

Full paper

Visual Control of Robots with Delayed Images

Carlos Perez-Vidal^a, Luis Gracia^{b,*}, Nicolas Garcia^a and Enric Cervera^c

^a Industrial Systems Engineering Department, Universidad Miguel Hernández, Avenida de la Universidad s/n, Edificio TorreBlanca, 03202 Elche, Spain

^b Instituto IDF, Universidad Politécnica de Valencia, Camino de Vera s/n, 46022 Valencia, Spain

^c Department of Computer Science and Engineering, Universitat Jaume I, Avenida de Vicent Sos Baynat s/n, 12071 Castellón de la Plana, Spain

Received 4 August 2008; accepted 11 November 2008

Abstract

This research develops a control scheme for visual servoing that explicitly takes into account the delay introduced by image acquisition and processing. For this purpose, a predictor block, i.e., an estimator that predicts several samples ahead of time, is properly included in the scheme. The proposed approach is analytically analyzed in terms of dynamics and steady-state errors, and compared to previous approaches. Furthermore, several simulations are comparatively shown in order to illustrate the benefits and limitations of the proposed control scheme. Finally, some experimental results using a turntable and a 3-d.o.f. Cartesian robot are provided in order to validate the analytical and simulation results.

© Koninklijke Brill NV, Leiden and The Robotics Society of Japan, 2009

Keywords

Control scheme, Kalman filter, prediction filters, visual servoing

1. Introduction

During the last few years, the use of visual servoing and visual tracking has been more and more common due to the increasing power of algorithms and computers. The beginnings of this ‘in fashion’ discipline started in 1973 when Shirai and Inoue first described [1] a novel method for ‘visual control’ of a robotic manipulator using a vision feedback loop. This method promised the ability to deal with real-time changes in the relative position of the part with respect to the robot as well as greater precision. The proposed method is loosely analogous to the way a human would observe, navigate to and grasp an object using visual feedback. The term ‘visual servoing’ was later coined to refer to the real-time, visual feedback control of a

* To whom correspondence should be addressed. E-mail: luigraca@isa.upv.es

robotic manipulator, where an appropriate error function is minimized. Robustness, a major benefit of visual servoing, is achievable because the accuracy is inherently independent on the hand–eye calibration in this method. As a result, even if the calibration process was erroneous, the final accuracy would not be affected. The ability to deal with real-time changes makes visual servoing ideal for tasks whereby the workpiece warps and distorts during the robotic operation (e.g., thermal distortion during welding). The increased accuracy of visual servoing makes it an ideal choice for automated assembly tasks where this characteristic is a major requirement. In this sense, a wide range of applications that use visual servoing are described in Ref. [2]. The scientific advances in visual servoing have been recently accelerated because of the fact that current processors can analyze the scenes supplied by the vision system in real-time in order to generate control actions for the robot. These advances have given rise to recent special issues on visual servoing in prestigious journals [3–5].

In Refs [6–9] the basic concepts of visual servoing, which were used in subsequent research, were collected and unified. There are two basic approaches for visual servoing depending on the used error signal: (i) image-based visual servoing, in which an error signal is measured directly in the image and mapped to actuator commands, and (ii) position-based visual servoing, in which computer vision techniques are used to reconstruct a representation of the three-dimensional (3-D) workspace of the robot, and actuator commands are computed with respect to the 3-D workspace. In an image-based system the pose estimation is solved implicitly (if the current view of the object matches the desired view, then the object must be in the desired relative pose). However, there are specific approaches for visual servoing that cannot be included in the previous classification (e.g., Ref. [10]).

On the other hand, in Ref. [11] the approaches for visual servoing are classified in two groups depending on the used low-level controller: (i) indirect visual servoing, in which a kinematic controller (usually a proportional controller) commands the robot in Cartesian coordinates (position or velocity) using the joint controllers of the robot itself, and (ii) direct visual servoing, in which the controller commands directly the motor's voltage or torque (therefore, a specific controller developed for visual servoing tasks is used). An example of direct visual servoing can be found in Ref. [12], in which automatic control theory in the larger domain of sensor-based control is used. Meanwhile, an example of a control scheme for indirect visual servoing can be found in Ref. [13], in which a proportional-derivative (PD) controller is used as kinematic controller (it is also used the extended Kalman filter to handle noisy image feature measurements). The previous control scheme has two main drawbacks: the velocity error (steady-state error when the reference is a first-order ramp) is non-zero (proved in Proposition 4 of Section 3) and it does not take into account the vision system delays. In this sense, in Ref. [14] an image-based indirect visual servoing is developed, where it is pointed out that the image processing delay produces an oscillating behaviour for high feedback gains.

In fact, one major control problem of visual servoing is to cope with the delay introduced by image acquisition and image processing. Much research about visual servoing [12–15] is focused on other aspects and simply ignores this delay in the control scheme (they usually use an indirect visual servoing scheme based on a proportional feedback), which is the main reason for limited tracking velocity and acceleration. One approach to overcome the delay introduced by the visual sensor is the usage of predictive algorithms, e.g., the Kalman filter, which is also useful to cope with the typically noisy signals. Basically, the unique proposed control scheme that considers the visual sensor delays is presented by Corke and Good [17, 18], which has been used in subsequent research [19, 20], although it has several drawbacks that are pointed out throughout the paper. In order to overcome them, the authors of this work develop a trajectory control-based scheme that takes into account the mentioned delay by properly including a predictor block, i.e., an estimator that predicts several samples ahead of time, in the scheme. In particular, the proposed approach is fully described in Section 2 and analytically analyzed (dynamics and steady-state errors) in Section 3. Furthermore, several simulations are comparatively shown in Section 4 to illustrate the benefits and limitations of the proposed control scheme, while other advantages are pointed out in Section 5. Finally, some experimental results using a turntable and a 3-d.o.f. Cartesian robot are provided in Section 6 to validate the analytical and simulation results.

2. Description of the Control Scheme

One usual objective in visual servoing, especially in tracking applications, is to bring the target to a position of the image plane and to keep it there for any object's movement. The most popular control scheme specifically designed for visual servoing is presented by Corke and Good [17, 18], which has been used in several subsequent research [19]. Figure 1 depicts this control scheme (hereinafter it will be simply referenced as proposed in Ref. [17]) with a block re-arrangement and with the notation of this work. In this scheme $R(z)$ represents the robot discrete model (including the joint controllers) and $V(z)$ the camera behaviour (Table 1 shows the nomenclature of the schemes), which is commonly modelled as a pure delay $V(z) = k_v z^{-d}$ (usually $d = 2$) [6–9, 17, 21, 22]. In fact, the delay block z^{-d}

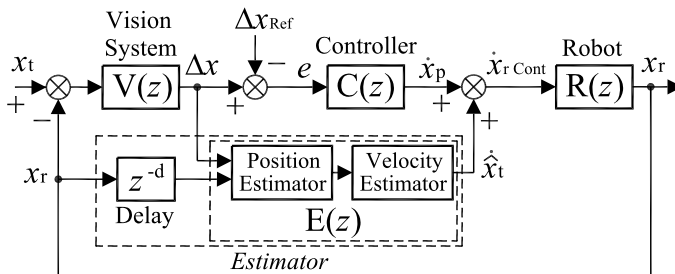


Figure 1. Control scheme developed by Corke and Good [17, 18] for visual servoing.

Table 1.

