Combining Pixel and Depth Information in Image-Based Visual Servoing

Enric Cervera, Philippe Martinet

Combining Pixel and Depth Information in Image-Based Visual Servoing

E. Cervera

Robotic Intelligence Lab.
Jaume-I University
12071 Castelló, Spain

P. Martinet

LASMEA
Blaise Pascal University of Clermont-Ferrand
63177 Aubière - Cedex, France

Abstract

Image-based visual servoing has been found generally satisfactory and robust in the presence of camera and hand-eye calibration errors. However, in some cases, singularities and local minima may arise. We propose a modification of the feature vector which alleviates this problem without need of introducing further information, i.e., only pixel coordinates and depth estimations are used. Using the task-function approach, we demonstrate the relationship between the velocity screw of the camera and the current and desired poses of the object in the camera frame. Experimental results on a real robotic platform illustrate the presented approaches.

1 Introduction

This work aims at improving the behavior of image-based visual servoing. As pointed out in [1, 5], in some cases, convergence and stability problems may occur. We propose a new feature vector based on the same information used by the classic approach, i.e. image point coordinates and depth estimation.

Using the *task-function* framework [8, 3], the velocity screw of the camera can be obtained. We will demonstrate that, for small changes of orientation, the computed screw depends only on the current and desired poses of the object in the camera frame, and consequently there are no singularities nor local minima in the trajectory of the camera.

Later, we study the case of directly using the 3D coordinates of object points, which turns to be a particular case of the previous one. Nonetheless, stronger results for the velocity screw are obtained for some particular geometric configurations of the target object. Though 3D visual features have been studied before (see e.g. [7]), we propose a new approach and present for the first time analytical results for the velocity screw based on a set of feature points.

The remainder of this section briefly reviews the classic image-based visual servoing approach and presents some definitions. In Section 2 the new feature vector is presented, together with the development of the interaction matrix and the velocity screw of the camera. Section 3 discusses the case of using the 3D coordinates of the points of the object. Experimental results are described in Section 4. Finally, Section 5 draws some conclusions and open issues.

1.1 Image-based visual servoing

In image-based visual servoing, a feature vector $\mathbf{s}$ has to reach a desired value $\mathbf{s}^*$. Usually, $\mathbf{s}$ is composed of the image coordinates of several points of the object.

The key issue in visual servoing is to find the relationship between the derivative of the feature vector and the velocity screw of the camera $\mathbf{v} = (\boldsymbol{\nu}^t, \boldsymbol{\omega}^t)^t$:

$$\dot{\mathbf{s}} = \mathbf{L}_s \mathbf{v} \tag{1}$$

where $\mathbf{L}_s$ is the *interaction matrix* or *Jacobian matrix*.

If the feature vector $\mathbf{s}_i$ is composed of the image coordinates (u_i, v_i) of a single point $\mathbf{p}_i$, then, as shown in [6], the interaction matrix is the presented in Equation 2 (shown at the top of next page) where Z_i is the z-coordinate of $\mathbf{p}_i$, $\mathbf{A}$ is the matrix of the camera intrinsic parameters (see e.g. [4] for more details)

$$\mathbf{A} = \begin{bmatrix} fk_u & -fk_u \cot\phi & u_0 \\ 0 & fk_v \sin\phi & v_0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \alpha_u & \alpha_{uv} & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3}$$

and where $(x_i, y_i)^t$ are obtained from the pixel coordinates by means of the relationship

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} \frac{1}{\alpha_u} & -\frac{\alpha_{uv}}{\alpha_u \alpha_v} \\ 0 & \frac{1}{\alpha_v} \end{bmatrix} \begin{bmatrix} u_i - u_0 \\ v_i - v_0 \end{bmatrix} \tag{4}$$

$$\mathbf{L}_{s_i}(\mathbf{s}_i, Z_i, \mathbf{A}) = \begin{bmatrix} \alpha_u & \alpha_{uv} \\ 0 & \alpha_v \end{bmatrix} \begin{bmatrix} -\frac{1}{Z_i} & 0 & \frac{x_i}{Z_i} & x_i y_i & -(1+x_i^2) & y_i \\ 0 & -\frac{1}{Z_i} & \frac{y_i}{Z_i} & (1+y_i^2) & -x_i y_i & -x_i \end{bmatrix} \quad (2)$$

If several image points are used, the interaction matrix is obtained by simply stacking the matrices for each point

$$\mathbf{L}_s = \begin{bmatrix} \vdots \\ \mathbf{L}_{s_i} \\ \vdots \end{bmatrix} \quad (5)$$

1.2 Definitions

Unless otherwise stated, the frame of reference is the camera frame. All the coordinates are expressed with respect to this frame of reference.

