The 2009 IEEE/RSJ International Conference on
Intelligent Robots and Systems
October 11-15, 2009 St. Louis, USA

3D Pose and Velocity Visual Tracking Based on Sequential Region of Interest Acquisition

Redwan Dahmouche, Nicolas Andreff, Youcef Mezouar and Philippe Martinet
LASMEA - CNRS - Université Blaise Pascal
63175 Aubière, France
{firstname.lastname}@lasmea.univ-bpclermont.fr

Abstract— **This paper presents a high speed visual tracking method based on non simultaneous subimages acquisition. This method is formulated as a virtual visual servoing scheme. The sequential acquisition of regions of interest has a double benefit on visual servoing. The first one is that this acquisition method allows to increase the visual control sampling frequency by reducing the data amount to acquire and to transmit by the camera. The second one is that the associated image projection model depends on the observed object pose and velocity. Thanks to this property, a new vision-based control law can be defined. The particularity of this control law is that the control output consists of the kinematic and the dynamic twists. This allows to enhance the control performance in trajectory tracking applications. The experimental results in high speed visual tracking application show the effectiveness of this approach.**

I. INTRODUCTION

In the last years, the use of visual observations to control the motions of robots has been extensively studied (approach referred in the literature as visual servoing, see [1], [2] as a list of reading). Indeed, computer vision can provide the robotic system with a powerful way of sensing the environment and can potentially reduce or obliterate the need for environmental modeling, which is extremely important when the robotic tasks require the robot to move in unknown and/or dynamic environments. Typical applications of visual servoing are the positioning of a robot and the tracking of objects using the visual information provided by an "eye-in-hand" or "eye-to-hand" camera. Visual servoing techniques are very effective since they close the control loop over the vision sensor. This yields a high robustness to disturbances as well as to calibration errors. Several kinds of visual servoing can be distinguished, according to the space where the visual features were defined. For instance, in image-based visual servo (IBVS), the visual features are defined in the image space while in position-based visual servoing (PBVS) the features are defined in the 3D space [1].

Using motion features in visual servoing instead of their positions seems also to be an interesting approach. In [3] the focus of expansion and 2D affine transformation parameters are used in the task function to achieve the alignment of the optical axis of the camera with the translational direction. In [4] the signal used in the closed loop control is composed of the feature positions and their first derivatives. To improve the behavior of visual servoing schemes in unknown complex scenes, two control laws based on image motion integration were proposed in [5]. In the first one the positions of the features in the image are first estimated from the optical flow and then used in a classical image based visual servoing. In the second one the relationship between image motion derivative and camera velocity and acceleration is exploited to achieve a positioning task.

However, note that none of these features were designed to deal with robot dynamics, while it has been shown that the use of kinematic approach has some limitations from the control performance point of view [6], [7]. Indeed, kinematic approaches overlook the robot dynamics. This simplification assumes that the low level controller (PID, in general) is sufficient to compensate the dynamic effects, which is true only in low velocities. However, dynamic effort compensation in the control loop is necessary to efficiently achieve high speed robotic tasks since dynamic effects increase with the velocity [8]. Thus, one of the main next challenges in the field of image based visual servoing is to deal with dynamics to improve the control performances. The work presented in this paper is a step towards this goal.

The main drawback in the use of vision in dynamic control is that the sampling frequency of standard vision systems is low compared to dynamic control frequencies. In practice, vision systems run at 25-30Hz and may be increased up to 120Hz while a dynamic control loop is typically cadenced at about 1 kHz. To adapt the two sampling rates, a possible approach is to design a predictive controller where latency caused by the visual system compensated by designing predictive controler [9]. However, from the control theory point of view, it would be more suitable to increase the image acquisition frame-rate to make vision systems more adequate to dynamic control [6], [7]. Fortunately, this is possible thanks to recent high speed CMOS cameras that run at more than $1kHz$. However, camera video rate is usually limited by the video rate bandwidth of the transmission interface. A possible approach to overcome this bottleneck is to develop a faster communication interface or to embed the signal processing close to the vision sensor [10], [11].

However, another solution is possible, which consists in grabbing only the few regions of interest (ROI) where the

This work was supported by Région d'Auvergne through the Innov@pôle project, by the European Union through the Integrated Project NEXT no. 0011815 and by the French ANR through the VIRAGO project.

