Robust image-based visual servoing using invariant visual information

Omar Tahri, Helder Araujo,, François Chaumette, Youcef Mezouar

Robust image-based visual servoing using invariant visual information

Omar Tahri, Helder Araujo, François Chaumette and Youcef Mezouar

Omar Tahri and Helder Araujo are with Institute for Systems and Robotics, Polo II 3030-290 Coimbra, Portugal, omartahri@isr.uc.pt, helder@isr.uc.pt
François Chaumette is with INRIA Rennes Bretagne Atlantique, campus de Beaulieu 35042 Rennes Cedex, France, francois.chaumette@irisa.fr
Youcef Mezouar is with the Institut Pascal, IFMA, Institut Pascal, BP 10448, F-63000 Clermont-Ferrand,France, youcef.mezouar@ifma.fr

Abstract

This paper deals with the use of invariant visual features for visual servoing. New features are proposed to control the 6 degrees of freedom of a robotic system with better linearizing properties and robustness to noise than the state of the art in image-based visual servoing. We show in this paper that by using these features the behavior of image-based visual servoing in task space can be significantly improved. Several experimental results are provided and validate our proposal.

Keywords:
Robust visual servoing, spherical projection

1. INTRODUCTION

Image-based visual servoing (IBVS) consists of using feedback information defined directly in the images provided by a vision sensor to control the motion of a dynamic system. Theoretically, IBVS is suitable only for small displacements. However in practice, it is robust and efficient for large displacements, but less than optimal in terms of 3-D motion [3]. A large spectrum of visual features can be extracted and used in the control loop. Besides, the choice of features directly influences the closed-loop dynamics in task-space. The points are the most simple and common features that can be extracted from an image. However, such features suffer from control coupling and non-linearities between the image space and 3D space. In

practice, this can lead to unpredictable behavior in the task space but also to divergence or convergence to local minima in some situations, especially when large displacements are to be performed. Other features can be derived from the points in image to improve the system behavior. Features including the distance between two points in the image plane and the orientation of the line connecting those two points were proposed in [8]. The relative area of two projected surfaces was proposed in [20] as a feature. A vanishing point and the horizon line were selected in [15], which ensures good decoupling between translational and rotational degrees of freedom (DOFs). In [11], the coordinates of points are expressed in a cylindrical coordinate system instead of the classical Cartesian one to improve the robot trajectory.

This paper belongs to a series of works aiming at the selection of optimal visual features for IBVS [17], [18], [19]. These previous works were concerned with approaches that consider performance measures to choose visual features with good decoupling and linearizing properties. The main idea is to apply non-linear transformations to classical image visual features (point coordinates, contours or image region) in order to obtain new visual information with better linearizing properties. The results obtained using such approaches have shown a superior behavior in task space and convergence rate with respect to point coordinates (for instance) [18]. However, applying non-linear transformations on data obtained from the sensor space changes the noise distribution. For instance, if the image noise is Gaussian white, the noise on invariant features proposed in [18] and [19] is no more Gaussian white since the applied transformation is nonlinear. In practice, if the noise level is low, the use of invariants allows obtaining adequate robustness. If the noise level increases, the use of invariants can lead to less robustness at convergence compared to the classical visual features. In this paper, further to proposing new invariants allowing better linearizing properties and decoupling than those proposed in [18] and [19], we address the problem of robustness to noise.

The features used in this paper are computed from the projection of points onto the unit sphere. This means that the method proposed can work not only with classical perspective cameras but can also be applied to wide-angle cameras obeying the unified model [9, 2]. Wide-angle cameras include catadioptric systems that combine mirrors and conventional cameras to create omnidirectional cameras providing 360^o panoramic views of a scene, or dioptric fish-eye lenses [1, 5]. It is highly desirable that such imaging systems have a single viewpoint [1], [16]. i.e there exists a single center of

projection, so that every pixel in the sensed image measures the irradiance of the light passing through the same viewpoint in one particular direction.

In the next section, the unified camera model and some basic definitions related to visual servoing are recalled as well as the notations used in the following of the paper. In Section 3, a new visual feature vector is proposed. More precisely, the first part of the feature vector we propose is defined by a new rotation vector computed from a set of projected points onto the unit sphere. We will show that this rotation vector has a direct link with the rotational motion (i.e the camera rotational velocities and the vector entries are linked with the identity matrix). Similarly to rotational velocities, a new feature derived from an invariant to rotation computed from projected points onto the unit sphere is proposed in this section to approach a linear system. The interaction matrix related to the proposed feature is derived and shown to behave as a constant matrix with respect to point depth under some assumptions. The invariant to rotation will be used to control the translational motion independently of the rotational ones. Furthermore, the sensitivity to image noise is considered by taking into account the noise propagation from image to the space of new features. Finally, in Section 4, numerous experimental results are presented confirming the validity of the proposed approach.

2. Background

2.1. Notations

The following notations will be used:

- $\mathbf{P} = (X, Y, Z)$: 3D point coordinates.
- $\mathbf{P_s} = (x_s, y_s, z_s)$: the coordinates of projected point onto the unit sphere.
- $\mathbf{P_v}$: virtual point defined by linear combination of the projected points onto the unit sphere.
- $\mathbf{m} = (x, y, 1)$: coordinates of projected point onto the image plane in metric units.
- $\mathbf{p}$: coordinates of projected point onto the image plane in pixels.

- $\cos\alpha_{ij} = \mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j}$: dot product between two projected points on the unit sphere.
- $d_{ij} = \sqrt{2-2\cos\alpha_{ij}}$: distance between two projected points on the unit sphere.
- s_{wij}: the new feature we propose to control the translations.
- Δ: area of triangles defined by three projected points on the sphere.
- I: A polynomial invariant to rotations; refer to (34)
- the variables followed by $*$ are computed for the camera desired pose.
- all the scalars are in italic.
- all the matrices and vectors are in bold
- $\mathbf{\Gamma}_v = (\mathbf{I} - \mathbf{v}\mathbf{v}^{\top})$ for any vector $\mathbf{v}$.

2.2. Camera Model

A unified model for central imaging systems has been proposed in [9]. It consists in modeling the central imaging systems by two consecutive projections: spherical and then perspective as shown in Fig. 1. The frames attached to the sphere $\mathcal{F}_m$ and to the perspective camera $\mathcal{F}_p$ are related by a simple translation of $-\xi$ along the Z-axis. Let $\mathbf{P}$ be a 3D point with coordinates $\mathbf{P} = (X, Y, Z)$ in $\mathcal{F}_m$. The world point $\mathbf{P}$ is projected onto:

$$\mathbf{m} = \begin{pmatrix} x, & y, & 1 \end{pmatrix} = \begin{pmatrix} \frac{X}{Z+\xi\|\mathbf{P}\|}, & \frac{Y}{Z+\xi\|\mathbf{P}\|}, & 1 \end{pmatrix} \tag{1}$$

and the coordinates of the projected points in the image plane are obtained after a plane-to-plane collineation $\mathbf{K}$: $\mathbf{p} = \mathbf{K}\mathbf{m}$, ($\mathbf{K}$ is a 3×3 matrix containing the camera intrinsic parameters). In the sequel, the matrix $\mathbf{K}$ and parameter ξ are assumed to be computed using a calibration method, for instance using the method proposed in [14]. In this case, the inverse projection onto the unit sphere can be obtained from:

$$\mathbf{P_s} = \gamma \begin{pmatrix} x, & y, & 1 - \frac{\xi}{\gamma} \end{pmatrix} \tag{2}$$

where

$$\gamma = \frac{\xi + \sqrt{1 + (1-\xi^2)(x^2+y^2)}}{1 + x^2 + y^2}.$$

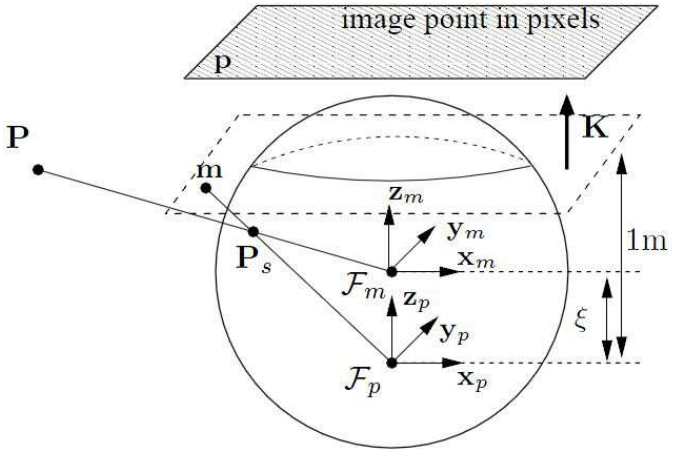


Figure 1: Unified image formation

The projection onto the unit sphere from the image plane is possible for all sensors obeying the unified model. In other words, it encompasses all sensors in this class namely [9]: perspective ($\xi = 0$) and catadioptric cameras ($\xi \neq 0$). A large class of fisheye cameras can also be represented by this model [6], [5].

2.3. Visual servoing

In few words, we recall that the time variation $\dot{\mathbf{s}}$ of the visual features $\mathbf{s}$ can be expressed linearly with respect to the relative camera-object kinematics screw $\boldsymbol{\tau} = (\boldsymbol{\upsilon}, \boldsymbol{\omega})$:

$$\dot{\mathbf{s}} = \mathbf{L}_\mathbf{s} \boldsymbol{\tau}, \tag{3}$$

where $\mathbf{L}_\mathbf{s}$ is the interaction matrix related to $\mathbf{s}$ [4]. In visual servoing, the control scheme is usually designed to reach an exponential decoupled decrease of the visual features error to their desired value $\mathbf{s}^*$. If we consider an eye-in-hand system observing a static object, the corresponding control law is [4]:

$$\boldsymbol{\tau}_c = -\lambda \widehat{\mathbf{L}_\mathbf{s}}^{+} (\mathbf{s} - \mathbf{s}^*), \tag{4}$$

where $\widehat{\mathbf{L}_\mathbf{s}}$ is a model or an approximation of $\mathbf{L}_\mathbf{s}$, $\widehat{\mathbf{L}_\mathbf{s}}^{+}$ the pseudo-inverse of $\widehat{\mathbf{L}_\mathbf{s}}$, λ a positive gain tuning the time to convergence, and $\boldsymbol{\tau}_c$ the camera velocity sent to the low-level robot controller.

3. Feature selection

Our goal is to select optimal visual features to control the translational and rotational motion. In other words, we seek to determine features that allow as much as possible linearizing properties between task space and image space and robustness to data noise. Selecting visual features by taking into account these two constraints will lead to a better behavior in the task space, avoiding local minima, divergence, and better accuracy.

3.1. Features to control rotation

In the following, we first recall some features previously proposed in the literature to control rotational motion. Then, a new rotation vector computed from a set of N points is defined. We show that this new feature has a linear relationship with respect to the rotational velocities.