Thus, let $\mathbf{p}_i = (X_i, Y_i, Z_i)^t$ be the coordinates of the i point of the object, expressed in the camera frame, at the *current* camera pose. Let $\mathbf{p}_i^*$ be its coordinates at the *desired* camera pose.

Without loss of generality, let us define a frame $\mathcal{F}_b$ attached to the object, whose origin $\mathbf{p}$ is the center of gravity of the object. Then, assuming that the object is composed of n points,

$$\mathbf{p} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{p}_i \quad (6)$$

Let the rotation matrix $\mathbf{R}$ be the rotation component of $\mathcal{F}_b$, i.e., the rotation between the camera and object frames. Identically, let $\mathbf{p}^*$ and $\mathbf{R}^*$ be respectively the position and orientation of the object in the desired camera frame.

The coordinates of the points can be readily expressed with respect to the object frame. Let ${}^b\mathbf{p}_i$ be such a coordinates vector, which is related to the coordinates in the camera frame by

$$\mathbf{p}_i = \mathbf{R}\,{}^b\mathbf{p}_i + \mathbf{p} \quad (7)$$

$$\mathbf{p}_i^* = \mathbf{R}^*\,{}^b\mathbf{p}_i + \mathbf{p}^* \quad (8)$$

From the previous definitions, it follows that

$$\sum_{i=1}^{n} {}^b\mathbf{p}_i = \mathbf{0} \quad and \quad \sum_{i=1}^{n} \mathbf{p}_i = n\mathbf{p} \quad (9)$$

Let us define the *skew-symmetric* matrix of vector $\mathbf{a}$ as

$$[\mathbf{a}]_\times = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix} \quad (10)$$

so that $\mathbf{a} \times \mathbf{b} = [\mathbf{a}]_\times \mathbf{b}$. Then

$$\sum_{i=1}^{n} \left[{}^b\mathbf{p}_i\right]_\times = \mathbf{0} \quad and \quad \sum_{i=1}^{n} [\mathbf{p}_i]_\times = n\,[\mathbf{p}]_\times \quad (11)$$

2 Combining pixels and depth information

The main advantage of image-based visual servoing is the direct utilization of image features, without the need to compute the three-dimensional pose of the target. However, in the original approach, depth information corresponding to each image point must be introduced in the Jacobian matrix, as well as the camera calibration parameters. In fact, if these informations are known, the three-dimensional coordinates of the points can be readily obtained, and 3D image features can be used.

2.1 The feature vector

However, the explicit calculus of the coordinates is not necessary. Assuming a camera model without distortion, the relationship between the pixel coordinates $(u_i, v_i)^t$ and the image coordinates $(x_i, y_i)^t$ is linear:

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} \alpha_u & \alpha_{uv} \\ 0 & \alpha_v \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \end{bmatrix} \quad (12)$$

Since the image coordinates are obtained from the perspective projection of the coordinates of each point $\mathbf{p}_i$ in the camera frame,

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \frac{1}{Z_i} \begin{bmatrix} X_i \\ Y_i \end{bmatrix} \quad (13)$$

the following relationship is readily obtained:

$$\begin{bmatrix} u_i Z_i \\ v_i Z_i \\ Z_i \end{bmatrix} = \begin{bmatrix} \alpha_u & \alpha_{uv} & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_i = \mathbf{A}\mathbf{p}_i \quad (14)$$

Let $\mathbf{s}_i = (u_i Z_i, v_i Z_i, Z_i)^t$ be the *feature vector* which corresponds to point i. Then:

$$\mathbf{s}_i = \mathbf{A}\mathbf{p}_i \quad (15)$$

and its derivative is

$$\dot{\mathbf{s}}_i = \mathbf{A}\dot{\mathbf{p}}_i \quad (16)$$

2.2 The interaction matrix

The velocity of point i of the object, expressed in terms of the velocity screw of the camera $\mathbf{v} = [\boldsymbol{\nu}^t, \boldsymbol{\omega}^t]^t$, is

$$\dot{\mathbf{p}}_i = -\boldsymbol{\nu} + [\mathbf{p}_i]_\times \boldsymbol{\omega} \tag{17}$$