Nomenclature for control schemes of Figs 1–3

Symbol	Description
k	discrete time step
d	vision system delay
$m(i)$	value of the discrete signal m at the discrete step time i
$V(z)$	discrete transfer function of the vision system
$E(z)$	discrete transfer function of the estimator/prediction filter
$C(z)$	discrete transfer function of the controller that corrects the position error
$Der(z)$	transfer function of the first-order discrete derivator, i.e., $(z - 1)/(T \cdot z)$
$R(z)$	discrete transfer function of the robot global behaviour
z^{-d}	discrete transfer function of the delay of d samples
x_r	robot position
\dot{x}_r	robot velocity
x_t	target position
\hat{x}_t	estimation of the target position
$\dot{\hat{x}}_t$	estimation of the target velocity
\dot{x}_p	output of the controller that corrects the position error
Δx	vision system output (equal to $x_t - x_r$)
Δx_{Ref}	desired difference between the target position and the robot position
$\dot{x}_r \text{ Cont}$	discrete command obtained by $C(z)$ to control the robot
$\hat{x}_r \text{ Ref}$	estimation of the robot reference
$\dot{\hat{x}}_r \text{ Ref}$	first-order time derivative of the estimated robot reference
$D(z)$	discrete transfer function of the low-level dynamic controller
$M(z)$	discrete transfer function of the low-level velocity estimator based in the encoder signals
$R'(s)/s$	low-level continuous robot model
u_r	control action obtained by the low-level dynamic controller to command the robot
ZOH	zero-order hold
T_h	(high/slow) sample time of the discrete kinematic control loop
T_s	(small/fast) sample time of the discrete dynamic control loop

is introduced in the diagram to make the step time of the variables agree. The vision system delay d is usually one or two sample times. It is one when the image processing time T_{proc} is negligible with respect to the image acquisition time T_{adq} , or *vice versa*, and both computations are carried out in one sample time. Meanwhile, it is two when both computation times (T_{proc} and T_{adq}) are similar and each one requires one sample time, i.e., a kind of segmentation is applied: there is one data per sample time with latency 2. In the scheme an estimation block $E(z)$ (e.g., a Kalman filter in Ref. [19]) is used to generate the feedforward signal $\dot{\hat{x}}_t$, i.e., the estimation of the first-order time-derivative of the target position. Meanwhile, the controller $C(z)$ minimizes the deviation (error) between Δx (difference between the target position x_t and the robot position x_r , which is obtained with the vision system and, therefore, is delayed d samples) and Δx_{Ref} (desired difference between the target position and the robot position) with the signal \dot{x}_p . In short, a classical trajectory

control is used as position/kinematic controller: a first-order time-derivative feed-forward plus a position correction.

This research proposes to use the control scheme of Fig. 2 (the step times of the different signals have been explicited), where the estimation block $E(z)$ takes into account that the measurements are delayed (step time $k - d$) and predicts the outputs for the current step time k . Another important modification with respect to the scheme of Fig. 1 is that the prediction filter output $\hat{x}_t(k)$ and the current robot position $x_r(k)$ are used in Fig. 2 (apart from Δx_{Ref}) as input for the kinematic controller, instead of the vision system output $\Delta x(k - d)$ and the delayed robot position $x_r(k - d)$. This allows us to make the position correction using values of the current step time k instead of values of the past step time $k - d$. The output of the block group that represents the kinematic controller is the control signal $\dot{x}_r Cont(k)$ that commands the robot. This kinematic controller represents, as before, the classical trajectory control: a first-order time-derivative feedforward of the robot reference $x_{r Ref}$ (i.e., $x_t - \Delta x_{Ref}$) plus a position error correction with the controller $C(z)$. Note that, the controller $C(z)$ uses the discrete derivator $Der(z)$ to generate the first-order time-derivative feedforward of the robot referente $x_{r Ref}$. However, it may be alternatively removed if the estimator predicts the current target velocity \dot{x}_t , apart from the current target position.

Meanwhile, the robot global behaviour $R(z)$ has been detailed in Fig. 3 with the low-level dynamic controller $D(z)$, which is usually discrete and corrects the robot velocity error, and the low-level continuous robot model $R'(s)/s$, which consists on the robot inertia (the robot intrinsic integrator is considered explicitly) and the

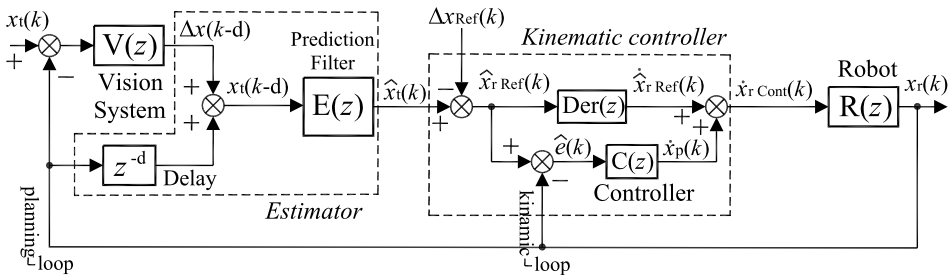


Figure 2. Control scheme developed in this work for visual servoing.

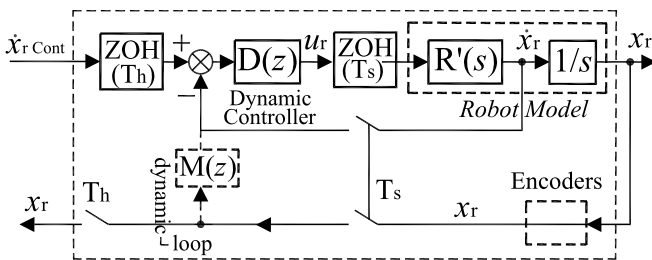


Figure 3. Detailed operation diagram of the robot behaviour $R(z)$.

actuator dynamics. Note that a possible velocity estimator $M(z)$ for the case that the robot velocity \dot{x}_r is obtained discretely from the robot encoders (e.g., if the backward Euler approximation is used $M(z) = (z - 1)/(z \cdot T_s)$), has been included. In the diagram two sample times have also been considered: a small (fast) sample time T_s for the internal robot joint loop (usually near to 1 ms) and a high (slow) sample time T_h given by the acquisition of the external robot joint loop (usually near to 40 ms in visual servoing applications for PAL and RS170 standards). In order to change from one sample time to the other and between continuous and discrete signals, two zero-order holds (ZOH) and two samplers have been included in the control scheme.

In short, in the scheme of Figs 2 and 3 there are three loops: the outer loop (planning loop) establishes dynamically the robot position reference $x_{r \text{ Ref}}$ (i.e., $x_t - \Delta x_{\text{Ref}}$) from the sensors, the outer-medium loop (kinematic loop) corrects the error of the position signal and the inner-medium loop (dynamic loop) corrects the error of the velocity signal. Moreover, there is a fourth loop: the inner loop (current loop) that corrects the error of the acceleration signal. However, it has not been represented in Fig. 3 because it is usually integrated within the actuator and its dynamics (given by the actuator electric components) is very fast. In the next sections the medium loops, i.e., kinematic and dynamic loops, will be simply referred to as the outer and inner loops. Note that the sensors (e.g., vision system) and/or planners give one scalar reference for each coordinate of the robot. In this sense, the scheme of Fig. 2 would be replicated for each coordinate as long as the robot control of each coordinate is decoupled. This is the case of a Cartesian robot with small (negligible) variations in the joint frictions due to the accelerations of other robot axes. Otherwise, the signals of Fig. 2 are vectors and the robot joint control is performed coupled in the block $R(z)$. Note that, in that case several blocks (i.e., inverse velocity kinematics and direct position kinematics) must be added in the robot block $R(z)$ to transform the Cartesian coordinates in joint coordinates and *vice versa*.

3. Dynamics and Steady-State Errors

3.1. Definitions and Assumptions

The pure kinematic framework is said to be when the low-level dynamic control is considered instantaneous, e.g., the robot inertia and the actuator dynamics are negligible.