3.1.1. Rotation vector to control the camera orientation

To control the rotational motion, a classical way is to use the rotation matrix ${}^{c*}\mathbf{R}_c$ between the current camera frame $\mathcal{F}_c$ and the desired one $\mathcal{F}_{c*}$. Canceling the rotational motions is equivalent to bring the value of ${}^{c*}\mathbf{R}_c$ to the identity matrix. As minimal representation of the rotation, the rotation vector $\theta\mathbf{u}$ is used. Recall that $0 < \theta < 2\pi$ is the rotation angle, while $\mathbf{u}$ defines the axis of rotation. In this case, canceling the rotational motion is equivalent to bring the rotation vector $\theta\mathbf{u}$ to a null vector. The rotation matrix can be computed from the rotation vector using the Rodrigues' rotation formula:

$$\mathbf{R} = \mathbf{I} + \sin(\theta)[\mathbf{u}]_\times + (1 - \cos(\theta))(\mathbf{u}\mathbf{u}^\top - \mathbf{I}) \tag{5}$$

where $[\mathbf{u}]_\times$ is the skew-symmetric matrix of the vector $\mathbf{u}$. Conversely, the rotation vector can be recovered from the rotation matrix by:

$$\theta = arcos(trace(\mathbf{R}) - 1) \tag{6}$$

and if $trace(\mathbf{R}) \neq 1$, then:

$$[\theta\mathbf{u}]_\times = \frac{1}{2}\frac{1}{sinc(\theta)}(\mathbf{R} - \mathbf{R}^\top) \tag{7}$$

where $sinc(\theta) = \frac{\sin(\theta)}{\theta}$. In $3D$ or $21/2D$ visual servoing schemes, using $\theta\mathbf{u}$ to control rotational motions ensures a linear link between the rotational velocities and the feature errors. However, computing $\theta\mathbf{u}$ requires a partial or a complete reconstruction of the camera pose to compute the rotation ${}^{c*}\mathbf{R}_c$. Nevertheless, as it will be explained in the following, a rotation vector can also be expressed directly in the image space.

3.1.2. Rotation vector feature from two points in the image

Recently, in [7] and [19], an angle-axis representation of a rotation matrix $\mathbf{R}$ computed from two projected points on the sphere has been considered to control rotational motions for visual servoing application. The idea behind the rotation formula given in these works is equivalent to attaching an orthonormal frame basis to each camera pose using two projected points onto the unit sphere. More precisely, let $\mathbf{P}_{s_1}$ and $\mathbf{P}_{s_2}$ be two projected points on the sphere for the current camera pose and ${\mathbf{P}^*}_{s_1}$ and $\mathbf{P}^*_{s_2}$ their corresponding projected points for the desired camera pose. From $\mathbf{P}_{s_1}$ and $\mathbf{P}_{s_2}$, it is possible to define an orthonormal basis ${}^c\mathbf{R}_s = [\mathbf{v}_1; \mathbf{v}_2; \mathbf{v}_3]$ (see Figure 2) such that:

$$\mathbf{v}_1 = \mathbf{P}_{s_1}, \tag{8}$$

$$\mathbf{v}_2 = \frac{\mathbf{P}_{s_2} - (\mathbf{P}_{s_2}^\top \mathbf{P}_{s_1})\mathbf{P}_{s_1}}{\| \mathbf{P}_{s_2} - (\mathbf{P}_{s_2}^\top \mathbf{P}_{s_1})\mathbf{P}_{s_1} \|}, \tag{9}$$

$$\mathbf{v}_3 = \mathbf{v}_1 \times \mathbf{v}_2. \tag{10}$$

and similarly an ortho-normal basis ${}^{c*}\mathbf{R}_{s*}$ using points ${\mathbf{P}^*}_{s_1}$ and $\mathbf{P}^*_{s_2}$ can be defined. If only a rotational motion is considered, the rotation matrix $\mathbf{R}$ between the current and the desired camera poses is determined by the matrix that transforms the vector basis ${}^c\mathbf{R}_s$ to ${}^{c*}\mathbf{R}_{s*}$:

$$\mathbf{R} = {}^{c*}\mathbf{R}_{s*} \, {}^c\mathbf{R}_s^\top \tag{11}$$

This property ensures a direct link between the rotation vector defined from ${}^{c*}\mathbf{R}_{s*} \, {}^c\mathbf{R}_s^\top$ and the rotational velocities as it has been proven in [19]. In the next paragraph, we define a rotation vector from a set of N points in the image.

3.1.3. Rotation vector feature from N points

For the sake of robustness, all projected points on the sphere should be used and not only two. In this work, we propose a way for defining a rotation vector using all the points. Our idea is based on the fact that the rotation matrix given by (11) can be obtained from two real projected points as well

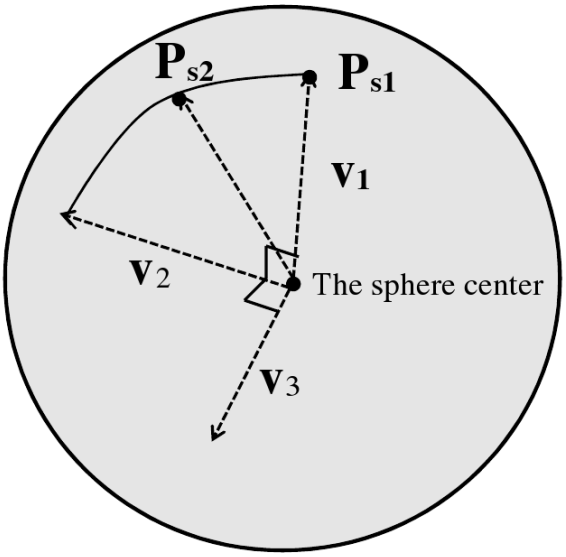


Figure 2: Definition of vector basis from 2 projected points on the unit sphere

as from two virtual points rigidly attached to the set of projected points. Let us consider:

$$\mathbf{P}_{v_1} = \sum_{i=1}^{N} a_{1i}\mathbf{P}_{s_i}, \quad \mathbf{P}_{v_2} = \sum_{i=1}^{N} a_{2i}\mathbf{P}_{s_i} \tag{12}$$

two virtual points obtained by a linear combination of the real set of projected points on the sphere. Then, from $\mathbf{P}_{v_1}$ and $\mathbf{P}_{v_2}$ an orthogonal basis can be defined as follows:

$$\mathbf{v}_{n1} = \frac{\mathbf{P}_{v_1}}{\| \mathbf{P}_{v_1} \|} \tag{13}$$

$$\mathbf{v}_{n2} = \frac{\mathbf{P}_{v_2} - \mathbf{P}_{v_2}^{\top}\mathbf{v}_{n1}\mathbf{v}_{n1}}{\| \mathbf{P}_{v_2} - \mathbf{P}_{v_2}^{\top}\mathbf{v}_{n1}\mathbf{v}_{n1} \|} \tag{14}$$

$$\mathbf{v}_{n3} = \mathbf{v}_{n1} \times \mathbf{v}_{n2} \tag{15}$$

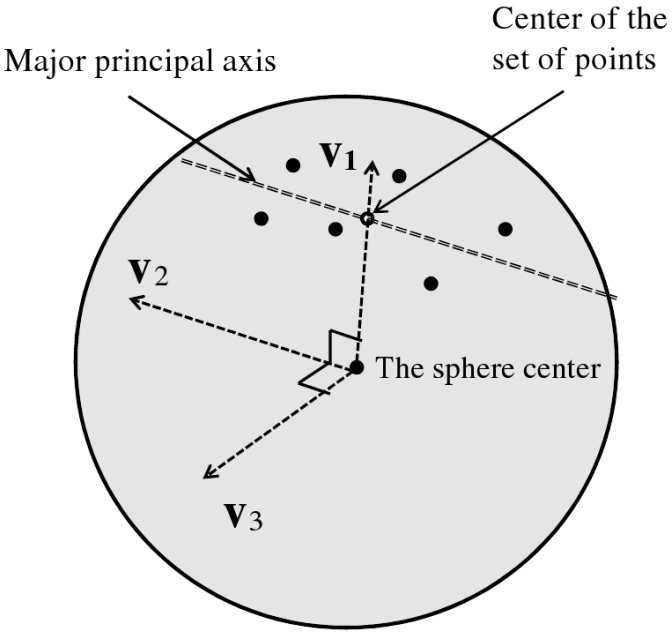


Figure 3: Definition of vector basis from N projected points on the unit sphere

Lemma 1. *If only a rotational motion is considered, the rotation matrix* $\mathbf{R}$ *between the current and the desired camera poses is determined by the matrix that transforms the vector basis* ${}^{c}\mathbf{R}_n$ *into* ${}^{c*}\mathbf{R}_{n*}$*:*

$$\mathbf{R} = {}^{c*}\mathbf{R}_{n*} \, {}^{c}\mathbf{R}_n^{\top} \tag{16}$$

where:

$${}^{c}\mathbf{R}_{\mathbf{n}} = [\mathbf{v}_{n1};\, \mathbf{v}_{n2};\, \mathbf{v}_{n3}]$$

and

$${}^{c*}\mathbf{R}_{\mathbf{n}*} = [\mathbf{v}^*_{n1};\, \mathbf{v}^*_{n2};\, \mathbf{v}^*_{n3}]$$

are computed using (12), (13), (14) and (15).

The proof of the previous lemma is detailed Appendix A. Note that the matrices ${}^{c}\mathbf{R}_{\mathbf{n}}^{\top}$ and ${}^{c*}\mathbf{R}_{\mathbf{n}*}$ are themselves rotation matrices since the triplet $(\mathbf{v}_{n1};\, \mathbf{v}_{n2};\, \mathbf{v}_{n3})$ forms a direct and ortho-normal basis. As will be shown in Section 3.1.5, equation (16) allows to obtain a direct link between the rotational velocities and the rotation vector computed from ${}^{c*}\mathbf{R}_{\mathbf{n}*} \, {}^{c}\mathbf{R}_{\mathbf{n}}^{\top}$.

3.1.4. Computation of the parameters a_{1i} and a_{2i}

To control the rotational motions, the rotation vector $\theta\mathbf{u}$ ($^{c}\mathbf{R}_{\mathbf{n}}\,{}^{n*}\mathbf{R}_{\mathbf{c}*}$) will be used. It is then necessary to determine the parameters a_{1i} and a_{2i} in (12). More precisely, we have to define two virtual points $\mathbf{P}^*_{v_1}$ and $\mathbf{P}^*_{v_2}$ and then express them as linear combinations of the projected points on the sphere $\mathbf{P}^*_{si}$ computed for the camera pose to be estimated. For the sake of simplicity, $\mathbf{P}^*_{v_1}$ and $\mathbf{P}^*_{v_2}$ are chosen to be unitary and perpendicular to each-other. In this case, the basis $[\mathbf{P}^*_{v_1};\ \mathbf{P}^*_{v_2};\ \mathbf{P}^*_{v_1} \times \mathbf{P}^*_{v_2}]$ is orthonormal.

