $\boldsymbol{t}_i^f = \boldsymbol{0}$, the second terms in the right member of (3.9) vanishes and the plane-parallax decomposition simplifies in the following equation:

$$\boldsymbol{p}_f \;\propto\; \boldsymbol{G}_i^f \boldsymbol{p}_i \;=\; \frac{Z_i}{Z_f}\, \boldsymbol{G}_i^f \boldsymbol{p}_i \quad . \tag{3.10}$$

In this last case image I_i is mapped to I_f just through the *Pixel Homography Matrix* $\boldsymbol{G}_i^f$. The Pixel Homography Matrix is a full rank matrix strictly related to the plane π, namely it is induced by π; $\boldsymbol{G}_i^f$ is only function of the C_i, C_f, and π relative displacements while is invariant with respect to the 3D target imaged points: moreover if the plane π and the initial camera frame C_i are fixed in the space, $\boldsymbol{G}_i^f$ has only 6 DOF, namely the six rigid body independent degrees of freedom to define the final camera frame C_f with respect to C_i (3 embedded in the orthogonal matrix $\boldsymbol{R}_i^f$ and 3 to define the translation vector $\boldsymbol{t}_i^f$). Considering the following kinematic relations:

$$\begin{aligned} \boldsymbol{R}_i^f \;&=\; \boldsymbol{R}_f^{i\,-1} \;=\; \boldsymbol{R}_f^{i\,T} \quad , \\ \boldsymbol{t}_i^f \;&=\; -\boldsymbol{R}_f^{i\,T} \boldsymbol{t}_f^i \quad , \end{aligned} \tag{3.11}$$

the pixel homography matrix of equation (3.6) can also be written as:

$$\boldsymbol{G}_i^f \;=\; \boldsymbol{K}\boldsymbol{R}_i^f \left[\boldsymbol{I} - \boldsymbol{t}_f^i \left(\frac{\boldsymbol{n}_i^T}{d_{\pi\, i}}\right)\right] \boldsymbol{K}^{-1} \quad , \tag{3.12}$$

or

$$\boldsymbol{G}_i^f \;=\; \boldsymbol{K}\boldsymbol{R}_f^{i\,T} \left(\boldsymbol{I} - \boldsymbol{t}\boldsymbol{s}_f^i \boldsymbol{n}_i^T\right) \boldsymbol{K}^{-1} \quad , \tag{3.13}$$

where $\boldsymbol{t}\boldsymbol{s}_f^i = (\boldsymbol{t}_f^i / d_{\pi\, i})$ is the scaled translation between C_i and C_f, being all the elements of equation (3.13) expressed with respect to C_i.
Let now define the *Euclidean Homography Matrix* $\boldsymbol{H}_i^f$ as:

$$\boldsymbol{H}_i^f \;=\; \boldsymbol{R}_f^{i\,T} \left(\boldsymbol{I} - \boldsymbol{t}\boldsymbol{s}_f^i \boldsymbol{n}_i^T\right) = \left(\boldsymbol{R}_i^f + \boldsymbol{t}\boldsymbol{s}_i^f \boldsymbol{n}_i^T\right); \tag{3.14}$$

by injecting (3.14) in (3.13), the pixel homography matrix can be finally expressed by:

$$\boldsymbol{G}_i^f = \boldsymbol{K}\boldsymbol{H}_i^f\boldsymbol{K}^{-1} \quad . \tag{3.15}$$

The Euclidean Homography $\boldsymbol{H}_i^f$ is the corresponding homography to $\boldsymbol{G}_i^f$ when the target image points are expressed in normalized coordinates (see equation (2.19) that is considering a calibration matrix $\boldsymbol{K} = \boldsymbol{I}$:

$$\tilde{\boldsymbol{m}}_f \;\propto\; \boldsymbol{H}_i^f\tilde{\boldsymbol{m}}_i \;=\; \frac{Z_i}{Z_f}\,\boldsymbol{H}_i^f\tilde{\boldsymbol{m}}_i \quad . \tag{3.16}$$

- $\boldsymbol{H}_i^f$ depends only on the scene structure and is invariant with respect to the particular camera adopted.
- as clearly visible in equation (3.14) $\boldsymbol{H}_i^f$ embeds all the kinematic parameters defining both the relative camera poses between C_i and C_f and the π plane orientation.

3.3 Structure from Motion with two calibrated images

One of the most important issues of subjects like Computer Vision or Photogrammetry, is to derive from $2D$ images as much *informations* as possible concerning the projected Euclidean $3D$ space; in this sense images represents a set of first raw measurements that need to be conveniently processed and condensed in order to infer informations about $3D$ world.
The problem of retrieving from two calibrated view the $3D$ scene structure linking the target of interest and the two image-related cameras is known in literature as the *two view Structure from Motion Problem (SFM)*. In this section we explain how SFM, that is by its own nature nonlinear, can be solved in linear fashion both dealing with a planar or three-dimensional target; from the SFM solutions we will advantage of many $3D$ "image based" informations that will be useful both in the control and in the planning strategies reported in the following chapter. The linear solution can be directly exploited in the image based control or can be taken as the first step to provide the initialization of a non linear algorithm when time processing allows it. An example of non linear method for 3D data estimation is given in [CH04]. Standard linear computer vision methods, such as the "8-point" algorithm exists to solve the SFM problem [HZ03], but in this work we prefer to address strategies that directly exploits the two views homography theory of section (3) resulting more stable results and free from degenerate configurations, as pointed out in [MCB00].
Suppose then to have two images I_i and I_f as in section (3) of a $3D$ target taken respectively from the camera frames C_i and C_f; suppose to be able to extract n pixel image target points from I_i and their corresponding from I_f; let us know finally the point correspondences in the two images (namely tho have already solved the feature matching problem) and to know the constant camera calibration matrix $\boldsymbol{K}$. By considering the previous assumptions, SFM consists in the estimation of the following set of $3D$ parameters:

$$\left\{\boldsymbol{R}_f^i,\ \boldsymbol{ts}_f^i,\ \boldsymbol{n}_i\right\} \tag{3.17}$$

SFM can be subdivided into three main steps:

- estimation the Pixel Homography $\boldsymbol{G}_i^f$ from I_i and I_f;
- computation the Euclidean Homography $\boldsymbol{H}_i^f$ from $\boldsymbol{G}_i^f$ and $\boldsymbol{K}$;
- decomposition of $\boldsymbol{H}_i^f$ in order to find cameras C_i and C_f relative rotation $\boldsymbol{R}_f^i$, scaled translation $\boldsymbol{ts}_f^i$ and target plane normal $\boldsymbol{n}_i$.

For a valid reference concerning the solution of the SFM problem we remand also to [MSKS03].

3.3.1 Estimation of the Pixel Homography

The first step to perform *SFM* process id to estimate the Pixel homography $\boldsymbol{G}_i^f$ from image data. When the n detected feature target points are distributed in $3D$ space (*general target case*), a general algorithm must be used; if on the contrary at least four target points lie on the same plane (*planar target case*) a simpler method can be adopted.

3.3.1.1 General Case

Among the n target points, let us consider three non collinear points $\boldsymbol{P}^i$, $i = 1, 2, 3$. $\boldsymbol{P}^i$ project, according to equation (2.22), in the homogeneous pixel points $\boldsymbol{p}_i^i$ and $\boldsymbol{p}_i^f$ respectively in I_i and I_f. Notice that these three points has to be chosen so as to maximize the areas of the two triangles defined by their images in both views (imaged points have not to be collinear as well). Let us call *Virtual Plane* the plane defined by the three $3D$ points $\boldsymbol{P}^i$, denoted in figure (3.2) as π. Since $\boldsymbol{P}^i$ belong to the same plane, we can

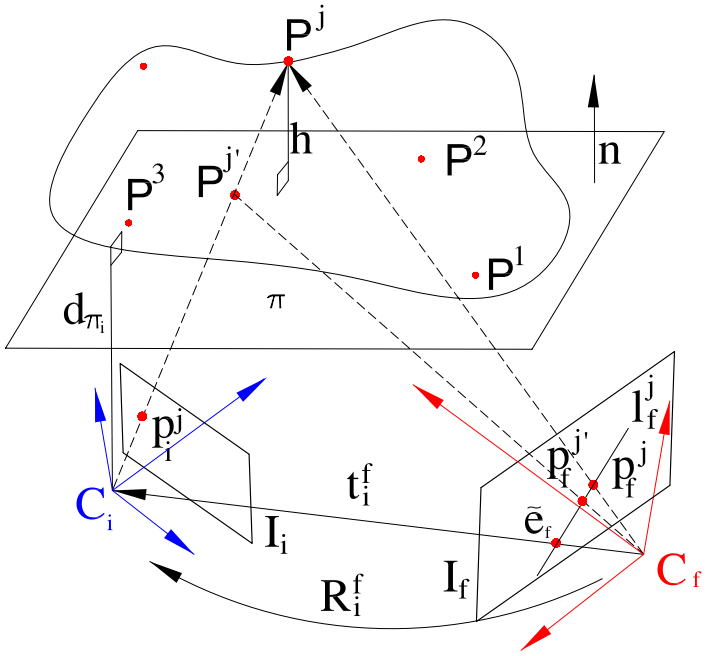


Figure 3.2: The two cameras, the $3D$ target and the corresponding images.

express the relation between their images through the unknown planar pixel homography

$\boldsymbol{G}_i^f$ according to equation (3.10):

$$\boldsymbol{p}_f^i \;\propto\; \boldsymbol{G}_i^f \boldsymbol{p}_i^i \qquad i \,=\, 1,2,3 \quad . \tag{3.18}$$

Let us remark that $\boldsymbol{G}_i^f$ is an homogeneous matrix defined up to a scalar factor, therefore one of its entries can be set to 1 without loss of generality.
When a general $3D$ target point $\boldsymbol{P}^j$ that does not belong to π, the line $\overline{O_i P^j}$ and the plane π intersect in a *virtual* $3D$ point $\boldsymbol{P}^{j'}$, as shown in the figure. $\boldsymbol{P}^{j'}$ and $\boldsymbol{P}^j$ project in the same point $\boldsymbol{p}_i^j$ in the initial image I_i, but they give rise to different image points in I_f (respectively the image point $\boldsymbol{p}_f^j$ and the *virtual image point* $\boldsymbol{p}_f^{j'} \propto \boldsymbol{G}_i^f \boldsymbol{p}_i^j$) : this is known as parallax effect (see section (3.1).
The full CCD perspective model, through the projection performed from $\boldsymbol{K}$ matrix, preserves collinearity; for this reason the points $\boldsymbol{O}_i$, $\boldsymbol{P}^{j'}$ and $\boldsymbol{P}^j$, belonging on the same $3D$ ray are projected in I_f respectively on the image points $\boldsymbol{e}_f$, $\boldsymbol{p}_f^{j'}$ and $\boldsymbol{p}_f^j$ belonging to the image line $\boldsymbol{l}_f^j$. Notice that $\boldsymbol{e}_f$ is *epipole* of I_f while $\boldsymbol{l}_f^j$ is an *epipolar line* (since it passes through the epipole).
$\boldsymbol{l}_f^j$ can thus be defined as the line through the points $\boldsymbol{p}_f^j$ and $\boldsymbol{p}_f^{j'}$; exploiting equation (3.18) and theorem (2.1.2.3), $\boldsymbol{l}_f^j$ we can thus write:

$$\boldsymbol{l}_f^j \;=\; \boldsymbol{p}_f^j \wedge \boldsymbol{G}_i^f \boldsymbol{p}_i^j \quad . \tag{3.19}$$

Let us choose other two general $3D$ target points $\boldsymbol{P}^k$ and $\boldsymbol{P}^l \notin \pi$, through their images, they respectively will give rise in I_f to the epipolar lines $\boldsymbol{l}_f^k$ and $\boldsymbol{l}_f^l$. Since all epipolar the lines of an image intersects at the epipole the following equation hold:

$$\left|\, \boldsymbol{l}_f^j \; \boldsymbol{l}_f^k \; \boldsymbol{l}_f^l \,\right| \;=\; 0 \quad , \tag{3.20}$$

where $|\quad|$ denotes the determinant operator. Relation (3.20) can be easily demonstrated with the following passages:

$$\begin{gathered}
\boldsymbol{e}_f \;=\; \alpha \left(\boldsymbol{l}_f^j \wedge \boldsymbol{l}_f^k\right) = \gamma \left(\boldsymbol{l}_f^j \wedge \boldsymbol{l}_f^l\right) \\
\boldsymbol{l}_f^j \wedge \left(\alpha \boldsymbol{l}_f^k\right) - \boldsymbol{l}_f^j \wedge \left(\gamma \boldsymbol{l}_f^l\right) \;=\; \boldsymbol{0} \\
\boldsymbol{l}_f^j \wedge \left(\alpha \boldsymbol{l}_f^k - \gamma \boldsymbol{l}_f^l\right) \;=\; \boldsymbol{0} \quad ;
\end{gathered} \tag{3.21}$$

with α and γ constants $\in (\mathbb{R} - 0)$. Equation (3.21) can be verified if and only if we have $\boldsymbol{l}_f^k = (\gamma/\alpha)\boldsymbol{l}_f^l$ or if $\boldsymbol{l}_f^j \;\propto\; \left(\alpha \boldsymbol{l}_f^k - \gamma \boldsymbol{l}_f^l\right)$, showing the three epipolar line vectors to be linearly dependent, thus verifying equation (3.20). By exploiting equation (3.19), relation (3.20) becomes:

$$\left|\, \boldsymbol{p}_f^j \wedge \boldsymbol{G}_i^f \boldsymbol{p}_i^j \quad \boldsymbol{p}_f^k \wedge \boldsymbol{G}_i^f \boldsymbol{p}_i^k \quad \boldsymbol{p}_f^l \wedge \boldsymbol{G}_i^f \boldsymbol{p}_i^l \,\right| \;=\; 0 \quad . \tag{3.22}$$

Equation (3.22) however is non-linear with respect to the unknown entries of the pixel homography matrix $\boldsymbol{G}_i^f$. In order to simplify the computation of $\boldsymbol{G}_i^f$ a change of projective

coordinates can be done. Let us define the two transformation matrices as follows:

$$\boldsymbol{M}_f = \left[\boldsymbol{p}_f^1\ \boldsymbol{p}_f^2\ \boldsymbol{p}_f^3\right]^{-1} \qquad \boldsymbol{M}_i = \left[\boldsymbol{p}_i^1\ \boldsymbol{p}_i^2\ \boldsymbol{p}_i^3\right]^{-1} \quad , \tag{3.23}$$

they have as columns the three image points used to define the virtual plane π. The transformed image points are thus defined respectively in I_f and I_i by these transformations:

$$\tilde{\boldsymbol{p}}_f^j = \boldsymbol{M}_f \boldsymbol{p}_f^j \qquad \tilde{\boldsymbol{p}}_i^j = \boldsymbol{M}_i \boldsymbol{p}_i^j \quad , \tag{3.24}$$

where with $\tilde{}$ we denote here the transformed entities. By applying transformations (3.24), it is clear that the coordinates of the three π imaged points become:

$$\left[\tilde{\boldsymbol{p}}_f^1\ \tilde{\boldsymbol{p}}_f^2\ \tilde{\boldsymbol{p}}_f^3\right] \quad = \quad \left[\tilde{\boldsymbol{p}}_i^1\ \tilde{\boldsymbol{p}}_i^2\ \tilde{\boldsymbol{p}}_i^3\right] \quad = \quad \boldsymbol{I} \quad \in \mathbb{R}^{3\times 3} \quad , \tag{3.25}$$

while transformed pixel homography matrix is easily derived considering equation (3.18):

$$\tilde{\boldsymbol{G}}_i^f = \boldsymbol{M}_f \boldsymbol{G}_i^f \boldsymbol{M}_i^{-1} \quad . \tag{3.26}$$

Notice that by applying the transformations (3.24), the three imaged points $\tilde{\boldsymbol{p}}_i^1$, $\tilde{\boldsymbol{p}}_i^2$ and $\tilde{\boldsymbol{p}}_i^3$ have become the so called *fixed points* of the transformed pixel homography $\tilde{\boldsymbol{G}}_i^f$ (namely the homography eigenvectors); furthermore they form according to (3.25) the canonical base of the transformed spaces; for the previous reasons the matrix $\tilde{\boldsymbol{G}}_i^f$ results diagonal:

$$\tilde{\boldsymbol{G}}_i^f = \begin{bmatrix} \tilde{g_u} & 0 & 0 \\ 0 & \tilde{g_v} & 0 \\ 0 & 0 & \tilde{g_w} \end{bmatrix} \quad ; \tag{3.27}$$

according to equation (3.27), once $\tilde{g_u}$, $\tilde{g_v}$ and $\tilde{g_w}$ are estimated, $\tilde{\boldsymbol{G}}_i^f$ and thus $\boldsymbol{G}_i^f$ can be finally retrieved. Equation (3.22) can be rewritten after the coordinates transformations as:

$$\left| \tilde{\boldsymbol{p}}_f^j \wedge \tilde{\boldsymbol{G}}_i^f \tilde{\boldsymbol{p}}_i^j \quad \tilde{\boldsymbol{p}}_f^k \wedge \tilde{\boldsymbol{G}}_i^f \tilde{\boldsymbol{p}}_i^k \quad \tilde{\boldsymbol{p}}_f^l \wedge \tilde{\boldsymbol{G}}_i^f \tilde{\boldsymbol{p}}_i^l \right| = 0 \quad . \tag{3.28}$$

The previous equation results now homogeneous and polynomial of degree three in the unknown $(\tilde{g_u}, \tilde{g_v}, \tilde{g_w})$: moreover it does not depend on the epipole thus avoiding problems due to epipolar singularity configurations. After some time-consuming computation equation (3.28) can be rearranged in the homogeneous form:

$$\boldsymbol{C}_{\tilde{g}}^{jkl} \boldsymbol{x} = 0 \quad , \tag{3.29}$$

where the entries of the *measurements row vector* $\boldsymbol{C}_{\tilde{g}}^{jkl} \in \mathbb{R}^{1\times 7}$ are given in [MCB00] and the vector $\boldsymbol{x}$ is defined by:

$$\boldsymbol{x}^T = \left[\ \tilde{g_u}^2\tilde{g_v}\quad \tilde{g_v}^2\tilde{g_u}\quad \tilde{g_u}^2\tilde{g_w}\quad \tilde{g_v}^2\tilde{g_w}\quad \tilde{g_w}^2\tilde{g_u}\quad \tilde{g_w}^2\tilde{g_v}\quad \tilde{g_u}\tilde{g_v}\tilde{g_w}\ \right] \quad , \tag{3.30}$$

that is a cubic combination of the three normalized homography $\tilde{\boldsymbol{G}}_i^f$ unknown entries. Selecting different feature points combinations by changing i,j and k indexes (i.e selecting different epipolar lines) from the available $n-3$ (three have been used to define the virtual plane π), we can derive other homogeneous equations analogous to (3.29), obtaining new

measurements vectors $\boldsymbol{C}_{\tilde{g}}^{jkl}$. All the derived measurements vectors $\boldsymbol{C}_{\tilde{g}}^{jkl}$ can be finally stacked together building up a an overall measurement matrix $\boldsymbol{C}_{\tilde{g}} \in \mathbb{R}^{m \times 7}$ where $m >> 7$ is the number of derived $\boldsymbol{C}_{\tilde{g}}^{jkl}$ raw vectors; the resulting equations system can be finally written as:

$$\boldsymbol{C}_{\tilde{g}}\, \boldsymbol{x} \;=\; \boldsymbol{0} \quad \in \mathbb{R}^{(m>>7)\times 1} \quad ; \tag{3.31}$$

providing an linear homogeneous system of m equations in the seven $\boldsymbol{x}$ unknowns. Since from n epipolar line (one for each image point but the three points defining π) can be formed up to $m = \frac{n!}{(6(n-3)!)}$ different sets of three epipolar lines, we have that at least eight $3D$ target points (three reference points and five supplementary points) are needed as imaged features to solve system (3.31).
Once the measurements matrix $\boldsymbol{C}_{\tilde{g}}$ has been computed, the problem can be solved by performing the SVD of $\boldsymbol{C}_{\tilde{g}} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T$ and by selecting as solution the column of $\boldsymbol{V}$ corresponding to the minimal singular value s_{ii} (0 in absence of noise). However, since $\boldsymbol{C}_{\tilde{g}} \in \mathbb{R}^{m \times 7}$ with $m >> 7$, it is possible to obtain the same solution from the SVD of $\boldsymbol{C}_{\tilde{g}}^T \boldsymbol{C}_{\tilde{g}} = \boldsymbol{V}\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{V}^T$, which is of dimension (7×7): memory space and time processing are thus minimized.
Finally the three $\tilde{\boldsymbol{G}}_i^f$ unknowns $\tilde{g_u}$, $\tilde{g_v}$ and $\tilde{g_w}$ of equation (3.27) can be computed by similarly solving a second linear homogeneous system:

$$\begin{bmatrix} -\bar{x}_2 & \bar{x}_1 & 0 \\ \bar{x}_5 & 0 & -\bar{x}_3 \\ -\bar{x}_7 & \bar{x}_3 & 0 \\ \bar{x}_7 & 0 & -\bar{x}_1 \\ -\bar{x}_4 & \bar{x}_7 & 0 \\ \bar{x}_4 & 0 & -\bar{x}_2 \\ -\bar{x}_6 & 0 & \bar{x}_7 \\ 0 & -\bar{x}_6 & \bar{x}_4 \end{bmatrix} \begin{bmatrix} \tilde{g_u} \\ \tilde{g_w} \\ \tilde{g_w} \end{bmatrix} \;=\; \boldsymbol{0} \quad . \tag{3.32}$$

Once $\tilde{\boldsymbol{G}}_i^f$ is estimated, it is easy to compute the pixel homography matrix $\boldsymbol{G}_i^f$ by inversion of equation (3.26).

3.3.1.2 Planar Target or Rotational Motion Case

If at least four $3D$ target points lies on a plane π or if the motion between C_i and C_f is a pure rotation, we can use a simpler algorithm of the previous shown above known in literature as the *Direct Linear Transformation (DLT) algorithm*. Suppose then to have $n \geq 4$ coplanar target points $\boldsymbol{P}^i$, $i = 1, ..., n \in \pi$ (or $\boldsymbol{0}$ as camera translation vector) respectively projecting in I_i and I_f in the homogeneous pixel points $\boldsymbol{p}_i^i$ and $\boldsymbol{p}_i^f$, according to equation 2.22. As shown in (3.2)we have:

$$\boldsymbol{p}_f^i \;\propto\; \boldsymbol{G}_i^f \boldsymbol{p}_i^i \qquad i = 1, ..., n \quad . \tag{3.33}$$

Note that this is an equation involving homogeneous vector: $\boldsymbol{p}_f^i$ and $\boldsymbol{G}_i^f \boldsymbol{p}_i^i$ indeed are not equal, they have the same direction but may differ in magnitude by anon zero scale factor.

This fact can be expressed in terms of the vector cross product:

$$\boldsymbol{p}_f^i \wedge \boldsymbol{G}_i^f \boldsymbol{p}_i^i \; = \mathbf{0} \quad \forall i, \quad i = 1, ..., n \quad . \tag{3.34}$$

This form will enable a simple linear solution for $\boldsymbol{G}_i^f$ estimation problem. If the $j-$th row of $\boldsymbol{G}_i^f$ is denoted by $\boldsymbol{g}^{j\,T}$, then we may write:

$$\boldsymbol{G}_i^f \boldsymbol{p}_i^i \; = \; \begin{pmatrix} \boldsymbol{g}^{1\,T} \boldsymbol{p}_i^i \\ \boldsymbol{g}^{2\,T} \boldsymbol{p}_i^i \\ \boldsymbol{g}^{3\,T} \boldsymbol{p}_i^i \end{pmatrix} \quad . \tag{3.35}$$

Recalling that, as shown in (2.22), the pixel homogeneous point $\boldsymbol{p}_f^i = [p_{f\,x}^i,\, p_{f\,y}^i,\, 1]^T$, the cross product of (3.34) may then be given explicitly as:

$$\boldsymbol{p}_f^i \wedge \boldsymbol{G}_i^f \boldsymbol{p}_i^i \; = \; \begin{bmatrix} p_{f\,y}^i (\boldsymbol{g}^{3\,T} \boldsymbol{p}_i^i) & - & \boldsymbol{g}^{2\,T} \boldsymbol{p}_i^i \\ \boldsymbol{g}^{1\,T} \boldsymbol{p}_i^i & - & p_{f\,x}^i (\boldsymbol{g}^{3\,T} \boldsymbol{p}_i^i) \\ p_{f\,x}^i (\boldsymbol{g}^{2\,T} \boldsymbol{p}_i^i) & - & p_{f\,y}^i (\boldsymbol{g}^{1\,T} \boldsymbol{p}_i^i) \end{bmatrix} \; = \; \mathbf{0} \quad . \tag{3.36}$$

Since $\boldsymbol{g}^{j\,T} \boldsymbol{p}_i^i \; = \; \boldsymbol{p}_i^{i\,T} \boldsymbol{g}^j$ for $j \; = \; 1, ..., 3$, this gives a set of three equation in the entries of $\boldsymbol{G}_i^f$, which may be written in the form:

$$\begin{bmatrix} \mathbf{0}^T & -\boldsymbol{p}_i^{i\,T} & p_{f\,y}^i \boldsymbol{p}_i^{i\,T} \\ \boldsymbol{p}_i^{i\,T} & \mathbf{0}^T & -p_{f\,x}^i \boldsymbol{p}_i^{i\,T} \\ -p_{f\,y}^i \boldsymbol{p}_i^{i\,T} & p_{f\,x}^i \boldsymbol{p}_i^{i\,T} & \mathbf{0}^T \end{bmatrix} \begin{pmatrix} \boldsymbol{g}^1 \\ \boldsymbol{g}^2 \\ \boldsymbol{g}^3 \end{pmatrix} \; = \; \mathbf{0} \quad \in \mathbb{R}^{9 \times 1} \quad . \tag{3.37}$$

These equations have the form $\boldsymbol{A}_i \, \boldsymbol{g} \; = \; \mathbf{0}$, where $\boldsymbol{A}_i$ is a (3×9) matrix, and $\boldsymbol{g}$ is a (9×1) vector made up of the entries of the pixel homography matrix $\boldsymbol{G}_i^f$:

$$\boldsymbol{g} \; = \; \begin{pmatrix} \boldsymbol{g}^1 \\ \boldsymbol{g}^2 \\ \boldsymbol{g}^3 \end{pmatrix} , \qquad \boldsymbol{G}_i^f \; = \; \begin{bmatrix} g_1 & g_2 & g_3 \\ g_4 & g_5 & g_6 \\ g_7 & g_8 & g_9 \end{bmatrix} \quad , \tag{3.38}$$

with g_i the $i-$th element of $\boldsymbol{g}$. The resulting equation of (3.38) $\boldsymbol{A}_i \, \boldsymbol{g} \; = \; \mathbf{0}$ is *linear* in the unknown $\boldsymbol{g}$. The matrix elements of $\boldsymbol{A}_i$ are *quadratic* in the known pixel image points coordinates of I_i and I_f.
Although (3.37) is composed by three equations, only two of them are linearly independent (since the third row is obtained, up to a scale, from the sum of $p_{f\,x}^i$ times the first row and $p_{f\,y}^i$ times the second). Thus each point correspondence $\boldsymbol{p}_i^i \rightarrow \boldsymbol{p}_f^i$ gives two equations in the entries of $\boldsymbol{G}_i^f$: for this reason it is usual to omit the third equation in the solving system. The set of solving equation finally become:

$$\begin{bmatrix} \mathbf{0}^T & -\boldsymbol{p}_i^{i\,T} & p_{f\,y}^i \boldsymbol{p}_i^{i\,T} \\ \boldsymbol{p}_i^{i\,T} & \mathbf{0}^T & -p_{f\,x}^i \boldsymbol{p}_i^{i\,T} \end{bmatrix} \begin{pmatrix} \boldsymbol{g}^1 \\ \boldsymbol{g}^2 \\ \boldsymbol{g}^3 \end{pmatrix} \; = \; \boldsymbol{A}_i \, \boldsymbol{g} \; = \; \mathbf{0} \quad \in \mathbb{R}^{9 \times 1} \quad , \tag{3.39}$$

where now $\boldsymbol{A}_i$ is a (2×9) matrix. Staking together all the $(2 \times n)$ equations derived from

the n point correspondences, we obtain the overall system equation:

$$\boldsymbol{A}\,\boldsymbol{g} \;=\; \boldsymbol{0} \quad \in \mathbb{R}^{9\times 1} \quad , \tag{3.40}$$

where $\boldsymbol{A}$ is the matrix coefficients built from the matrix rows $\boldsymbol{A}_i$ contributed from each correspondence. If $n = 4$ $\boldsymbol{A}$ is a 8×9 with rank 8: a solution for $\boldsymbol{g}$ and thus for $\boldsymbol{G}_i^f$ is provided by vector belonging to the 1-dimensional *null space* of $\boldsymbol{A}$. Such a solution can be only determined up to a non-zero scale factor. However $\boldsymbol{G}_i^f$ is an homogeneous matrix defined up to a scale, so the solution $\boldsymbol{g}$ gives the required $\boldsymbol{G}_i^f$. A scale may be arbitrarily chosen for $\boldsymbol{g}$ by imposing that $\|\boldsymbol{g}\| = 1$.
If the number of available correspondences $n > 4$, then the set of equation in (3.40) is over determined. If points pixel coordinates are exact $\boldsymbol{A}$ will still have rank 8, a one-dimensional null space, and there is an exact solution for $\boldsymbol{h}$. This is will not the case that occurs when we have inexact noisy data (i.e. with real pixel measurements); in this case, instead of finding an impossible exact solutions for $\boldsymbol{g}$ we search for an optimal one as stated by the following constrained optimization problem: *find* $\boldsymbol{h}$ *that minimize* $\|\boldsymbol{A}\,\boldsymbol{g}\|$ *subject to the usual constraint* $\|h\| = 1$. The solution to the previous problem, as similarly done to solve (3.31) is given by performing the SVD of $\boldsymbol{A} = \boldsymbol{U}\,\boldsymbol{S}\,\boldsymbol{V}^T$ and by choosing as solution for $\boldsymbol{g}$ the (unit) column of $\boldsymbol{V}$ corresponding to the smallest singular value s_{ii} of $\boldsymbol{A}$.

3.3.2 Computation of the Euclidean Homography

Notice that in general the pixel collineation matrix $\boldsymbol{G}_i^f$, estimated in subsection (3.3.1) either with the general or with the simplified method, do not match the matrix defined in equations (3.12 - 3.13) but differs from it by a residual scale factor. This is a consequence of the fact that the homography between two views is a projective transformation of $\mathbb{P}^{2\times 1}$ that can be defined by any of the ∞^1 (3×3) matrices with the same 8 element ratios (see subsection (2.1.6)). The estimated $\boldsymbol{G}_i^f$, according to (3.6), will thus have the following form:

$$\boldsymbol{G}_i^f \;=\; \lambda\,\boldsymbol{K}\left[\boldsymbol{R}_i^f + \boldsymbol{t}_i^f\left(\frac{\boldsymbol{n}_i^T}{d_{\pi\,i}}\right)\right]\boldsymbol{K}^{-1} \quad , \tag{3.41}$$

where λ is the residual scale factor. Since the camera calibration matrix $\boldsymbol{K}$ is known, we can compute from the estimated $\boldsymbol{G}_i^f$ the following matrix:

$$\tilde{\boldsymbol{H}}_i^f = \boldsymbol{K}^{-1}\,\boldsymbol{G}_i^f\,\boldsymbol{K} \quad . \tag{3.42}$$

Notice that $\tilde{\boldsymbol{H}}_i^f$ is the Euclidean Homography $\boldsymbol{H}_i^f$ up to the residual factor λ - see equations (3.15) and (3.14):

$$\tilde{\boldsymbol{H}}_i^f \;=\; \lambda\boldsymbol{H}_i^f \;=\; \lambda\left[\boldsymbol{R}_i^f + \boldsymbol{t}_i^f\left(\frac{\boldsymbol{n}_i^T}{d_{\pi\,i}}\right)\right] \;=\; \lambda\left(\boldsymbol{R}_i^f + \boldsymbol{t}\boldsymbol{s}_i^f\boldsymbol{n}_i^T\right) \quad . \tag{3.43}$$

It is clear from the previous equation that to estimate the Euclidean Homography matrix $\boldsymbol{H}_i^f$ it is just necessary to determine the residual scaling factor λ. It can be demonstrated

that for a matrix of the form $\tilde{\boldsymbol{H}}_i^f$ we have:

$$|\lambda| \;=\; \sigma_2\,(\tilde{\boldsymbol{H}}_i^f) \quad , \tag{3.44}$$

where $\sigma_2\,(\tilde{\boldsymbol{H}}_i^f) \in \mathbb{R}$ is the second largest singular value of $\tilde{\boldsymbol{H}}_i^f$. To prove the previous equation let us define the following vector:

$$\boldsymbol{u} \;=\; \frac{1}{d_{\pi\,i}}\boldsymbol{R}_i^{f\,T}\boldsymbol{t}_i^f \;=\; -\frac{1}{d_{\pi\,i}}\,\boldsymbol{t}_f^i \quad \in \mathbb{R}^{3\times 1} \quad ; \tag{3.45}$$

then, recalling that $\|\boldsymbol{u}\|^2 \;=\; \boldsymbol{u}^T\boldsymbol{u}$, with simple passages we have:

$$\tilde{\boldsymbol{H}}_i^{f\,T}\tilde{\boldsymbol{H}}_i^f \;=\; \lambda^2\left(\boldsymbol{I} + \boldsymbol{u}\boldsymbol{n}_i^T + \boldsymbol{n}_i\boldsymbol{u}^T + \|\boldsymbol{u}\|^2\,\boldsymbol{n}_i\boldsymbol{n}_i^T\right) \quad . \tag{3.46}$$

Notice that the vector $(\boldsymbol{u}\wedge\boldsymbol{n}_i) \in \mathbb{R}^{3\times 1}$, which is orthogonal to both $\boldsymbol{u}$ and $\boldsymbol{n}_i$, is an *eigenvector* of $\tilde{\boldsymbol{H}}_i^{f\,T}\tilde{\boldsymbol{H}}_i^f$ with *eigenvalue* λ^2 that is:

$$\tilde{\boldsymbol{H}}_i^{f\,T}\tilde{\boldsymbol{H}}_i^f\,(\boldsymbol{u}\wedge\boldsymbol{n}_i) \;=\; \lambda^2(\boldsymbol{u}\wedge\boldsymbol{n}_i) \quad . \tag{3.47}$$

Hence $|\lambda|$ is a singular value of $\tilde{\boldsymbol{H}}_i^f$. We only have to show that it is the second largest. Let be:

$$\boldsymbol{v} \;=\; \|\boldsymbol{u}\|\,\boldsymbol{n}_i \quad , \quad \boldsymbol{w} \;=\; \frac{\boldsymbol{u}}{\|\boldsymbol{u}\|} \quad \in \mathbb{R}^{3\times 1} \quad ; \tag{3.48}$$

it is easy to prove that:

$$\boldsymbol{Q} \;=\; \boldsymbol{u}\boldsymbol{n}_i^T + \boldsymbol{n}_i\boldsymbol{u}^T + \|\boldsymbol{u}\|^2\,\boldsymbol{n}_i\boldsymbol{n}_i^T \;=\; (\boldsymbol{w}+\boldsymbol{v})\,(\boldsymbol{w}-\boldsymbol{v})^T - \boldsymbol{w}\boldsymbol{w}^T \quad . \tag{3.49}$$

The matrix $\boldsymbol{Q}$ has a positive, negative and a 0 eigenvalue (except that when $\boldsymbol{u} \propto \boldsymbol{n}_i$, $\boldsymbol{Q}$ will have two repeated 0 eigenvalue). In any case, $\tilde{\boldsymbol{H}}_i^{f\,T}\tilde{\boldsymbol{H}}_i^f$ has λ^2 as its second-largest eigenvalue. Then if $\{\sigma_1,\,\sigma_2,\,\sigma_3\}$ are the singular values of $\tilde{\boldsymbol{H}}_i^f$ recovered from linear least-square estimation, we can compute the Euclidean Homography as:

$$\boldsymbol{H}_i^f \;=\; \frac{\tilde{\boldsymbol{H}}_i^f}{\sigma_2\,(\tilde{\boldsymbol{H}}_i^f)} \quad . \tag{3.50}$$

This recovers $\boldsymbol{H}_i^f$ up to the form $\boldsymbol{H}_i^f \;=\; \pm\,(\boldsymbol{R}_i^f + \boldsymbol{t}\boldsymbol{s}_i^f\boldsymbol{n}_i^T)$. To get the right sign, remembering equation (3.16), it is sufficient to choose the solution that ensures the following constraint:

$$\boldsymbol{m}_f^{i\,T}\left(\boldsymbol{H}_i^f\boldsymbol{m}_f^i\right) \;>\; 0 \quad \forall\, i,...,n \quad , \tag{3.51}$$

where n is the number of point belonging to the plane π. Condition (3.51) is equivalent to impose the positiveness of the depths Z_i for all the points $\in \pi$.
Thus, the Euclidean Homography $\boldsymbol{H}_i^f$ between I_i and I_f can be uniquely determined from the image pixel points coordinates and their correspondences.

3.3.3 Euclidean Homography Decomposition

In the previous section we have recovered the Euclidean Homography $\boldsymbol{H}_i^f$ in the following form:

$$\boldsymbol{H}_i^f \;=\; \left[\boldsymbol{R}_i^f + \boldsymbol{t}_i^f \left(\frac{\boldsymbol{n}_i^T}{d_{\pi\, i}}\right)\right] \;=\; \left(\boldsymbol{R}_i^f + \boldsymbol{t}\boldsymbol{s}_i^f \boldsymbol{n}_i^T\right) \quad , \tag{3.52}$$

with $\boldsymbol{t}\boldsymbol{s}_i^f = (\boldsymbol{t}_i^f \,/\, d_{\pi\, i})$. Now, to solve the SFM problem, we have to study how to decompose $\boldsymbol{H}_i^f$ into its motion and structure parameters:

$$\left\{\boldsymbol{R}_i^f,\, \boldsymbol{t}\boldsymbol{s}_i^f,\, \boldsymbol{n}_i\right\} \quad . \tag{3.53}$$

Theorem 3.3.3.1. *There are at most two physically possible solutions for the decomposition of* $\boldsymbol{H}_i^f$ *into* $\left\{\boldsymbol{R}_i^f,\, \boldsymbol{t}\boldsymbol{s}_i^f,\, \boldsymbol{n}_i\right\}$ *rising from four analytically existing solutions.*

To demonstrate the above statement, it is useful to point out some key aspects:

- The Euclidean Homography $\boldsymbol{H}_i^f$ is a non-singular matrix.
- Since rotation matrices preserves the norms of the rotated vectors, from (3.52) we can state that $\boldsymbol{H}_i^f$ *preserves the length of any vector* $\boldsymbol{a} \in \mathbb{R}^{3\times 1}$ *orthogonal to* $\boldsymbol{n}_i$, namely:

$$\forall\, \boldsymbol{a} \perp \boldsymbol{n}_i \quad \boldsymbol{H}_i^f \boldsymbol{a} = \boldsymbol{R}_i^f \boldsymbol{a} \quad \Rightarrow \quad \left\|\boldsymbol{H}_i^f \boldsymbol{a}\right\| = \left\|\boldsymbol{R}_i^f \boldsymbol{a}\right\| = \|\boldsymbol{a}\| \quad . \tag{3.54}$$

- When the plane spanned by the vectors that are orthogonal to $\boldsymbol{n}_i$ is known, then, as a consequence also two solutions for $\boldsymbol{n}_i$ are known (the two reversed plane normal).

Let us first recover all the possible solution for $\boldsymbol{n}_i$ keeping in mind the previous aspects. The symmetric matrix $\boldsymbol{H}_i^{f\,T}\boldsymbol{H}_i^f$ is *definite-positive* for definition: it will then have three real distinct non-zero eigenvalues $\sigma_1^2 \geq \sigma_2^2 \geq \sigma_3^2 \geq 0$; from subsection (3.3.2) we know also that $\sigma_2 = 1$ so that also $\sigma_2^2 = 1$. $\boldsymbol{H}_i^{f\,T}\boldsymbol{H}_i^f$ can thus be diagonalized by the orthonormal eigenvector matrix $\boldsymbol{V} \in SO(3)$ such that:

$$\boldsymbol{H}_i^{f\,T}\boldsymbol{H}_i^f \;=\; \boldsymbol{V}\boldsymbol{\Delta}\boldsymbol{V}^T \quad , \tag{3.55}$$

where $\boldsymbol{\Delta} = diag\,\{\sigma_1^2,\, \sigma_2^2,\, \sigma_3^2\}$ is the diagonal eigenvalues matrix and $\boldsymbol{V} = [\boldsymbol{v}_1\, \boldsymbol{v}_2, \boldsymbol{v}_3]$. Moreover recalling the result obtained in equation (3.47) we have:

$$\boldsymbol{v}_2 \;=\; (\boldsymbol{u} \wedge \boldsymbol{n}_i) \;=\; (-\frac{1}{d_{\pi\, i}}\, \boldsymbol{t}_f^i \wedge \boldsymbol{n}_i) \quad , \tag{3.56}$$

From the eigenvector definition we have:

$$\boldsymbol{H}_i^{f\,T}\boldsymbol{H}_i^f \boldsymbol{v}_1 \;=\; \sigma_1^2 \boldsymbol{v}_1\,, \quad \boldsymbol{H}_i^{f\,T}\boldsymbol{H}_i^f \boldsymbol{v}_2 \;=\; \boldsymbol{v}_2\,, \quad \boldsymbol{H}_i^{f\,T}\boldsymbol{H}_i^f \boldsymbol{v}_3 \;=\; \sigma_3^2 \boldsymbol{v}_3\,. \tag{3.57}$$

Hence the unit eigenvector $\boldsymbol{v}_2$ is orthogonal both to $\boldsymbol{t}_f^i$ and $\boldsymbol{n}_i$: its length is thus preserved under the mapping $\boldsymbol{H}_i^f$. Let us define the other two vectors:

$$\boldsymbol{u}_1 \;=\; \frac{\sqrt{1-\sigma_3^2}\boldsymbol{v}_1 \;+\; \sqrt{\sigma_2^2-1}\boldsymbol{v}_3}{\sqrt{\sigma_1^2-\sigma_3^2}}\,, \quad \boldsymbol{u}_2 \;=\; \frac{\sqrt{1-\sigma_3^2}\boldsymbol{v}_1 \;-\; \sqrt{\sigma_2^2-1}\boldsymbol{v}_3}{\sqrt{\sigma_1^2-\sigma_3^2}}\,. \tag{3.58}$$

By exploiting the following properties

$$\begin{cases} \boldsymbol{v}_i^T \boldsymbol{v}_j \;=\; 0 \\ \boldsymbol{v}_i^T \boldsymbol{v}_i \;=\; 1 \\ \|\bullet\|^2 \;=\; [\bullet]^T\,[\bullet] \quad , \end{cases} \tag{3.59}$$

with $i \neq j;\ i,j \;=\; (1,2,3)$, it is easy to verify that both $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are unit vectors and their lengths is preserved through the map $\boldsymbol{H}_i^f$. Notice that $\boldsymbol{v}_2$ is orthogonal both to $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ since the last two are linear combinations of $\boldsymbol{v}_1$ and $\boldsymbol{v}_3$. Furthermore it is easy to verify that $\boldsymbol{H}_i^f$ preserves the length of any vector belonging to the two subspaces defined by the following basis:

$$S_1 \;=\; span\,\{\boldsymbol{v}_2,\, \boldsymbol{u}_1\}\,, \quad S_2 \;=\; span\,\{\boldsymbol{v}_2,\, \boldsymbol{u}_2\}\,. \tag{3.60}$$

To do this is sufficient to prove these two equivalence:

$$\begin{aligned} \left\| \boldsymbol{H}_i^f(\lambda_2 \boldsymbol{v}_2 \,+ \lambda_1 \boldsymbol{u}_1) \right\|^2 \;&=\; \|\lambda_2 \boldsymbol{v}_2 \,+ \lambda_1 \boldsymbol{u}_1\|^2 \quad , \\ \left\| \boldsymbol{H}_i^f(\lambda_2 \boldsymbol{v}_2 \,+ \lambda_3 \boldsymbol{u}_2) \right\|^2 \;&=\; \|\lambda_2 \boldsymbol{v}_2 \,+ \lambda_3 \boldsymbol{u}_2\|^2 \quad , \end{aligned} \tag{3.61}$$

with $\forall\ \lambda_1,\ \lambda_1,\ \lambda_3 \;\in \mathbb{R}$. Now we have found two planes, one defined by S_1 and another one

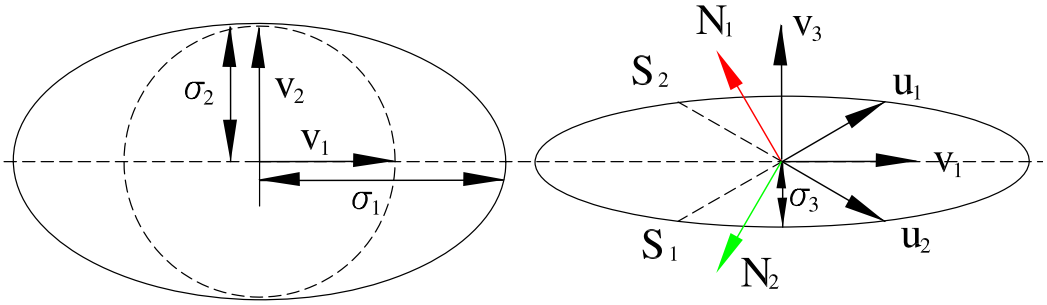


Figure 3.3: In terms of singular vectors ($\boldsymbol{v}_1$, $\boldsymbol{v}_2$, $\boldsymbol{v}_3$) and singular values ($\boldsymbol{\sigma}_1$, $\boldsymbol{\sigma}_2$, $\boldsymbol{\sigma}_3$) of the matrix $\boldsymbol{H}_i^f$, there are two candidates subspaces S_1 and S_2 where the vectors length is preserved by the homography $\boldsymbol{H}_i^f$.

defined by S_2, whose vectors preserve their magnitudes through the map $\boldsymbol{H}_i^f$: therefore, according the previous pointed out aspects, all the admissible candidates normals for $\boldsymbol{n}_i$ are defined by these four unit vectors, that are orthogonal both to S_1 and S_2 as shown in figure (3.3):

$$\pm\,(\boldsymbol{v}_2 \wedge \boldsymbol{u}_1) \quad , \quad \pm(\boldsymbol{v}_2 \wedge \boldsymbol{u}_2) \quad . \tag{3.62}$$

In this case, since any vectors orthogonal $\boldsymbol{n}_i$ preserves its norm through $\boldsymbol{H}_i^f$, the following

relations can be written:

$$\begin{gathered} \boldsymbol{H}_i^f \boldsymbol{v}_2 \;=\; \boldsymbol{R}_i^f \boldsymbol{v}_2; \qquad\qquad \boldsymbol{H}_i^f \boldsymbol{u}_i \;=\; \boldsymbol{R}_i^f \boldsymbol{u}_i; \\ (\boldsymbol{H}_i^f \boldsymbol{v}_2) \wedge (\boldsymbol{H}_i^f \boldsymbol{u}_i) \;=\; (\boldsymbol{R}_i^f \boldsymbol{v}_2) \wedge (\boldsymbol{R}_i^f \boldsymbol{u}_i) \;=\; \boldsymbol{R}_i^f (\boldsymbol{v}_2 \wedge \boldsymbol{u}_i) \quad , \end{gathered} \tag{3.63}$$

where $i \;=\; 1, 2$. By exploiting the relations (3.63), we can build up four orthonormal bases of $\mathbb{R}^3$:

$$\begin{aligned} \boldsymbol{U}_1 \;&=\; [\boldsymbol{v}_2,\, \boldsymbol{u}_1,\, (\boldsymbol{v}_2 \wedge \boldsymbol{u}_1)]\,, \qquad \boldsymbol{W}_1 \;=\; \left[\boldsymbol{H}_i^f \boldsymbol{v}_2,\, \boldsymbol{H}_i^f \boldsymbol{u}_1,\, (\boldsymbol{H}_i^f \boldsymbol{v}_2) \wedge (\boldsymbol{H}_i^f \boldsymbol{u}_1)\right]; \\ \boldsymbol{U}_2 \;&=\; [\boldsymbol{v}_2,\, \boldsymbol{u}_2,\, (\boldsymbol{v}_2 \wedge \boldsymbol{u}_2)]\,, \qquad \boldsymbol{W}_2 \;=\; \left[\boldsymbol{H}_i^f \boldsymbol{v}_2,\, \boldsymbol{H}_i^f \boldsymbol{u}_2,\, (\boldsymbol{H}_i^f \boldsymbol{v}_2) \wedge (\boldsymbol{H}_i^f \boldsymbol{u}_2)\right]. \end{aligned} \tag{3.64}$$

Notice that they are respectively two by two rotated by the matrix $\boldsymbol{R}_i^f$:

$$\boldsymbol{R}_i^f \boldsymbol{U}_1 \;=\; \boldsymbol{W}_1, \qquad \boldsymbol{R}_i^f \boldsymbol{U}_2 \;=\; \boldsymbol{W}_2. \tag{3.65}$$

By inverting equations (3.65) we obtain two candidates for the rotation $\boldsymbol{R}_i^f$ that finally splits into four analytically admissible solution for the Structure from Motions problem (due to the four $\boldsymbol{n}_i$ candidates of equations-3.62). The four solutions for $\boldsymbol{R}_i^f$ and $\boldsymbol{n}_i$ are given by row in table (3.1). In all solutions the scaled translation $\boldsymbol{t}_i^f/(d_{\pi\, i})$ is finally

Solution	$\boldsymbol{R}_i^f$	$\boldsymbol{n}_i$
1	$\boldsymbol{W}_1 \boldsymbol{U}_1^T$	$(\boldsymbol{v}_2 \wedge \boldsymbol{u}_1)$
2	$\boldsymbol{W}_1 \boldsymbol{U}_1^T$	$-(\boldsymbol{v}_2 \wedge \boldsymbol{u}_1)$
3	$\boldsymbol{W}_2 \boldsymbol{U}_2^T$	$(\boldsymbol{v}_2 \wedge \boldsymbol{u}_2)$
4	$\boldsymbol{W}_2 \boldsymbol{U}_2^T$	$-(\boldsymbol{v}_2 \wedge \boldsymbol{u}_2)$

Table 3.1: The four analytically admissible solutions of the Euclidean Homography decomposition.

calculated by equation (3.52) with the following equation:

$$\boldsymbol{t}_i^f/(d_{\pi\, i}) \;=\; (\boldsymbol{H}_i^f - \boldsymbol{R}_i^f)\boldsymbol{n}_i \quad . \tag{3.66}$$

In order to reduce the number of physically possible solutions, we can exploit the *positive depth constraint*. Since the camera is able to visualize only $3D$ target points that are in

front of it, a physically possible solution must respect the following conditions:

$$\begin{cases} \boldsymbol{n}_i^T \boldsymbol{k}_i = n_{i3} > 0 \quad , \\ \boldsymbol{n}_f^T \boldsymbol{k}_f = n_{f3} > 0 \quad , \end{cases} \tag{3.67}$$

where $\boldsymbol{n}_f = \boldsymbol{R}_i^f \boldsymbol{n}_i$ is the normal of π expressed in the camera frame C_f and $\boldsymbol{k}_i$, $\boldsymbol{k}_f$ are respectively the camera focal axis of C_i and C_f both two expressed by the unit vector [0, 0, 1]. By imposing conditions (3.67) we can discard two solution from the four of table (3.1), since their normal directions are two by two reversed. *We finally have at most two physically admissible solutions for the decomposition problem of (3.52)-(3.53).*
To select the true solution from the remaining two physically possible we proceed as follows:

- If we have a $3D$ target and we have used the general algorithm of section (3.3.1.1), to estimate the pixel homography $\boldsymbol{G}_i^f$, it is sufficient to repeat all the SFM procedure by choosing a different virtual plane π; we finally compare the four solutions arising from the two SFM runs (two plus two physically possible) and choose the solution with the same rotation $\boldsymbol{R}_i^f$ in both runs.