The discrete kinematic framework is said to be the pure kinematic framework where the kinematic controller is discrete, i.e., $T_h \neq 0$. In that case, $R(z) = \text{Int}(z) = T_h/(z - 1)$ (discrete integrator), see Fig. 3.

The continuous kinematic framework is said to be the pure kinematic framework where the kinematic controller is continuous, i.e., $T_h = 0$. In that case, $R(z) = 1/s$ (continuous integrator), see Fig. 3.

A dynamic framework is said to be when the low-level dynamic control is not considered instantaneous.

It is said that the position/velocity/acceleration error of a system is null if the steady-state error of the output signal is zero when the reference (input) is a zero/first/second-order ramp.

The perfect estimator is said to be the estimator that estimates the output variables with no error, e.g., $\hat{x}_t(k) = x_t(k)$, $\hat{x}_{r \text{ Ref}}(k) = x_{r \text{ Ref}}(k)$, $\hat{e}(k) = e(k)$, etc.

The discrete/continuous ideal framework is said to be the discrete/continuous kinematic framework with the perfect estimator.

Assumption 1. In order to focus on the control capabilities of the different control schemes for visual servoing, a perfect estimator will be assumed in this section.

Assumption 2. The kinematic loop sample time T_h is a synchronised multiple of the dynamic loop sample time T_s , i.e., $T_h = n \cdot T_s$, where n is the sample time ratio.

Assumption 3 (Only for Propositions 2, 3 and 4). The total system (including the control scheme) is stable, i.e., the transient of the forced and free response converges to zero.

In order to simplify the notation, the same symbol will be used in the next subsections for the variables regardless of the domain considered: t , s or z .

3.2. Propositions

Proposition 1. For the control scheme proposed in this research (Fig. 2) and under a continuous kinematic framework (i.e., the kinematic controller is continuous, i.e., $T_h = 0$, and the low-level dynamic control is instantaneous) the error converges exponentially to zero for a proportional kinematic controller (i.e., $C(z) = K$ with $K > 0$) as long as the robot reference trajectory $x_{r \text{ Ref}}(t)$ (i.e., the target trajectory $x_t(t)$ minus the vision system reference trajectory $\Delta x_{\text{Ref}}(t)$) is continuous.

Proof. Since the low-level dynamic control is instantaneous, we have:

$$\dot{x}_{r \text{ Cont}} = \dot{x}_r. \quad (1)$$

The continuous kinematic control is given by:

$$e = x_{r \text{ Ref}} - x_r \quad (2)$$

$$\dot{x}_{r \text{ Cont}} = \dot{x}_{r \text{ Ref}} + K e. \quad (3)$$

Replacing (3) in (1) and using the first-order time-derivatives of (2), we obtain:

$$\dot{x}_r = \dot{x}_{r \text{ Ref}} + K e \quad (4)$$

$$\dot{e} + K e = 0 \quad (5)$$

$$e(t) = e(0)e^{-Kt}. \quad (6)$$

Note that the robot reference trajectory $x_{r \text{ Ref}}(t)$ (i.e., the target trajectory $x_t(t)$ minus the vision system reference trajectory $\Delta x_{\text{Ref}}(t)$) has to be continuous in order

to avoid impossible (i.e., infinite) control actions in (3). Using the same procedure of this proposition, we easily obtain that the control scheme proposed by Corke for visual servoing (Fig. 1) has exponential error convergence if and only if $x_{r \text{ Ref}}(t)$ is continuous (again) and the vision system reference trajectory $\Delta x_{\text{Ref}}(t)$ is constant, which is more restrictive than the only previous condition.

Proposition 2. For the control scheme proposed in this research (Fig. 2) and assuming that the low-level controller $D(z)$ (Fig. 3) has an integral action, the steady-state error is null iff (if and only if) the second-order time-derivative of the robot reference $x_{r \text{ Ref}}$ is null.

Proof. From Fig. 3 and assuming that $T_s \neq 0$, the transfer function between the low-level discrete control action u_r and the robot discrete position $x_{r T_s}$ (both with sample time T_s) is:

$$\frac{x_{r T_s}(z)}{u_r(z)} = \mathcal{Z}_{k=t/T_s} \left[\mathcal{L}^{-1} \left[\frac{R'(s)(1 - e^{-T_s s})}{s^2} \right] \right] = \frac{LR(z)}{\text{Der}(z)}, \tag{7}$$

where $\mathcal{Z}_k[\cdot]$ is the Z-transform with respect to the discrete time variable k .

Using (7) and considering the backward Euler approximation as a low-level velocity estimator, i.e., $M(z) = \text{Der}(z) = (z - 1)/(T_s \cdot z)$, the low-level robot behaviour $R_{T_s}(z)$ (i.e., without including the high-level ZOH and sampler) is:

$$\frac{x_{r T_s}(z)}{\dot{x}_{r \text{ Cont } T_s}(z)} = R_{T_s}(z) = \frac{D(z)LR(z)/\text{Der}(z)}{1 + D(z)LR(z)}, \tag{8}$$

where $\dot{x}_{r \text{ Cont } T_s}$ is the control signal after the ZOH $_{T_h}$ that commands robot. The low-level discrete controller $D(z)$ with an integral action can be rewritten as:

$$D(z) = D''(z) + k_I/(z \cdot \text{Der}(z)) = D'''(z)/\text{Der}(z). \tag{9}$$

From (8) and (9), we obtain:

$$R_{T_s}(z) = \frac{D'''(z)LR(z)/\text{Der}(z)}{\text{Der}(z) + D'''(z)LR(z)} = \frac{DR(z)/\text{Der}(z)}{\text{Der}(z) + DR(z)}, \tag{10}$$

where $R_{T_s}(z) \text{Der}(z)$ has unit static gain. Note that, any discrete transfer function with unit static gain can be rewritten as $X(z)/(\text{Der}(z) + X(z))$. In order to obtain $R(z)$, ZOH $_{T_h}$ and the kinematic sampler T_h have to be included in the previous transfer function $R_{T_s}(z)$ (10), e.g., analytically with the impulse response method or numerically with the command `d2d` in Matlab. However, it can be proved that $R(z)$ has the same form of $R_{T_s}(z)$ (10), since the ZOH and the kinematic sampler do not change it, i.e.:

$$\frac{x_r(z)}{\dot{x}_{r \text{ Cont}}(z)} = R(z) = \frac{N(z)/\text{Der}(z)}{\text{Der}(z) + N(z)}. \tag{11}$$

The kinematic control loop of Fig. 2 is given by:

$$e(z) = x_{r \text{ Ref}}(z) - x_r(z) \tag{12}$$

$$\dot{x}_{r \text{ Cont}}(z) = \text{Der}(z)x_{r \text{ Ref}}(z) + C(z)e(z). \tag{13}$$

Substituting (13) and (12) in (11), we obtain:

$$\frac{e(z)}{x_{r \text{ Ref}}(z)} = \frac{1 - \text{Der}(z)R(z)}{1 + C(z)R(z)}. \tag{14}$$

Note that, when the first-order time derivative and integrative operations are applied to a discrete signal $v(k)$ (e.g., $x_{r \text{ Ref}}$ and all the signals present in the discrete kinematic controller of Fig. 2), we obtain the discrete derivator and integrator:

$$\dot{v}(z) = \mathcal{Z}_k[\dot{v}(k)] = \text{Der}(z)\mathcal{Z}_k[v(k)] = \text{Der}(z)v(z) \tag{15}$$

$$v(z) = \mathcal{Z}_k\left[\int \dot{v}(k)\right] = \text{Int}(z)\mathcal{Z}_k[\dot{v}(k)] = \text{Int}(z)\dot{v}(z). \tag{16}$$

From (11), (14) and (15), the error transfer function is:

$$\frac{e(z)}{\ddot{x}_{r \text{ Ref}}(z)} = \frac{1}{\text{Der}^2(z) + N(z)\text{Der}(z) + C(z)N(z)}. \tag{17}$$

Therefore, and assuming that the system is stable (Assumption 3), the steady-state error is null iff the steady-state value for the second-order time derivative of $x_{r \text{ Ref}}$ is null:

$$e(\infty) = 0 \quad \text{iff} \quad \ddot{x}_{r \text{ Ref}}(\infty) = 0 \quad \longleftrightarrow \quad \ddot{x}_t(\infty) - \Delta\ddot{x}_{\text{Ref}}(\infty) = 0. \tag{18}$$

A very similar demonstration can be obtained for a continuous low-level controller, i.e., for $D(s)$ instead of $D(z)$.

