ijrr

The International Journal of
Robotics Research
0(0) 1–18

DOI: 10.1177/0278364911436082
ijr.sagepub.com
SAGE

Dynamic visual servoing from sequential regions of interest acquisition

Redwan Dahmouche[1], Nicolas Andreff[2], Youcef Mezouar[1], Philippe Martinet[1] and Omar Ait-Aider[1]

Abstract

One of the main drawbacks of vision-based control that remains unsolved is the poor dynamic performances caused by the low acquisition frequency of the vision systems and the time latency due to processing. We propose in this paper to face the challenge of designing a high-performance dynamic visual servo control scheme. Two versatile control laws are developed in this paper: a position-based dynamic visual servoing and an image-based dynamic visual servoing. Both control laws are designed to compute the control torques exclusively from a sequential acquisition of regions of interest containing the visual features to achieve an accurate trajectory tracking. The presented experiments on vision-based dynamic control of a high-speed parallel robot show that the proposed control schemes can perform better than joint-based computed torque control.

Keywords

Visual servoing, dynamics, high-speed vision, visual tracking, parallel robots

1. Introduction

1.1. Visual servo control challenges

The integration of vision in the control loop of robotic systems allows for relative positioning in a dynamic environment. Therefore, uncertainties in the initial position of the piece on which the robot has to work or in the robot kinematic model do not really affect control performances as it does in joint-based control. However, even if vision-based control has many advantages compared with model-based control, it still has some drawbacks. The first is the poor control performances which represents a significant obstacle to be an effective industrial solution (Kragic and Vincze 2009). This drawback is closely related to the low acquisition frequency of standard cameras which is about 50–60 Hz. To adapt visual measures to robot control, the 'look-and-move' control hierarchy was designed (Weiss et al. 1987). Nowadays, this is the most widely used vision-based control architecture.

The first vision-based control architecture that was developed is the 'look-then-move' (Corke and Good 1996; Zhang and Pstrowski 1999; Vincze 2000) (originally named 'static look-and-move') (Tani et al. 1977; Birk et al. 1979). In the late 1970s, the computation time was so long[1] that it was necessary to compute the pose error while the robot is stopped. Since the robot motion is jerky, the performances of such a control law are poor.

As the computation performances of calculators increased, another visual servoing architecture, known as 'look-and-move' (Chaumette et al. 1991; Hutchinson et al. 1996; Vincze 2000) (originally named 'dynamic look-and-move' (Weiss et al. 1987)), was proposed to improve the control performances and has become the classical visual servoing architecture simply called visual servoing (Hutchinson et al. 1996; Chaumette and Hutchinson 2006). It is now the most popular architecture and it is referred in the literature simply as visual servoing. The 'look-and-move' control hierarchy is composed of two nested loops running in real-time, usually at different frequencies. The high-level loop (at camera frequency) computes the kinematic control vector to drive the robot to the desired configuration. Since the controller has to send torques to the actuators and not velocities, the role

[1]LASMEA - CNRS - Université Blaise Pascal, Aubière, France
[2]FEMTO-ST - CNRS, Université de Franche Comté, Besançon, France

Corresponding author:
Redwan Dahmouche, LASMEA - CNRS - Université Blaise Pascal, Campus des Cézeaux, 24 avenue des Landais, 63175 Aubière, France.
Email: redwan.dahmouche@univ-bpclermont.fr

of the low-level loop (at least >100 Hz) is to servo the robot velocity. This is usually done through a conventional controller (typically PID or, slightly better, PD + gravity) exploiting joint sensing.

However, this control architecture does not fit high-performance control because: (1) the latency on the visual measures caused by sensor exposure, image transmission and data processing considerably affects the performance and the stability of the control law (Vincze et al. 2002; Zhang et al. 2003); (2) the first-order exponential convergence, usually chosen in this control scheme, is not the fastest response that can be obtained; (3) the error accumulation of the two nested loops is not the most appropriate control architecture to improve accuracy; (4) choosing velocities as a control vector does not allow the compensation of the robot dynamics which is crucial when controlling high-speed robots. As a consequence, the compensation of a slow vision system by a fast low-level control loop is not the best solution for high-performance vision-based control design.

An alternative control architecture that could improve performances is the 'dynamic visual servo' control (Weiss et al. 1987). Indeed, in this control hierarchy, the visual loop delivers directly the joint torques from visual sensing (i.e. no nested loop) which resolves most of the drawbacks related to the 'look-and-move' control architecture. However, classical visual systems do not fit the typical dynamic control requirements in terms of latency and acquisition frequency (Corke and Good 1996). Moreover, dynamic control requires not only robot configuration estimation (as usual) but also the estimation of the configuration derivative (robot velocity) for system regulation and tracking efficiency improvement. This represents another issue in the context of dynamic visual servo since usually vision does not provide velocity estimation.

1.2. Improving vision-based control performances

To improve the stability and performance of the vision-based control, two main approaches have been proposed. The first consists of reducing the effects of low frequency and latency by designing suitable control laws. The second approach, which is more suitable, is to effectively improve the acquisition and processing performances of the vision system to fulfill dynamic control requirements.

Gangloff and de Mathelin (2003) adopted these two approaches and proposed a predictive control of a six-degree-of-freedom (6-d.o.f.) serial robot based on the 'look-and-move' structure. The high-level loop of the proposed control architecture is about 120 Hz whereas the low-level loop runs at 500 Hz. The idea of the control law is to compensate for the latency of the visual system through prediction. Note that the acquisition frequency (120 Hz) is obtained by reducing the camera field of view (640 × 240 pixels). Reducing the image size even more (256 × 256 pixels) allowed the frequency to be increased up to 500 Hz (Cuvillon et al. 2006). However, cutting down the image size reduces either the camera resolution or field of view which is not permitted in most vision-based applications.

An image-based dynamic control applied to a blimp was proposed by Zhang and Pstrowski (1999). The proposed control scheme is based on a single point in the image. This allows for the decoupling and the analytical expression of the dynamics of the blimp in the image. However, this *ad hoc* expression does not extend to other systems nor to the observation of several image points. Moreover, the latency caused by the measure by vision is not very harmful because the blimp has much slower dynamics than a manipulator.

For pick-and-place applications, an interesting vision-based dynamic control law combines vision measures and joint sensors to compensate for the robot dynamics and compute the control vector (Kelly et al. 2000). However, this control law also exploits a single point and assumes prior knowledge of its depth. In addition, even if the control hierarchy is of the form 'direct-feedback' it remains, nevertheless, a multi-frequency control scheme. Thus, the position correction is obtained by vision at a frequency of 50 Hz while the velocity is corrected from joint measurements which are updated at 400 Hz. The local stability of the control loop and experiments have been demonstrated only on a 2-d.o.f. servo from a point fixed at a constant and known depth from the camera.

Although vision sensors are able to acquire images at very high frequencies (Etoh et al. 2003). The acquisition frequency of vision systems is typically limited by the communication interface bandwidth. To avoid this bottleneck, several solutions were proposed in the literature. An interesting one consists of embedding the processing close to the sensor as proposed by Ishii et al. (1996) where a 1 kHz acquisition system was designed. However, the image resolution was considerably low (128 × 128). In addition, embedding the processing close to the sensor may not be convenient since it requires dedicated hardware and software development which makes the system more complex and less scalable. This is why 'smart-cameras' that integrate processing are usually designed to specific applications. The most appropriate solution to increase the acquisition frequency is probably to use cameras that dynamically select and transmit only the regions of interest (ROIs) in the image that contain the relevant visual information (Ulrich et al. 2004; Paccot et al. 2008; Dahmouche et al. 2009).

1.3. Vision-based dynamic control of parallel robots

Vision is particularly relevant to the control of a large class of robots called 'parallel kinematic machines' (PKMs) (Paccot et al. 2009). PKMs (or parallel robots) are closed chain mechanisms with a platform (or end-effector) connected to the robot by several kinematic chains (Merlet

2009). One advantage of such mechanical structures is that, unlike serial robots, the actuators can be rigidly fixed to the robot frame. Releasing the robot from the burden of the actuators can then significantly reduce the carried weight and thus improve the dynamic performances of the system. High accelerations, never achieved by serial robots, can then be reached (Nabat et al. 2005).

It was shown by Dasgupta and Choudhury (1999), Khalil and Ibrahim (2004), Callegari et al. (2006), and Paccot et al. (2009) that the end-effector pose is more appropriate for parallel robot control than joint configuration. Indeed, the joint configuration of a PKM does not fully describe the robot configuration since it was shown that a single joint configuration of the Gough–Stewart platform (a well-known parallel robot) may correspond to 40 real configurations (Dietmaier 1998). Owing to this particularity, the forward kinematic model of this class of robots is usually not trivial to compute (a huge body of literature is devoted to this problem, which can be entered through (Merlet 2004)). In addition, it is subject, in the same way as serial robots, to modeling and numerical errors (Boye and Pritschow 2005). Indeed, the reliability of the end-effector pose estimation depends on to the completeness of the robot geometry modeling and to the identification accuracy of the model (calibration) whereas the platform configuration usually fully describes the robot configuration (without ambiguity). Vision appears then to be a relevant solution for parallel robot control since it allows the platform pose to be sensed.

Paccot et al. (2008) obtained a first result on position-based dynamic visual servoing (PBDVS) of a parallel robot. To fulfill the dynamic control requirements the acquisition frequency was increased by grabbing a single configurable (position and size) ROI in the image containing all of the visual features. This acquisition method was available from an off-the-shelf CMOS camera and allowed for a 500 Hz pose estimation using Dementhon's algorithm (Paccot et al. 2008). However, even if this work demonstrated the feasibility of the control law, the obtained accuracy was not fully satisfactory because of the noise in velocity estimation which was obtained by the numerical differentiation of the pose.

1.4. Contributions and organization of the paper

To the best of the authors' knowledge, no generic 6-d.o.f. image-based dynamic visual servoing (IBDVS) has been presented yet. Indeed, as stated above, all image-based dynamic control laws previously proposed were designed for particular applications (Corke and Good 1996; Zhang and Pstrowski 1999; Kelly et al. 2000) where the robotic systems can be controlled from a single point. As a matter of fact, they have not been extended to more complex systems (more features or more d.o.f.). In addition, the obtained performances were limited since the low acquisition frequency of the proposed control laws results in the reduction of the control gains (Corke and Good 1996), the use of joint sensor as a complement to the visual measures (Kelly et al. 2000) or the control of a poorly responsive blimp (Zhang and Pstrowski 1999).

The first contribution of this paper is the development of an efficient 6-d.o.f. IBDVS. This control law uses a high-speed sequential acquisition of ROIs in the images to achieve a high-performance dynamic control where the robot dynamics can be compensated for either from vision (if the robot dynamics can be computed from the end-effector pose and velocity) or from joint configuration and velocity. A second contribution of this paper is the improvement of the PBDVS performances compared with Paccot et al. (2008). This improvement is achieved by adopting a new approach to estimate simultaneously the 3D pose and velocity without any numerical differentiation. Basically, the velocity estimates are obtained from the artifacts in the image generated by the delay between the sequential acquisition of the ROIs (Ait-Aider et al. 2006; Dahmouche et al. 2009). The third contribution is the experimental validation of the methods on a high-speed parallel kinematic manipulator (Orthoglide (Chablat and Wenger 2003)) without any other sensing than vision.

In the context of dynamic control of parallel robots, we have seen in the previous section that it is more relevant to exploit the end-effector pose rather than the joint configuration to design the control laws. The contributions in parallel robot control is twofold. First, the control performances benefit from the intrinsic properties of vision-based control schemes (relative positioning, etc.). Second, the proposed control laws respect the natural representation of parallel robots since the control is achieved through robot end-effector sensing (no need of any joint sensing).

Finally, although the proposed control laws are validated in the context of parallel robots control, their versatile formulation should yield simple adaptation to serial robots. The differences between parallel robots and serial robots control will be pointed out in the paper.