In any given frame basis $[\mathbf{P}^*_{v_1};\ \mathbf{P}^*_{v_2};\ \mathbf{P}^*_{v_1} \times \mathbf{P}^*_{v_2}]$, each projected point onto the sphere can be expressed as:

$$\mathbf{P}^*_{si} = b_{1i}\mathbf{P}^*_{v_1} + b_{2i}\mathbf{P}^*_{v_2} + b_{3i}\mathbf{P}^*_{v_1} \times \mathbf{P}^*_{v_2} \tag{17}$$

Let $\mathbf{B}$ be the $3 \times N$ matrix that defines the coordinates of all the projected points on the new frame basis. We have:

$$\mathbf{P}^*_{st} = \mathbf{P}_{v*}\,\mathbf{B}, \tag{18}$$

where:

$$\mathbf{P}^*_{st} = [\mathbf{P}^*_{s1}\ \ \mathbf{P}^*_{s2} \ldots \mathbf{P}^*_{sN}],$$
$$\mathbf{P}^*_{v} = [\mathbf{P}^*_{v_1}\ \ \mathbf{P}^*_{v_2}\ \ \mathbf{P}^*_{v_1} \times \mathbf{P}^*_{v_2}],$$

and

$$\mathbf{B} = \mathbf{P}^{\top}_{v*}\,\mathbf{P}^*_{st} \tag{19}$$

In practice, a_{1i} and a_{2i} have to be chosen such that their corresponding virtual points are robust to noise. Our choice is based on characteristic features of 3D structure—the center of gravity of the directions defined by the points and the principal axis of the directions (also defined by the points). More precisely, the first virtual point is defined by:

$$\mathbf{P}^*_{v_1} = \frac{1}{\| \sum_{i=1}^{N} \mathbf{P}^*_{si} \|} \sum_{i=1}^{N} \mathbf{P}^*_{si}. \tag{20}$$

which corresponds to $a_{1i} = \frac{1}{\|\sum_{i=1}^{N} \mathbf{P}^*_{si}\|}$. The second virtual point $\mathbf{P}^*_{v_2}$ is chosen as the unitary vector perpendicular to $\mathbf{P}^*_{v_1}$ that lays on the plane defined by $\mathbf{P}^*_{v_1}$ and the major principal axis of the set of the projected points on the sphere (see Fig. 3). The choice of the main principal axis as second axis allows having the majority of the points in its direction. Now as $\mathbf{P}^*_{v_1}$ and

$\mathbf{P}^*_{v_2}$ have been determined, the matrix $\mathbf{B}$ can be computed using (19). From (17), it can be obtained that:

$$\mathbf{P}^*_{si} - b_{1i}\mathbf{P}^*_{v_1} = b_{2i}\mathbf{P}^*_{v_2} + b_{3i}\mathbf{P}^*_{v_1} \times \mathbf{P}^*_{v_2} \tag{21}$$

By replacing $\mathbf{P}^*_{v_1}$ by $\frac{\sum_{i=1}^{N}\mathbf{P}^*_{si}}{\|\sum_{i=1}^{N}\mathbf{P}^*_{si}\|}$, we obtain for all the projected points onto the sphere:

$$\mathbf{P}^*_{st}\mathbf{C} = \left[\begin{array}{cc} \mathbf{P}^*_{v_2} & \mathbf{P}^*_{v_1} \times \mathbf{P}^*_{v_2} \end{array}\right] \mathbf{B_{23}} \tag{22}$$

where $\mathbf{B_{23}}$ is composed by the two last rows of $\mathbf{B}$ and $\mathbf{C}$ is an $N \times N$ matrix defined by $c_{lm} = 1 - \frac{b_{1l}}{\|\sum_{i=1}^{N}\mathbf{P}^*_{si}\|}$ if $l = m$ and $c_{lm} = -\frac{b_{1l}}{\|\sum_{i=1}^{N}\mathbf{P}^*_{si}\|}$ for $l \neq m$. By inverting (22), we obtain:

$$\left[\begin{array}{cc} \mathbf{P}^*_{v_2} & \mathbf{P}^*_{v_1} \times \mathbf{P}^*_{v_2} \end{array}\right] = \mathbf{P}^*_{st}\,\mathbf{C}\,{\mathbf{B_{23}}}^{+}. \tag{23}$$

The parameters a_{2i} are then obtained as the first column of the matrix $\mathbf{C}\,{\mathbf{B_{23}}}^{+}$. Now as the coefficients a_{1i} and a_{2i} are defined, the vector basis for each camera pose can be obtained using (13), (14) and (15).

3.1.5. Interaction matrices

In this paragraph, we compute the interaction matrix related to $\theta\mathbf{u}({}^{c}\mathbf{R}_{n}\,{}^{n*}\mathbf{R}_{c*})$. In order to do so, let us first recall the result obtained in [19] for the case where the rotation is defined as $\theta\mathbf{u}({}^{c}\mathbf{R}_{s}\,{}^{s*}\mathbf{R}_{c*})$. In [12], [19], it has been shown that:

$$\frac{d\theta\mathbf{u}}{dt} = -\mathbf{L}_{\omega}(\theta,\mathbf{u})\boldsymbol{\zeta} \tag{24}$$

where

$$\boldsymbol{\zeta} = {}^{c}\mathbf{R}_{s}[\,{}^{s}\mathbf{R}_{c}\,{}^{c}\dot{\mathbf{R}}_{s}]^{\times}, \tag{25}$$

$$\mathbf{L}_{\omega}(\theta,\mathbf{u}) = -\mathbf{I} + \frac{\theta}{2}[\mathbf{u}]_{\times} - \left(1 - \frac{\mathrm{sinc}(\theta)}{\mathrm{sinc}^2(\theta/2)}\right)[\mathbf{u}]^2_{\times}\,, \tag{26}$$

with $\mathrm{sinc}(x) = \sin(x)/x$ and $[\mathbf{M}]^{\times}$ is the vector associated with the antisymmetric matrix $\mathbf{M}$. Since ${}^{c}\mathbf{R}_{s} = [\mathbf{v}_1\ \mathbf{v}_2\ \mathbf{v}_3]$, it can be shown that the vector $\boldsymbol{\zeta}$ can be obtained by [19]:

$$\boldsymbol{\zeta} = [\mathbf{v_1}\ \mathbf{v_2}\ \mathbf{v_3}] \begin{bmatrix} -\mathbf{v_2}^\top \dot{\mathbf{v}}_\mathbf{3} \\ -\mathbf{v_3}^\top \dot{\mathbf{v}}_\mathbf{1} \\ \mathbf{v_2}^\top \dot{\mathbf{v}}_\mathbf{1} \end{bmatrix}. \tag{27}$$

By taking the derivative of $\mathbf{v_1}$, $\mathbf{v_2}$ and $\mathbf{v_3}$ and after tedious computations, it has been shown in [19] that:

$$\boldsymbol{\zeta} = \mathbf{L}_{\omega,v}\boldsymbol{v} - \boldsymbol{\omega}, \tag{28}$$

with

$$\mathbf{L}_{\omega,v} = \frac{1}{\| \mathbf{P_1} \|}(\delta_p \mathbf{v_1}\mathbf{v_3}^\top + \mathbf{v_2}\mathbf{v_3}^\top - \mathbf{v_3}\mathbf{v_2}^\top)$$

and $\delta_p = \frac{(\mathbf{P_{s1}}^\top \mathbf{P_{s2}})\,\|\mathbf{P_2}\| - \|\mathbf{P_1}\|}{\|\mathbf{P_2}\|\,\|\Gamma_{\mathbf{P_{s1}}}(\mathbf{P_{s2}} - \mathbf{P_{s1}})\|}$. Identically to the result shown in [19], the time variation of $\theta\mathbf{u}({}^c\mathbf{R}_n\ {}^{n*}\mathbf{R}_{c*})$ can be obtained by:

$$\frac{d\theta\mathbf{u}}{dt} = -\mathbf{L}_\omega(\theta, \mathbf{u})\boldsymbol{\zeta}_n \tag{29}$$

where $\boldsymbol{\zeta}_n$ is defined by:

$$\boldsymbol{\zeta}_n = [\mathbf{v_{n1}}\ \mathbf{v_{n2}}\ \mathbf{v_{n3}}] \begin{bmatrix} -\mathbf{v_{n2}}^\top \dot{\mathbf{v}}_\mathbf{n3} \\ -\mathbf{v_{n3}}^\top \dot{\mathbf{v}}_\mathbf{n1} \\ \mathbf{v_{n2}}^\top \dot{\mathbf{v}}_\mathbf{n1} \end{bmatrix}. \tag{30}$$

with ${}^c\mathbf{R}_n = [\mathbf{v}_{n1}\ \mathbf{v}_{n2}\ \mathbf{v}_{n3}]$. Let us express $\boldsymbol{\zeta}_n$ as:

$$\boldsymbol{\zeta}_n = \boldsymbol{\zeta}_{n\boldsymbol{v}}\boldsymbol{v} + \boldsymbol{\zeta}_{n\boldsymbol{\omega}}\boldsymbol{\omega} \tag{31}$$

By taking the derivative of the formulas of $\mathbf{v_{n1}}$, $\mathbf{v_{n2}}$ and $\mathbf{v_{n3}}$ and after tedious computations, it can be obtained that (the details are given in Appendix B):

$$\boldsymbol{\zeta}_{n\boldsymbol{\omega}} = -\mathbf{I_3} \tag{32}$$

This result shows that the direct link between $\theta\mathbf{u}({}^c\mathbf{R}_s\ {}^{s*}\mathbf{R}_{c*})$ and the rotational velocities is preserved using the new features $\theta\mathbf{u}({}^c\mathbf{R}_n\ {}^{n*}\mathbf{R}_{c*})$. Furthermore, we have:

$$\boldsymbol{\zeta}_{n\boldsymbol{v}} = \delta_{13}\mathbf{v_{n1}}\mathbf{v_{n3}}^\top + \delta_{23}[\mathbf{v_{n1}}]_\times \tag{33}$$

where

$$\begin{cases} \delta_{13} = \sum_{i=1}^{N} \frac{(a_{1i}\mathbf{P}_{v_2}^{\top}\mathbf{v_{n1}} - a_{2i}\|\mathbf{P}_{v_1}\|)(-\mathbf{I}_3 + \mathbf{P}_{s_i}\mathbf{P}_{s_i}^{\top})}{\|\mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^{\top}\mathbf{v_{1n}})\mathbf{v_{1n}}\| \, \|\mathbf{P}_{v_1}\| \, \|\mathbf{P_i}\|} \cdot \\ \delta_{23} = \sum_{i=1}^{N} \frac{a_{1i}}{\|\mathbf{P}_{v_1}\|\|\mathbf{P_i}\|}(-\mathbf{I}_3 + \mathbf{P}_{s_i}\mathbf{P}_{s_i}^{\top}) \end{cases}$$

3.2. Features to control translations

3.2.1. Invariants to rotations

Using the spherical projection model, the shape of an object does not change under rotational motions of the sphere. After a rotational motion of the sensor frame, it can easily be shown that the projected shape onto the sphere undergoes the same rotational motion as the coordinates of the 3D points of the object expressed in the camera frame. This means that the invariants to rotation in 3D space remain invariant if the 3D points are projected onto the unit sphere. In the following, first, we will recall some invariants proposed in previous works to control the translational motions. Then, a new invariant feature is proposed and its corresponding interaction matrix is calculated.

In [18], two different invariants have been proposed to control the translational motions:

- An invariant polynomial defined by:

$$I = m_{200}m_{020} - m_{200}m_{002} + m_{110}^2 + m_{101}^2 - m_{020}m_{002} + m_{011}^2 \tag{34}$$

  where $m_{i,j,k}$ is the 3D moment of order $i+j+k$ computed from a discrete set of points defined by the following classical equation:

$$m_{i,j,k} = \sum_{h=1}^{N} x_h^i \; y_h^j \; z_h^k \tag{35}$$

  where $(x_h,\ y_h,\ z_h)$ are the coordinates of the h^{th} point and N is the number of points. In order to ensure the non-singularity of the interaction matrix, the set of points is divided into four subsets (each subset must contain at least three points). For each subset of points the invariant given by (34) is computed, which allows obtaining 4 invariants to rotation to control the 3 translational *DOFs*.

