Efficient high-speed vision-based computed torque control of the orthoglide parallel robot.

Redwan Dahmouche, Nicolas Andreff, Youcef Mezouar, Philippe Martinet

2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

Efficient High-speed Vision-based Computed Torque Control of the Orthoglide Parallel Robot

Redwan Dahmouche [1][3], Nicolas Andreff [1][2], Youcef Mezouar [1] and Philippe Martinet [1][3]

[1] {firstname.lastname}@lasmea.univ-bpclermont.fr
[2] {firstname.lastname}@femto-st.fr

***Abstract*— Vision has often been considered as not suitable for dynamic control of robots. The experimental results presented in this paper show that it is possible to perform better with a vision based dynamic control than with a model-based control. These results were obtained using a Cartesian computed torque control fed back, without any joint sensing, by a novel Cartesian pose and velocity estimator. The latter is designed as a virtual visual servoing scheme based on sequential acquisition of sub-images and a constant acceleration motion assumption.**

I. Introduction

It was shown in [1], [2] that the most appropriate space for parallel robot (or parallel kinematic manipulator) control is the Cartesian space. The main reason for this is that, contrarily to serial robots, the most natural representation space for parallel robots is the Cartesian configuration of the end-effector [1], [3], [4]. In addition, the dynamic coupling between legs in such robots being important even at low speed [2], it imposes to compensate for dynamics in the control law as soon as high performances are expected [5].

The most natural expression for the dynamic control, taking into account the a priori knowledge over the dynamic model, is thus a Cartesian space computed torque control, as depicted in Fig. 1. As can be seen on the latter, the key point in implementing a Cartesian space computed torque control consists in how to obtain the robot Cartesian pose $\mathbf{x}$ and its time derivative $\dot{\mathbf{x}}$. The two main approaches to get these measures differ by the kind of the sensors used to measure the configuration of the robot, that can either be proprioceptive or exteroceptive sensors. The implementation of the Cartesian space computed torque control using proprioceptive sensors (namely, joint sensors) requires the computation of the forward kinematic model. However, the latter is usually not trivial to compute (a huge literature is devoted to this problem, which can be entered through [6]). In addition,

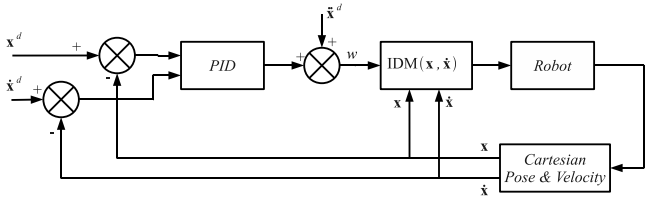

Fig. 1. Cartesian space computed torque control scheme of parallel robots.

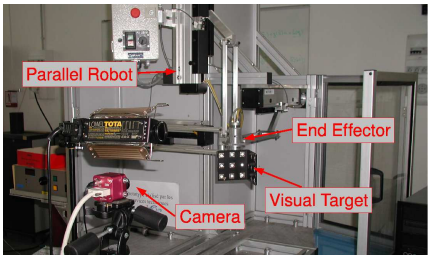

Fig. 2. Set-up for vision based control of parallel robot "Orthoglide".

it is subject to modeling and numerical errors [5] since the reliability of the end-effector pose estimation depends on to the completeness of the modeling of the robot geometry and to the identification accuracy of this model (calibration). Moreover, the pose is estimated using the minimal number of data, making it sensitive to the slightest noise in the joint sensors. Last, and maybe least, the end-effector pose velocity can either be obtained, in this case, by numerically differentiating the estimated pose over time or by feeding the differential kinematic model with joint velocities (whatever the way the latter are obtained).

One solution to these issues is to use an exteroceptive sensor. Indeed, measures provided by such a sensor are independent from the modeling and/or calibration of the mechanical system. However, these measures must reach an acceptable accuracy as well as an acceptable frequency. Optical sensing can satisfies these requirements. Furthermore, it is contactless and fairly insensitive to changes in an industrial environment. Among optical sensing, vision is, to a growing extent, also preferable to laser tracking since it is mechanically passive.