The following section presents the theoretical background of the virtual visual servoing for simultaneous pose and velocity estimation. Section 3 is devoted to the pose and velocity estimation, under a piecewise constant acceleration motion assumption, exploited by the PBDVS. The proposed IBDVS is presented in Section 4 where the requirements for an 'ideal' high-performance IBDVS are first defined and a control law is then proposed to satisfy these requirements. Section 5 presents the experimental setup and some implementation details. Finally, Section 6 shows the experimental results obtained with the proposed control law and a comparison with a classical joint-based computed torque control is discussed.

2. Virtual visual servoing framework for pose and velocity estimation

For the completeness of the paper, let us reformulate here some background results from Dahmouche et al. (2009)

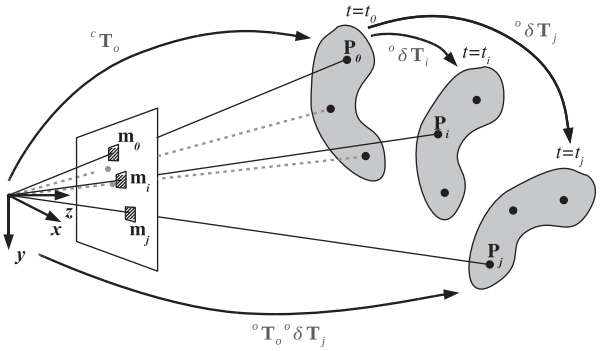

Fig. 1. Projection model of a set of points in the case of sequential acquisition. At the time instant t_0, the relative pose between the object and the robot is ${}^c\mathbf{T}_o$. The relative pose at t_i and t_j are, respectively, ${}^c\mathbf{T}_o\ \delta\mathbf{T}_i$ and ${}^c\mathbf{T}_o\ \delta\mathbf{T}_j$ which means that the projected points do not correspond to the same pose of the object.

that are useful. In the sequential image features acquisition method, a rigid object, abstracted as a set of 3D points, is observed by successively grabbing one single sub-image containing a single visual point at a time (see Figure 1).

The projection model of a set of 3D points $\mathbf{P}_i$ resulting in a visually deformed shape of the object in the image (Ait-Aider et al. 2006; Dahmouche et al. 2008, 2009) is thus given by

$$\forall i = 1 \ldots n \quad \tilde{\mathbf{m}}_i \equiv [\mathbf{K} \mid \mathbf{0}_{3\times 1}]\ {}^c\mathbf{T}_o\ \delta\mathbf{T}_i\ {}^o\tilde{\mathbf{P}}_i \qquad (1)$$

where n is the number of 2D–3D correspondences, ${}^o\tilde{\mathbf{P}}_i$ are the homogeneous coordinates of point $\mathbf{P}_i$ in the object reference frame, $\tilde{\mathbf{m}}_i$ are the homogeneous coordinates of the associated point projection in the camera plane ($\mathbf{m}_i$), $\equiv$ is the projective equality, ${}^c\mathbf{T}_o$ the homogeneous transformation matrix between the object and camera frames at a reference time t_{ref} and $\delta\mathbf{T}_i$ the displacement between t_{ref} and the ith point acquisition time t_i. Finally, $\mathbf{K}$ is the matrix containing the camera intrinsic parameters, whilst lens distortion is not shown here for the sake of clarity but is compensated for.

The specificity of this acquisition method is that the projection model of a rigid object depends on the object pose and velocity. Indeed, the displacement $\delta\mathbf{T}_i$ is nothing but the integration of the object velocity between the reference time t_{ref} and the grabbing time t_i. Then, the estimation method consists essentially in minimizing the reprojection error built upon (1):

$$\min_{{}^c\mathbf{T}_o,\delta\mathbf{T}_i} \frac{1}{2}\sum_{i=1}^{n} \|\pi(\,[\mathbf{K} \mid \mathbf{0}_{3\times 1}]\,{}^c\mathbf{T}_o\delta\mathbf{T}_i\,{}^o\tilde{\mathbf{P}}_i) - \mathbf{m}_i^*\|^2 \qquad (2)$$

where $\pi(\cdot)$ represents the nonlinear formulation of the perspective projection and $\mathbf{m}_i^*$ the desired points position.

As the reprojection error is nonlinear, the optimization problem (2) can be solved using an iterative numerical scheme. One elegant method, taking into account the specific structure of $SE(3)$, is to use the virtual visual servoing paradigm (Marchand and Chaumette 2002; Dahmouche et al. 2009). This can be seen as an iterative scheme where the linearization is done in $se(3)$ rather than in $\mathbf{R}^6$.

The aim of this method is to minimize an error $\mathbf{e}_u$ which can be chosen as the reprojection error (Chaumette and Hutchinson 2006):

$$\mathbf{e}_u = \mathbf{m} - \mathbf{m}^* \qquad (3)$$

where $\mathbf{m}$ and $\mathbf{m}^*$ are two vectors of dimension $2n \times 1$ containing, respectively, the estimated and the desired 2D points positions.

One can obtain an exponential decrease of the error by imposing

$$\dot{\mathbf{e}}_u = -\lambda\ \mathbf{e}_u. \qquad (4)$$

The derivative of the reprojection error with respect to the virtual time u is given by

$$\frac{d\mathbf{e}_u}{du} = \frac{d\mathbf{m}}{du} = \frac{d}{du}\pi([\mathbf{K} \mid \mathbf{0}_{3\times 1}]\ {}^c\mathbf{T}_o(u,t)\,\delta\mathbf{T}_i(u,t)\ {}^o\tilde{\mathbf{P}}_i)\,. \qquad (5)$$

It can be shown that (5) can be rewritten as follows (see the Appendix for details):

$$\frac{d\mathbf{e}_u}{du} = \mathbf{L}\begin{pmatrix}\boldsymbol{\tau}_u\\ \dot{\boldsymbol{\tau}}_u\end{pmatrix} \qquad (6)$$

where $\mathbf{L}$ is a $(2n \times 12)$ matrix which relates the object velocity $\boldsymbol{\tau}_u$ and acceleration $\dot{\boldsymbol{\tau}}_u$ to the image velocity of the set of image points $\mathbf{m}$. The subscript u indicates that the velocity and acceleration twists are virtual, i.e. they do not correspond to twists related to the actual robot but to twists related to the virtual robot (evolving along the virtual time u) to converge to the same state as the real one.

The virtual control vector is finally obtained from (3), (4) and (6) as follows:

$$\begin{pmatrix}\boldsymbol{\tau}_u\\ \dot{\boldsymbol{\tau}}_u\end{pmatrix} = -\lambda\mathbf{L}^+\ \left(\mathbf{m}({}^c\hat{\mathbf{T}}_o, \hat{\boldsymbol{\tau}}) - \mathbf{m}^*(t)\right), \lambda > 0 \qquad (7)$$

where ${}^c\hat{\mathbf{T}}_o$ and $\hat{\boldsymbol{\tau}}$ are the previous estimates of ${}^c\mathbf{T}_o$ and $\boldsymbol{\tau}$. Note that $\mathbf{L}$, $\mathbf{m}$ and $\mathbf{m}^*$ are composed of rows that are evaluated at the successive time instant. In the context of virtual visual servoing framework (see Figure 2), this equation provides the pose and velocity correction vector. The object velocity is thus obtained by integrating the acceleration $\dot{\boldsymbol{\tau}}_u$ and the pose is obtained by exploiting both parts of the virtual control vector (velocity and acceleration). It might be used in the standard 'look-and-move' framework but we will see in the next section how this can be used in the more efficient 'dynamic direct visual servoing' framework by involving the accelerations and the inverse dynamic model (IDM).

3. Position-based dynamic visual servoing

3.1. The motion model

The work presented by Dahmouche et al. (2009) made the assumption that the velocity was piecewise constant

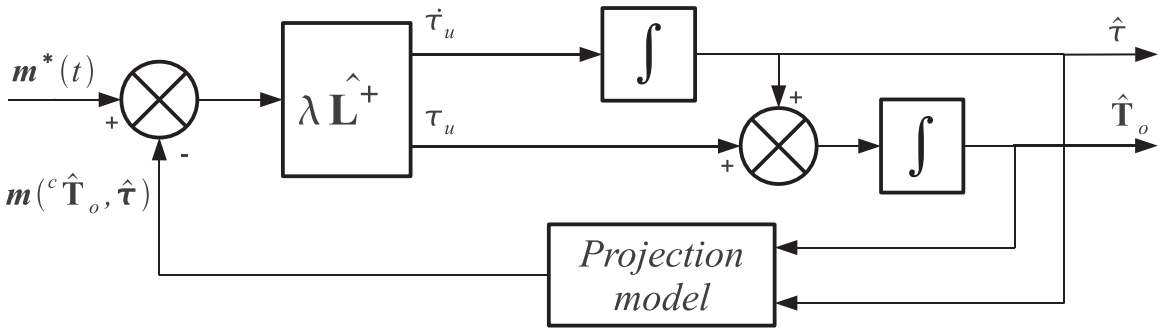

Fig. 2. Pose and velocity estimation under constant velocity assumption, using the virtual visual servoing paradigm.

over the time interval during which the sub-images were grabbed. However, this assumption is no longer appropriate for computed torque control. Indeed, under this control scheme, the torques applied to the actuators are computed from the output of the inverse dynamic model (IDM), which is, in turn, fed with the pseudo-control vector ($\mathbf{w}$) made of the acceleration to apply to the end-effector. Thus, over a control period, only the acceleration can be assumed constant (up to the internal regulation of the torques in the actuators). Therefore, the motion model of the target has to be reformulated under the latter assumption, which essentially boils down to two things: computing $\delta\mathbf{T}_i$ in order to compute the projection model and the interaction matrix and computing the control vector.

Under the assumption that the acceleration is constant over the control sampling period, $\boldsymbol{\tau}(t)$ is the end-effector twist which represents the integral of the robot end-effector acceleration:[2]

$$\boldsymbol{\tau}(t_i) = \boldsymbol{\tau}(t_{\text{ref}}) + \int_{t_{\text{ref}}}^{t_i} \dot{\boldsymbol{\tau}}_i \, dt \tag{8}$$

where $\dot{\boldsymbol{\tau}}_i = [\dot{\mathbf{v}}_i, \dot{\omega}_i]$ is the value of the platform acceleration at the sample time $t_i = i\ T_a$, T_a being the sub-image acquisition period. This period has to be taken smaller than the control period T_c in order to gather enough object points to update the pose and velocity between two control updates.

The integration of the acceleration to obtain the object translation velocity from (8) can be then written as

$$\mathbf{v}_i = \mathbf{v}_{\text{ref}} + \sum_{k=0}^{i-1} \dot{\mathbf{v}}_k \ T_a. \tag{9}$$

The rotation space being nonlinear, the integration of the rotation acceleration without simplification introduces an unnecessary computational burden. However, the instantaneous rotation axis direction of the platform being usually designed constant or slowly variable at trajectory planning time, a simplification of this motion model is to consider only the acceleration component which is parallel to the rotation velocity. In this case, the rotation velocity will have a constant direction $\mathbf{u}_\omega$ and a uniformly variable amplitude ($\dot{\omega}$) over one control sampling period. The object velocity and rotation displacement are hence obtained by integrating the projection of the piecewise constant rotational acceleration $\dot{\omega}$ on the rotational velocity axis $\mathbf{u}_\omega$:

$$\omega_i = \omega_0 + \sum_{k=0}^{i-1} (\dot{\omega}_k \cdot \mathbf{u}_\omega) \ T_a \ \mathbf{u}_\omega. \tag{10}$$

After obtaining the velocity expressions, one obtains respectively the translation and rotation parts ($\delta\mathbf{R}_i$ and $\delta\mathbf{t}_i$) of $\delta\mathbf{T}_i$ by integrating the obtained velocities:

$$\delta\mathbf{t}_i = \int_{t_{\text{ref}}}^{t_i} \mathbf{v}(t) \ dt = \sum_{k=0}^{i-1} \left(\frac{1}{2} \dot{\mathbf{v}}_k \ T_a^2 + \mathbf{v}_k \ T_a \right) \tag{11}$$

and

$$\delta\theta\mathbf{u}_i = \sum_{k=0}^{i-1} \left(\frac{1}{2} (\dot{\omega}_k \cdot \mathbf{u}_\omega) \ T_a^2 \ \mathbf{u}_\omega + \omega_k T_a \right) \tag{12}$$

where $\delta\theta\mathbf{u}_i$ is the rotation displacement vector.