Proposition 3. For the scheme proposed by Corke and Good [17] (Fig. 1), and assuming that the low-level controller $D'(z)$ has an integral action, the steady-state error is null iff both first-order time-derivative of the robot reference $x_{r \text{ Ref}}$ and second-order time-derivative of the target x_t are null.

Proof. Expressions (11) and (12) are also valid for this proposition because the robot block $R(z)$ is the same one (Fig. 3). In this case the kinematic controller is given by:

$$\dot{x}_{r \text{ Cont}}(z) = z^{-d}(\dot{x}_t(z) + C(z)e(z)) \tag{19}$$

$$\dot{x}_t(z) = \Delta\dot{x}_{\text{Ref}}(z) + \dot{x}_{r \text{ Ref}}(z), \tag{20}$$

where x_t , Δx_{Ref} and $x_{r \text{ Ref}}$ are discrete signals.

From (11), (12), (15), (19) and (20), and with the same procedure used in Proposition 2, the error expression and the null error condition are obtained as:

$$e(z) = \frac{(\ddot{x}_{r \text{ Ref}}(z) + N(z)((1 - z^{-d})\dot{x}_{r \text{ Ref}}(z) - z^{-d}\Delta\dot{x}_{\text{Ref}}(z)))}{\text{Der}^2(z) + N(z)\text{Der}(z) + z^{-d}C(z)N(z)} \tag{21}$$

$$e(\infty) = 0 \quad \text{iff} \quad \Delta\dot{x}_{\text{Ref}}(\infty) = 0 \quad \text{and} \quad \ddot{x}_t(\infty) = 0. \tag{22}$$

Note that, (22) is more restrictive than (18).

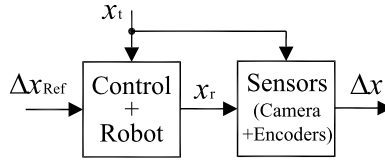


Figure 4. ‘Black box’ representation of the system.

Proposition 4. For the control scheme proposed in Ref. [13], where a PD kinematic controller with no feedforward is used, the steady-state error is null iff the first-order time-derivative of the robot reference $x_{r\text{ Ref}}$ (i.e., the target position x_t minus the vision system reference Δx_{Ref}) is null.

Proof. Expressions (11) and (12) are also valid for this proposition. In this case the kinematic controller is given by:

$$\dot{x}_{r\text{ Cont}}(z) = K_1 e(z) + K_2 \dot{e}(z), \tag{23}$$

where $x_{r\text{ Cont}}$ and e are discrete signals.

The error expression and the null error condition are obtained from (11), (12), (15) and (23) as:

$$e(z) = \frac{\ddot{x}_{r\text{ Ref}}(z) + N(z)\dot{x}_{r\text{ Ref}}(z)}{\text{Der}^2(z) + N(z)((K_2 + 1)\text{Der}(z) + K_1)} \tag{24}$$

$$e(\infty) = 0 \quad \text{iff} \quad \dot{x}_{r\text{ Ref}}(\infty) = 0, \tag{25}$$

where (25) is one order more restrictive than (18).

3.3. Transfer Functions

Figure 4 shows a ‘black box’ representation of the control scheme of Figs 1 and 2. In particular, there are two inputs, the vision system reference Δx_{Ref} (an arbitrary modifiable signal, i.e., the reference) and the target position x_t (a kind of perturbation), and one output, the vision system output Δx . Therefore, the output signal Δx is related with the input signals, Δx_{Ref} and x_t , through two transfer functions, $F_1(z)$ and $F_2(z)$, i.e.:

$$\Delta x(z) = F_1(z)\Delta x_{\text{Ref}}(z) + F_2(z)x_t(z). \tag{26}$$

These transfer functions can be easily obtained for the control scheme of Fig. 1 [17] and Fig. 2 (referred with subscripts c and p, respectively) using the block diagram simplification method:

$$\begin{aligned} F_{1p}(z) &= \frac{R(z)(C(z) + \text{Der}(z))}{1 + R(z)C(z)} \\ F_{2p}(z) &= \frac{1 + R(z)(C(z)(1 - E(z)) - \text{Der}(z))}{1 + R(z)C(z)} \end{aligned} \tag{27}$$

$$F_{1c}(z) = \frac{R(z)C(z)}{1 + z^{-d}R(z)C(z)} \quad F_{2c}(z) = \frac{1 - z^{-d}R(z)E(z)Der(z)}{1 + z^{-d}R(z)C(z)}. \quad (28)$$

Both control schemes can be compared computing the previous transfer functions under the discrete ideal framework (i.e., $E(z) = 1$ and $R(z) = \text{Int}(z) = T_h/(z - 1)$):

$$F_{1p \text{ ideal}}(z) = \frac{1 + C(z) \text{Int}(z)}{1 + C(z) \text{Int}(z)} = 1 \quad F_{2p \text{ ideal}}(z) = \frac{1 - 1}{1 + C(z) \text{Int}(z)} = 0 \quad (29)$$

$$F_{1c \text{ ideal}}(z) = \frac{C(z) \text{Int}(z)}{1 + z^{-d}C(z) \text{Int}(z)} \quad F_{2c \text{ ideal}}(z) = \frac{(1 - z^{-d}E(z))}{1 + z^{-d}C(z) \text{Int}(z)}. \quad (30)$$

Therefore, for the proposed approach the output is always perfect (assuming that the control action value respects the actuator limits and that the initial error is null) and there is a full disturbance rejection, i.e., $F_{2p \text{ ideal}} = 0$. Meanwhile, for the scheme proposed in Ref. [17] (Fig. 1) this does not happen, although ideal conditions are considered. This scheme uses a two-step filter estimation (Fig. 1) with no kind of prediction and can be improved if the estimator predicts d samples ahead of time. With this modification, the second elements of (28) and (30) become:

$$F_{2cm}(z) = \frac{1 - R(z)E(z)Der(z)}{1 + z^{-d}R(z)C(z)} \quad F_{2cm \text{ ideal}}(z) = 0, \quad (31)$$

and, therefore, a full disturbance rejection is obtained, although the output is not perfect like in (29).

4. Simulation

Several simulations are designed in this section to illustrate the propositions of the previous section and the stability limits. They consider the typical vision system delay d of 2 (see Section 2), which will be also used in the next section. Moreover, a proportional controller, i.e., $C(z) = K$, will be assumed hereinafter. The results obtained for the control scheme proposed by Corke and Good [17] (Fig. 1) are plotted in thick dashed lines and those for the one proposed in this research (Fig. 2) in thick solid lines.