The feasibility of this approach was shown in [7] where a computed torque control of parallel robot based on high-speed vision was performed. However, the pose was obtained from the simple Dementhon algorithm [8] and was too noisy

[1] LASMEA - CNRS - Université Blaise Pascal, 24 avenue des Landais, 63175 Aubière, France.
[2] FEMTO-ST - CNRS - 32 avenue de l'observatoire, 25044 Besançon, France.
[3] IFMA - Institut Français de Mécanique Avancée, Campus de Clermont-Ferrand les Cézeaux - BP265, 63175 Aubière.

This work was supported by Région d'Auvergne through the Innov@pôle project and by the french ANR JCJC Project VIRAGO. The authors would like to warmly thank people at IRCCyN and more specifically Damien Chablat and Wisama Khalil for having given us access to the Orthoglide.

to get conclusive results for the vision-based control over joint sensor-based control, although the former held the comparison with the latter. In addition, the velocity was obtained by numerical time derivation of the pose, which expectedly multiplies the noise by the high sampling frequency of the dynamic control. Using low-pass filter for reducing noise is not appropriate though because of the phase lag introduced by the filter, which may affect the stability of the closed loop system [9], [10]. The scientific and technical bolt of this control law is thus conditionned by the achievement of a fast and accurate pose and velocity estimation of the parallel robot end-effector.

In [11], a vision system based on a sequential acquisition of selected regions of interest (namely, sub-images just large enough to contain the visual information) was used to have a high-speed pose and velocity computation. This vision system has two main benefits. The first one is that this acquisition strategy allows one to reduce the data amount to be transmitted from the camera to the process unit and then to consequently increase the acquisition frequency by using the communication interface bandwidth in a more efficient way. The second benefit is that the sequential acquisition of the visual features introduces visual motion-related artifacts in the whole image, thanks to which the end-effector velocity can be estimated without any numerical derivation.

However, that work was based on the assumption that the end-effector velocity was constant during the sequential acquisition of the whole image features. As a consequence of this constant velocity assumption, the velocity is estimated with a constant delay because of the velocity tracking error. Due to the non-linearity of the dynamic model, it is hard to take into account such a delay, resulting in loss of control stability [9], [10]. In addition, the acquisition frequency being higher than the control frequency, the assumption of a constant velocity is not coherent with the dynamic control since the motion model is closer to a constant acceleration model.

The contribution of this paper is to propose a novel high-speed vision-based computed torque control of parallel robots using the sequential sub-images acquisition method. To do so, the estimation method for Cartesian pose and velocity presented in [11] is improved and adapted to dynamic control purposes. In this context, the motion and the projection models are extended to a constant acceleration assumption to be more coherent, as stated, with the real robot motion. This modification also allows one to eliminate the velocity tracking delay, making the system more stable.

The following section presents the theoretical background of the virtual visual servoing for simultaneous pose and velocity estimation. Section III is devoted to adapting the pose and velocity estimation method to a piecewise constant acceleration motion assumption. The proposed control scheme is then presented in Section IV whereas Section V presents the experimental setup on the Orthoglide robot (Fig 2) and some implementation details. Finally, Section VI shows the experimental results using the proposed control law with comparison to a classical joint-based control scheme.

II. Pose and velocity estimation under the virtual visual servoing framework

Let us reformulate here some background results from [11] that are useful for the completeness of the paper.

In the sequential region of interest acquisition method, a known object, abstracted as a set of 3-d points, is observed by successively grabbing one single sub-image containing a single visual point at a time.

The projection model of a set of 3-d points $\mathbf{P}_i$ is thus given by:

$$\forall i = 1..n \quad \tilde{\mathbf{m}}_i \equiv [\mathbf{K} \mid \mathbf{0}] \ {}^c\mathbf{T}_o \ {}^c\delta\mathbf{T}_i \ {}^o\tilde{\mathbf{P}}_i \tag{1}$$

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, $\equiv$ 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 ${}^c\delta\mathbf{T}_i$ the displacement between t_{ref} and the i^{th} point acquisition time t_i. Finally, $\mathbf{K}$ is the matrix containing the camera intrinsic parameters, whilst lens distorsion is not shown here for clarity sake 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 ${}^c\delta\mathbf{T}_i$ is nothing but the integration of the object velocity between t_{ref} and t_i:

$${}^c\delta\mathbf{T}_i = \int_{t_{ref}}^{t_i} r(\boldsymbol{\tau}(t))dt \tag{2}$$

where $\boldsymbol{\tau} = [v, \omega]$ is the object velocity twist and r is the reshaping operator which transforms the kinematic twist into a 4×4 matrix.