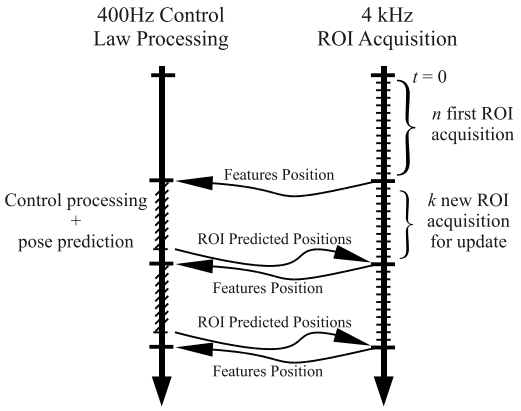

Fig. 1. High speed vision system chronogram where processing is based on the sequential acquisition of small sub-images containing the features.

effective information is located [12]. For visual servoing purposes, a single ROI can be selected around the whole object to be observed or several ROI can be selected around each visual feature. In the latter case, the minimal amount of pixels is grabbed and transmitted. Yet, grabbing several ROI simultaneously is expensive (more silicium) when standard off-the-shelf CMOS cameras can provide with one ROI at a time. This has the drawback of creating a time delay between the different visual feature acquisition. Nevertheless, in [13] the authors used this acquisition approach to provide pose and velocity estimation of a high speed moving object using a numerical optimization.

This paper replaces the authors work above in the visual servoing framework. Indeed, as shown in [14], pose estimation can be expressed as virtual visual servoing if one takes into account the specific properties of the special Euclidean group SE(3). The contribution of this paper is thus a novel visual control law based on the spatio-temporal projection model for a rigid object associated to the sequential grabbing. This control scheme enables to achieve an effective compensation of the tracking error which improves the performance of the visual servoing. From the practical point of view, the ROI grabbing implemented using multithread programming, where acquisition and processing run simultaneously in two different threads, allows to achieve a 4kHz ROI acquisition frequency and increases the vision control sampling frequency up to 400Hz (Fig. 1).

In the next section is briefly recalled the projection model when using the ROI acquisition method. Section II presents the proposed visual servoing approach. The interaction matrix related to the proposed control law is presented in section III. Finally, section IV describes the experimental setup and presents the obtained results.

II. Pose and velocity tracking

This section presents a framework which can be used in visual servoing, pose and velocity estimation and tracking. Indeed, it has been shown that all these applications share the same theoretical formalism [15], [14]. The visual servoing principle consists in moving the robot end-effector from an arbitrary initial pose to the desired one defined by features position in the image. The objective is to minimize the projection errors between the actual and desired feature configurations. The control law defines the Cartesian velocities to allow this minimization. In pose estimation through virtual servoing, the idea is to use the visual servoing control law to move a virtual (simulated) camera from the actual to the desired configuration. The difference between the real and the virtual visual control is that in the real visual servoing, the current features configuration is observed through the sensor and the desired features positions (or trajectories) defined in the image while in the virtual visual servoing, the desired features positions are defined by the image grabbed by the camera and the current features configurations are estimated using the projection model from the current estimated pose. In this case, the intrinsic camera parameters and the observed object model must be known. The proposed framework is formulated here in the context of visual tracking of fast object with pose and velocity estimation. The same formulation can be easily used in the context of visual servoing.

First, let us recall the pinhole projection model of a set of n rigidly linked points acquired at different time instants [16], [13]. The Cartesian coordinates of the set of points with respect to the object frame is noted ${}^o\mathbf{P}_i$, $\forall i = 1..n$. Their image projections in the camera frame is noted $\mathbf{m}_i = (u_i, v_i)^{\mathrm{T}}$. The associated homogeneous representations of ${}^o\mathbf{P}_i$ and $\mathbf{m}_i$ are respectively noted $\tilde{\mathbf{m}}_i = \left(\mathbf{m}_i^{\mathrm{T}}, 1\right)^{\mathrm{T}}$ and $\tilde{\mathbf{P}}_i = \left(\mathbf{P}_i^{\mathrm{T}}, 1\right)^{\mathrm{T}}$. The time varying projection model of the rigid set of points where each feature is grabbed at $t = t_i$ can be written as a function of the configuration of the object at the time reference $({}^c\mathbf{R}_o,\ {}^c\mathbf{t}_o)$ the and its rotation ${}^c\mathbf{R}_{o_i}$ and translation ${}^c\mathbf{t}_{o_i}$ displacements between the reference and the grabbing times [16], [13]:

$$\forall i = 1..n,\ s_i\ \tilde{\mathbf{m}}_i(t_i) = \mathbf{K}\left({}^c\mathbf{R}_{o_i} \quad {}^c\mathbf{t}_{o_i}\right){}^o\tilde{\mathbf{P}}_i \tag{1}$$

where $\mathbf{K}$ is the intrinsic camera parameters matrix, and s_i is a scale factor which depends on the 3D point depth. The lens distortion is assumed to be compensated for.