Figure 5 shows several simulation results for different values of $\{K, T_h\}$ under the discrete kinematic framework when the target position varies sinusoidally with time. For the first two simulations (Fig. 5a and 5b), the error signals converge to zero, although this convergence is more exponential as long as the kinematic sample time T_h tends to zero (Fig. 5b), which confirms Proposition 1. Moreover, the behaviour of the control scheme proposed by Corke and Good in [17] is more oscillating than the control scheme proposed in this research and achieves instability before, i.e., for a lower value of $K \cdot T_h$, i.e., for 0.618 (Fig. 5c) instead of 2 (Fig. 5d).

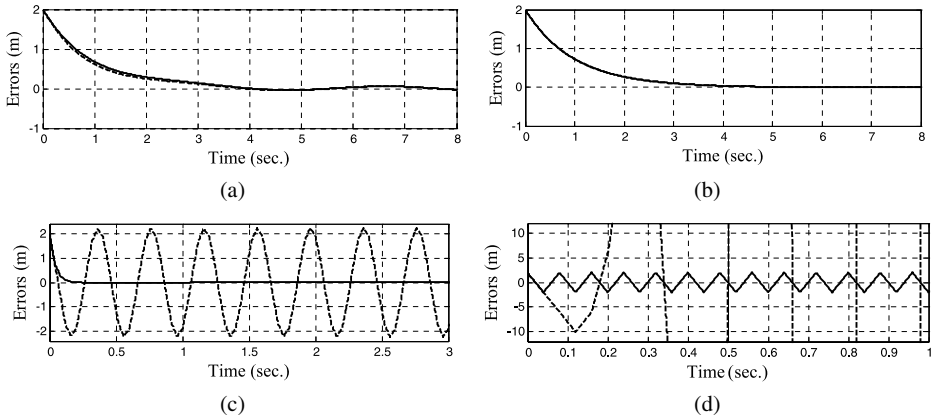


Figure 5. Simulation for $\{R(z) = \text{Int}(z), x_r(t) = 2 \cdot \cos(t \cdot \pi/2) \text{ m}, d = 2, \Delta x_{\text{Ref}} = 0, x_r(0) = 0\}$. (a) $T_h = 0.04 \text{ s}, K = 1$. (b) $T_h = 0.002 \text{ s}, K = 1$. (c) $T_h = 0.04 \text{ s}, K = K_c \text{ ideal max} \approx 0.618/T_h = 15.46$. (d) $T_h = 0.04 \text{ s}, K = K_p \text{ ideal max} = 2/T_h = 50$.

Next, several simulations are developed considering the general case of a dynamic framework. Note that the sample time of a discrete control should be small enough compared to the dynamics of the closed loop transfer function. In other words, it makes no sense to speed up a discrete control if the sample time is not going to effectively reconstruct the signals (Shannon–Nyquist sampling theorem) and, therefore, give rise to instability. In this sense, we can easily obtain from the denominator of (14) or (29) that under the discrete kinematic framework ($R(z) = \text{Int}(z) = T_h/(z - 1)$) the maximum gain K of the kinematic controller of Fig. 2 that makes the system stable is:

$$K_p \text{ ideal max} = 2/T_h, \tag{32}$$

and, therefore, the following relationship, between the kinematic loop gain (i.e., kinematic loop dynamics) and kinematic loop sample time T_h , will be considered:

$$K = K_p = K_c = K_p \text{ ideal max}/5 = 0.4/T_h. \tag{33}$$

In the same way, the low-level sample time T_s will be implicitly considered when assuming the dynamics of the low-level robot transfer function $R_{T_s}(z)$. The following relationship will be used:

$$T_s = t_{st \ s}(99\%)/20, \tag{34}$$

where $t_{st \ s}$ is the setting time given by $R_{T_s}(z) \text{ Der}(z)$. The low-level robot transfer function $R_{T_s}(z)$ is one integrator multiplied by a transfer function with unit static gain. This transfer function will be considered, for example, as a second-order system with 20% of overshoot and setting time $t_{st \ s}$ (34):

$$R_{T_s}(z) = \frac{T_s}{z - 1} \cdot \frac{0.125z + 0.1057}{z^2 - 1.3758z + 0.6065}. \tag{35}$$

The global robot behavior $R(z)$ in Fig. 3 is obtained from (35) for a specific value of the sample time ratio $n = T_h/T_s$, e.g., with the command `d2d` in Matlab.

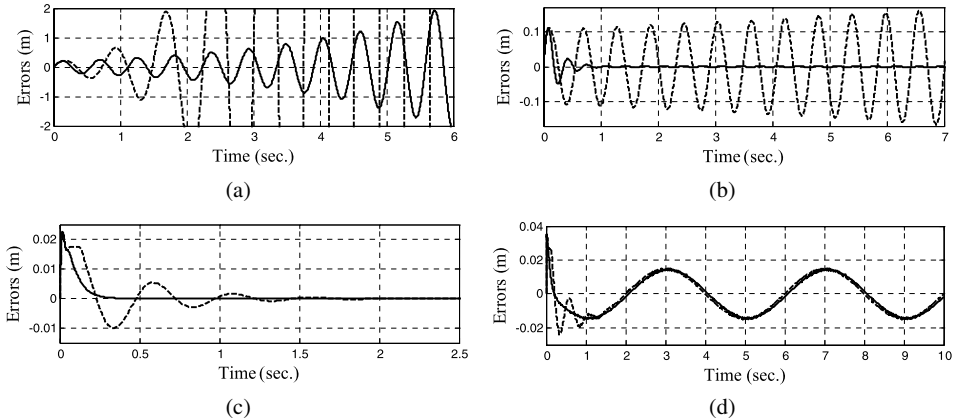


Figure 6. Simulations for $\{(33), (34), (35)\}$ and $\{d = 2; \Delta x_{Ref} = 0, x_r(0) = 0, T_h = 0.04 \text{ s}, T_s = T_h/n\}$. (a) $x_t(t) = 2 \cdot t \text{ m}, n = 1$. (b) $x_t(t) = 2 \cdot t \text{ m}, n = 2$. (c) $x_t(t) = 2 \cdot t \text{ m}, n = 10$. (d) $x_t(t) = 2 \cdot \sin(t \cdot \pi/2) \text{ m}, n = 10$.

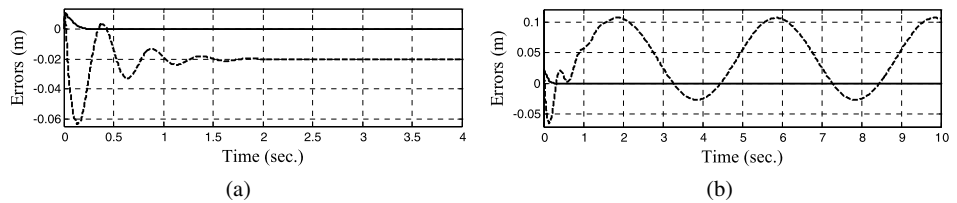


Figure 7. Simulations for $\{(33), (34), (35)\}$ and $\{d = 2, x_r(0) = 0, T_h = 0.04 \text{ s}, T_s = T_h/n, n = 10\}$. (a) $x_t(t) = 2 \cdot t \text{ m}, \Delta x_{Ref} = t \text{ m}$. (b) $x_t(t) = 2 \cdot \sin(t \cdot \pi/2) \text{ m}, \Delta x_{Ref} = 2 \cdot \sin(t \cdot \pi/2) - 2 \cdot t \text{ m}$.