Thus, the derivative of the feature vector is

$$\dot{\mathbf{s}}_i = \begin{bmatrix} -\mathbf{A} & \mathbf{A}\,[\mathbf{p}_i]_\times \end{bmatrix} \mathbf{v} \tag{18}$$

For a feature vector composed of a set of points, i.e., $\mathbf{s} = (\ldots, \mathbf{s}_i^t, \ldots)^t$, the interaction matrix $\mathbf{L}_s$ is:

$$\mathbf{L}_s = \begin{bmatrix} \vdots & \\ -\mathbf{A} & \mathbf{A}\,[\mathbf{p}_i]_\times \\ \vdots & \end{bmatrix} \tag{19}$$

where each $\mathbf{p}_i$ can be obtained from the corresponding $\mathbf{s}_i$ since, from Equation 15,

$$\mathbf{p}_i = \mathbf{A}^{-1}\mathbf{s}_i \tag{20}$$

where

$$\mathbf{A}^{-1} = \begin{bmatrix} \frac{1}{\alpha_u} & -\frac{\alpha_{uv}}{\alpha_u \alpha_v} & -\frac{-\alpha_v u_0 + \alpha_{uv} v_0}{\alpha_u \alpha_v} \\ 0 & \frac{1}{\alpha_v} & -\frac{v_0}{\alpha_v} \\ 0 & 0 & 1 \end{bmatrix} \tag{21}$$

which always exists, since neither α_u nor α_v are null.

Then, the interaction matrix can be written in terms of the feature vector only as

$$\mathbf{L}_s = \begin{bmatrix} \vdots & \\ -\mathbf{A} & \mathbf{A}\left[\mathbf{A}^{-1}\mathbf{s}_i\right]_\times \\ \vdots & \end{bmatrix} \tag{22}$$

Usually, the pseudo-inverse of the interacion matrix is computed numerically at each iteration of the control algorithm. However, it may be derived symbolically, as we show in next section. For that purpose, the expression of the matrix in Equation 19 is utilized.

We do not intend to use the symbolic result in the control loop, but it will be useful to calculate the velocity screw of the camera.

We will demonstrate that, for small rotations of the camera, such velocity screw can be approximated by a compact expression which *only depends on the current and desired poses of the object* in the camera frame.

2.3 Computation of the pseudo-inverse matrix

In general, matrix $\mathbf{L}_s$ is not square, thus it is not invertible. However, a pseudo-inverse matrix $\mathbf{L}_s^+$ can be calculated as,

$$\mathbf{L}_s^+ = \left(\mathbf{L}_s^t \mathbf{L}_s\right)^{-1} \mathbf{L}_s^t \tag{23}$$

so that $\mathbf{L}_s^+ \mathbf{L}_s = \mathbf{I}$. Let us calculate the product $\mathbf{L}_s^t \mathbf{L}_s$ of the interaction matrix by its transpose:

$$\begin{bmatrix} \cdots & -\mathbf{A}^t & \cdots \\ & -[\mathbf{p}_i]_\times \mathbf{A}^t & \end{bmatrix} \begin{bmatrix} \vdots & \\ -\mathbf{A} & \mathbf{A}\,[\mathbf{p}_i]_\times \\ \vdots & \end{bmatrix} = \begin{bmatrix} n\mathbf{A}^t\mathbf{A} & -n\mathbf{A}^t\mathbf{A}\,[\mathbf{p}]_\times \\ n\,[\mathbf{p}]_\times \mathbf{A}^t\mathbf{A} & -n\,[\mathbf{p}]_\times \mathbf{A}^t\mathbf{A}\,[\mathbf{p}]_\times - \mathbf{S} \end{bmatrix} \tag{24}$$

where,

$$\mathbf{S} = \mathbf{R}\mathbf{M}\mathbf{R}^t \tag{25}$$

$$\mathbf{M} = \sum_{i=1}^{n} \left[{}^b\mathbf{p}_i\right]_\times \mathbf{R}^t\mathbf{A}^t\mathbf{A}\mathbf{R}\left[{}^b\mathbf{p}_i\right]_\times \tag{26}$$