- If we deal with a planar target and we have thus used the algorithm of section (3.3.1.2), to estimate the pixel homography $\boldsymbol{G}_i^f$, it is necessary to provide a third *auxiliary target view* I_a to find for the real solution of SFM decomposition. Also in this case it is necessary to repeat the SFM procedure a second time by using I_i and I_a views and finally choose from the four arisen solutions in the two runs the one with the same π normal $\boldsymbol{n}_i$.

We finally are able to estimate the structure and motion parameters (3.17) among the camera frames C_i, C_f and the plane π through the target image points coordinates and their correspondences, with a calibrated camera $\boldsymbol{K}$.
Notice that, after the SFM solution, the only unknown in equation (3.52) remains the distance $d_{\pi\, i}$ of C_i origin from the plane π. This is a consequence of the fact that image informations can provide the structure of the $3D$ scene but not its real dimension: retrieving the factor $d_{\pi\, i}$ would be as to determine the $3D$ target and scene dimensions. This concept is summarized in the literature by saying that SFM solution provides a *Scaled Euclidean Reconstruction (SER)* of the scene.

Chapter 4

Visual Servoing

Nowadays the choice of vision sensors to control in real-time complex robotic systems has become a common use, thanks to the increasing performances of both modern computers and hardware components. Various techniques have been proposed to control the behavior of a dynamic system exploiting the visual information provided by one ore more cameras inside a feedback control-loop: these approaches are known in literature as Visual Servoing [HHC96].
Visual Servoing concerns several field of research including vision systems, robotics and automation control: it can be a useful solution for a wide range of applications in the control many different dynamics systems like manipulators arms, mobile robots aircraft, computer aided surgery etc.
This chapter describe how a robotic 6 DOF system provided with a single camera mounted on its end-effector can be on-line controlled by exploiting only the visual information available at frame-rate from the camera video stream. The chapter is organized as follows: in the first section an overview of the main visual servoing approaches is given while the second is focused in *Image Based Visual Servoing*; the third section instead addresses the problem of image path planning generation to increase the performances and the stability of the image based control approach.

4.1 An Overview of Visual Servoing Approaches

As pointed out above, Visual Servoing exploits the visual informations provided from one or more cameras in a feedback control loop to drive the controlled dynamic system in a goal configuration. In particular, using of Visual Servoing, a robot can be positioned with respect to a target placed in its workspace, by minimizing the differences between the current target view I_c (correspondent to the current robot configuration) and the goal or desired camera view I_f (correspondent to the desired robot configuration). This specific control task is called in literature *Positioning Task*.
Visually controlled systems may differ in the control architecture , in the number of camera used (mono, stereo or multi-camera configurations) and in their placing in the system. When the cameras are attached to the ground frame, pointed to the manipulator, the vision system is called *Eye to Hand* while if the cameras are mounted on the robot end effector the system is called *Eye in hand*. In this thesis we will refer in particular to a mono-camera system in eye in hand configuration.

A fundamental classification of visual servoing approaches depends on the design of the control architecture: two are the the main adopted choices. The first control architecture is called *"direct visual servoing"* since the vision-based controller directly computes the input of the dynamic system (i.e. the robot actuators). The visual servoing is carried out at a very fast control updating frequency (at least 100 Hz, with rate of 10 ms): this approach gives good performances in terms of system time-response but can be used only when expensive high frame rate camera are available and when low time consuming algorithm can be used for image processing (only with very simple visual features). The second control scheme can be called, contrary to the first one, *"indirect visual servoing"* since the vision-based control computes a reference control law which is sent to a second a low level controller of the dynamic system (inner control loop), that is usually the standard one provided by the vendor of the manipulator. Most of the visual servoing proposed in the literature follows an indirect control scheme which is called *"dynamic look-and-move"*. In this case the servoing of the inner control loop (generally the rate is 10 ms) must be faster than the visual servoing one which speed is limited by the camera frame rate (usually the maximum rate is limited to 50 ms).

4.1.1 Key control steps

Consider to have a $6DOF$ manipulator with a monocular eye in hand visual system and let be $\boldsymbol{q} = [q_1, q_2, ..., q_6]^T \in \mathbb{R}^{n\times 1}$ the joint variables vector defining the manipulator configuration in the *joints space*. Let us define *kinematic pose* of a coordinate system C the (6×1) vector

$$\boldsymbol{r} = \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{\phi} \end{bmatrix} \quad , \tag{4.1}$$

where $\boldsymbol{p} = [p_x, p_y, p_z]^T$ is the coordinate vector of C origin and $\boldsymbol{\phi} = [\phi, \theta, \psi]^T$ is a minimal representation of C attitude (i.e. the Euler angles or the RPY representation) all being expressed with respect to a reference frame W attached to the ground. Notice moreover that the robot end-effector pose $\boldsymbol{r}_e$ can be expressed through the joint variables vector $\boldsymbol{q}$ by the *manipulator direct kinematic function*:

$$\boldsymbol{r} = \boldsymbol{k}(\boldsymbol{q}) \quad . \tag{4.2}$$

The function $\boldsymbol{k}$ can be computed by knowing the geometrical manipulator parameters (i.e the Denavit-Hartenberg parameters) after the execution of the *robot kinematic calibration*. Let be $\boldsymbol{M}_e^c$ the 4×4 constant homogeneous matrix defining the the camera frame with respect to the robot end effector frame (the camera is fixed with respect to the end effector):

$$\boldsymbol{M}_c^e = \begin{bmatrix} \boldsymbol{R}_c^e & \boldsymbol{t}_c^e \\ \boldsymbol{0}^T & 1 \end{bmatrix} \quad , \tag{4.3}$$

where $\boldsymbol{R}_c^e \in \mathbb{R}^{3\times 3}$ and $\boldsymbol{t}_c^e \in \mathbb{R}^{3\times 1}$ are respectively the rotation matrix and position vector of the camera frame with respect to the end effector frame. Notice that the camera pose $\boldsymbol{r}_c$ is completely defined if both $\boldsymbol{r}_e$ and $\boldsymbol{M}_c^e$ are known. Various calibration techniques have been developed to estimate $\boldsymbol{M}_c^e$ with an already calibrated camera intrinsics $\boldsymbol{K}$ [AHE01].

In order to execute a positioning task with a visual servoing system, many steps have to be performed; here we summarize the most important:

- *Goal View Acquisition* - First a goal target view I_f corresponding to a goal manipulator configuration $\boldsymbol{r}_f$ (and to a goal camera pose C_f) has to be stored in memory: in general in fact the robot will be in an initial configuration $\boldsymbol{r}_i$ corresponding to the camera frame C_i providing the target initial image I_i different from the goal configuration.

- *Features Extraction* - A set of well detectable *visual features* (or *image features*) $\boldsymbol{f}$ has to be chosen both from I_i and I_f. A visual feature can be defined by various image primitives like points, lines, ellipses or more complex contours. Several techniques of Image Processing can be used in the extraction process like i.e. the Harris detector for points or the Canny algorithm in case of line or more complex contours. In spite of a complex automatic process, in this work to the user is able to choose the feature in both images by a mouse-pick selection. The number of chosen visual features has to be sufficient to control all the 6 DOF of the manipulator (see later).

- *Feature Matching* - the problem of find the corresponding image features between I_i and I_f (i.e. the image of the same $3D$ points, lines, circles, etc.. in both images) is known as matching problem; also in this case it is particularly difficult to automate the matching process (the problem is not yet solved) especially when we have a large camera displacement between C_i and C_f thus providing consistent image differences between I_i and I_f. Also the matching problem has been overcome in this work by considering the selection order during the mouse-pick procedure of the feature selection in I_i and I_f (the selection order identifies the feature correspondences).

- *System Error Definition* - when the previous step are executed it is necessary to define the system error vector $\boldsymbol{e}(\boldsymbol{f}) \in \mathbb{R}^{n\times 1}$ as function of the chosen visual features $\boldsymbol{f}$ with $n \geq 6$. The control system will have to minimize $\boldsymbol{e}$ by providing opportune references to the manipulator actuators in order to drive the robot to the goal pose corresponding to the goal target view I_f.

- *Features Tracking* - During the on-line positioning phase, at each time-step of the visual controller, the image features have to be tracked in the actual image in order to refresh their coordinates. Several tacking techniques exist to perform feature tracking: for points feature the most largely used is the Lucas-Kanade optical flow estimation algorithm. Features tracking is *local* operation in the image to be done on-line (a local matching in a square pixel window between the i and the $(i-1)$ images): it must be as less time-consuming as possible in order to ensure good performances of the control. Feature tracking in fact represents the most computational demanding phase of all the other the on-line control steps.

From the control scheme point of view, Visual Servoing approaches can be subdivided into three different main groups, depending on the type of error function $\boldsymbol{e}$ that is adopted to regulate the system: Position Based Visual Servoing (PBVS), Image based Visual Servoing (IBVS) and Hybrid Visual Servoing.

4.1.2 PBVS

In Position Based Visual Servoing the error is computed by estimating from images some 3D features of interest (like the desired camera rotation and translation etc...): this approach allows the user to fully define the robot trajectory in 3D space but has proved to be very sensitive with respect to both camera and robot calibration errors and usually requires the knowledge of the $3D$ target model [HHC96].

4.1.3 IBVS

In Image based Visual Servoing, on the other hand, the feedback control loop is directly closed in the image since the system error is defined by primitives directly extracted from images (such as 2D points, lines, or more complex shapes [ECR92]).
Image based approach in general does not require a priori knowledge of the target (model-free technique) and is known to be really robust with respect to both camera and robot calibration errors [Esp93]. On the other hand only local stability of the controlled system has been proved: that is why, during task executions, singularities and local minima of the control law may occur especially when the initial and the desired view of the target are noticeably different (large camera displacements) [Cha99]. To overcome singularity and local minima problems a convenient path planning in the image can be exploited. Hereafter we summarize the state of the art concerning the various image based visual servoing techniques: each technique has different strength points and drawbacks and thus is more suitable for certain applications.

4.1.3.1 Invariant IBVS

Visual servoing methods typically require the a priori knowledge of internal calibration data. A particular research trend is that of the development of strategies that relax this requirement, thus allowing to work with uncalibrated settings. In the approach proposed in [Mal04], image features are mapped onto a special projective space which is invariant to changes in camera intrinsics. Although the control error is invariant with respect to camera intrinsics, these parameters are still required to estimate the Jacobian; for this reason, calibration errors may affect the stability of the control law. A second way to avoid off-line calibration is to rely on self-calibration techniques developed by the computer vision community, and exploit the (partial) knowledge of scene structure and/or camera parameters to perform on-line camera calibration—see e.g. [HZ03]. The main limitation of self-calibration approaches is that they require a priori information that may not be available.

4.1.3.2 Weighted features IBVS

To cope with target occlusions and problems like features disappearing also for the limited camera field of view a particular technique of image based visual servoing has been developed: this strategy uses a weight function in the control law for each controlled feature [GAMO05]. This technique has also been tested with the invariant approach above mentioned.

4.1.3.3 Homography Based IBVS

A novel image based technique called Homography Based Visual Servoing have also to be mentioned: this technique uses directly the planar Euclidean homography to define the system error and results locally stable [BM06]. Homography based Visual Servoing, differently from the other IBVS strategies, does not need any feature depth estimate in the control scheme since is based on a $2D$ innovative control law.

4.1.4 Hybrid Visual Servoing

The hybrid visual servoing scheme, also known as $2-1/2-D$ Visual Servoing, adopts a system error that is defined both with image and with Cartesian primitives, mixing together different types of features: this method ensures good analytical stability properties (sufficient conditions for global stability has been proved) [MCB99]; however also in this approaches 3D robot trajectory is not predictable and the target may go out of the camera field of view, causing the loss of the visual informations (the visual features) and thus the failure of the task.

4.2 Control

In this Section we will show the control strategy of Visual Servoing systems showing in detail the Image Based approach. First of all let us recall some notions of differential kinematics that will be useful in the following of the work. Let thus be I, J and K three different reference frames and let be $\boldsymbol{P}$ a $3D$ point attached to I. The *twist screw* of I with respect to J expressed in K with center in $\boldsymbol{P}$ (attached to I) is defined by the following (6×1) vector:

$$ {}^{P}_{k}\boldsymbol{w}^{i}_{j} = \begin{bmatrix} {}^{P}_{k}\boldsymbol{v}^{i}_{j} \\ {}^{P}_{k}\boldsymbol{\omega}^{i}_{j} \end{bmatrix} \quad , \tag{4.4} $$

where ${}^{P}_{k}\boldsymbol{v}^{i}_{j} =, [v_x,\, v_y,\, v_z]^T$ is the velocity of the point $\boldsymbol{P}$ with respect to J expressed in K and ${}^{P}_{k}\boldsymbol{\omega}^{i}_{j} = [\omega_x,\, \omega_y,\, \omega_z]^T$ is the instantaneous rotational velocity (angular velocity) of I with respect to J expressed in K as well. The twist screw ${}^{P}_{k}\boldsymbol{w}^{i}_{j}$, also called velocity screw, defines completely the differential cinematic motion of I with respect to J. Notice that usually is convenient to choose as point of the twist screw $\boldsymbol{P}$ the origin of I itself: in this case the notation can be simplified to:

$$ \boldsymbol{P} \equiv \boldsymbol{O}_I \quad \rightarrow \quad {}^{P}_{k}\boldsymbol{w}^{i}_{j} = {}_{k}\boldsymbol{w}^{i}_{j} \quad ; \tag{4.5} $$

in this case the ${}^{P}_{k}\boldsymbol{v}^{i}_{j} = {}_{k}\boldsymbol{v}^{i}_{j}$ is equal to the translational velocity of I with respect to J. To simply express the twist screw ${}_{k}\boldsymbol{w}^{i}_{j}$ with respect to another frame L it is sufficient to exploit the rotation matrix $\boldsymbol{R}^{l}_{k}$ having as columns the K unit vectors expressed with respect to L:

$$ {}_{l}\boldsymbol{w}^{i}_{j} = \begin{bmatrix} \boldsymbol{R}^{l}_{k} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}^{l}_{k} \end{bmatrix} = {}^{l}_{k}\boldsymbol{W}\ {}_{k}\boldsymbol{w}^{i}_{j} \quad . \tag{4.6} $$

Let us define another frame M fixed with respect to I and suppose to know the twist screw ${}_i\boldsymbol{w}_j^i$ expressed in I. if we want to evaluate the twist screw M with respect to J expressed in M (changing both the frame for twist computation and the frame in which the screw is expressed), it can be shown that the following equation must be used:

$$ {}_m\boldsymbol{w}_j^m = \begin{bmatrix} \boldsymbol{R}_i^m & [\boldsymbol{t}_i^m]_\times \boldsymbol{R}_i^m \\ \boldsymbol{0} & \boldsymbol{R}_i^m \end{bmatrix} {}_i\boldsymbol{w}_j^i = {}_i^m\boldsymbol{V}\ {}_i\boldsymbol{w}_j^i \quad , \tag{4.7} $$

where $[\quad]_\times$ denotes the skew symmetric matrix operator applied to the (3×1) translation vector $\boldsymbol{t}_i^m$. The skew symmetric operator is finally defined as follow:

$$ [\boldsymbol{u}]_\times = \begin{bmatrix} 0 & -u_3 & u_2 \\ u_3 & 0 & -u_1 \\ -u_2 & u_1 & 0 \end{bmatrix} \quad , \tag{4.8} $$

where $\boldsymbol{u} = [u_1,\, u_2,\, u_3]^T$.

4.2.1 The Interaction Matrix

Let us choose now a set of adequate target visual features $\boldsymbol{f}$ to control the $6\,DOF$ of the manipulator. Let us built up from $\boldsymbol{f}$ the *system state vector* $\boldsymbol{x} \in \mathbb{R}^{m\times 1}$ with $m \geq 6$. For the Image Based Visual Servoing addressed in this work $\boldsymbol{x}$ will be directly formed with the extracted image features. Since we assumed to deal with a *fixed target*, the visual feature $\boldsymbol{f}$ and consequently the state vector $\boldsymbol{x}$ are only function of the time varying camera pose $\boldsymbol{r}_c(t)$, according to the definition (4.1):

$$ \boldsymbol{x} = \boldsymbol{x}(\boldsymbol{r}_c(t)) = \boldsymbol{x}(\boldsymbol{p}_c(t)\,, \boldsymbol{\phi}_c(t)) \quad , \tag{4.9} $$

Let be $\boldsymbol{w}$ the (6×1) camera twist screw with respect to the ground frame W, expressed in the camera frame C and centered in the camera origin $\boldsymbol{O}_c$:

$$ \boldsymbol{w} \doteq {}_c\boldsymbol{w}_w^c = [\boldsymbol{v}^T,\, \boldsymbol{\omega}^T\,]^T = [v_x,\, v_y,\, v_z,\, \omega_x,\, \omega_y,\, \omega_z]^T \quad ; \tag{4.10} $$

by deriving the equation (4.9), we obtain the following differential kinematic relation:

$$ \dot{\boldsymbol{x}} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{r}_c}\frac{\partial \boldsymbol{r}_c}{\partial t} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{r}_c}\dot{\boldsymbol{r}}_c = \boldsymbol{J}(\boldsymbol{x}, r_c(t))\,\boldsymbol{w} \quad . \tag{4.11} $$

The $(m\times 6)$ matrix $\boldsymbol{J}$ in the above equation is called *Interaction Matrix* or *Image Jacobian* and is function of the state system vector (visual features) and of the $3D$ camera pose. The Interaction Matrix represents a local linear mapping between feature velocity $\dot{\boldsymbol{x}}$ in the image and the camera twist-screw $\boldsymbol{w}$ in the Cartesian space. Notice that the analytic "Jacobian" $\left[\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{r}_c}\right] \in \mathbb{R}^{m\times 6}$ however results slightly different from $\boldsymbol{J}$ since the time derivative of the attitude angles $\dot{\boldsymbol{\phi}}_c$ is different from the instantaneous rotational velocity $\boldsymbol{\omega}$:

$$ \boldsymbol{\omega} = \boldsymbol{T}\,\dot{\boldsymbol{\phi}}_c \quad . \tag{4.12} $$

If the Euler angles are used as attitude minimal representation, the (3×3) transformation matrix $\boldsymbol{T}$ in equation (4.12) results:

$$\boldsymbol{T} = \begin{bmatrix} 0 & -\sin\phi & cos\phi \sin\theta \\ 0 & \cos\phi & \sin\phi \sin\theta \\ 1 & 0 & \cos\theta \end{bmatrix} \quad . \tag{4.13}$$

4.2.1.1 Points as Visual Features

Consider now as visual feature a normalized image point $\tilde{\boldsymbol{m}}_i$, as defined in equation (2.19) and let $\boldsymbol{P}_i = [X_i, Y_i, Z_i]^T$ the coordinates of the corresponding $3D$ target point expressed in the camera frame C. The point $\tilde{\boldsymbol{m}}_i$ can be easily obtained from the corresponding extracted image pixel point $\boldsymbol{p}_i$ by inversion of equation (2.27) since $\boldsymbol{K}$ is assumed to be known. Let be $\boldsymbol{m}_i$ the non-homogeneous point normalized coordinates:

$$\boldsymbol{m}_i = \begin{bmatrix} X_i/Z_i \\ Y_i/Z_i \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \quad . \tag{4.14}$$

The velocity of $\boldsymbol{P}_i$ with respect to the world frame W (absolute velocity) expressed from the relative camera frame C results:

$$\dot{\boldsymbol{P}}_{iA} = \dot{\boldsymbol{P}}_{iT} + \dot{\boldsymbol{P}}_{iC} \quad , \tag{4.15}$$

where $\dot{\boldsymbol{P}}_{iT}$ is called rigid body velocity and $\dot{\boldsymbol{P}}_{iC}$ is the point relative velocity seen from C. The rigid body point velocity can be expressed by:

$$\dot{\boldsymbol{P}}_{iT} = \boldsymbol{v} + [\boldsymbol{\omega}]_\times \boldsymbol{P_i} = \left[\boldsymbol{I} - [\boldsymbol{P}_i]_\times\right] \boldsymbol{w} \quad . \tag{4.16}$$

where $\boldsymbol{v}$ and $\boldsymbol{\omega}$ are respectively the translational and rotational components of the camera twist screw $\boldsymbol{w}$ defined in (4.10) and $\boldsymbol{I}$ the (3×3) identity matrix. The point relative velocity on the contrary results:

$$\dot{\boldsymbol{P}}_{iC} = \frac{\partial \boldsymbol{P}_i}{\partial t} = [\dot{X}_i, \dot{Y}_i, \dot{Z}_i]^T \quad . \tag{4.17}$$

Since we deal with a fixed target and moving camera $\dot{\boldsymbol{P}}_{iA} = \boldsymbol{0}$ and equation (4.15) becomes:

$$\dot{\boldsymbol{P}}_{iC} = -\dot{\boldsymbol{P}}_{iT} \quad , \tag{4.18}$$

or better, by injecting equation (4.16):

$$\dot{\boldsymbol{P}}_{iC} = \left[-\boldsymbol{I} \ [\boldsymbol{P}_i]_\times\right] \boldsymbol{w} \quad . \tag{4.19}$$

By computing the time derivative of the normalized image point $\boldsymbol{m}_i$ we have:

$$\dot{\boldsymbol{m}}_i = \begin{bmatrix} \dot{x}_i \\ \dot{y}_i \end{bmatrix} \dot{\boldsymbol{P}}_{iC} = \begin{bmatrix} 1/Z_i & 0 & -X_i/Z_i^2 \\ 0 & 1/Z_i & -Y_i/Z_i^2 \end{bmatrix} \dot{\boldsymbol{P}}_{iC} \quad . \tag{4.20}$$

Finally by injecting (4.19) in (4.20), we obtain the Interaction matrix for an normalized image point $\boldsymbol{m}_i$:

$$\dot{\boldsymbol{m}}_i = \boldsymbol{J}_i \boldsymbol{w} \;=\; \begin{bmatrix} -\frac{1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_i y_i & -(1+x_i^2) & y_i \\ 0 & -\frac{1}{Z_i} & \frac{y_i}{Z_i} & (1+y_i^2) & -x_i y_i & -x_i \end{bmatrix} \boldsymbol{w}\,. \tag{4.21}$$

Notice that $\boldsymbol{J}_i$ depends on the point image coordinates x_i, y_i and on the $3D$ point depth Z_i with respect to the camera frame C, varying with the camera pose:

$$\boldsymbol{J}_i \;=\; \boldsymbol{J}_i(\boldsymbol{m}_i(t), Z_i(t)) \quad . \tag{4.22}$$

While the image point coordinates can be directly measured from the actual image, the unknown depth Z_i will be on-line estimated in the control phase by exploiting the adaptive law described in [CA01]. In general if we use $n > (m/2)$ points as target visual features, namely to track at least 4 different image points, we can make explicit the system state vector introduced in (4.9) as:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{m}_1^T & \boldsymbol{m}_2^T & \dots & \boldsymbol{m}_n^T \end{bmatrix}^T . \tag{4.23}$$

From (4.21), stacking together the interaction matrices $\boldsymbol{J}_i$ for all the considered image points, we build up on the other side the global system image jacobian $\boldsymbol{J} \in \mathbb{R}^{2n\times 6}$:

$$\boldsymbol{J} = \begin{bmatrix} \boldsymbol{J}_1^T & \boldsymbol{J}_2^T & \dots & \boldsymbol{J}_n^T \end{bmatrix}^T . \tag{4.24}$$

By eploiting equations (4.21), (4.23) and (4.24), we finally obtain the differential kinematic equation of the system:

$$\dot{\boldsymbol{x}} = \boldsymbol{J}(\boldsymbol{x}, \boldsymbol{Z_x})\boldsymbol{w}. \tag{4.25}$$

4.2.2 System Dynamic

Let us recall now that the twist screw of the camera $\boldsymbol{w} = {}_c\boldsymbol{w}_w^c$ can be related to the twist screw of the robot end effector ${}_w\boldsymbol{w}_w^e$ expressed in W coordinates by the composition of the two transformation of equations (4.7) and (4.6):

$$\boldsymbol{w} \;=\; {}_c\boldsymbol{w}_w^c \;=\; {}_e^c\boldsymbol{V}\; {}_e\boldsymbol{w}_w^e \;=\; {}_e^c\boldsymbol{V}\; {}_w^e\boldsymbol{W}(\boldsymbol{q})\, {}_w\boldsymbol{w}_w^e \;=\; {}_e^c\boldsymbol{F}(\boldsymbol{q})\, {}_w\boldsymbol{w}_w^e \tag{4.26}$$

where ${}_w\boldsymbol{w}_w^e$ is the end effector screw with respect to W, expressed in W and centered in E origin. The overall change of screws matrix ${}_e^c\boldsymbol{F}(\boldsymbol{q})$ in (4.26) results:

$${}_e^c\boldsymbol{F}(\boldsymbol{q}) \;=\; {}_e^c\boldsymbol{V}\; {}_w^e\boldsymbol{W}(\boldsymbol{q}). \tag{4.27}$$

Notice that, since we deal with an eye-in-hand system, the matrix ${}_e^c\boldsymbol{V}$, built with $\boldsymbol{R}_e^c$ and $\boldsymbol{t}_e^c$, is constant (see also eq.(4.3)) while the matrix ${}_w^e\boldsymbol{W}(q)$, formed with the rotation matrix $\boldsymbol{R}_w^e(q)$, is function of the time varying robot configuration $\boldsymbol{q}(t)$. Finally the twist screw ${}_w\boldsymbol{w}_w^e$ can be related to the robot velocity in the joints space by the *Geometric Robot*

Jacobian:

$$ {}_{w}\boldsymbol{w}_{w}^{e} = \boldsymbol{J}_{G}(q)\dot{\boldsymbol{q}} \quad . \tag{4.28} $$

Joining together the three equations (4.11), (4.26) and (4.28), we obtain the total dynamic equation for the considered eye in hand system:

$$ \dot{\boldsymbol{x}} = \boldsymbol{J}\,{}_{e}^{c}\boldsymbol{F}\boldsymbol{J}_{G}\,\dot{\boldsymbol{q}} + \frac{\partial \boldsymbol{x}}{\partial t} \quad . \tag{4.29} $$

The term $\left(\frac{\partial \boldsymbol{x}}{\partial t}\right)$ in the above equation takes into account for target motions with respect to the ground frame W: $\left(\frac{\partial \boldsymbol{x}}{\partial t}\right)$ vanishes for motionless targets like the one considered in this work.

4.2.2.1 System Error

Making the hypotheses to have a controller that fully compensate for the manipulator dynamics (ideal controller), to have a perfect kinematic model of the robot and to remain far from kinematic manipulator singularities, we can choose as control input to regulate the visual system directly the twist camera screw $\boldsymbol{w}$ that can be rewritten according to (4.29) as:

$$ \boldsymbol{w} = {}_{c}\boldsymbol{w}_{w}^{c} = {}_{e}^{c}\boldsymbol{F}\boldsymbol{J}_{G}\,\dot{\boldsymbol{q}} \tag{4.30} $$

In this case we can define the system error as follow:

$$ \boldsymbol{e} = (\boldsymbol{x}_{d} - \boldsymbol{x}) \quad \in \mathbb{R}^{n\times 1}. \tag{4.31} $$

where $\boldsymbol{x}_{d}$ and $\boldsymbol{x}$ are the state vectors respectively in the reference and in the actual system configurations.
By using points as visual features the system error $\boldsymbol{e}$ corresponds to the difference between the reference and the actual normalized image points: $\boldsymbol{e}$ is thus fully defined in the image according to the image based visual servoing strategy of section (4.1.3).

4.2.2.2 Control Law

By inspection of (4.11), we can use a control law based on the image error $\boldsymbol{e}$ and on the left-pseudoinverse of the estimate interaction matrix $\hat{\boldsymbol{J}}^{\dagger}$:

$$ \boldsymbol{w} = \hat{\boldsymbol{J}}^{\dagger}(k\boldsymbol{e} + \dot{\boldsymbol{x}}_{d})\,, \tag{4.32} $$

where $\hat{\boldsymbol{J}}^{\dagger} = \left(\hat{\boldsymbol{J}}^{T}\hat{\boldsymbol{J}}\right)^{-1}\hat{\boldsymbol{J}}^{T}$ and k is a positive gain. Expression (4.32) is valid only when $\hat{\boldsymbol{J}}$ has full rank (there must be at least 6 linearly independent raws on $\boldsymbol{J}$). The control law (4.32) contains both an error proportional term and a feed-forward term. Notice that the vector $\dot{\boldsymbol{x}}_{d}$ in the feed forward term represents the time derivative of the reference state vector:

- if the IBVS control doesn't provide any image planning $\boldsymbol{x}_{d}$ corresponds to the goal view I_{f} features set $\Rightarrow \boldsymbol{x}_{d} = \boldsymbol{x}_{f} = cost$;
- when path planning is done $\boldsymbol{x}_{d} \neq cost$ corresponds with the planned image trajectories ranging from $\boldsymbol{x}_{i}$ in I_{i} to $\boldsymbol{x}_{f}$ in I_{f}.

4.2.3 Stability Conditions

To prove the stability of the system we use the following Lyapunov candidate:

$$V = \frac{1}{2}\boldsymbol{e}^T\boldsymbol{e}\,. \tag{4.33}$$

The time derivate of (4.33), by using (4.31),(4.32) and (4.11) results:

$$\begin{aligned}
\dot{V} &= \boldsymbol{e}^T\dot{\boldsymbol{e}} && (4.34)\\
\dot{V} &= \boldsymbol{e}^T(\dot{\boldsymbol{x}}_d - \dot{\boldsymbol{x}})\\
\dot{V} &= \boldsymbol{e}^T(\dot{\boldsymbol{x}}_d - \boldsymbol{J}\boldsymbol{w})\\
\dot{V} &= \boldsymbol{e}^T(\dot{\boldsymbol{x}}_d - \boldsymbol{J}\hat{\boldsymbol{J}}^\dagger(k\boldsymbol{e} + \dot{\boldsymbol{x}}_d))\\
\dot{V} &= -k\boldsymbol{e}^T\boldsymbol{J}\hat{\boldsymbol{J}}^\dagger\boldsymbol{e} + \boldsymbol{e}^T(\boldsymbol{I} - \boldsymbol{J}\hat{\boldsymbol{J}}^\dagger)\dot{\boldsymbol{x}}_d\,. && (4.35)
\end{aligned}$$

If a good estimate of $\boldsymbol{J}$ is computed, we have that $\hat{\boldsymbol{J}} \approx \boldsymbol{J}$ and:

$$\dot{V} = -k\boldsymbol{e}^T\boldsymbol{J}\boldsymbol{J}^\dagger\boldsymbol{e} + \boldsymbol{e}^T(\boldsymbol{I} - \boldsymbol{J}\boldsymbol{J}^\dagger)\dot{\boldsymbol{x}}_d\,. \tag{4.36}$$

Notice that the controlled system results *locally stable* if the following condition is satisfied:

$$\dot{V} \leq 0 \quad ; \tag{4.37}$$

let us now investigate what are the conditions for local stability. At first it is necessary to point out some remarks on equation (4.36):

- The first term $-(\boldsymbol{e}^T\boldsymbol{J}\hat{\boldsymbol{J}}^\dagger\boldsymbol{e})$ is semidefinite negative and vanishes if and only if $\boldsymbol{e}$ belongs to the $(m-6)$ dimensional null space of $\boldsymbol{J}^T$: $\mathcal{N}(\boldsymbol{J}^T)$. Notice that $\mathcal{N}(\boldsymbol{J}^T)$ is the orthogonal space to the 6 dimensional space $\mathcal{I}(\boldsymbol{J})$, namely the image space of $\boldsymbol{J}$.
- It is easy to show that the second term $(I - \boldsymbol{J}\boldsymbol{J}^\dagger)$ is a projector $\mathcal{N}(\boldsymbol{J}^T)$: thus, if $\boldsymbol{e}$ or $\dot{\boldsymbol{x}}_d \notin \mathcal{N}(\boldsymbol{J}^T)$ or also when $\dot{\boldsymbol{x}}_d = \boldsymbol{0}$ this term vanishes; otherwise nothing can be stated on its sign.

The system error $\boldsymbol{e}$ can be rearranged as follow:

$$\boldsymbol{e} = \boldsymbol{e}_\mathcal{N} + \boldsymbol{e}_\mathcal{I} \quad , \tag{4.38}$$

where $\boldsymbol{e}_\mathcal{N}$ and $\boldsymbol{e}_\mathcal{I}$ are the error components respectively belonging to $\mathcal{N}(\boldsymbol{J}^T)$ and $\mathcal{I}(\boldsymbol{J})$. By imposing condition (4.37) and exploiting equations (4.36) and (4.38) we easily obtain:

$$\dot{V} \leq -k\,\sigma_m\,\|\boldsymbol{e}_\mathcal{I}\|^2 + \|\dot{\boldsymbol{x}}_{d\mathcal{N}}\|\,\|\boldsymbol{e}_\mathcal{N}\|^2 \leq 0 \quad , \tag{4.39}$$

where $\dot{\boldsymbol{x}}_{d\mathcal{N}}$ is the component of $\dot{\boldsymbol{x}}_d$ belonging to $\mathcal{N}(\boldsymbol{J}^T)$ and σ_m is the smallest of the 6 positive non-zero singular values of the symmetric matrix $(\boldsymbol{J}\boldsymbol{J}^\dagger)$.
To ensure local stability it is thus sufficient to have:

$$k\,\sigma_m\,\|\boldsymbol{e}_\mathcal{I}\|^2 \geq \|\dot{\boldsymbol{x}}_{d\mathcal{N}}\|\,\|\boldsymbol{e}_\mathcal{N}\|^2 \quad . \tag{4.40}$$

The above inequality and then local stability are thus verified when one of the two following option is chosen:

1. to set $\dot{\boldsymbol{x}}_d = \mathbf{0} \rightarrow \dot{\boldsymbol{x}}_{d\mathcal{N}} = \mathbf{0}$, without providing any image path planning, thus resulting in $\boldsymbol{x}_d = \boldsymbol{x}_f$. This choice however is not advisable for two reasons:
   - *Singularities* of the control law especially may arise when I_i and I_f are very different (for large camera displacements and thus large initial error $\boldsymbol{e}$): in singularity the rank of the interaction matrix $\boldsymbol{J}$ decreases thus giving rise to very high camera twist screw $\boldsymbol{w}$ (see control law (4.32)) and provoking the failure of the positioning task.
   - *Local Minima* may occur in this case if the system reach the condition $\boldsymbol{e}_{\mathcal{I}} = \mathbf{0}$. Local Minima are in fact configuration where the error $\boldsymbol{e} \neq \mathbf{0} \in \mathcal{N}(\boldsymbol{J}^{\mathcal{T}})$ thus resulting in a control twist screw $\boldsymbol{w} = \mathbf{0}$ (according to (4.32) with $\dot{\boldsymbol{x}}_{\boldsymbol{d}} \in \mathbf{0}$): as a consequence the system converges in a configuration different from the goal one even if $\boldsymbol{e} \neq \mathbf{0}$ and positioning task fails. Notice that if we had a square Interaction matrix the null space $\mathcal{N}(\boldsymbol{J}^{\mathcal{T}})$ would be equal to $\mathbf{0} \in \mathbb{R}^{m\times 1}$ and Local Minima would not exist: this situation could be verified in our case by using just 3 points as visual features; however in such a case the system would be unfortunately affected by Singularity drawbacks as known from literature [Cha99].
2. to keep $\|\boldsymbol{e}_{\mathcal{I}}\|$ as much as possible bigger than $\|\boldsymbol{e}_{\mathcal{N}}\|$ and to minimize $\|\dot{\boldsymbol{x}}_{d\mathcal{N}}\|$ by performing an adequate image path planning on the references $\dot{\boldsymbol{x}}_{\boldsymbol{d}}$ To reach this goal it will be necessary at first to maintain the system error as smaller ("local") as possible since both $\mathcal{N}(\boldsymbol{J}^{\mathcal{T}})$ and $\mathcal{I}(\boldsymbol{J})$ are two orthogonal subspaces that are variable with the local linear mapping $\boldsymbol{J}(\boldsymbol{x}(t), \boldsymbol{r}_c(t))$ and at second to set *feasible configurations* (namely feasible target views) for the reference state vector $\boldsymbol{x}_d$.

In order to avoid all the problems related with the above first option it is essential perform an image path planning on the reference image features $\boldsymbol{x}_d$. The image planning strategy will have also to take into account the $3D$ motion constraint of the camera in the generation of the planned image trajectories so as to generate feasible reference target views.

4.3 Image Path Planning

Consider to deal with a target identified at least by $n \geq 4$ points $\boldsymbol{P}^i$, $i = 1, ..., n$ without any point collinear with other two belonging to a plane π, respectively projecting in I_i and I_f in the homogeneous pixel points $\boldsymbol{p}_i^i$ and $\boldsymbol{p}_i^f$, according to equation (2.22). Refer thus to the system described in section (4.2.1.1) for the definition of the state vector $\boldsymbol{x}$ and of the interaction matrix $\boldsymbol{J}$. To perform image path planning, the structure of the Euclidean Homography $\boldsymbol{H}_i^f$, defined in (3.14), can be exploited. Between the initial an the goal views in fact, the following equation holds:

$$\tilde{\boldsymbol{m}}_f \propto \boldsymbol{H}_i^f \tilde{\boldsymbol{m}}_i = \frac{Z_i}{Z_f} \boldsymbol{H}_i^f \tilde{\boldsymbol{m}}_i \quad , \tag{4.41}$$

where we have recalled (3.16), referring to image points in normalized coordinates. Notice that $\boldsymbol{H}_i^f$ can be estimated by the method reported in section (3.3.1.2) from I_i and I_f image points coordinates and their correspondences. Recall that the image path planning is needed to set the time-varying reference image trajectory $\boldsymbol{x}_d(t)$ defined by:

$$\boldsymbol{x}_d(t) = \left[\boldsymbol{m}_{d1}^T(t)\ \boldsymbol{m}_{d2}^T(t)\ ...\ \boldsymbol{m}_{dn}^T(t)\right]^T \quad \in \mathbb{R}^{(2n\times 1)} \tag{4.42}$$

Similarly to equation (4.41) the planned image trajectories, as shown in figure (4.1), can be generated by defining a *Time-varying Reference Euclidean Homography* $\boldsymbol{H}_d(t)$ such that for a generic reference feature point $\tilde{m}_d(t)$ of $\boldsymbol{x}_d(t)$ we have:

$$\tilde{\boldsymbol{m}}_d(s(t)) \propto \boldsymbol{H}_d(s(t))\tilde{\boldsymbol{m}}_{\boldsymbol{i}} \quad \rightarrow \quad \boldsymbol{x}_d(s(t)) = \boldsymbol{f}(\boldsymbol{H}_d(s(t))). \tag{4.43}$$

where $\tilde{m}_i$ is the corresponding image point in I_i and $s = s(t)$ is a suitable scalar time law that have to be chosen opportunely.

The IBVS control input, namely the camera twist-screw $\boldsymbol{w}$, can thus be generated

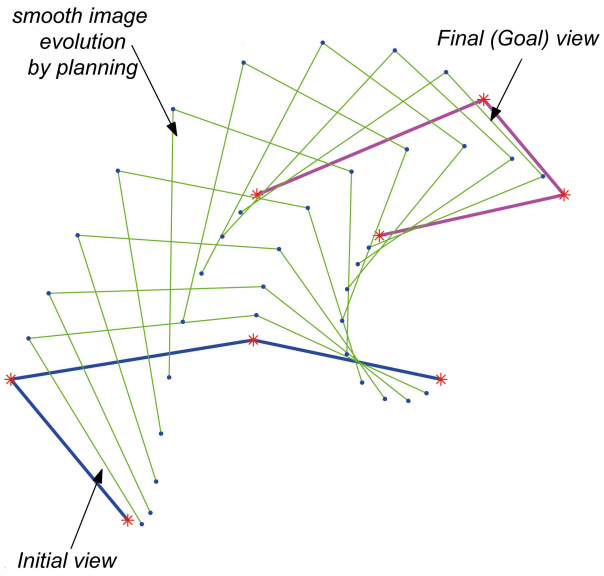


Figure 4.1: Evolution of the planning trajectories $\boldsymbol{x}_d(s(t))$.

smoothly varying the reference homography $\boldsymbol{H}_d(s)$ between the initial and the goal views. In this work the planning of the reference paths for the IBVS control is generated *off-line*. Here we summarize the main steps of our IBVS strategy:

- *Off-Line Steps*

  1. Estimate $\boldsymbol{H}_i^f$ from the pixel point coordinates $\boldsymbol{p}_i^i$ and $\boldsymbol{p}_f^i$ $i = 1, ..., n$ and their correspondences through the algorithm described in (3.3.1.2).
  2. Define an opportune planning Homography $\boldsymbol{H}_d(s)$ with the recovered camera kinematic parameters from I_i and I_f (we will see i the following how it can be done).
  3. Choose for an adequate time law $s(t)$ ranging from 0 to 1.

4. Generate the planned image trajectories $\boldsymbol{x}_d(s(t))$ by exploiting equation (4.43) for the n feature points.
5. Compute the time derivative of the reference trajectories $\dot{\boldsymbol{x}}_d(s(t))$.

- *On-Line Steps*

1. Estimate the actual system Interaction Matrix $\hat{\boldsymbol{J}}(\boldsymbol{x}, Z_x)$ according to (4.21) by using the actual tracked points normalized coordinates ($\boldsymbol{m}_1$, $\boldsymbol{m}_2$, ... $\boldsymbol{m}_n$,) and by adaptively estimating the actual $3D$ point depths ($\hat{Z}_1$, $\hat{Z}_2$, ... $\hat{Z}_n$,) through the law described in [CA01].
2. Compute the image error $\boldsymbol{e} = (\boldsymbol{x}_d - \boldsymbol{x})$.
3. Calculate the camera twist screw $\boldsymbol{w}$ according to (4.32) and send it to the robot controller to minimize $\boldsymbol{e}$ between actual and the planned reference views.

Since in the control section (4.2) we have already shown how to execute the On-line Steps in the following we will show in detail the image path planning strategy and thus how the above Off-line Steps can be performed.

4.3.1 The Time Law

To obtain smooth variations of the reference image features $\boldsymbol{x}_d(s(t))$ we define the time law $s(t)$ as a quintic-polynomial ranging from the initial time $t_i = 0$ to the final planning time t_f:

$$s(\tau) = a(\tau)^5 + b(\tau)^4 + c(\tau)^3 + d(\tau)^2 + e(\tau) + f, \tag{4.44}$$

with $\tau = (t/t_f)$. The coefficients in s can be found as the solution of (6×6) a linear system satisfying the following boundary conditions imposed on $s(t)$ ant its first two order time derivatives $\dot{s(t)}$ and $\ddot{s(t)}$:

$$\begin{cases} s(0) = 0, \quad \dot{s}(0) = 0, \quad \ddot{s}(0) = 0, \\ \\ s(t_f) = 1, \quad \dot{s}(t_f) = 0, \quad \ddot{s}(t_f) = 0\,. \end{cases} \tag{4.45}$$

4.3.2 The Euclidean Reference Homography

To ensure that the generic feature point reference planning trajectory $\tilde{\boldsymbol{m}}_d$ will range from $\tilde{\boldsymbol{m}}_i$ to $\tilde{\boldsymbol{m}}_f$ according to $s(t)$, the planning reference homography $\boldsymbol{H}_d(s) \in \mathbb{R}^{3\times3}$, according to equation (4.43) must verify the following boundary conditions:

$$\tilde{\boldsymbol{m}}_i \rightarrow \tilde{\boldsymbol{m}}_d \rightarrow \tilde{\boldsymbol{m}}_f \quad \Rightarrow \quad \begin{cases} \boldsymbol{H}_d(0) = \boldsymbol{I} \\ \\ \boldsymbol{H}_d(1) = \boldsymbol{H}_i^f \end{cases} \tag{4.46}$$

According to equation (3.14) the estimated euclidean homography $\boldsymbol{H}_i^f$ between I_i and I_f results:

$$\boldsymbol{H}_i^f = \boldsymbol{R}_f^{iT} \left(\boldsymbol{I} - \boldsymbol{t}\boldsymbol{s}_f^i \boldsymbol{n}_i^T \right) \tag{4.47}$$

where $\boldsymbol{R}_f^i = \boldsymbol{R}_i^{f\,T}$ and $\boldsymbol{ts}_f^i$ are respectively the rotation matrix and the scaled translation defining the goal camera frame C_f with respect to C_i and $\boldsymbol{n}_i$ is the plane π normal expressed in C_i. Let us recall also that the scaled translation $\boldsymbol{ts}_f^i$ is defined by:

$$\boldsymbol{ts}_f^i = \frac{\boldsymbol{t}_f^i}{d_i} \tag{4.48}$$

that is the origin of C_f expressed in C_i divided by the distance d_i of C_i origin from the plane π. To plan $\boldsymbol{H}_d(s)$ compatible with the rigid camera motion, we thus exploit

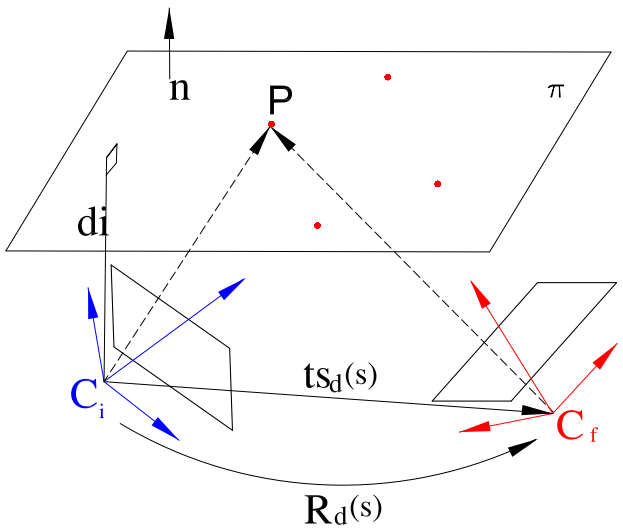


Figure 4.2: The reference homography $\boldsymbol{H}_d(s)$ is obtained from the reference rotation $\boldsymbol{R}_d(s)$ and from the reference scaled translation $\boldsymbol{ts}_d(s)$.