Therefore, the simulations of Figs 6 and 7 have been carried out for the dynamic framework (35), and the relationships (33) and (34) between the inner and outer loop dynamics and the corresponding loop sample time. In particular, Fig. 6 shows that the behaviour of the control scheme proposed by Corke and Good [17] is more oscillating than the control scheme proposed in this research. In fact, for $n = 1$ and $n = 2$ Corke and Good’s control scheme is unstable (Fig. 6a and 6b); meanwhile, the control scheme proposed here is unstable only for $n = 1$ (Fig. 6a), which agrees with the second advantage pointed out in the next section. Moreover, the velocity error (the robot reference is $x_{r \text{ Ref}} = x_t - \Delta x_{Ref}$) is null if the system is stable (Fig. 6c). Meanwhile, the acceleration error is non-zero because the steady-state error is non-null when the target position is a sinusoidal wave (Fig. 6d). These simulation results agree with Proposition 2.

To illustrate the differences between (18) and (22), i.e., between Propositions 2 and 3, the simulations of Fig. 7 have been developed. In particular, it can be seen in both Fig. 7a $\{\ddot{x}_t = \Delta \ddot{x}_{Ref} = 0, \Delta \dot{x}_{Ref} \neq 0\}$ and Fig. 7b $\{(\ddot{x}_t - \Delta \ddot{x}_{Ref}) = 0, \ddot{x}_t \neq 0\}$ that the steady-state error is null for the control scheme proposed in this research and non-zero for that proposed in Ref. [17].

In short, the proposed control scheme has the following two limitations:

- Under the kinematic framework (e.g., the dynamics of the low-level dynamic loop is negligible) the kinematic loop dynamics, which are given by controller $C(z)$, have to be slow enough to effectively reconstruct the signals with sample time T_h , so that instability is avoided (Fig. 5).
- The dynamics given by the low-level dynamic loop must be faster than that of the kinematic loop to guarantee stability, assuming that both loops are separately stable (Fig. 6 and Figs 11 and 12 in Section 6.1).

However, the previous two limitations are also qualitatively present in the control scheme developed by Corke and Good [17], which is quantitatively more limited with respect to the previous two points, i.e., instability arises before (see Figs 5 and 6).

5. Other Advantages of the Proposed Scheme

The control scheme of Fig. 2 (proposed in this research) improves that of Fig. 1 (proposed by Corke and Good [17]) basically in three ways:

(i) Figure 2 considers the general case of a non-constant vision system reference Δx_{Ref} by including it in the time-derivative feedforward. This gives rise to a less restrictive condition for zero steady-state error (see Propositions 2 and 3).

(ii) The time-derivative feedforward of the robot reference is obtained in Fig. 2 for the current step time, since the estimator $E(z)$ predicts d samples ahead of time. Meanwhile, the control scheme proposed by Corke and Good (see Fig. 1) used a two-step estimator $E(z)$ with no kind of prediction and, therefore, the delay d is ignored.

(iii) The position error signal, which is the input to the controller block $C(z)$, is obtained in Fig. 2 for the current step time. Evidently, the error signal used in Fig. 1, which is directly computed with the delayed vision system output, will have more error than if the estimator/predictor output had been used (in the worst case it will be the same if the uncertainty of the process model used in the estimator is very high).

Other advantages of the proposed control scheme are:

(i) It allows working with one high (slow) sample time T_h , given by the outer loop sensors (e.g., vision system), and another small (fast) sample time T_s (or even continuously), given by the inner loop sensors (e.g., encoders).

(ii) It is easy to guarantee the stability if the inner loop is much faster than the outer loop and both loops are separately stable (see Fig. 6).

(iii) The steady-state error would be null for any continuous robot reference if its time-derivative varies smoothly with respect to the dynamics of the inner loop as long as the sample time of the outer loop can effectively reconstruct the first time-derivative of the reference (Nyquist–Shannon sampling theorem).

6. Experimental Results

6.1. Configuration Set-up

For the experimentation, the elements of Fig. 8 have been used. A turntable, which consists of a disk with a circle painted (in another color) near the border, provides a visual characteristic. This turntable/disk is moved by an asynchronous electrical motor (Siemens 3~Mot. 1LA7060-4AB60) driven by a Micromaster 440 (6SE6440-2UC13-7AA0). In front of the turntable, the end-effector of a 3-d.o.f. Cartesian robot is placed with a uEYE USB2.0 camera (UI-2310-C, 640h-480v configured for 25 frames/s, i.e., $T_h = 40$ ms) (Fig. 9a). The image plane of the camera is parallel to the turntable. Two d.o.f. of the Cartesian robot are used to follow the visual characteristic in a tracking task. This visual characteristic is ac-

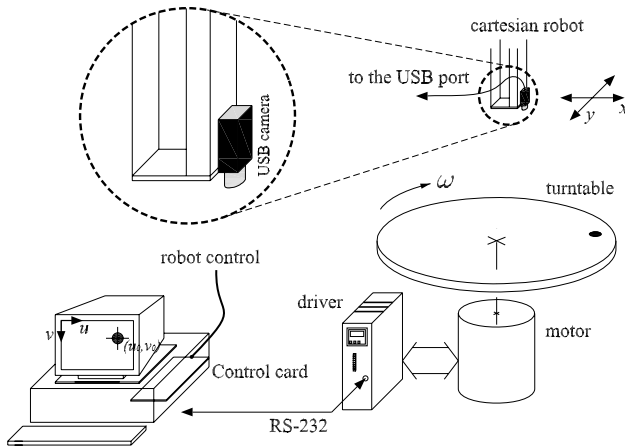


Figure 8. Schematic representation of the elements used for visual servoing experimental results.



(a)



(b)

Figure 9. Cartesian robot developed at the Miguel Hernandez University for visual servoing. (a) Configuration set-up: turntable and camera. (b) Global view of the Cartesian robot.

quired by the camera and processed ($d = 2$) to obtain its ‘center of mass’ using a color-based blob detection algorithm (OpenCV). The Cartesian robot (Fig. 9b), whose modeling equations can be found in Ref. [23], is 1.7 m high, 2.3 m long and 1.4 m wide, and is moved by three AC servo-motors (synchronous) manufactured by Siemens (1FK6 series) with the following characteristics: for X - and Y -axis: torque at $\omega = 0$: 16 Nm, nominal torque: 10.5 Nm; for Z -axis: torque at $\omega = 0$: 11 Nm, nominal torque: 6 Nm.

The following control algorithms have been implemented in the PC: the low-level joint velocity controllers of Fig. 3; the control scheme of Fig. 2 (proposed here for visual servoing) or (alternatively) the control scheme of Fig. 1 [17] and the prediction filter described in Ref. [20], which is based on a fuzzy system that combines different filters: linear interpolation, Kalman filter, $\alpha\beta(\gamma)$ filter, etc. The ‘Control card’ used (Fig. 8) is the NI PCI 6024E. The ‘start’ signal is provided by a PC, the turntable begins the movement and the robot is controlled depending on the visual feature position to maintain it in the center of the image plane.