- The area of triangles built by three projected points on the sphere:

$$\Delta = \frac{1}{2} \parallel (\mathbf{P}_{s_2} - \mathbf{P}_{s_1}) \times (\mathbf{P}_{s_3} - \mathbf{P}_{s_1}) \parallel \tag{36}$$

where $\mathbf{P}_{s_1} = (x_{s_1}, y_{s_1}, z_{s_1})$, $\mathbf{P}_{s_2} = (x_{s_2}, y_{s_2}, z_{s_2})$, $\mathbf{P}_{s_3} = (x_{s_3}, y_{s_3}, z_{s_3})$ are the coordinates of the vertices of the triangle projected onto the unit sphere. To control the 3 translational *DOFs*, the areas of all possible triangles or 4 selected ones such that the interaction matrix is not singular are used.

In order to obtain a system behavior close to that of a linear system, two invariants $I_l = \frac{1}{\sqrt{I}}$ and $\Delta_l = \frac{1}{\sqrt{\Delta}}$ have been derived from I and Δ in [18]. Such features allowed obtaining an interaction matrix that is less sensitive to depth variations.

Recently, in [19], three invariants to camera rotations have been defined using the Cartesian distances between the spherical projections of three points d_{12}, d_{13} and d_{23} defined by:

$$d_{ij} = \sqrt{2 - 2\cos\alpha_{ij}} \tag{37}$$

where $\cos\alpha_{ij} = \mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j}$. As for I and Δ, a new feature s_{ij} is derived in this work from d_{ij} to obtain more linearizing properties. In practice, the distance between two projected points behaves as a function inversely proportional to the distances of the corresponding 3D points to the center of projection. Therefore, $s_{ij} = \frac{1}{d_{ij}}$ will be close to a function proportional to the distances from the corresponding 3D points to the center of projection.

3.2.2. Interaction matrix and linearizing properties

As mentioned in the beginning of this section, our goal is to select features minimizing the non-linearities between the task space and feature space. This is equivalent to selecting features such their corresponding interaction matrix is not too much influenced by the camera position with respect to the object.

The interaction matrix that links the variation of d_{ij} with respect to translational displacement is given by:

$$\mathbf{L}_{d_{ij}} = \begin{bmatrix} \mathbf{L}_{d_{ij}}\boldsymbol{v} & 0 & 0 & 0 \end{bmatrix} \tag{38}$$

the three last entries of $\mathbf{L}_{d_{ij}}$ are null thanks to the invariance to rotations, while the three first ones corresponding to the translational motion are given by:

$$\mathbf{L}_{d_{ij}}\boldsymbol{v} = -\frac{\mathbf{P}_{s_i}^{\top}\mathbf{L}_{\mathbf{P}_{s_j}}\boldsymbol{v} + \mathbf{P}_{s_j}^{\top}\mathbf{L}_{\mathbf{P}_{s_i}}\boldsymbol{v}}{d_{ij}} \tag{39}$$

where $\mathbf{L}_{\mathbf{P}_{s_i}}\boldsymbol{v}$ and $\mathbf{L}_{\mathbf{P}_{s_j}}\boldsymbol{v}$ are the interaction matrices that relate the variations of the point coordinates on the unit sphere to the camera translational displacements. This interaction matrix has the following form [10]:

$$\mathbf{L}_{\mathbf{P}_{s_i}}\boldsymbol{v} = \frac{-\mathbf{I} + \mathbf{P}_{s_i}\mathbf{P}_{s_i}^{\top}}{\parallel \mathbf{P_i} \parallel} \tag{40}$$

where $\parallel \mathbf{P_i} \parallel$ is the distance of the 3D point to the center of the sphere. After inserting (40) in (39), we obtain:

$$\mathbf{L}_{d_{ij}}\boldsymbol{v} = -\frac{1}{d_{ij}}\left((-\frac{1}{\parallel \mathbf{P_j} \parallel} + \frac{\mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j}}{\parallel \mathbf{P_i} \parallel})\mathbf{P}_{s_i}^{\top} + (-\frac{1}{\parallel \mathbf{P_i} \parallel} + \frac{\mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j}}{\parallel \mathbf{P_j} \parallel})\mathbf{P}_{s_j}^{\top}\right) \tag{41}$$

Further to the invariance to rotation, it is also possible to decrease the non-linearities between the image space and 3D space. Indeed, the distance d_{ij} on the sphere behaves as function which is approximately inversely proportional to the point depths $\parallel \mathbf{P_i} \parallel$. This means that its corresponding interaction matrix depends on the square of the inverse of the point depths. On the other hand, the inverse of the distance behaves approximately as a linear function of the points depths. This should allow obtaining more linearizing properties between the image space and 3D space. Consequently, we propose to use $s_{ij} = 1/d_{ij}$ for all possible combinations of two projected points. Let us consider the case when the "mean" distance R of the points to the sphere center is such that $R \approx \parallel \mathbf{P_i} \parallel \approx \parallel \mathbf{P_j} \parallel$ as shown in Figure 4. In this case, we have:

$$\mathbf{L}_{s_{ij}}\boldsymbol{v} \approx \frac{(-1 + \mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j})}{R}\frac{(\mathbf{P}_{s_i} + \mathbf{P}_{s_j})^{\top}}{d_{ij}^3} \tag{42}$$

Note that $-1 + \mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j} = -\frac{d_{ij}^2}{2}$, then (42) can be written as:

$$\mathbf{L}_{s_{ij}}\boldsymbol{v} \approx \frac{\parallel \mathbf{P}_{s_i} + \mathbf{P}_{s_j} \parallel}{2R\, d_{ij}}\frac{(\mathbf{P}_{s_i} + \mathbf{P}_{s_j})^{\top}}{\parallel \mathbf{P}_{s_i} + \mathbf{P}_{s_j} \parallel} \tag{43}$$

From Figure 4, we have:

$$\frac{d_{ij}}{D_{ij}} = \frac{1}{R} \tag{44}$$

By combining (44) with (43), we obtain:

$$\mathbf{L}_{s_{ij}}\boldsymbol{v} \approx \frac{\parallel \mathbf{P}_{s_i} + \mathbf{P}_{s_j} \parallel}{2D_{ij}}\frac{(\mathbf{P}_{s_i} + \mathbf{P}_{s_j})^{\top}}{\parallel \mathbf{P}_{s_i} + \mathbf{P}_{s_j} \parallel} \tag{45}$$

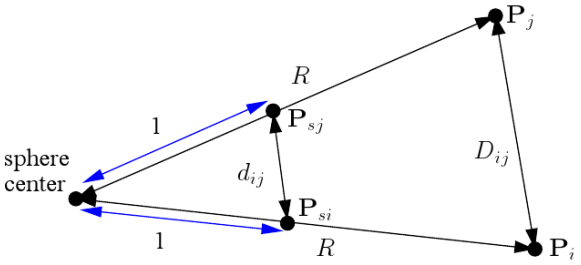


Figure 4: Relation between 3D distance and distance between projected point on the sphere

Note that $\frac{(\mathbf{P}_{s_i}+\mathbf{P}_{s_j})^\top}{\|\mathbf{P}_{s_i}+\mathbf{P}_{s_j}\|}$ is the unitary vector that passes through the middle of the two points $\mathbf{P}_{s_i}$ and $\mathbf{P}_{s_j}$ and also $\| \mathbf{P}_{s_i} + \mathbf{P}_{s_j} \| \approx 2$ if $R >> D_{ij}$. This means that the matrix $\mathbf{L}_{s_{ij}}\boldsymbol{v}$ behaves as a constant matrix when point depth increases. This allows the system to behave as a linear one.

3.2.3. Sensitivity to noise

In Section 3.1.3, a rotation vector robust to image noise was defined by using all the projected points in order to control the rotational velocities. For the translational velocities, as was previously described, feature $\frac{1}{d_{ij}}$ depends almost linearly on the point depths. However, this feature is a non-linear function of the point coordinates in the image plane. Therefore the propagation of noise from the image to the feature s_{ij} should be taken into account. Let us start with the sensitivity of a projected point onto the sphere with respect to noise in the image plane. Taking the derivative of (2), the variation in the coordinates of the point projected onto the sphere as a function of the variation in the coordinates in the image points (noise-meters) is obtained by (using first order approximation):

$$\Delta\mathbf{P}_s = \mathbf{J}_{\mathbf{P}_s/\mathbf{m}}\Delta\mathbf{m} \tag{46}$$

where:

$$\mathbf{J}_{\mathbf{P}_s/\mathbf{m}} = \begin{bmatrix} \gamma + x\frac{\partial\gamma}{\partial x} & x\frac{\partial\gamma}{\partial y} & 0 \\ y\frac{\partial\gamma}{\partial x} & \gamma + y\frac{\partial\gamma}{\partial y} & 0 \\ \frac{\partial\gamma}{\partial x} & \frac{\partial\gamma}{\partial y} & 0 \end{bmatrix} \tag{47}$$

with:

$$\begin{aligned}\frac{\partial\gamma}{\partial x} &= \frac{x}{1+x^2+y^2}\left(\frac{(1-\xi^2)}{\sqrt{1+(1-\xi^2)(x^2+y^2)}}-2\gamma\right)\\ \frac{\partial\gamma}{\partial y} &= \frac{y}{1+x^2+y^2}\left(\frac{(1-\xi^2)}{\sqrt{1+(1-\xi^2)(x^2+y^2)}}-2\gamma\right)\end{aligned} \tag{48}$$

where γ and ξ have been defined in Section 2.2. Therefore, the variation of $\mathbf{P}_s$ with respect to image points in pixels is obtained by:

$$\Delta\mathbf{P}_s = \mathbf{J}_{\mathbf{P}_s/\mathbf{m}}\mathbf{K}^{-1}\Delta\mathbf{p} \tag{49}$$

Furthermore, from $d_{ij} = \sqrt{2-2\mathbf{P}_{si}^{\top}\mathbf{P}_{sj}}$, we have:

$$\Delta d_{ij} = -\frac{1}{d_{ij}}(\mathbf{P}_{sj}^{\top}\,\Delta\mathbf{P}_{si} + \mathbf{P}_{si}^{\top}\,\Delta\mathbf{P}_{sj}) \tag{50}$$

As a result of (47) and (50), the variation of $s_{ij} = \frac{1}{d_{ij}}$ with respect to noise in the coordinates of the image points (in pixels) is obtained by:

$$\Delta s_{ij} = \mathbf{J}_{s_{ij}/\mathbf{p}}\begin{bmatrix}\Delta\mathbf{p}_i\\ \Delta\mathbf{p}_j\end{bmatrix} \tag{51}$$

where $\mathbf{J}_{s_{ij}/\mathbf{p}} = \begin{bmatrix}\mathbf{P}_{s_j}^{\top}\ \mathbf{J}_{\mathbf{P}_{si}/\mathbf{m_i}}\mathbf{K}^{-1} & \mathbf{P}_{s_i}^{\top}\ \mathbf{J}_{\mathbf{P}_{sj}/\mathbf{m_j}}\mathbf{K}^{-1}\end{bmatrix}/d_{ij}^3$. In order to take into account the noise propagation effect of the non-linear mapping from the image point coordinates to the features s_{ij}, each visual feature should be weighted by $\frac{1}{\|\mathbf{J}_{s_{ij}^*/\mathbf{p}^*}\|}$ computed using the image points coordinates corresponding to the desired pose. More precisely, we use all possible combinations of $s_{wij} = \frac{1}{d_{ij}}\frac{1}{\|\mathbf{J}_{s_{ij}^*/\mathbf{p}^*}\|}$. The interaction matrix that links the variations of the new feature $s_{wij} = \frac{1}{\|\mathbf{J}_{s_{ij}^*/\mathbf{p}^*}\|}\frac{1}{d_{ij}}$ to the translational velocities is then obtained by:

$$\mathbf{L}_{s_{wij}\boldsymbol{v}} = \frac{1}{\|\,\mathbf{J}_{s_{ij}^*/\mathbf{p}^*}\,\|\;d_{ij}^3}\left((-\frac{1}{\|\,\mathbf{P_j}\,\|}+\frac{\mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j}}{\|\,\mathbf{P_i}\,\|})\mathbf{P}_{s_i}^{\top}+(-\frac{1}{\|\,\mathbf{P_i}\,\|}+\frac{\mathbf{P}_{s_i}^{\top}\mathbf{P}_{s_j}}{\|\,\mathbf{P_j}\,\|})\mathbf{P}_{s_j}^{\top}\right) \tag{52}$$