Euclidean Homography structure of (4.48) as shown in figure (4.2) obtaining:

$$\boldsymbol{H}_d(s) = \boldsymbol{R}_d(s)^T(\boldsymbol{I} - \boldsymbol{ts}_d(s)\boldsymbol{n}_i{}^T) \tag{4.49}$$

where to verify the boundary conditions (4.46) it is natural to impose the following kinematic conditions:

$$\begin{cases} \boldsymbol{R}_d(0) = \boldsymbol{I}, \quad \boldsymbol{R}_d(1) = \boldsymbol{R}_f^i, \\ \\ \boldsymbol{ts}_d(0) = \boldsymbol{0}, \quad \boldsymbol{ts}_d(1) = \boldsymbol{ts}_f^i\,. \end{cases} \tag{4.50}$$

The vector $\boldsymbol{ts}_d = \frac{\boldsymbol{t}_d}{d_i}(s)$ represents the scaled translation between the planned scaled camera frame and the initial one. To estimate the unknown parameters $\boldsymbol{R}_f^i$, $\boldsymbol{ts}_f^i$ and $\boldsymbol{n}_i$ from $\boldsymbol{H}_i^f$ it is necessary to perform the Euclidean homography normalization and decomposition respectively reported in section (3.3.2) and (3.3.3). The definition of $\boldsymbol{H}_d(s)$, considering equation (4.49), is now split into the following two sub-problems: the *Reference Rotation matrix* $R_d(s)$ definition and the *the reference scaled translation vector* $\boldsymbol{ts}_d(s)$ definition.

4.3.2.1 The Reference Rotation Matrix

The Reference rotation matrix $R_d(s)$ in equation (4.49) can be defined by exploiting *Rodriguez formula*, derived from the matrix exponential form:

$$\boldsymbol{R}_d(s) = \boldsymbol{I} + [\boldsymbol{u}]_\times \sin(\theta_d(s)) + [\boldsymbol{u}]_\times^2 (1 - \cos(\theta_d(s)) \quad . \tag{4.51}$$

Let us assume for the *reference rotation angle* $\theta_d(s)$ the following law:

$$\theta_d(s) = \theta_f^i \, s(t) \tag{4.52}$$

so as to obtain the boundary conditions $\theta_d(0) = 0$ for $t = 0$ and $\theta_d(1) = \theta_f^i$ for $t = t_f$. By injecting the previous equation in (4.51) we finally obtain:

$$\boldsymbol{R}_d(s) = \boldsymbol{I} + [\boldsymbol{u}]_\times \sin(\theta_f^i \, s) + [\boldsymbol{u}]_\times^2 (1 - \cos(\theta_f^i \, s)) \quad , \tag{4.53}$$

where $\boldsymbol{u}$ is the unit vector identifying the finite rotation axis between C_i and C_f and θ_f^i is the overall rotation angle. $[\ ]_\times$ denotes, as already stated, the skew operator.
The parameter set $(\boldsymbol{u}, \theta_f^i)$ known as axis-angle representation, can be estimated from the known rotation matrix $\boldsymbol{R}_f^i$ as follows:

$$\begin{aligned} \cos(\theta_f^i) &= \frac{1}{2}(tr(\boldsymbol{R}_f^i) - 1) \quad , \\ \sin(\theta_f^i) &= \sqrt{1 - \cos(\theta_f^i)^2} \quad , \\ \theta_f^i &= \text{atan2}(\sin(\theta_\text{f}^\text{i}) \, , \, \cos(\theta_\text{f}^\text{i})), \end{aligned} \tag{4.54}$$

and

$$[\boldsymbol{u}]_\times = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} = \frac{\boldsymbol{R}_f^{iT} - \boldsymbol{R}_f^i}{2\sin(\theta_f^i)} \quad . \tag{4.55}$$

Notice moreover that $\boldsymbol{u}$ is the unit length eigenvector or $\boldsymbol{R}_f^i$ associated with the eigenvalue 1. From expressions (4.54) we always obtain $0 \leq \theta_{if} \leq \pi$.

4.3.2.2 The Reference Scaled Translation

To define the time dependent reference scaled translation of the camera $\boldsymbol{t}_d(s)/d_i = \boldsymbol{ts}_d(s)$ of equation (4.49) we must choose a trajectory that identifies the position of the scaled reference camera frame origin during the task considering also the boundary conditions of (4.50).
Since the parameter d_i is unknown, we only are able to define the shape of the reference trajectory and not its real dimension (we define, in fact, a scaled trajectory). Notice that whatever the unknown scale factor, the desired trajectories in the image plane remain the same. So, if the control is synthesized in the image plane, the positioning task can be fulfilled regardless of the scale factor. Moreover we observe that the knowledge of d_i is equivalent to the knowledge of the real dimensions of the whole scene (object of interest and reference trajectory).
Among the infinite shapes for the reference translation we choose for our planning the

helicoidal trajectory; we consider indeed the helicoidal shape as the most appropriate to join the initial and desired positions of the camera since it perfectly harmonizes traslation with rotation [AF05]. A generic helix is characterized by these canonical parametric equations:

$$\begin{cases} x(\gamma(s)) = r\cos(\gamma(s)) \\ y(\gamma(s)) = r\sin(\gamma(s)) \\ z(\gamma(s)) = \frac{p}{2\pi}\gamma(s) \\ 0 \leq \gamma(s) \leq \gamma_1 \end{cases} , \tag{4.56}$$

where x,y and z are the coordinates of the curve expressed in a canonical cartesian frame defined by E as shown in figure (4.3). This right-handed coordinate system has the z-axis coincident with the axis of the helix and the x-axis defined by the vector perpendicular to the helix axis through initial point of the curve (the point defined by $s = 0$). The helix we choose has the z-axis direction coincident with the unit vector $\boldsymbol{u}$ and the total angle γ_1 covered by the curve equal to θ_f^i, as shown in the Figure. We also assume :

$$\gamma(s) = \theta_d(s) = \theta_f^i \, s \quad ; \tag{4.57}$$

in this way we obtain for the camera motion a translation that is perfectly harmonized with the rotation. To completely define our scaled reference helix we must still find:

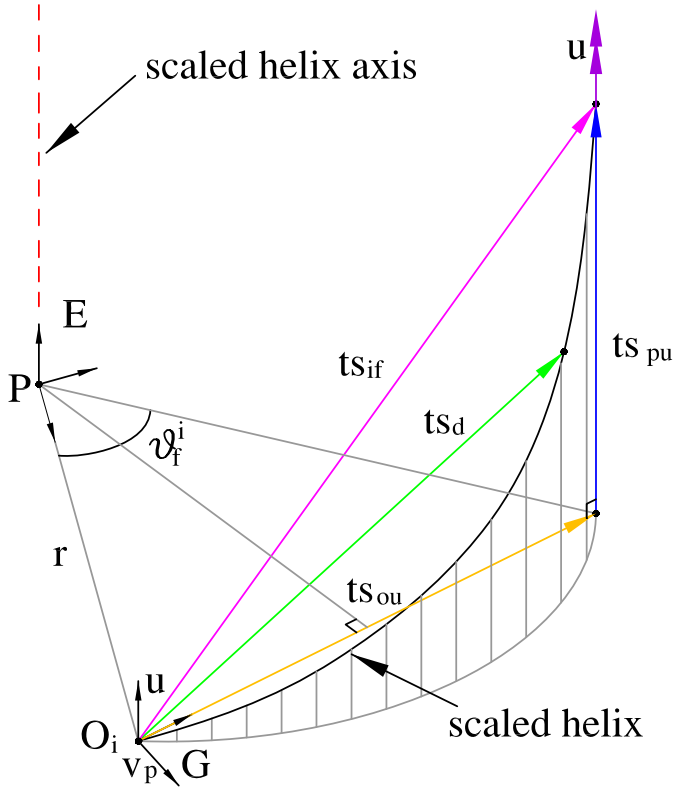


Figure 4.3: Reference scaled helix used for the camera translation path in the general motion case.

- the pitch of the helix p;

- the radius of the helix r;
- the origin of the canonical coordinate system E in the scaled 3D space.

In order to find these parameters we decompose the overall scaled translation $\boldsymbol{ts}_f^i$ of (4.47) in its parallel and orthogonal components to the unit vector $\boldsymbol{u}$, respectively $\boldsymbol{ts}_{pu}$ and $\boldsymbol{ts}_{ou}$. We have:

$$\boldsymbol{ts}_{pu} = (\boldsymbol{ts}_f^{i\,T}\boldsymbol{u})\boldsymbol{u} \quad , \tag{4.58}$$

$$\boldsymbol{ts}_{ou} = \boldsymbol{ts}_f^i - \boldsymbol{ts}_{pu} \quad . \tag{4.59}$$

Now we have another three sub cases:

(1) - $\boldsymbol{ts}_{pu} \neq \boldsymbol{0}$ and $\boldsymbol{ts}_{ou} \neq \boldsymbol{0}$;

(2) - $\boldsymbol{ts}_{pu} \neq \boldsymbol{0}$ and $\boldsymbol{ts}_{ou} = \boldsymbol{0}$;

(3) - $\boldsymbol{ts}_{pu} = \boldsymbol{0}$ and $\boldsymbol{ts}_{ou} \neq \boldsymbol{0}$.

Sub case - 1: $\boldsymbol{ts}_{pu} \neq \boldsymbol{0}$ and $\boldsymbol{ts}_{ou} \neq \boldsymbol{0}$ − In this sub case it is easy to check that:

$$p/(2\pi) = (\boldsymbol{ts}_f^{i\,T}\boldsymbol{u})/\theta_f^i \quad . \tag{4.60}$$

From the previous equation the helix pitch can be obtained:

$$p = \frac{2\pi}{\theta_f^i}(\boldsymbol{ts}_f^{i\,T}\boldsymbol{u}) \quad . \tag{4.61}$$

We can note from (4.61) that if we have $(\boldsymbol{ts}_{pu}^T\boldsymbol{u}) > 0$ we get a *right* helix while if $(\boldsymbol{ts}_{pu}^T\boldsymbol{u}) < 0$ we get a *left* helix. With simple geometric passages we obtain the radius as:

$$r = \frac{|\boldsymbol{ts}_{ou}|}{2\sin(\theta_f^i/2)} \quad . \tag{4.62}$$

The radius and the pitch of the real reference helix, respectively r_r and p_r, are unknown variables expressed from these relations:

$$r_r = rd_i \quad , \quad p_r = pd_i \quad . \tag{4.63}$$

To fix the scaled helix in the space we have to finally find the coordinate of the origin of the frame E, defined as O_e, respect the initial frame C_i of the camera.
Let be G a right-handed coordinate system with the same origin of C_i and orthogonal unit vectors defined as follow:

$$\begin{cases} \boldsymbol{v}_p = \frac{\boldsymbol{ts}_{ou}}{|\boldsymbol{ts}_{ou}|} \wedge \boldsymbol{u} \\ \boldsymbol{v}_{t\perp u} = \frac{\boldsymbol{ts}_{ou}}{|\boldsymbol{ts}_{ou}|} \\ \boldsymbol{u} \end{cases} ; \tag{4.64}$$

from the last relations we can compute the (4×4) transformation matrix from C_i to G:

$$\boldsymbol{M}_G^{C_i} = \begin{bmatrix} \boldsymbol{v}_p & \boldsymbol{v}_{t\perp u} & \boldsymbol{u} & \mathbf{0} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_G^{C_i} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix} \quad . \tag{4.65}$$

The position and the orientation of E expressed in G are respectively represented by the vector $\boldsymbol{ts}_E^G$ and by the rotation matrix $\boldsymbol{R}_E^G$; from simple geometry considerations, we obtain:

$$\boldsymbol{ts}_E^G = [\ -r\cos(\theta_f^i/2) \quad r\sin(\theta_f^i/2) \quad 0\ ]^T \quad , \tag{4.66}$$

$$\boldsymbol{R}_E^G = \begin{bmatrix} \cos(\theta_f^i/2) & \sin(\theta_f^i/2) & 0 \\ -\sin(\theta_f^i/2) & \cos(\theta_f^i/2) & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad . \tag{4.67}$$

The transformation matrix from G to E can be expressed as:

$$\boldsymbol{M}_E^G = \begin{bmatrix} \boldsymbol{R}_E^G & \boldsymbol{ts}_E^G \\ \mathbf{0}^T & 1 \end{bmatrix} \quad . \tag{4.68}$$

Now we are able to identify the placement of E from the coordinate system C_i through the transformation matrix $\boldsymbol{M}_E^{C_i}$:

$$\boldsymbol{M}_E^{C_i} = \boldsymbol{M}_G^{C_i} \boldsymbol{M}_E^G \quad . \tag{4.69}$$

From the equations (4.56) and (4.57) we obtain the reference scaled position of the camera expressed in E:

$$\boldsymbol{ts}_{d(E)} = \begin{cases} x(s) = r\cos(\theta_f^i s)) \\ y(s) = r\sin(\theta_f^i s) \\ z(s) = \frac{p}{2\pi}\theta_f^i s \end{cases} \tag{4.70}$$

Using equation (4.69) we obtain the reference scaled translation expressed in C_i:

$$\tilde{\boldsymbol{ts}}_d = \boldsymbol{M}_E^{C_i}\, \tilde{\boldsymbol{ts}}_{d(E)}, \tag{4.71}$$

with $\tilde{\boldsymbol{ts}}_d = [\boldsymbol{ts}_d, 1]^T$ and $\tilde{\boldsymbol{ts}}_{d(E)} = [\boldsymbol{ts}_{d(E)}, 1]^T$, thus finally obtaining the reference scaled translation $\boldsymbol{ts}_d$.

Sub case - 2: $\boldsymbol{ts}_{pu} \neq \mathbf{0}$ and $\boldsymbol{ts}_{ou} = \mathbf{0}$ − From equation (4.62) we have in this sub case $r = 0$. In this situation the helix degenerates into a line and the scaled reference translation can easily be computed:

$$\boldsymbol{ts}_d = s\, \boldsymbol{ts}_f^i \tag{4.72}$$

Sub case - 3: $\boldsymbol{ts}_{pu} = \mathbf{0}$ and $\boldsymbol{ts}_{ou} \neq \mathbf{0}$ − From the equations (4.58) and (4.61) we have $p = 0$. In this sub case the helix degenerates into an arc of circle and the vector $\boldsymbol{ts}_{d(E)}$

of equation (4.70) become:

$$\boldsymbol{ts}_{d(E)} = \begin{cases} x(s) = r\cos(\theta_f^i s) \\ y(s) = r\sin(\theta_f^i s) \\ z(s) = 0 \end{cases} \quad . \tag{4.73}$$

4.3.2.2.1 Pure rotational motion case In this case we have only to produce a rotation of the camera to reach the desired view, in fact equation (4.47) become:

$$\boldsymbol{H}_i^f = \boldsymbol{R}_f^{iT} \quad . \tag{4.74}$$

We extract as in in the previous subsection the unit vector $\boldsymbol{u}$ and the angle θ_f^i from the rotation matrix $\boldsymbol{R}_f^i$ and we plan the rotation of the camera using equation (4.53):

$$\boldsymbol{R}_d(s) \;=\; \boldsymbol{I} + [\boldsymbol{u}]_\times \sin(\theta_f^i\, s) + [\boldsymbol{u}]_\times^2 (1 - \cos(\theta_f^i\, s)) \quad .$$

4.3.2.2.2 Pure translational motion case In this last case equation (4.47) becomes:

$$\boldsymbol{H}_i^f \;=\; (\boldsymbol{I} - \frac{\boldsymbol{t}_f^i}{d_i} \boldsymbol{n}_i{}^T) \tag{4.75}$$

and the camera need only to translate to reach the desired position. We plan the desired trajectory along a line joining the initial and desired position using the parameterization expressed in (4.72):

$$\boldsymbol{ts}_d \;=\; s\boldsymbol{ts}_f^i$$

4.3.3 Optimization of the desired trajectory

It could be useful to do some further modifications to the helicoidal planning defined above in order to better suite any admissible initial and desired view of the target and respective camera poses.
The first modification comes from the fact that the reference helicoidal trajectory extends in the robot workspace and in some cases, because of its convexity, brings the camera too much close to the target: in this situation some feature point can leave the camera field of view and the task is compromised. To avoid these cases it is sufficient *to have the convexity of the helicoidal reference trajectory always turned from the opposite part respect to the target.*
To reach this goal we at first consider the frame G defined in (4.64): we note that any helix produced from the planning lies in the half space delimited by the coordinate plane with normal $\boldsymbol{v}_p$ characterized by its positive direction. Since we assume the normal $\boldsymbol{n}_i$ to the target plane π, obtained from the decomposition of $\boldsymbol{H}_i^f$, to have its positive direction toward the half space delimited by π containing the target, we operate in this manner:

- we maintain the helix generated from the previous planning if $(\boldsymbol{v}_p^T\, \boldsymbol{n}_i \leq 0)$
- we instead modify the reference helix if $(\boldsymbol{v}_p^T\, \boldsymbol{n}_i > 0 =$ (in these cases, in fact, the convexity of the curve is turned toward the target).

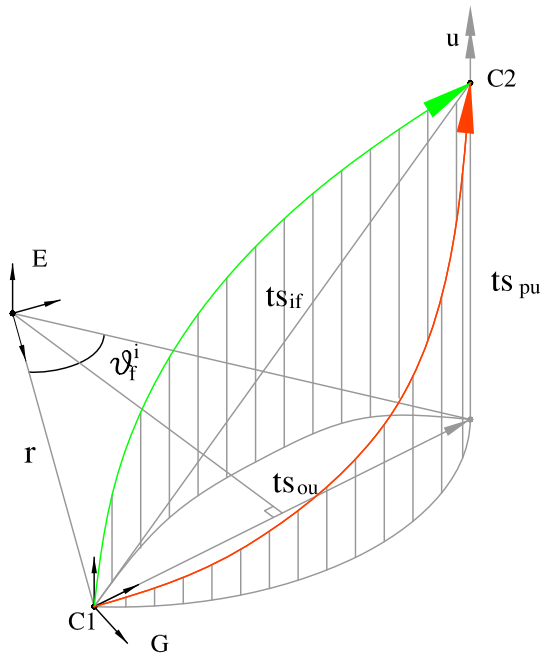


Figure 4.4: First available modification for scaled helix planning: to plan on the green helix arc instead of on the red one.

The change we operate if ($\boldsymbol{v}_p^t\, \boldsymbol{n}_i > 0$) is to plan the reference position trajectory on an helix symmetric to the old one respect to the coordinate plane of W with normal $\boldsymbol{v}_p$ as shown in figure (4.4). To operate this symmetry it is sufficient to introduce a matrix of the form:

$$\boldsymbol{S} = \mathrm{diag}(-1, 1, 1, 1) \quad . \tag{4.76}$$

in equation (4.69). This equation become:

$$\boldsymbol{M}_{E_{simm}}^{C_i} \;=\; \boldsymbol{M}_{G}^{C_i}\, \boldsymbol{S}\, \boldsymbol{M}_{E}^{G} \quad , \tag{4.77}$$

where with E_{simm} we have defined the canonical frame associated to the modified symmetric helix. The second modification we produce to optimize the camera translation path rises from the fact that in some complex tasks the orientation planning brings the camera to turn toward the opposite part respect to the target: also in these cases the target gets out of the camera field of view and the task is compromised. These unfortunate cases might arise when the initial view, the desired or both views are particular views in which the camera focal axis intersects the object plane Π in a point with negative depth. To resolve this problem we plan the camera orientation and translation on a different helix characterized by these new parameters as shown in figure (4.5):

$$\boldsymbol{u}_{new} = -\boldsymbol{u} \qquad , \qquad \theta_{if\,new} = 2\pi - \theta_{if} \quad . \tag{4.78}$$

This modification makes the camera to join the desired frame always maintaining the camera toward the target. Notice that, even if in some unfortunate cases the feature points would leave the camera field of view, the paths planning in the image space permits to prevent these situations and so to avoid them choosing a different reference scaled 3D

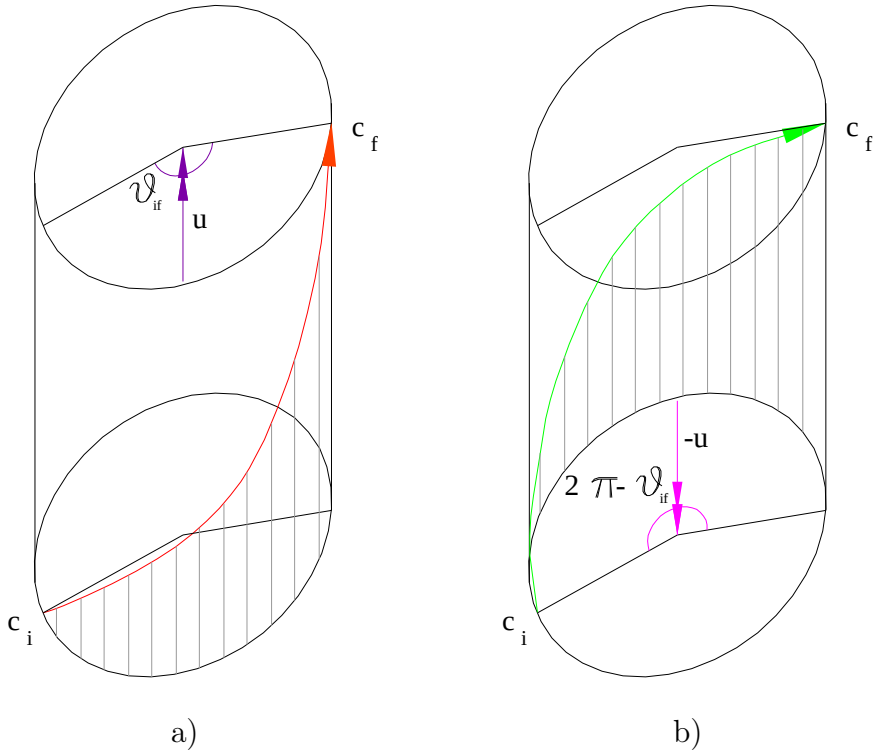


Figure 4.5: Second available modification for scaled helix planning. a) Helix before the modification. b) Helix after the modification.

trajectory for the camera.

Chapter 5

IBVS from Coaxial Circles

In this chapter is presented a self-calibrated approach to visual servoing with respect to a wide class of non planar targets. Valid targets are those that can be modeled through *a pair of coaxial circles* plus *one point* on either circle plane. This a priori assumption on target shape makes it possible to self-calibrate a full perspective camera (fixed internal parameters) from two views, by exploiting the method originally developed in [CDBP05] for a single view of a solid of revolution. The special target structure also allows the reformulation of the servoing problem in terms of camera positioning with respect to a virtual planar target. Calibration data are used to recover the scaled Euclidean structure of the virtual plane, together with scaled camera relative pose. Pose disambiguation is achieved by generating a synthetic third view of the target from the second one. The approach includes an off-line 2D trajectory planning strategy by which the camera follows a 3D helicoidal path around an arbitrary axis, which can be conveniently chosen as the target axis itself. The chapter is organized as follows. In Sect. 5.1.1 the target model is defined; its imaged properties are then discussed in Sect. 5.1.2 together with calibration and 3D estimation issues. In Sect. 5.1.3 the control scheme is described, and an algorithm for helicoidal path planning is given. Next Chapter includes for completion results obtained from simulation tests.

5.1 The Approach

5.1.1 Modeling

The visual servoing approach proposed in this chapter works for a class of targets which can be modeled through a pair of coaxial circles. This geometric structure is shared by a large class of man-made objects, including solids of revolution such as bottles, vases, lamps and any lathe-crafted objects as a special case. An additional requirement for targets is the existence of a well detectable point different from a circle center, and placed anywhere on either circle planes: this point (referred to in this work as *reference point* $\boldsymbol{P}_r$) can be related either to the visual texture of the object, or to special artifacts on it—e.g., the handle of a cup. Fig. 5.1(a) provides an example of solid of revolution target. Only two views of the target, referred to as *initial* and *desired* view, are required in the proposed approach. Moreover, the approach is self-calibrated, the only condition on the camera internal parameters, embedded in the matrix $\boldsymbol{K}$, is that they remain unchanged during task execution. For any target, the plane π_r through the circles' axis and the reference

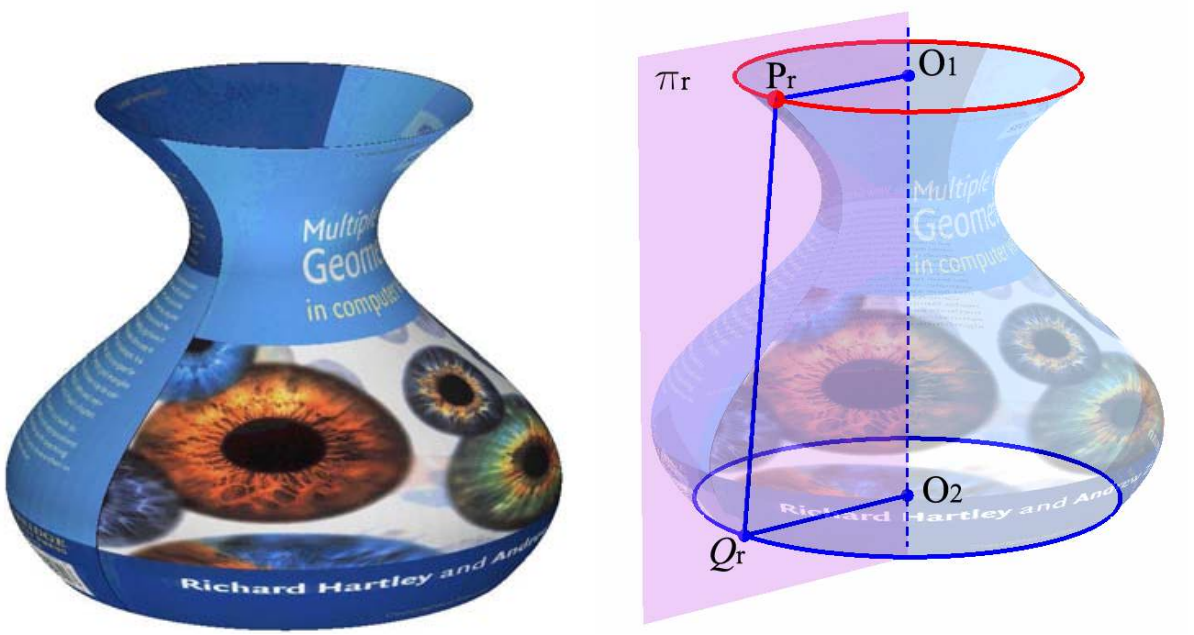


Figure 5.1: A solid of revolution target and its geometry. (a): A view of the target. (b): The features relevant for the approach.

point can be defined. Consequently, the servoing task can be reformulated in terms of initial and final visual appearance of a *reference quadrangle* on π_r, with vertexes $\boldsymbol{P}_r$, $\boldsymbol{O}_1$, $\boldsymbol{O}_2$, $\boldsymbol{Q}_r$—see Fig. 5.1(b). The control approach in thus finally equivalent to the feature point one described above in in Chapter (4.2).

5.1.2 Vision

5.1.2.1 Imaged target and related 2D features

Anytime during task execution, perspective projection of the 3D object onto the image plane gives rise to the two ellipses $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ (i.e., the images of the two coaxial circles, represented as 3×3 symmetric homogeneous matrices), and the imaged reference point $\boldsymbol{p}_r$ (represented as a homogeneous 3-vector).
The axial symmetry properties of the 3D target induce a projectively symmetric 2D configuration of the ellipses, which is characterized by the imaged symmetry axis $\boldsymbol{l}_s$ and the vanishing point $\boldsymbol{v}_\infty$ of the normal direction of the plane passing through $\boldsymbol{l}_s$ and the camera center. Two other fundamental entities characterizing the imaged target are the imaged circular points $\boldsymbol{i}$ and $\boldsymbol{j}$ of the pencil of planes orthogonal to the target axis (ref. section (2.1.4).
As shown in [CDBP05], the image entities $\boldsymbol{l}_s$, $\boldsymbol{v}_\infty$, $\boldsymbol{i}$ and $\boldsymbol{j}$ can all be computed from the knowledge of the ellipses $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ and their conditions of visibility. Given $\boldsymbol{C}_1$, $\boldsymbol{C}_2$ and $\boldsymbol{p}_r$ (which without loss of generality will be assumed as belonging to $\boldsymbol{C}_1$), the vanishing line $\boldsymbol{l}_\infty = (\boldsymbol{i} \wedge \boldsymbol{j})$ of the plane pencil above is used at any time during task execution to compute the remaining imaged vertexes of the reference quadrangle, namely $\boldsymbol{o}_1$, $\boldsymbol{o}_2$, and $\boldsymbol{q}_r$—see Fig. 5.2. For line and point projective geometry refer to section (2.2). Explicitly,

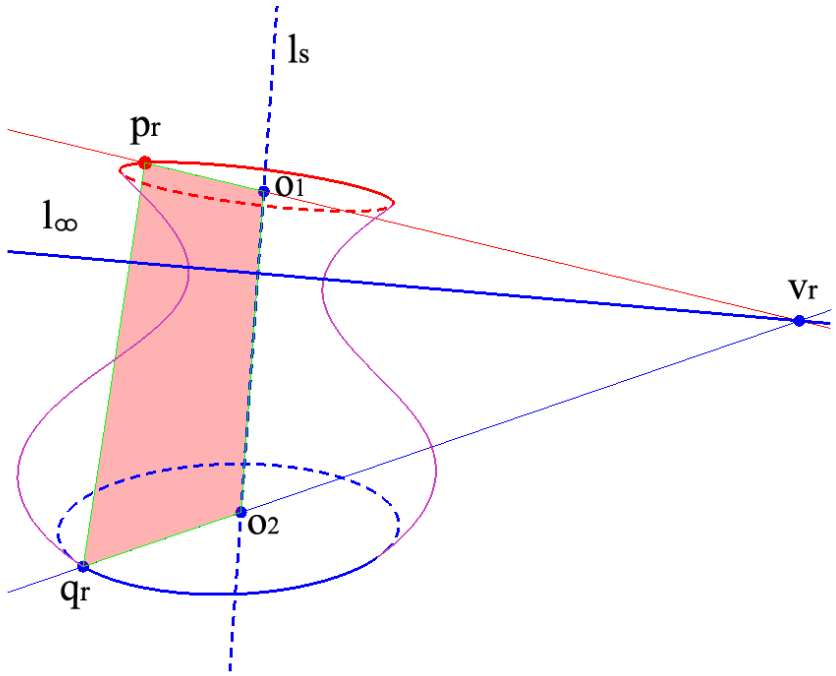


Figure 5.2: Construction of the imaged target geometry.

the imaged circle centers are computed from the pole-polar relationships (ref. to section 2.1.3):

$$\begin{cases} \boldsymbol{o}_1 &= \boldsymbol{C}_1^{-1}\boldsymbol{l}_\infty \\ \boldsymbol{o}_2 &= \boldsymbol{C}_2^{-1}\boldsymbol{l}_\infty \end{cases} . \tag{5.1}$$

Once these are known, the vanishing point of the line $(\boldsymbol{O}_1 \wedge \boldsymbol{P}_r)$ is computed as:

$$\boldsymbol{v}_r = (\boldsymbol{o}_1 \wedge \boldsymbol{p}_r) \wedge \boldsymbol{l}_\infty \quad . \tag{5.2}$$

Finally, the fourth vertex $\boldsymbol{q}_r$ is obtained by first intersecting the line $(\boldsymbol{o}_2 \wedge \boldsymbol{v}_r)$ with the ellipse $\boldsymbol{C}_2$, and then choosing, between the two possible solutions, the one farthest from $\boldsymbol{v}_r$. During task execution, the position of the imaged vertexes of the reference quadrangle are computed at each frame by estimates of both the two ellipses and the reference point.

5.1.2.2 Self-calibration

The imaged target axis $\boldsymbol{l}_s$ and the vanishing point $\boldsymbol{v}_\infty$ are related to each other as [WMC03]:

$$\boldsymbol{l}_s = \omega\, \boldsymbol{v}_\infty \ , \tag{5.3}$$

where ω is the *image of the absolute conic* (IAC) $\boldsymbol{\Omega}_\infty$ defined in section (2.2.4). ω carries out all the information about the camera calibration matrix $\boldsymbol{K}$ and its internal parameters; it can be shown in fact that:

$$\omega = (\boldsymbol{K}\boldsymbol{K}^T)^{-1} \quad . \tag{5.4}$$

Other two constraints on $\boldsymbol{K}$ which are induced by the special structure of the target are:

$$\boldsymbol{i}^T \omega\, \boldsymbol{i} = \boldsymbol{j}^T \omega\, \boldsymbol{j} = \boldsymbol{0} \ . \tag{5.5}$$

In [CDBP05] it is demonstrated that, for each view, the system composed by eqs. 5.3 and 5.5 provides three independent constraints on ω, and hence on the five internal camera parameters as for the full perspective CCD model defined in (2.4.3). Therefore, under the hypothesis of unchanged $\boldsymbol{K}$, a full camera calibration can be obtained from the image data of both the initial and final target views.

5.1.2.3 Scaled 3D measurements

Also in this case, for the purpose of trajectory planning (later addressed in Sect. 5.1.3), estimates of the 3D structure of the reference quadrangle and the relative camera displacement are required. Although three of them are "virtual" (namely not directly extracted from the image), the four reference quadrangle vertexes can be effectively considered as four point features: the problem is thus reduced to a simple planar target case. Thanks to the above observation the 3D scene structure can be estimated by solving one more time the structure from motion problem described in section (3.3).

First, the 3×3 planar pixel homography $\boldsymbol{G}_i^f$, mapping the corresponding vertexes of the initial and final imaged quadrangles, is computed as shown in (3.3.1.2) as the solution of the following system:

$$[\boldsymbol{p}_{r\,f} \;\; \boldsymbol{o}_{1\,f} \;\; \boldsymbol{o}_{2\,f} \;\; \boldsymbol{q}_{r\,f}] = \boldsymbol{G}_i^f \; [\boldsymbol{p}_{r\,i} \;\; \boldsymbol{o}_{1\,i} \;\; \boldsymbol{o}_{2\,i} \;\; \boldsymbol{q}_{r\,i}] \;\; . \tag{5.6}$$

The calibration matrix $\boldsymbol{K}$, estimated whit the self-calibration technique above, is then used to compute the so called "Euclidean homography" $\boldsymbol{H}_i^f = \boldsymbol{K}^{-1}\boldsymbol{G}_i^f\boldsymbol{K}$. Finally, by performing the normalization described in (3.3.2), the Euclidean homography can be decomposed through the procedure (3.3.3) into its rotational and parallax components recalling the form (4.47):

$$\boldsymbol{H}_i^f \;\; = \;\; \boldsymbol{R}_f^{i\,\mathrm{T}}(\boldsymbol{I} - \boldsymbol{t}\boldsymbol{s}_f^i \, \boldsymbol{n}_{i\,r}^{\mathrm{T}}) \;\; , \tag{5.7}$$

where, as explained in section (3.1) we have:

- $\boldsymbol{R}_f^i$ is the rotation matrix between the initial ($\mathcal{F}_i$) and final ($\mathcal{F}_f$) camera frames;
- $\boldsymbol{t}\boldsymbol{s}_f^i$ is the scaled translation between $\mathcal{F}_i$ and $\mathcal{F}_f$;
- $\boldsymbol{n}_{i\,r}^{\mathrm{T}}$ is the unit normal to π_r.

Notice that all the elements are expressed with respect to $\mathcal{F}_i$. As specified in (3.3.3), the planar homography decomposition problem from a palnar target has a twofold ambiguity, which can be removed by the additional information provided by a third view of the plane. In this work, a real third view of the target is actually not needed, being it sufficient to exploit the symmetry properties of the target. To this aim, a *synthetic view* of the quadrangle can be generated by arbitrarily choosing, in the final image, a new point $\boldsymbol{p}_v \in \boldsymbol{C}_1$ other than $\boldsymbol{p}_r$, and using the method expounded above to obtain the corresponding fourth quadrangle vertex (Fig. 5.3).

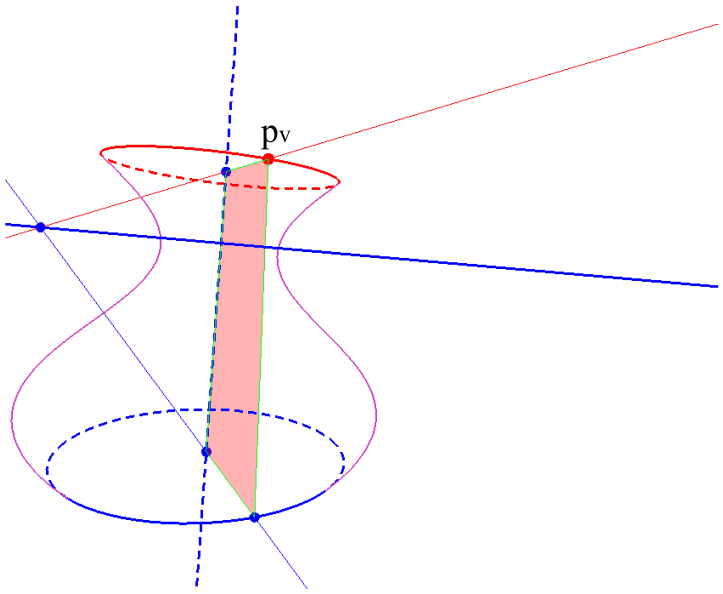


Figure 5.3: Construction of the synthetic view for pose disambiguation.

5.1.3 Control

5.1.3.1 Control scheme

Now let us calculate the normalized coordinates of the four quadrangle vertexes from the pixel ones by inverting equation (2.27):

$$\tilde{\boldsymbol{p}}_r^n = \boldsymbol{K}^{-1}\boldsymbol{p}_r \quad \tilde{\boldsymbol{o}}_1^n = \boldsymbol{K}^{-1}\boldsymbol{o}_1 \quad \tilde{\boldsymbol{o}}_2^n = \boldsymbol{K}^{-1}\boldsymbol{o}_2 \quad \tilde{\boldsymbol{q}}_r^n = \boldsymbol{K}^{-1}\boldsymbol{q}_r \quad , \tag{5.8}$$

The state system vector $\boldsymbol{x}$ (see 4.2.1) can be built in this case with the four quadrangle vertexes in normalized inhomogeneous coordinates:

$$\boldsymbol{x} = [\boldsymbol{p}_r^{n\,\mathrm{T}} \;\; \boldsymbol{o}_1^{n\,\mathrm{T}} \;\; \boldsymbol{o}^{n\,\mathrm{T}}2 \;\; \boldsymbol{q}_r^{n\,\mathrm{T}}]^{\mathrm{T}} \tag{5.9}$$

be the 8-vector constructed using, in a generic view, the imaged vertexes of the reference quadrangle. Recall that the system dynamics is expressed by:

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\,\boldsymbol{w} \quad , \tag{5.10}$$

where $\boldsymbol{w} = [\boldsymbol{v}^{\mathrm{T}} \;\; \boldsymbol{\omega}^{\mathrm{T}}]^{\mathrm{T}} \in \mathbb{R}^{6\times 1}$ is the twist velocity screw of the camera expressed in the camera frame and $\boldsymbol{J} = [\boldsymbol{J}_1^{\mathrm{T}} \;\; \boldsymbol{J}_2^{\mathrm{T}} \;\; \boldsymbol{J}_3^{\mathrm{T}} \;\; \boldsymbol{J}_4^{\mathrm{T}}]^{\mathrm{T}}$, its generic block $\boldsymbol{J}_k$ is the quadrangle vertexes interaction matrix like (4.21). The system can thus be visually controlled like in the planar target case, with point as visual features, described in section (4.2.1.1). The control law used is [AC99, MC02]:

$$\boldsymbol{w} \;=\; \hat{\boldsymbol{J}}^{\dagger}[\lambda\boldsymbol{e} + \dot{\boldsymbol{x}}_d] \quad , \tag{5.11}$$

where $\hat{\boldsymbol{J}}^{\dagger}$ is the pseudo-inverse of the estimated Jacobian obtained through the on-line adaptive depth estimation law proposed in [CA01], λ is a positive gain, $\boldsymbol{e} = (\boldsymbol{x}_d - \hat{\boldsymbol{x}})$ is the image error (difference between the desired and the actual normalized imaged quadrangle vertexes), and $\dot{\boldsymbol{x}}_d$ is the reference evolution of quadrangle appearance resulting from image

path planning.

5.1.3.2 Path planning

Off-line image path planning is performed by generating, at each time step t, a Euclidean homography $\boldsymbol{H}_d(t)$ by which the reference imaged quadrangle is given by

$$[\lambda_1 \boldsymbol{p}^n_{r\,d}(t)\ \ \lambda_2 \boldsymbol{o}^n_{1\,d}(t)\ \ \lambda_3 \boldsymbol{o}^n_{2\,d}(t)\ \ \lambda_4 \boldsymbol{q}^n_{r\,d}(t)] = \boldsymbol{H}_d(t)\ [\boldsymbol{p}^n_{r\,i}\ \ \boldsymbol{o}^n_{1\,i}\ \ \boldsymbol{o}^n_{2\,i}\ \ \boldsymbol{q}^n_{r\,i}]\ , \tag{5.12}$$

where $(\lambda_1, \lambda_2, \lambda_3, \lambda_4)$ are scale factors that can be removed by normalizing the third coordinates of the planned vertexes to 1. While in section (4.3) the special case of helicoidal planning with constant radius and axis identical to the instantaneous axis of motion has discussed, here we address the general case of variable radius and arbitrary axis.
In this case, the planned Euclidean homography is designed so that the camera follows a helicoidal trajectory in the 3D space, whose radius and axis depend on the overall displacement $[\boldsymbol{R}^i_f\ \boldsymbol{ts}^i_f]$ between the initial and current camera frames. Following the decomposition of eq. (4.47), the planning homography can be written as

$$\boldsymbol{H}_d(t)\ \ =\ \ \boldsymbol{R}^{i\,\mathrm{T}}_d(t)(\boldsymbol{I} - \boldsymbol{t}^{i\,\mathrm{T}}_d(t)\,\boldsymbol{n}^{\mathrm{T}\,i}_r)\ , \tag{5.13}$$

where $[\boldsymbol{R}^{i\,\mathrm{T}}_d(t)\,\boldsymbol{t}^{i\,\mathrm{T}}_d(t))]$ denotes the desired 3D displacement of the camera at time t with respect to frame $\mathcal{F}_i$. As in section (4.3.2.1), the planned rotation $\boldsymbol{R}^{i\,\mathrm{T}}_d(t)$ can be expressed through the Rodriguez' formula [MSKS03] as

$$\boldsymbol{R}^{i\,\mathrm{T}}_d(t) = \boldsymbol{I} + \left[{}^i\boldsymbol{u}\right]_\times \sin(\theta^i_f\, s(t)) + \left[{}^i\boldsymbol{u}\right]^2_\times\ [1 - \cos(\theta^i_f\, s(t))]\ , \tag{5.14}$$

where $[\,\cdot\,]_\times$ is the skew-symmetric operator, $({}^i\boldsymbol{u}, \theta^i_f)$ is the axis-angle representation of the overall camera rotation matrix $\boldsymbol{R}^i_f$, and $s(t)$ can be chosen as the smooth monotonic function of time law defined in section (4.3.1) with $s(t_i) = 0$ and $s(t_f) = 1$. In order to plan a helicoidal trajectory around an arbitrary 3D axis L, the desired scaled translation ${}^i\widetilde{\boldsymbol{\tau}}_d(t)$ is computed by the following algorithm—see also Fig. 5.4.

i) Let π_i and π_f be the planes through the target axis L and the initial and final camera centers, respectively. Compute the unit vectors ${}^i\boldsymbol{n}_i$ and ${}^i\boldsymbol{n}_f = \boldsymbol{R}^{i\,f}_f \boldsymbol{n}_f$ normal to the planes using the imaged axes $\boldsymbol{l}_i$ and $\boldsymbol{l}_f$ and the formula $\boldsymbol{n} = \boldsymbol{K}^{\mathrm{T}}\boldsymbol{l}/\|\boldsymbol{K}^{\mathrm{T}}\boldsymbol{l}\|$.

ii) Compute the direction of the helicoidal axis as ${}^i\boldsymbol{u}_l = ({}^i\boldsymbol{n}_i \times {}^i\boldsymbol{n}_f)/\|{}^i\boldsymbol{n}_i \times {}^i\boldsymbol{n}_f\|$. Compute the vector component ${}^i\widetilde{\boldsymbol{\tau}}^{\perp}_f$ perpendicular to ${}^i\boldsymbol{u}_l$ of the overall scaled translation $\boldsymbol{ts}^i_f$.

iii) Compute the (scaled) distances between L and the initial and final camera centers respectively as $r_i = |{}^i\widetilde{\boldsymbol{\tau}}^{\perp}_f \times {}^i\boldsymbol{n}_i|$ and $r_f = |{}^i\widetilde{\boldsymbol{\tau}}^{\perp}_f \times {}^i\boldsymbol{n}_f|$. Compute the angle between the planes π_i and π_f as

$$\alpha^i_f = \arccos\left(\frac{r_i^2 + r_f^2 - \|{}^i\widetilde{\boldsymbol{\tau}}^{\perp}_f\|^2}{2 r_i r_f}\right)\ . \tag{5.15}$$

iv) Compute the (scaled) pitch of the planned helix as

$$p_d = \frac{2\pi}{\alpha_f^i} \, {}^i\widetilde{\boldsymbol{\tau}}_f^{\mathrm{T}} \boldsymbol{u}_l \ . \tag{5.16}$$

v) Define the right-handed coordinate frame $\mathcal{F}_h$ with the following characteristics: (a) (scaled) origin on L, at the point at minimum distance from the initial camera center; (b) x-axis orthogonal to L and directed toward the initial camera center; (c) z-axis parallel to $\boldsymbol{u}_l$. The planned (scaled) helix equations in this frame are

$${}^h\widetilde{\boldsymbol{\tau}}(t)_d = \begin{bmatrix} r(t)\cos(\alpha_f^i s(t)) \\ r(t)\sin(\alpha_f^i s(t)) \\ (p_d/2\pi)\alpha_f^i s(t) \end{bmatrix} \ , \tag{5.17}$$

where $r(t) = r_i + (r_f - r_i)\, s(t)$ represents the variable helicoidal radius.

vi) Finally, compute the desired scaled translation ${}^i\widetilde{\boldsymbol{\tau}}_d(t)$ from ${}^h\widetilde{\boldsymbol{\tau}}(t)_d$ by the change of frame mapping $\mathcal{F}_h \mapsto \mathcal{F}_i$.

Notice that, as $\boldsymbol{ts}_f^i$ is defined up to a scale factor, the lack of knowledge about the real distance between the initial and final camera centers does not affect the planning in any way.
Although the algorithm above applies to any arbitrarily chosen axis, the special structure of the target suggests that the most natural choice for the axis is that of the target axis itself. In the following chapter, this choice will be discussed and compared with the one proposed in (4.3.2.1).
One of the remarkable properties of off-line planning is the possibility to forecast whether image features will remain or not in the camera field of view during task execution.
Fig. 5.5 shows the three planar loci for camera center where singularities occur. In planes π_1 and π_2, one of the ellipses degenerates into a line segment; in π_r the vertexes of the imaged quadrangle are all aligned. Singularities occurring on-line can be pointed out by visual analysis: They can be bypassed by opening the control loop near degeneracy conditions, and using a "reduced" control law including the feedforward term and the last useful estimate of the imaged Jacobian [AC99].
Choosing the helicoidal axis coincident with the target axis has the further advantage that degenerate ellipses are avoided, provided that both the initial and final camera centers both lie below, in the middle or above π_1 and π_2.

