Proceedings of the 2006 IEEE/RSJ
International Conference on Intelligent Robots and Systems
October 9 - 15, 2006, Beijing, China

3D Pose Visual Servoing Relieves Parallel Robot Control from Joint Sensing

Tej Dallej, Nicolas Andreff, Youcef Mezouar and Philippe Martinet
LASMEA - CNRS - Université Blaise Pascal/IFMA, 63175 Aubière, France
Email: {firstname.lastname}@lasmea.univ-bpclermont.fr
http://www.lasmea.univ-bpclermont.fr/Control

***Abstract*— In this paper, we show that visual feedback reduces the complexity of parallel robot Cartesian control. Namely, 3D pose visual servoing, where the end-effector pose is indirectly measured and used for regulation, is shown to be well suited to this task since it relieves the control from the difficult forward kinematic problem. Moreover, this complexity reduction is not coming with an increase of the implementation complexity since off-the-shelf hardware and software are now available for visual servoing. It is also shown that such a control gets rid of joint sensors. All this makes 3D pose visual servoing the most straightforward Cartesian control for parallel robots. Experimental results are provided using an open source visual servoing C++ library.**

I. INTRODUCTION

Controlling parallel robots is a hard task since joint motions are highly coupled due to the existence of closed kinematic chains. In our opinion, there has not been given yet any theoretically satisfying generic solution for their Cartesian control. To support this assumption, let us have an overview of the classical control schemes that are used in the literature.

First of all, it should be stated that this overview follows the way people chronologically derive control schemes usually, namely by following the increasing complexity order *for serial robots*. At the end of this overview, we hope the reader will be convinced that this order is not following the complexity increase *for parallel robots*.

The easiest control law for Cartesian positioning in serial robotics that can be ported to parallel robotics is joint control with Cartesian reference (Figure 1). The main and only advantage of this control both in serial and parallel robotics is to be easily implemented: no model is needed during control. However, even in the case of serial robots, it does not ensure convergence to the desired Cartesian pose, since the desired joint values are computed from the latter through the numerical inversion of the forward kinematic model. The final Cartesian error is thus very sensitive to the modeling and numerical errors. A way to get rid of such errors is to learn the joint values associated to the desired Cartesian pose. Now, in the case of parallel robots, the inverse kinematic model has usually a closed-form expression, which means that this control is simpler than in the serial case. However, additional drawbacks appear for parallel robots. The first one is that joint control does not take at all into account the kinematic closures. Hence, such a control may yield internal forces that may damage the robot, and to the least, energy is wasted. A

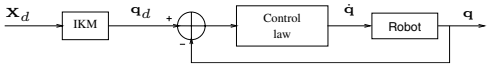

Fig. 1. Joint control with Cartesian reference trajectory

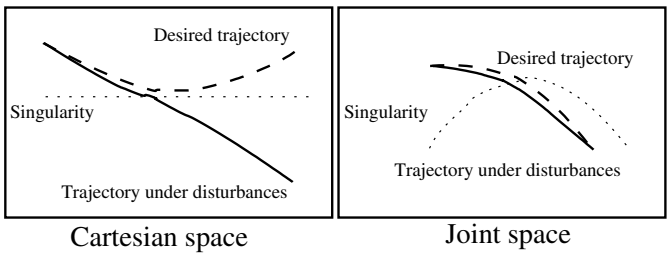

Fig. 2. Effect of disturbances near a singularity in joint control for parallel robots

second drawback is due to the duality of parallel mechanisms with respect to serial robots: while a serial robot end-effector pose is uniquely defined by its joint values and the forward kinematic model, the parallel robot joints are uniquely defined by its end-effector pose and the inverse kinematic model. This means that there might exist several admissible parallel robot configurations with the same joint values but different end-effector poses [1]. These configurations are located in different workspace regions that are separated by parallel singularities, *i.e.* robot configurations where the end-effector can move even though the joints are not moving. Consequently, if the robot passes through such a configuration, the joint trajectory will not be modified while the end-effector trajectory may be strongly affected by a small perturbation, thus having the robot switch from one region to another one. Finally, convergence is ensured in the joint space but is not in the Cartesian space (Figure 2). To overcome these drawbacks, one may perform task planning in the Cartesian space to find a path from the current pose to the desired one passing far away enough from the singularities. Nevertheless, it is not satisfying either for the mind. Indeed, what is "far away enough" with respect to calibration errors in the inverse kinematic model and to disturbances occurring during control ?