In high speed vision, the ROI acquisition period can be assumed small enough to consider the target object velocity as piecewise constant. The object diplacement can then be modeled as a time integral of the frame translation and rotation velocities in the 3D space. Considering the translation velocity as constant in the reference frame (camera frame), the translation of the object can simply be integrated as:

$${}^c\mathbf{R}_o\ {}^o\delta\mathbf{t} = {}^c\delta\mathbf{t} = \int_{t_0}^{t_i} {}^c\mathbf{V}\ dt = {}^c\mathbf{V}\ \Delta t_i \tag{2}$$

and the rotation defined by Rodrigues formula:

$${}^o\delta\mathbf{R}_i = \mathbf{I} + \frac{\sin(\|{}^o\boldsymbol{\omega}\|\,\Delta t_i)}{\|{}^o\boldsymbol{\omega}\|}\,[{}^o\boldsymbol{\omega}]_\times + \frac{1-\cos^2(\|{}^o\boldsymbol{\omega}\|\,\Delta t_i)}{\|{}^o\boldsymbol{\omega}\|^2}\,[{}^o\boldsymbol{\omega}]_\times^2 \tag{3}$$

where ${}^c\mathbf{V}$ and ${}^o\boldsymbol{\omega}$ are respectively the translation and the rotation velocities of the object frame and Δt_i the integration time.

As proposed in [17], one has to define a task function which represents the objective criterion to minimize in the

control loop. Thus, let us define the task function as follows:

$$\mathbf{e} = \mathbf{C}\left(\mathbf{m}(\mathbf{r}, \tau_o) - \mathbf{m}^*(t)\right) \tag{4}$$

where $\mathbf{m}(\mathbf{r}, \tau_o)$ is the vector of the projected features in the image plane which depends on the object pose $\mathbf{r} \in SE(3)$ in the camera frame and $\tau_o = [{}^o\mathbf{V}_o^T, {}^o\boldsymbol{\omega}_o^T]^T$ its kinematic twist. $\mathbf{m}^*(\mathbf{t})$ is the desired features positions and $\mathbf{C}$ is the combination matrix. Note that classically the task function is defined as a vector with 6 entries to constraint the 6 *dofs*. However, the used projection model depends on 12 parameters (6 for the pose and 6 for the velocity). This means that the task function is a vector with 12 entries, and that the combination matrix is $12 \times 2n$ where n is the number of features.

From (4), the time derivative of the task function can be written as:

$$\dot{\mathbf{e}} = \frac{\partial \mathbf{e}}{\partial \mathbf{r}} \frac{d\mathbf{r}}{dt} + \frac{\partial \mathbf{e}}{\partial \tau_o} \frac{d\tau_o}{dt} + \frac{\partial \mathbf{e}}{\partial t} \tag{5}$$

The previous equation can be written as a function of the kinematic and dynamic twists as follows:

$$\dot{\mathbf{e}} = \left[\frac{\partial \mathbf{e}}{\partial \mathbf{r}} \mathbf{A}, \quad \frac{\partial \mathbf{e}}{\partial \tau_o} \right] \begin{bmatrix} \tau \\ \dot{\tau} \end{bmatrix} + \frac{\partial \mathbf{e}}{\partial t} \tag{6}$$

where the matrix $\mathbf{A}$ depends on the choice of the pose parameters (Euler, Quaternions, etc) [18]. As we will see in the sequel, we do not have to estimate it in practice since the interaction matrix is directly obtained from kinematic equations.

Assuming that the combination matrix is locally constant, the task function time derivative can be written as a function of the kinematic and dynamic twists as follows:

$$\dot{\mathbf{e}} = \mathbf{C}\mathbf{L}_{e2d} \begin{bmatrix} \tau \\ \dot{\tau} \end{bmatrix} - \mathbf{C} \frac{d\mathbf{m}^*(t)}{dt} \tag{7}$$

where $\mathbf{L}_{e2d}$ is the $2n \times 12$ interaction matrix that relates the velocities of the n image features to the object kinematic and dynamic twists.

A first order exponential decrease of the task function can be obtained by imposing:

$$\dot{\mathbf{e}} = -\lambda \ \mathbf{e} \tag{8}$$

where λ is a positive proportional gain which tunes the convergence speed.

The kinematic and dynamic twists which allow the convergence of the process are obtained from (7) and (8):

$$\begin{bmatrix} \tau \\ \dot{\tau} \end{bmatrix} = -\lambda \left(\mathbf{C}\mathbf{L}_{e2d}\right)^{-1} \mathbf{e} + \mathbf{C}\dot{\mathbf{m}}^*(t) \tag{9}$$

A classical choice of the combination matrix is $\mathbf{C} = \hat{\mathbf{L}}_{e2d}^+$, where $\hat{\mathbf{L}}_{e2d}$ is an estimate of the interaction matrix.