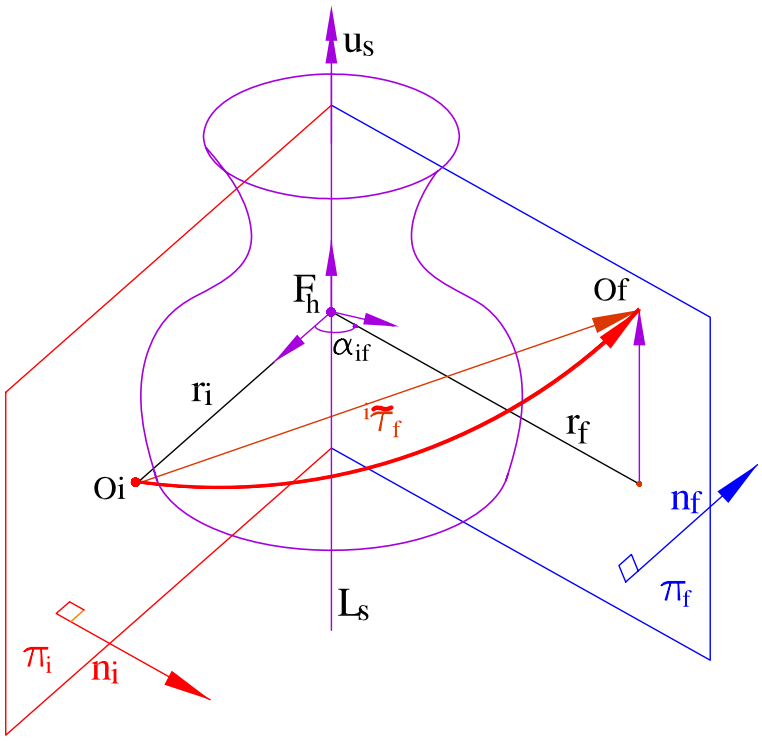


Figure 5.4: Geometric construction for the desired scaled translation.

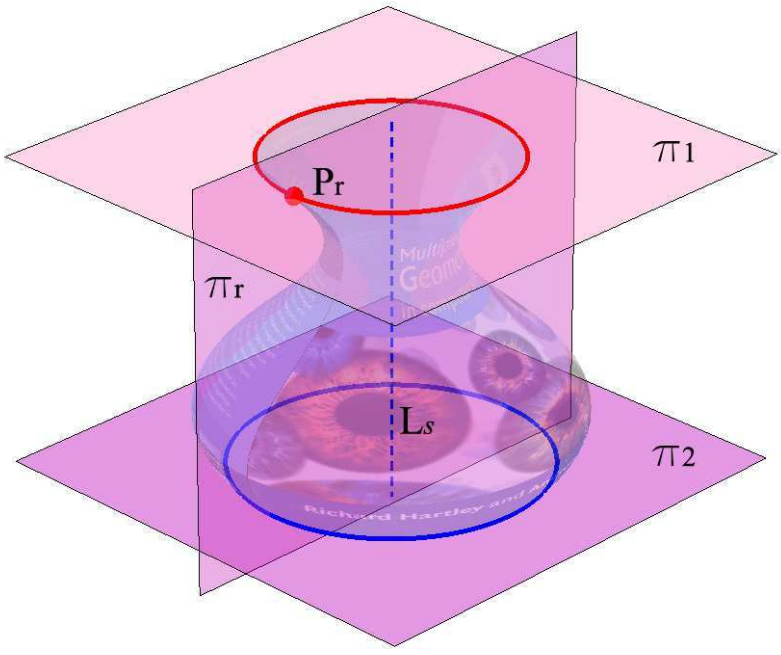


Figure 5.5: Vision and control singularity planes for camera center position.

Chapter 6

Results

To demonstrate the feasibility of the addressed planning approach, both simulation results using Matlab© and real experiment have been carried out. The chapter is organized as follows: first simulations for positioning tasks are reported by using both a planar square target and an axial symmetric object; then several results concerning IBVS experiments on planar targets in different conditions are shown. Both simulation and experiments demonstrate the good performances of the proposed approach also for large camera displacements.

6.1 Simulations

Numerical results have been performed by the development a simulator reproducing the scene and the controlled system dynamic taking also into account on-line depth estimation. For the axial symmetric target case moreover analysis have been performed showing the influence of Gaussian noise in features extraction on control performances.

6.1.1 Servoing from points

Simulation results for the planar target case are obtained using a full projective camera with the following intrinsic matrix:

$$\boldsymbol{K} = \begin{bmatrix} 458 & 0.01 & 323 \\ 0 & 462 & 237 \\ 0 & 0 & 1 \end{bmatrix} \quad . \tag{6.1}$$

We consider for these simulations a CCD with resolution of (640×480) like most of the usually adopted cameras for visual servoing systems. The target is defined by 4 well-detectable points at the vertexes of a $(0.4\,\mathrm{m} \times 0.4\,\mathrm{m})$ square . The initial and the desired camera poses are defined with respect to the target frame T having its origin at the center of the square and z-axis pointing inside the target plane π. A camera pose is defined by:

- the vector (t_x, t_y, t_z), expressed in meters-$[m]$, identifying the camera frame origin with respect to the target frame T;
- the (ZYZ) Euler angles (ϕ, θ, ψ), expressed in degrees-$[deg]$, identifying the camera frame orientation with respect to T.

6.1.1.1 Simulation I

In this simulation the initial camera frame C_i is defined with respect to T by:

$t_{xi} = -0.6$	$t_{yi} = 0.7$	$t_{zi} = -1$
$\phi_i = -35$	$\theta_i = 40$	$\psi_i = 65$

C_f instead defined is with respect to T by:

$t_{xf} = 1.5$	$t_{yf} = 2.8$	$t_{zf} = -3$
$\phi_f = 90$	$\theta_f = -50$	$\psi_f = 170$

The corresponding initial and goal target images with the image point trajectories are shown in Figure 6.1 in normalized coordinates. As shown in Figure 6.2 the control system well drives the actual features to the goal image along the planned trajectories. Both image error $\boldsymbol{e}$ and estimated depth errors decrease to zero, as respectively reported in Figure 6.3-a and 6.3-b, keeping smooth and feasible the control camera velocity screw $\boldsymbol{w}$ as can be observed in Figures 6.9 and 6.5.

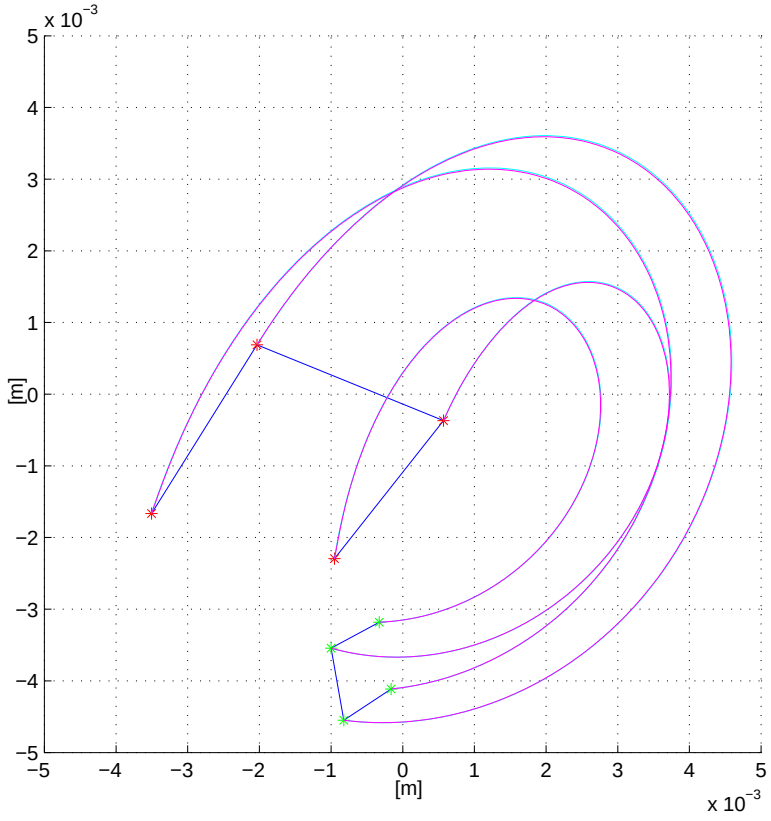


Figure 6.1: Initial (left) and goal (right) target views with planned and executed trajectories in the normalized image space. The executed trajectories are overlapped with the reference ones.

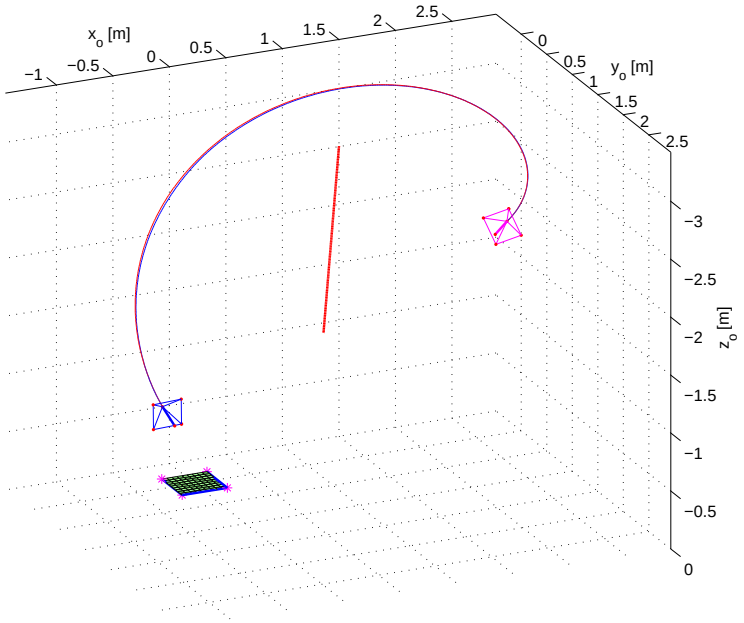
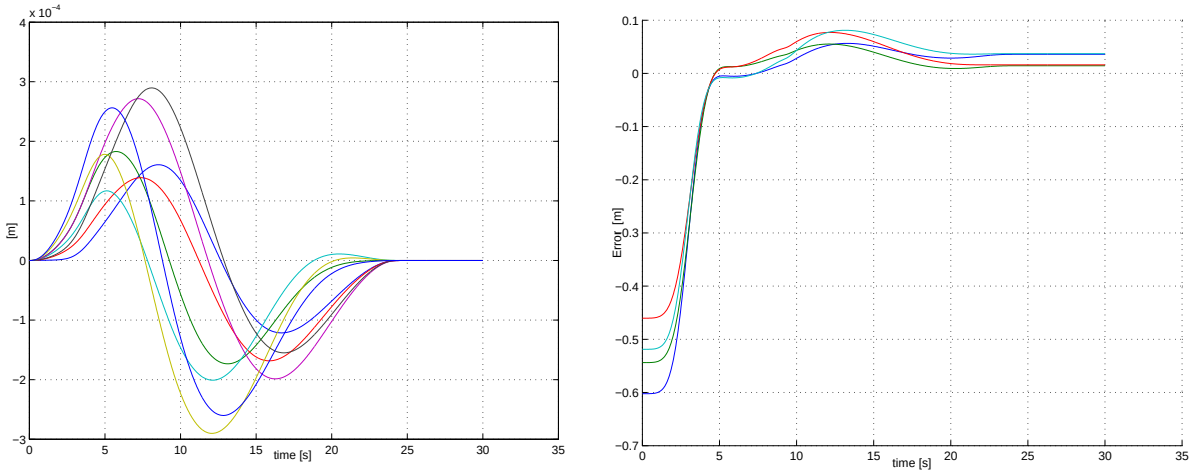


Figure 6.2: Reference (rescaled to the actual scale factor d_i) and real camera trajectories in 3D space ranging from the initial camera pose (bottom-left) to the goal camera pose (top).



a) b)

Figure 6.3: a) The 8 image errors $\boldsymbol{e}$ components during task execution: as shown the components evolution is smooth according to the quintic time law $s(t)$. b)Estimated depth errors $(Z_i - \hat{Z}_i)$ $i = 1, \ldots, 4$ for the tracked feature point during task execution: as pictured depth errors starting from large initial values is minimized toward 0. The depth estimates are computed exploiting the adaptive law addressed in [CA01].

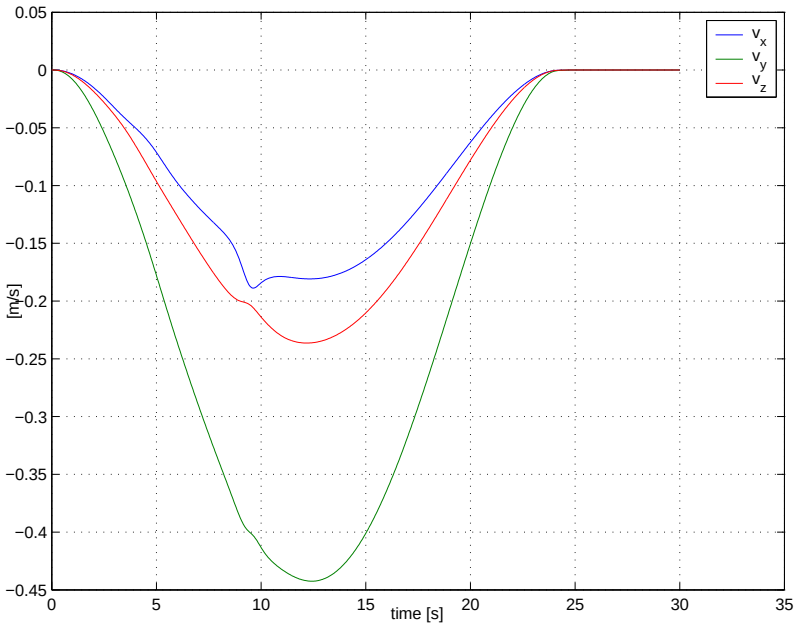


Figure 6.4: a) Camera twist screw $\boldsymbol{w}$ translational velocity components (v_x, v_y v_z) versus time. All the 3 screw components evolves smoothly according to the quintic polynomial time law for the image planning trajectory $s(t)$.

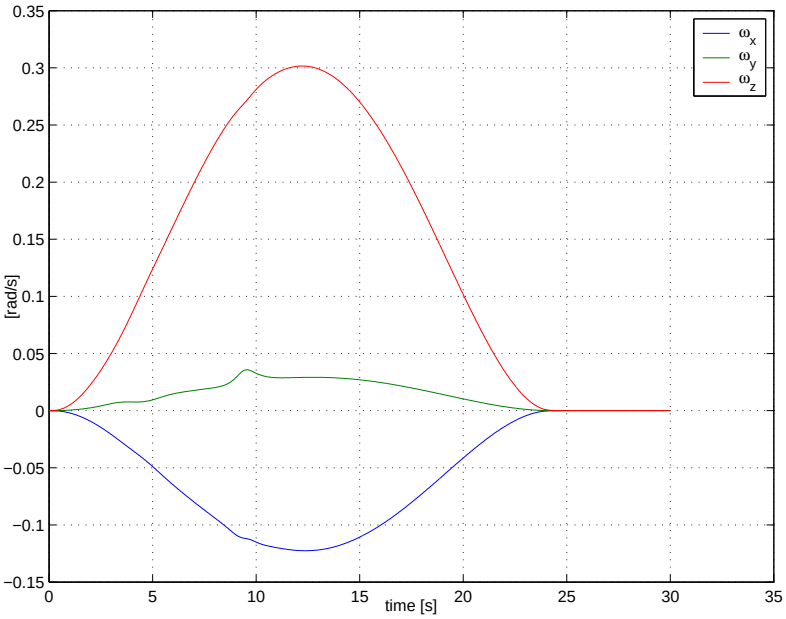


Figure 6.5: Camera twist screw $\boldsymbol{w}$ rotational velocity components (ω_x, ω_y ω_z) versus time. All the 3 screw components evolves smoothly according to the quintic polynomial time law for the image planning trajectory $s(t)$.

6.1.1.2 Simulation II

Now we show a more difficult positioning task simulation where the goal camera view has an higher perspective "distortion" than in the previous simulation. The initial camera frame C_i is defined with respect to T by:

$t_{x\,i} = -0.4$	$t_{y\,i} = 0.5$	$t_{z\,i} = -1$
$\phi_i = -25$	$\theta_i = 30$	$\psi_i = 35$

C_f instead defined is with respect to T by:

$t_{x\,f} = 0.6$	$t_{y\,f} = 1.8$	$t_{z\,f} = -0.1$
$\phi_f = 90$	$\theta_f = -100$	$\psi_f = 20$

The corresponding initial and goal target images with the image point trajectories images are shown in Figure 6.6 in reprojected pixel point coordinates. As shown in Figure 6.7 the control system well drives the actual features to the goal image along the planned trajectories. Both image error $\boldsymbol{e}$ and estimated depth errors decrease to zero as respectively reported in Figure 6.8-a and 6.8-b keeping smooth and feasible the control camera velocity screw $\boldsymbol{w}$ as can be observed in Figures 6.9 and 6.10.

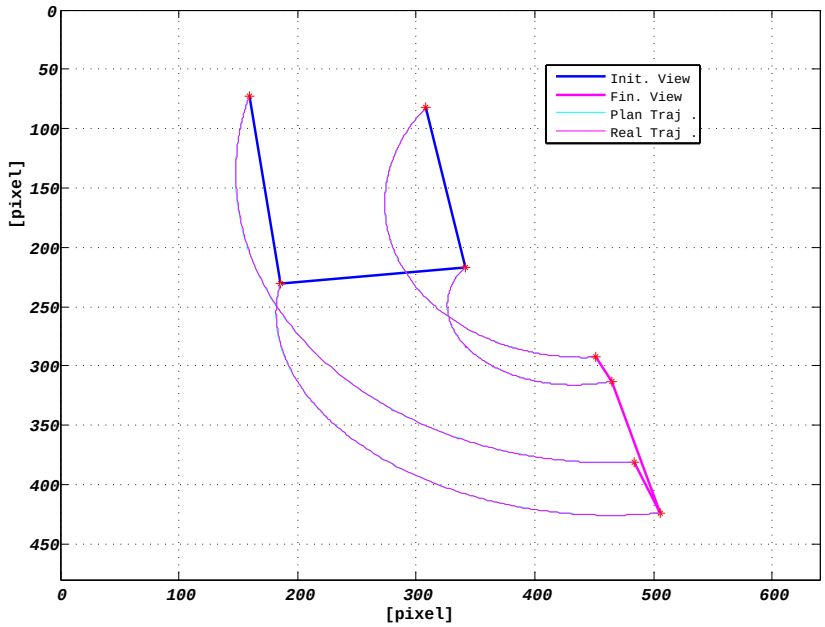


Figure 6.6: Initial (left) and goal (right) target views with planned and executed trajectories in the image space. The executed trajectories are overlapped with the reference ones.

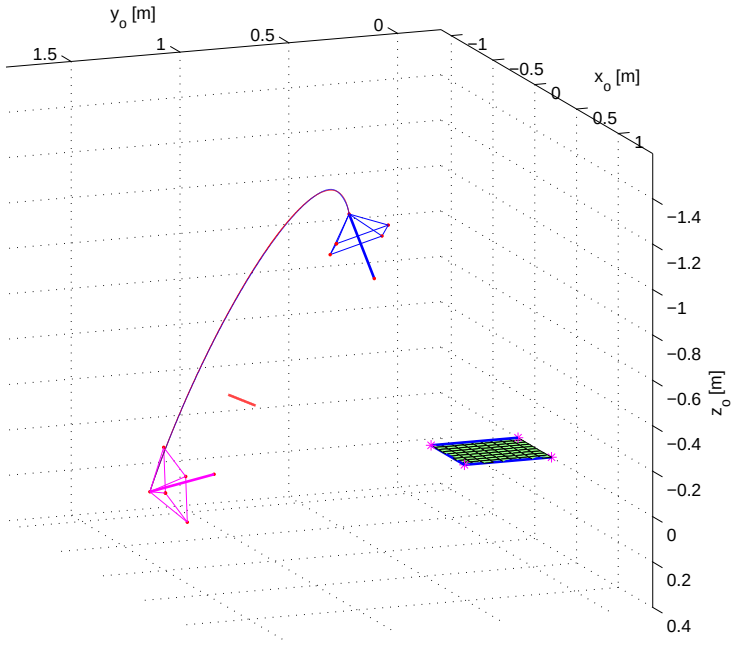
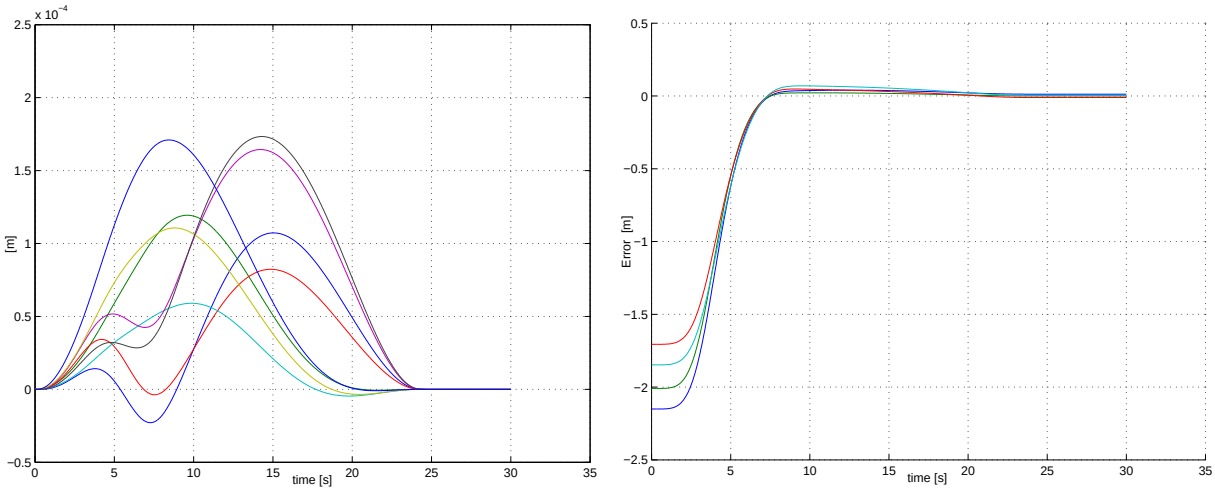


Figure 6.7: Reference (rescaled to the actual scale factor d_i) and real camera trajectories in 3D space ranging from the initial camera pose (top) to the goal camera pose (bottom-left).



a) b)

Figure 6.8: a) The 8 image errors $\boldsymbol{e}$ components during task execution: as shown the components evolution is smooth according to the quintic time law $s(t)$. b) Estimated depth errors $(Z_i - \hat{Z}_i)$ $i = 1, \ldots, 4$ for the tracked feature point during task execution: as pictured depth errors starting from large initial values is minimized toward 0. The depth estimates are computed exploiting the adaptive law addressed in [CA01].

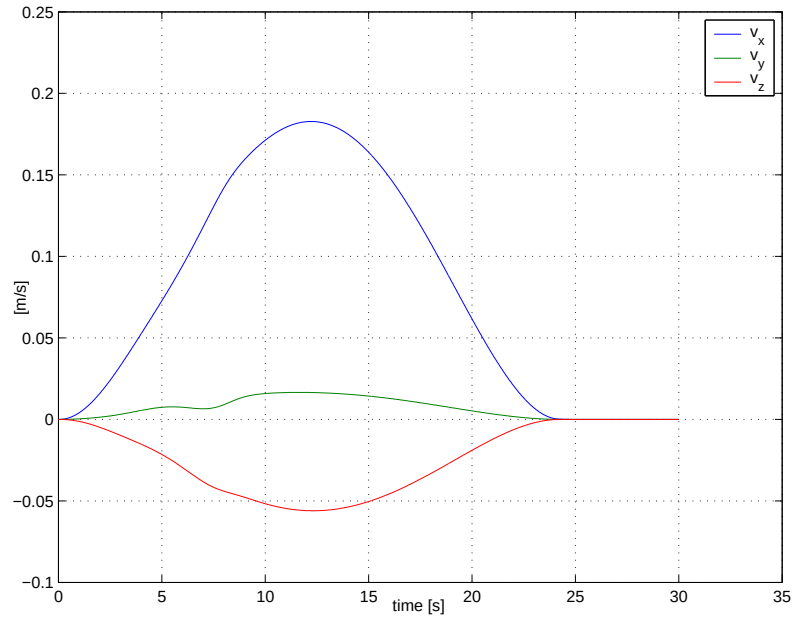


Figure 6.9: a) Camera twist screw $\boldsymbol{w}$ translational velocity components (v_x, v_y v_z) versus time. All the 3 screw components evolves smoothly according to the quintic polynomial time law for the image planning trajectory $s(t)$.

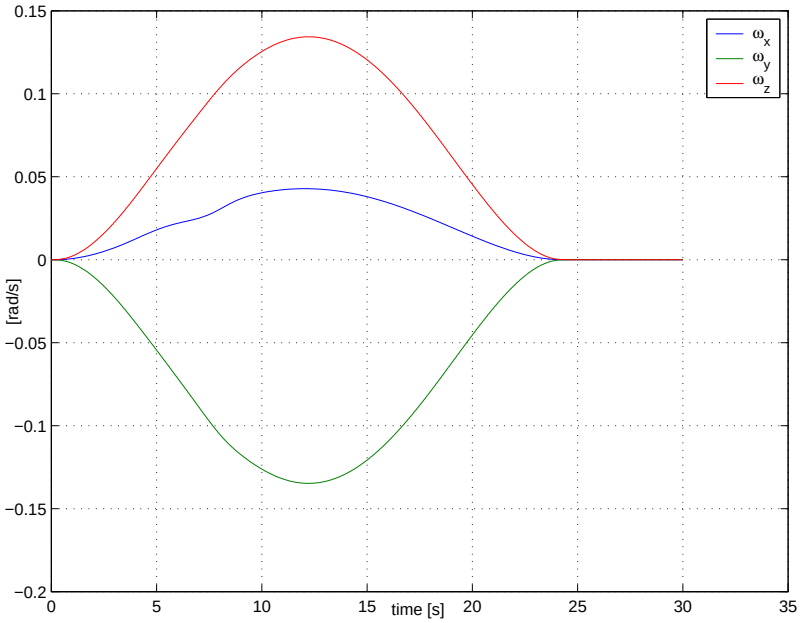


Figure 6.10: Camera twist screw $\boldsymbol{w}$ rotational velocity components (ω_x, ω_y ω_z) versus time. All the 3 screw components evolves smoothly according to the quintic polynomial time law for the image planning trajectory $s(t)$.

6.1.2 Servoing from coaxial circles

To gain an insight into the approach of section (5), a simulator for general image-based visual servoing tasks with an eye-in-hand robotic system has been developed, and various tests have been carried out. The simulation software allows to define the 3D shape of the target models (two circles plus one point), and then to define visual tasks by generating arbitrary initial and goal views. Off-line path planning can be defined with respect to an arbitrary axis in space. On-line task execution includes the generation of successive views reflecting the actual camera motion and noise conditions, and the estimation of visual parameters from each view. The actual target visibility constraints (self-occlusions) are taken into account. Ellipses are estimated by using the least square fitting algorithm proposed in [FPF99]. In all the tests, views have been generated using a solid of revolution target, a perspective camera with (640×480) pixels, and a calibration matrix

$$\mathtt{K} = \begin{bmatrix} 468.2 & 1.0 & 310.0 \\ & 427.2 & 230.0 \\ & & 1.0 \end{bmatrix} . \tag{6.2}$$

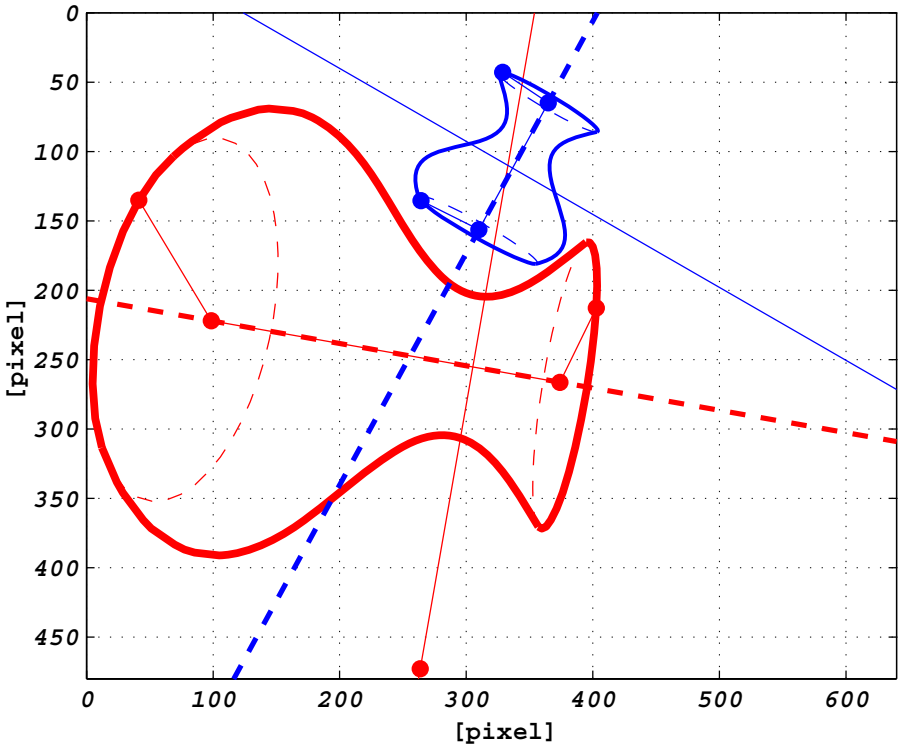
As for the simulated planar target cases of subsection (6.1.1), tasks executed in the absence of image noise are performed with negligible error—both on camera calibration and positioning accuracy—with respect to ground truth values, thus proving the theoretical correctness of the formulation. In the following, tests performed in the presence of zero mean additive Gaussian noise on ellipse points are discussed.

6.1.2.1 Off-line noise

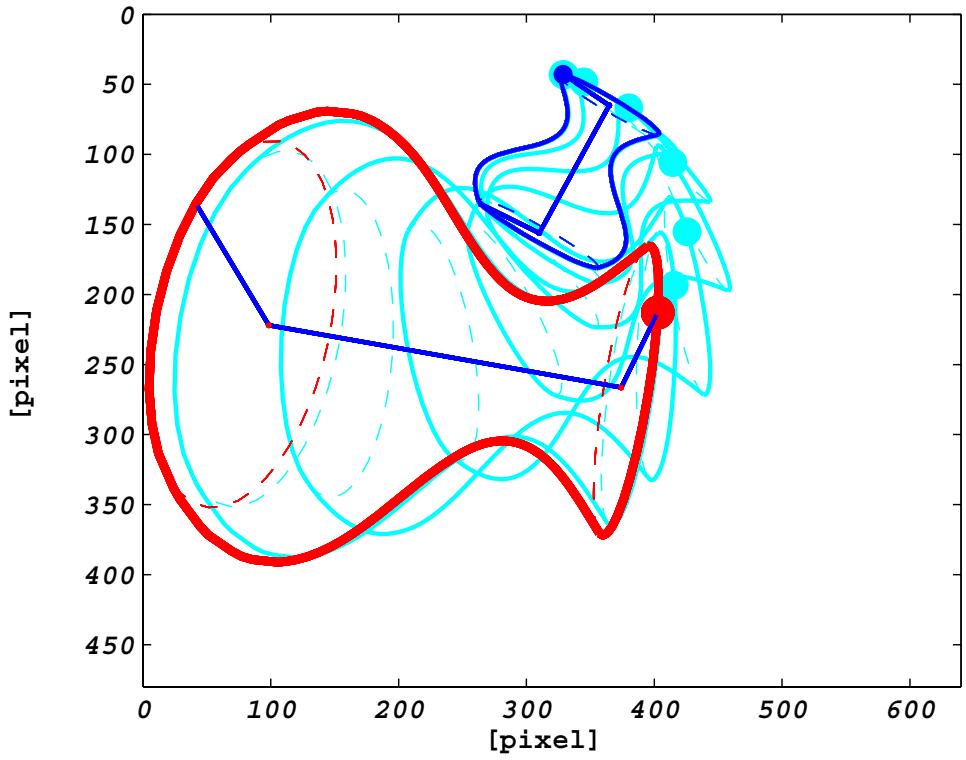
The first test concerns the case of off-line noise only, and is aimed at focusing on the effects of a noisy self-calibration (and hence on system behavior) on task accuracy—see also [Esp93]. Due to image noise, the estimated calibration matrix is

$$\hat{\mathtt{K}} = \begin{bmatrix} 496.3 & 5.7 & 331.4 \\ & 407.5 & 211.6 \\ & & 1.0 \end{bmatrix} . \tag{6.3}$$

Fig. 6.11(a) illustrates the initial and goal views used for the test. The evolution of target appearance is summarized in Fig. 6.11(b). Fig. 6.12(a) shows both the planned and the actual trajectories of the imaged vertexes of the reference quadrangle. Noisy trajectories reach the goal points with negligible error. Inspection of Fig. 6.12(b), reporting the time evolution of the eight components of the image error $\mathbf{e}$, provides an insight into task accuracy. Camera velocities during task execution are shown in Fig. 6.13.

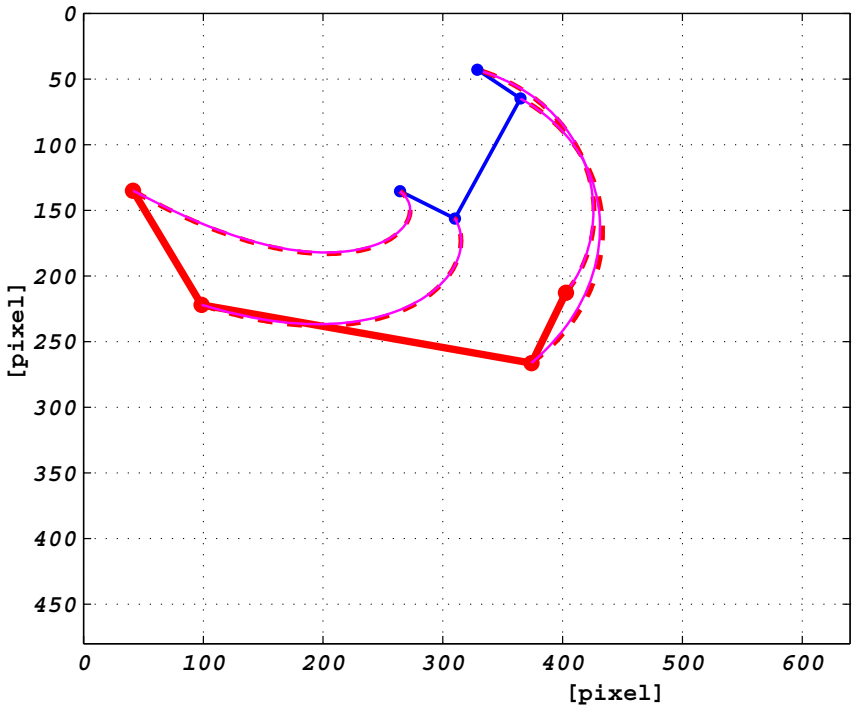


(a)

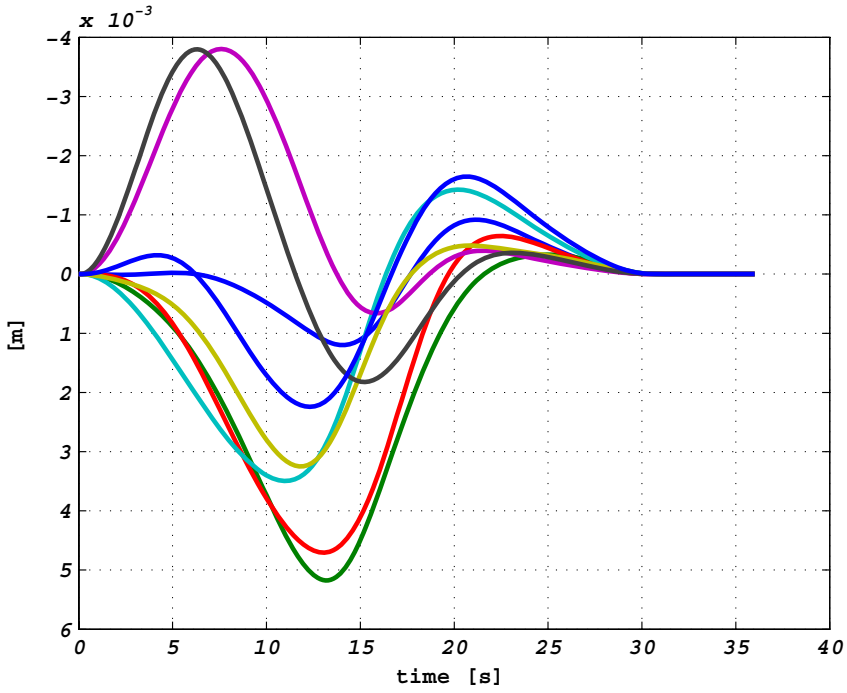


(b)

Figure 6.11: (a): Initial (thicker line) and final views for the servoing task considered in the off-line noise case. (b): Evolution of target appearance.



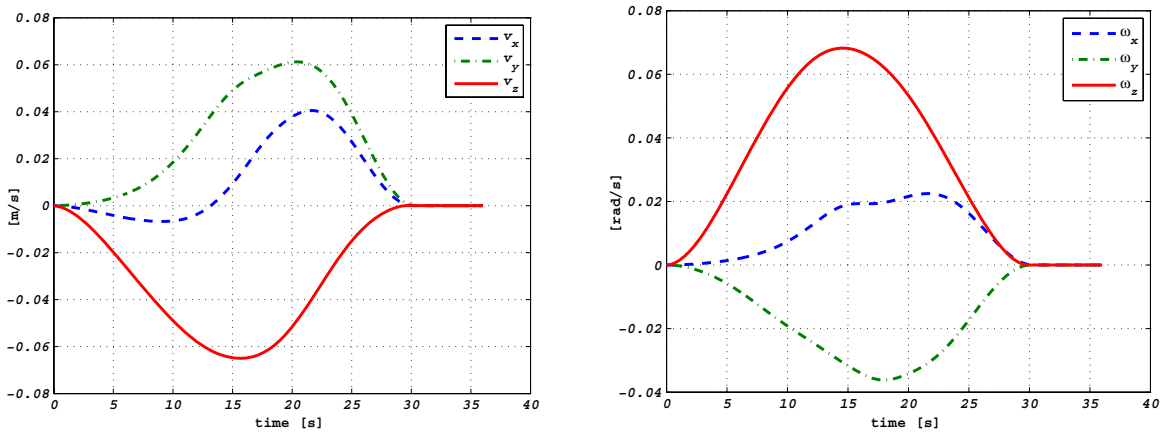
(a)



(b)

Figure 6.12: (a): Planned *vs* actual trajectories of the vertexes of the imaged quadrangle in the off-line noise case. Thicker lines refer to the initial view. (b): The eight components of the image error **e**.

Velocity profiles are smooth and physically feasible (in particular, with zero velocity at the initial and final task times), thanks to the choice of an appropriate quintic polynomial for the planning function $s(t)$.



(a) (b)

Figure 6.13: Camera velocities. (a): Translational camera velocities $\boldsymbol{v}$. (b): Angular camera velocities $\boldsymbol{\omega}$.

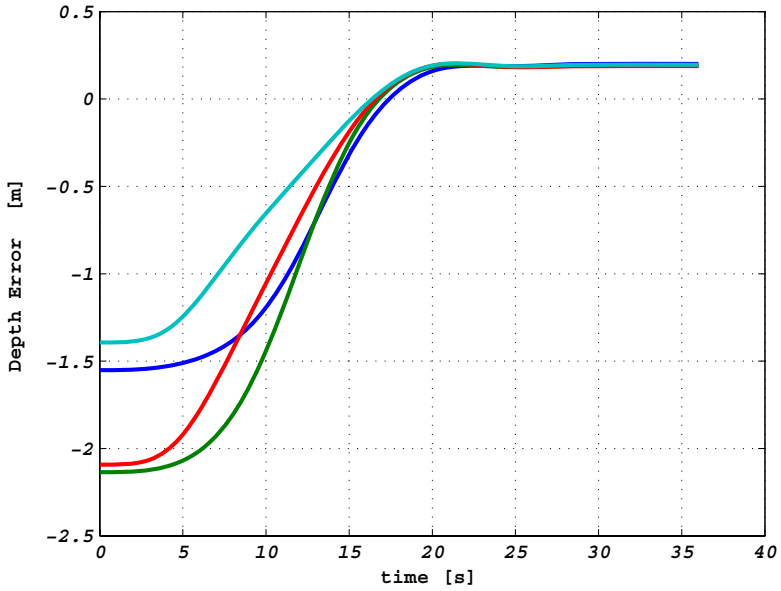


Figure 6.14: Adaptive depth estimation error.

Fig. 6.14 reports on adaptive depth estimation errors during task execution for the four quadrangle vertexes. Depth values have been initialized by arbitrarily fixing to unity the unknown scale factor of camera translation. Notice that, even if the depth errors are initially quite large, they quickly decrease, although not zero, due the properties of the gradient adaptation law. As a result of depth adaptation, the unknown scale factor can be eventually estimated, making it possible to get an estimate of the real size (e.g., in

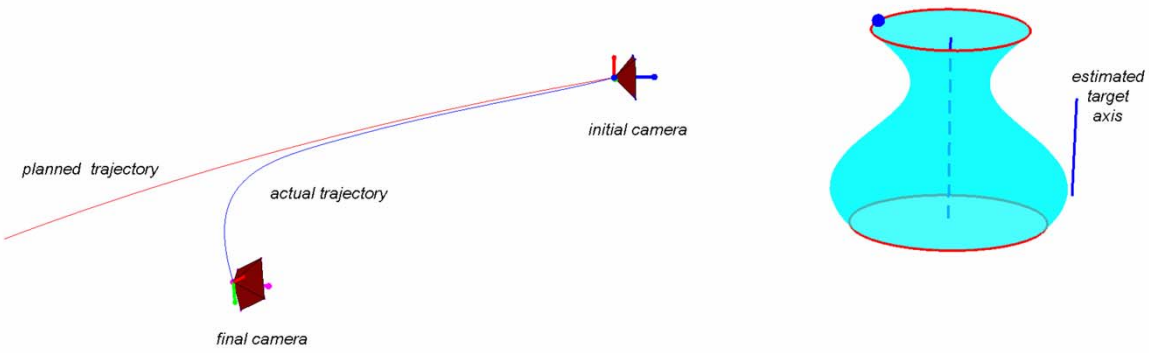


Figure 6.15: Planned *vs* actual 3D camera trajectories in the off-line noise case.

millimeters) of the target—a fact which is impossible to obtain using standard computer vision multi-view approaches with any number of target images.
Fig. 6.15 shows the planned and actual trajectories of the camera. Regardless of the noisy planning trajectory due to gross errors in the estimate of the scaled target axis, the final camera position attained at the end of the actual trajectory has been found to exactly correspond to the ground truth. An interpretation of this is that the errors in the Euclidean homography `H` induced by a wrong estimate of `K` are such that the projective homography `G` is anyway correct. This is one of the beneficial effects of closing the control loop in the image.

6.1.2.2 On-line and off-line noise

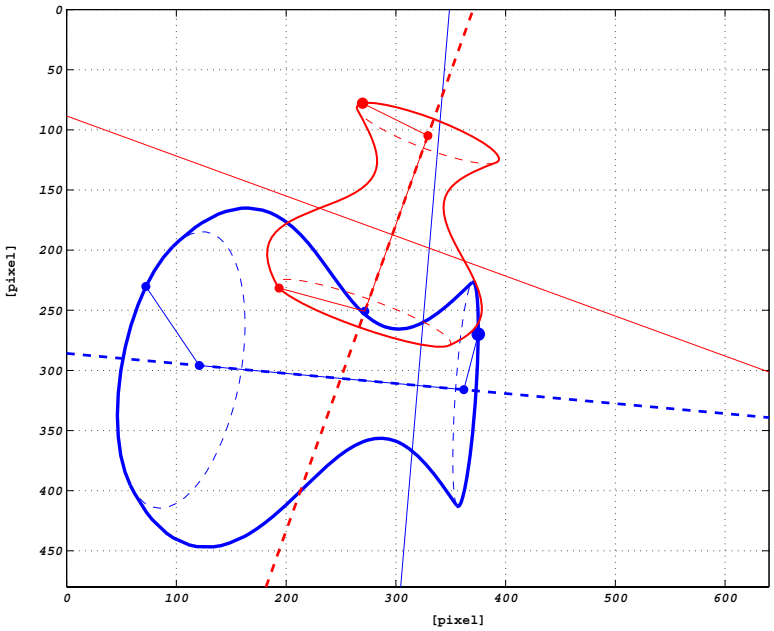
The second test reports on the effects of the combination of off-line and on-line noise on task accuracy. The noise standard deviation was 0.2 pixel. The estimated calibration matrix is in this case

$$\hat{\mathtt{K}} = \begin{bmatrix} 517.3 & 3.1 & 322.7 \\ & 470.9 & 212.3 \\ & & 1.0 \end{bmatrix}, \tag{6.4}$$

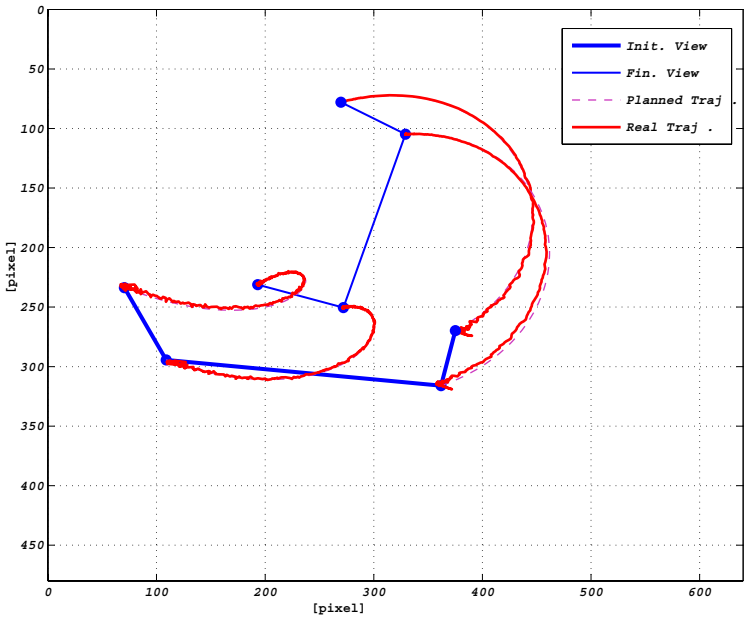
which is affected by a larger error than in the previous case.

Fig. 6.16(a) shows the task at hand, and Fig. 6.16(b) shows the image trajectories for the quadrangle vertexes. Despite the jittering, which is due in part to the use of ideal actuators with zero response time, actual trajectories follow quite well the planned trajectories. Fig. 6.17 shows the discrepancy between the planned and the actual 3D camera trajectories. The same comments of Fig. 6.15 apply here. The 3D error with respect to the overall camera displacement is about 3.6% for translations, and 1.8% for rotations.

In order to test the sensitivity of the approach to image noise, a Monte Carlo test has also been performed, for noise standard deviation ranging from 0 to 1 pixel. For each noise level, 1000 trials were executed. Fig. 6.18 shows the average and standard deviation of the 3D camera positioning error on both the translational and rotational components. Results are shown in percentage with respect to the ground truth camera displacement between the two views. Errors almost linearly increase with noise values. Even if they show the same qualitative behavior, translational errors are larger than rotational ones; this is due to the fact that, since rotations do not actually depend on depth estimates—see



(a)



(b)

Figure 6.16: (a): Initial and final views for the servoing task in the off-line/on-line noise case. (b): Planned *vs* actual trajectories of the vertexes of the imaged quadrangle. Thicker lines refer to the initial view.