Then, the estimation method consists essentially in minimizing the reprojection error built upon (1):

$$\min_{{}^c\mathbf{T}_o, \boldsymbol{\tau}} \frac{1}{2} \sum_{i=1}^{n} \|\mathbf{m}_i - \pi([\mathbf{K} \mid \mathbf{0}] \ {}^c\mathbf{T}_o \int_{t_{ref}}^{t_i} r(\boldsymbol{\tau}(t))dt \ {}^o\tilde{\mathbf{P}}_i)\|^2 \tag{3}$$

where $\pi(.)$ represents the non-linear formulation of perspective projection.

As it is non-linear, it is solved by an iterative numerical scheme. One elegant method, taking into account the specific structure of $SE(3)$, is to use the virtual visual servoing paradigm [12], [11]. This can be seen as an iterative scheme where the linearization is done in $se(3)$ rather than in $\mathbf{R}^6$. It is usually presented as taking the derivative of the above criterion with respect to time, but it should be presented as taking the derivative of the above with respect to a virtual time u upon which depend the minimization variables ${}^c\mathbf{T}_o = {}^c\mathbf{T}_o(u,t)$ and $\boldsymbol{\tau} = \boldsymbol{\tau}(u,t)$:

$$\frac{d\mathbf{m}_i}{du} = \frac{d}{du}\pi([\mathbf{K} \mid \mathbf{0}] \ {}^c\mathbf{T}_o(u,t) \int_{t_{ref}}^{t_i} r(\boldsymbol{\tau}(u,t))dt \ {}^o\tilde{\mathbf{P}}_i) \tag{4}$$

which can be shown to rewrite as follows [11]:

$$\frac{d\mathbf{m}_i}{du} = \mathbf{L} \begin{pmatrix} \boldsymbol{\tau}_u \\ \dot{\boldsymbol{\tau}}_u \end{pmatrix} \tag{5}$$

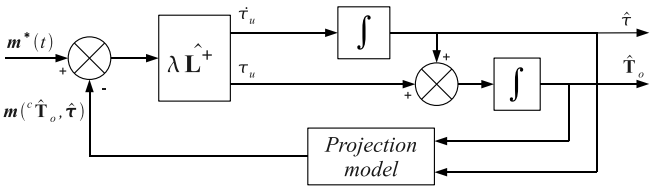

Fig. 3. Pose and velocity estimation under constant velocity assumption, using the virtual visual servoing paradigm.

where the subscript u indicates that the velocity and acceleration twists are virtual, i.e. do not correspond to actual 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 actual one.

An expression of $\mathbf{L}$, built by stacking the individual interaction matrices $\mathbf{L}_i, i = 1..n$ associated to each point, is:

$$\mathbf{L}_i = {}^{2d}\mathbf{J}_{3d_i}.\left(\mathbf{L}_{3d_i} + \Delta t_i \mathbf{H}_{3d_i} \quad \Delta t_i \mathbf{L}_{3d_i}\right) \tag{6}$$

where

- ${}^{2d}\mathbf{J}_{3d_i}$ is the well-known Jacobian of the 2d perspective image point with respect to the 3d point [11];
- $\Delta t_i = t_i - t_{ref}$;
- $\mathbf{L}_{3d}$ is the well-known 3D point interaction matrix;
- $\mathbf{H}_{3d_i} = \left(2\left[\omega\right]_\times \quad \left[{}^c\tilde{\mathbf{P}}_i(t) \times \omega\right]_\times\right)$

and where most of the above expressions depend on the coordinates ${}^c\tilde{\mathbf{P}}_i$ of the 3d point in the camera frame and, hence, on ${}^c\mathbf{T}_o$ and ${}^c\delta\mathbf{T}_i$.