For $\mathbf{C} = \hat{\mathbf{L}}_{e2d}^+$ and assuming that $\left(\hat{\mathbf{L}}_{e2d}^+ \mathbf{L}_{e2d} \approx \mathbf{I}\right)$ the control output is computed as follows:

$$\begin{bmatrix} \tau \\ \dot{\tau} \end{bmatrix} = -\lambda \hat{\mathbf{L}}_{e2d}^+ (\mathbf{m}(\mathbf{r}, \tau_o) - \mathbf{m}^*(\mathbf{t})) + \hat{\mathbf{L}}_{e2d}^+ \dot{\mathbf{m}}^*(t) \tag{10}$$

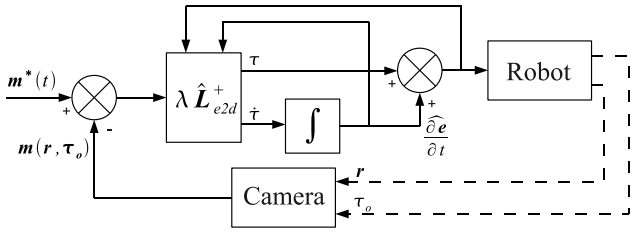

Fig. 2. Control scheme based on sequential ROI acquisition.

In this equation, the term $\hat{\mathbf{L}}_{e2d}^+ \dot{\mathbf{m}}^*(t) = \dfrac{\partial \hat{e}}{\partial t}$ represents the compensation of the tracking error induced by the target motion which is an estimation of the relative velocity and acceleration of the target. It is often considered as a disturbance but here, this disturbance can be estimated.

Indeed, in conventional visual servoing the computed kinematic twist represents a correction of the current configuration. If the process converges, the integration of the kinematic twist provides an estimation of the current pose. In the proposed method, the virtual control output provides a 12 element vector which is composed of the kinematic and the dynamic twists associated to the relative pose and velocity correction between the camera and the target. In the proposed virtual control law the integration of the kinematic twist provides an estimation of the current pose, but also the computed dynamic twist represents a correction of the current velocity. Thus, the integration of the dynamic twist represents an estimation of the current velocity. The relative target velocity with respect to the camera can then be estimated as follows:

$$\hat{\tau}^*(k) = \hat{\tau}^*(k-1) + \int_0^{T_e} \hat{\tau} \ dt \tag{11}$$

where T_e is the sampling period of the estimation process.

Since this velocity represents an estimation of the target velocity it can be used to compensate the tracking error induced by the target motion. On the other side, the projection model assumes a constant velocity of the moving target. The acceleration can thus be neglected. The tracking error compensation term can then be estimated by:

$$\widehat{\frac{\partial \mathbf{e}}{\partial t}} = \begin{bmatrix} \hat{\tau}^* \\ \mathbf{0} \end{bmatrix} \tag{12}$$

Finally, the estimated pose and velocity are used to update the interaction matrix (see Fig. 2).

III. Interaction matrix computation

Let us consider an object moving with a translation ${}^c\mathbf{V}_o$ and rotation ${}^c\boldsymbol{\omega}_o$ velocities with respect to the camera frame, and let $\mathbf{P}_i(t)$ be a point of this object. Note that the relative velocity of the object can be caused either by the camera motion or by the object motion. Let us consider in the sequel the case of a moving object. The integration of the object motion during Δt_i maps the 3D point from $\mathbf{P}_i(t)$ into $\mathbf{P}_i(t + \Delta t_i)$. Note that even if the kinematic twist of the moving object is assumed constant, the motion trajectory of the 3D point is not linear because of the non linear structure of $SE(3)$. A first order linearization of this motion model

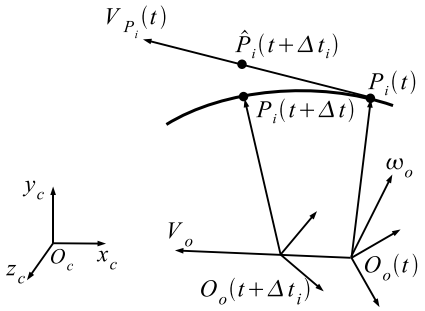

Fig. 3. First order linearization of the motion model.

consists in assuming the velocity of the 3D point constant during Δt_i. In this case, the integration of the 3D point velocity maps $\mathbf{P}_i(t)$ into $\hat{\mathbf{P}}_i(t+\Delta t_i)$ instead of $\mathbf{P}_i(t+\Delta t_i)$ (see Fig. 3).