The inverse matrix of the above product $\mathbf{L}_s^t \mathbf{L}_s$ is

$$\begin{bmatrix} \frac{1}{n}(\mathbf{A}^t\mathbf{A})^{-1} + [\mathbf{p}]_\times \mathbf{S}^{-1}\,[\mathbf{p}]_\times & -[\mathbf{p}]_\times \mathbf{S}^{-1} \\ \mathbf{S}^{-1}\,[\mathbf{p}]_\times & -\mathbf{S}^{-1} \end{bmatrix} \tag{27}$$

which exists as long as $\mathbf{M}$ has an inverse, since $\mathbf{S}^{-1} = \mathbf{R}\mathbf{M}^{-1}\mathbf{R}^t$. Finally, the pseudo-inverse matrix is

$$\mathbf{L}_s^+ = \begin{bmatrix} \cdots & -\frac{1}{n}(\mathbf{A}^t\mathbf{A})^{-1}\mathbf{A}^t + [\mathbf{p}]_\times \mathbf{T}_i & \cdots \\ & \mathbf{T}_i & \end{bmatrix} \tag{28}$$

where

$$\mathbf{T}_i = \mathbf{R}\mathbf{M}^{-1}\left[{}^b\mathbf{p}_i\right]_\times \mathbf{R}^t\mathbf{A}^t \tag{29}$$

2.4 Computation of the velocity screw

Prior to the calculation of the velocity screw, we must define the error vector $\mathbf{e}$ between the current feature vector s and the desired one $\mathbf{s}^*$:

$$\begin{aligned} \mathbf{e} = \mathbf{s} - \mathbf{s}^* &= \begin{bmatrix} \vdots \\ \mathbf{A}\,(\mathbf{p}_i - \mathbf{p}_i^*) \\ \vdots \end{bmatrix} \\ &= \begin{bmatrix} \vdots \\ \mathbf{A}\left((\mathbf{p} - \mathbf{p}^*) + (\mathbf{R} - \mathbf{R}^*)\,{}^b\mathbf{p}_i\right) \\ \vdots \end{bmatrix} \end{aligned} \tag{30}$$

The velocity screw is then calculated by means of the task function approach [8, 3]:

$$\mathbf{v} = -\lambda \mathbf{L}_s^+ \mathbf{e} \tag{31}$$

From the above definitions and the properties presented in Equations 9 and 11,

$$\begin{aligned} \mathbf{v} &= -\lambda \begin{bmatrix} -\frac{1}{n}\sum_{i=1}^{n}(\mathbf{p}-\mathbf{p}^*) \\ \mathbf{0} \end{bmatrix} \\ &\quad -\lambda \begin{bmatrix} [\mathbf{p}]_\times \sum_{i=1}^{n} \mathbf{T}_i \mathbf{A}(\mathbf{R}-\mathbf{R}^*)^b\mathbf{p}_i \\ \sum_{i=1}^{n} \mathbf{T}_i \mathbf{A}(\mathbf{R}-\mathbf{R}^*)^b\mathbf{p}_i \end{bmatrix} \\ &= -\lambda \begin{bmatrix} (\mathbf{p}^*-\mathbf{p}) + [\mathbf{p}]_\times \mathbf{R}\mathbf{W} \\ \mathbf{R}\mathbf{W} \end{bmatrix} \end{aligned} \tag{32}$$

where

$$\mathbf{W} = \sum_{i=1}^{n} \mathbf{M}^{-1} \left[{}^b\mathbf{p}_i\right]_\times \mathbf{R}^t\mathbf{A}^t\mathbf{A}(\mathbf{R}-\mathbf{R}^*)^b\mathbf{p}_i \tag{33}$$

According to the properties of rotation matrices, the above difference is

$$\mathbf{R}-\mathbf{R}^* = -\mathbf{R}\left[\mathbf{u}\right]_\times^2 (1-\cos\theta) - \mathbf{R}\left[\mathbf{u}\right]_\times \sin\theta \tag{34}$$

where $\mathbf{u}\theta$ are the axis and the angle which correspond to the rotation matrix product $\mathbf{R}^t\mathbf{R}^*$.