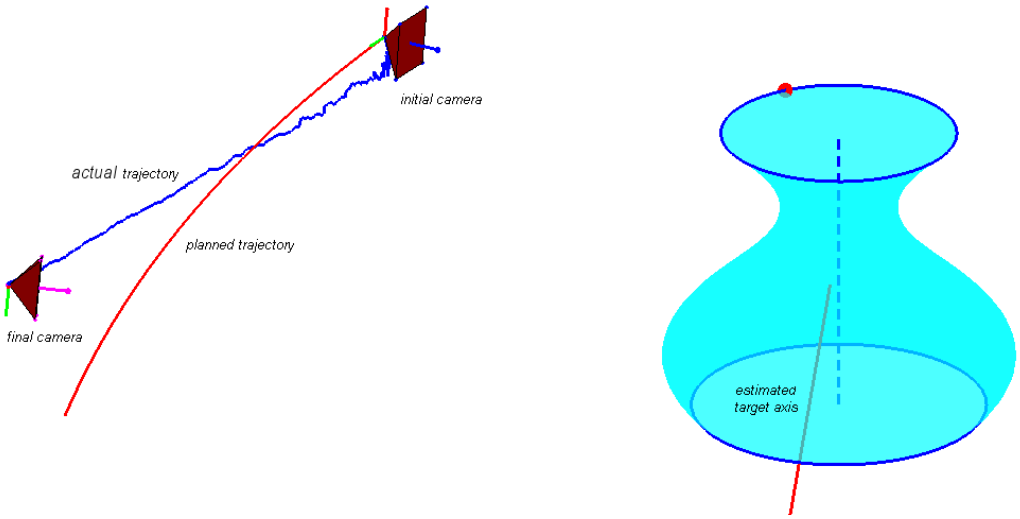
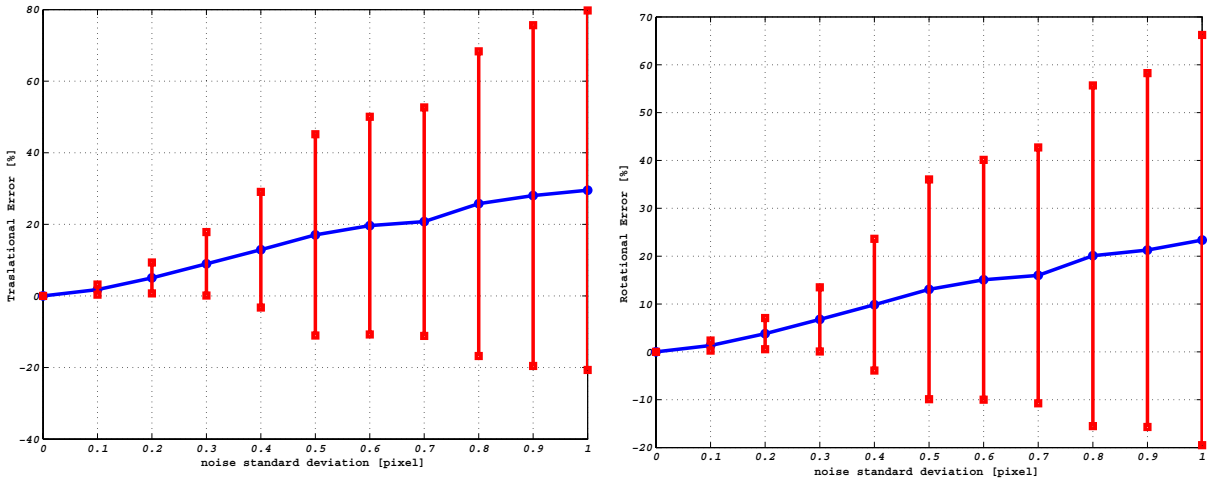


Figure 6.17: Planned *vs* actual 3D camera trajectories in the off-line/on-line noise case.

eq. 4.21—depth estimation errors only affect translational velocities.



a) b)

Figure 6.18: Positioning error *vs* noise standard deviation. The mean is represented by a polygonal line; the standard deviation by vertical bars of width 2σ centered around the mean value. Translational and rotational errors are respectively reported in $[m]$ and $[deg]$ with respect to ground truth values. (a): Translational error. (b): Rotational error.

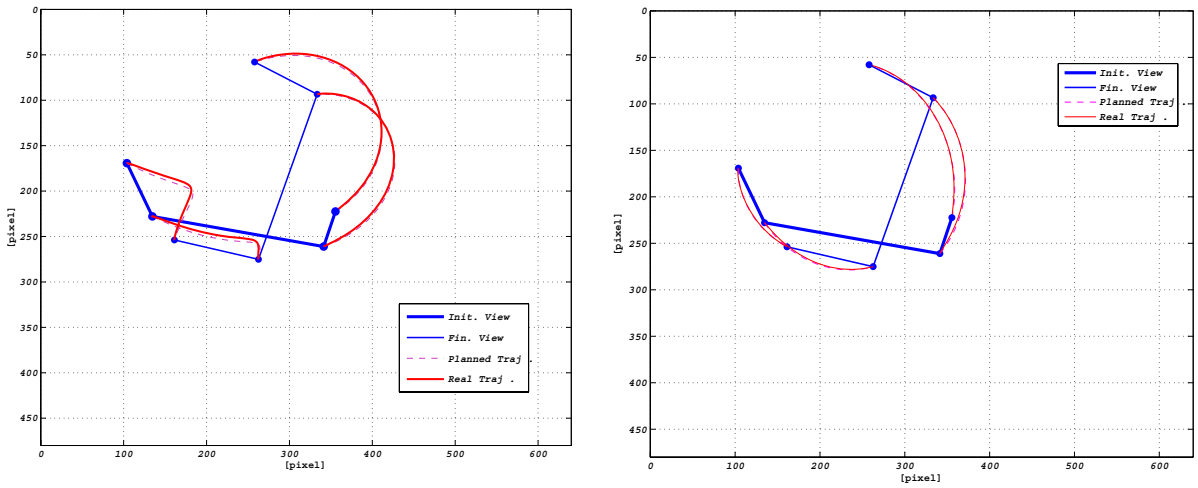


Figure 6.19: Actual *vs* planned imaged trajectories (thicker lines refer to the initial view) using different helicoidal axes. (a): Target axis. (b): Instantaneous rotational axis.

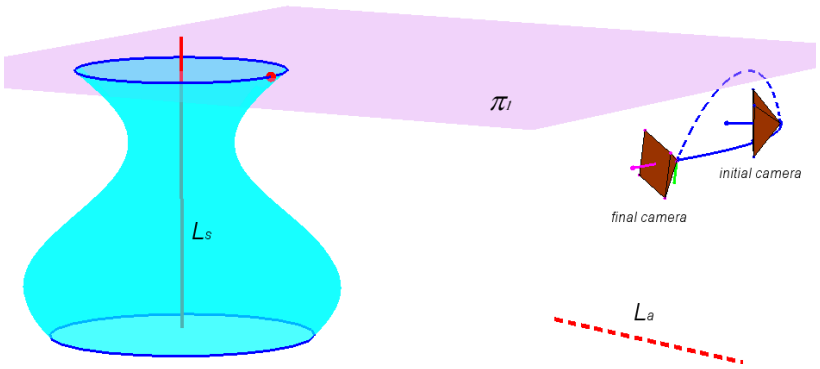


Figure 6.20: Helicoidal trajectories for two different axis choices. The solid line refers to the target axis choice.

6.1.2.3 Choice of the helicoidal axis

A noiseless test has also been performed in order to compare the two different choices of the helicoidal axis—namely, target axis L_s, and instantaneous rotation axis L_a—in off-line path planning mentioned earlier in the paper. Fig. 6.19 shows the different 2D imaged trajectories arising in the two cases, while Fig. 6.20 shows the corresponding 3D camera trajectories. It is worth noticing that, although the choice of the target axis as the helicoidal axis gives rise to a 2D trajectory which is apparently unnatural, the corresponding 3D helix has a larger radius than the other, thus requiring smaller accelerations for the actuators. Moreover, the 3D trajectory generated by the instantaneous rotational axis choice gets closer to the singularity plane.

6.2 Experiments

To perform the experiments 6 DOF Kuka© manipulator has been used with a IEEE 1394 firewire camera mounted on its end effector. Control software components has been developed in OROCOS (Open RObot Control Software) a C++ open source tool developed by the Autonomous Compliant Motion Group of the Mechanical Engineering Department in the Katholieke Universiteit of Leuven (Belgium) [Bru01], [Soe06].
The control software has been compiled on a real time platform with RTL installed (Real Time Linux), driving the input for the robot actuators. Feature point matching and tracking has been performed by using Intel© OpenCV computer vision library [Ope01].
The target was defined by 4 well-detectable points at the vertexes of a ($0.09\text{m} \times 0.09\text{m}$) square. A first set of experiments have been devoted to test the software implementation correctness and the overall system behavior as a consequence of camera intrinsic and extrinsic (evaluated with respect to the end effector frame E) estimated parameters. In a second set of experiments IBVS performances with the proposed image path planning strategy has been fully investigated. A third set of experiment has also carried out by selecting four feature points from more "natural" targets. System robustness with respect to camera calibration error has also been tested in a a final set of experiments.
The visual control (point tracking, $\boldsymbol{e}$ and $\boldsymbol{w}$ computations) has performed at a frequency of 60 Hz (the maximum frame-rate available from the camera) while the robot actuators controller run at frequency of 100 Hz (faster inner feedback control-loop) according to the static look and move architecture described in section (4.1).
The firewire camera has calibrated using the Matlab© Camera Calibration toolbox developed by Jean Yves Bouguet from California Institute of Technology [Bou07].
The resulting intrinsic camera parameters are the following:

$$\boldsymbol{K} = \begin{bmatrix} 1120.10 & 0 & 342.05 \\ 0 & 1120.19 & 199.57 \\ 0 & 0 & 1 \end{bmatrix}, \tag{6.5}$$

with a CCD of resolution 640×480. The positioning task initial and goal system configurations are described by the end effector poses E defined with respect to the ground frame W of the manipulator. Let be thus (t_x, t_y, t_z), the vector, expressed in meters-[m], identifying E origin with respect to W and (ϕ, θ, ψ), the three Roll-Pitch-Yaw angles (RPY), expressed in degrees-[deg], identifying E attitude with respect to W.
To evaluate the positioning performances the attitude and the translational errors at the end of the task have been reported. The translational error is defined as the differences in [mm] between the goal and the reached end effector translation vectors, respectively $\boldsymbol{t}_f$ and $\boldsymbol{t}_r$ expressed in W:

$$\boldsymbol{e}_p = \boldsymbol{t}_f - \boldsymbol{t}_r = [e_{p\,x},\, e_{p\,y},\, e_{p\,z}]^T \quad . \tag{6.6}$$

The attitude error defined as the difference in [deg] between the goal and the reached RPY rotation angles vectors, respectively $\boldsymbol{\phi}_f$ and $\boldsymbol{\phi}_r$ of E frame expressed in W:

$$\boldsymbol{e}_a = \boldsymbol{\phi}_f - \boldsymbol{\phi}_r = [e_{a\,\phi},\, e_{a\,\theta},\, e_{a\,\psi}]^T \quad . \tag{6.7}$$

In (6.2.1) a first experiment without planning and small camera displacement is shown. In (6.2.2) we report the results concerning experiments executed with the proposed path planning mechanism with respect to an "ad hoc build" planar target as defined above. In section (6.2.3) we report two experiments done with more "natural" targets show as to demonstrate the generality of the approach. Finally in (6.2.4) the system robustness with respect to camera calibration error is proved. Experiments exhibit the very good performances of the planning approach despite the change the changing of the camera poses and the large displacements of the positioning tasks. Some problem arises when the light reflected from the target surface causes feature disappearing: in such a case the strategy referenced in (4.1.3.2) could be used to ensure satisfactory performances as well. The system shown hereafter has resulted to be robust up to the 40% of error in the entries of the camera calibration matrix $\boldsymbol{K}$.

6.2.1 First Trial without Path Planning

6.2.1.1 Experiment I

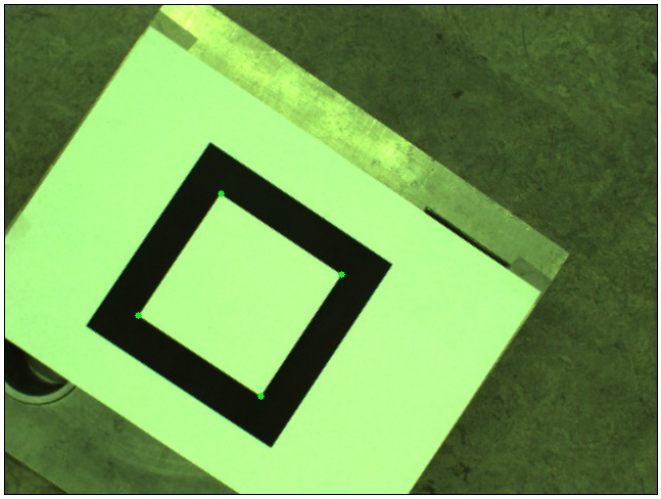
In this first experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{xi} = -0.9$	$t_{yi} = -0.4$	$t_{zi} = 0.9$	$[m]$
$\phi_i = -89.4$	$\theta_i = 17.8$	$\psi_i = 34.3$	$[deg]$

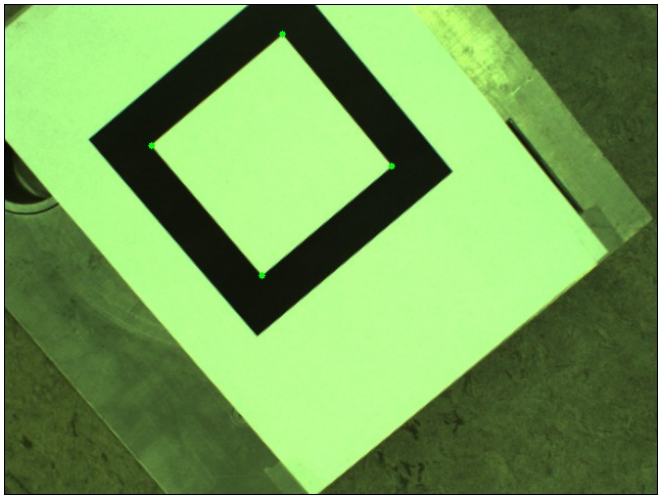
The final end effector pose E_f is instead defined with respect to W by:

$t_{xf} = -0.9$	$t_{yf} = -0.4$	$t_{zf} = 0.8$	$[m]$
$\phi_f = -95.1$	$\theta_f = 18.79$	$\psi_f = 48.58$	$[deg]$

Figure (6.21) shows the initial and final target views (I_i and I_f) grabbed from the camera;



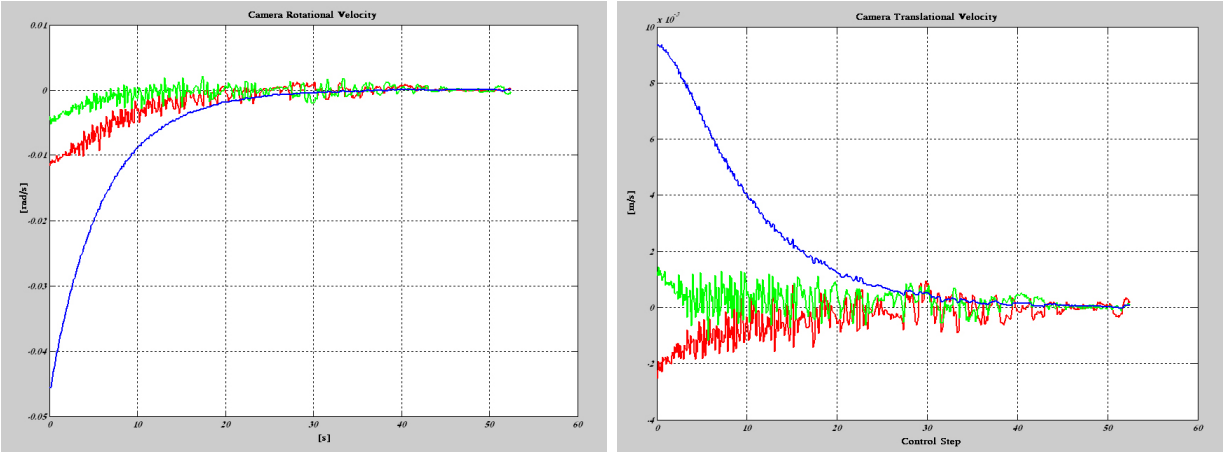
a)



b)

Figure 6.21: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.22) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control law (4.32) without the feed-forward term derived from path planning ($\dot{\boldsymbol{x}}_d = \mathbf{0}$).



a) b)

Figure 6.22: a) Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$. b) Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$.

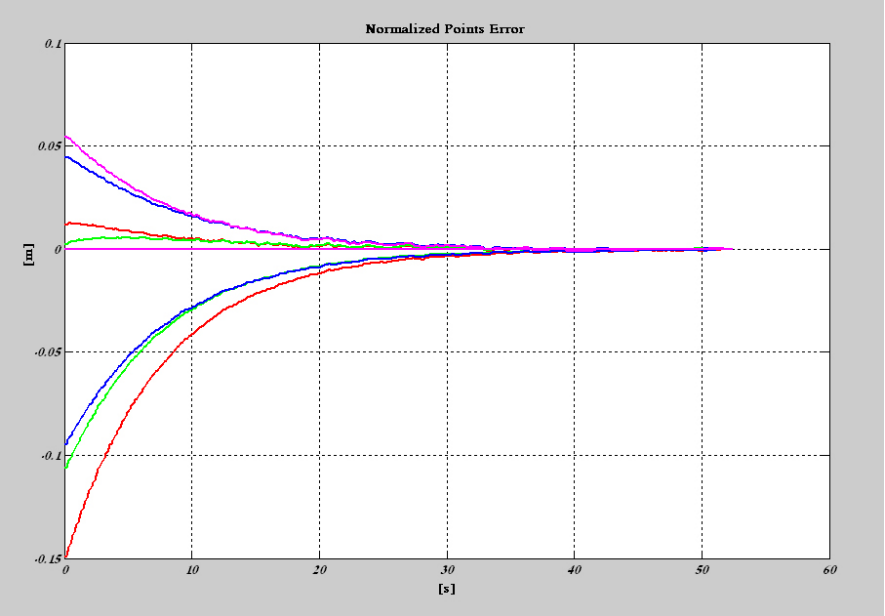


Figure 6.23: The eight normalized image error $\boldsymbol{e}$ components converge to zero inducing the camera twist screw $\boldsymbol{w}$ of figure (6.22).

The $3D$ end-effector origin trajectories measured during task execution is reported in Figure (6.24):the end effector reaches the goal position for this small positioning task camera displacements. In Figure (6.23) reports the image error $\boldsymbol{e}$ components As clearly

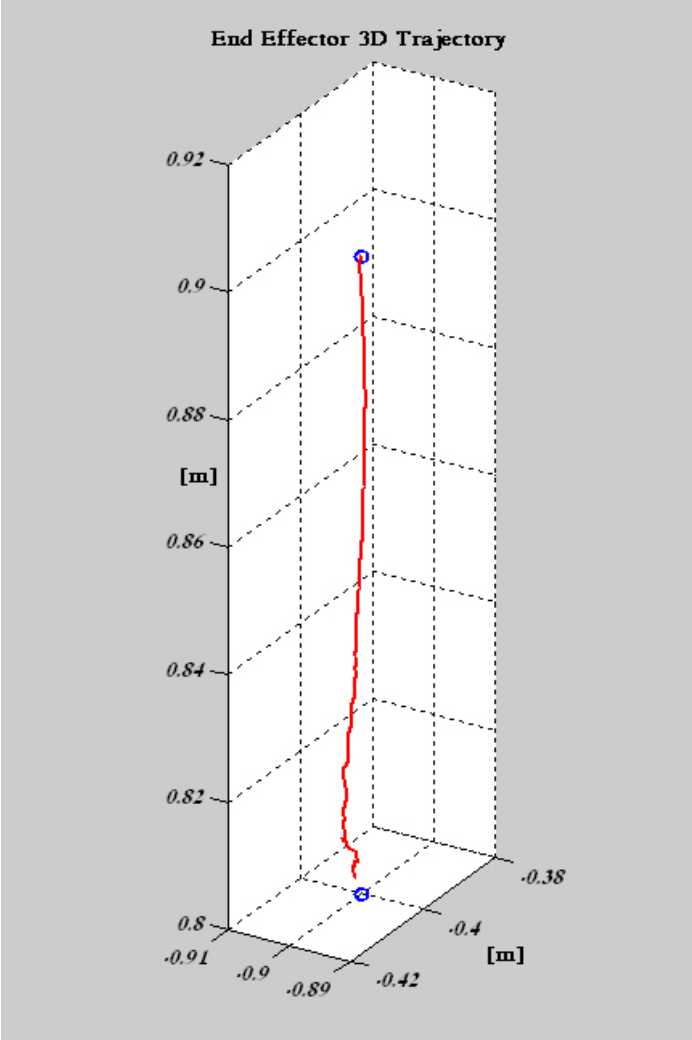


Figure 6.24: End-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (bottom green circle).

visible $\boldsymbol{e}$ components converges monotonically from the initial values to zero.

In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 2.1$	$e_{py} = 1.5$	$e_{pz} = 1.7$	$[mm]$
$e_{a\phi} = 0.127$	$e_{a\theta} = 0.110$	$e_{a\psi} = 0.206$	$[deg]$

Notwithstanding the absence of the image path planning we can notice the good performances of the control system in case of small camera displacements.

6.2.2 Experiments with " ad hoc " Target

6.2.2.1 Experiment I

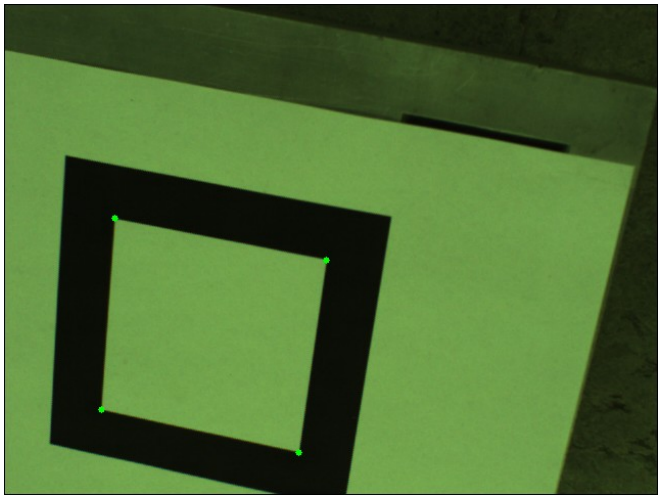
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.62$	$[m]$
$\phi_i = -112$	$\theta_i = 30$	$\psi_i = 0$	$[deg]$

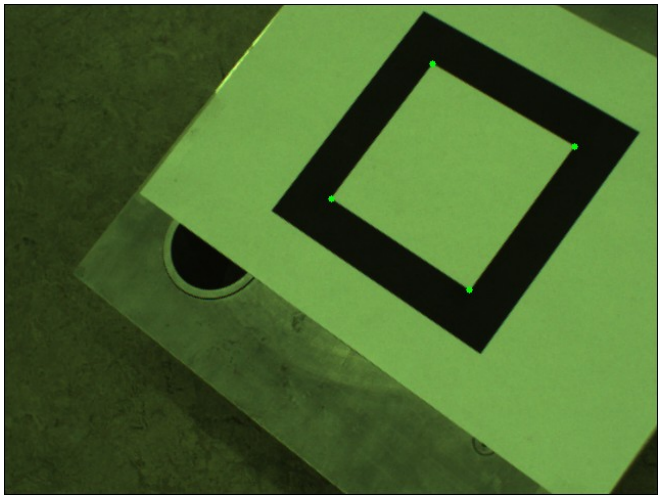
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -1.2$	$t_{y\,f} = 0.6$	$t_{z\,f} = 0.8$	$[m]$
$\phi_f = -37$	$\theta_f = -31.5$	$\psi_f = 40$	$[deg]$

Figure (6.25) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.25: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.26) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.27) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

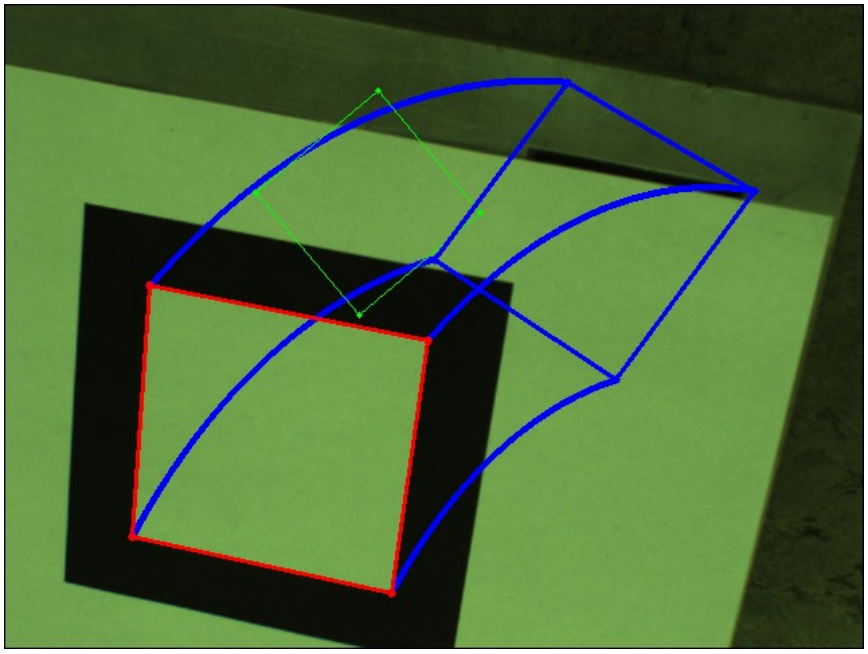


Figure 6.26: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

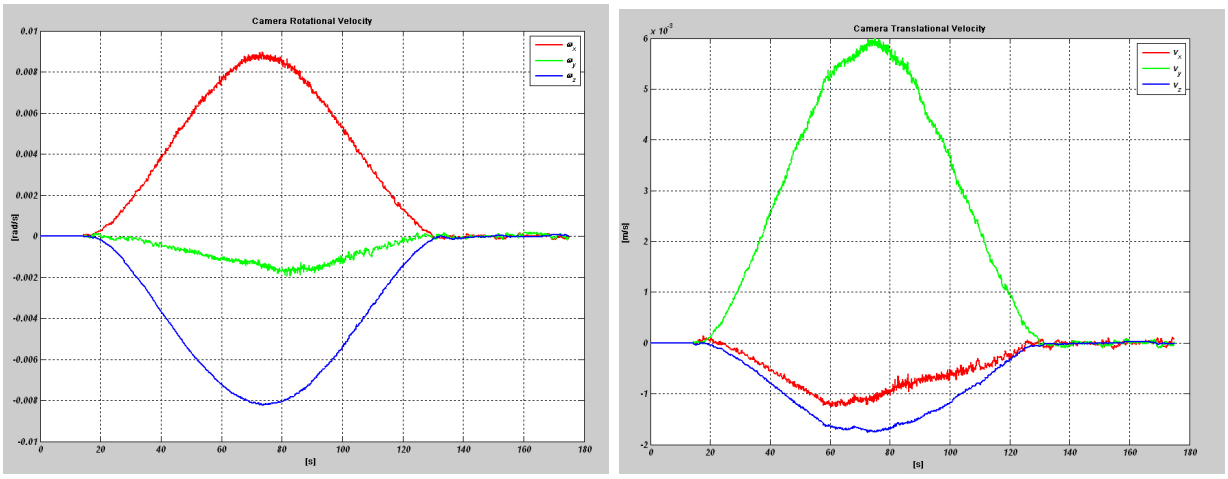


Figure 6.27: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$-left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.28): as clearly visible the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement. In Figure (6.29) a) and b) are respectively reported image error $\boldsymbol{e}$ components

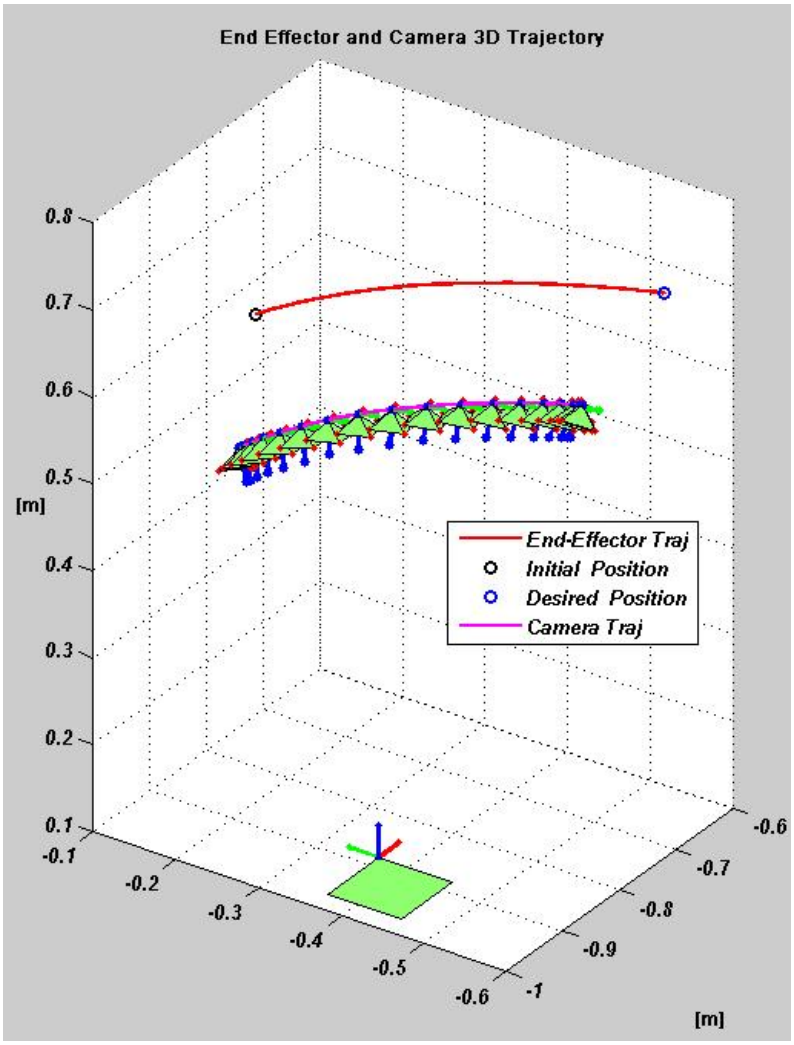
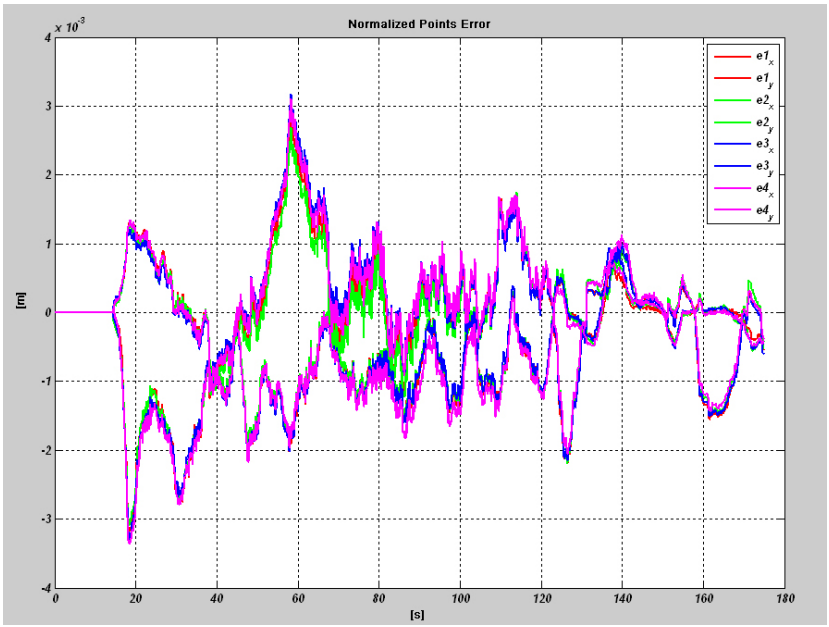


Figure 6.28: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

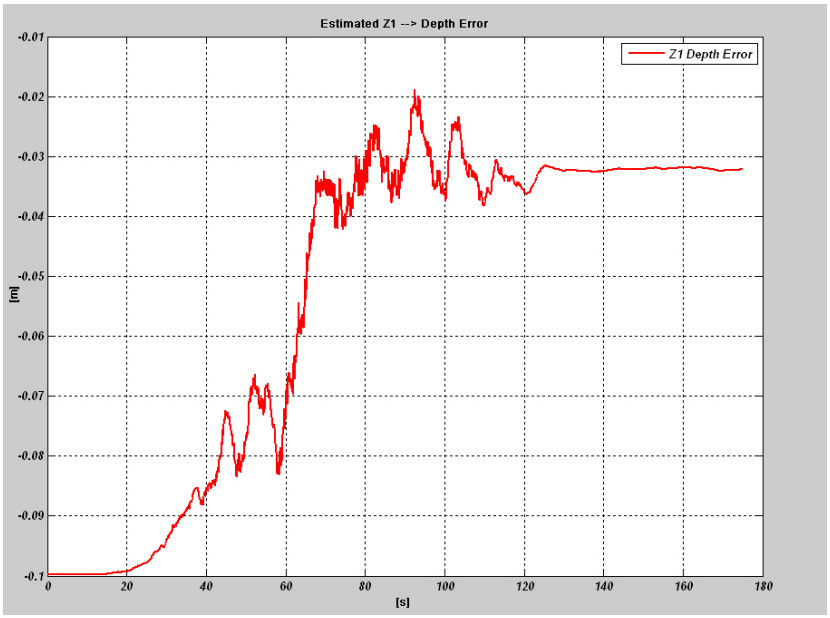
and the $3D$ depth estimation error for the first tracked feature-point. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The adaptive estimation law, as clear from figure (6.29)-b, gives noisy depth estimates but converges to a better estimation of the target point unknown depth Z_1. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 0.4210$	$e_{py} = 0.1240$	$e_{pz} = -0.3610$	$[mm]$
$e_{a\phi} = 0.12$	$e_{a\theta} = -0.01$	$e_{a\psi} = 0.01$	$[deg]$

Notice the nice performances of the IBVS system.



a)



b)

Figure 6.29: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.27). b) Depth estimation error $[m]$ on the first target point.

6.2.2.2 Experiment II

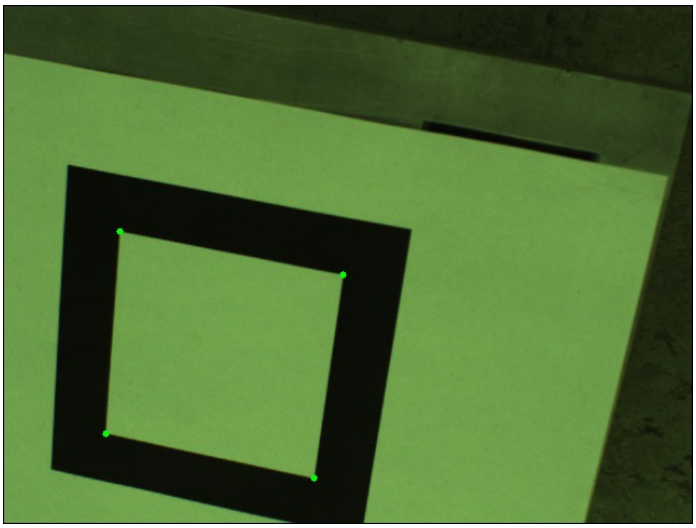
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.62$	$[m]$
$\phi_i = -112$	$\theta_i = 30$	$\psi_i = 0$	$[deg]$

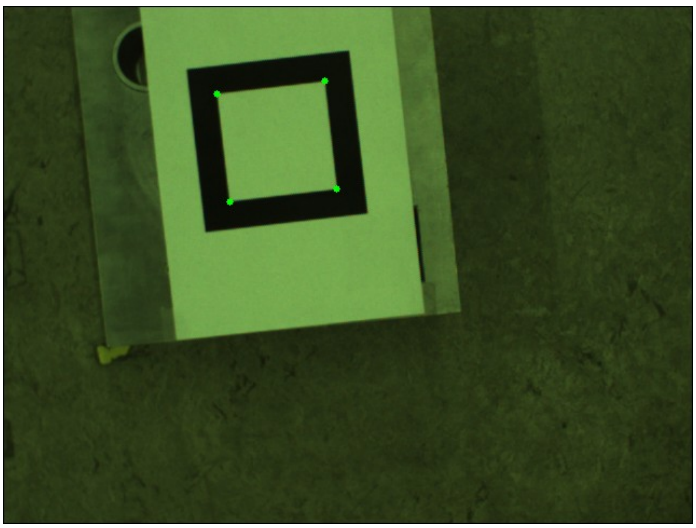
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.8$	$t_{y\,f} = -0.3$	$t_{z\,f} = 1.2$	$[m]$
$\phi_f = -91.7$	$\theta_f = 23$	$\psi_f = 83$	$[deg]$

Figure (6.30) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.30: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.31) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.32) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

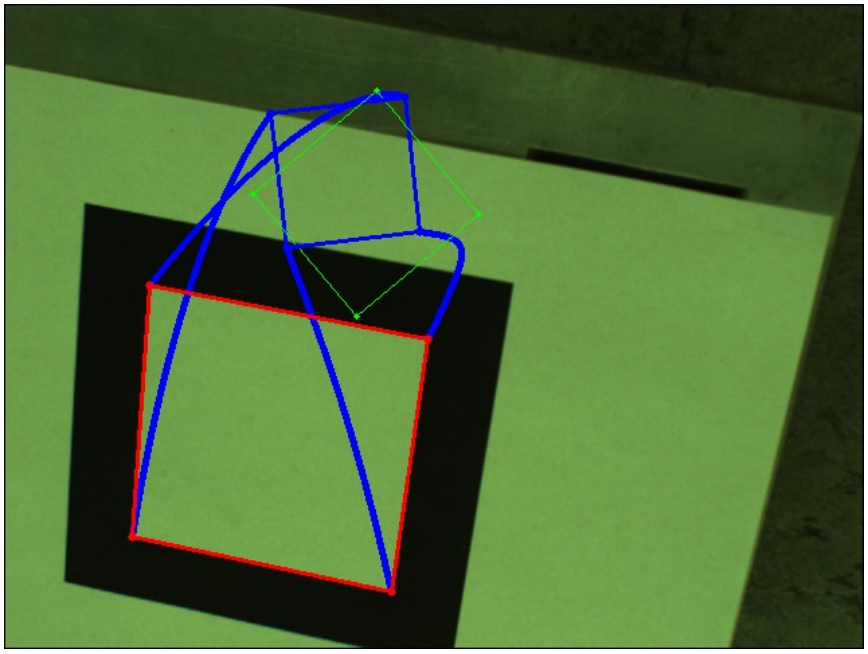


Figure 6.31: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth with increasing noise at the task end due to the decreasing size of the target image (the camera is increasing its distance from the target).

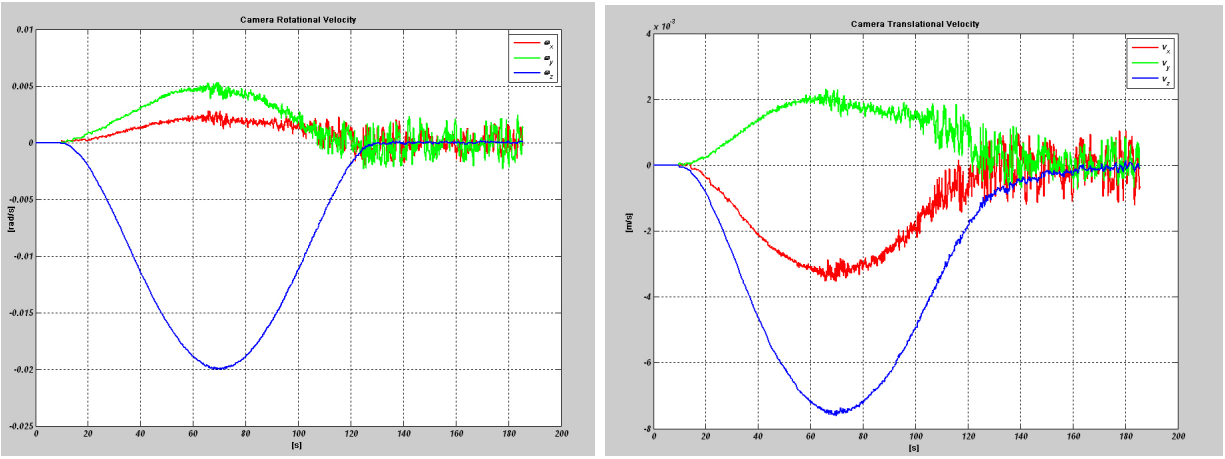


Figure 6.32: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$-left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.28): as clearly visible the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement. In Figure (6.34)-a is reported image error $\boldsymbol{e}$ components the tracked feature-

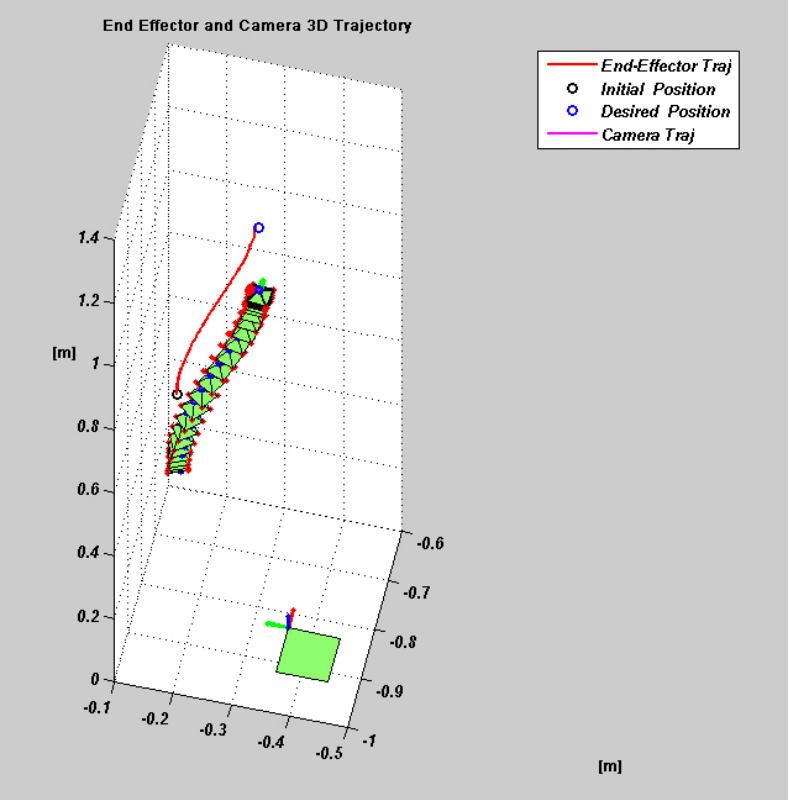
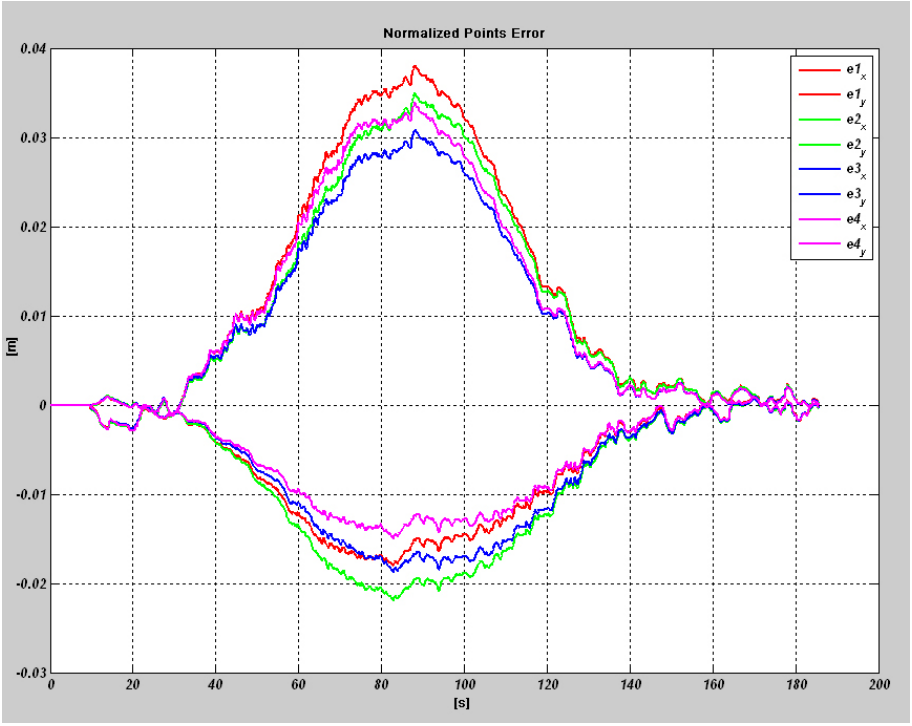


Figure 6.33: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

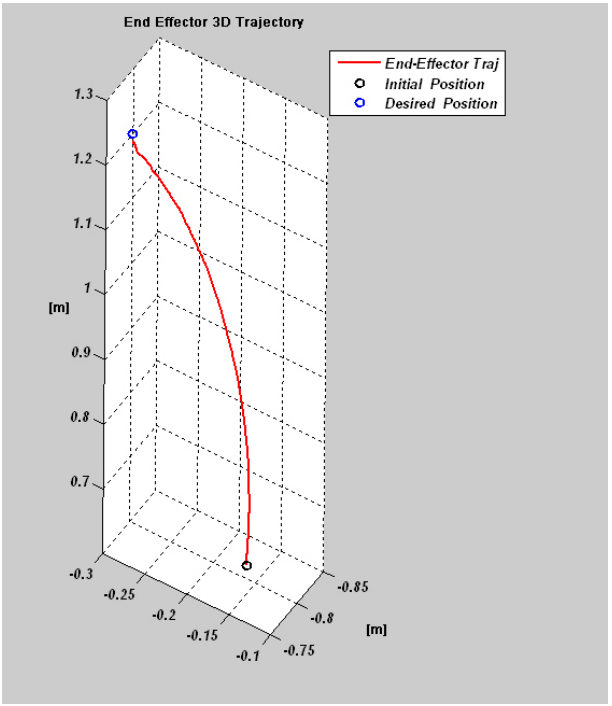
point. Normalized image error $\boldsymbol{e}$ components in this experiment grows more than in the previous one reflecting however the time law derivative $\dot{s}(t)$ shape: this is due to the larger camera displacement between the initial and the goal poses. In Figure (6.34)-b the measured end-effector $3D$ trajectory is shown: as planned through the image paths it results in an helix arc. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 8.2$	$e_{py} = 5.3$	$e_{pz} = 2.8$	$[mm]$
$e_{a\phi} = 0.13$	$e_{a\theta} = -0.28$	$e_{a\psi} = 0.46$	$[deg]$

Also in this case the system performances results satisfactory. Notice that by increasing the feature point a more accurate could be obtained.



a)



b)

Figure 6.34: a) Normalized image error $\boldsymbol{e}$ components: in this experiments the error components are bigger than the previous experiment ones: this is due to the larger camera displacement. The error evolution during time also reflect the quintic time law $s(t)$ defined for the path planning. b) End effector measured trajectory.

6.2.2.3 Experiment III

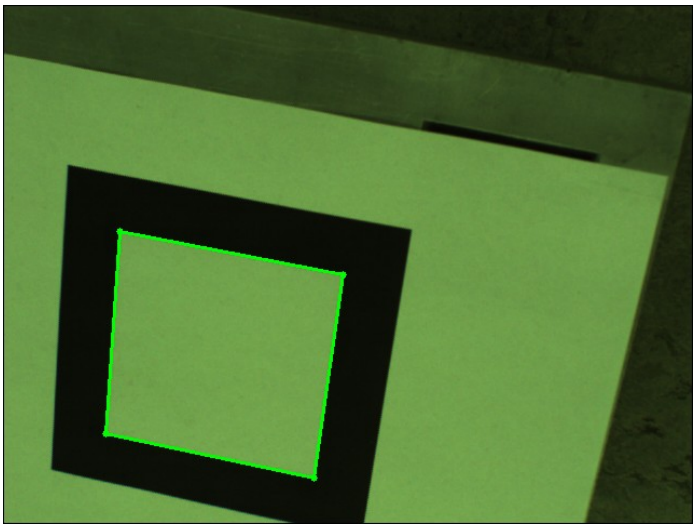
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.62$	$[m]$
$\phi_i = -112$	$\theta_i = 30$	$\psi_i = 0$	$[deg]$

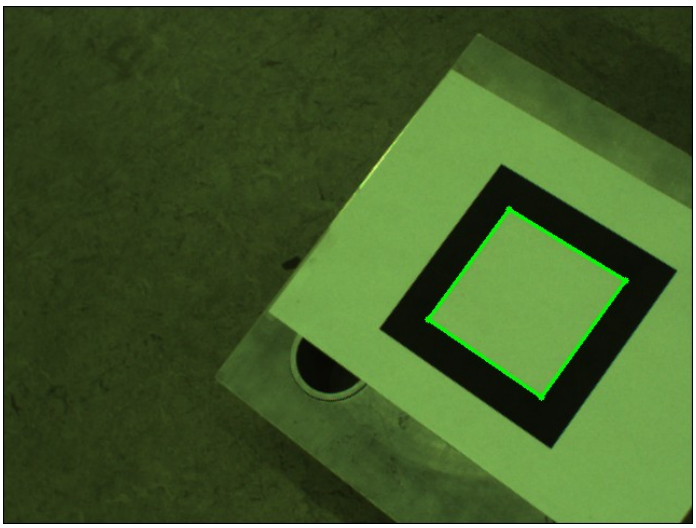
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.65$	$t_{y\,f} = -0.55$	$t_{z\,f} = 0.9$	$[m]$
$\phi_f = -68.8$	$\theta_f = 34.4$	$\psi_f = 51.6$	$[deg]$

Figure (6.35) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.35: a) Initial target view I_i b) Goal target view I_f.