The associated homogeneous rotation matrix of the object displacement $\delta\mathbf{R}_i$ can then be obtained from the rotation vector using Rodrigues formula or, equivalently, the exponential matrix map (expm) (Iserles et al. 2000):

$$\delta\mathbf{R}_i = \text{expm}([\delta\theta\mathbf{u}_i]_\times) \triangleq \exp(\delta\theta\mathbf{u}_i). \tag{13}$$

3.2. Pose and velocity estimation

In the previous section we have obtained the pose and velocity evolution of the robot's end-effector. The role of the virtual visual servoing is to regulate these variables to converge into the real ones thanks to the virtual control law (7). This is done simply by integrating the correction of the current state (pose and velocity). The robot acceleration being assumed constant over the control period T_c, we obtain the estimated pose and velocity as follows:

$$\hat{\boldsymbol{\tau}}_{j+1} = \hat{\boldsymbol{\tau}}_j + (\dot{\boldsymbol{\tau}}_j + \dot{\boldsymbol{\tau}}_{u_j}) \, T_c \tag{14}$$

$$\hat{\mathbf{t}}_{j+1} = \hat{\mathbf{t}}_j + \frac{1}{2} \dot{\mathbf{v}}_j T_c^2 + (\hat{\mathbf{v}}_j + \mathbf{v}_{u_j}) \ T_c \tag{15}$$

$$\hat{\mathbf{R}}_{j+1} = \hat{\mathbf{R}}_j \exp\left(\frac{1}{2} \left(\dot{\omega}_j \cdot \hat{\mathbf{u}}_\omega \right) T_c^2 \hat{\mathbf{u}}_\omega + (\hat{\omega}_j + \omega_{u_j}) \, T_c \right) \tag{16}$$

where $\hat{\mathbf{v}}_j$ and $\hat{\omega}_j$ are the translational and rotational estimated velocities of the end-effector ($\hat{\boldsymbol{\tau}}_j = [\hat{\mathbf{v}}_j, \ \hat{\omega}_j]$).

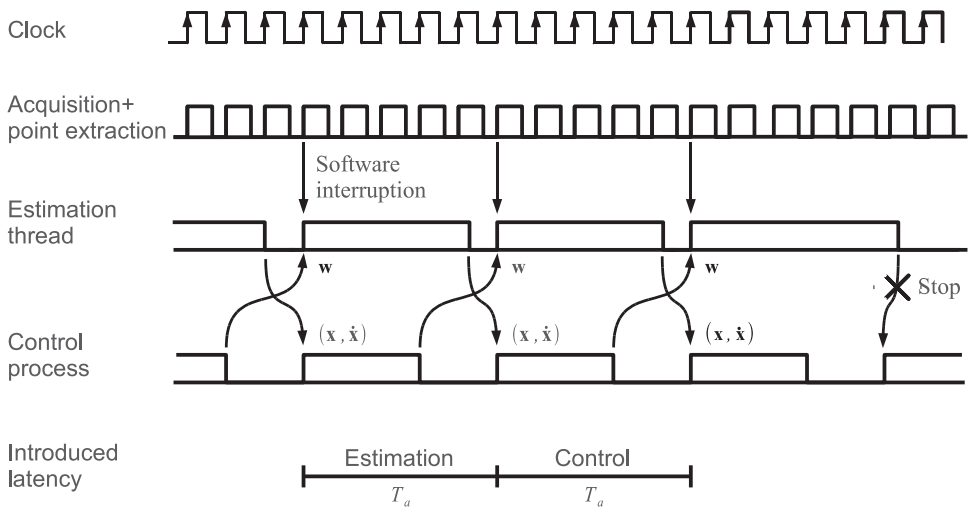

Fig. 3. Chronogram of the control process.

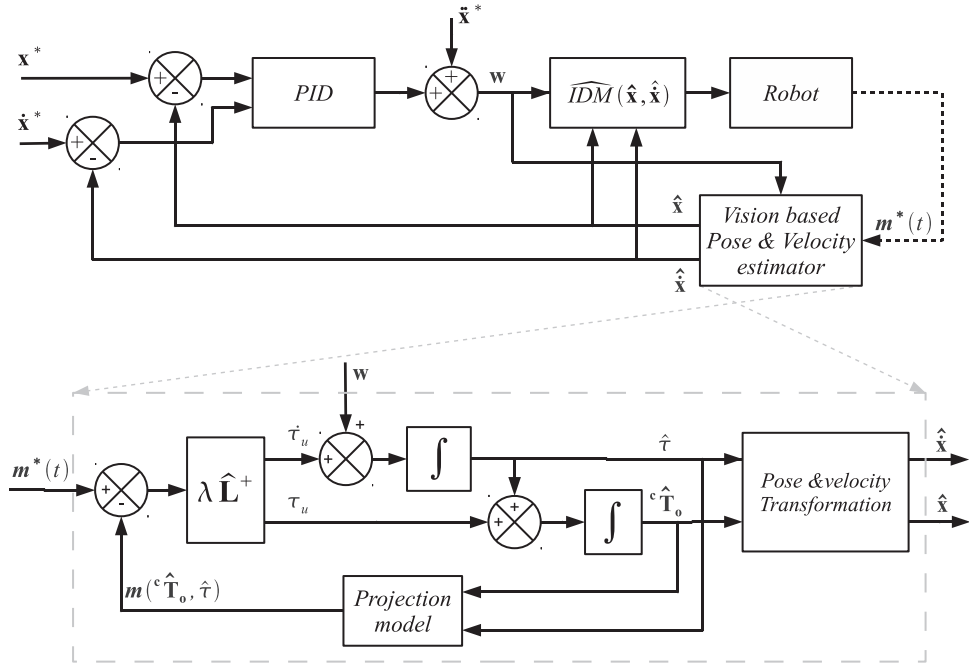

Fig. 4. The vision-based Cartesian computed torque control based on sequential image acquisition is composed of a standard Cartesian computed torque control (upper loop) and of a virtual visual servoing estimator (lower loop).

Note that here, the pose and velocity estimator is launched at each control sample time t_j (see Figure 3). Thus, the reference time t_{ref} is set at each control sample time to $t_{\text{ref}} = t_j$. Finally, the expressions (11) and (13) are now exploitable both for being inserted into the projection model (1) and into the interaction matrix expression (53) that requires an estimation of the 3D points coordinates.

3.3. Control scheme

The PBDVS control scheme (Figure 4) is composed of a virtual visual servoing control loop and a dynamic control loop. The virtual visual servoing control loop estimates the end-effector pose and velocity while the real control loop aims at regulating the estimated state with respect to the desired one under the computed torque control scheme.

The obtained pose and velocity are then transformed into the robot state space representation $(\mathbf{x}, \dot{\mathbf{x}})$ to be used in the dynamic control loop for the regulation and dynamic compensation. The IDM of the robot is thus provided from visual pose and velocity measures of the platform in the case of parallel robot (Khalil and Ibrahim 2004; Dahmouche et al. 2010), which makes the control theoretically

free from joint sensing, or from joints measures when controlling parallel robots. The control vector which represents the joints torques is thus given in the case of parallel robots as follows (Khalil and Ibrahim 2007):

$$\Gamma = \mathbb{D}^T \left(\mathbb{M}\mathbf{w} + \mathbb{C}(\mathbf{x}, \dot{\mathbf{x}}) + \mathbb{G}(\mathbf{x}) + \sum_{j=1}^{k} \mathbb{J}_{jp}^T \mathbb{J}_j^{-1} \mathbb{H}_j \right) \quad (17)$$

where:

- $\mathbf{w}$ represents the controller output (PID or PD, for instance) which corresponds to the robot platform acceleration (Figure 4);
- $\mathbb{D}$ is the forward instantaneous kinematic matrix of the manipulator;
- $\mathbb{M}$ is the inertia matrix of the robot;
- $\mathbb{C}(\mathbf{x}, \dot{\mathbf{x}})$ represents the Coriolis and centrifuge forces and $\mathbb{G}(\mathbf{x})$ the gravitational attraction;
- $\mathbb{H}_j$ is the IDM of the jth leg;
- $\mathbb{J}_{jp}$ is the Jacobian linking the last leg joint variables to the end-effector Cartesian variables and $\mathbb{J}_j^{-1}$ the jth legs inverse kinematic matrix.

Note that the case of parallel robots is considered in Section 6.

In the case of serial robots, the control torques are computed as follows:

$$\Gamma = \mathbb{M}(q)\ \ddot{q} + \mathbb{C}(q, \dot{q})\ \dot{q} + \mathbb{G}(q) \quad (18)$$

where $\mathbb{M}(q)$ is the inertia matrix of the robot, $\mathbb{C}(q, \dot{q})$ represents the Coriolis and centrifuge forces and $\mathbb{G}(q)$ is the gravity force. Here, the robot acceleration $\ddot{q}$ is obtained from the pseudo-control vector $\mathbf{w}$ (the controller output) and the second-order kinematic model of the robot. Note that the IDM for serial robots is written in terms of the joints position and velocity. In this case, the implementation of the PBDVS requires the computation of the joints position and velocity from the end-effector state (pose and velocity) using the inverse kinematic model of the robot.

4. Image-based dynamic visual servoing

4.1. Dynamic control in the image

In classical visual servoing the robot can be modeled as a simple integrator which corresponds to a first-order system. The conventional approach to control this system is to impose a first-order decrease of the task function (Espiau et al. 1992). However, this approach is not suitable in direct dynamic visual servo control since the control law provides torques and not velocities.

To design an image-based dynamic control, we propose an approach inspired from classical computed torque control where the objective is to servo the system through a proportional and derivative actions. The closed-loop corresponds then to a second-order system of the form:

$$\ddot{\mathbf{x}}^* - \ddot{\mathbf{x}} + K_v(\dot{\mathbf{x}}^* - \dot{\mathbf{x}}) + K_p(\mathbf{x}^* - \mathbf{x}) = 0 \quad (19)$$

were $\mathbf{x}$, $\dot{\mathbf{x}}$ and $\ddot{\mathbf{x}}$ are, respectively, the robot configuration, velocity and acceleration in a suitable representation. The superscript * denotes the desired value of the considered variable. In practice, the proportional and derivative control gains, respectively K_p and K_v, can be tuned to obtain the desired response.

To design the dynamic visual servoing control law, let us define the task function as follows (Samson et al. 1991):

$$\mathbf{e} = \mathbf{C}\,(\mathbf{m}(\mathbf{x}, \dot{\mathbf{x}}) - \mathbf{m}^*) \quad (20)$$

where $\mathbf{e}$ is the task function and $\mathbf{C}$ the combination matrix. The pair $\mathbf{m}^*$ and $\mathbf{m}(\mathbf{x}, \dot{\mathbf{x}})$ (denoted by $\mathbf{m}$ in the sequel) refers to the desired and observed features positions in the image. In the case of sequential ROI acquisition, these positions depend on the relative pose and velocity of the target (Dahmouche et al. 2009).

In contrast to the classical methods, we aim at obtaining a second-order (instead of the usual first-order) decay of the task function. This can be formulated as follows:

$$\ddot{\mathbf{e}} + \Lambda_v\, \dot{\mathbf{e}} + \Lambda_p\, \mathbf{e} = 0 \quad (21)$$

where Λ_p and Λ_v are diagonal matrices containing the control gains.

The robot dynamics are thus formulated from image features rather than from the robot configuration (most often unmeasurable) as in (19). Now, the problem is to define the task function $\mathbf{e}$ and to handle properly the time differentiation of the nonlinear task function.