If θ is small, we can take the approximations $\sin\theta \approx \theta$ and $\cos\theta \approx 1$. Then,

$$\begin{aligned} \mathbf{W} &\approx \sum_{i=1}^{n} \mathbf{M}^{-1} \left[{}^b\mathbf{p}_i\right]_\times \mathbf{R}^t\mathbf{A}^t\mathbf{A}\left(-\mathbf{R}\left[\mathbf{u}\right]_\times \theta\right) {}^b\mathbf{p}_i \\ &\approx \mathbf{M}^{-1}\sum_{i=1}^{n} \left[{}^b\mathbf{p}_i\right]_\times \mathbf{R}^t\mathbf{A}^t\mathbf{A}\mathbf{R}\left[{}^b\mathbf{p}_i\right]_\times \mathbf{u}\theta \end{aligned} \tag{35}$$

which, according to the definition of $\mathbf{M}$ in Equation 26, leads to

$$\mathbf{W} \approx \mathbf{u}\theta \tag{36}$$

Thus, the velocity screw is

$$\mathbf{v} \approx -\lambda \begin{bmatrix} (\mathbf{p}^*-\mathbf{p}) + [\mathbf{p}]_\times \mathbf{R}\mathbf{u}\theta \\ \mathbf{R}\mathbf{u}\theta \end{bmatrix} \tag{37}$$

2.5 Motion of the object in the camera frame

Let us calculate the velocity of the center of gravity of the object $\dot{\mathbf{p}}$. From Equations 17 and 32,

$$\begin{aligned} \dot{\mathbf{p}} &= -\lambda\left(-(\mathbf{p}^*-\mathbf{p}) - [\mathbf{p}]_\times \mathbf{R}\mathbf{W} + [\mathbf{p}]_\times \mathbf{R}\mathbf{W}\right) \\ &= \lambda(\mathbf{p}^*-\mathbf{p}) \end{aligned} \tag{38}$$

Thus, the center of gravity of the object moves along a *straight line trajectory* from its initial to its final position in the camera frame. Consequently, the object is most likely to remain in the camera field of view during the whole task.

3 The use of 3D point coordinates

The presented control law utilizes the pixel coordinates and depth estimation of each point, which are the same requirements as those of classic image-based visual servoing. However, additional interesting results can be obtained if the feature vector is composed of the coordinates of the points of the object, i.e., $\mathbf{s} = (\ldots, \mathbf{p}_i^t, \ldots)$.

One should note that any additional information is needed for the computation of such coordinates, which are obtained from pixels and depth estimates as shown in Equation 20.

3.1 The interaction matrix

Instead of repeating the development, all of the previous theoretical results can be used by means of simply replacing $\mathbf{A}$ by the identity matrix. Thus, the new interaction matrix is

$$\mathbf{L}_s = \begin{bmatrix} & \vdots & \\ -\mathbf{I} & & [\mathbf{p}_i]_\times \\ & \vdots & \end{bmatrix} \tag{39}$$

and its pseudo-inverse is

$$\mathbf{L}_s^+ = \begin{bmatrix} \cdots & -\frac{1}{n}\mathbf{I} + [\mathbf{p}]_\times \mathbf{T}_i & \cdots \\ & \mathbf{T}_i & \end{bmatrix} \tag{40}$$

where

$$\mathbf{T}_i = \mathbf{R}\mathbf{M}^{-1}\left[{}^b\mathbf{p}_i\right]_\times \mathbf{R}^t \tag{41}$$

$$\mathbf{M} = \sum_{i=1}^{n} \left[{}^b\mathbf{p}_i\right]_\times^2 \tag{42}$$

3.2 The velocity screw

The same approximation of the velocity screw of the camera can be obtained. Nonetheless, *an exact computation of the velocity screw for any rotation* can be obtained if $\mathbf{M}$ is a diagonal matrix (some objects for $\mathbf{M}$ is a diagonal matrix are, e.g., the tetrahedron, the square and the cube).

From the properties of skew-symmetric matrices,

$$\mathbf{M} = \sum_{i=1}^{n} {}^b\mathbf{p}_i {}^b\mathbf{p}_i{}^t - \sum_{i=1}^{n} {}^b\mathbf{p}_i{}^t {}^b\mathbf{p}_i \mathbf{I} \tag{43}$$

Since both $\mathbf{M}$ and the second sum are diagonal, then the first sum is diagonal too, i.e.,

$$\sum_{i=1}^{n} {}^b\mathbf{p}_i {}^b\mathbf{p}_i{}^t = \alpha\mathbf{I} \tag{44}$$

where α is a positive scalar.