6.2. Turntable Experiments

Figure 10 depicts the low-level joint control for the X - and Y -axis of the Cartesian robot. In both cases the joint velocity reference is 0.7 m/s. It can be seen that the setting time (99% criteria) is approximately 0.33 s for the X -axis, i.e., $t_{st \text{ din } x} = 0.33$ s, and 0.25 s for the Y -axis, i.e., $t_{st \text{ din } y} = 0.25$ s. Moreover, a low-pass filter will be applied to the measured velocities in the visual servoing control scheme in order to avoid noisy signals (Fig. 10) and a saturation of 1.5 m/s is considered in the control schemes for the control signal $\dot{x}_{r \text{ Cont}}$ that commands the robot. Using the turntable, two sinusoidal movements are generated in the X - and Y -axis. The combination of these sinusoidal movements is the circular trajectory of the visual characteristic to be tracked ($\Delta x_{\text{Ref}} = 0$). The frequency of these signals depends on the angular velocity ω of the turntable, in this case $\omega = 2 \cdot \pi / 2.5$ rad/s, their amplitude value is 0.177 m (distance from the black circle with respect to the axle of the turntable) and their offset value depends on the robot initial position. The

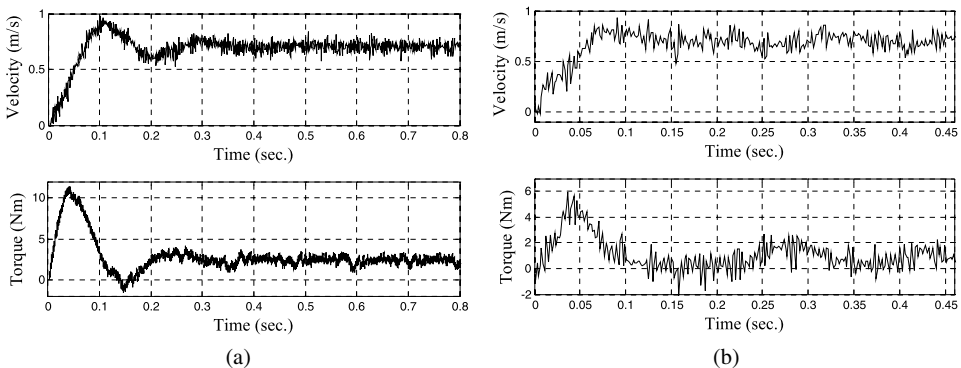


Figure 10. Low-level velocity control. (a) X -axis. (b) Y -axis.

resolution in the image plane, which is given by the camera resolution and the distance from the turntable to the image plane, is 400 pixels/m.

Figure 11 shows the comparative experimental results obtained for the control scheme proposed here (Fig. 2) and for the one proposed by Corke (Fig. 1): the thin solid line is the estimation of the target position/velocity, the thick solid line is the robot position/velocity obtained using the control scheme proposed in this research (Fig. 2) and the thick dashed line is the robot position/velocity obtained using the control scheme of Corke (Fig. 1). The kinematic loop gain used for the X -axis is $K_x = 3$ and for the Y -axis is $K_y = 4$. These gains imply that the setting times of the kinematic loops are $t_{st \text{ kin } x} \approx 5/3 = 1.66 = 5 \cdot t_{st \text{ din } x}$ and $t_{st \text{ kin } y} \approx 5/4 = 1.25 = 5 \cdot t_{st \text{ din } y}$, i.e., the kinematic loops are significantly slower than the dynamic loops and, therefore, stability is guaranteed. In fact, it can be seen in Fig. 11 that both control schemes have a very similar and satisfactory behavior, because the vision system delay $d = 2$ (0.08 s) is negligible with respect to the kinematic loop dynamics. Another experimental comparison is shown in Fig. 12 (the

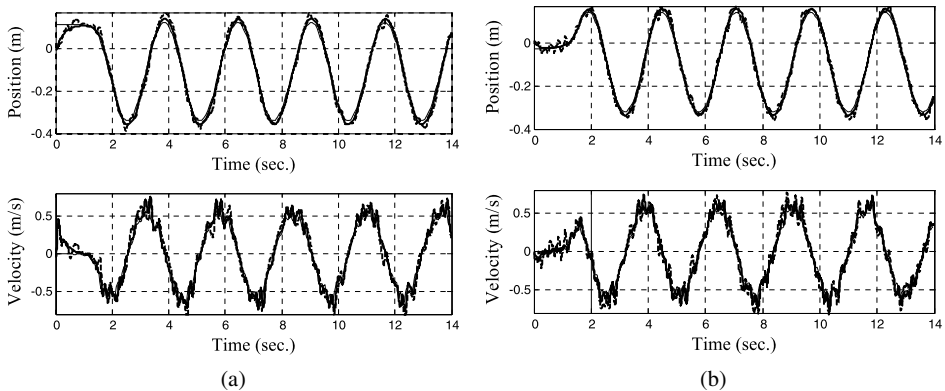


Figure 11. Experimental comparison using the turntable. (a) X -coordinate. (b) Y -coordinate.

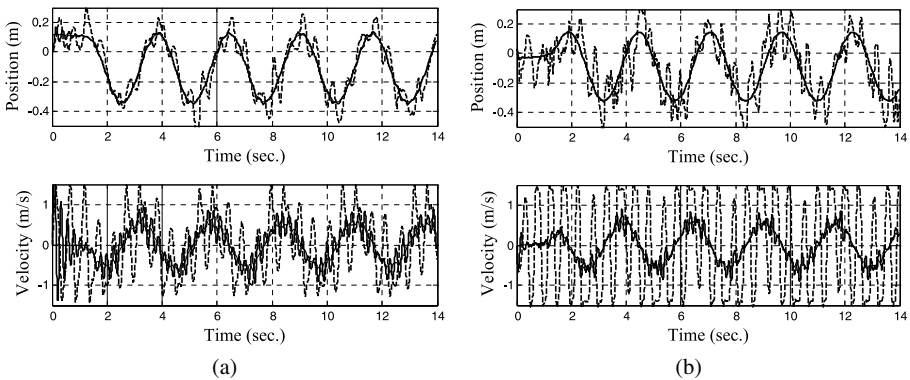


Figure 12. Experimental comparison for a more critical kinematic loop dynamics. (a) X -coordinate. (b) Y -coordinate.

variables representation is the same one that in Fig. 11) for the kinematic loop gains $K_x = 15$ and $K_y = 20$. In this second case the setting times of the kinematic loops are approximately equal to the ones of the dynamic loops, i.e., $t_{st \text{ kin } x} \approx t_{st \text{ din } x}$ and $t_{st \text{ kin } y} \approx t_{st \text{ din } y}$, and, therefore, stability is not guaranteed (Fig. 6b). In fact, for these more critical gains, Fig. 12 shows that the approach proposed here has clearly better performance than the one proposed by Corke, which is virtually unstable since the robot velocities change between the saturation values (± 1.5 m/s) and the position error is permanently significant.

It is interesting to remark that, from the experimental point of view, the estimation accuracy of the predictor and the accuracy of the robot model used are very important in practice. In fact, another (practical) limitation of the control scheme must be added to the two indicated at the end of Section 4:

- The inaccuracy of the predictor and the unmodeled dynamics (e.g., nonlinearities, sensor dynamics, low-level control dynamics. . .) have to be small enough in order to avoid instability.

This means that the limit values for stability (e.g., maximum controller gain, maximum vision system delay, etc.) are lower in real experimentation than in simulation. For example, under the kinematic framework the proposed control scheme (Fig. 2) should never be unstable, regardless of the value of the vision system delay d . Nevertheless, the inaccuracy of the predictor and the unmodeled dynamics (e.g., in this case the low-level dynamic loop) cause instability for a certain value of the vision system delay. In particular, Fig. 13 shows the experimental result for the control scheme of Fig. 2 with $K_x = 3$, $K_y = 4$ and $d = 14$. Therefore, the global system becomes unstable for approximately a vision system delay of 0.56 s, i.e., around the fourth part of the turntable period.

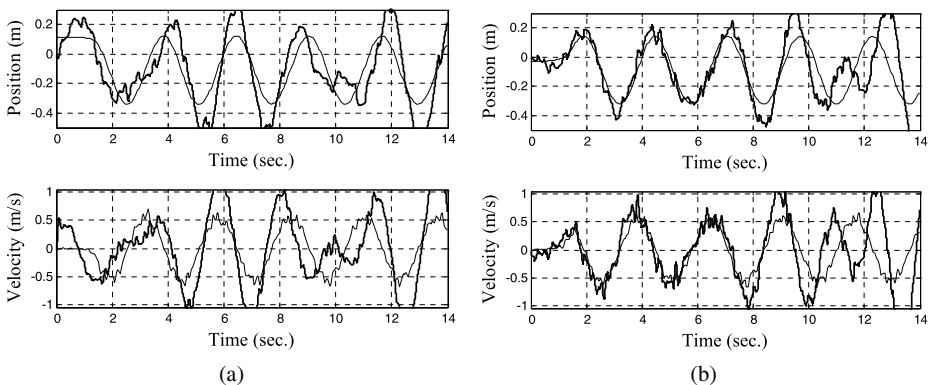


Figure 13. Experimental result for a vision system delay of 0.56 s ($d = 14$). (a) X-coordinate. (b) Y-coordinate.