4.2. IBDVS task function

A single image acquired by a classical visual acquisition system does not contain velocity information. As a result, it is not possible to obtain the second derivative of the task function without numerical differentiation (which would lead to noise amplification). A significant advantage of exploiting a set of visual features acquired at different time instants is that they contain pose and velocity information about the observed motion. It is thus possible to extract pose and velocity errors without any use of numerical differentiation.

However, dealing with visual features acquired at different time instants raises some new issues since conventionally the features' position errors in the image correspond to a pose error and its derivative to a velocity error. This is clearly not true with the task function (20) when using non-simultaneous acquisition since the features positions in the images depend on the pose and the velocity. One has thus to choose adequately the combination matrix of the task function so that it represents only the robot configuration error. In other words, the task function must be independent from the robot velocity. Furthermore, to avoid numerical differentiation of the task function, we propose to define a second task function related to the velocity errors. We will now see how the two task functions

are defined and how they can be adequately combined to design a coherent and consistent control law.

For this purpose, let us define the two task functions as follows:

$$\mathbf{e} = \mathbf{C}(\mathbf{m} - \mathbf{m}^*) \tag{22}$$
$$\mathbf{e}' = \mathbf{C}'(\mathbf{m} - \mathbf{m}^*) \tag{23}$$

where $\mathbf{C}$ and $\mathbf{C}'$ are combination matrices. The role of the first task function is only to constrain the desired position whereas the second task function is designed to represent only the velocity error. Towards this aim, $\mathbf{e}$ must be made independent from the velocity error and $\mathbf{e}'$ independent from the pose error which can be done by adequately choosing $\mathbf{C}$ and $\mathbf{C}'$. These conditions can be formalized as follows:

$$\frac{\partial \mathbf{e}}{\partial \dot{\boldsymbol{\tau}}} = 0 \tag{24}$$
$$\frac{\partial \mathbf{e}'}{\partial \boldsymbol{\tau}} = 0. \tag{25}$$

To find the combination matrices allowing to satisfy these two conditions, let us express the derivatives of the task functions with respect to the robot velocity and acceleration:

$$\begin{aligned}\dot{\mathbf{e}} = \mathbf{C}(\dot{\mathbf{m}} - \dot{\mathbf{m}}^*) &= \mathbf{C}\,\mathbf{L}\begin{bmatrix}\boldsymbol{\tau}\\ \dot{\boldsymbol{\tau}}\end{bmatrix} - \mathbf{C}\,\mathbf{L}\begin{bmatrix}\boldsymbol{\tau}^*\\ \dot{\boldsymbol{\tau}}^*\end{bmatrix}\\ &= \mathbf{C}\,\mathbf{L}\begin{bmatrix}\boldsymbol{\tau} - \boldsymbol{\tau}^*\\ \dot{\boldsymbol{\tau}} - \dot{\boldsymbol{\tau}}^*\end{bmatrix}\end{aligned} \tag{26}$$

where $\mathbf{L}$ is the interaction matrix relating the features velocity to the kinematic twist and the acceleration.

Decomposing the ($2n \times 12$) interaction matrix $\mathbf{L}$ into its two ($2n \times 6$) blocks $\mathbf{L}_{\boldsymbol{\tau}}$, $\mathbf{L}_{\dot{\boldsymbol{\tau}}}$ related to the kinematic twist and to the acceleration ($\mathbf{L} = [\mathbf{L}_{\boldsymbol{\tau}}, \mathbf{L}_{\dot{\boldsymbol{\tau}}}]$) yields the following expression of the time derivative of the two task functions:

$$\dot{\mathbf{e}} = \mathbf{C}(\dot{\mathbf{m}} - \dot{\mathbf{m}}^*) = \mathbf{C}\mathbf{L}_{\boldsymbol{\tau}}(\boldsymbol{\tau} - \boldsymbol{\tau}^*) + \mathbf{C}\mathbf{L}_{\dot{\boldsymbol{\tau}}}(\dot{\boldsymbol{\tau}} - \dot{\boldsymbol{\tau}}^*) \tag{27}$$
$$\dot{\mathbf{e}}' = \mathbf{C}'(\dot{\mathbf{m}} - \dot{\mathbf{m}}^*) = \mathbf{C}'\mathbf{L}_{\boldsymbol{\tau}}(\boldsymbol{\tau} - \boldsymbol{\tau}^*) + \mathbf{C}'\mathbf{L}_{\dot{\boldsymbol{\tau}}}(\dot{\boldsymbol{\tau}} - \dot{\boldsymbol{\tau}}^*) \tag{28}$$

From these two equations, we deduce that the conditions to satisfy (24) and (25) are

$$\mathbf{C}\mathbf{L}_{\dot{\boldsymbol{\tau}}} = 0_{6\times 6} \tag{29}$$
$$\mathbf{C}'\mathbf{L}_{\boldsymbol{\tau}} = 0_{6\times 6}. \tag{30}$$

Furthermore, the system behavior can be improved by decoupling the elements of the kinematic screw and the accelerations by imposing secondary conditions:

$$\mathbf{C}\mathbf{L}_{\boldsymbol{\tau}} = \mathbf{I}_{6\times 6} \tag{31}$$
$$\mathbf{C}'\mathbf{L}_{\dot{\boldsymbol{\tau}}} = \mathbf{I}_{6\times 6}. \tag{32}$$

The matrices $\mathbf{C}$ and $\mathbf{C}'$ can be obtained by exploiting the properties of the pseudo-inverse. By definition, for $n \geq 6$,

$$\mathbf{L}^+\,\mathbf{L} = \mathbf{I}_{12\times 12}. \tag{33}$$

Let us decompose the pseudo-inverse of the interaction matrix into two blocks:

$$\mathbf{L}^+ = \begin{bmatrix}\mathbf{L}_u^+\\ \mathbf{L}_l^+\end{bmatrix} \tag{34}$$

where $\mathbf{L}_u^+$ and $\mathbf{L}_l^+$ are, respectively, the $6 \times 2n$ upper and lower sub-matrices of the interaction matrix pseudo-inverse.

The property (33) yields

$$\mathbf{L}_u^+\,\mathbf{L}_{\boldsymbol{\tau}} = \mathbf{I} \tag{35}$$
$$\mathbf{L}_l^+\,\mathbf{L}_{\boldsymbol{\tau}} = 0 \tag{36}$$
$$\mathbf{L}_u^+\,\mathbf{L}_{\dot{\boldsymbol{\tau}}} = 0 \tag{37}$$
$$\mathbf{L}_l^+\,\mathbf{L}_{\dot{\boldsymbol{\tau}}} = \mathbf{I}. \tag{38}$$

These properties will be exploited in the next section to define the combination matrices $\mathbf{C}$ and $\mathbf{C}'$ that satisfy the control law constraints (29) to (32).

4.3. Control design

By setting $\mathbf{C} = \widehat{\mathbf{L}}_u^+$ and $\mathbf{C}' = \widehat{\mathbf{L}}_l^+$ respectively as an estimate of the upper part and the lower part of the interaction matrix pseudo-inverse, Equations (27) and (28) become

$$\dot{\mathbf{e}} = \widehat{\mathbf{L}}_u^+\mathbf{L}_{\boldsymbol{\tau}}(\boldsymbol{\tau} - \boldsymbol{\tau}^*) + \widehat{\mathbf{L}}_u^+\mathbf{L}_{\dot{\boldsymbol{\tau}}}(\dot{\boldsymbol{\tau}} - \dot{\boldsymbol{\tau}}^*) \tag{39}$$
$$\dot{\mathbf{e}}' = \widehat{\mathbf{L}}_l^+\mathbf{L}_{\boldsymbol{\tau}}(\boldsymbol{\tau} - \boldsymbol{\tau}^*) + \widehat{\mathbf{L}}_l^+\mathbf{L}_{\dot{\boldsymbol{\tau}}}(\dot{\boldsymbol{\tau}} - \dot{\boldsymbol{\tau}}^*). \tag{40}$$

Under the assumption that $\widehat{\mathbf{L}}^+$, is accurately estimated (i.e. satisfies (35)–(38)), Equations (39) and (40) become

$$\dot{\mathbf{e}} = \boldsymbol{\tau} - \boldsymbol{\tau}^* \tag{41}$$
$$\dot{\mathbf{e}}' = \dot{\boldsymbol{\tau}} - \dot{\boldsymbol{\tau}}^*. \tag{42}$$

The derivative of the task function $\mathbf{e}$ depends only on the kinematic twist while the derivative of the task function $\mathbf{e}'$ depends only on acceleration. Therefore, the two task functions are decoupled and the secondary conditions (31) and (32) are satisfied.

We can deduce from (41) and (42) that the task function $\dot{\mathbf{e}}'$ is equal to the time derivative of the task function $\mathbf{e}$:

$$\dot{\mathbf{e}} = \mathbf{e}'. \tag{43}$$

The second derivative of the task function $\mathbf{e}$ can then be written in terms of the derivative of $\mathbf{e}'$ as follows:

$$\ddot{\mathbf{e}} = \dot{\mathbf{e}}' = \widehat{\mathbf{L}}_l^+(\dot{\mathbf{m}} - \dot{\mathbf{m}}^*) = \dot{\boldsymbol{\tau}} - \widehat{\mathbf{L}}_l^+\dot{\mathbf{m}}^*. \tag{44}$$

The substitution of Equations (22), (23), (43) and (44) into (21) gives

$$\dot{\boldsymbol{\tau}} = \widehat{\mathbf{L}}_l^+\dot{\mathbf{m}}^* + \Lambda_v\widehat{\mathbf{L}}_l^+(\mathbf{m}^* - \mathbf{m}) + \Lambda_p\widehat{\mathbf{L}}_u^+(\mathbf{m}^* - \mathbf{m}). \tag{45}$$

Interestingly by combining the two sub-matrices $\widehat{\mathbf{L}}_l^+$ and $\widehat{\mathbf{L}}_u^+$, we obtain a more compact expression of the dynamic visual servoing control law:

$$\dot{\boldsymbol{\tau}} = -[\Lambda_p, \Lambda_v]\,\widehat{\mathbf{L}}^+(\mathbf{m} - \mathbf{m}^*) + \widehat{\mathbf{L}}_l^+\dot{\mathbf{m}}^*. \tag{46}$$

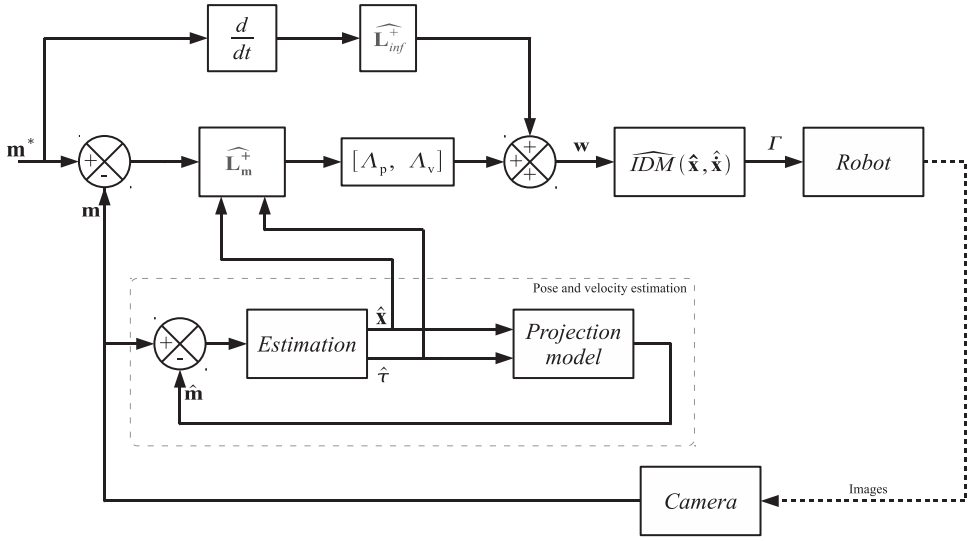

Fig. 5. Image-based dynamic visual servoing. The pseudo-control vector (acceleration) is computed from visual features. Pose and velocity estimation is provided thanks to the method presented in Section 2.