Alternately, and much preferably in serial robotics, control can be performed in the Cartesian space (Figure 3). Doing so, one frees oneself from much of the numerical errors coming from the numerical inversion of the forward kinematic model.

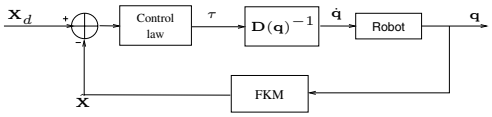

Fig. 3. Cartesian control for serial robots

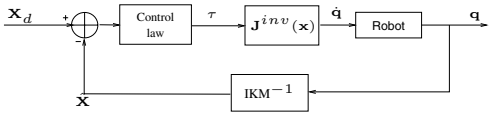

Fig. 4. Cartesian control for parallel robots

In fact, such a control follows exactly the same algorithm as this numerical inversion, the difference being that the update step on the joint values estimation is replaced by joint motion. The consequence of this is that such a control requires the estimation of the end-effector pose from the joint values and the forward kinematic model. It remains thus very sensitive to modeling errors. In the case of parallel robots (Figure 4), this drawback is here again amplified. Indeed, solving for the forward kinematic problem is an ill-posed problem since it requires either non-linear optimization or high-order polynomial solving [2], [3] and may have several solutions (up to 40 [4] real solutions for the reference Gough-Stewart platform [5], [6]). In the case of parallel robot identification, Daney proved [7] that inverting the inverse kinematic model of a parallel robot in a non-linear iterative optimization (which is what control essentially is !) may yield numerical instabilities. Transposed to the control case, this result means that convergence can not be guaranteed. However, one huge advantage of such a control scheme for parallel robots is that the joint velocities it generates are obtained as the output of the differential inverse kinematic model, which filters out any inadmissible joint motion with respect to the kinematic constraints.

To ease this difficult problem of controlling parallel robots, a lot of research is going on either in innovative structural synthesis [8] (where people try to design parallel mechanisms with analytical or semi-analytical forward kinematic models) or in intelligent solutions to the forward kinematic problem (either numerical solutions [9] or novel solutions such as the use of redundant metrology [10] which is mechanism dependent). Notice that, as far as we know, parallel robot and machine manufacturers seem to be desperately seeking the solution in the old serial mechanism recipe consisting in tightening manufacturing and assembly tolerances.

In our opinion, the generic solution for controlling any parallel robot has to be found in *really* taking into account the specific kinematic properties of parallel robots for control rather than in applying patches to the classical serial robot control. Consequently, our objective is here to show that there exists a generic type of control well fitted to parallel robots. It takes fully advantage of the main property of almost all parallel robots: the state (in the full automatic control meaning of this term) of a parallel robot is its end-effector pose with respect to its base, not its joint values.

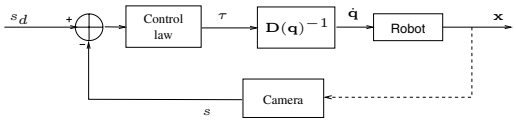

Fig. 5. Visual servoing for serial robots

To make control robust with respect to modeling errors, serial robotics invented visual servoing [11], [12]. Visual servoing is essentially a control scheme where control is performed in a sensor space (Figure 5), which should be the image of the Cartesian space by a diffeomorphism (for instance, the image of a rigid set of points attached to the controlled Cartesian frame). Basically, visual servoing allows to replace the forward kinematic model in the feedback by a camera *measuring*, explicitly or not, the end-effector pose. If control is performed directly in the image (image-based visual servoing), one gets rid of almost all modeling errors since the latter only appear in the robot differential kinematic model and the so-called interaction matrix [13], which play the role of Jacobian matrices (without being theoretically a Jacobian matrices since the Cartesian space is not a vector space). Since modeling errors do not appear any more in the regulated error (end-effector pose is not estimated *via* a model), only the transient phase might be affected by them but not the convergence.

Consequently, the contributions of this paper are to show how visual servoing methods, well known in the serial case, extend to the parallel case and to show that visual servoing is certainly the best choice for kinematic control of parallel robots. Indeed, it is perfectly fitted to the Cartesian control of any parallel robot (Figure 6) since it allows for higher robustness, as stated above, simplifies the control and replaces joint sensors. Moreover, we will show formally that among the various visual servoing techniques, 3D pose visual servoing [14] is, for parallel robots, the canonical one, which is effectively the choice made in [15], [16], [17] for parallel robots with a reduced number of DOF. To do so, we will recall in Section II some basic concepts related to Cartesian control, pointing out the differences between serial and parallel robots. Then, Section III will show why 3D pose visual servoing is unavoidable and replace properly this control scheme in the framework of non-linear control theory. Finally, Section IV will show experimental validation results and Section V will end the paper on a discussion.