A. 3D point interaction matrix

Let ${}^c\mathbf{P}_i$ be the representation of $\mathbf{P}_i$ in the camera frame. The first order linearization of the 3D point motion in the camera frame can be written as follows:

$${}^c\hat{\mathbf{P}}_i(t+\Delta t_i) = {}^c\mathbf{P}_i(t) + \Delta t_i \frac{d^c\mathbf{P}_i(t)}{dt} \tag{13}$$

The 3D point velocity at $(t+\Delta t_i)$ is obtained from the time derivative of (13):

$${}^c\dot{\hat{\mathbf{P}}}_i(t+\Delta t_i) = {}^c\dot{\mathbf{P}}_i(t) + \Delta t_i \frac{d^c\dot{\mathbf{P}}_i(t)}{dt} \tag{14}$$

where velocity and acceleration of the 3D point are given by:

$${}^c\dot{\mathbf{P}}_i(t) = {}^c\mathbf{V}_o + {}^c\omega_o \times {}^c\mathbf{OP}_i(t) \tag{15}$$

$$\begin{aligned} {}^c\ddot{\mathbf{P}}_i(t) &= {}^c\dot{\mathbf{V}}_o + 2{}^c\omega_o \times {}^c\mathbf{V}_o + {}^c\dot{\omega}_o \times {}^c\mathbf{OP}_i(t) \\ &+ {}^c\omega_o \times ({}^c\omega_o \times {}^c\mathbf{OP}_i(t)) \end{aligned} \tag{16}$$

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

Equations (15) and (16) can be rewritten in matrix form as a function of the kinematic and the dynamic twists of the object (${}^c\tau_o$ and ${}^c\dot{\tau}_o$) expressed in the camera frame:

$${}^c\dot{\mathbf{P}}_i(t) = \mathbf{L}_{3d}{}^c\tau_o \tag{17}$$

$${}^c\ddot{\mathbf{P}}_i(t) = \begin{bmatrix}\mathbf{H}_{3d_i},\ \mathbf{L}_{3d_i}\end{bmatrix} \begin{bmatrix} {}^c\tau_o \\ {}^c\dot{\tau}_o \end{bmatrix} \tag{18}$$

where $\mathbf{L}_{3d}$ is the classical 3D point interaction matrix:

$$\mathbf{L}_{3d_i} = \begin{bmatrix}\mathbf{I}\ , & -[{}^c\mathbf{OP}_i]_\times\end{bmatrix}$$

and $\mathbf{H}_{3d_i}$ relates the 3D point acceleration to the object velocity:

$$\mathbf{H}_{3d_i} = \begin{bmatrix} 2[{}^c\omega_o]_\times\ , & \left[[{}^c\mathbf{OP}_i(t)]_\times {}^c\omega_o\right]_\times \end{bmatrix}$$

By substituting (17) and (18) into (14), the 3D point velocity can then be written as a function of the object kinematic and dynamic twists:

$${}^c\dot{\hat{\mathbf{P}}}_i(t+\Delta t_i) = \mathbf{L}_{\widehat{e3d_i}} \begin{bmatrix} {}^c\tau_o \\ {}^c\dot{\tau}_o \end{bmatrix} \tag{19}$$

where $\mathbf{L}_{\widehat{e3d_i}}$ is the 3D point interaction matrix under the first order approximation is given by:

$$\mathbf{L}_{\widehat{e3d_i}} = \begin{bmatrix}\mathbf{L}_{3d_i} + \Delta t_i \mathbf{H}_{3d_i},\ \Delta t_i \mathbf{L}_{3d_i}\end{bmatrix} \tag{20}$$

B. 2D point interaction matrix

The 2D interaction matrix defines the relation between the 2D point motion and the kinematic and the dynamic twists:

$$\frac{d\mathbf{m}_i(t)}{dt} = \mathbf{L}_{\widehat{e2d_i}} \begin{bmatrix} {}^c\tau_o \\ {}^c\dot{\tau}_o \end{bmatrix} \tag{21}$$

This matrix can be trivially computed from the 3D interaction matrix as follows:

$$\frac{d\mathbf{m}_i(t)}{dt} = \frac{\partial \mathbf{m}_i(t)}{\partial {}^c\mathbf{P}_i(t)} \frac{d^c\mathbf{P}_i(t)}{dt} = \frac{\partial \mathbf{m}_i(t)}{\partial {}^c\mathbf{P}_i(t)} \mathbf{L}_{\widehat{e3d_i}} \begin{bmatrix} {}^c\tau_o \\ {}^c\dot{\tau}_o \end{bmatrix} \tag{22}$$

By identification of the previous equation with (21), the 2D interaction matrix can be written as:

$$\mathbf{L}_{\widehat{e2d_i}} = \frac{\partial \mathbf{m}_i(t)}{\partial {}^c\mathbf{P}_i(t)} \mathbf{L}_{\widehat{e3d_i}} \tag{23}$$

where the term $\dfrac{\partial \mathbf{m}_i(t)}{\partial {}^c\mathbf{P}_i(t)}$ provides the matrix relation between the feature position in the image and the Cartesian coordinates of the 3D point:

$$\frac{\partial \mathbf{m}_i(t)}{\partial {}^c\mathbf{P}_i(t)} = \begin{bmatrix} \alpha_u & \alpha_{uv} \\ 0 & \alpha_v \end{bmatrix} \begin{bmatrix} \dfrac{1}{z_i} & 0 & -\dfrac{x_i}{z_i^2} \\ 0 & \dfrac{1}{z_i} & -\dfrac{y_i}{z_i^2} \end{bmatrix} \tag{24}$$

α_u, α_u and α_{uv} being the scale factors of the intrinsic camera parameters matrix and $(x_i, y_i, z_i)^{\mathrm{T}}$ the 3D point coordinates in the camera frame.