From (5) and the error between the vectors $\mathbf{m}^*(t)$ and $\mathbf{m}({}^c\hat{\mathbf{T}}_o, \hat{\boldsymbol{\tau}})$ composed respectively of the stacked measured point images and of the corresponding stacked outputs of the projection model (1), one estimates the virtual velocity and acceleration updates that reduce the error:

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

where ${}^c\hat{\mathbf{T}}_o$ and $\hat{\boldsymbol{\tau}}$ are the previous estimates of ${}^c\mathbf{T}_o$ and $\boldsymbol{\tau}$. Since the velocity and the acceleration are not independent variables, integrating them separately to get the new pose and velocity estimates would yield a velocity tracking delay. In fact, the estimated acceleration update $\dot{\boldsymbol{\tau}}_u$ has to be used as a feedforward term that adds to the estimated velocity update as depicted in (Fig. 3).

III. Vision-based pose and velocity estimation under the constant acceleration assumption

The work in [11] made the assumption that the velocity was piecewise constant over the time interval during which the sub-images were grabbed. However, this assumption is not valid anymore for computed torque control. Indeed, under this control, the torques applied to the actuators are computed from the output of the inverse dynamic model, which is, in turn, fed with the pseudo-control vector made of the acceleration to apply to the end-effector. Thus, over a control period, the acceleration can be assumed constant (up to the internal regulation of the torques in the actuators).

Therefore, the above method needs be reformulated under the latter assumption, which essentially boils down to two things: computing ${}^c\delta\mathbf{T}_i$ in order to compute the projection model and the interaction matrix and updating the feedforward term.

Under the assumption that the acceleration is constant over the control sampling period, $\boldsymbol{\tau}(t)$ is itself the integral of the robot end-effector acceleration (actually, the dynamic twist $\dot{\boldsymbol{\tau}} = [\dot{v}, \dot{\omega}]$ of the platform expressed in the end-effector frame):

$$\boldsymbol{\tau}(t) = \boldsymbol{\tau}(t_{ref}) + \int_{t_{ref}}^{t} \dot{\boldsymbol{\tau}}\ dt, \forall t \in [t_{ref}, t_i[\tag{8}$$

and so (2) becomes

$${}^c\delta\mathbf{T}_i = \int_{t_{ref}}^{t_i} \mathbf{r}\left(\boldsymbol{\tau}(t_{ref}) + \int_{t_{ref}}^{t} \dot{\boldsymbol{\tau}}\ dt\right) dt \tag{9}$$

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

Assuming that the platform dynamic twist is constant between two successive control samples, the integration of the translational acceleration to obtain the object pose and velocity v_i can be written as:

$$v_i = v_{ref} + \sum_{k=0}^{i-1} \dot{v}_k\ T_a \tag{10}$$

Considering the dynamic twist constant in the camera frame and not in the moving object frame, for the reasons exposed in [11], one gets the translational part $\delta\mathbf{t}_i$ of $\delta\mathbf{T}_i$:

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

The rotation space being non linear, 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 and a uniformally variable norm over one control sampling period. The object velocity and rotation displacement are hence obtained by integrating the projection of the rotational acceleration $\dot{\omega}$ on the rotational velocity axis $\mathbf{u}_\omega$:

$$\omega_i = \omega_0 + \sum_{k=0}^{i-1} (\dot{\omega}_k \cdot \mathbf{u}_\omega)\ T_a\ \mathbf{u}_\omega \tag{12}$$

and

$$\delta\theta\mathbf{u}_i = \sum_{k=0}^{i-1} \left(\frac{1}{2}(\dot{\omega}_k \cdot \mathbf{u}_\omega)\ T_a^2\ \mathbf{u}_\omega + \omega_k T_a\right) \tag{13}$$

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) [13]:

$$\delta\mathbf{R}_i = \text{expm}([\delta\theta\mathbf{u}_i]_\times) \triangleq \exp(\delta\theta\mathbf{u}_i) \tag{14}$$

Finally, the expressions (11) and (14) are now exploitable both for being inserted into the projection model (1) and into the interaction matrix expression (6).

IV. Vision-based computed torque control of parallel robots

The control scheme (Fig. 4) is composed of a virtual visual servoing control loop and a dynamic control loop. The virtual visual servoing control loop task 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 computed torque control scheme of parallel robots.

Note that here, the pose and velocity estimator is launched at each control sample time t_j (see Fig. 5). Thus, the reference time t_{ref} is set at each control sample time to $t_{ref} = t_j$.