3.3. Conclusion

To summarize, the new feature vector $\mathbf{s_n}$ we propose is composed of two parts. The first part $\mathbf{s_t}$ is devoted to control the translational motions. It is composed of the set of s_{wij} computed from all combinations of two projected points on the sphere. The second part is devoted to control the rotational motions. It is defined by the rotation vector $\theta\mathbf{u}$ described in paragraph 3.1.3. The interaction matrix related to $\mathbf{s_n}$ can be expressed as follows:

$$\mathbf{L_{s_n}} = \begin{bmatrix} \mathbf{L}_{n\boldsymbol{v}} & 0 \\ -\mathbf{L}_{\omega}(\theta,\mathbf{u})\boldsymbol{\zeta}_{n\boldsymbol{v}} & \mathbf{L}_{\omega}(\theta,\mathbf{u}) \end{bmatrix} \tag{53}$$

where $\mathbf{L}_{n\boldsymbol{v}}$ is the interaction matrix that links the variation of $\mathbf{s_t}$ and the translational motions (see (52)) and where $\mathbf{L}_{\omega}(\theta,\mathbf{u})$ and $\boldsymbol{\zeta}_{n\boldsymbol{v}}$ are given respectively by (26) and (33). This leads to the following visual servoing control law:

$$\begin{cases} \boldsymbol{v} = -\lambda \mathbf{L}^{+}_{n\boldsymbol{v}} \left(\mathbf{s}_t - \mathbf{s}^{*}_t\right) \\ \boldsymbol{\omega} = -\boldsymbol{\zeta}_{n\boldsymbol{v}}\boldsymbol{v} + \lambda\theta\mathbf{u} \end{cases} \tag{54}$$

since $\mathbf{L}^{-1}_{\omega}(\theta\mathbf{u})\,\theta\mathbf{u} = -\theta\mathbf{u}$.

4. Experimental results

In the following, simulation results using a conventional camera projection model are firstly presented. Then experimental results with real images using a fisheye camera are described.

4.0.1. Simulation results using conventional camera model

In the following simulations, a conventional camera model with focal scaling factors $F_x = F_y = 800\ pixels$ and coordinates of the principal point $u_x = u_y = 400\ pixels$ is used to generate the image point coordinates. Five different invariants to rotations $\cos\alpha_{ij}$, d_{ij}, s_{Δ}, $1/d_{ij}$ and s_{wij} are tested to check which ones allow a control law behavior close to a linear system and more robustness to noise. The invariant computed from the polynomial I is excluded from the test since it has already been shown in [18] that using s_{Δ} a better behavior can be obtained.

For all the following simulations, the scalar control gain is set up at 0.1. In order to test the robustness to noise, a Gaussian white noise with standard deviation equal to 1 pixel is added to the coordinates of each point in the

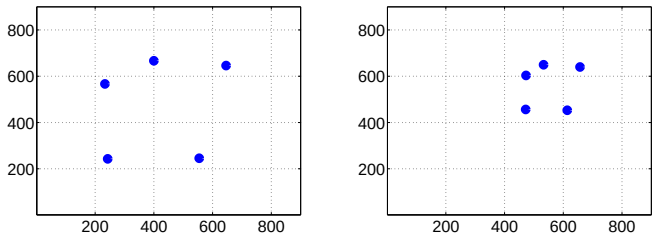


Figure 5: Points image for simulation 1: a) points image for the pose 1, b) points image for the pose 2

image, during servoing. Two camera poses are considered: pose 1 is defined by the 3D point coordinates (55) and pose 2 is defined by the 3D point coordinates (56). Only translational motions are considered in this part.

$$\mathbf{X_4} = \begin{bmatrix} -0.2 & -0.2 & 0.2 & 0 & 0.4 \\ -0.2 & 0.2 & -0.2 & 0.4 & 0.4 \\ 1.01 & 0.95 & 1.03 & 1.2 & 1.3 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{55}$$

$$\mathbf{X_5} = \begin{bmatrix} 0.2 & 0.2 & 0.6 & 0.4 & 0.8 \\ 0.15 & 0.55 & 0.15 & 0.75 & 0.75 \\ 2.21 & 2.15 & 2.23 & 2.4 & 2.5 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{56}$$

The translational motion between the pose 1 and the pose 2 is given by $\mathbf{t} = [0.4\ \ 0.35\ \ 1.2] meter$. The images of the points for the pose 1 and 2 are shown respectively in Figs 5.a and 5.b

In order to test the linearizing behavior of each invariant to rotation, visual servoing is performed from pose 1 to pose 2 and then from pose 2 to pose 1. If a visual feature allows a linear mapping between the image and task space, the behavior of the velocities when the camera moves from pose 1 to pose 2 or moves from pose 2 to pose 1 has to be symmetrical.

The behavior of the error on the features when the camera moves from pose 1 to pose 2 and from pose 2 to pose 1 are shown respectively in column 1 and in column 2 in Fig. 6. From this figure, it can be seen that all errors on features decrease in similar and satisfactory way independently of the way the camera moves. However, from the same figure, it can be seen that the

effect of the noise on the entries of the features error vector is not uniform when s_Δ and $1/d_{ij}$ are used (refer to Figs 6.i, 6.j, 6.m and 6.n).

The behavior of the translational velocities using each feature is shown in column 3 and column 4 of the same figure. From these plots, the decrease of the velocities using $\cos\alpha_{ij}$ and d_{ij} is non-symmetrical and leads to different speeds of convergence: the convergence is much faster if the camera has to move from the pose 2 to the pose 1 than from 1 to 2 (compare Figs 6.c, 6.d and 6.g , 6.h). Note also that the velocities at the first iteration are almost 10 times bigger in Fig. 6.d than in Fig. 6.c for instance, which is far from a symmetrical behavior. This shows that the interaction matrices of $\cos\alpha_{ij}$ and d_{ij} are very different for the camera positions 1 and 2. On the contrary, it can be seen that the features s_Δ, $1/d_{ij}$ and s_{wij} allow obtaining an almost symmetrical decrease of the velocities and then the same speed of convergence if the camera moves from the pose 1 to the pose 2 or from pose 2 to pose 1 (compare 6.k and 6.l, 6.o and 6.p , 6.s and 6.t).

Concerning sensitivity to noise, it can be noticed that the velocities are more noisy when the camera has to move from the poses 2 to 1 than from the pose 1 to the pose 2. This can be explained by the fact that the size of the 'object' in the image corresponding to the pose 1 is much bigger than the image of the object corresponding to the pose 2. Finally, from the velocity plots, it can be seen that the velocities obtained using the feature s_{wij} are by far the less sensitive to noise (refer to Figs 6.s and 6.t)

In the second simulation, a pure rotation defined by vector $\theta\mathbf{u} = [-23.75 \; -19.7900 \; -79.1750]$ *degrees* is considered. The desired pose is defined by (56). The initial image of the points is shown in Fig. 7.e (dots in blue). The results obtained are shown in Fig. 7. From this figure, it can be seen that the camera performs exactly the required rotational motion without any undesired translational motion (refer to Figs 7.c and 7.d). Indeed, since the considered translational motion is null, the translational velocities computed using the invariants to rotations are null (the small variations of the translational velocities are due to noise). The trajectory of the points in the image corresponding to the performed motion in $3D$ is shown on Fig. 7.e. On the other hand, Figure 8 shows the results obtained using the point coordinates as features. Since the control is made using the point coordinates, the trajectories of the points in the image are close to straight lines (refer to Fig. 8.d) and the decrease of the error of the features is also satisfactory (refer to Fig. 8.a). On the contrary, due to coupling between the translational and rotational velocities, the corresponding velocities applied to the camera

are far from satisfactory (refer to Figs 8.b and 8.c). Indeed, a strong undesired translational motion is generated while the motion to perform is a pure rotation.

4.0.2. Experimental results with real images obtained using fisheye camera

The experiments were performed using an Afma Cartesian robotics platform and Visp software [13]. Furthermore, a camera with a fish-eye lens has been used. The unified projection with a distortion parameter $\xi = 1.50$, central point coordinates $(u_x, v_x) = (316.90,\ 244.50)$ $pixels$, and focal scaling factors given by $(F_x, F_y) = (655.69,\ 658.38)$ $pixels$ was used as projection model of the camera. For all the following experiments, the scalar control gain is set up at 0.1 and the interaction matrix computed for the current pose of the camera is used at each iteration.

In the first experiment, two camera poses 1 and 2 separated by the translation $\mathbf{t} = [-0.4,\ 0,\ 0.5]$ $meter$ are considered. The images corresponding to the poses 1 and 2 of the camera are shown in Figs 9.a and 9.b. Four different invariants to rotations $\cos\alpha_{ij}$, d_{ij}, $1/d_{ij}$ and s_{wij} were tested. The results obtained are shown in Figure 11. They confirm those obtained previously using conventional perspective camera model. Indeed, from this same figure, it can be seen that the effect of the noise on the entries of the features error vector is not uniform when $1/d_{ij}$ is used (refer to Figs 11.i and 11.j). The behavior of the translational velocities using each feature is shown in column 3 and column 4 of the same figure. From these plots, it can be seen that the features $1/d_{ij}$ and s_{wij} allow obtaining an almost symmetrical decrease of the velocities and then the same speed of convergence if the camera moves from the pose 1 to the pose 2 or from pose 2 to pose 1 (compare Fig 11.k to Fig 11.l and Fig 11.o to Fig 11.p). On the contrary, the decrease of the velocities using $\cos\alpha_{ij}$ and d_{ij} is non-symmetrical and leads to different speeds of convergence: the convergence is much faster if the camera moves from pose 2 to pose 1 than from pose 1 to pose 2 (compare Figs 11.c, 11.d and 11.g , 11.h). Note also that the velocities at the first iteration are almost 10 times bigger in Fig. 11.d than in Fig. 11.c for instance, which is far from a symmetrical behavior. Finally, from the velocity plots, it can be seen that the velocities obtained using the feature s_{wij} are by far the less sensitive to noise (refer to Figs 11.o and 11.p)

The second experiment corresponded to a 'pure' rotation given by vector $\theta\mathbf{u} = [-15.47,\ -7.47,\ -61.83]$ $degrees$. The image points corresponding to the initial and to the desired poses are shown respectively in Figures 10.a and