II. KINEMATICS

In this section, using the notation in Table I, we mainly want to remind the differences between serial and parallel mechanisms, then to point out the fundamental consequence thereof concerning control.

The end-effector pose of a serial mechanism can be expressed in closed-form from the joint values using the so-called forward kinematic model:

$$X = f(\mathbf{q}) \tag{1}$$

- boldface characters and capital boldface characters denote respectively vectors and matrices.
- $\mathcal{F}_b$, $\mathcal{F}_e$, $\mathcal{F}_c$, $\mathcal{F}_p$ denote respectively the base, end-effector, camera and pattern reference frames.
- ${}^i\mathbf{T}_j = \begin{pmatrix} {}^i\mathbf{R}_j & {}^i\mathbf{t}_j \\ \mathbf{0} & \mathbf{1} \end{pmatrix}$ is the homogeneous matrix associated to the rigid transformation from $\mathcal{F}_i$ to $\mathcal{F}_j$.
- ${}^i\mathbf{v}$ is vector $\mathbf{v}$ expressed in $\mathcal{F}_i$.
- $\mathbf{q}$ is the joint vector.
- $X \in SE(3)$ is the end-effector pose independently from its representation.
- τ is the Cartesian velocity, ${}^i\tau_j$ is the Cartesian velocity of the origin of $\mathcal{F}_j$ expressed in $\mathbf{F}_i$.
- $\mathbf{x}$ is the state vector of the state space representation.
- $\mathbf{K}$ is a negative scalar constant matrix.
- $\mathbf{u_c}$ is the vector of inputs (or forcing function) of the state space representation.
- $[\mathbf{a}]_\times$ is the skew symmetric matrix associated with vector $\mathbf{a}$.
- $\mathbf{M}^+$ is the pseudo-inverse of $\mathbf{M}$.

TABLE I

NOTATION USED THROUGHOUT THE PAPER.

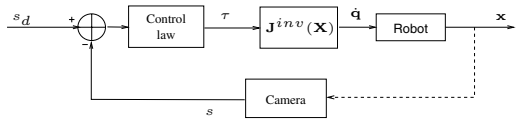

Fig. 6. Visual servoing for parallel robots

The expression of this relation may vary according to the representation which is chosen for the end-effector pose X.

From this expression, one can obtain the differential forward kinematic model, expressing the end-effector Cartesian velocity from the joint velocities, through formal time derivation:

$$\tau = \mathbf{D}(\mathbf{q})\dot{\mathbf{q}} \tag{2}$$

Thus, for serial mechanisms, the models depend only on the joint values. Consequently, *the state of a serial robot is the joint value vector*.

On the other hand, most parallel mechanisms own an inverse kinematic model, giving a closed-form expression of the relation from the end-effector pose to the joint values:

$$\mathbf{q} = g(X) \tag{3}$$

Time differentiating (3), one can similarly obtain the differential inverse kinematic model, expressing the joint velocities from the end-effector Cartesian velocity:

$$\dot{\mathbf{q}} = \mathbf{D}^{inv}(X)\tau \tag{4}$$

Thus, for parallel mechanisms, the models depend on the end-effector pose. Consequently, *the state of a parallel robot is any representation of the end-effector pose X*.

Notice, once again, that the differential inverse kinematic model, which is the heart of Cartesian control, has a closed-form expression for parallel mechanisms while it has to be numerically evaluated for serial mechanisms. Consequently, it should be more natural to perform Cartesian control for parallel mechanisms than for serial ones, *provided that one has a correct estimate or measure of the end-effector pose*.

III. VISUAL SERVOING

A. A short reminder on visual servoing

Visual servoing is based on the so-called interaction matrix $\mathbf{L_s}$ which relates the instantaneous relative Cartesian motion τ between the camera and the scene, to the time derivative of the vector $\mathbf{s}$ of all the visual primitives that are used through [18]:

$$\dot{\mathbf{s}} = \mathbf{L_s}\tau \tag{5}$$

where τ can be expressed at any convenient point and in any convenient reference frame.