Note that the choice of gain matrix specifies the evolution of the task function. Moreover, as we impose a second-order decay of the task function, the steady-state error is null when tracking a constant velocity. This property should provide good tracking performances.

The interaction matrix depends on the pose and velocity of the platform of the robot. These parameters can be obtained using the algorithms presented in Section 2. Note also that errors on the estimation of the pose and the velocity have only a weak effect on the accuracy and the stability of the system since these estimations are not used as a control feedback in contrast to PBDVS. The joints torques are then computed through the inverse dynamic model as in PBDVS using (17) and (18) in the case of parallel and serial robots, respectively. An illustration of the IBDVS control scheme is given in Figure 5.

Remark. Let us point out again that the proposed control scheme is based on the assumption that the estimate of the interaction matrix is sufficiently accurate to consider that $\widehat{\mathbf{L}}^{+}\mathbf{L} \approx \mathbf{I}$. This hypothesis may appear strong but it is similar to the one implicitly assumed in the stability proof of the standard visual servoing when the reference image varies. It is also similar to the assumption made in the proof of stability of computed torque control. In practice, this assumption is quite realistic because of two main reasons. The first is that the pose and velocity errors are typically small in trajectory tracking applications. In this case, the update of the combination matrices at each iteration provides an accurate linearization of the control, especially in high control frequency. The second reason is that the estimation algorithm allows measurements of the pose and the velocity to be obtained with a good accuracy (Dahmouche et al. 2009) which should yield a good estimate of the interaction matrix.

5. Implementation

The proposed method was validated on the Orthoglide robot (Chablat and Wenger 2003) with the setup shown in Figure 6. The control system architecture is composed of an off-the-shelf 'Photon focus MV-D1024-TrackCam' camera, a standard PC and the robot itself, controlled by a real-time PSpace DSP robot controller (Figure 7). The PC handles the image acquisition and the pose and velocity estimation process, while the DSP computes the torque control and handles the low-level control and security.

More precisely, the PC controls the camera to achieve a high-speed sequential sub-images acquisition by running the acquisition and the estimation processes in two parallel threads. The acquisition thread is triggered by the robot controller clock at a 4 kHz frequency, corresponding to the camera acquisition frequency. In this thread, the position of the current sub-image is predicted from the previously estimated pose and velocity, then the sub-image is grabbed and analyzed and the extracted point image coordinates are forwarded to the estimation process through a shared memory containing the image coordinates of 16 target points (this number was empirically chosen). The estimation process (which runs in another thread) is requested each $k = 10$ ROIs acquisition. It runs then at 400 Hz, the same sampling frequency as the robot controller. The estimation process uses the 16 points contained in the shared buffer to update the pose and velocity estimate. Figure 8 illustrates the acquisition and estimation processes.

The robot end-effector pose and velocity are estimated by the PC and transmitted to the DSP card via an industrial RS-422 serial interface. Since the PC is constrained by the camera driver to run under a non-real-time operating system, the estimation process takes an unpredictable amount of time, which is nevertheless expected to fit the 400 Hz robot controller frequency and secured by a watchdog timer.

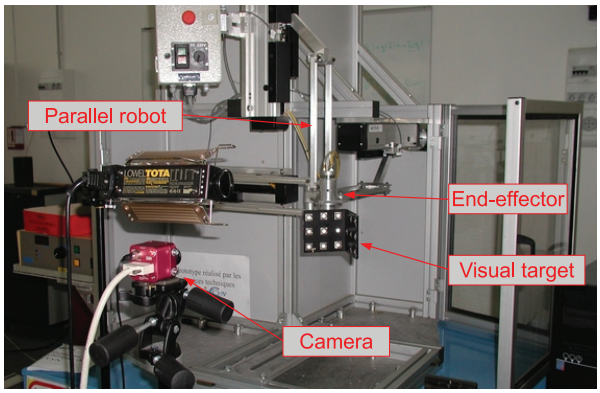

Fig. 6. Experimental setup for position-based and image-based visual servoings on the Orthoglide.

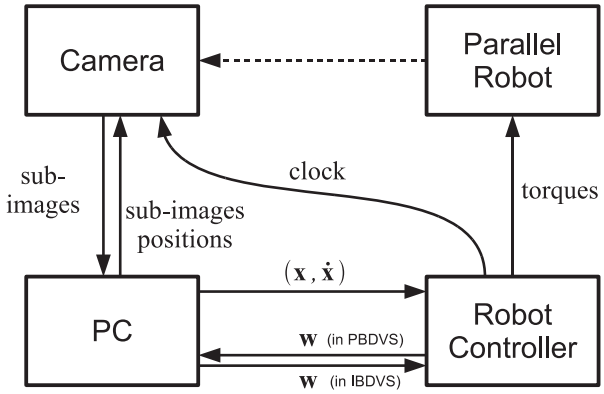

Fig. 7. Data flow in the implemented control architecture. In PBDVS the pseudo-control vector $\mathbf{w}$ is computed by the DSP card and transmitted to the PC to improve pose and velocity estimation while in IBDVS $\mathbf{w}$ is directly computed in the PC and transmitted to the robot controller.

Finally, in the PBDVS the control vector processed by the robot controller is sent back to the estimation process on the PC through the RS-422 link while in IBDVS the control vector is computed in the PC from image errors and sent to the DSP card to process dynamic compensation from the measured pose and velocity.

While implementing the control schemes, one has to take care of the sampling effect and the delay introduced by the acquisition and processing time. However, in our case, the control law sampling frequency is high enough (400 Hz) compared with the control gain (for instance, 25 rad s^{-1} in PBVS) to neglect the sampling effect. In addition, one advantage of the parallel implementation of acquisition/points extraction and estimation is that the time delay introduced by the processing time is considerably reduced particularly in sequential acquisition of ROIs. Indeed, the points are extracted one by one at very high speed (4 kHz) and not all at the same time as in global acquisition which can be time consuming. The time lag introduced by acquisition, estimation and dynamic control is equivalent to two sampling periods as it can be seen in the control process chronogram (Figure 3). In addition, as we will see, the observation of the end-effector allows us to measure some dynamic phenomenon such as flexibilities and backlashes. These nonlinearities in the system lead us to prefer fairly robust control approaches (PD or PID, for instance) to the use of linear automatic control tools (such as Z-transform) that are not adapted to nonlinear systems.

Note that the implementation of the proposed control laws take all their meaning in a parallel robot control application since the platform pose and velocity are usually sufficient to define the robot configuration. This makes any use of joint sensor dispensable. The implementation of the control law on a serial robot may require either the computation of the joint configuration and velocity though the inverse kinematic model or the use of a joint sensor to resolve the possible ambiguities of the robot configuration from the end-effector pose.

6. Experimental results

6.1. Preliminary tests

To have a relevant interpretation of the results, it is necessary to characterize both the vision system and the robot in terms of reliability and accuracy before proceeding to the implementation of the proposed control scheme. With the lack of other exteroceptive measurements (interferometric laser, for instance), the only practical way to identify possible defaults of either of the two systems (vision or robot) is by proceeding to some preliminary tests.

6.1.1. Vision accuracy First of all, different static poses were estimated during several seconds at the operating frequency (400 Hz). The standard deviations of the corresponding position $stdev(\mathbf{t})$ and velocity $stdev(v)$ estimation noise were measured:

$$stdev(\mathbf{t}) = [2.67,\ 4.05,\ 3.45]\ 10^{-5}\text{m}$$

and

$$stdev(v) = [2.04,\ 3.2,\ 5.75]\ 10^{-3}\text{m s}^{-1}.$$

Note that these values are considerably small, meaning a very stable estimation of the pose between two iterations (including a stable feature extraction) even though there might exist a bias between the mean estimation of the pose and the actual one. However, the average reprojection error is only of 0.19 pixel/point indicates that even this bias is small too. This will be confirmed by the results obtained in high-speed trajectories (Section 6.2) where the residual error is still less than a pixel.

6.1.2. Backlashes and flexibilities Concerning the robot, some joint flexibilities and backlashes were noticed. To characterize the resulting motion of the platform, an effort was applied by hand on the latter while the brakes were on and its displacement was measured with vision. The

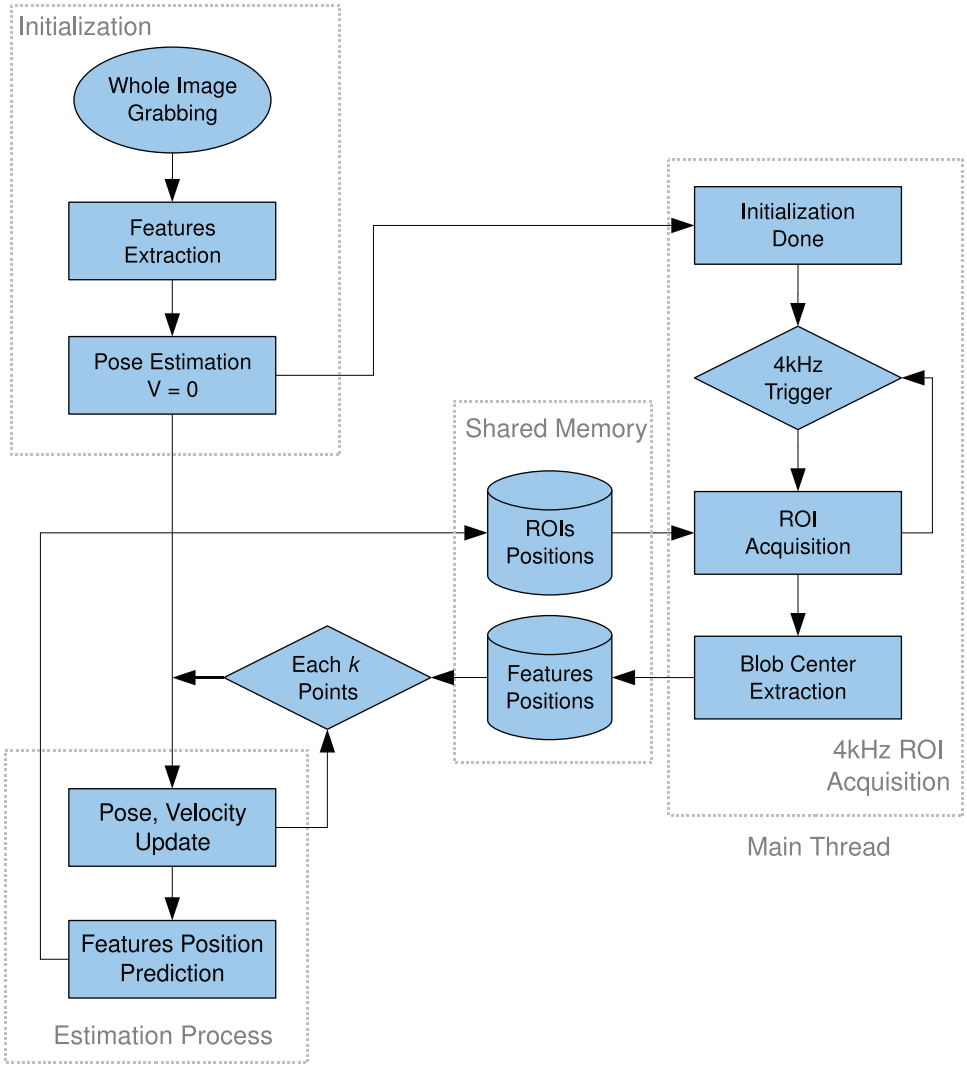

Fig. 8. High-speed parallel processing timing diagram.

resulting displacement of the target in the Cartesian space is

$$\delta\mathbf{x} = [10.0\text{ mm},\ 8.1\text{ mm},\ 7.1\text{mm},\ 2.38^\circ,\ 2.65^\circ,\ 2.93^\circ]^{\mathrm{T}}$$

while the mean points displacement in the image is $[\delta u,\ \delta v] = [25.9,\ 20.3]$ pixels. This displacement is much larger than the measurement residual error. These results give an idea of the correspondence between image errors and Cartesian error. In addition, they show that in static, the vision-based pose estimation of the platform is more accurate than the model-based estimation. This is underlined to be one of the most important benefits of exteroceptive sensors.