IV. Experimental results

The virtual visual servoing has been implemented in C++ using Numerical Template Toolbox NT_2 [19] on an Intel P4 processor PC. The acquisition process is performed by a "Photon focus Track Cam" based on ROI grabbing method. The acquisition mode consists in selecting a Region Of Interest (ROI) that contains one blob at each sampling time. After a sufficient number of acquisitions, the control process can be launched (Fig. 1). The position of the next k object points can then be predicted using the control output, and the ROI positions are set to center the predicted features. The visual pattern used in this experiment contains $n = 16$ reflective circular blobs detected by computing the first order image moment of the thresholded ROI. Using more points than the minimum number (6 points) allows to reduce the measure noise effect and reduce the probability of degeneracy. However, to identify the degenerative configurations, the interaction matrix has to be studied, which have not been done yet. Nevertheless, no degeneracy has occurred during the many simulations and experiments already performed.

To increase the number of ROI updated at each control step, the acquisition and estimation algorithms run in two different threads (as opposed to [13] where a single thread was running). The main thread is synchronized to the robot

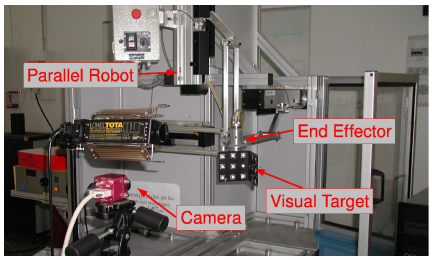

Fig. 4. Experimental setup: trajectory tracking of a visual pattern mounted on a high speed parallel robot end-effector.

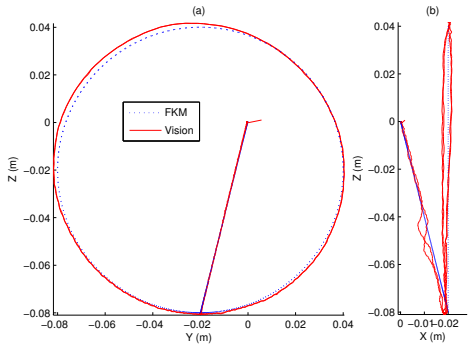

Fig. 5. Robot end-effector motion estimation obtained by virtual control and by the reconstruction of Cartesian trajectory using the Forward Kinematic Model of the robot. (a) Trajectories projection on YZ plane. (b) Trajectories projection on XZ plane.

controller by an external trigger that delivers a 4kHz acquisition frequency signal. At each ROI acquisition the position of the blob is stored in a shared buffer between the two threads. The control process which runs in another thread is requested each $k = 10$ ROI acquisition. It thus runs at 400Hz.

For the validation of the proposed approach, an experiment of virtual visual servoing of a high speed moving object in "eye to hand" configuration was performed (Fig. 4). In this experiment the motion of a visual pattern mounted on a parallel robot ("Orthoglide" [20]) end-effector is estimated using the presented control law. The virtual visual servoing and the robot controller in the joint space are synchronised at 400 Hz. The robot trajectory is composed of a 5th order interpolated linear and circular displacements (Fig. 5). In the first phase, the robot moves from the initial position to the circular trajectory starting point following a linear trajectory of about $8cm$. In the second phase, the robot executes two circular trajectories of $6cm$ radius. Finally, the end-effector returns to its initial position following a linear path. During the full trajectory execution, the highest speed and acceleration reached by the robot are respectively about $1m/s$ and $5m/s^2$ which represents the maximal tangential acceleration of the robot. The normal acceleration due to the circular trajectory is $16.67m/s^2$. The initialization of the process consists in performing a static pose estimation and setting the velocity as zero. After the identification of the homogeneous transformation between the robot frame and the camera frame, the motion obtained from vision and from the FKM can be both represented in the same frame.