Figure (6.36) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.37) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

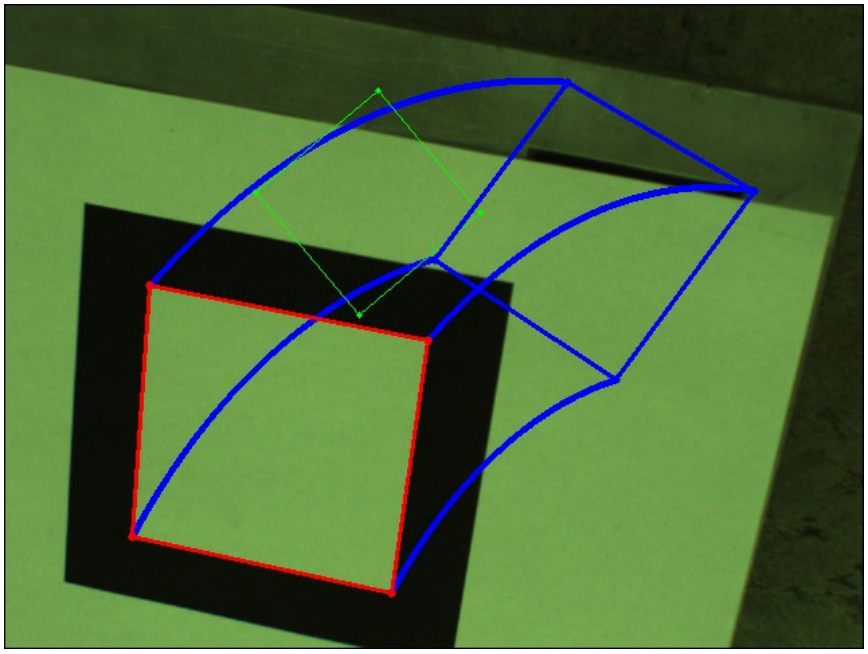


Figure 6.36: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

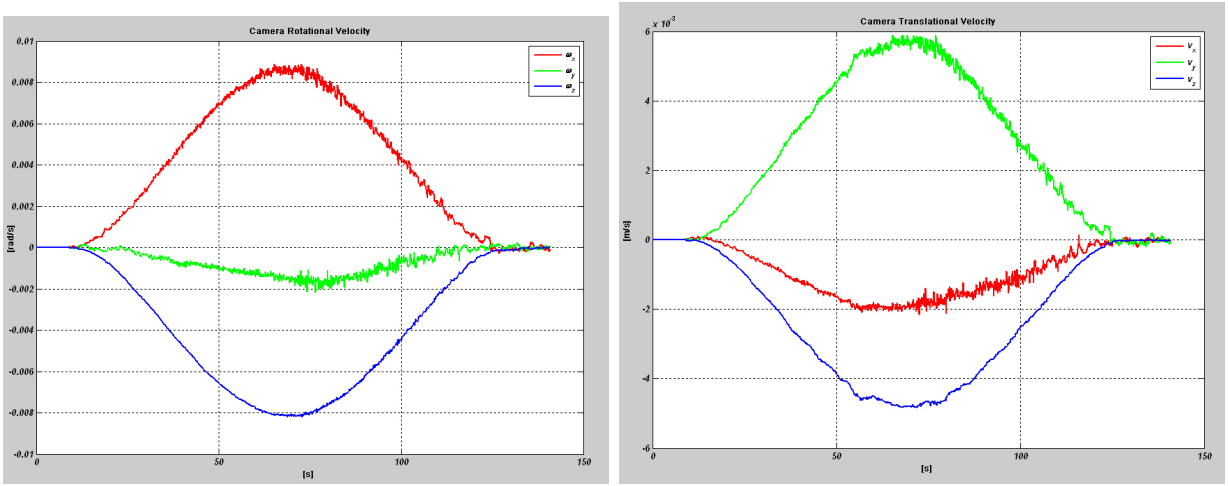


Figure 6.37: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.38): the camera moves also in this case on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$. In Figure (6.39) a) and b) are re-

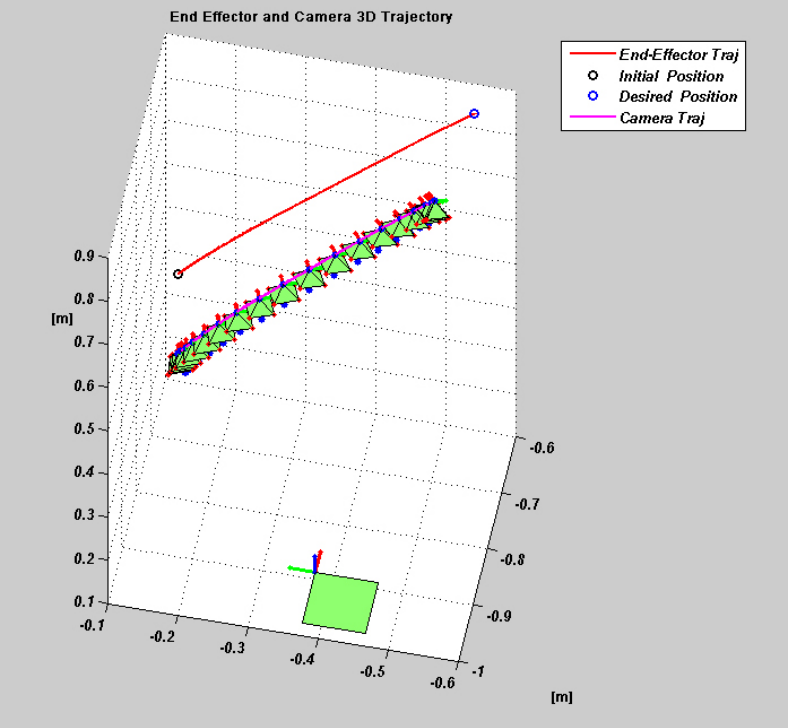
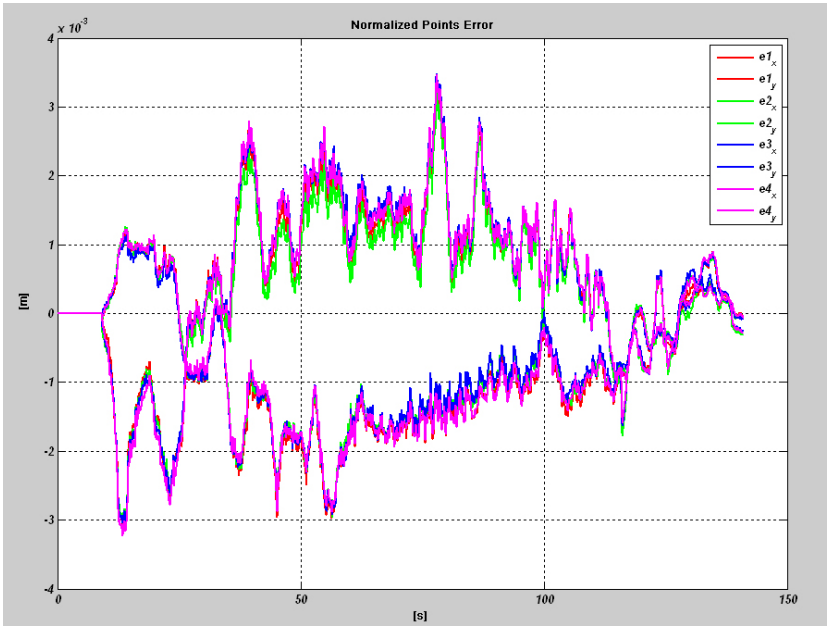


Figure 6.38: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

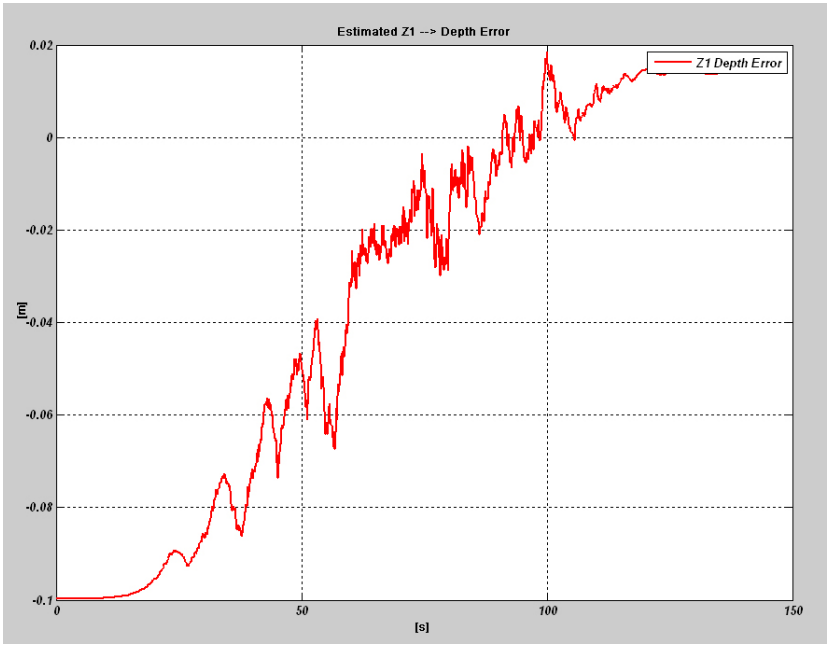
spectively reported image error $\boldsymbol{e}$ components and the $3D$ depth estimation error for the first tracked feature-point. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The adaptive estimation law, as clear from figure (6.39)-b, gives noisy depth estimates but converges to a better estimation of the target point unknown depth Z_1. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 0.09$	$e_{py} = 0.65$	$e_{pz} = 0.29$	$[mm]$
$e_{a\phi} = -0.13$	$e_{a\theta} = -0.018$	$e_{a\psi} = -0.16$	$[deg]$

Notice the very good performances of the IBVS system for this positioning task.



a)



b)

Figure 6.39: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.37). b) Depth estimation error $[m]$ on the first target point: the error decrease to a smaller value during task execution.

6.2.2.4 Experiment IV

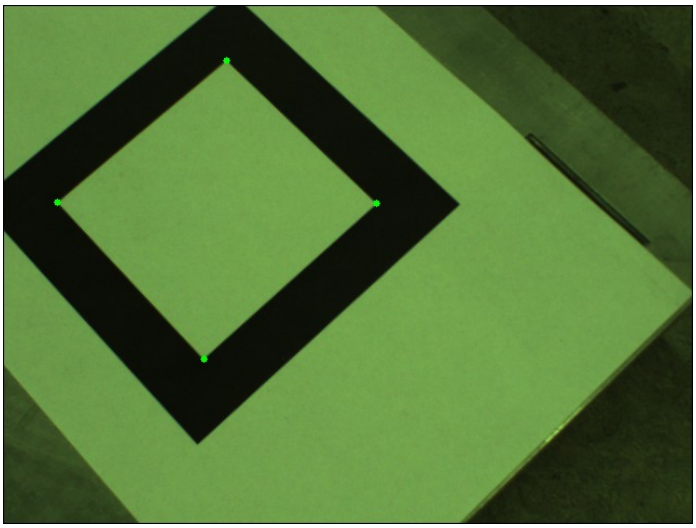
In this fourth experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.67$	$t_{y\,i} = -0.5$	$t_{z\,i} = 0.6$	$[m]$
$\phi_i = -91.7$	$\theta_i = 23$	$\psi_i = 83.1$	$[deg]$

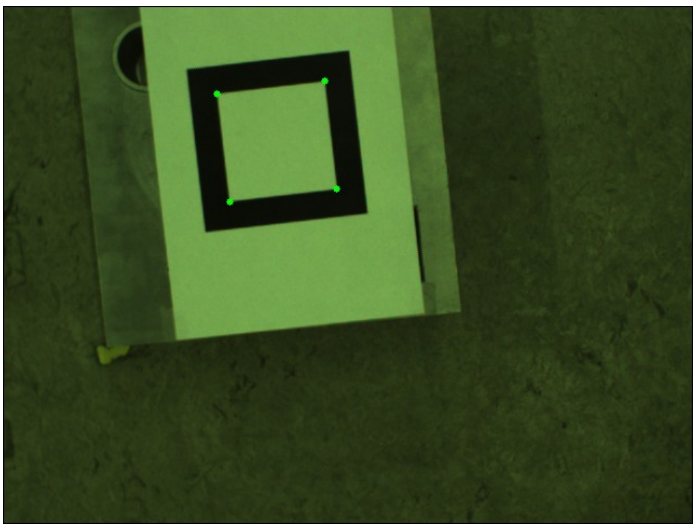
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.8$	$t_{y\,f} = -0.3$	$t_{z\,f} = 1.2$	$[m]$
$\phi_f = -68.8$	$\theta_f = 22.9$	$\psi_f = 54.4$	$[deg]$

Figure (6.40) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.40: a) Initial target view I_i b) Goal target view I_f.

Figure (6.41) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.42) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

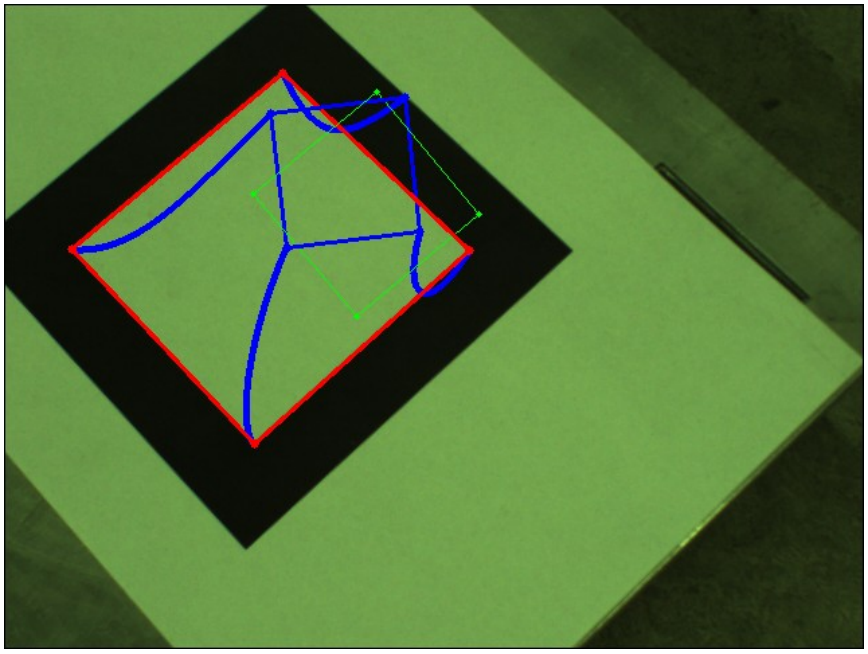


Figure 6.41: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

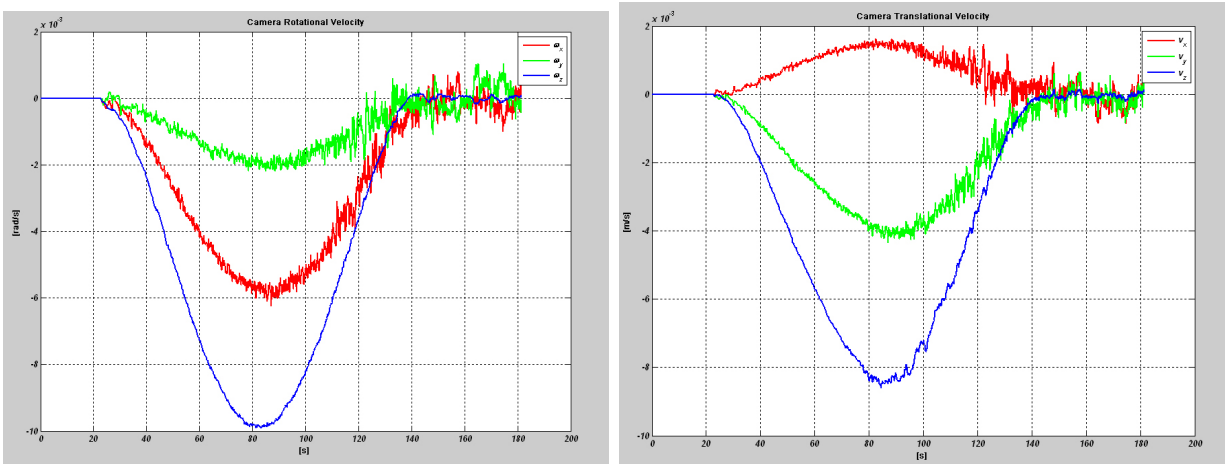


Figure 6.42: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.43): the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$. In Figure (6.44) a) and b) are respectively reported

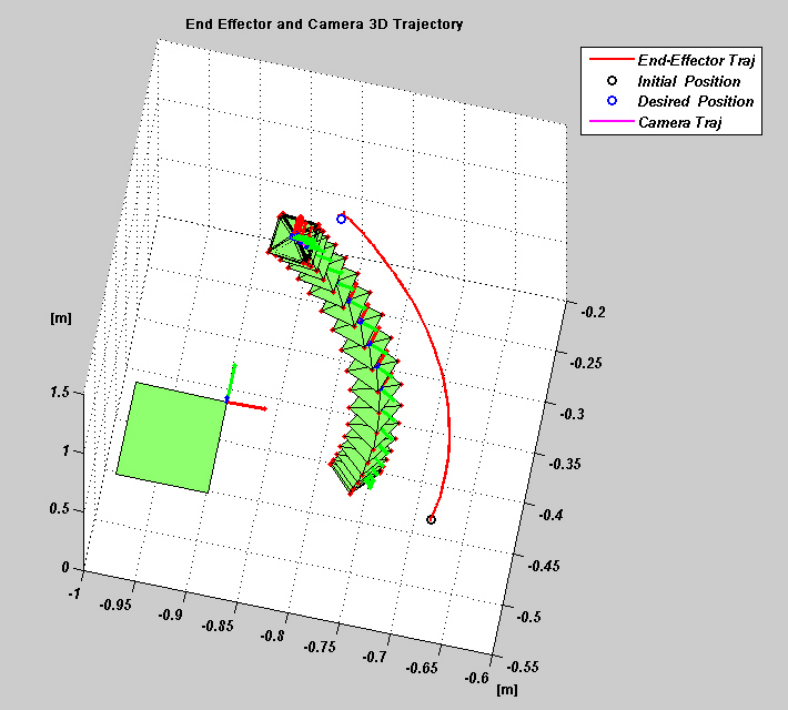
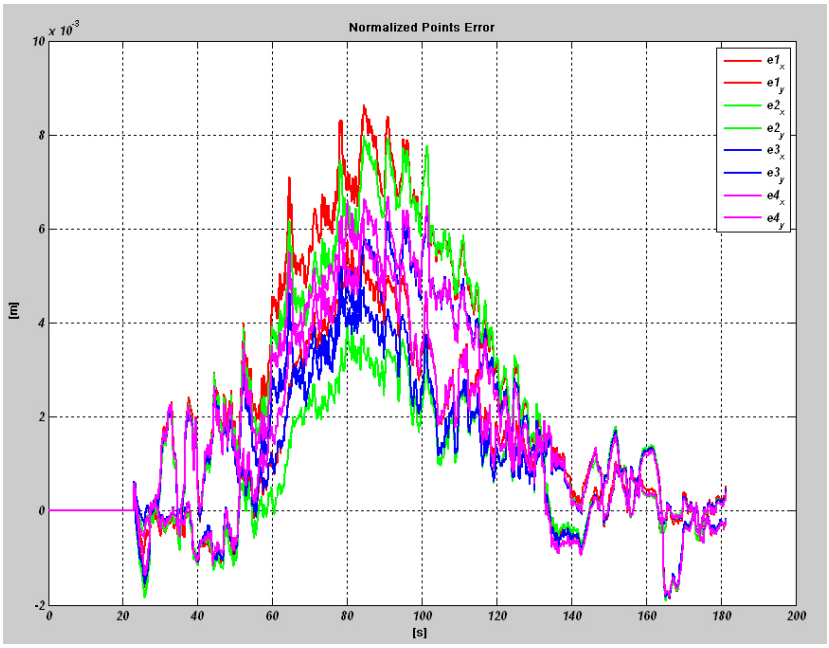


Figure 6.43: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

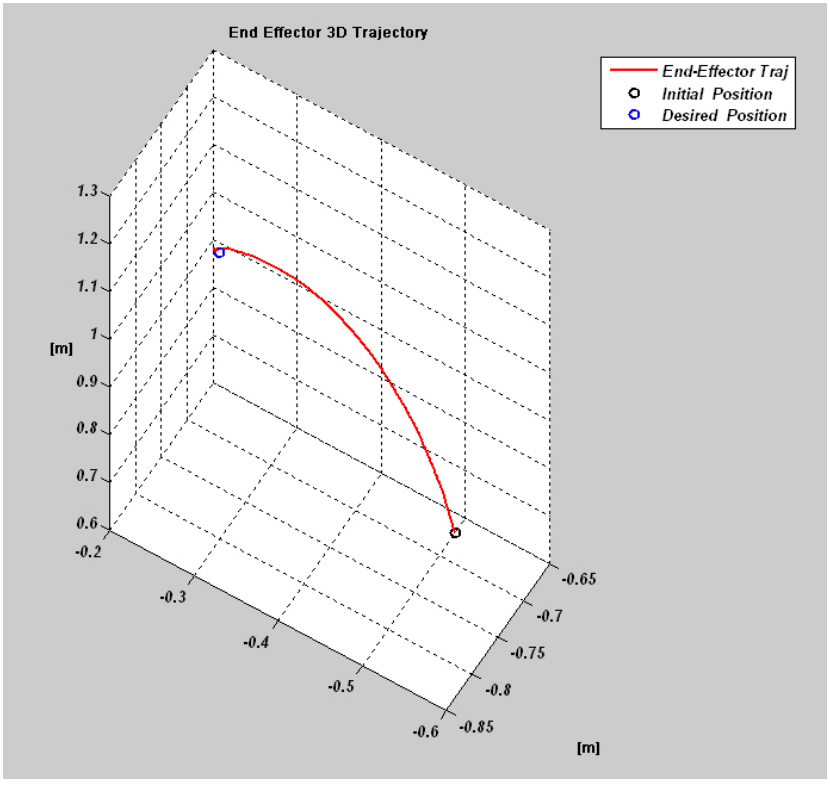
image error $\boldsymbol{e}$ components and the $3D$ end effector measured trajectory. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The $3D$ end effector trajectory follows an helix arc driving the system to the goal view. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 3.2$	$e_{py} = 3.9$	$e_{pz} = 1$	$[m]$
$e_{a\phi} = -0.069$	$e_{a\theta} = 0.265$	$e_{a\psi} = -0.131$	$[deg]$

Notice the good performances of the IBVS system for this positioning task.



a)



b)

Figure 6.44: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.42). b) End effector measured trajectory ent to the goal position (blue circle).

6.2.2.5 Experiment V

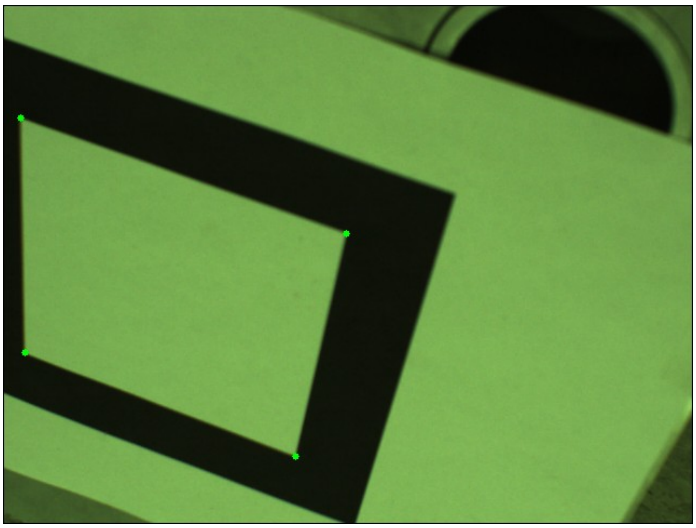
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.1$	$t_{z\,i} = 0.4$	$[m]$
$\phi_i = -137.5$	$\theta_i = 33.2$	$\psi_i = -5.7$	$[deg]$

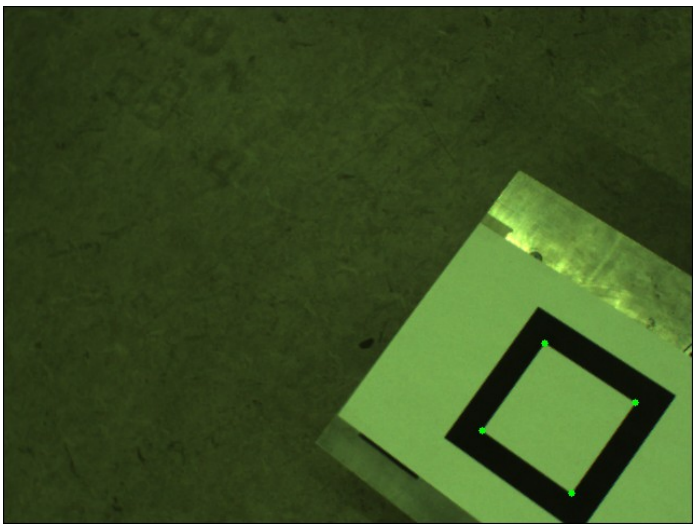
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -1.2$	$t_{y\,f} = 0.6$	$t_{z\,f} = 0.8$	$[m]$
$\phi_f = -37$	$\theta_f = -31.5$	$\psi_f = 65.9$	$[deg]$

Figure (6.45) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.45: a) Initial target view I_i b) Goal target view I_f.

Notice in this case the big differences between I_i and I_f. Figure (6.46) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.47) shows the

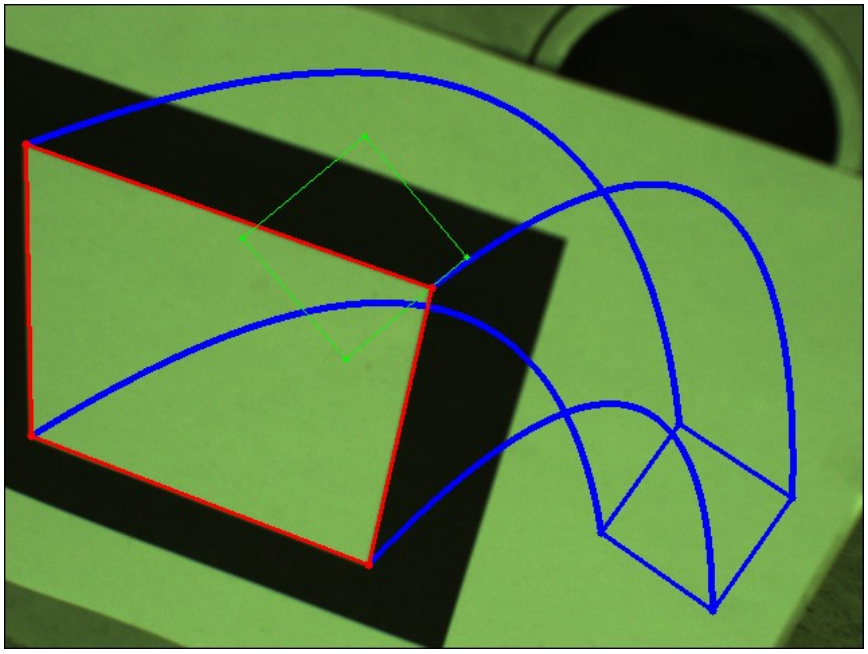


Figure 6.46: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

components of the camera twist screw $\boldsymbol{w}$ induced from the control law: they result very smooth and no noisy according to the time law $s(t)$ in section (4.3.1).

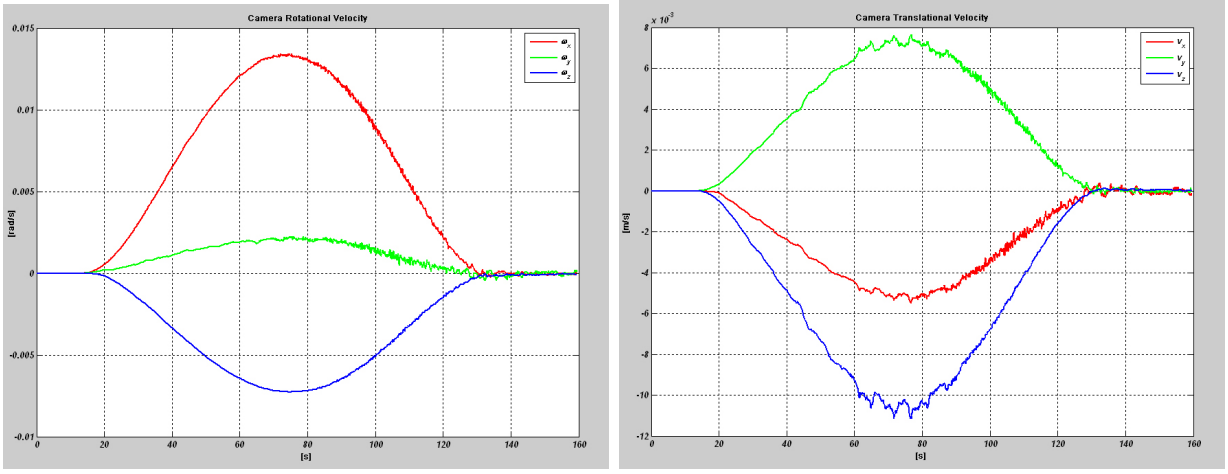


Figure 6.47: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.28): the camera is driven on a helical-shape induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement: notice the large camera displacement for this task. In Figure (6.49) a) and b) are respectively re-

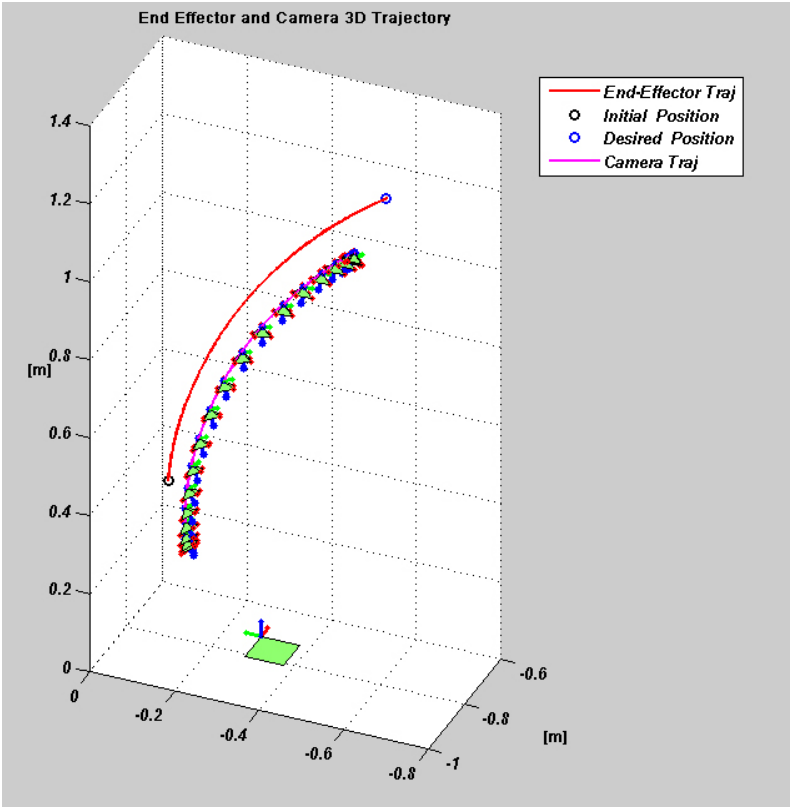
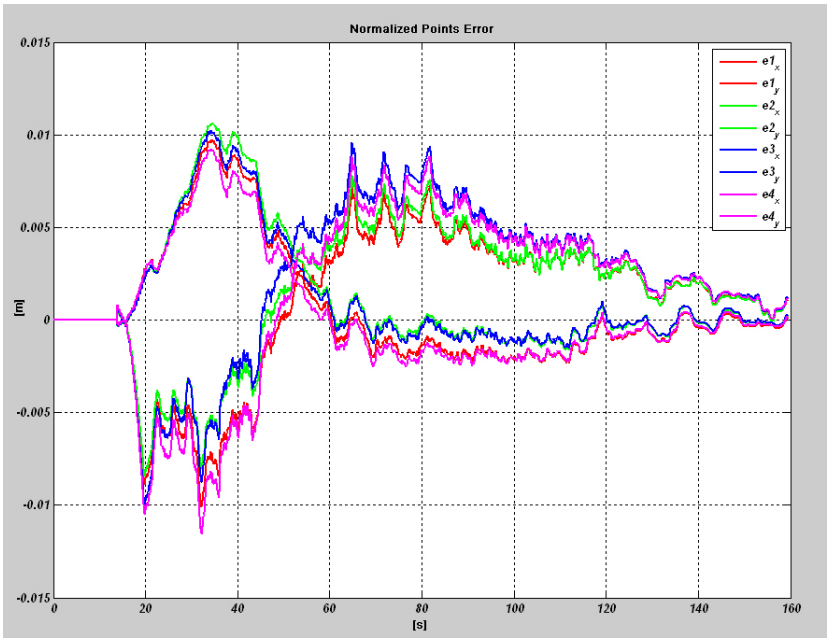


Figure 6.48: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

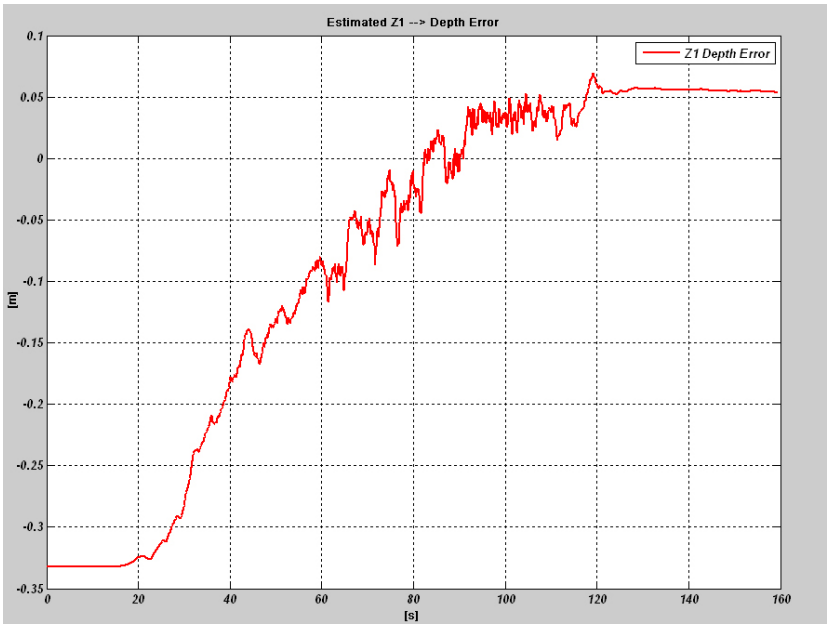
ported image error $\boldsymbol{e}$ components and the $3D$ depth estimation error for the first tracked feature-point. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The adaptive estimation law, as clear from figure (6.49)-b, gives noisy depth estimates but converges to a better estimation of the target point unknown depth Z_1. In the following table is reported the positioning task error as defined in section (6.2).

$e_{p\,x} = 3.15$	$e_{p\,y} = 3.75$	$e_{p\,z} = -1$
$e_{a\,\phi} = -0.057$	$e_{a\,\theta} = 0.165$	$e_{a\,\psi} = 0.095$

Notice the nice performances of the IBVS system also for this large camera displacement task.



a)



b)

Figure 6.49: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.47). b) Depth estimation error $[m]$ on the first target point.

6.2.2.6 Experiment VI

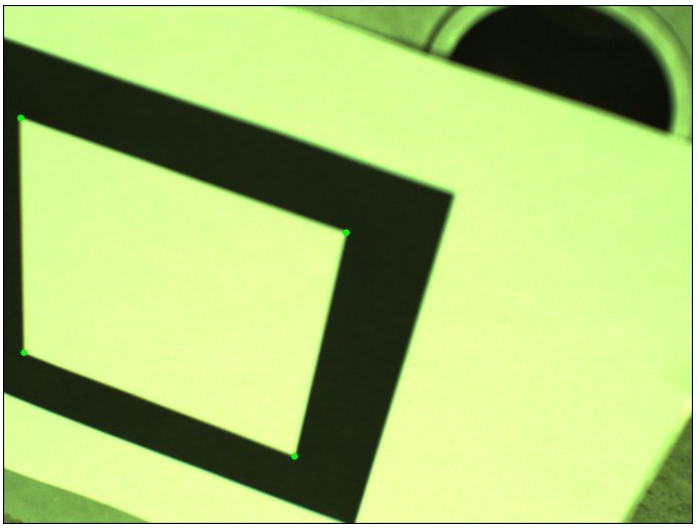
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.1$	$t_{z\,i} = 0.4$	$[m]$
$\phi_i = -137.5$	$\theta_i = 33.2$	$\psi_i = -5.7$	$[deg]$

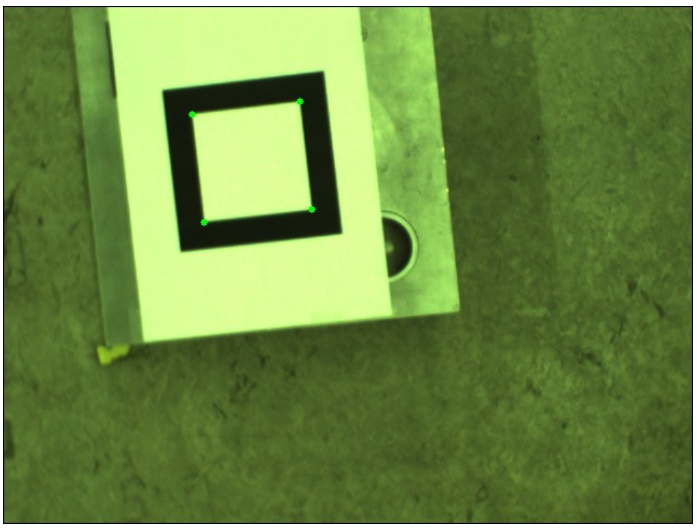
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.8$	$t_{y\,f} = -0.3$	$t_{z\,f} = 1.2$	$[m]$
$\phi_f = -91.7$	$\theta_f = 22.9$	$\psi_f = 83.1$	$[deg]$

Figure (6.50) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.50: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.51) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); the figure shows also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.52) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

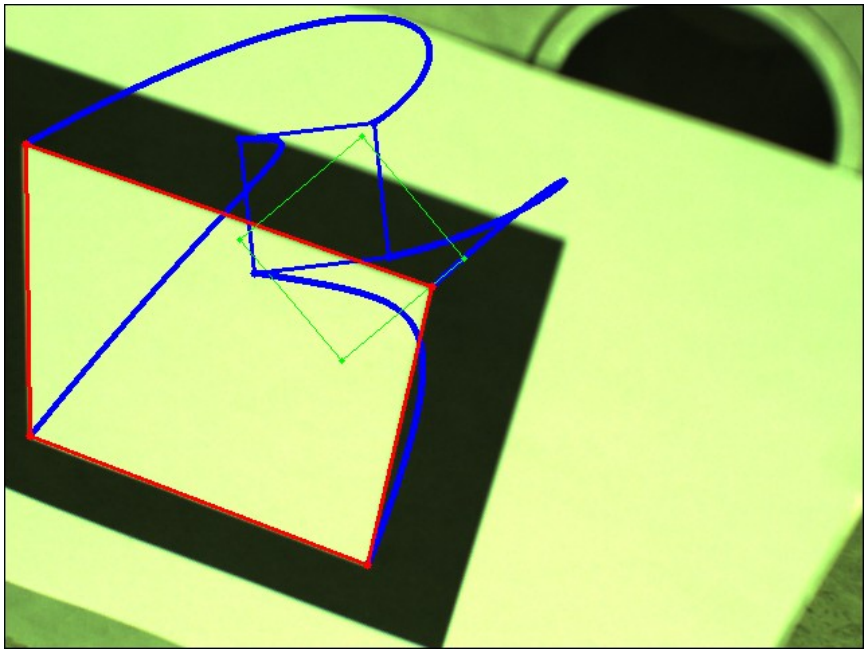


Figure 6.51: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

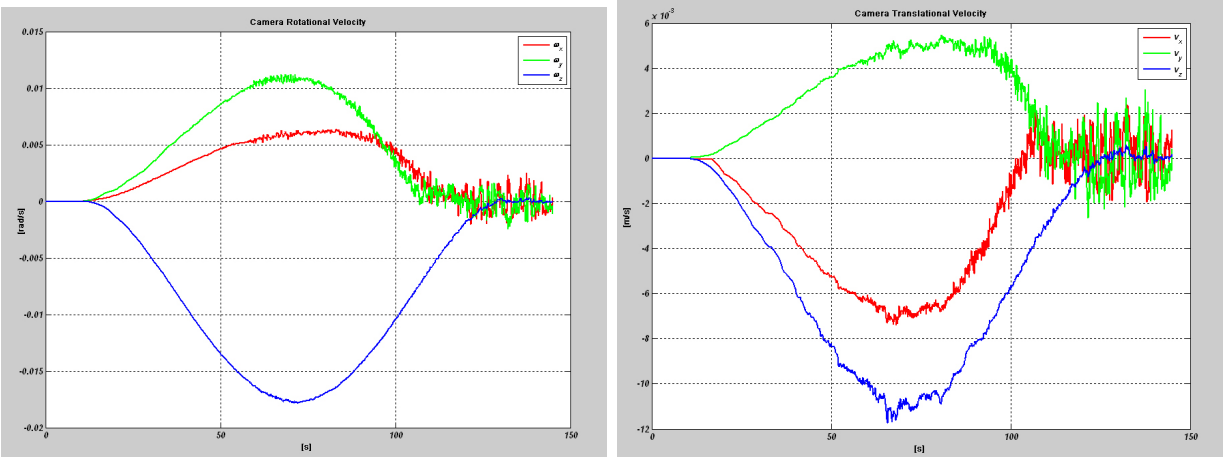


Figure 6.52: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.53): the camera is driven on a helicoidal trajectory induced from the image path planning $\boldsymbol{x}_d$ finally reaching the goal pose. In Figure (6.54) a) and

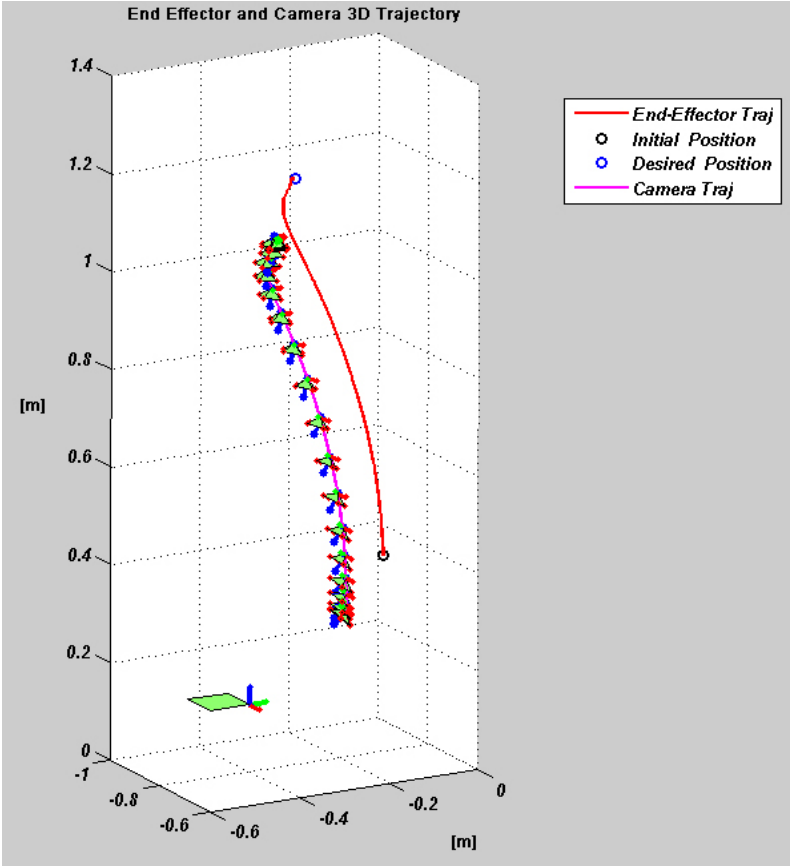
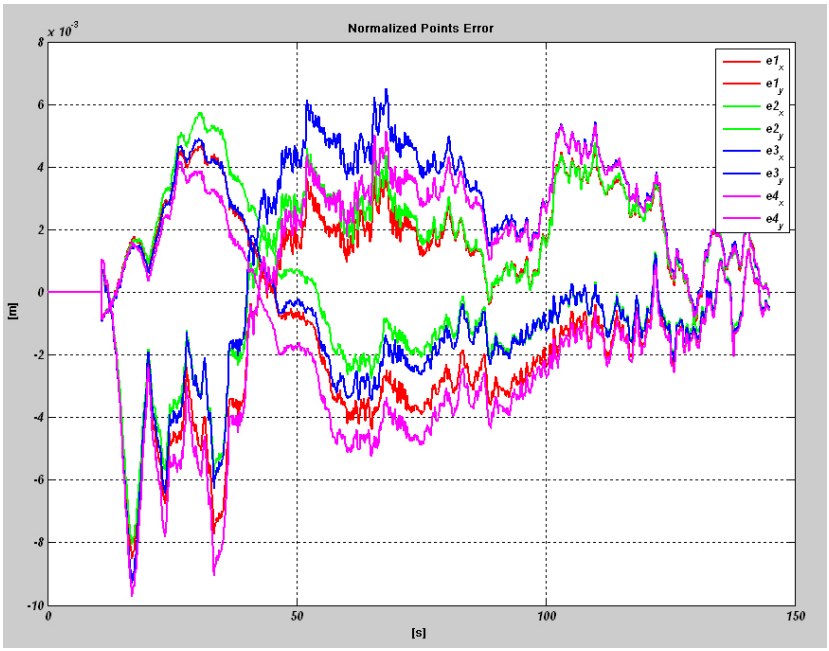


Figure 6.53: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

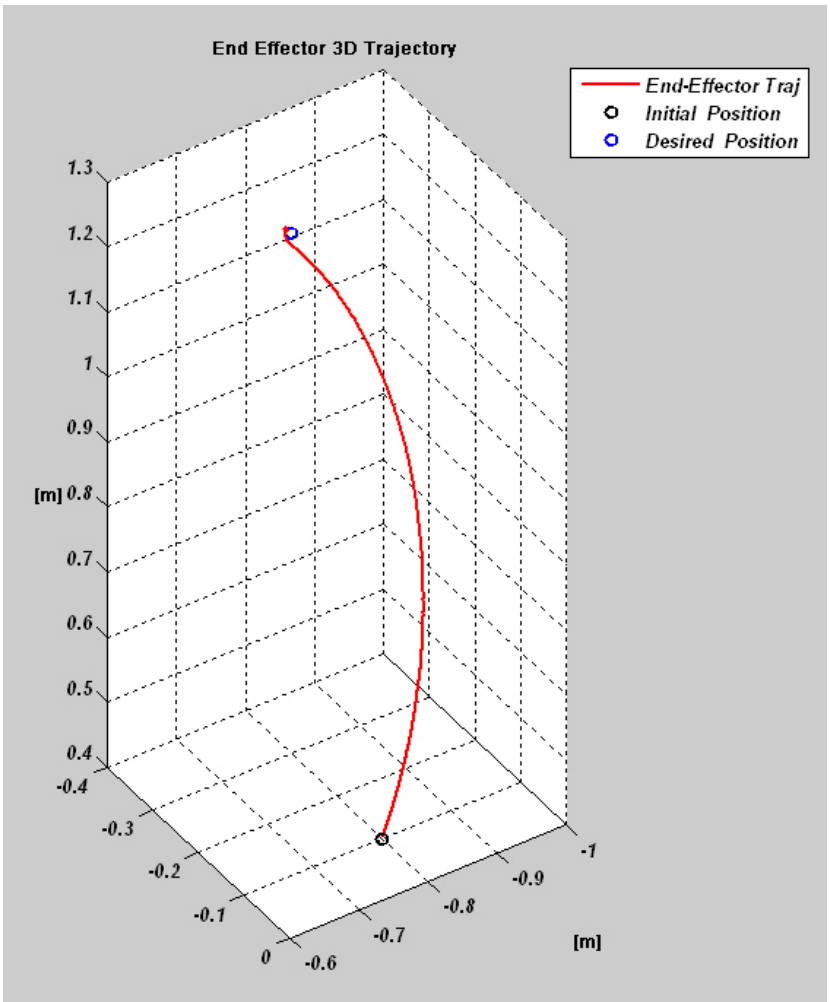
b) are respectively reported image error $\boldsymbol{e}$ components and the $3D$ end effector measured trajectory As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The end effector trajectory well follow an helical shape. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 0.9$	$e_{py} = 4.5$	$e_{pz} = 1.1$	$[mm]$
$e_{a\phi} = -0.013$	$e_{a\theta} = 0.248$	$e_{a\psi} = -0.128$	$[deg]$

Notice the fulfillment of the positioning task also in this case



a)



b)

Figure 6.54: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.52). b) Measured $3D$ end effector trajectory.