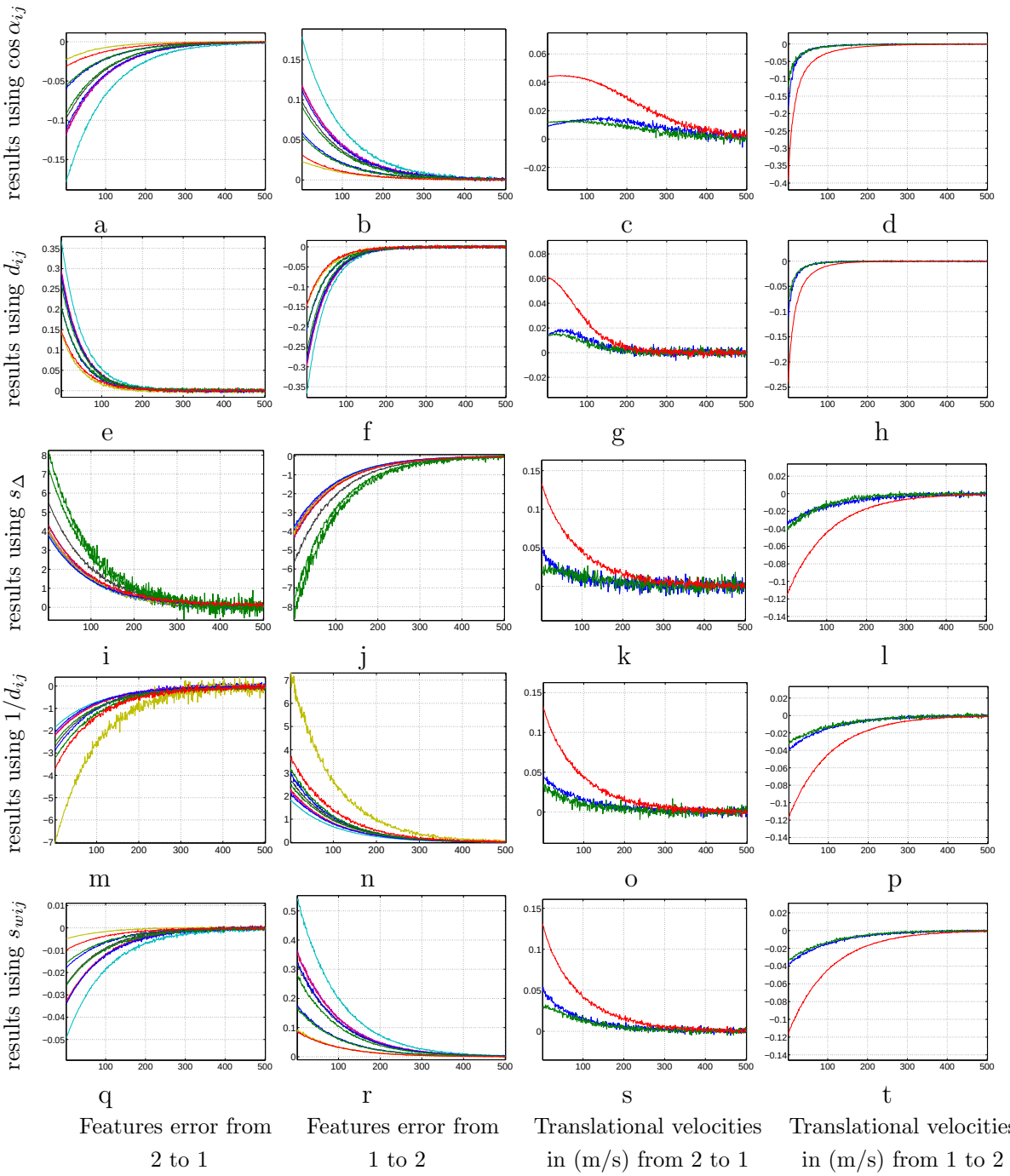


Figure 6: Simulation results when only translational motions are controlled

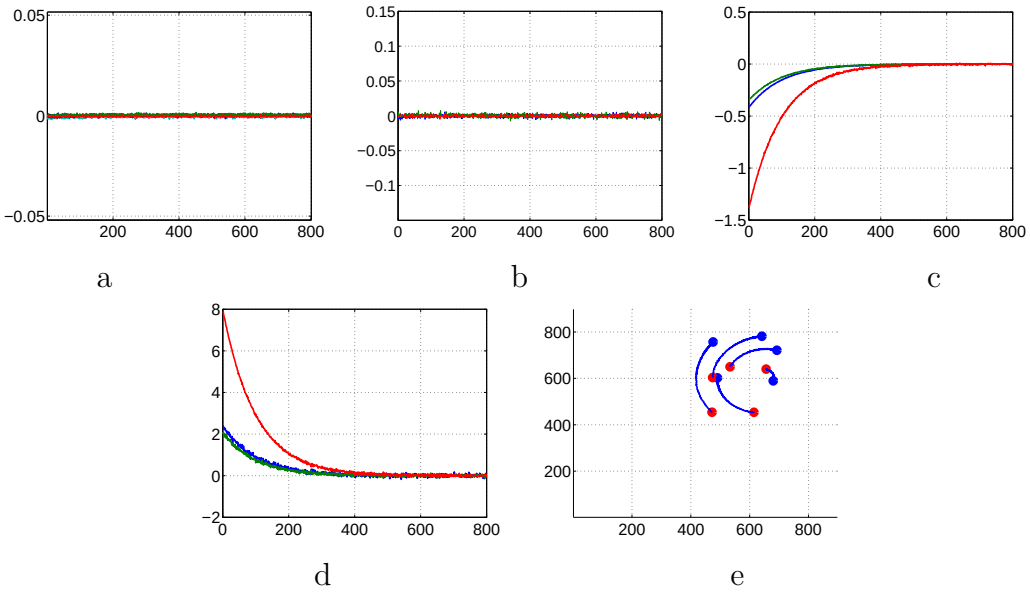


Figure 7: Simulation results for pure rotational motion: a) errors on features used to control translations, b) translational velocities in m/s, c) errors on features used to control rotations, d) rotational velocities in $degrees/s$ e) trajectory of the points in the image

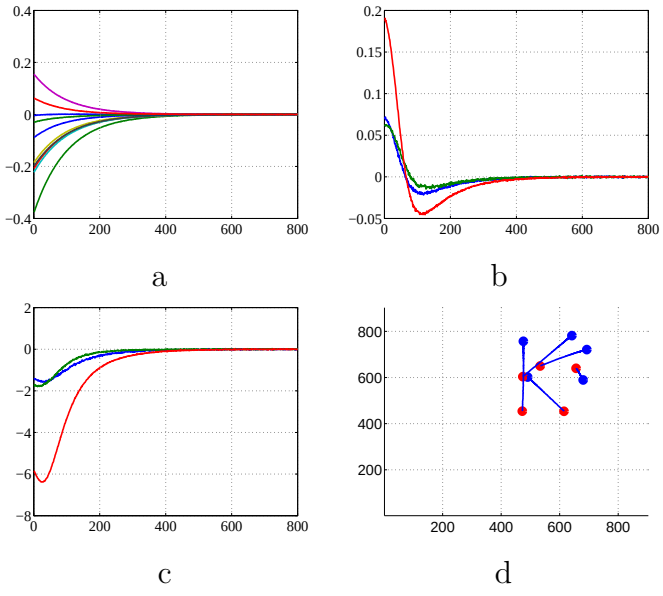
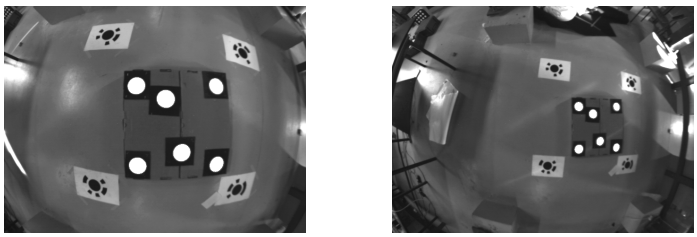


Figure 8: Simulation results for pure rotational motion using point coordinates as features: a) error on features, b) translational velocities in m/s, c) rotational velocities in $degrees/s$, d) trajectory of the points in the image

9.a. The results obtained using features vector $\mathbf{s_n}$ are shown in Figure 12. As for the results obtained using conventional camera model, it can be seen that the camera performs a pure rotation since the translational velocities are null.

In the final experiment, a camera displacement involving rotational and translational motions was considered. The initial and desired images are given respectively in Figs 10.b and 9.a. The results obtained are shown in Fig. 13. From the plots of the velocities and the feature errors, it can be seen once again that an adequate decrease of the feature errors in the image space as well as in the task space is obtained.



a b

Figure 9: Experimental results when only translational motions are controlled: a) image points for the pose 1, b) image points for the pose 2

a b

Figure 10: Initial image: a) case where only a pure rotation is considered, b) case of displacement involving translational and rotational motions

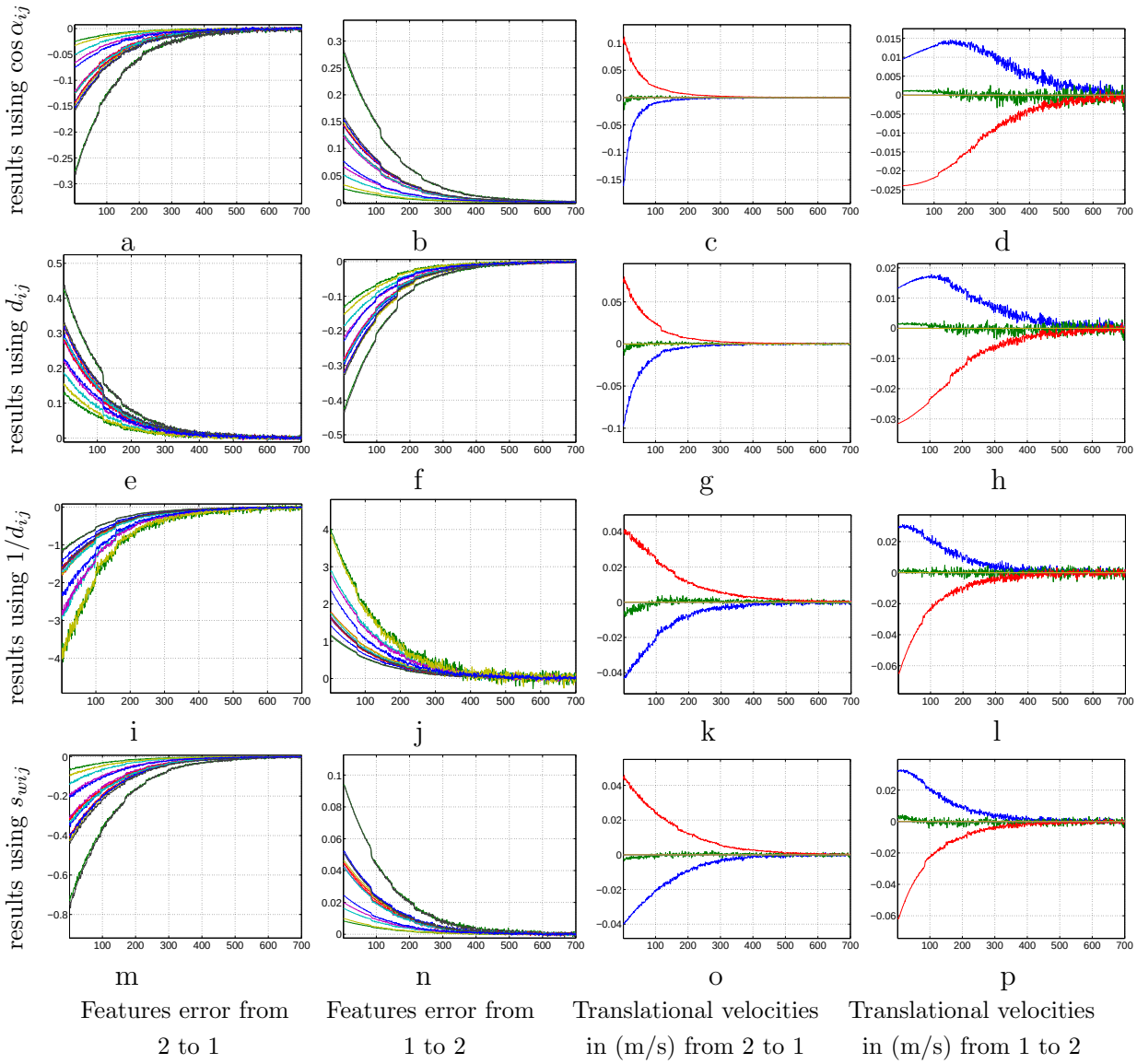


Figure 11: Experimental results when only translational motions are controlled.