6.2. *Position-based dynamic visual servoing*

After achieving this procedure, the proposed control law was implemented. Note that the use of the pose and velocity estimation method in Dahmouche et al. (2009) in the control law leads to robot instability. The main instability cause, as stated, is the velocity tracking delay due to the assumption of a constant velocity. On the opposite, the implementation of the improved method presented in Section 2 allows us to not only stabilize the robot but also to control it to 100% of its speed. However, since the flexibilities and backlashes are measured, they are accounted for by the control law and this causes the robot to oscillate. To reduce this phenomenon, the natural frequency of the closed-loop system was decreased by scaling down the PID controller gain by 20%.

In the results given below, the reference trajectory is an oblique circle of 6 cm radius which is twice traveled through. The maximum velocity reached during the trajectory execution is 1 m s^{-1} which corresponds to 16.4 m s^{-2} tangential acceleration and 16.67 m s^{-2} normal acceleration. The two trajectory laps are achieved in less than 1.4 s.

Figure 9 shows the reference trajectory, the trajectory realized under vision-based computed torque control and

Table 1. Mean (in mm) and standard deviation (in mm) using the model-based and vision-based control.

	Model-based control			Vision-based control		
	x	*y*	*z*	*x*	*y*	*z*
Mean	0.19	0.1	0.28	0.11	−0.08	−0.06
Standard deviation	2.42	1.47	1.76	2.21	1.6	1.64

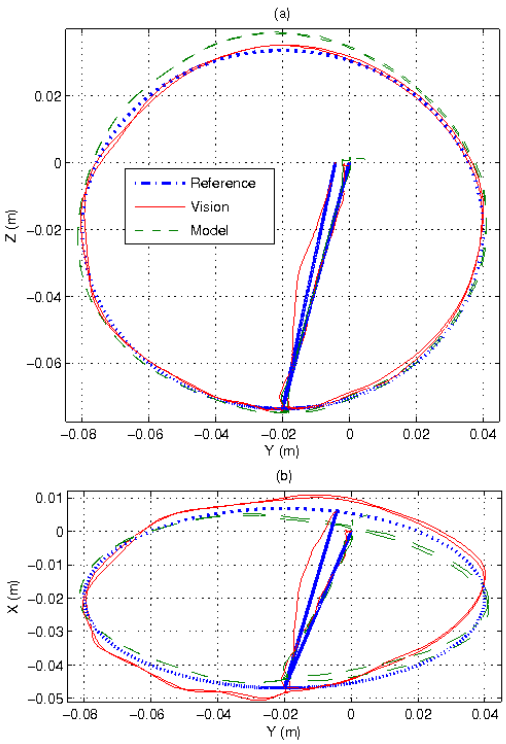

Fig. 9. Trajectories in space: reference, vision-based and joint-based (computed from the kinematic model) along (a) the *x*-axis and (b) the *z*-axis.

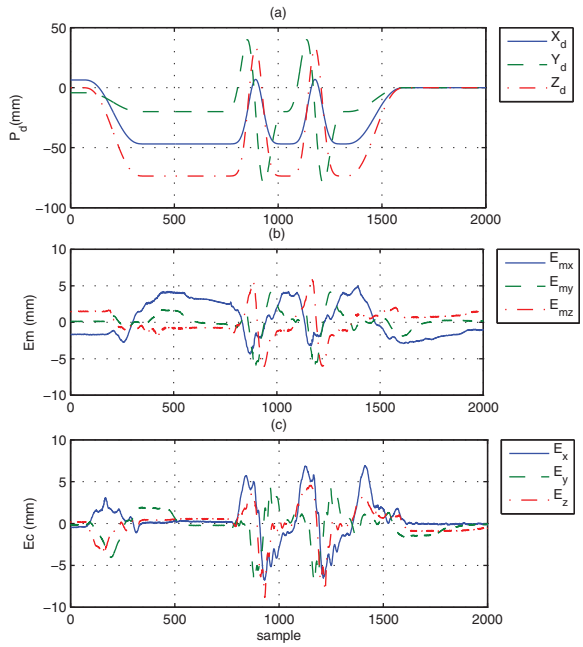

Fig. 10. Desired trajectory (top) with respect to time and errors for model-based (center) and vision-based (bottom) control.

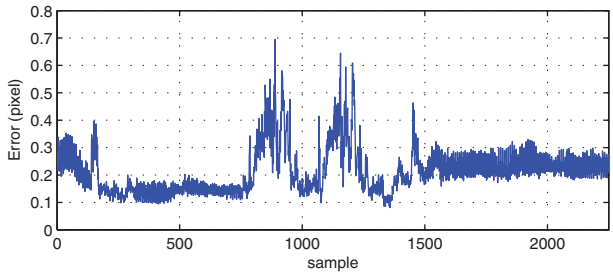

Fig. 11. Image reprojection error with respect to time.

that achieved under model-based computed torque control. Both trajectories were recorded by vision, since the latter has shown better accuracy than the model. First of all, note that the trajectory obtained with the vision-based control seems to be as smooth as that obtained with the model-based control. One should also notes that the model-based trajectory radius seems to be larger than the reference trajectory while this is, visibly, not the case of the vision-based trajectory. Indeed, the mean radius of the model-based trajectory and the vision-based trajectory are 61.34 and 60.20 mm, respectively. In addition, the algebraic distance between the trajectories and the reference circle (normal distance) is smaller for the vision-based control (2.74 mm) than for the model-based control (3.25 mm).

Figure 10 represents the desired positions and, respectively, the errors obtained from the application of the model-based and vision-based controls. One notices first that the vision-based control error is smaller than the measured backlashes. In addition, a comparison between both errors reveals that model-based control errors are important in statics as well as in dynamics. Errors in statics using the vision-based control are smaller than those obtained using the model-based control and almost equivalent in dynamics. This is confirmed by the respective means and standard deviations of the tracking errors (Table 1).

Figure 11 shows the pose and velocity tracking errors in the image. Note that even at this high speed, the image reprojection errors remain smaller than 1 pixel. It also seems that the image error increases with acceleration. This may be caused by the errors between the estimated inverse

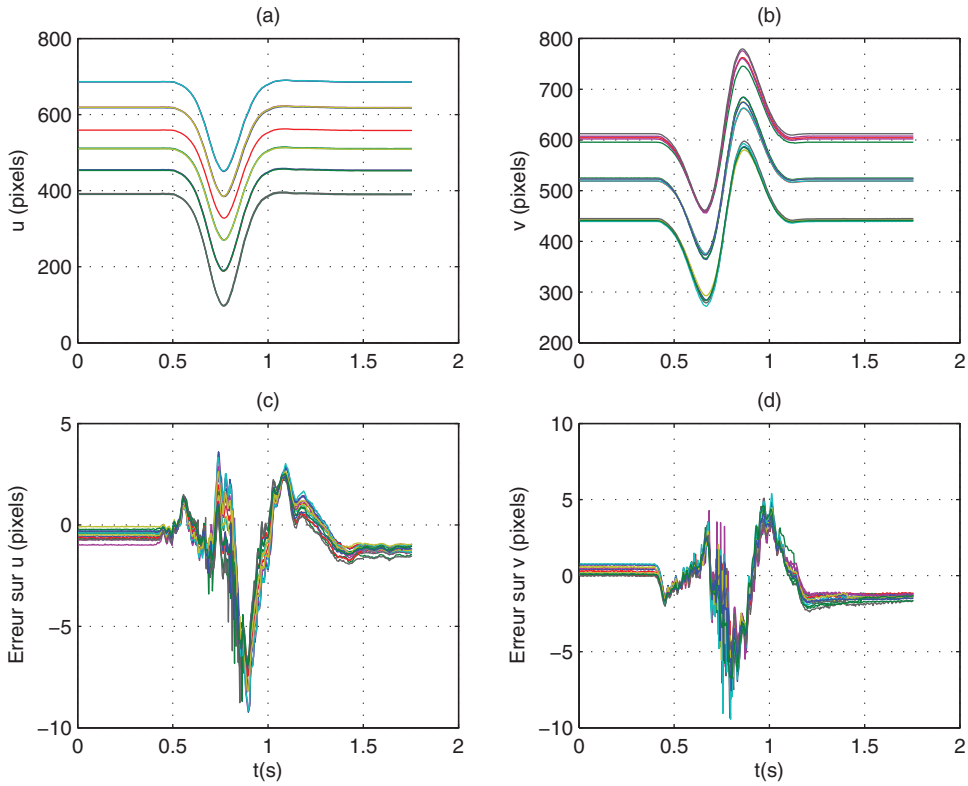

Fig. 12. Features positions in the image in (a) the *u*-axis and (b) the *v*-axis, and the positions errors in (c) the *u*-axis and (d) the *v*-axis.

Table 2. Features tracking errors in the image in pixels.

	u	*v*
Mean	−0.73	−0.50
Standard deviation	1.63	1.70

dynamic model and the actual one, which can be shown to yield a computed torque control error proportional to the control acceleration. Nevertheless, the error in the image remains much smaller than projection residuals due to flexibilities and backlashes, and hence the dynamic errors given in Table 1 are certainly due to the latter.

6.3. *Image-based dynamic visual servoing*

One difficulty in the implementation of the IBDVS is the generation of the reference trajectory in the image. Several path planning algorithms that have been proposed in the literature (see Mezouar and Chaumette 2002, for instance) can certainly be adapted to our context. However, to validate the proposed control law, the learning technique was preferred to eliminate any errors or inaccuracies in the synthesized path that could mislead the results analysis. The desired image trajectory was obtained through the 'teaching-by-showing' technique. The positions of visual features were recorded during the execution of the Cartesian trajectory obtained by the vision-based 3D dynamic control. In the results given below, the reference trajectory is similar to that used in the previous experiments.

To improve the performance of the control, a mean filter is used to reduce the noise on the speed error. Moreover, to reduce the effect of friction, a PID regulator is implemented instead of a PD. The integral action is implemented by integrating the pose error as it is done in a classical way. The gains of the PID controller were selected to provide a critical response of the third-order with a triple pole.

As can be seen in Figure 12, the obtained trajectory is smooth and fits the reference trajectory quite properly. The maximum error shown in the same figure remains lower than the error in the image due to the backlashes and flexibilities. Table 2 shows the mean and standard deviation errors in the image between the reference trajectory and the trajectory obtained by the 2D dynamic visual servoing.[3]

Figure 13 shows the desired and the vision-based estimate of the Cartesian trajectories. The obtained Cartesian trajectory is smooth and fits pretty well the desired trajectory. The position evolution of the platform measured by vision and the positioning error in space are shown in Figure 14. This latter shows that the positioning error is smaller than the flexibilities and backlashes which means that the errors caused by the mechanical defects of the robot are partially compensated for. Figure 15 shows also that the desired velocity is well tracked. Statistics on position and velocity errors including the standard deviations on the three axes presented in Table 3 confirm this comment.