Then, one achieves exponential decay of an error $\mathbf{e}(\mathbf{s}, \mathbf{s}^*)$ between the current primitive vector $\mathbf{s}$ and the desired one $\mathbf{s}^*$ using a proportional linearizing and decoupling control scheme of the form:

$$\tau = -\lambda \hat{\mathbf{L}}_\mathbf{s}^+ \mathbf{e}(\mathbf{s}, \mathbf{s}^*) \tag{6}$$

where τ is used as a pseudo-control variable and is usually converted through the differential inverse kinematic model of the robot into joint velocity inputs.

According to the nature of visual primitive, there exist many visual servoing techniques ranging from position-based visual servoing (PBVS) [19] to image-based visual servoing (IBVS) [12], most of them based on point features but one also find other visual primitives such as lines [20] or image moments [21]. To simplify the discussion in [22], PBVS schemes yield straight trajectories in the non-linear Cartesian space but can not guarantee the visibility constraint because the trajectories in the linear image space are curved, while IBVS has the opposite behavior and namely yield smaller rotational motion. Additionally IBVS is usually considered as not requiring any end-effector pose estimation since only depth distribution of the observed points is needed. Nevertheless, IBVS is not robust to errors on this distribution [23].

To try to take advantage of both schemes, hybrid-based visual servoing schemes (HBVS) were proposed such as [24], which only requires to extract the relative orientation and relative depth in the Cartesian space of the servoed object from the homography between the current and desired image. In some way, one can consider that they allow for a relative end-effector pose, up to a scale factor, without knowing the object 3D structure.

Remind also that PBVS exists under two main forms: 3D points PBVS [19] where the reconstructed 3D coordinates of points on the observed pattern are used as visual primitives and 3D pose PBVS [14], [25] (or 3D pose visual servoing) where a minimal representation of the camera-to-pattern pose is used. Note that 3D pose PBVS is the form which requires the highest amount of 3D reconstruction from images. Therefore, every visual servoing scheme can be applied once the requirements for 3D pose PBVS are met: every visual servoing interaction matrix can be fully computed from the 3D pose.

Finally, there are two visual servoing configurations. In the eye-in-hand configuration, the camera (defined by its reference frame $\mathcal{F}_c$) is rigidly fixed onto the end-effector and is observing a pattern (defined in the reference frame $\mathcal{F}_p$) attached to the world frame. On the opposite, the eye-to-hand

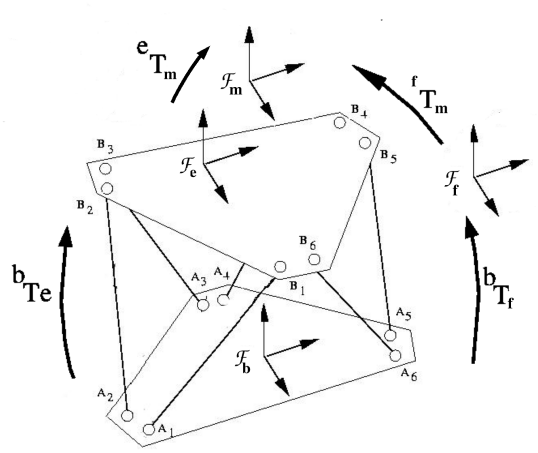

Fig. 7. Generic configuration for parallel robot visual servoing

configuration is such that the camera is attached to the world frame and the pattern is mounted onto the end-effector. In both case, a *mobile* reference frame $\mathcal{F}_m$ is attached to the end-effector frame $\mathcal{F}_e$ and a *fixed* one ($\mathcal{F}_f$) is attached with respect to the base frame $\mathcal{F}_b$.

B. The state space notation for control

Consider now (Figure 7) a parallel mechanism equipped with a camera and a pattern in any configuration (eye-in-hand or eye-to-hand). Under this configuration, it is trivial to see that the rigid transformation from the fixed frame to the mobile frame (which can be estimated easily [26] and with high accuracy [27] by vision) is similar to the base to end-effector transformation, up to two constant changes of frames. Hence, the camera-to-pattern pose is an adequate representation of the end-effector pose X. Consequently, the end-effector pose appears both in 3D pose visual servoing and in the robot kinematics. As stated above, any visual servoing scheme can then be applied. Nevertheless, let us show that *3D pose PBVS is the easiest and most straightforward choice for control*.