Then, from Equations 33, 34 and 44,

$$\begin{aligned}
\mathbf{W} &= -\mathbf{M}^{-1} \sum_{i=1}^{n} \left[{}^b\mathbf{p}_i \right]_\times \left[\mathbf{u} \right]_\times^2 {}^b\mathbf{p}_i (1 - \cos\theta) \\
&\quad + \mathbf{M}^{-1} \sum_{i=1}^{n} \left[{}^b\mathbf{p}_i \right]_\times^2 \mathbf{u} \sin\theta \\
&= -\mathbf{M}^{-1} \sum_{i=1}^{n} \left[{}^b\mathbf{p}_i \right]_\times \mathbf{u}\mathbf{u}^t {}^b\mathbf{p}_i (1 - \cos\theta) + \mathbf{u} \sin\theta \\
&= \mathbf{M}^{-1} \left[\mathbf{u} \right]_\times \sum_{i=1}^{n} {}^b\mathbf{p}_i {}^b\mathbf{p}_i{}^t \mathbf{u} (1 - \cos\theta) + \mathbf{u} \sin\theta \\
&= \alpha \mathbf{M}^{-1} \left[\mathbf{u} \right]_\times \mathbf{u} (1 - \cos\theta) + \mathbf{u} \sin\theta \\
&= \mathbf{u} \sin\theta \qquad (45)
\end{aligned}$$

The velocity screw of the camera is,

$$\mathbf{v} = -\lambda \begin{bmatrix} (\mathbf{p}^* - \mathbf{p}) + [\mathbf{p}]_\times \mathbf{R}\mathbf{u} \sin\theta \\ \mathbf{R}\mathbf{u} \sin\theta \end{bmatrix} \qquad (46)$$

which is valid for all $\mathbf{u}\theta$.

As shown in Equation 38, the object will follow a straight line path in the camera frame. However, care must be taken if $\frac{\pi}{2} < \theta \leq \pi$, since $\sin\theta$ decreases. Consequently, the camera *accelerates* during the first phase of the motion until $\theta = \frac{\pi}{2}$ then decelerates until convergence. This may cause unexpected behavior if the initial velocity of the camera is near to the maximum limit in order to improve convergence times.

Finally, if $\theta = \pi$ then the camera does not rotate at all, which will produce inconsistent results if the translational motion *does not change the orientation of the object with respect to the camera*, e.g. if the object is positioned along the straight line which joins the current and desired camera positions.

4 Experimental results

The control law has been implemented on a robotic platform with 6 degrees of freedom and an eye-in-hand configuration. The target object is composed of four points which define a tetrahedron. Three-dimensional coordinates are obtained from the pose of the object, which is extracted from the images and an internal model with Dementhon's algorithm [2].

Initial and desired poses of the camera are shown in Table 1. As explained in [6], this is a difficult task since the displacement is important and the object moves towards the border of the image.

Pose	Translation (mm)	Rotation $\mathbf{u}\theta$ (deg)
Initial	0 0 -500	0 0 0
Desired	-225 249 -408	7 37 -70

Table 1: Initial and desired poses of the camera.

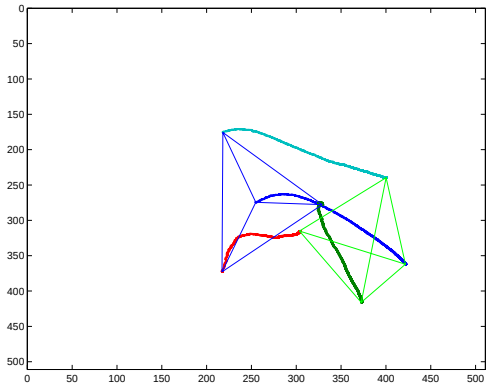

Figure 1: Object trajectory in the image when using image point features.

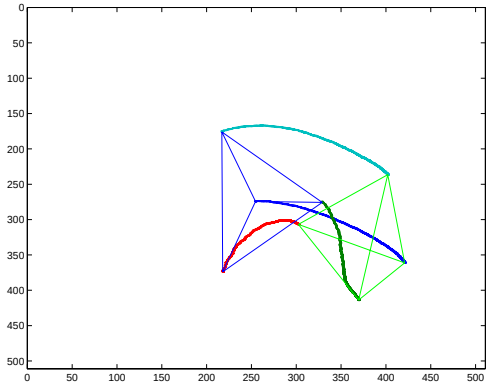

Figure 2: Object trajectory in the image when using pixel/depth features.