6.2.2.7 Experiment VII

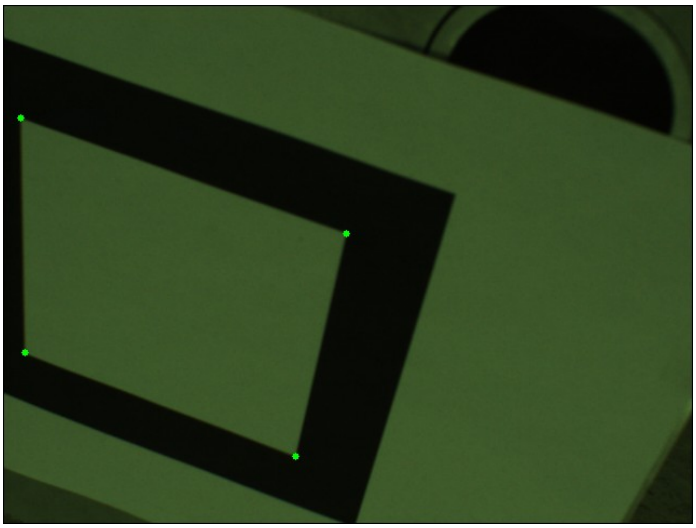
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.1$	$t_{z\,i} = 0.4$	$[m]$
$\phi_i = -137.5$	$\theta_i = 33.2$	$\psi_i = -5.7$	$[deg]$

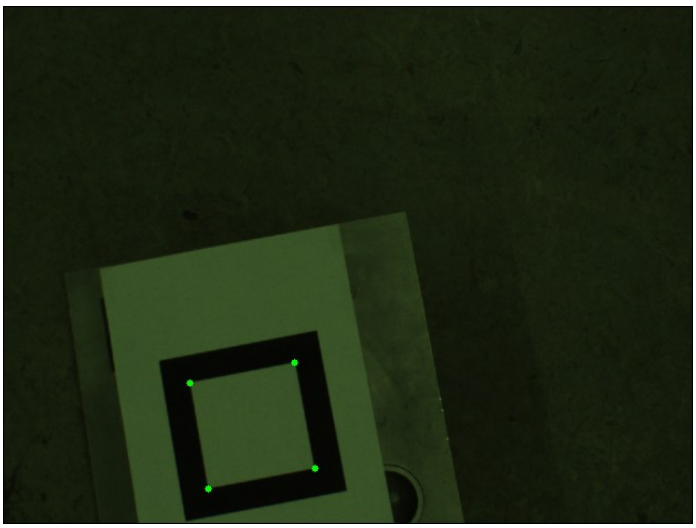
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.8$	$t_{y\,f} = 0.3$	$t_{zf} = 1.2$	$[m]$
$\phi_f = -80.2$	$\theta_f = 29.9$	$\psi_f = 68.8$	$[deg]$

Figure (6.55) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.55: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.56) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.57) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

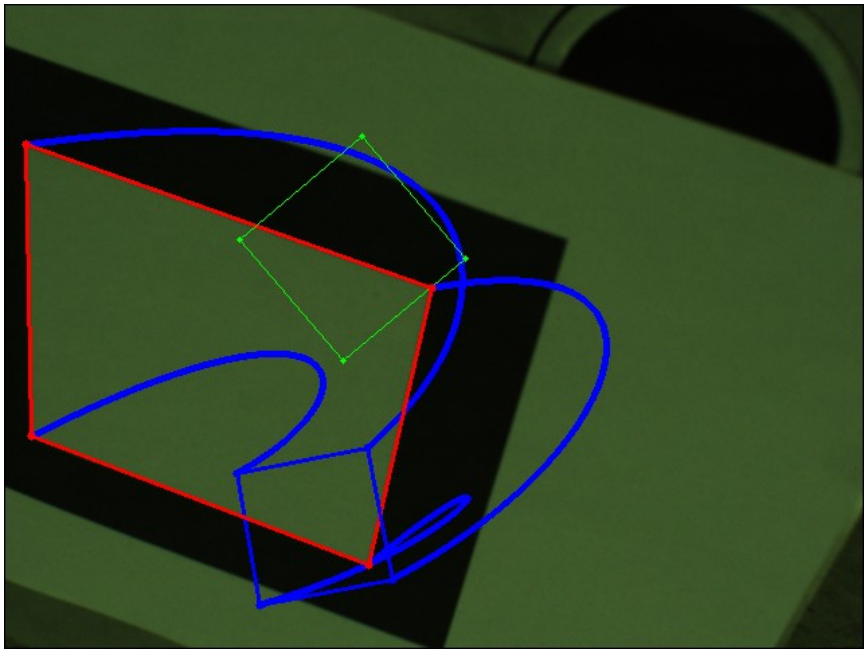


Figure 6.56: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

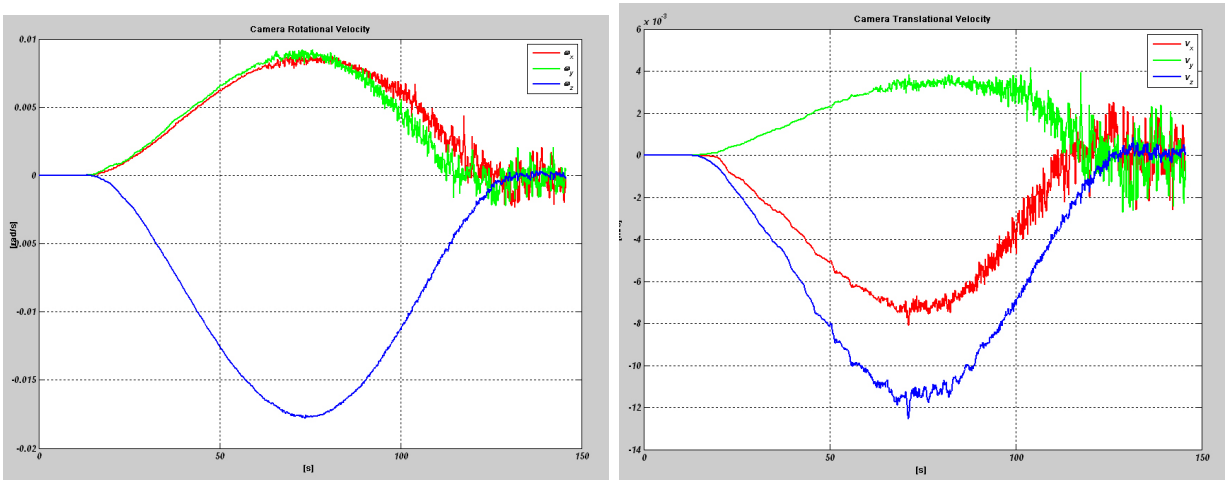


Figure 6.57: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.28): as clearly visible the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement. In Figure (6.59) a) and b) are respectively reported image error $\boldsymbol{e}$ components

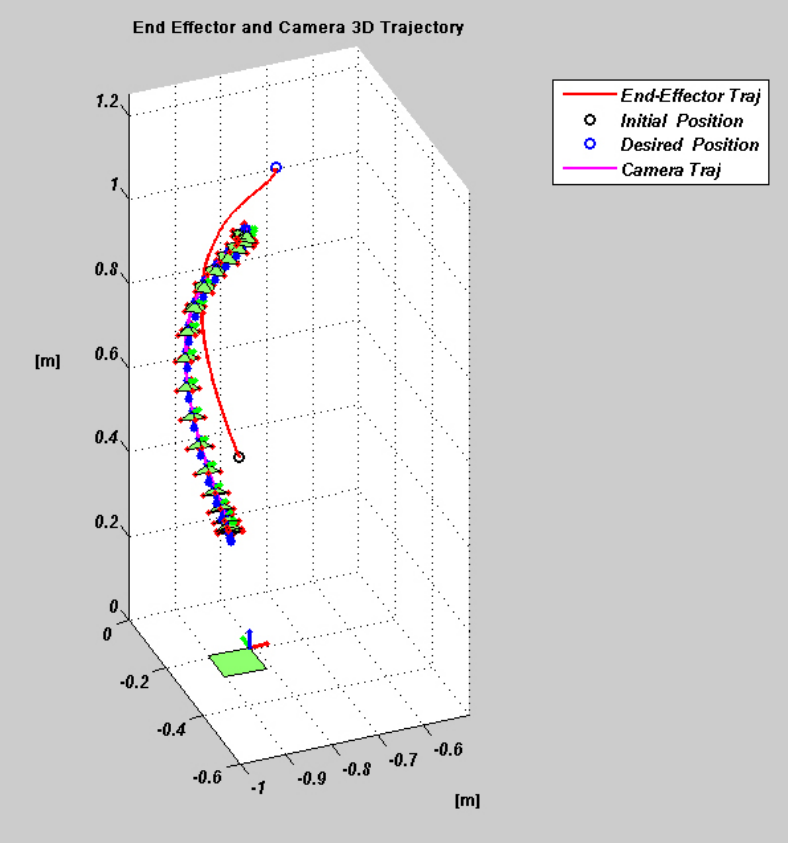
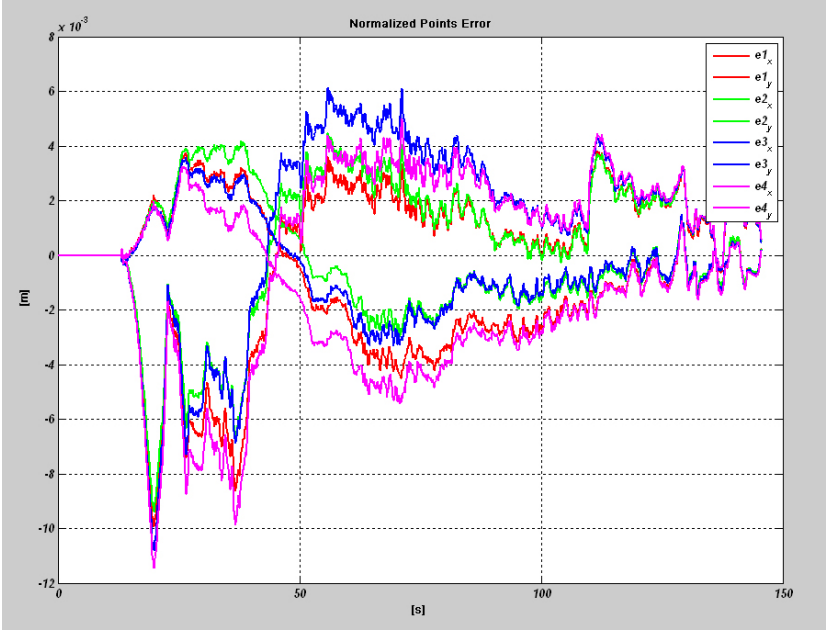


Figure 6.58: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

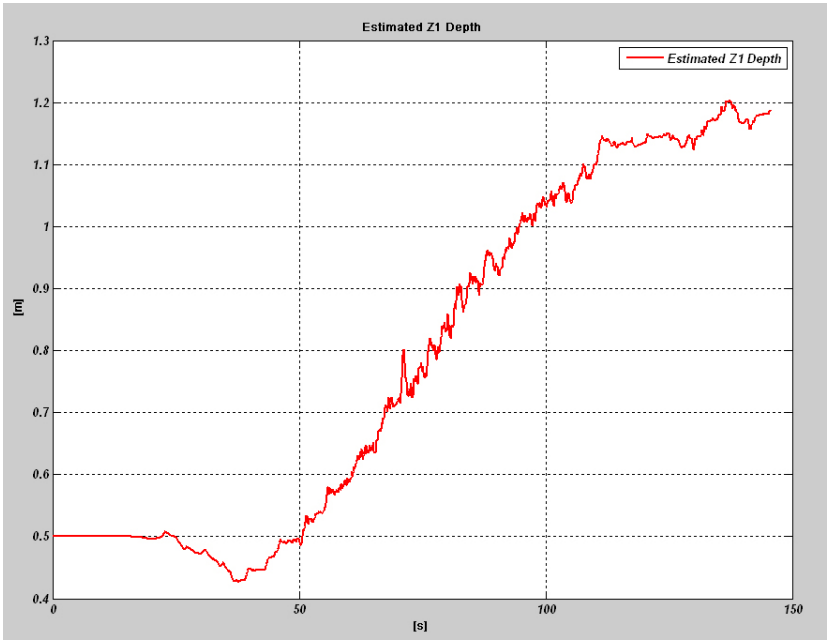
and the $3D$ depth estimate for the first tracked feature-point. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The adaptive estimation law, as clear from figure (6.59)-b, gives noisy depth estimates but the overall behavior follows the increasing camera distance from the target point Z_1. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 7.1$	$e_{py} = 9.3$	$e_{pz} = 2.5$	$[mm]$
$e_{a\phi} = 0.253$	$e_{a\theta} = 0.531$	$e_{a\psi} = 0.540$	$[deg]$

The IBVS system shows also in this case satisfactory perfomances.



a)



b)

Figure 6.59: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.57). b) Depth estimate $[m]$ on the first target point.

6.2.2.8 Experiment VIII

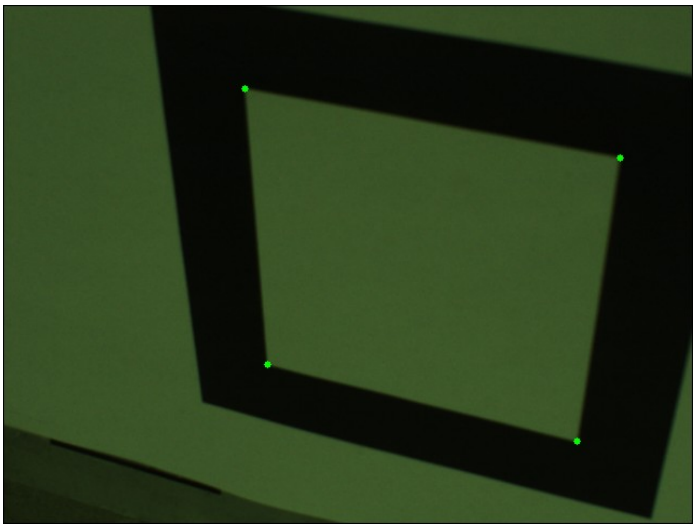
In this case the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.4$	$[m]$
$\phi_i = -137.5$	$\theta_i = 35.5$	$\psi_i = -17.2$	$[deg]$

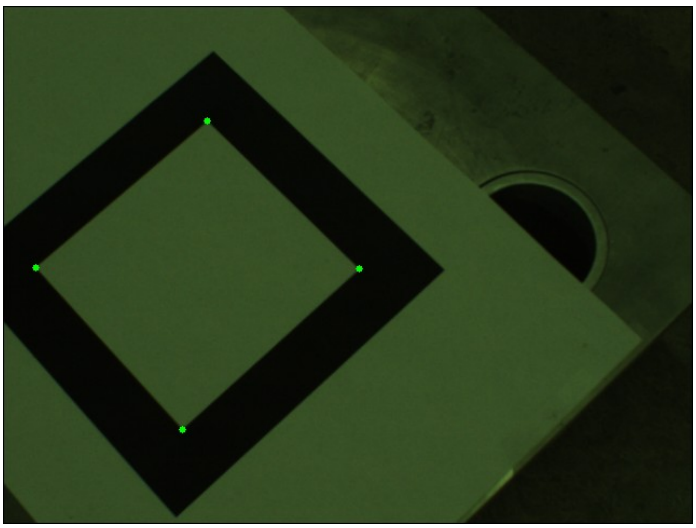
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.67$	$t_{y\,f} = -0.5$	$t_{z\,f} = 0.6$	$[m]$
$\phi_f = -68.75$	$\theta_f = 22.9$	$\psi_f = 54.4$	$[deg]$

Figure (6.60) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.60: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.61) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.62) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

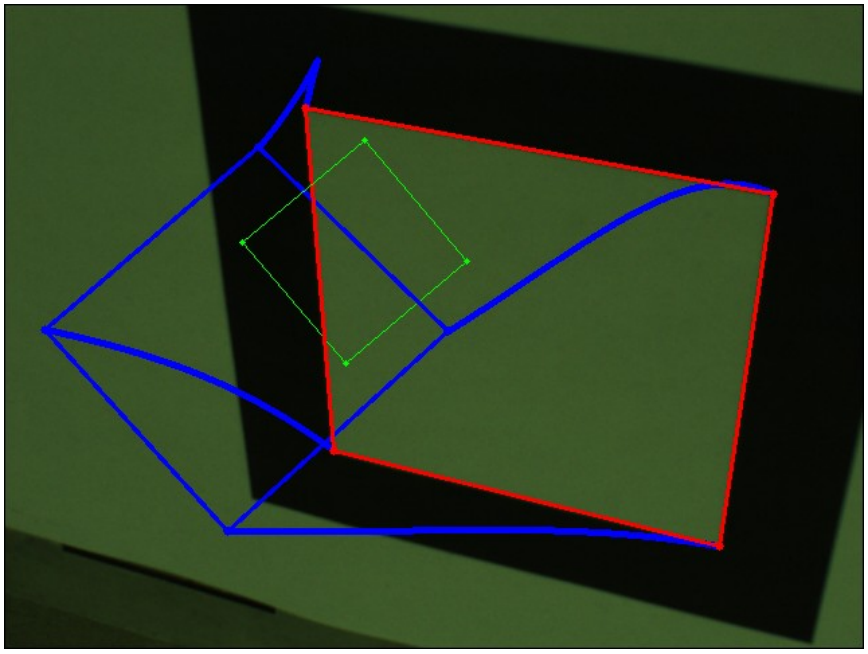


Figure 6.61: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

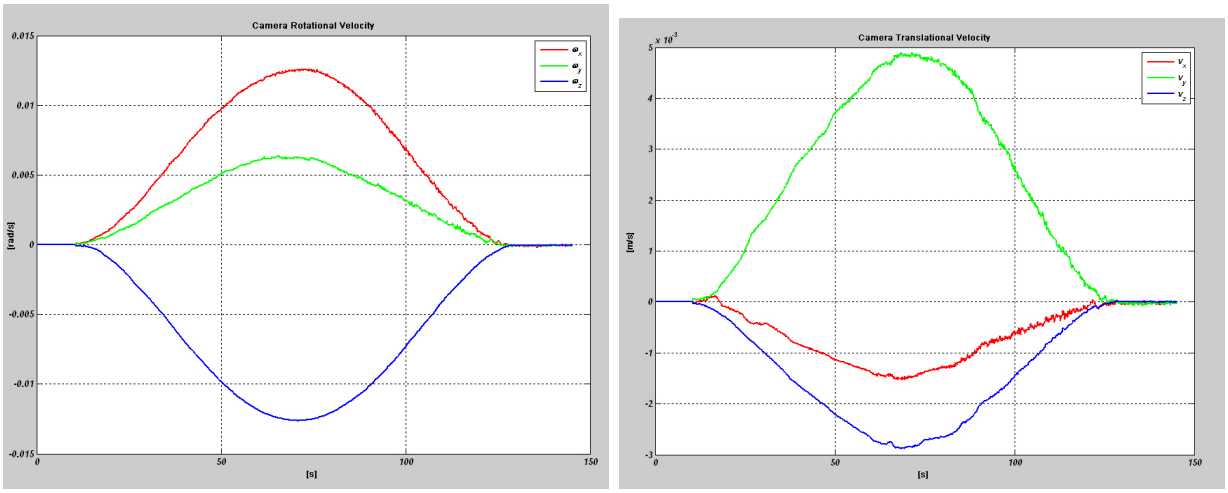


Figure 6.62: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.28): as clearly visible the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement. In Figure (6.64) a) and b) are respectively reported image error $\boldsymbol{e}$ components

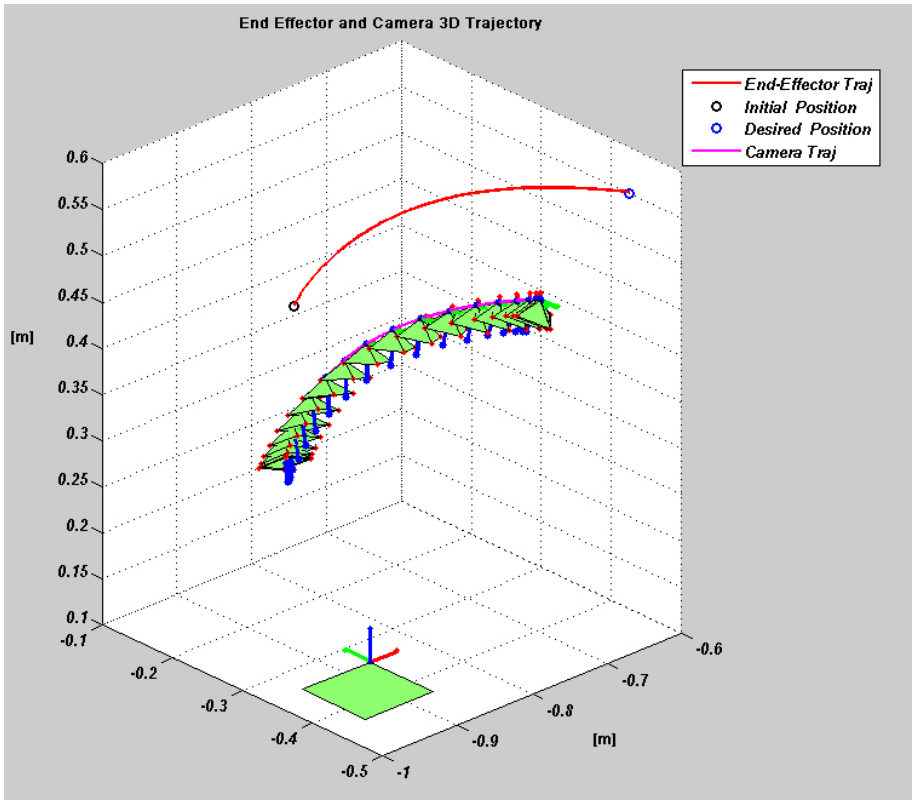
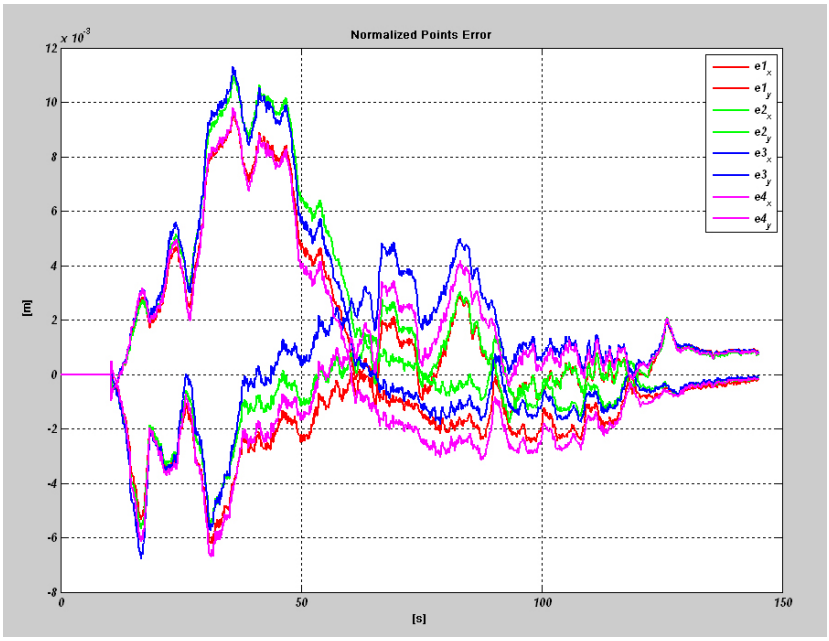


Figure 6.63: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

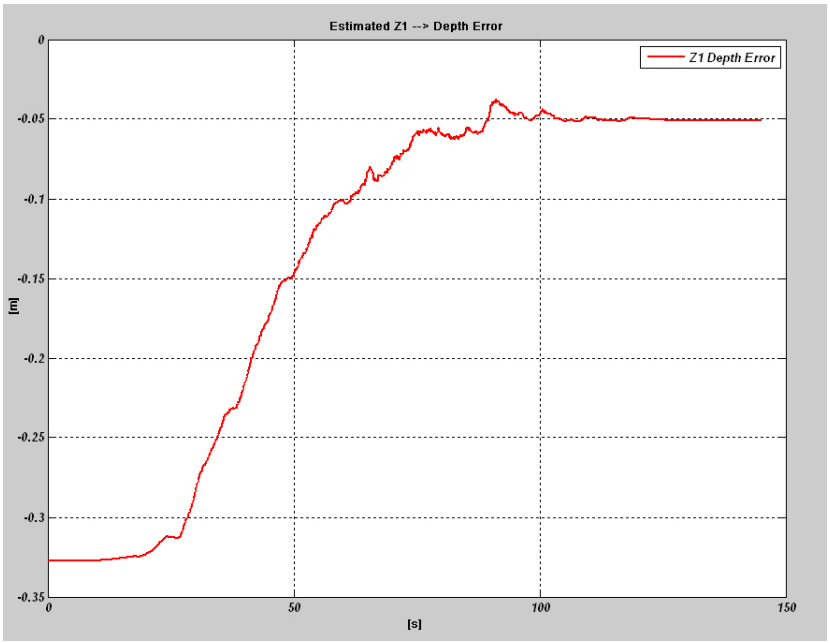
and the $3D$ depth estimation error for the first tracked feature-point. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The adaptive estimation law, as clear from figure (6.64)-b, gives converges to a good estimation of the target point unknown depth Z_1. In the following table is reported the positioning task error as defined in section (6.2).

$e_{p\,x} = 0.54$	$e_{p\,y} = -0.22$	$e_{p\,z} = -0.98$	$[mm]$
$e_{a\,\phi} = -0.066$	$e_{a\,\theta} = -0.023$	$e_{a\,\psi} = 0.011$	$[deg]$

Notice the very good performances of the IBVS system for this positioning task.



a)



b)

Figure 6.64: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.62). b) Depth estimation error $[m]$ on the first target point.

6.2.2.9 Experiment IX

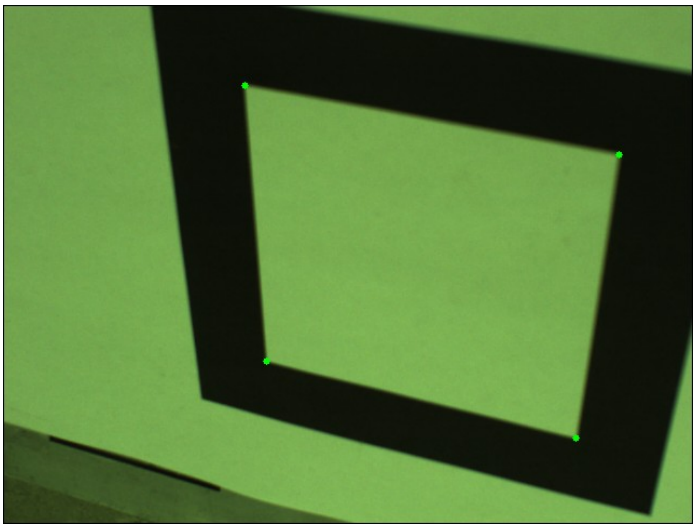
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.4$	$[m]$
$\phi_i = -137.5$	$\theta_i = 35.5$	$\psi_i = -17.2$	$[deg]$

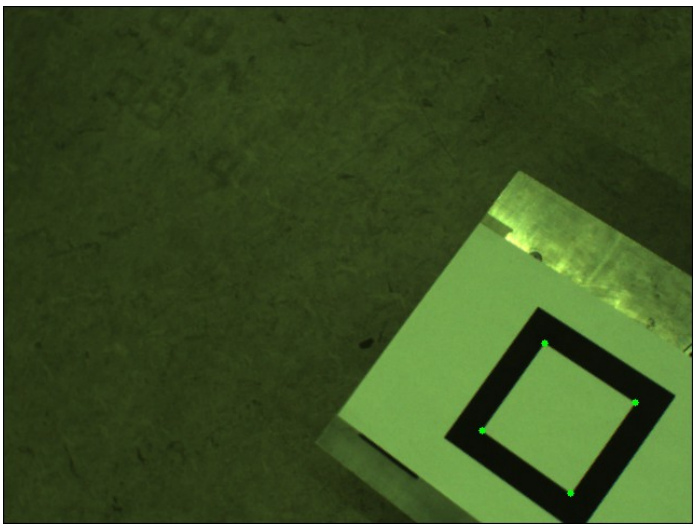
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.65$	$t_{y\,f} = -0.55$	$t_{z\,f} = 1.15$	$[m]$
$\phi_f = -68.75$	$\theta_f = 34.4$	$\psi_f = 45.8$	$[deg]$

Figure (6.65) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.65: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.66) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.67) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

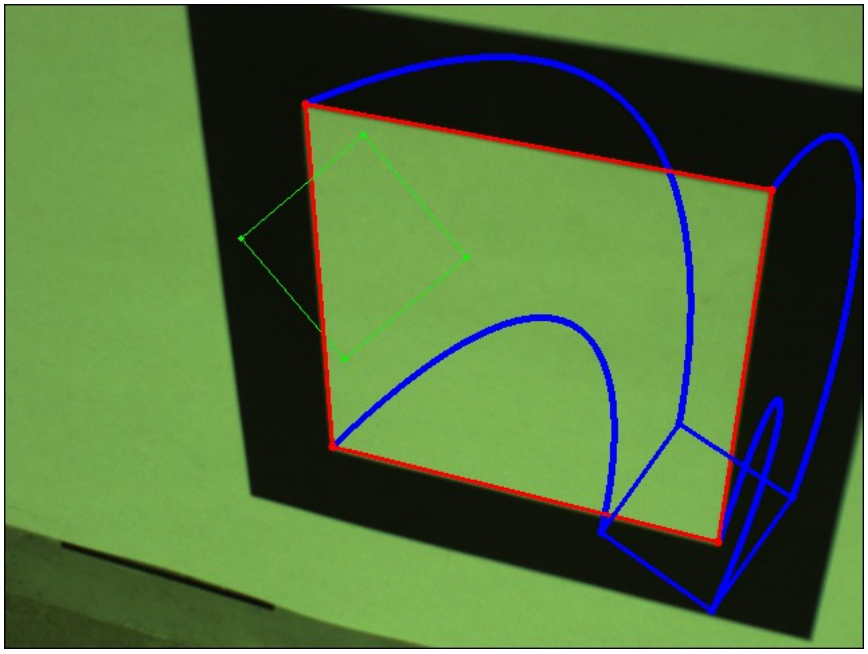


Figure 6.66: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

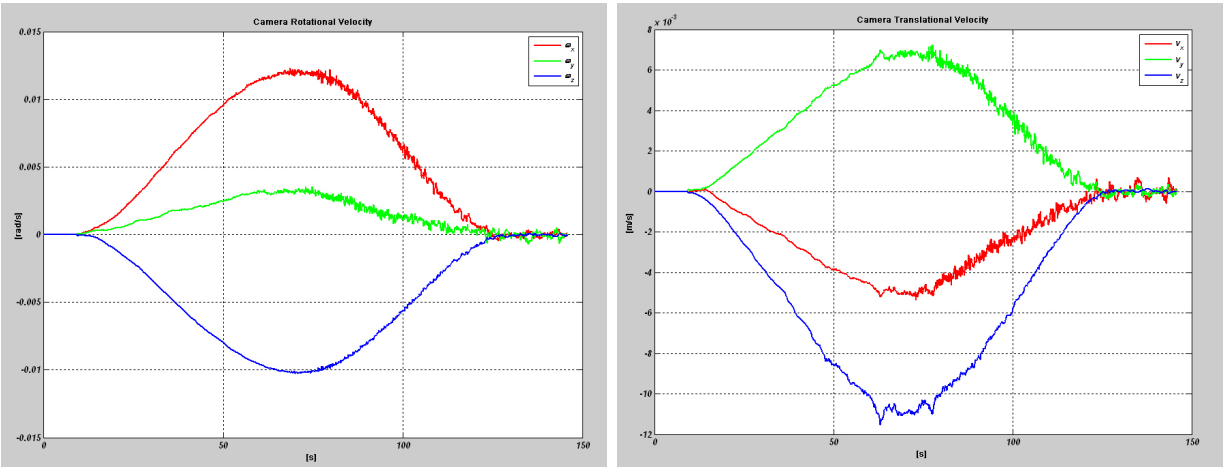


Figure 6.67: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.28): as clearly visible the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement. In Figure (6.69) a) and b) are respectively reported image error $\boldsymbol{e}$ components

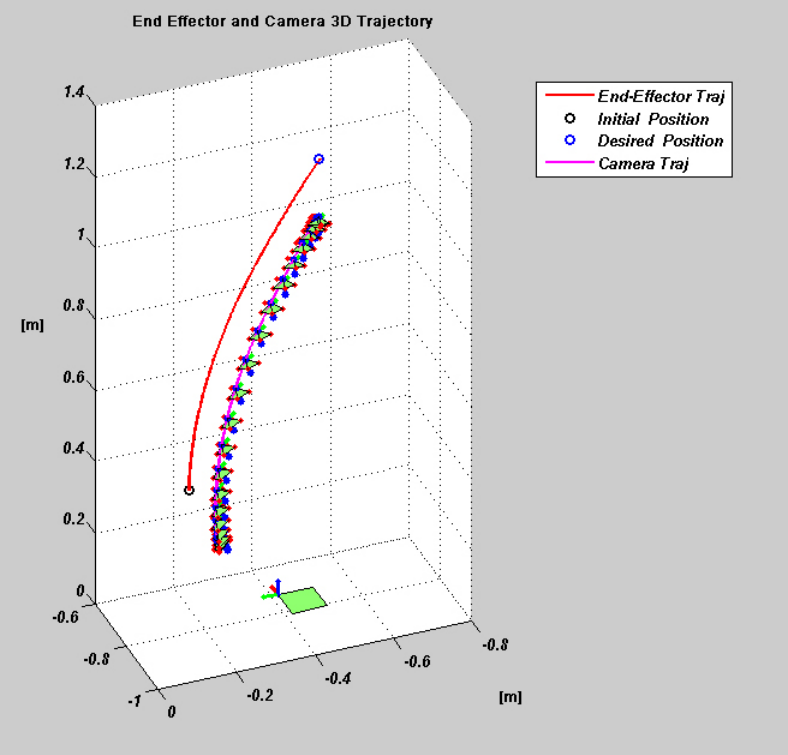
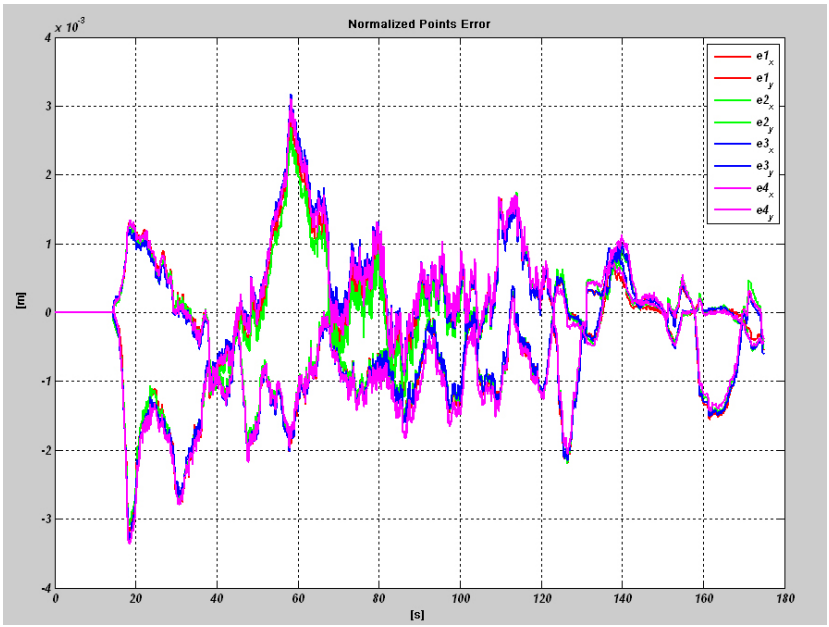


Figure 6.68: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

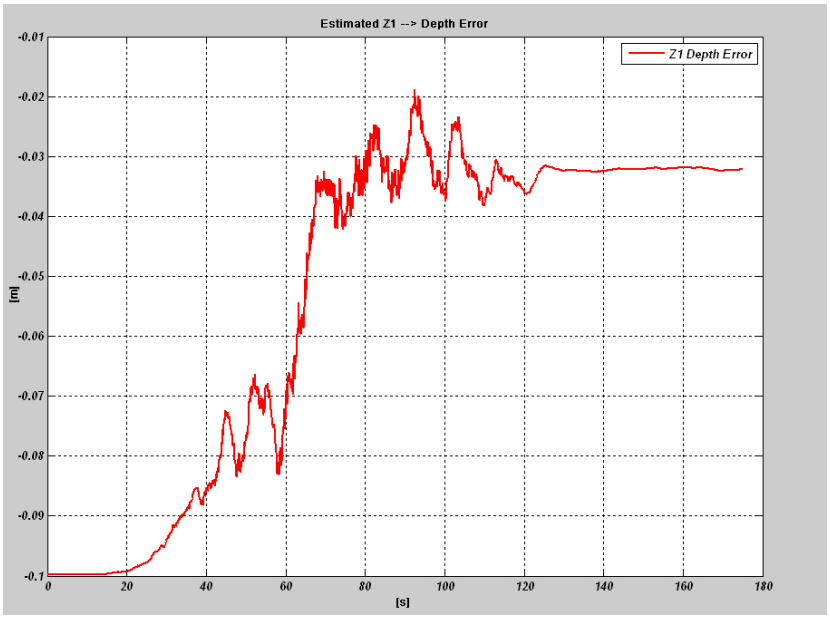
and the $3D$ depth estimation error for the first tracked feature-point. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The adaptive estimation law, shown in figure (6.69)-b, gives in this case a satisfactory estimate of the point depth Z_1. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 1.6$	$e_{py} = 3.8$	$e_{pz} = 0.2$	$[mm]$
$e_{a\phi} = -0.164$	$e_{a\theta} = 0.160$	$e_{a\psi} = -0.229$	$[deg]$

Also in this experiment the IBVS system demonstrates the good performances.



a)



b)

Figure 6.69: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.27). b) Depth estimation error $[m]$ on the first target point.

6.2.2.10 Experiment X

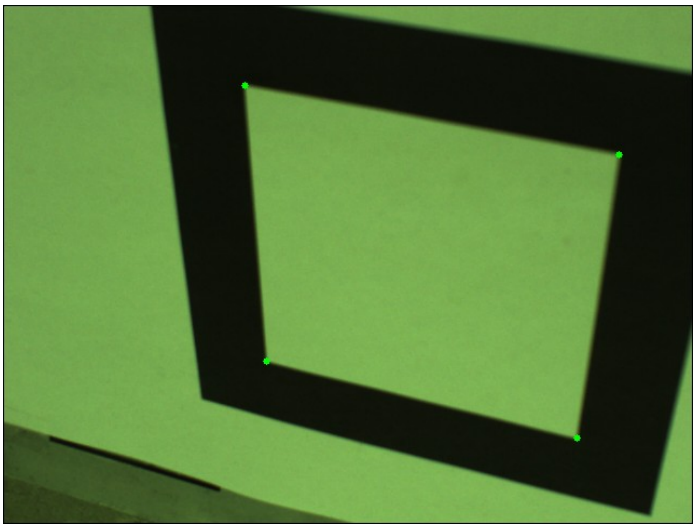
In this experiment the initial end effector camera frame E_i is defined w.r.t. W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.4$	$[m]$
$\phi_i = -137.5$	$\theta_i = 35.5$	$\psi_i = -17.2$	$[deg]$

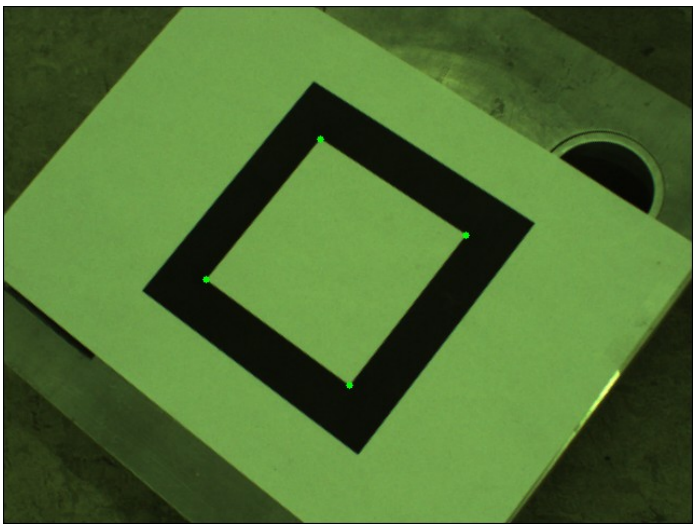
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.65$	$t_{y\,f} = -0.55$	$t_{z\,f} = 0.7$	$[m]$
$\phi_f = -68.7$	$\theta_f = 28.6$	$\psi_f = 45.8$	$[deg]$

Figure (6.70) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.70: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.71) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.72) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

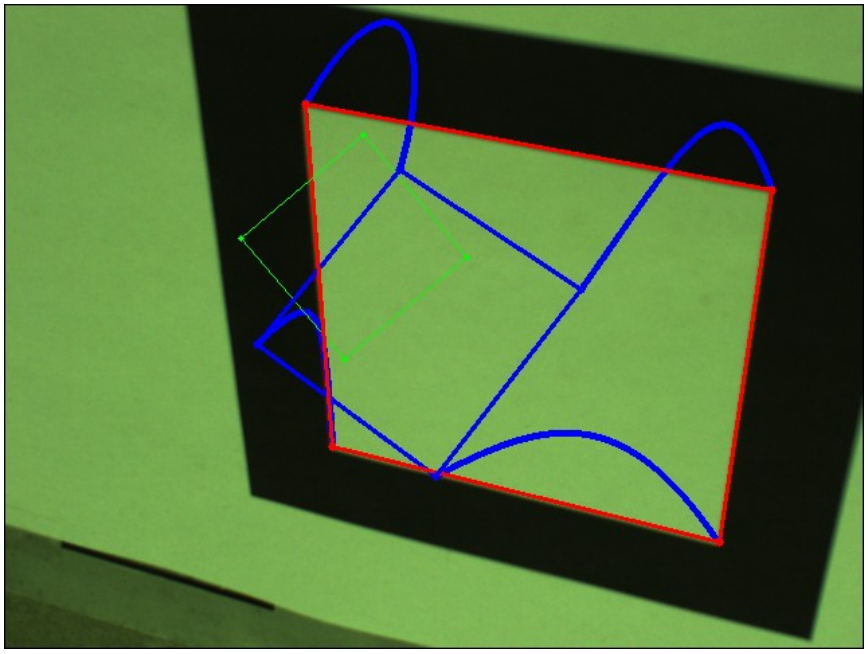


Figure 6.71: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

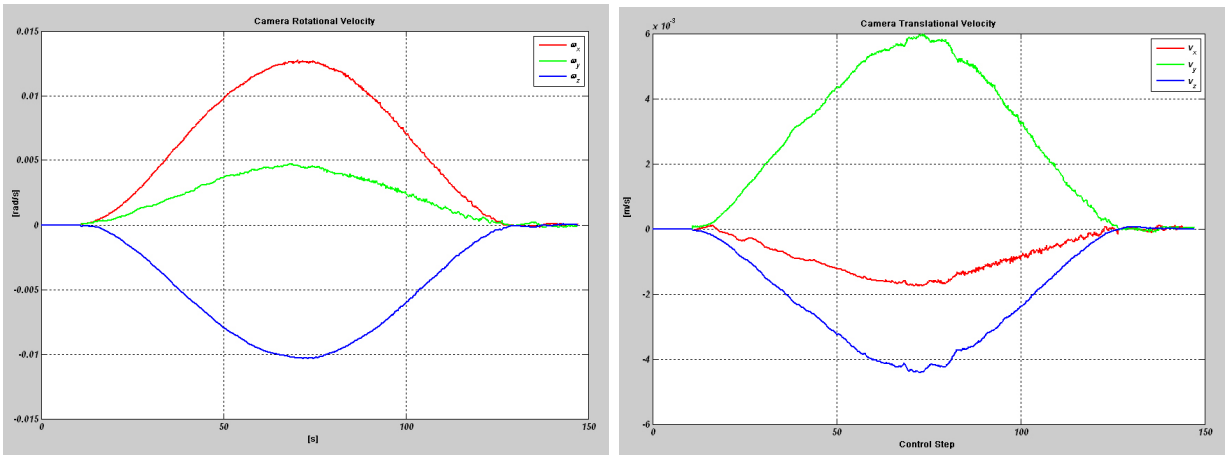


Figure 6.72: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.28): as clearly visible the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement. In Figure (6.74) a) and b) are respectively reported image error $\boldsymbol{e}$ components

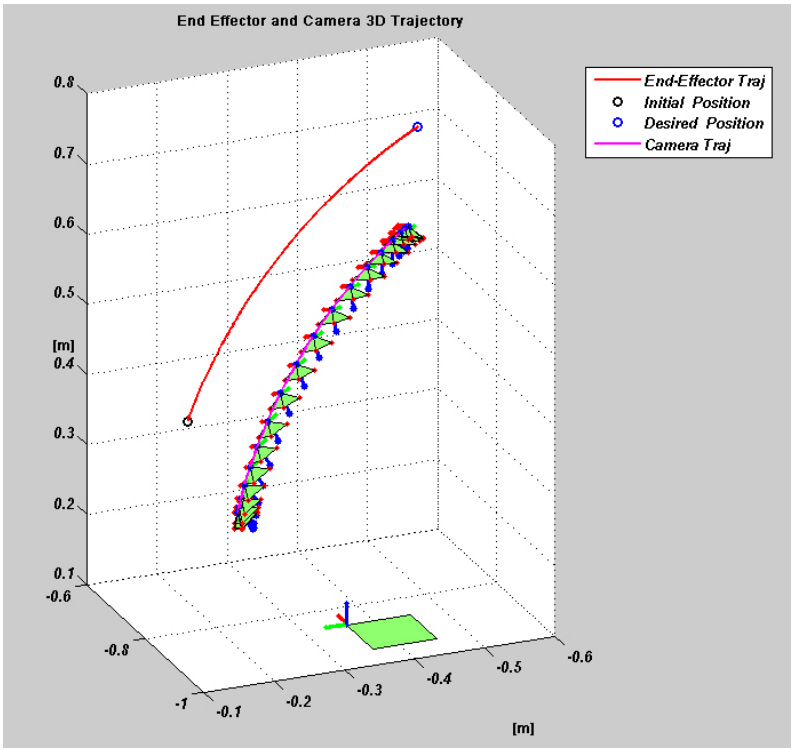
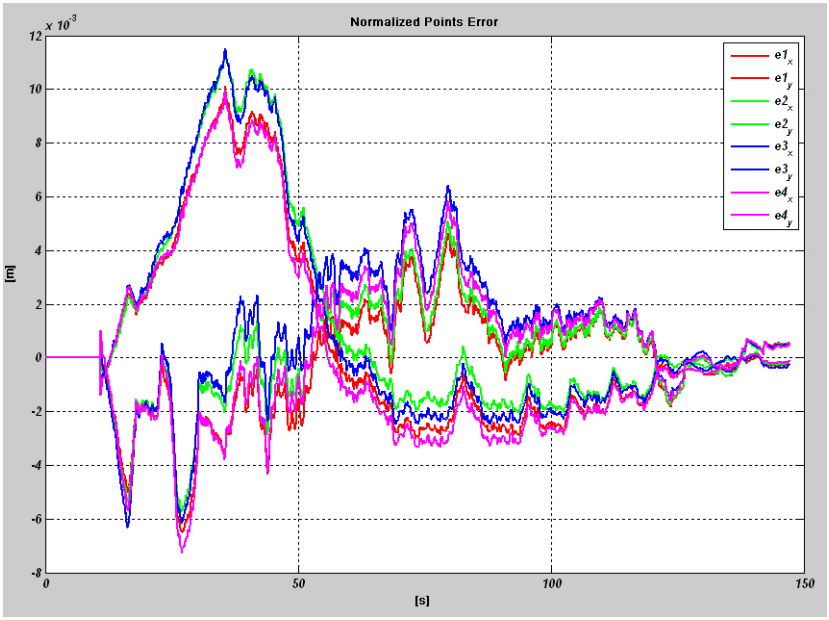


Figure 6.73: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

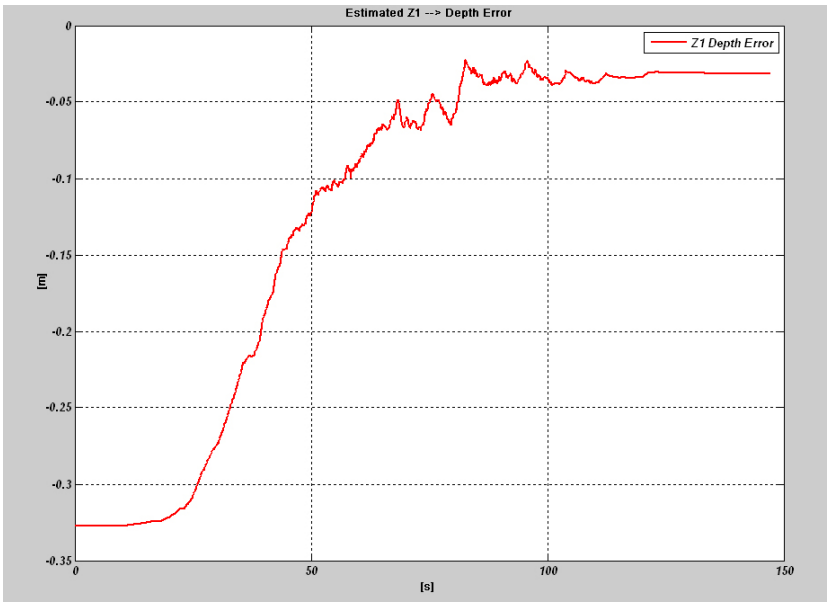
and the $3D$ depth estimation error for the first tracked feature-point. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The adaptive estimation law, as clear from figure (6.74)-b, minimize the depth error converging to a good estimate of Z_1. In the following table is reported the positioning task error as defined in section (6.2).