7. Conclusions

The major contribution of this research is the proposed control scheme (Fig. 2) for visual servoing that explicitly takes into account the vision system delay by properly including a predictor block, i.e., an estimator that predicts several samples ahead of time, in the scheme. In particular, it has been shown analytically that this new scheme improves in several ways (conditions for zero steady-state error; transfer functions and condition for exponential error convergence, both under the ideal framework; additional advantages, etc.) the performance of the control scheme proposed in Refs [17, 18], which has been widely accepted by the international scientific community and that has been used in several subsequent well-known research [19] and conference contributions [20]. The benefits and limitations of the proposed approach have been comparatively illustrated by simulations and by experiments using a 3-d.o.f. Cartesian robot. In this sense, the experimental results validate both analytical and simulation results.

Acknowledgements

This work was supported in part by the Generalitat Valenciana (Research Project GVPRE/2008/168) and Bancaja Savings Bank. The authors would like to thank the anonymous reviewers for their valuable comments that helped us to improve the quality of this paper.

References

1. Y. Shirai and H. Inoue, Guiding a robot by visual feedback assembling tasks, *Pattern Recognit.* **5**, 99–108 (1973).
2. P. Corke, Visual control of robot manipulators — a review, in: *Visual Servoing*, K. Hashimoto (Ed.), pp. 1–31. World Scientific, Singapore (1993).
3. Special issue on visual servoing, *IEEE Trans. Robotics Automat.* **12** (5) (1996).
4. Special issue on visual servoing, *IEEE Robotics Automat. Mag.* **5** (4) (1998).
5. Special issue on visual servoing, *Int. J. Robotics Res.* **22** (10/11) (2003).
6. S. Hutchinson, G. Hager and P. Corke, A tutorial on visual servo control, *IEEE Trans. Robotics Automat.* **12**, 651–668 (1996).
7. *International Online Course on Visual Servoing Techniques*, 17 October–15 December (2005). Available at: <http://www.robot.uji.es/IOCoViST/>
8. F. Chaumette and S. Hutchinson, Visual servo control — part I: basic approaches, *IEEE Robotics Automat. Mag.* **13**, 82–90 (2006).
9. F. Chaumette and S. Hutchinson, Visual servo control — part II: advanced approaches, *IEEE Robotics Automat. Mag.* **14**, 109–118 (2007).
10. E. Malis, F. Chaumette and S. Boudet, 2-1/2-D Visual Servoing, *IEEE Trans. Robotics Automat.* **15**, 238–250 (1999).
11. L. E. Weiss, A. C. Sanderson and C. P. Neuman, Dynamic sensor-based control of robots with visual feedback, *IEEE Trans. Robotics Automat.* **3**, 404–417 (1987).
12. B. Espiau, F. Chaumette and P. Rives, A new approach to visual servoing in robotics, *IEEE Trans. Robotics Automat.* **8**, 313–326 (1992).

13. W. J. Wilson, C. C. W. Hulls and G. S. Bell, Relative end-effector control using Cartesian position-based visual servoing, *IEEE Trans. Robotics Automat.* **12**, 684–696 (1996).
14. K. Hashimoto, T. Ebine and H. Kimura, Visual servoing with hand-eye manipulator — optimal control approach, *IEEE Trans. Robotics Automat.* **12**, 766–774 (1996).
15. C. Monroy, R. Kelly, M. Artega and E. Bugarin, Remote visual servoing of a robot manipulator via Internet2, *J. Intell. Robotic Syst.* **49**, 171–187 (2007).
16. N. Garcia, E. Malis, R. Aracil and C. Perez, Continuous visual servoing despite the changes of visibility in image features, *IEEE Trans. Robotics* **21**, 1214–1220 (2005).
17. P. I. Corke and M. C. Good, Dynamic effects in visual closed-loop systems, *IEEE Trans. Robotics Automat.* **12**, 671–683 (1996).
18. P. I. Corke, *Visual Control of Robots: High Performance Visual Servoing*, *Mechatronics*. Research Studies Press, New York, NY (1996).
19. S. Chroust and M. Vincze, Improvement of the prediction quality for visual servoing with a switching Kalman filter, *Int. J. Robotics Res.* **22**, 905–922 (2003).
20. C. Perez, N. Garcia, J. M. Sabater, J. M. Azorin, O. Reinoso and L. Gracia, Improvement of the visual servoing task with a new trajectory predictor the fuzzy Kalman filter, in: *Proc. Int. Conf. on Informatics in Control, Automation and Robotics*, Angers, pp. 133–140 (2007).
21. M. T. Tommiska, Area-efficient implementation of a fast square root algorithm, in: *Proc. IEEE Int. Caracas Conf. on Devices, Circuits and Systems*, Cancun, pp. S18-1–S18-4 (2000).
22. M. Vincze and G. D. Hager, *Robust Vision for Vision-Based Control of Motion*. Wiley–IEEE Press, Piscataway, NJ (2000).
23. C. Perez, O. Reinoso, N. Garcia, J. M. Sabater and L. Gracia, Modelling of a direct drive industrial robot, *Trans. Eng. Comput. Technol.* **17**, 329–334 (2006).

About the Authors



Carlos Perez-Vidal received the BS in Industrial Engineering (1998), MS in Control Engineering (2000) and PhD in Industrial Technologies (2008) from the Technical University of Valencia and Miguel Hernandez University, Spain. He is an Associate Professor of Control and Systems Engineering at Miguel Hernandez University (Spain) and researcher at the Systems Engineering and Automatic Control Division. His current research interests are robotics, direct visual servoing, automation and control. He has been active since 2001 in R&D within several projects on advanced robotics and vision.



Luis Gracia received the BS in Electronic Engineering, MS in Control Systems Engineering and PhD in Automation and Industrial Computer Science from the Technical University of Valencia (UPV), Spain, in 1998, 2000 and 2006, respectively. He held a PhD Fellowship for 1 year at the Department of Systems Engineering and Control, UPV, where he was employed as an Assistant Professor in 2002, becoming Associate Professor in 2007. His research interests include wheeled mobile robots, robotic manipulators, system modeling and control.



Nicolas Garcia received the BS in Electronic Engineering from the Technical University of Valencia (1992), MS in Control Engineering from the University of Murcia (1996), Master in Design, Robotics and Industrial Automation from the University of Murcia (1996–1997) and a PhD in Control Engineering from the Miguel Hernandez University of Elche. He is an Associate Professor of Control and Systems Engineering at Miguel Hernández University (Spain), and researcher at the Virtual Reality and Robotics Lab. His current research interests are robotics, visual servoing, telerobotics and human–robot interaction.



Enric Cervera completed his PhD in Computer Science, in 1997, and he achieved a position as Associate Professor of Computer Science and Artificial Intelligence at Jaume I University, in 1999. Since then, he has led several research projects funded by the Spanish Government and is currently collaborating in a FP6 European project. He has published several research articles in international journals, conference proceedings and book chapters. He has served as reviewer of several top international journals and on the program committee of several international conferences (IROS'04–06, IASTED). His present research deals with collaborative approaches to robotics and sensor-based control.