5. Conclusions

In this paper, we have proposed new visual features for image-based visual servoing from a set of matched points. We have firstly proposed a rotation vector computed from all the projected points and providing direct link with

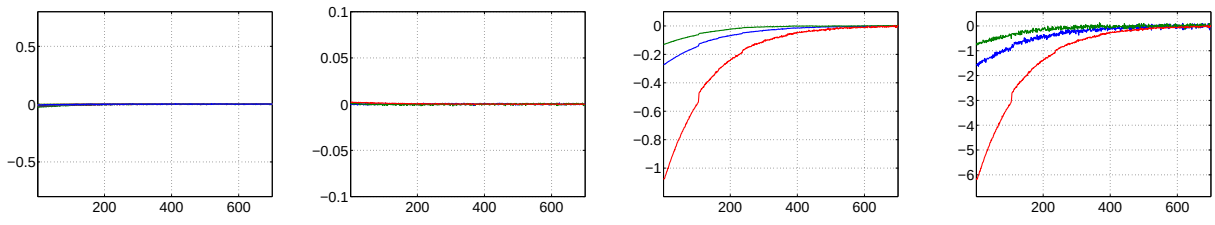


Figure 12: Result for pure rotational motion $\theta u = [-15.47,\ -7.47,\ -61.83]$ *degrees*: a) errors on features to control translations, b) translational velocities in m/s, c) errors on features to control rotations, d) rotational velocities in *degrees*$/s$

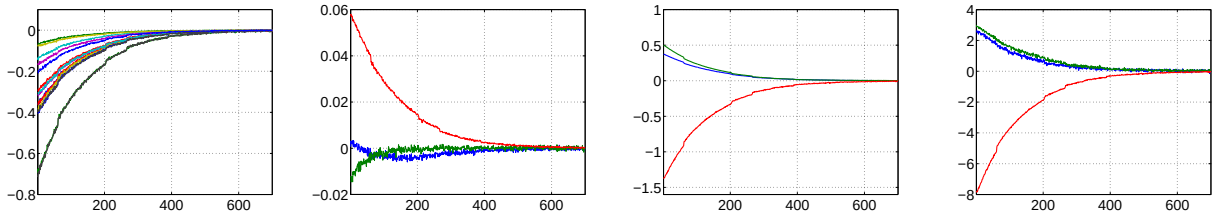


Figure 13: Result for generic motion: a) errors on features used to control translations, b) translational velocities in m/s, c) errors on features used to control rotations, d) rotational velocities in *degrees*$/s$

the rotational velocities. Then, a set of invariants to rotation has been proposed and used to control the translational velocities. The propagation of noise from the image points coordinates to the features space has been taken into account to improve the robustness of the control law with respect to measurement noise. The theoretical results have been validated with several simulation and experimental results.

Appendix A: proof of the Lemma 1

If only a rotational motion is considered between the current frame and the desired frame, we have:

$$\mathbf{P}^*_{v_1} = \sum_{i=1}^{N} a_{1i}\mathbf{P}^*_{s_i} = \sum_{i=1}^{N} a_{1i}\ {}^{c*}\mathbf{R}_c\mathbf{P}_{s_i} = {}^{c*}\mathbf{R}_c\,\mathbf{P}_{v_1} \tag{57}$$

$$\mathbf{P}^*_{v_2} = \sum_{i=1}^{N} a_{2i}\mathbf{P}^*_{s_i} = \sum_{i=1}^{N} a_{2i}\ {}^{c*}\mathbf{R}_c\mathbf{P}_{s_i} = {}^{c*}\mathbf{R}_c\,\mathbf{P}_{v_2} \tag{58}$$

By combining these equations with (13), we obtain:

$$\mathbf{v}^*_{n1} = \frac{{}^{c*}\mathbf{R}_c\,\mathbf{P}_{v_1}}{\sqrt{\mathbf{P}^\top_{v_1}\,{}^{c*}\mathbf{R}^\top_c\,{}^{c*}\mathbf{R}_c\,\mathbf{P}_{v_1}}} \tag{59}$$

from which, we obtain:

$$\mathbf{v}^*_{n1} = {}^{c*}\mathbf{R}_c\,\mathbf{v}_{n1} \tag{60}$$

Identically, it is possible to prove that:

$$\mathbf{v}^*_{n2} = {}^{c*}\mathbf{R}_c\,\mathbf{v}_{n2} \tag{61}$$

Combining (60), (61) and (15) it yields:

$$\mathbf{v}^*_{n3} = {}^{c*}\mathbf{R}_c\,\mathbf{v}_{n3} \tag{62}$$

Therefore, by combining (60), (61) and (62), one obtains:

$${}^{c*}\mathbf{R}_{n*} = {}^{c*}\mathbf{R}_c\,{}^{c}\mathbf{R}_n \tag{63}$$

Finally, by multiplying both sides of the last equation by ${}^{c}\mathbf{R}^\top_n$, we obtain

$${}^{c*}\mathbf{R}_{n*}\,{}^{c}\mathbf{R}^\top_n = {}^{c*}\mathbf{R}_c\,{}^{c}\mathbf{R}_n\,{}^{c}\mathbf{R}^\top_n \tag{64}$$

since ${}^{c}\mathbf{R}_n\,{}^{c}\mathbf{R}^\top_n = \mathbf{I}_3$, therefore:

$${}^{c*}\mathbf{R}_{n*}\,{}^{c}\mathbf{R}^\top_n = {}^{c*}\mathbf{R}_c \tag{65}$$

Appendix B: computation of the tangent vector $\boldsymbol{\zeta_n}$

Let us first determine $\dot{\mathbf{v}}_{\mathbf{n1}}$, $\dot{\mathbf{v}}_{\mathbf{n2}}$ and $\dot{\mathbf{v}}_{\mathbf{n3}}$. From (13), it can be obtained:

$$\dot{\mathbf{v}}_{\mathbf{n1}} = \frac{\Gamma_{\mathbf{v_{n1}}}}{\| \mathbf{P}_{v_1} \|} \dot{\mathbf{P}}_{v_1} \tag{66}$$

where $\Gamma_{\mathbf{v_{ni}}} = \mathbf{I}_3 - \mathbf{v_{ni}}\mathbf{v_{ni}}^\top$. Since $\mathbf{v_{n2}}^\top \Gamma_{\mathbf{v_{n1}}} = \mathbf{v_{n2}}^\top$ and $\mathbf{v_{n3}}^\top \Gamma_{\mathbf{v_{n1}}} = \mathbf{v_{n3}}^\top$, we have

$$\mathbf{v_{n2}}^\top \dot{\mathbf{v}}_{\mathbf{n1}} = \mathbf{v_{n2}}^\top \frac{\dot{\mathbf{P}}_{v_1}}{\| \mathbf{P}_{v_1} \|} \tag{67}$$

and

$$-\mathbf{v_{n3}}^\top \dot{\mathbf{v}}_{\mathbf{n1}} = -\mathbf{v_{n3}}^\top \frac{\dot{\mathbf{P}}_{v_1}}{\| \mathbf{P}_{v_1} \|} \tag{68}$$

By taking the derivative of (14), we obtain:

$$\dot{\mathbf{v}}_{\mathbf{n2}} = \frac{\Gamma_{\mathbf{v_{n2}}}\Gamma_{\mathbf{v_{n1}}}}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top \mathbf{v_{1n}})\mathbf{v_{1n}} \|} \dot{\mathbf{P}}_{v_2} - \frac{\Gamma_{\mathbf{v_{n2}}}(\mathbf{P}_{v_2}^\top \mathbf{v_{n1}}\mathbf{I_3} + \mathbf{v_{n1}}\mathbf{P}_{v_2}^\top)\Gamma_{\mathbf{v_{n1}}}}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top \mathbf{v_{1n}})\mathbf{v_{1n}} \| \, \| \mathbf{P}_{v_1} \|} \dot{\mathbf{P}}_{v_1} \tag{69}$$

Furthermore, the time variation of $\mathbf{v_{n3}}$ can be computed using:

$$\dot{\mathbf{v}}_{\mathbf{n3}} = [\mathbf{v_{n1}}]_\times \dot{\mathbf{v}}_{\mathbf{n2}} - [\mathbf{v_{n2}}]_\times \dot{\mathbf{v}}_{\mathbf{n1}} \tag{70}$$

Since $\mathbf{v_{n2}}^\top [\mathbf{v_{n2}}]_\times$ is null, we have:

$$-\mathbf{v_{n2}}^\top \dot{\mathbf{v}}_{\mathbf{n3}} = -\mathbf{v_{n2}}^\top [\mathbf{v_{n1}}]_\times \dot{\mathbf{v}}_{\mathbf{n2}} = \mathbf{v_{n3}}^\top \dot{\mathbf{v}}_{\mathbf{n2}} \tag{71}$$

By plugging (69) into (71) and taking into account that $\mathbf{v_{n3}}^\top \Gamma_{\mathbf{v_{n1}}} = \mathbf{v_{n3}}^\top \Gamma_{\mathbf{v_{n2}}} = \mathbf{v_{n3}}^\top$ and $\mathbf{v_{n3}}^\top \mathbf{v_{n1}} = 0$, we obtain:

$$-\mathbf{v_{n2}}^\top \dot{\mathbf{v}}_{\mathbf{n3}} = \frac{\mathbf{v_{n3}}^\top \dot{\mathbf{P}}_{v_2}}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top \mathbf{v_{1n}})\mathbf{v_{1n}} \|} - \frac{\mathbf{v_{n3}}^\top (\mathbf{P}_{v_2}^\top \mathbf{v_{n1}}) \dot{\mathbf{P}}_{v_1}}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top \mathbf{v_{1n}})\mathbf{v_{1n}} \| \, \| \mathbf{P}_{v_1} \|} \tag{72}$$

Furthermore, after plugging (67), (68) and (72) into (30), we obtain:

$$\begin{aligned} \boldsymbol{\zeta} = & \left(-\mathbf{v_{n2}}\mathbf{v_{n3}}^\top + \mathbf{v_{n3}}\mathbf{v_{n2}}^\top - \frac{\mathbf{v_{n1}}\mathbf{v_{n3}}^\top (\mathbf{P}_{v_2}^\top \mathbf{v_{n1}})}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top \mathbf{v_{1n}})\mathbf{v_{1n}} \|} \right) \frac{\dot{\mathbf{P}}_{v_1}}{\| \mathbf{P}_{v_1} \|} \\ & + \frac{\mathbf{v_{n1}}\mathbf{v_{n3}}^\top}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top \mathbf{v_{1n}})\mathbf{v_{1n}} \|} \dot{\mathbf{P}}_{v_2} \end{aligned} \tag{73}$$

Remind that the interaction matrix related to points on the unit sphere [10] is given by:

$$\mathbf{L_{P_s}} = \begin{bmatrix} \frac{-\mathbf{I_3}+\mathbf{P}_s\mathbf{P}_s^\top}{\|\mathbf{P}\|} & [\mathbf{P}_s]_\times \end{bmatrix} \tag{74}$$