Figure 5 shows the 3D trajectories obtained from robot joints sensors and from vision, projected on the YZ (Fig. 5a) and XZ (Fig. 5b) planes. At first glance, the two trajectories seem to fit pretty well. However, one can note that the vision based trajectory estimation is not a perfect circle because of the drifts that appear in the diagonal directions. The difference between the two estimated trajectories is due to an error which may be attributed either to the model based estimation (because of flexibilities and the backlashes) or to the vision based estimation. Actually, a manual effort applied to the end-effector (on which the visual pattern is mounted), when the robot is static (Brakes engaged), reveals flexibilities and backlashes of about $\pm 4mm$ in translation and about ± 5 deg in rotation. Since the Forward Kinematic Model does not take into account flexibilities nor backlashes, the drifts do not appear in the 3D trajectory computed from joints sensors. During the circular trajectory execution, a centripetal force is applied to the robot which displace the tool tip outside the circle. Note also that in the vision estimated trajectory, the two laps are correctly overlapped which validates the repeatability of the estimation.

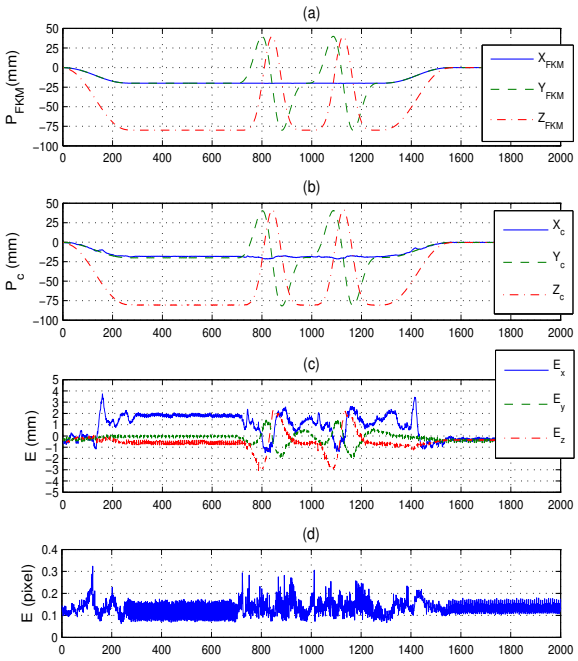

Fig. 6. (a) End effector Cartesian position measure from robot joint sensor using the Forward Kinematic Model. (b) 3D motion obtained by visual trajectory tracking. (c) Position error between the two trajectories. (d) Residual image errors normalized by the number of points n.

Figure 6a shows the trajectory reconstructed from the joint values and Fig. 6b the trajectory obtained by the vision system. The two trajectories are smooth and seem to be alike. One can note that X_c variable is the most noisy. In fact, the position obtained from vision is expressed in the robot frame and the X axis of the robot is coincident with the Z axis of the camera frame which correspond to depth (the most noisy direction in vision). The error between the two trajectories is shown in Fig. 6c where one can see that the position error at the initial and the final position is less than one millimeter while it is more important in the rest of the trajectory, even when the target is static (sample $i \in [300,\ 700]$). The fact that position errors as well as the image error (Fig. 6d) at

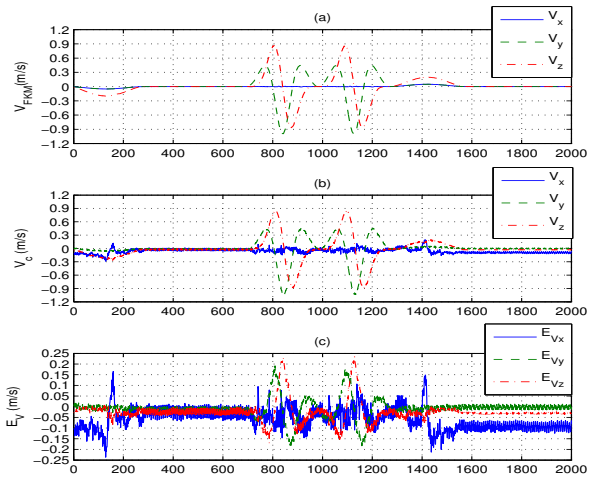

Fig. 7. (a) End effector velocity estimated by numerical derivation of the Forward Kinematic Model. (b) Vision based estimated velocities. (c) Velocity error.

the beginning and at the end of the trajectory is almost the same means that there is no error accumulation over time during the tracking. In addition, contrarily to the position errors, image errors in statics are characterized by the same mean (0,1287 pixel) and standard deviation (0.0241 pixel) in the two configurations corresponding to the beginning of the circular trajectory ($i \in [300,\ 700]$) and the end of the whole trajectory ($i \in [1600,\ 2000]$). Further more, comparing between the obtained Cartesian and image error through error propagation indicates that the positions error is much larger (about 10 times) than the corresponding image error. All this confirms that the errors are most probably on the FKM side.