Table 3. IBDVS 3D position (mm) and velocity (mm s^{-1}) errors.

	x	y	z	v_x	v_y	v_z
Mean	−0.47	−0.79	0.17	0.16	0.01	−0.04
Standard deviation	0.65	0.68	2.00	38.6	16.8	29.5

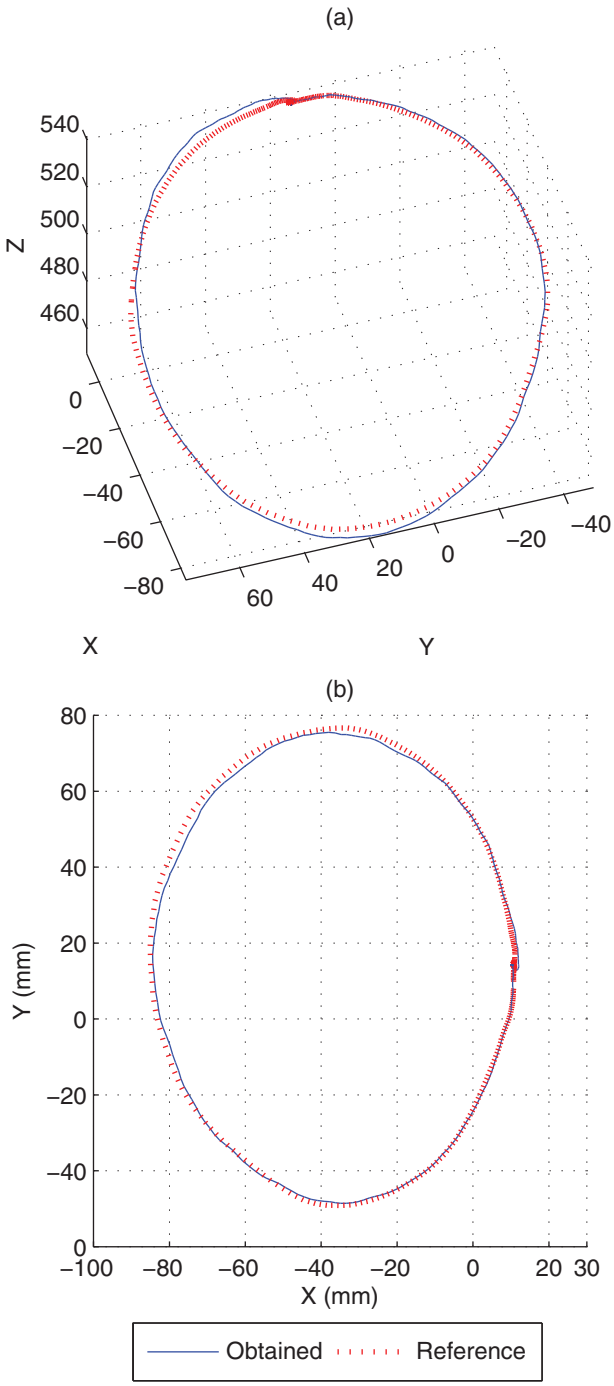

Fig. 13. The desired and the obtained dynamic vision-based Cartesian trajectories: (a) 3D view and (b) view along the z-axis.

Note also that this control law performs better than a classical joint-based control law and even better than the PBDVS since the obtained standard deviations for the three control laws are: joint-based dynamic control (3.33 mm), PBDVS (3.18 mm) and IBDVS (2.21 mm).

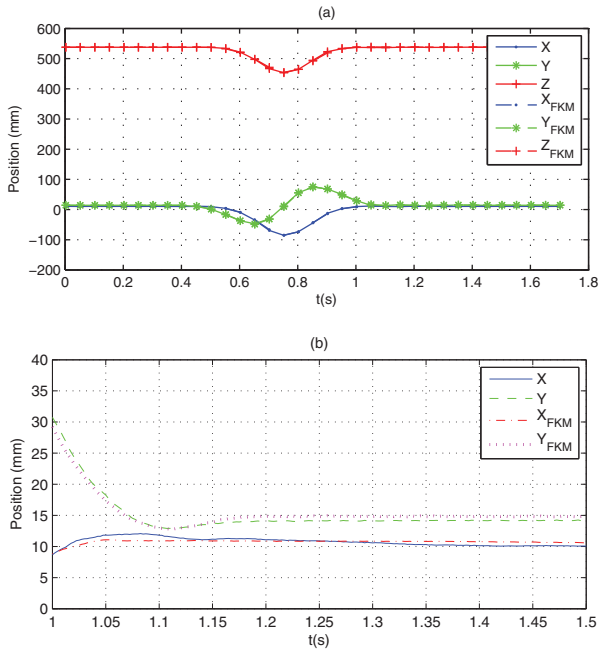

Fig. 14. (a) The desired and the obtained Cartesian positions with IBDVS and (b) the position error.

It was noted during the experiments that increasing control gains makes the robot oscillate. To identify the source of these oscillations (control or robot mechanics), the control gains were increased until the appearance of significant oscillations (initial gains multiplied by 2 which corresponds to an angular frequency of 50 rad s^{-1}). Figure 16(a) shows the trajectories reconstructed from joint measures and the forward kinematic model with the same 2D dynamic control. It can be seen in the zoomed curves (Figure 16(b)) that the trajectory reconstructed by vision reveals oscillations which also appear in the image (Figure 17). However, these oscillations are much weaker in the path obtained by the kinematic model. This means that small oscillations at the joints level result in larger displacements of the platform. The most plausible explanation for this phenomenon is that the small oscillations at the joints are amplified by the flexibilities and the backlashes between the actuators and the platform. This also explains why vision-based control performs better than model-based control.

Moreover, the mechanical defects introduce phase shifts in the movement between the actuators and the platform.

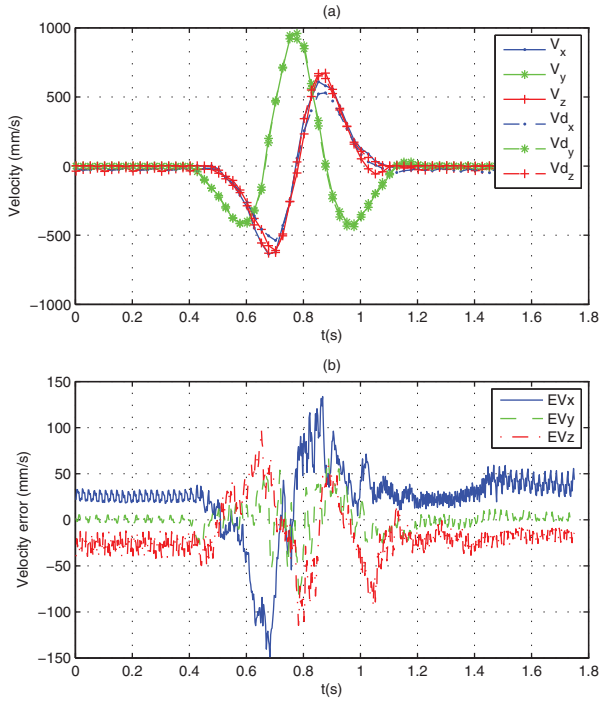

Fig. 15. (a) The desired and the obtained dynamic visual servoing Cartesian velocities with IBDVS and (b) the velocity error.

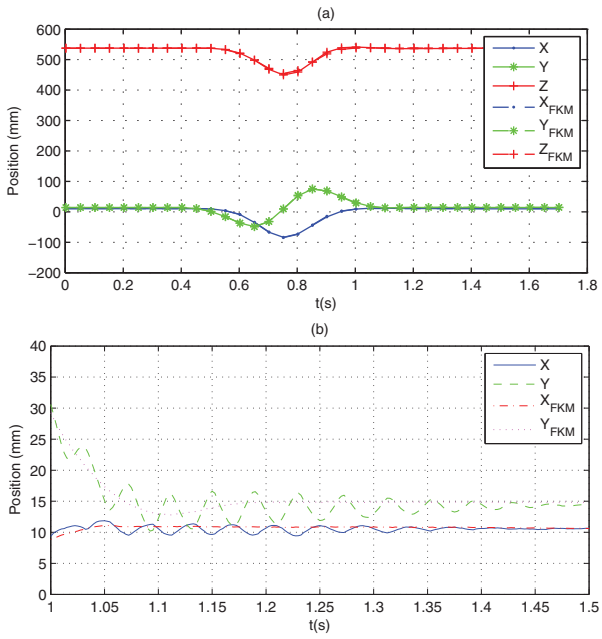

Fig. 16. Comparison between the desired and the obtained Cartesian positions with control gains multiplied by two with respect to Figure 14: (a) normal view and (b) zoomed view.

This phase shift increases with the natural frequency of the control (proportional gains), which can lead the system into an oscillatory mode. Therefore, the control gains are, most likely, limited by the imperfections of the robot and not by the vision-based control.

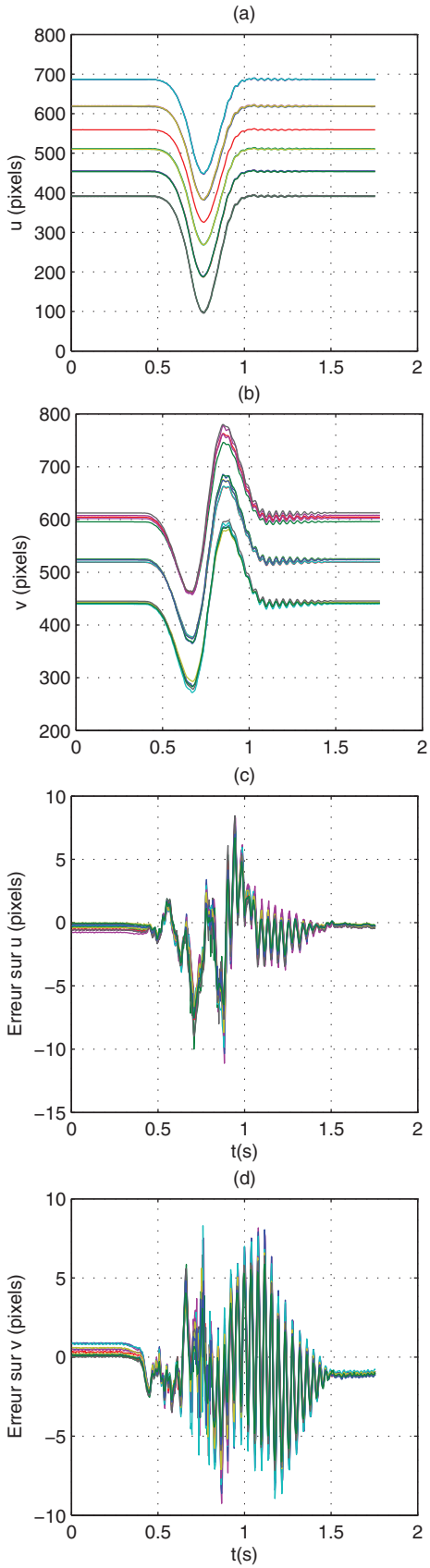

Fig. 17. Features positions in the image along (a) the u-axis and (b) the v-axis, and the positions errors along (c) the u-axis and (d) the v-axis for a control angular frequency of 50 rad s^{-1}.

7. Conclusion

In this paper, two dynamic visual servo control schemes based on a high-speed acquisition system have been presented. The first is a position (and velocity)-based dynamic visual servoing (PBDVS). This control scheme uses the estimated pose and velocity of the robot platform to compensate for the robot dynamics and regulate the system. The second control scheme regulates the system directly in the image. The experimental results show that the proposed visual servo control law has globally better performances than the classical joint-based dynamic control. These results strengthen the idea that vision-based control can perform better than joint-based control. Note also that the proposed control laws are not limited to the sequential acquisition of ROIs but can be used with any high-speed acquisition system since it is possible to consider successive acquisitions to implement the proposed algorithms.

This paper opens a large span of research topics. The first is to re-project the different development in the 'look-and-move' approach into the dynamic control. Different control laws using different visual features could be proposed. Another research topic is to design dynamic vision-based control laws that take into account and compensate for more dynamic phenomena such as flexibilities, backlashes and other mechanical defects. Considering parallel robot control problem, it appears that it is also relevant to control this class of robots from leg observation (Dallej et al. 2007; Özgür et al. 2011). One interesting perspective is thus to generalize this concept into dynamic control of a parallel robot from leg observation.

Notes

1. Typically, more than a second.
2. Acceleration is assumed constant in the reference frame (i.e. the camera frame).
3. This error in the image, which represents the difference between the actual and the desired trajectories, must not be mixed up with the reprojection errors in the process of estimating the pose and the velocity.

Funding

This work was supported by Région d'Auvergne through the Innov@pôle project and by the French ANR JCJC Project VIRAGO.

Acknowledgements

The authors would like to thank Wisama Khalil and his group for giving us access to the Orthoglide robot and providing us with their original joint-based computed-torque control source code.

References

Ait-Aider O, Andreff N, Lavest J and Martinet P (2006) Simultaneous object pose and velocity computation using a single view from a rolling shutter camera. In *Proceedings of the 9th European Conference on Computer Vision, ECCV'06*, Graz, Austria, vol. 2, pp. 56–68.