The interaction matrices related to $\mathbf{P}_{v_1}$ and $\mathbf{P}_{v_2}$ can be obtained by taking the derivative of (12) and combining the result with (74):

$$L_{\mathbf{P}_{v_1}} = \begin{bmatrix} \sum_{i=1}^{N} \frac{a_{1i}}{\|\mathbf{P_i}\|}(-\mathbf{I}_3 + \mathbf{P}_{s_i}\mathbf{P}_{s_i}^\top) & [\mathbf{P}_{v_1}]_\times \end{bmatrix} \tag{75}$$

$$L_{\mathbf{P}_{v_2}} = \begin{bmatrix} \sum_{i=1}^{N} \frac{a_{2i}}{\|\mathbf{P_i}\|}(-\mathbf{I}_3 + \mathbf{P}_{s_i}\mathbf{P}_{s_i}^\top) & [\mathbf{P}_{v_2}]_\times \end{bmatrix} \tag{76}$$

Let us consider that $\boldsymbol{\zeta}_n = \boldsymbol{\zeta}_{n\boldsymbol{v}}\boldsymbol{v} + \boldsymbol{\zeta}_{n\boldsymbol{\omega}}\boldsymbol{\omega}$ and show that the direct link between $\boldsymbol{\zeta}$ and the rotational velocities obtained in [19] is still valid for the new features (i.e. $\boldsymbol{\zeta}_{n\boldsymbol{\omega}} = -\mathbf{I}_3$). Combining (75), (76) and (73), it can be obtained:

$$\begin{aligned} \boldsymbol{\zeta}_{n\boldsymbol{\omega}} = &\left(-\mathbf{v_{n2}}\mathbf{v_{n3}}^\top + \mathbf{v_{n3}}\mathbf{v_{n2}}^\top - \frac{\mathbf{v_{n1}}\mathbf{v_{n3}}^\top(\mathbf{P}_{v_2}^\top\mathbf{v_{n1}})}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top\mathbf{v_{n1}})\mathbf{v_{n1}} \|} \right) \frac{[\mathbf{P}_{v_1}]_\times}{\| \mathbf{P}_{v_1} \|} \\ &+ \frac{\mathbf{v_{n1}}\mathbf{v_{n3}}^\top}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top\mathbf{v_{n1}})\mathbf{v_{n1}} \|}[\mathbf{P}_{v_2}]_\times \end{aligned} \tag{77}$$

We have $\frac{[\mathbf{P}_{v_1}]_\times}{\|\mathbf{P}_{v_1}\|} = [\mathbf{v_{n1}}]_\times$, which leads to:

$$\boldsymbol{\zeta}_{n\boldsymbol{\omega}} = -\mathbf{v_{n2}}\mathbf{v_{n3}}^\top[\mathbf{v_{n1}}]_\times + \mathbf{v_{n3}}\mathbf{v_{n2}}^\top[\mathbf{v_{n1}}]_\times + \mathbf{v_{n1}}\mathbf{v_{n3}}^\top \left(\frac{[\mathbf{P}_{v_2}]_\times - (\mathbf{P}_{v_2}^\top\mathbf{v_{n1}})[\mathbf{v_{n1}}]_\times}{\| \mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top\mathbf{v_{n1}})\mathbf{v_{n1}} \|} \right) \tag{78}$$

Identically, it can be noticed that $\frac{[\mathbf{P}_{v_2}]_\times - (\mathbf{P}_{v_2}^\top\mathbf{v_{n1}})[\mathbf{v_{n1}}]_\times}{\|\mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top\mathbf{v_{n1}})\mathbf{v_{n1}}\|} = [\mathbf{v_{n2}}]_\times$. Then (78) can be written as:

$$\boldsymbol{\zeta}_{n\boldsymbol{\omega}} = -\mathbf{v_{n2}}\mathbf{v_{n3}}^\top[\mathbf{v_{n1}}]_\times + \mathbf{v_{n3}}\mathbf{v_{n2}}^\top[\mathbf{v_{n1}}]_\times + \mathbf{v_{n1}}\mathbf{v_{n3}}^\top[\mathbf{v_{n2}}]_\times \tag{79}$$

Since the triplet ($\mathbf{v_{n1}}$, $\mathbf{v_{n2}}$, $\mathbf{v_{n3}}$) is a direct and ortho-normal basis, we have:

$$-\mathbf{v_{n3}}^\top[\mathbf{v_{n1}}]_\times = (\mathbf{v_{n1}} \times \mathbf{v_{n3}})^\top = -\mathbf{v_{n2}}^\top, \tag{80}$$

$$\mathbf{v_{n2}}^\top[\mathbf{v_{n1}}]_\times = -(\mathbf{v_{n1}} \times \mathbf{v_{n2}})^\top = -\mathbf{v_{n3}}^\top, \tag{81}$$

$$\mathbf{v_{n3}}^\top [\mathbf{v_{n2}}]_\times = -(\mathbf{v_{n2}} \times \mathbf{v_{n3}})^\top = -\mathbf{v_{n1}}^\top \tag{82}$$

By plugging the last three equations in (79), we obtain:

$$\boldsymbol{\zeta}_{n\boldsymbol{\omega}} = -\mathbf{v_{n2}}\mathbf{v_{n2}}^\top - \mathbf{v_{n3}}\mathbf{v_{n3}}^\top - \mathbf{v_{n1}}\mathbf{v_{n1}}^\top = -{}^c\mathbf{R}_n\, {}^c\mathbf{R}_n^\top = -\mathbf{I_3} \tag{83}$$

Furthermore, we have:

$$\boldsymbol{\zeta}_{n\boldsymbol{v}} = \delta_{13}\mathbf{v_{n1}}\mathbf{v_{n3}}^\top + \delta_{23}(\mathbf{v_{n3}}\mathbf{v_{n2}}^\top - \mathbf{v_{n2}}\mathbf{v_{n3}}^\top) \tag{84}$$

where

$$\begin{cases} \delta_{13} = \sum_{i=1}^{N} \frac{(a_{1i}\mathbf{P}_{v_2}^\top \mathbf{v_{n1}} - a_{2i}\|\mathbf{P}_{v_1}\|)(-\mathbf{I}_3 + \mathbf{P}_{s_i}\mathbf{P}_{s_i}^\top)}{\|\mathbf{P}_{v_2} - (\mathbf{P}_{v_2}^\top \mathbf{v_{1n}})\mathbf{v_{1n}}\|\, \|\mathbf{P}_{v_1}\|\, \|\mathbf{P_i}\|}. \\ \delta_{23} = \sum_{i=1}^{N} \frac{a_{1i}}{\|\mathbf{P}_{v_1}\| \|\mathbf{P_i}\|}(-\mathbf{I}_3 + \mathbf{P}_{s_i}\mathbf{P}_{s_i}^\top) \end{cases}$$

Equation (84) can be simplified by noticing that $\mathbf{v_{n3}}\mathbf{v_{n2}}^\top - \mathbf{v_{n2}}\mathbf{v_{n3}}^\top = [\mathbf{v_{n2}} \times \mathbf{v_{n3}}]_\times = [\mathbf{v_{n1}}]_\times$. Therefore, we have:

$$\boldsymbol{\zeta}_{n\boldsymbol{v}} = \delta_{13}\mathbf{v_{n1}}\mathbf{v_{n3}}^\top + \delta_{23}[\mathbf{v_{n1}}]_\times \tag{85}$$

Acknowledgment

The authors would like to thank the support of project Morfeu–PTDC/EEA-CRO/108348/2008 funded by the Portuguese Science Foundation (FCT) and the support from PHC PESSOA 2011 (25116ND).

References

[1] S. Baker and S. Nayar. A theory of catadioptric image formation. *Int. Journal of Computer Vision*, 35(2):175–196, November 1999.

[2] J.P. Barreto and H. Araujo. Issues on the geometry of central catadioptric image formation. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–422–II–427 vol.2, 2001.

[3] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In D. Kriegman, G . Hager, and A.S. Morse, editors, *The Confluence of Vision and Control*, pages 66–78. LNCIS Series, No 237, Springer-Verlag, 1998.

[4] F. Chaumette and S. Hutchinson. Visual servo control, Part I: Basic approaches. In IEEE Robotics and Automation Magazine, pages 82–90, volume 13(4), December, 2006.

[5] J. Courbon, Y. Mezouar, L. Eck, and P. Martinet. A generic fisheye camera model for robotic applications. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'07*, pages 1683–1688, San Diego, CA, USA, 2007.

[6] J. Courbon, Y. Mezouar, and P. Martinet. Evaluation of the unified model on the sphere for fisheye cameras in robotic applications. *Advanced Robotics*, 6(8):947–967, 2012.

[7] N.J. Cowan and Dong Eui Chang. Geometric visual servoing. *IEEE Trans. on Robotics*, 21(6):1128 – 1138, Dec. 2005.

[8] J. Feddema and O. Mitchell. Vision-guided servoing with feature-based trajectory generation IEEE Transaction on Robotics and Automation, vol. 5, no. 5, pp. 691-700, Oct. 1989.

[9] C. Geyer and K. Daniilidis. A Unifying Theory for Central Panoramic Systems and Practical Implications. In Computer Vision- ECCV 2000 (pp. 445-461). Springer Berlin Heidelberg.

[10] T. Hamel and R. Mahony. Visual servoing of an under-actuated dynamic rigid body system: an image-based approach. *IEEE Trans. on Robotics and Automation*, 18(2):187–198, April 2002.

[11] M. Iwatsuki and N. Okiyama, A new formulation of visual servoing based on cylindrical coordinates system with shiftable origin. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'02* Lausanne, Switzerland, Oct. 2002, pp. 354–359.

[12] E. Malis. *Contributions à la modélisation et à la commande en asservissement visuel*. PhD thesis, Université de Rennes 1, November 1998.

[13] E. Marchand, F. Spindler, and F. Chaumette. Visp for visual servoing: a generic software platform with a wide class of robot control skills. *IEEE Robotics and Automation Magazine*, 12(4):40–52, December 2005.

[14] C. Mei and P. Rives. Single view point omnidirectional camera calibration from planar grids. In *IEEE Int. Conf. on Robotics and Automation*, pages 3945–3950, April 2007.

[15] P. Rives and J. Azinheira. Linear structures following by an airship using vanishing points and horizon line in a visual servoing scheme, In *IEEE Int. Conf. on Robotics and Automation*, New Orleans, LA,Apr. 2004, pp. 255–260.

[16] T. Svoboda and T. Pajdla. Epipolar geometry for central catadioptric cameras. *Int. Journal on Computer Vision*, 49(1):23–37, August 2002.

[17] O. Tahri and F. Chaumette. Point-based and region-based image moments for visual servoing of planar objects. *IEEE Trans. on Robotics*, 21(6):1116–1127, December 2005.

[18] O. Tahri, Y. Mezouar, F. Chaumette, and P. Corke. Decoupled image-based visual servoing for cameras obeying the unified projection model. *IEEE Trans. on Robotics*, 26(4):684 – 697, August 2010.

[19] R. Tatsambon Fomena, O. Tahri, and F. Chaumette. Distance-based and orientation-based visual servoing from three points. *IEEE Trans. on Robotics*, 27(2):256–267, April 2011.

[20] L. Weiss, A. C. Sanderson, and C. P. Neuman. Dynamic sensor-based control of robots with visual feedback *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 5, pp. 404–417, Oct. 1987.