$e_{px} = 0.70$	$e_{py} = 0.84$	$e_{pz} = -0.38$	$[mm]$
$e_{a\phi} = -0.084$	$e_{a\theta} = 0.075$	$e_{a\psi} = -0.097$	$[deg]$

The IBVS system results very accurate also for this positioning task.



a)



b)

Figure 6.74: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.72). b) Depth estimation error $[m]$ on the first target point.

6.2.3 Experiments with " natural " Target

6.2.3.1 Experiment I

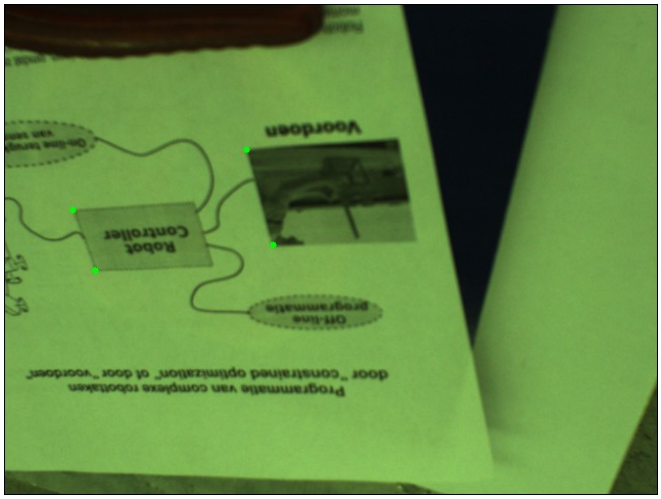
The initial end effector camera frame E_i is defined with respect to W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.45$	$[m]$
$\phi_i = -130$	$\theta_i = 31.5$	$\psi_i = -5.7$	$[deg]$

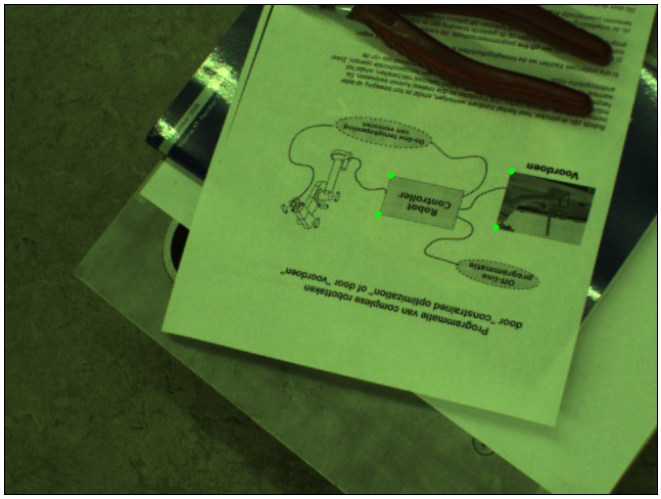
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.65$	$t_{y\,f} = -0.55$	$t_{z\,f} = 0.7$	$[m]$
$\phi_f = -68.7$	$\theta_f = 34.3$	$\psi_f = 45.8$	$[deg]$

Figure (6.75) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.75: a) Initial target view I_i. b) Goal target view I_f.

Figure (6.76) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.77) shows the components of the camera twist screw $\boldsymbol{w}$ induced from the control

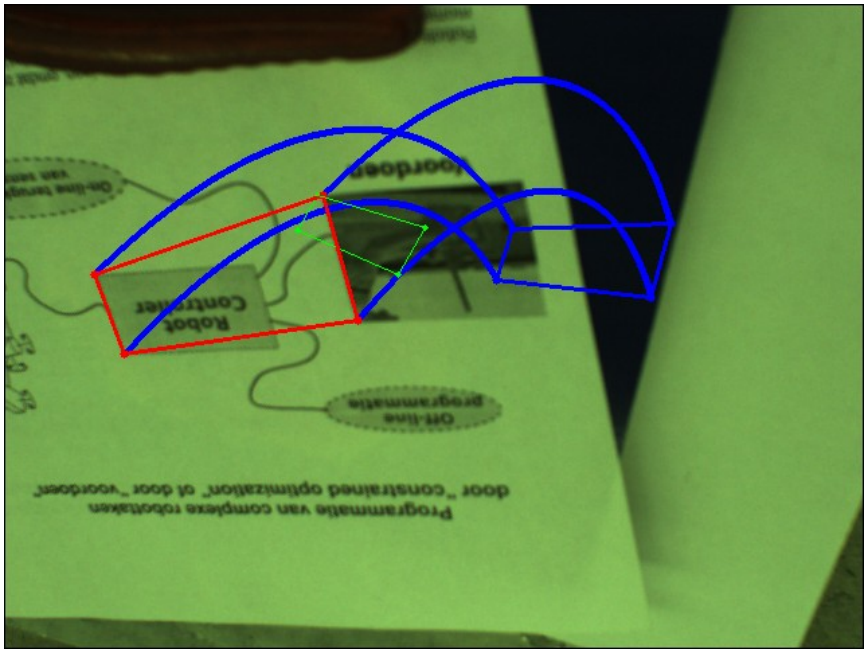


Figure 6.76: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

law: they result smooth and feasible as set by the quintic polynomial time law $s(t)$ in section (4.3.1).

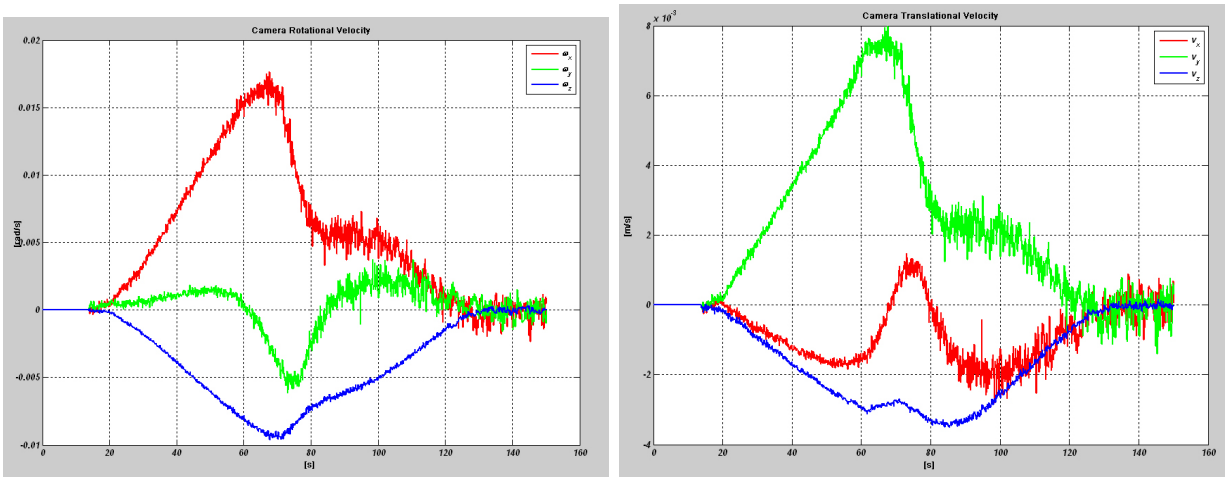


Figure 6.77: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.78): as clearly visible the camera moves on a helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$ finally resulting in a harmonious movement. In Figure (6.79) a) and b) are respectively reported image error $\boldsymbol{e}$ components

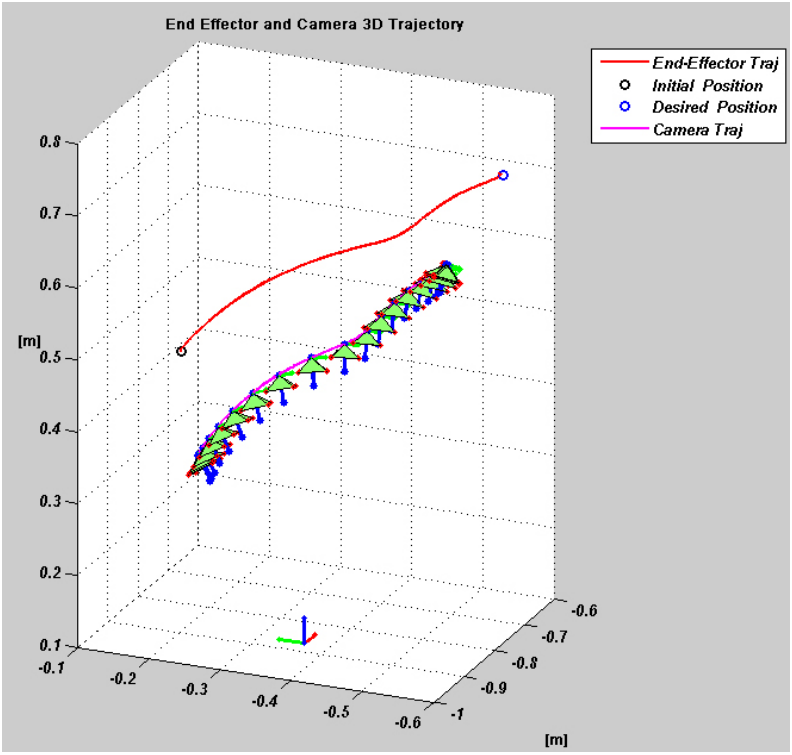
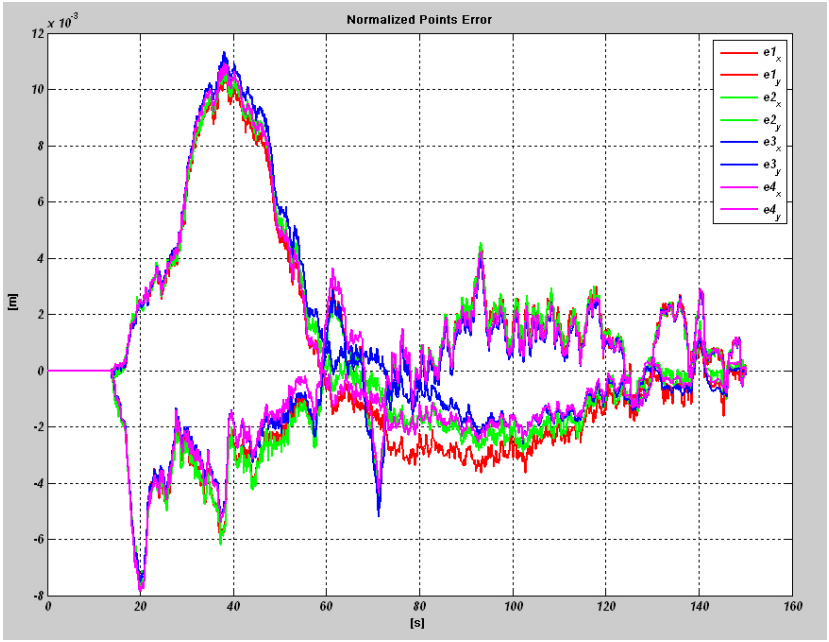


Figure 6.78: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector reach the goal position (blue circle).

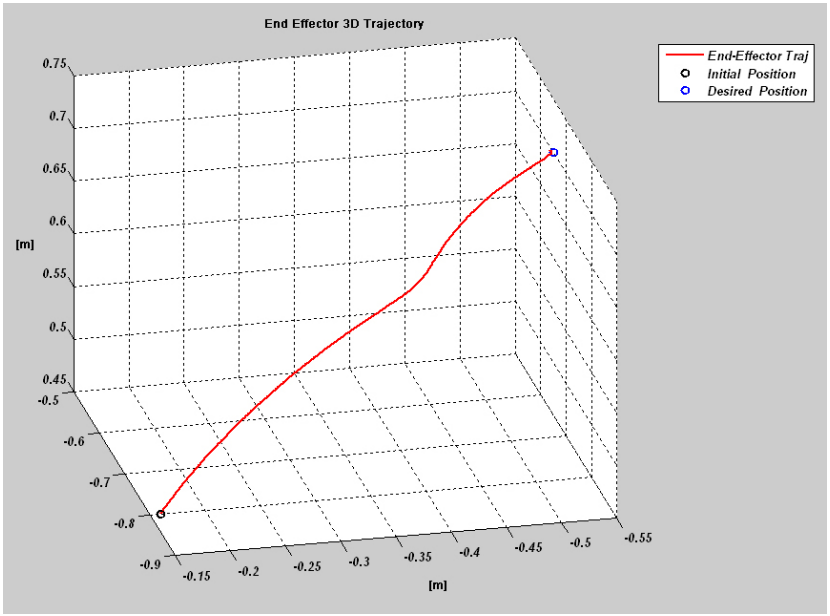
and the $3D$ end-effector trajectory. As clearly visible $\boldsymbol{e}$ components remain very small during task execution conveniently exploiting the local stability properties of the control law. The camera $3D$ trajectory in (6.79)-b,follows an helical shape. In the following table is reported the positioning task error as defined in section (6.2).

$e_{p\,x} = 5.8$	$e_{p\,y} = 4.2$	$e_{p\,z} = 4$	$[mm]$
$e_{a\,\phi} = 0.309$	$e_{a\,\theta} = 0.085$	$e_{a\,\psi} = 0.687$	$[deg]$

Notice the very good performances of the IBVS system also with this more "natural" target.



a)



b)

Figure 6.79: a) Normalized image error $\boldsymbol{e}$ components: although very noisy the error components remain very small during the whole task thus giving rise to the smooth twist screw components profiles of Figure (6.27). b) $3D$ end effector trajectory $[m]$ registered during task execution.

6.2.3.2 Experiment II

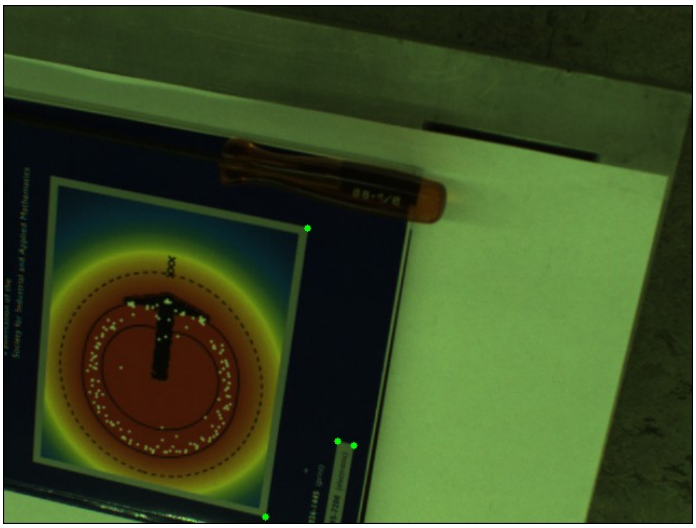
The initial end effector camera frame E_i is defined with respect to W by:

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.62$	$[m]$
$\phi_i = -112.9$	$\theta_i = 29.79$	$\psi_i = 0$	$[deg]$

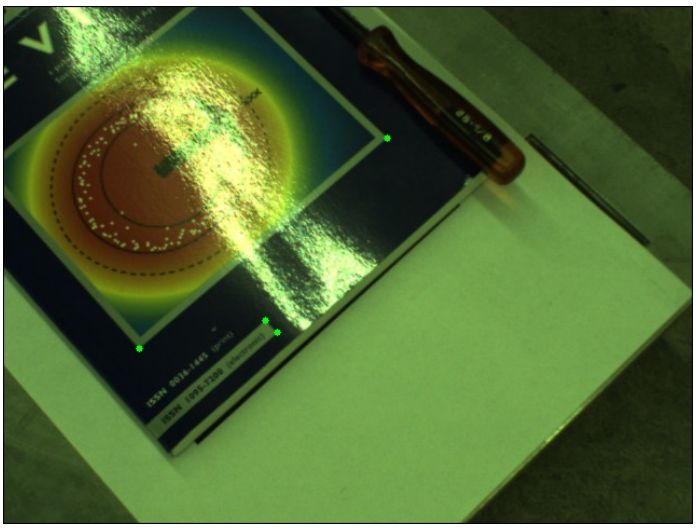
The final end effector pose E_f is instead defined with respect to W by:

$t_{x\,f} = -0.67$	$t_{y\,f} = -0.5$	$t_{z\,f} = 0.6$	$[m]$
$\phi_f = -68.7$	$\theta_f = 22.9$	$\psi_f = 54.4$	$[deg]$

Figure (6.75) shows the initial and final target views (I_i and I_f) grabbed from the camera;



a)



b)

Figure 6.80: a) Initial target view I_i. b) Goal target view I_f.

Notice in this case the particular target shape: the chosen 4 feature points point form a non convex quadrangle. Figure (6.81) reports the reference image planned trajectory $\boldsymbol{x}_d$ reprojected in pixel (blue lines); this figure reports also in thin green the auxiliary view I_a, used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3). Figure (6.82) shows the components of the camera twist screw $\boldsymbol{w}$ induced

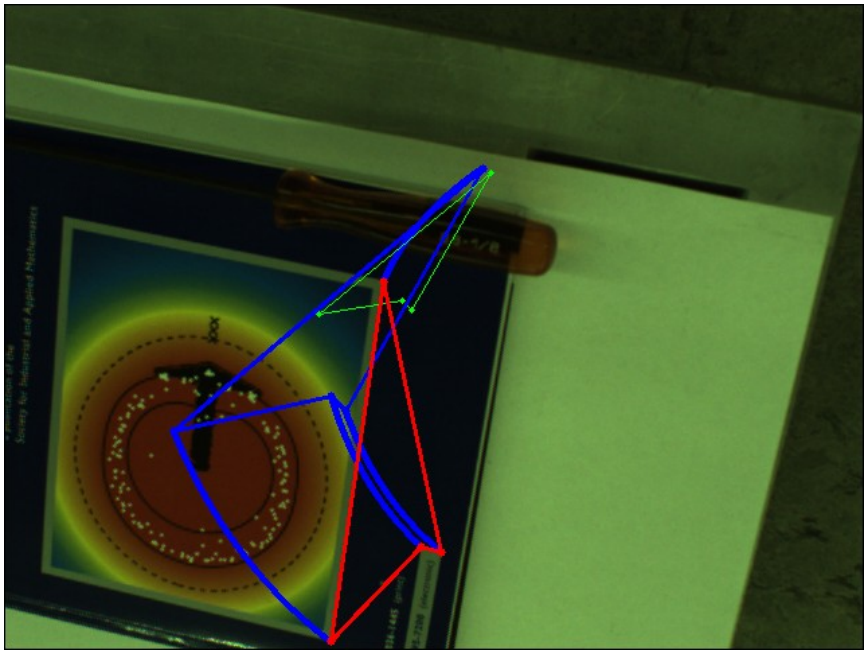


Figure 6.81: Pixel planned image trajectories $\boldsymbol{x}_d$ (blue lines) during the off-line phase.

from the control law: they have some spikes due to the bad light reflection properties of the target.

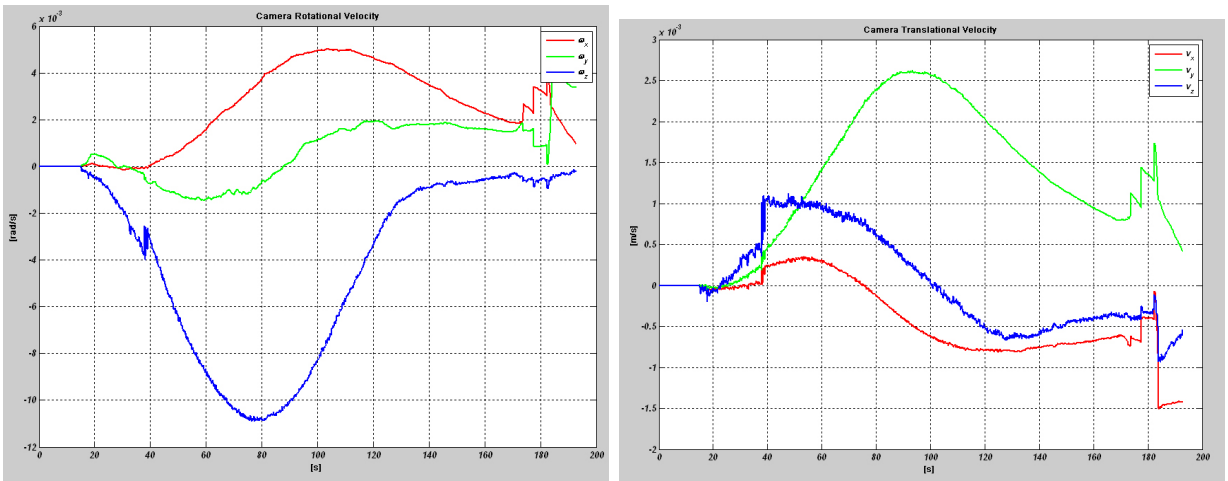


Figure 6.82: Angular camera velocity components (ω_x, ω_y, ω_z) in $[rad/s]$ -left side. Translational camera velocity components (v_x, v_y, v_z) in $[m/s]$-right side.

The $3D$ camera frame and end-effector origin trajectories measured during task execution are reported in Figure (6.83): despite the problems due to light reflection of the target the camera moves on the helical-shape trajectory induced from the image path planning $\boldsymbol{x}_d$. In Figure (6.84) a) and b) are respectively reported image error $\boldsymbol{e}$ components and

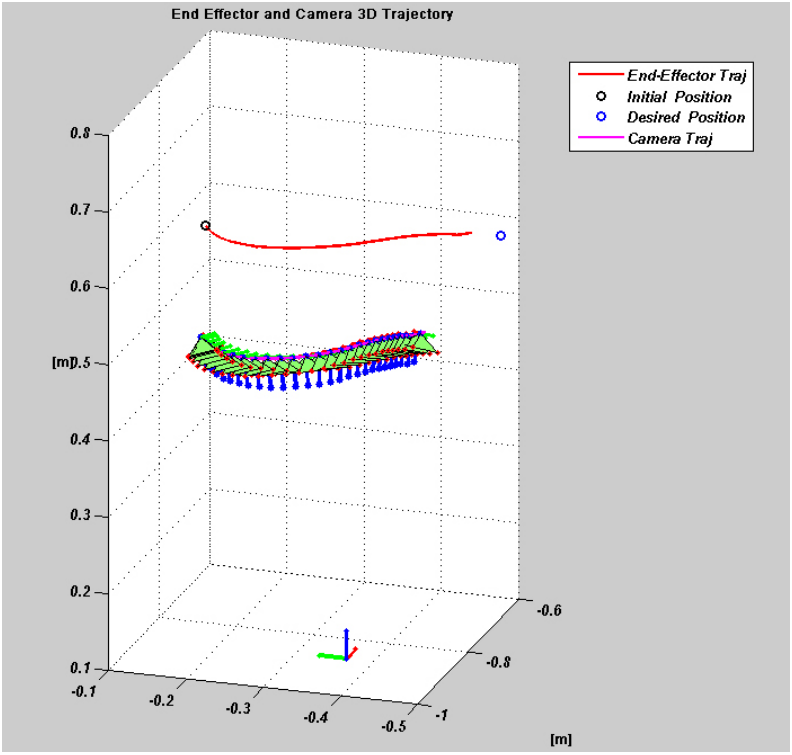
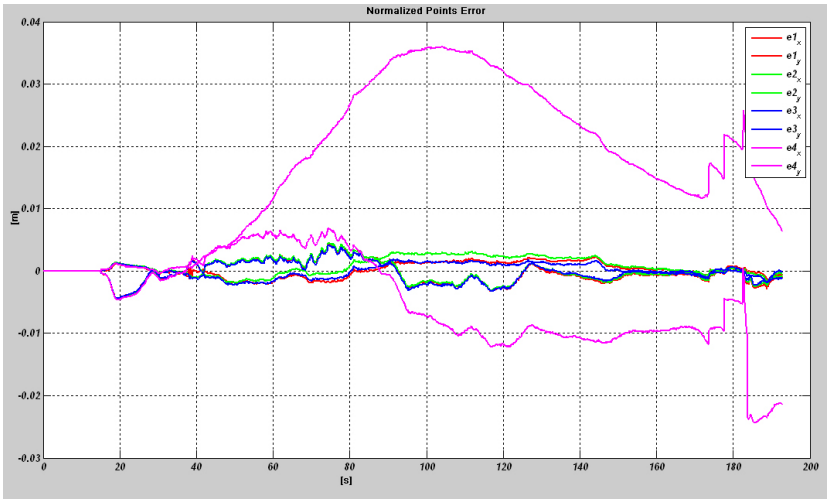


Figure 6.83: Envelope of the 3D camera poses and end-effector origins trajectories (red line) resulting from the experiment. Notice that the end effector in this case stops before the goal position (blue circle): this is due to feature disappearing consequent to light reflections problem .

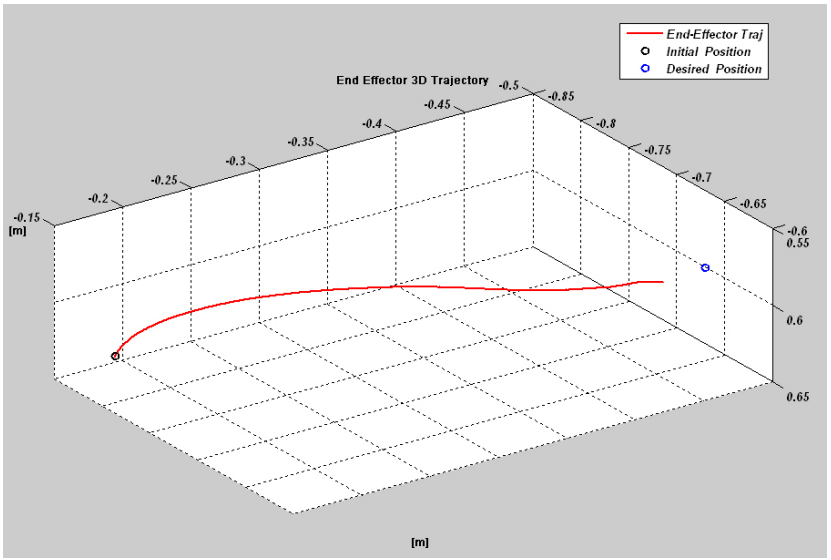
the $3D$ end-effector trajectory. $\boldsymbol{e}$ components follows the time law $s(t)$ induced behavior up to grow in consequence of light reflection phenomena. The end-effector trajectory in (6.84)-b, however follow the induced helical shape. In the following table is reported the positioning task error as defined in section (6.2).

$e_{p\,x} = 20$	$e_{p\,y} = 45$	$e_{p\,z} = 8.4$	$[mm]$
$e_{a\,\phi} = -1.25$	$e_{a\,\theta} = -4.67$	$e_{a\,\psi} = 0.56$	$[deg]$

Notice that for this positioning task the light reflection phenomena cause the partial fail of the task.



a)



b)

Figure 6.84: a) Normalized image error $\boldsymbol{e}$ components. b) $3D$ end effector trajectory $[m]$ registered during task execution.

6.2.4 Experiments with coarse Camera Calibration

In this section we report the experimental results obtained by using a coarse camera intrinsic $\boldsymbol{K}$ to investigate the system robustness with respect to camera calibration errors. Remember that the camera calibration matrix, according to the full perspective camera model of equation (2.26) with 0 skew factor, is defined as follow:

$$\boldsymbol{K} = \begin{bmatrix} \alpha_x & 0 & o_x \\ 0 & f\alpha_y & o_y \\ 0 & 0 & 1 \end{bmatrix} . \tag{6.8}$$

For this purpose the same positioning task have been performed five times increasing the error on $\boldsymbol{K}$ entries from 0% to 40%, at step of 10%, as shown in the following table:

Exp. N°	α_x	α_y	o_x	o_y	*Err*
1	1120.10	1120.19	342.05	199.57	0%
2	1008.09	1008.17	307.85	179.61	10%
3	896.08	896.15	273.64	159.66	20%
4	784.07	784.13	239.43	139.70	30%
5	672.06	672.11	205.23	119.74	40%

where each row correspond to the indicated percentage of error in the camera intrinsic matrix. For this mentioned task the initial end effector camera frame E_i is defined with respect to the world robot frame W by :

$t_{x\,i} = -0.8$	$t_{y\,i} = -0.16$	$t_{z\,i} = 0.62$
$\phi_i = -112.97$	$\theta_i = 28.65$	$\psi_i = 0$

while the goal end effector pose E_f is instead defined with respect to W by :

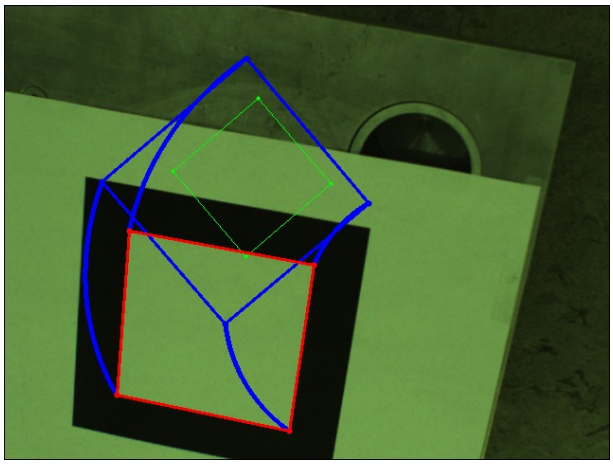
$t_{x\,f} = -0.9$	$t_{y\,f} = -0.4$	$t_{zf} = 0.7$
$\phi_f = -95.11$	$\theta_f = 18.79$	$\psi_f = 48.59$

The following table summarizes the positioning task errors resulting from the the five experiments according to error translation and rotation vectors definitions in equations (6.6) and (6.7):

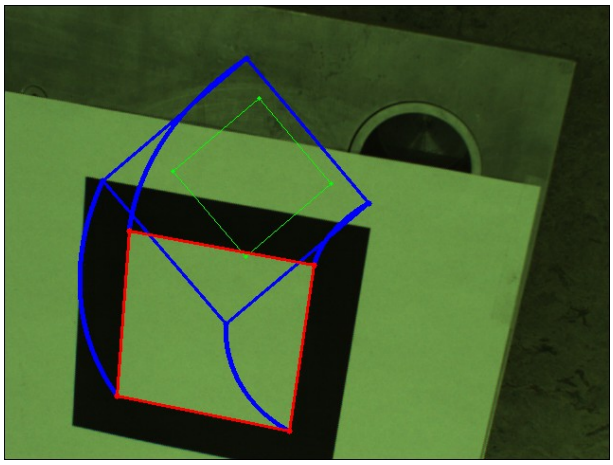
Exp. N°	$e_{p\,x}\,[mm]$	$e_{p\,y}\,[mm]$	$e_{p\,z}\,[mm]$	$e_{a\,\phi}\,[deg]$	$e_{a\,\theta}\,[deg]$	$e_{a\,\psi}\,[deg]$
1	1.0	1.1	0.7	0.082	0.031	0.100
2	−16	0.7	0.1	−0.444	−0.733	−1.17
3	7.2	4.7	0.1	−0.824	3.595	−2.332
4	8.5	47.2	0.4	−0.882	4.094	−2.602
5	−0.7	42	−0.7	−0.882	2.823	−2.555

As resulting from the table, the feedback image control, notwithstanding the often large camera calibration errors, drives the system toward the goal $3D$ configuration: this is one of the nice properties of the IBVS approach. Path planning ensures in these cases, the stability of the system for large camera displacements as well. Moreover off-line path

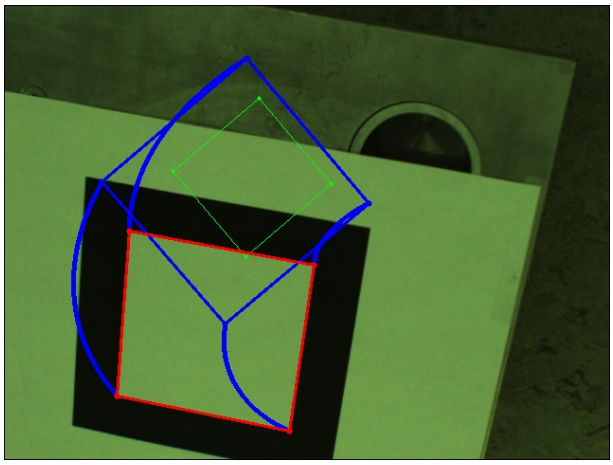
planning addressed in this thesis gives the possibility to choose the $3D$ shape of the camera trajectory for task accomplishment. In Figure (6.85) the pixel reprojected planned image trajectories (blue lines) driving to the goal view I_f (blue quadrangles) are shown for the the five positioning task runs. In figure (6.86) the corresponding $3D$ measured end-effector origin and camera trajectory are reported. As reported, also in $3D$ space at larger calibration errors correspond higher curvature and longer $3D$ end effector trajectories: the goal pose anyway is reached in all cases. For calibration errors larger than 40% the positioning has failed since the planned image paths would have driven the robot out of its workspace during task accomplishment.



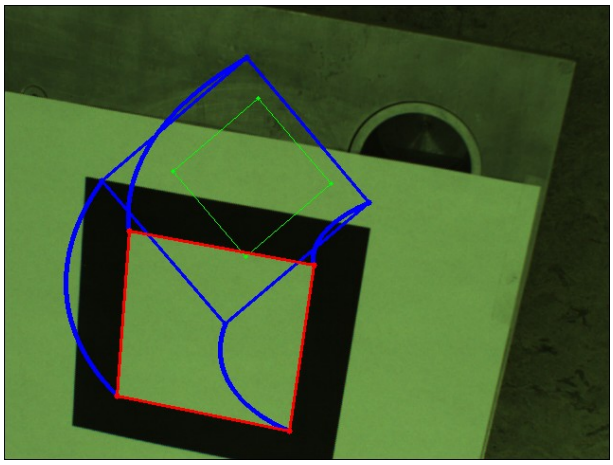
0% calibration error.



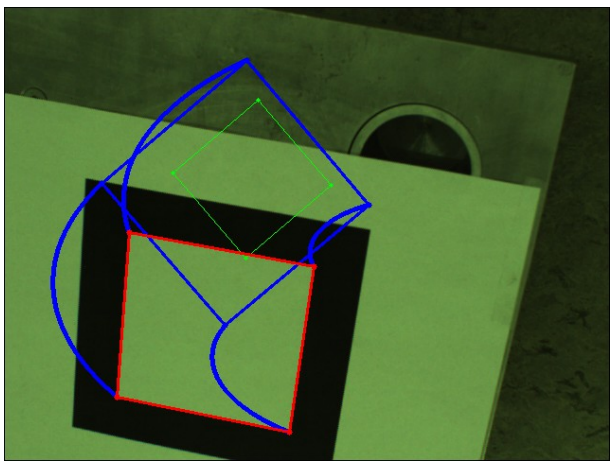
10% calibration error.



20% calibration error.

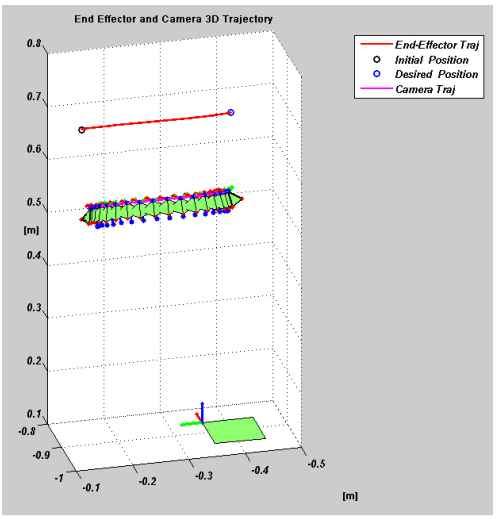


30% calibration error.

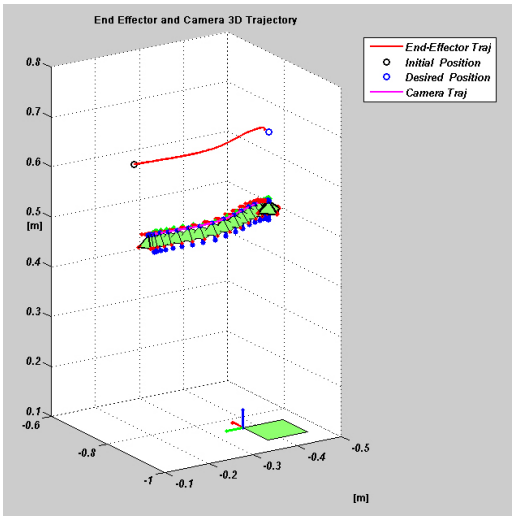


40% calibration error.

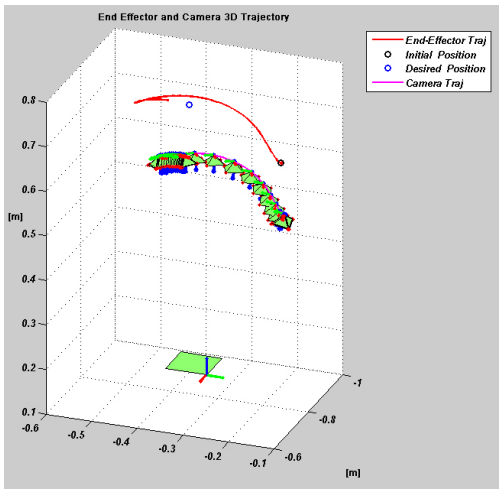
Figure 6.85: Image planning trajectories - blue lines - at increasing camera calibration error ranging from 0% to 40% at step of 10%: as shown at larger calibration errors correspond larger curvatures for the image planned trajectories. In green is reported the auxiliary view I_a necessary used tho solve the Euclidean homography decomposition for a planar target explained in Section (3.3.3).



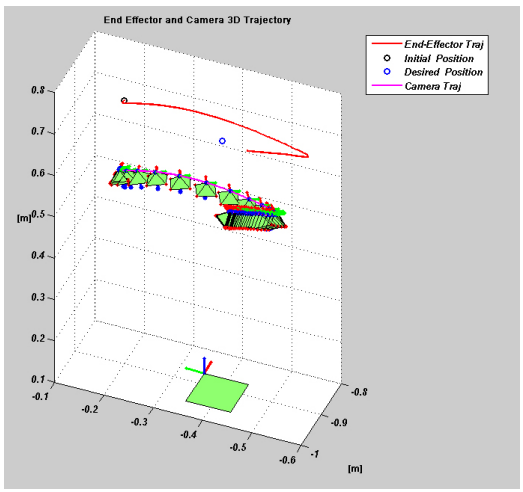
0% calibration error.



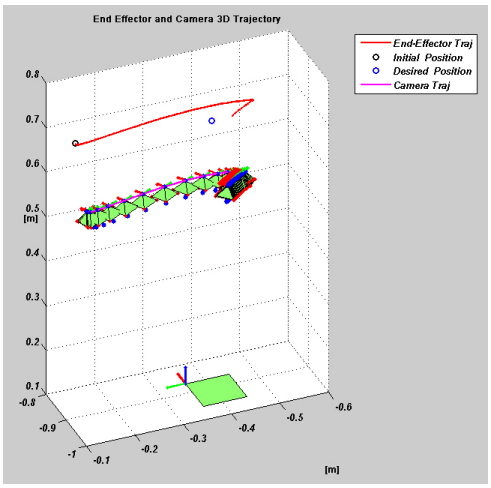
10% calibration error.



20% calibration error.



30% calibration error.



40% calibration error.

Figure 6.86: $3D$ end-effector origin and camera trajectories at increasing camera calibration error ranging from 0% to 40% at step of 10%: as shown at larger calibration errors the $3D$ end effector trajectories increase their lenghts and their curvatures.

Chapter 7

Conclusions

In this dissertation the control of a robotic system through an Image Visual Servoing approach is presented: the work considers a $6DOF$ manipulator with a camera mounted in eye-in-hand-configuration. The control is used to position the robot with respect to a fixed target placed in the camera field of view.
The scheme exploits image points or ellipses plus one point respectively for planar and axial-symmetric targets as visual features. The feature matching control step is manually performed by the user while imaged points are on-line automatically tracked.
The control strategy benefits of an innovative off-line image path planning strategy obtained from a scaled Euclidean reconstruction of the scene. By mean of this image paths is possible to drive the robot end effector and camera trajectories along an helical arc, joining the initial and the goal poses. Planned trajectories results feasible (i.e. compliant with the camera rigid body motion) and induce smooth velocity screws to the robot actuators avoiding singularity and local minima in particular for large camera displacements. Since path planning is executed off-line, it is possible to tune the planing parameter so as to prevent image feature to leave the camera field of view during task accomplishment.
The presented results concern simulation and real robot experiments. Simulation have been realized by the implementation of a Matlab© software while the experiment have been executed on by means of an anthropomorphic Kuka© robot provided with a fire-wire camera on its end effector.
Both simulation and real experiment shows the feasibility of the proposed approach and the good performances of the controlled system also in presence of camera calibration errors.
Some problems arises in the tracking part when the only available visual features are "unstable" due to light reflection phenomena or complex target textures. From this point of view it would be an challenging subject for future research to study innovative techniques able to deal with more "natural" target, eventually coping with feature disappearing and self-occlusions.

Bibliography

[AC99] B. Allotta and C. Colombo. On the use of linear camera-object interaction models in visual servoing. *IEEE Journal on Robotics and Automation*, 15(2):350–357, 1999.

[AF05] B. Allotta and D. Fioravanti. 3D motion planning for image-based visual servoing tasks. In *Proc. IEEE International Conference on Robotics and Automation (ICRA'05)*, Barcelona, Spain, 2005.

[AHE01] N. Andreff, R. Horaud, and B. Espiau. Robot hand-bye calibration using structure-from-motion. *International Journal of Robotics Research*, 20(3):228–248, March 2001.

[BM06] S. Benhimane and E. Malis. Homography-based 2d visual servoing. In *IEEE International Conference on Robotics and Automation*, Orlando, USA, 2006.

[Bou07] J. Y. Bouguet. Camera calibration toolbox for matlab. 2007. http://www.vision.caltech.edu/bouguetj/.

[Bru01] H. Bruyninckx. Open robot control software. 2001. http://www.orocos.org/.

[CA01] F. Conticelli and B. Allotta. Discrete-time robot visual feedback in 3D positioning tasks with depth adaptation. *IEEE/ASME Transaction on Mechatronics*, 6(3):356–363, 2001.

[CDBP05] C. Colombo, A. Del Bimbo, and F. Pernici. Metric 3D reconstruction and texture acquisition of surfaces of revolution from a single uncalibrated view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(1):99–114, 2005.

[CH04] G. Chesi and K. Hashimoto. A simple technique for improving camera displacement estimation in eye-in-hand visual servoing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1239–1242, 2004.

[Cha99] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. *The confluence of vision and control (D. Kriegman, G. Hager and A. Morse Eds.)*, 237:66–78, 1999. Lecture Notes in Control and Information Systems. Springer.

[DFD03] W. E. Dixon, Y. Fang, and D. M. Dawson. Adaptive range identification for exponential visual servo tracking. In *The 18th IEEE International Symposium on Intelligent Control-ISIC'03*, 2003. ISIC-0287.

[ECR92] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Journal on Robotics and Automation*, 8(3):313–326, 1992.

[Esp93] B. Espiau. Effect of calibration errors on visual servoing in robotics. In *Proc. 3rd International Workshop on Experimental Robotics*, pages 182–192, Kyoto, Japan, 1993.

[FPF99] A. W. Fitzgibbon, M. Pilu, and R. B. Fisher. Direct least square fitting of ellipses. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(5):476–480, 1999.

[GAMO05] N. Garcia-Aracil, E. Malis, and Others. Continuous visual servoing despite the changes of visibility in image features. *IEEE Transactions on Robotics*, 21(6):147–161, December 2005.

[HHC96] S. Hutchinson, G. D. Hager, and P. I. Corke. Tutorial on visual servo control. *IEEE Journal on Robotics and Automation*, 12(5):651–670, 1996.

[HZ03] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[Mal04] E. Malis. Visual servoing invariant to changes in camera-intrinsic parameters. *IEEE Journal on Robotics and Automation*, 20(1):72–81, 2004.

[MC02] Y. Mezouar and F. Chaumette. Path planning for robust image-based control. *IEEE Journal on Robotics and Automation*, 18(4):534–549, 2002.

[MCB99] E. Malis, F. Chaumette, and S. Boudet. 2-1/2-d visual servoing. *IEEE Journal on Robotics and Automation*, 15(2):238–250, 1999.

[MCB00] E. Malis, F. Chaumette, and S. Boudet. 2 1/2 d visual servoing with respect to unknown objects through a new estimation scheme of camera displacement. *International Journal of Computer Vision*, 37(1):79–97, June 2000.

[MSKS03] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry. *An Invitation to 3D Vision: From Images to Geometric Models*. Springer Verlag, 2003.

[Ope01] Open computer vision library. 2001. http://www.intel.com/ research/ mrl/ research/ opencv/.

[Soe06] P. Soetens. A software framework for real-time and distributed robot and machine control. In *PhD thesis, Department of Mechanical Engineering, Katholieke Universiteit*, Leuven, Belgium, 2006.

[WMC03] K.-Y. K. Wong, P. Mendonça, and R. Cipolla. Camera calibration from surfaces of revolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(2):147–161, 2003.