Figure 7a shows the end-effector velocity obtained by numerical derivation of the Cartesian trajectory and Fig. 7b presents the vision based computed velocity. The two curves match. However, the cross-correlation function of the two signals reveals that the vision based velocity estimation is subject to a small delay of about 3 samples. This delay affects velocity tracking as shown in Fig. 7c where one can see that for each axis, velocity error peaks coincide with a null velocity and a high tangent acceleration. Velocity is then affected by acceleration and not by high speed, which is expectable because of the piecewise constant velocity assumption of the motion model. To reduce this delay, one can increase the scale factor λ to increase the convergence speed. However, a high gain value make the system more responsive to disturbances. To have an optimal response, a trade off has to be made.

V. Conclusion

This paper presented a new concept of high speed visual servoing. The first advantage of this approach is that the grabbing method reduces the image data acquired by the camera without loss of the useful information. This allows to increase the acquisition frequency (4kHz in the presented experiments) and improve the vision system efficiency. In addition, the sequential acquisition provides the velocity estimation which is used to improve tracking. Since a rather accurate pose and velocity are available at high sampling rate, this vision system can be used either in estimation (kinematic identification, of instance) or tracking. The application of this method to the effective control of the Orthoglide is under development.

Acknowledgements

The authors would like to warmly thank the IRCCyN and particularly Damien Chablat for having provided the robot "Orthoglide" which allowed the validation of this work.

References

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

[2] F. Chaumette and S. Hutchinson. Visual servo control, part II: Advanced approaches. *IEEE Robotics and Automation Magazine*, 14(1):109–118, March 2007.

[3] V. Sundareswaran, P. Bouthemy, and F. Chaumette. Exploiting image motion for active vision in a visual servoing framework. *Int. Journal of Robotics Research*, 15(6):629–645, December 1996.

[4] P. Martinet, F. Berry, and J. Gallice. Use of first derivative of geometric features in visual servoing. In *IEEE Int. Conf. on Robotics and Automation*, pages 22–28, April 1996.

[5] A. Crétual and F. Chaumette. Visual servoing based on image motion. *Int. Journal of Robotics Research*, 20(11):857–877, Novembre 2001.

[6] M. Vincze. Dynamics and system performance of visual servoing. In *IEEE Int. Conf. on Robotics and Automation*, pages 644–649, April 2000.

[7] P.I. Corke. Dynamic issues in robot visual-servo systems. In *Int. Symp. on Robotics Research ISRR95*, pages 488–498. Springer, 1995.

[8] P.I. Corke and M.C. Good. Dynamic effects in visual closed-loop systems. *IEEE Transaction on Robotics and Automation*, 12(5):671–683, October 1996.

[9] J. Gangloff and M. de Mathelin. High-speed visual servoing of a 6 DOF manipulator using multivariable predictive control. *Advanced Robotics. Special issue: advanced 3D vision and its application to robotics*, 17(10):993–1021, December 2003.

[10] Y. Nakabo, M. Ishikawa, H. Toyoda, and S. Mizuno. 1ms column parallel vision system and its application of high speed target tracking. In *IEEE Int. Conf. Robotics and Automation*, pages 650–655, San Francisco, USA, April 2000.

[11] P. Chalimbaud and F. Berry. Embedded active vision system based on an FPGA architecture. In *EURASIP Journal on Embedded Systems*, 2007.

[12] M. Ulrich, M. Ribi, P. Lang, and A. Pinz. A new high speed CMOS camera for real-time tracking application. In *IEEE Int. Conf. on Robotics and Automation*, New Orleans, april 2004.

[13] R. Dahmouche, O. Ait-Aider, N. Andreff, and Y. Mezouar. High-speed pose and velocity measurement from vision. In *Proc. IEEE Int. Conf. Robotics and Automation*, pages 107–112, May 2008.

[14] E. Marchand and F. Chaumette. Virtual visual servoing: A framework for real-time augmented reality. In G. Drettakis and H.-P. Seidel, editors, *EUROGRAPHICS 2002 Conference Proceeding*, volume 21(3) of *Computer Graphics Forum*, pages 289–298, Saarebrcken, Germany, September 2002.

[15] E. Malis and S. Benhimane. A unified approach to visual tracking and servoing. *Robotics and Autonomous Systems*, 52(1):39–52, 2005.

[16] O. Ait-Aider, N. Andreff, J.M. Lavest, and P. Martinet. Simultaneous object pose and velocity computation using a single view from a rolling shutter camera. In *Proceedings of the 9th European Conf. on Computer Vision*, volume 2, pages 56–68, Graz, Austria, May 7-13 2006.

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

[18] W. Khalil and E. Dombre. *Modeling identification and control of robots*. Hermes Penton Science, 2002.

[19] J. Falcou and J. Serot. E.V.E., an object oriented SIMD library. *Scalable Computing: Practice and Experience*, 6(4):31–41, December 2005.

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