Note also that the update of the feedforward term of the pose and velocity estimator from the constant velocity assumption (Fig 3) to the constant acceleration (Fig. 4) consists in adding a feedforward acceleration term. In the proposed control loop, it is taken from the pseudo-control vector $\mathbf{w}$ of the computed torque control. Since the estimator hopefully takes less time than the control period, the integrators in the feed-forward term are reset at each control sample time and work all along the control period T_c.

Therefore, the update step for the pose and velocity estimator from one control sample to the other is given by:

$$\hat{\boldsymbol{\tau}}_{j+1} = \hat{\boldsymbol{\tau}}_j + \int_0^{T_c} (\dot{\boldsymbol{\tau}}_{u_j} + \boldsymbol{A}_j\ \mathbf{w}_j)dt \tag{15}$$

$$\hat{\mathbf{t}}_{j+1} = \hat{\mathbf{t}}_j + \frac{1}{2}\dot{v}_{\mathbf{w}_j}T_c^2 + \hat{v}_j\ T_c \tag{16}$$

$$\hat{\mathbf{R}}_{j+1} = \hat{\mathbf{R}}_j \exp\left(\frac{1}{2}\left(\dot{\omega}_{\mathbf{w}_j}\cdot\hat{\mathbf{u}}_\omega\right)T_c^2\hat{\mathbf{u}}_\omega + \hat{\omega}_j T_c\right) \tag{17}$$

where $\boldsymbol{A}_j$ is the transformation matrix of the acceleration control vector to a dynamic twist. $\dot{v}_{\mathbf{w}_j}$ and $\dot{\omega}_{\mathbf{w}_j}$ are the translational and rotational acceleration components of the control vector.

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.

V. Implementation

The proposed method was validated on the Orthoglide robot [14] with the set-up shown in Fig. 2. 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 (Fig. 6). The PC is taking care of the image acquisition and the pose and velocity estimation process,

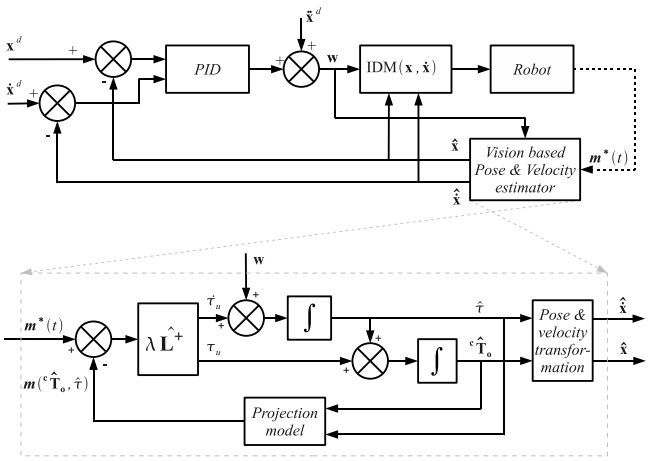

Fig. 4. The vision-based Cartesian computed torque control based on sequential image acquisition is composed of a standard Cartesian computed torque control (upper loop) and of a virtual visual servoing estimator (lower loop).

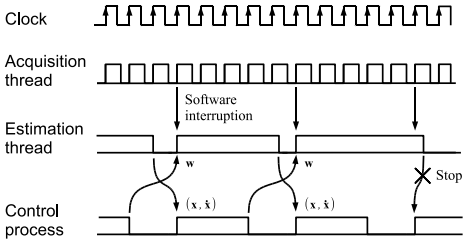

Fig. 5. Control chronogram

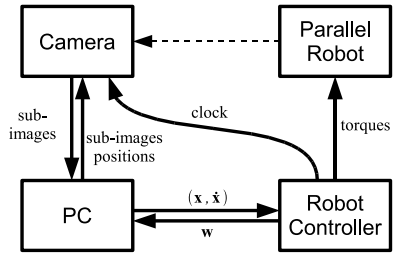

Fig. 6. Data flow in the implemented control architecture

while the DSP computes the computed 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 4kHz 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 all the points of the object (here, to the number of 16, which an empirical optimal number).