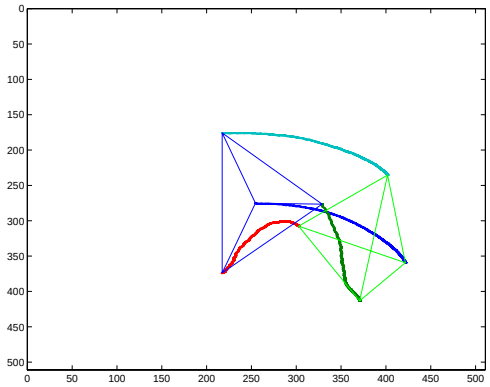

Figure 3: Object trajectory in the image when using 3D point features.

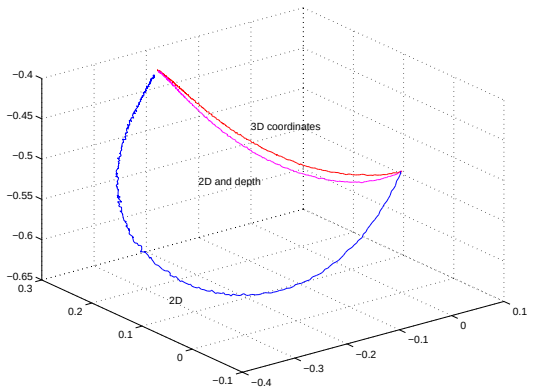

Figure 4: Trajectories of the camera.

Figures 1, 2 and 3 depict the trajectories of the object in the image plane. Since the involved rotation is high, the points of the object do not follow straight lines from the initial to the final positions. Though trajectories are more curved in the proposed approaches than in image-based visual servoing, the object reemains in the field of view of the camera during the complete servoing task.

In addition, the trajectory of the camera in the proposed approaches is not as elongated as in the image-based scheme, thus preventing the manipulator from reaching its joint limits. Camera trajectories in an absolute frame are depicted in Figure 4.

5 Conclusion

The use of new visual features in image-based visual servoing has been proposed, without the need of additional information, i.e., only the pixel coordinates and the estimated depth of each point is used.

Such new features result in a linear control law, from which the computed screw can be obtained, resulting in a motion of the object along a straight path in the camera frame; thus, the object is most likely to remain in the field of view of the camera during the visual servoing task.

The use of 3D coordinates of the points of the object, obtained from the image features, the estimated depth and the camera parameters, is proposed too. We demonstrate that it is a particular case of the first proposed scheme. Nevertheless, an exact value of the velocity screw is obtained, provided that an additional geometric property of the target object is met.

Both proposed approaches exhibit a better behavior than the classic image-based one, since the trajectories of the camera in the absolute frame are less elongated when the change of orientation is high. Thus, the risk of the robot going out of its joint limits is lowered.

Future work includes a comparative study of the efficiency and robustness of the proposed schemes and the classic image-based and position-based visual servoing approaches, particularly in face of camera calibration errors.

Acknowledgements

Support from Fundació Caixa Castelló - Bancaixa for a temporal stay of E. Cervera in Blaise Pascal University is gratefully acknowledged.

References

[1] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In G. Hager, D. Kriegman, and A. Morse, editors, *The Confluence of Vision and Control*. Springer Verlag, 1998.

[2] D. F. Dementhon and L. S. Davis. Model-based object pose in 25 lines of code. *International Journal of Computer Vision*, 15(1/2):123–141, June 1995.

[3] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8(3):313–326, June 1992.

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

[5] S. Hutchinson, G. D. Hager, and P. I. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, October 1996.

[6] E. Malis. *Contributions à la Modélisation et à la Commande en Asservissement Visuel*. PhD thesis, Université de Rennes, France, November 1998.

[7] P. Martinet, J. Gallice, and D. Khadraoui. Vision-based control law using 3d visual features. In *World Automation Congress, WAC'96, Robotics and Manufacturing Systems*, pages 497–502, Montpellier, France, May 1996.

[8] C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: the Task Function Approach*, volume 22 of *Oxford Engineering Science Series*. Clarendon Press, Oxford, United Kingdom, 1991.