Birk R, Kelley J, Wilson L, Badami V, Brownell T, Chen N, et al. (1979) *General Methods to Enable Robots with Vision to Acquire, Orient and Transport Workpieces*. Technical report, University of Rhode Island.

Boye T and Pritschow G (2005) New transformation and analysis of a *N*-DOF LINAPOD with six struts for higher accuracy. *Robotica* 23: 555–560.

Callegari M, Palpacelli M and Principi M (2006) Dynamics modelling and control of the 3-RCC translational platform. *Mechatronics* 16: 589–605.

Chablat D and Wenger P (2003) Architecture optimization of a 3-dof translational parallel mechanism for machining applications, the orthoglide. *IEEE Trans Robotics Automation* 19: 403–410.

Chaumette F and Hutchinson S (2006) Visual servo control, part I: Basic approaches. *IEEE Robotics Automation Mag* 13: 82–90.

Chaumette F, Rives P and Espiau B (1991) Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. In *IEEE International Conference on Robotics and Automation*, Sacramento, CA, pp. 2248–2253.

Corke P and Good M (1996) Dynamic effects in visual closed-loop systems. *IEEE Trans Robotics Automation* 12: 671–683.

Cuvillon L, Gangloff J, de Mathelin M and Forgione A (2006) Towards robotized beating heart TECABG: assessment of the heart dynamics using high-speed vision. *Computer Aided Surg* 11: 267–277.

Dahmouche R, Ait-Aider O, Andreff N and Mezouar Y (2008) High-speed pose and velocity measurement from vision. In *Proceedings IEEE International Conference on Robotics and Automation (ICRA'08)*, Pasadena, CA, pp. 107–112.

Dahmouche R, Andreff N, Mezouar Y and Martinet P (2009) 3D pose and velocity visual tracking based on sequential region of interest acquisition. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'09)*, Piscataway, NJ, pp. 5426–5431.

Dahmouche R, Andreff N, Mezouar Y and Martinet P (2010) Efficient high-speed vision-based computed torque control of the orthoglide parallel robot. In *Proceedings IEEE International Conference on Robotics and Automation (ICRA'10)*, Anchorage, AK, pp. 644–649.

Dallej T, Andreff N and Martinet P (2007) Image-based visual servoing of the I4R parallel robot without proprioceptive sensors. In *International Conference on Robotics and Automation (ICRA'07)*, Rome, pp. 1709–1714.

Dasgupta B and Choudhury P (1999) A general strategy based on the newton-euler approach for the dynamic formulation of parallels manipulators. *Mech Machine Theory* 34: 801–824.

Dietmaier P (1998) The Stewart–Gough platform of general geometry can have 40 real postures. In *Advances in Robot Kinematics: Analysis and Control*. Dordrecht: Kluwer Academic Publishers, pp. 1–10.

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

Etoh T, Poggemann D, Kreider G, Mutoh H, Theuwissen A, Ruckelshausen A, et al. (2003) An image sensor which captures 100

consecutive frames at 1000000 frames/s. *IEEE Trans Electron Devices* 50: 144–151.
Gangloff J and de Mathelin M (2003) High-speed visual servoing of a 6 DOF manipulator using multivariable predictive control. *Advanced Robotics* 17: 993–1021.
Hutchinson SA, Hager GD and Corke PI (1996) A tutorial on visual servo control. *IEEE Trans Robotics Automation* 12: 651–670.
Iserles A, Munthe-Kaas H, Nørsett S and Zanna A (2000) Lie-group methods. *Acta Numerica* 9: 215–365.
Ishii I, Nakabo Y and Ishikawa M (1996) Target tracking algorithm for 1 ms visual feedback system using massively parallel processing. In *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2309–2314.
Kelly R, Carelli R, Nasisi O, Kuchen B and Reyes F (2000) Stable visual servoing of camera-in-hand robotic systems. *IEEE/ASME Trans Mechatron* 5: 39–48.
Khalil W and Dombre E (2002) *Modeling Identification and Control of Robots*. Paris: Hermes Penton Science.
Khalil W and Ibrahim O (2004) General solution for the dynamic modeling of parallel robots. In *IEEE International Conference on Robotics and Automation*, New Orleans, LA, pp. 3665–3670.
Khalil W and Ibrahim O (2007) General solution for the dynamic modeling of parallel robots. *J Intell Robotic Syst* 49: 19–37.
Kragic D and Vincze M (2009) Vision for robotics. *Found Trends Robot* 1: 1–78.
Marchand E and Chaumette F (2002) Virtual visual servoing: a framework for real-time augmented reality. *Eurographics* 21: 289–298.
Merlet J (2004) Solving the forward kinematics of Gought-type parallel manipulator with interval analysis. *Int J Robotics Res* 23: 221–235.
Merlet JP (2009) *Parallel Robots*. Dordrecht: Springer.
Mezouar Y and Chaumette F (2002) Path planning for robot image-based control. *IEEE Trans Robotics Automation* 18(4): 534–549.
Nabat V, de la O Rodriguez M, Company O, Krut S and Pierrot F (2005) PAR4: very high speed parallel robot for pick-and-place. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, pp. 553–558.
Özgür E, Bouton N, Andreff N and Martinet P (2011) Dynamic control of the Quattro robot by the leg edgels. In *IEEE International Conference on Robotics and Automation (ICRA'11)*, Shanghai, China, pp. 2731–2736.
Paccot F, Andreff A and Martinet P (2009) A review on the dynamic control of parallel kinematic machines: theory and experiments. *Int J Robotics Res* 28: 395–416.
Paccot F, Lemoine P, Andreff N, Chablat D and Martinet P (2008) A vision-based computed torque control for parallel kinematic machines. In *IEEE International Conference on Robotics and Automation*, pp. 1556–1561.
Samson C, Espiau B and Le Borgne M (1991) *Robot Control: The Task Function Approach*. Oxford: Oxford University Press.
Tani K, Abe M and Ohno T (1977) High precision manipulators with visual sense. In *Proceedings of the 7th International Symposium on Industrial Robots, SME and RIA*.
Ulrich M, Ribi M, Lang P and Pinz A (2004) A new high speed CMOS camera for real-time tracking application. In *IEEE International Conference on Robotics and Automation*, New Orleans, LA, pp. 5195–5200.
Vincze M (2000) Dynamics and system performance of visual servoing. In *IEEE International Conference on Robotics and Automation*, pp. 644–649.
Vincze M, Ayromlou M, Chroust S, Zillich M, Ponweiser W and Legenstein D (2002) *Dynamic Aspects of Visual Servoing and a Framework for Real-time 3D Vision for Robotics* (*Lecture Notes in Computer Science*, vol. 2238). Berlin: Springer, pp. 101–121.
Weiss L, Sanderson A and Neuman C (1987) Dynamic sensor-based control of robots with visual feedback. *IEEE J Robotics Automation* 3: 404–417.
Zhang H and Ostrowski JP (1999) Visual servoing with dynamics: Control of an unmanned blimp. In *IEEE International Conference on Robotics and Automation*, pp. 648–623.
Zhang J, Lumia R, Wood J and Starr G (2003) Delay dependent stability limits in high performance real-time visual servoing systems. In *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, vol. 1, pp. 485–491.

Appendix: Interaction matrix computation

A simple estimation of the interaction matrix can be obtained from the time derivative of the first-order approximation of the projection model (Dahmouche et al. 2009). This interaction matrix can be obtained in two steps: the computation of the interaction matrix of the 3D point then the computation of the interaction matrix of the 2D point.

The first-order approximation of the 3D point position in the camera frame can be written as follows:

$$ {}^{c}\mathbf{P}_i(t_i) \approx {}^{c}\mathbf{P}_i(t_0) + \Delta t_i {}^{c}\dot{\mathbf{P}}_i(t_0)\,. \tag{47} $$

The 3D point velocity at $(t + \Delta t_i)$ is obtained from the time derivative of (47):

$$ {}^{c}\dot{\mathbf{P}}_i \approx {}^{c}\dot{\mathbf{P}}_i(t_0) + \Delta t_i {}^{c}\ddot{\mathbf{P}}_i(t_0)\,. \tag{48} $$

The instantaneous velocity $({}^{c}\dot{\mathbf{P}}_i)$ and the acceleration $({}^{c}\ddot{\mathbf{P}}_i)$ of the 3D point can be written as a function of the object pose and velocity as follows:

$$ {}^{c}\dot{\mathbf{P}}_i(t) = {}^{c}\mathbf{v}_o + {}^{c}\omega_o \times {}^{c}\mathbf{OP}_i \tag{49} $$

$$ \begin{aligned} {}^{c}\ddot{\mathbf{P}}_i(t) = {}^{c}\dot{\mathbf{v}}_o + 2{}^{c}\omega_o \times {}^{c}\mathbf{v}_o + {}^{c}\dot{\omega}_o \times {}^{c}\mathbf{OP}_i \\ + {}^{c}\omega_o \times ({}^{c}\omega_o \times {}^{c}\mathbf{OP}_i) \end{aligned} \tag{50} $$

where ${}^{c}\mathbf{OP}_i$ is the position of the 3D point in the object frame represented in the camera frame.

Equations (49) and (50) can be rewritten in matrix form as a function of the kinematic and the dynamic twists and substituted into (48):

$$ \dot{\mathbf{P}}_i(t_i) = \mathbf{L}_{\mathbf{P}_i} \begin{bmatrix} {}^{c}\boldsymbol{\tau}_o \\ {}^{c}\dot{\boldsymbol{\tau}}_o \end{bmatrix} \tag{51} $$

where $\mathbf{L}_{\mathbf{P}_i}$ is the 3D point interaction matrix given by

$$ \mathbf{L}_{\mathbf{P}_i} = \left[\mathbf{L}_{3d_i} + \Delta t_i \boldsymbol{H}_{\mathbf{P}_i}, \Delta t_i \mathbf{L}_{3d_i}\right] \tag{52} $$

with

- $\Delta t_i = t_i - t_{\text{ref}}$;
- $\mathbf{L}_{3d_i} = \begin{bmatrix} \mathbf{I}, & -[{}^c\mathbf{OP}_i]_\times \end{bmatrix}$ is the well-known 3D point interaction matrix;
- $\boldsymbol{H}_{\mathbf{P}_i} = \begin{pmatrix} 2[{}^c\omega_o]_\times & [{}^c\mathbf{P}_i(t) \times {}^c\omega_o]_\times \end{pmatrix}$.

Thanks to the relation between the 2D and 3D point velocities given by ${}^{2d}\mathbf{J}_{3d_i} = \dfrac{\partial \mathbf{m}_i(t)}{\partial \mathbf{P}_i(t)}$, the 2D point interaction matrix $\mathbf{L}_i$ can be written as a function of the 3D point interaction matrix as follows:

$$\mathbf{L}_i = {}^{2d}\mathbf{J}_{3d_i} \ \mathbf{L}_{\mathbf{P}_i} = {}^{2d}\mathbf{J}_{3d_i} \begin{bmatrix} \mathbf{L}_{3d_i} + \Delta t_i \boldsymbol{H}_{\mathbf{P}_i}, \Delta t_i \mathbf{L}_{3d_i} \end{bmatrix} \quad (53)$$

where the well-known Jacobian ${}^{2d}\mathbf{J}_{3d_i}$ of the 2D perspective image point with respect to the 3D point ${}^c\mathbf{P}_i = (x_i, y_i, z_i)^{\mathrm{T}}$ is given by

$${}^{2d}\mathbf{J}_{3d_i} = \begin{bmatrix} \dfrac{1}{z_i} & 0 & -\dfrac{x_i}{z_i^2} \\ 0 & \dfrac{1}{z_i} & -\dfrac{y_i}{z_i^2} \end{bmatrix}. \quad (54)$$

An expression of $\mathbf{L}$ ($2n \times 12$) is built by stacking the individual interaction matrices $\mathbf{L}_i$, $i = 1 \ldots n$ associated with each point.