The robot end-effector pose and velocity are estimated by the PC and transmitted to the DSP card via the industrial RS-422 serial interface. Since the PC is constrained by the

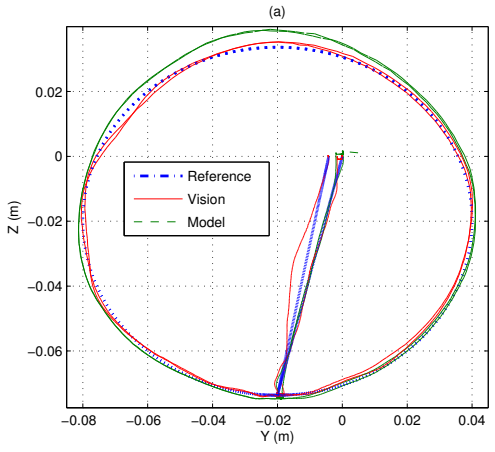

Fig. 7. Trajectories in space

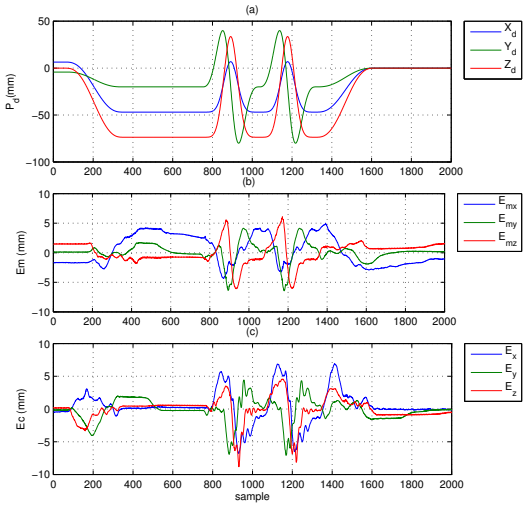

Fig. 8. Desired trajectory (top) with respect to time and errors for model-based (center) and vision-based (bottom) control

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 with the 400Hz robot controller frequency. The estimation process is thus only soft-real-time synchronized with the robot controller, so the latter contains a watchdog checking the correct reception of the estimates. Finally, the control vector processed by the robot controller is sent back to the estimation process on the PC through the RS-422 link.

VI. Experimental results

A. Robot stiffness and vision based estimation characterization

To have a relevant interpretation of the results, it is necessary to characterize both the vision system and the parallel

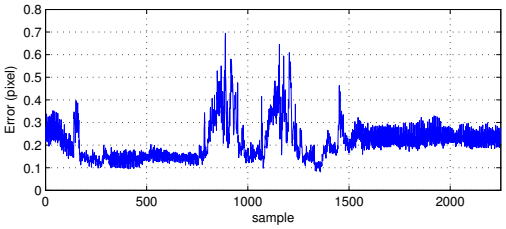

Fig. 9. Image reprojection error with respect to time

robot in terms of reliability and accuracy before proceeding to the implementation of the proposed control scheme. In lack of other exteroceptive measurements (interferometric laser, for instance), the only practical way to identify possible defaults of the two systems is by proceeding to some tests and confronting the results.

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: $stdev(\mathbf{t}) = [2.67,\ 4.05,\ 3.45]\ 10^{-5}\mathrm{m}$ and $stdev(v) = [2.04,\ 3.2,\ 5.75]\ 10^{-3}\mathrm{m/s}$.

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 small corresponding normalized projection residuals (0.19 pixel/point) indicates that this bias is small too (up to potential singularities which have not been encountered yet in practice).

Concerning the parallel robot, some joint flexibilities and backlashes have been noticed. To characterize the resulting motion of the platform, a manual effort was applied on it while the brakes are engaged and its displacement was measured with vision. The resulting displacement of the target in the Cartesian space is $\delta\mathbf{x} = [10.0\mathrm{mm},\ 8.1\mathrm{mm},\ 7.1\mathrm{mm},\ 2.38^{\circ},\ 2.65^{\circ},\ 2.93^{\circ}]^{\mathrm{T}}$ and the mean points displacement in the image is $[\delta u,\ \delta v] = [25.9,\ 20.3]$ pixels, which 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 benefit of exteroceptive sensors. After characterizing some properties of the the robot and vision system, the identification of the extrinsic camera parameters in the robot frame were achieved on smooth trajectories to avoid stimulating the robot flexibilities and backlashes. Yet, $3\mathrm{mm}$ residual errors remain.

B. Vision based computed torque control

After achieving this procedure, the proposed control law was implemented. Note that the use of the pose and velocity estimation method in [11] 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 estimation method presented here allows not only to stabilize the robot but also to control it to 100% of its speed. Though, since flexibilities and backlashes are measured, they are accounted for by the control law but this causes the robot to oscillate. To reduce this phenomena, the natural frequency of the closed loop system was decreased by scaling down the PID regulator gain by 20%.

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

Figure (7) shows the reference trajectory, the trajectory realized under vision-based computed torque control and the one achieved under model-based computed torque control. Both trajectories were recorded by vision, since the latter has proven more accurate than the model.

First of all, note that the vision-based control trajectory seems to be as smooth as the model-based control. One also notices that the model-based control trajectory radius seems to be bigger 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 respectively 61.34mm and 60.20mm. In addition, the algebraic distance between the trajectories to the reference circle (normal distance) is smaller for the vision-control 2.74mm than for the model-based control 3.25mm.

Figure (8) 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 where static errors in the vision-based control are smaller than model-based control and almost equivalent in dynamics. This is confirmed by the respective means and standard deviations of the tracking errors (Tab. I).

	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
std dev.	2.42	1.47	1.76	2.21	1.6	1.64

TABLE I

MEAN (IN MM) AND STANDARD DEVIATION (IN MM) OF THE MODEL-BASED AND VISION-BASED CONTROL

Figure (9) 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 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 so are the dynamic errors given in Tab. I certainly due to the latter.

VII. CONCLUSION

This paper presented a new vision-based computed torque control law. This control scheme uses a high-speed visual sensor to measure the pose and the velocity of the parallel robot platform in regulation and dynamic compensation. Measure being achieved in the task space, there is no need to compute the forward kinematic model which may include parametric errors or simplistic assumptions (no flexibilities, for instance). In addition, sensing the platform pose and velocity makes dynamic computation and compensation much easier than from joint reading. The presented results show that the presented vision based dynamic control law may be used even on a robot with flexibilities and backlashes and can even be more accurate than the model based control. The next step of this work is to design an image-based vision-based computed torque control. The combination of this approach with traditional joint space control schemes to control backlashes seems also to be a promising way.

REFERENCES

[1] M. Callegari, M.P. Palpacelli, and M. Principi. Dynamics modelling and control of the 3-rcc translational platform. *Mechatronics*, 16(10):589 – 605, 2006.

[2] F. Paccot, N. Andreff, and P. Martinet. A review on the dynamic control of parallel kinematic machines: Theory and experiments. *Int. J. Rob. Res.*, 28(3):395–416, 2009.

[3] B. Dasgupta and P. Choudhury. A general strategy based on the newton-euler approach for the dynamic formulation of parallels manipulators. *Mechanism and Machine Theory*, 34:801824, 1999.

[4] W. Khalil and O. Ibrahim. General sollution for the dynamic modeling of parallel robots. In *IEEE Int. Conference on Robotics and Automation*, pages 3665–3670, New Orleans, April 2004.

[5] T. Boye and G. Pritschow. New transformation and analysis of a n-dof linapod with six struts for higher accuracy. *Robotica*, 23(5):555–560, 2005.

[6] J.P. Merlet. Solving the forward kinematics of gough-type parallel manipulator with interval analysis. *The International Journal of Robotics Research*, 23:221235, 2004.

[7] F. Paccot, P. Lemoine, N. Andreff, D. Chablat, and Philippe Martinet. A vision-based computed torque control for parallel kinematic machines. In *IEEE International Conference on Robotics and Automation*, pages 1556–1561, April 2008.

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

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

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

[11] R. Dahmouche, N. Andreff, Y. Mezouar, and P. Martinet. 3d pose and velocity visual tracking based on sequential region of interest acquisition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, USA, October 2009. To appear.

[12] 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.

[13] A. Iserles, H.Z. Munthe-Kaas, S.P. Norsett, and A. Zanna. Lie-group methods. *Acta Numerica*, 9:215–365, 2000.

[14] 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.