In this control scheme and the generic configuration in Figure 7, the visual primitive $\mathbf{s}$ should be chosen as [14]:

$$\mathbf{s} = \begin{pmatrix} \mathbf{t} \\ \underline{\mathbf{u}}\theta \end{pmatrix} \tag{7}$$

where $\mathbf{t} = {}^m\mathbf{t}_{m^*}$ is the position error or translation between the current ($\mathcal{F}_m$) and desired ($\mathcal{F}_m^*$) mobile frame, while $\underline{\mathbf{u}}\theta$ is the orientation error, decomposed as the axis $\underline{\mathbf{u}}$ and angle θ of the rotation ${}^m\mathbf{R}_{m^*}$ between these two frames. Notice that $\mathbf{s}$ is not a vector, contrary to most statements in the literature.

Associated to this error, the interaction matrix in (5) becomes square [24], [14]:

$$\mathbf{L_s} = \begin{pmatrix} -\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{L}_w \end{pmatrix} \tag{8}$$

with

$$\mathbf{L}_w = \mathbf{I}_3 - \frac{\theta}{2}[\underline{\mathbf{u}}]_\times + \left(1 - \frac{sinc(\theta)}{sinc^2(\frac{\theta}{2})}\right)[\underline{\mathbf{u}}]_\times^2 \tag{9}$$

and can be analytically inverted [24].

Notice that, the vision-based task $\mathbf{e}$, needs be servoed to 0, which is coherent with the state feedback control. Hence, noting $\mathbf{x} = \mathbf{e}$ the state of the parallel robot, $\mathbf{y} = \mathbf{e}$ the output of the control law and $\mathbf{u_c} = \tau_m$ the pseudo-control vector, we can reformulate the 3D pose visual servoing problem as a proper non-linear state feedback control scheme:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u_c} \tag{10}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u_c} \tag{11}$$

where $\mathbf{A} = \mathbf{0}_3$, $\mathbf{B} = \mathbf{L_s}$, $\mathbf{C} = \mathbf{I}_3$ and $\mathbf{D} = \mathbf{0}_3$.

Notice that this state space representation is non-linear since $\mathbf{B} = \mathbf{B}(\mathbf{x})$.

C. Control Law

According to classical non-linear control, we can choose either a linear state feedback to control the system (tangent linearization):

$$\mathbf{u_c} = \mathbf{B}^{-1}(0)\mathbf{K}\mathbf{x} \tag{12}$$

or a non-linear state feedback (exact linearization):

$$\mathbf{u_c} = \hat{\mathbf{B}}^{-1}(\mathbf{x})\mathbf{K}\mathbf{x} \tag{13}$$

Notice that in the tangent case, the interaction matrix $\mathbf{B}(0) = \mathbf{L_s}(\mathbf{s} = \mathbf{0})$ simplifies into $\mathbf{B}(0) = \left(\begin{smallmatrix} -\mathbf{I}_3 & \mathbf{0}_3 \\ \mathbf{0}_3 & \mathbf{I}_3 \end{smallmatrix}\right)$.

Projecting the control law into the end-effector frame yields the generic 3D pose PBVS control law, valid for any robot (serial or parallel):

$${}^e\tau_e = \begin{pmatrix} {}^e\mathbf{R}_m & [{}^e\mathbf{t}_m]_\times {}^e\mathbf{R}_m \\ 0 & {}^e\mathbf{R}_m \end{pmatrix} {}^m\tau_m \tag{14}$$

where ${}^m\tau_m$ is given either by (12) or (13) expressed in the mobile frame.

Specifying this results for a generic parallel robot yields the actual joint velocity control signal

$$\dot{\mathbf{q}} = {}^e\mathbf{D}_e^{inv}(X)\,{}^e\tau_e \tag{15}$$

where X is represented by ${}^b\mathbf{T}_e$, which can be computed along two possibilities depending on the configuration.

a) Eye-to-hand system: Here, $\mathcal{F}_m = \mathcal{F}_p$ and $\mathcal{F}_f = \mathcal{F}_c$ and hence

$${}^b\mathbf{T}_e = {}^b\mathbf{T}_c\,{}^c\mathbf{T}_p\,{}^p\mathbf{T}_e \tag{16}$$

where ${}^b\mathbf{T}_c$ and ${}^p\mathbf{T}_e$ are known by calibration and ${}^c\mathbf{T}_p$ is measured by vision.

b) Eye-in-hand system: Now, $\mathcal{F}_f = \mathcal{F}_p$ and $\mathcal{F}_m = \mathcal{F}_c$ and hence

$${}^b\mathbf{T}_e = {}^b\mathbf{T}_p\,{}^c\mathbf{T}_p^{-1}\,{}^c\mathbf{T}_e \tag{17}$$

where ${}^b\mathbf{T}_p$ and ${}^c\mathbf{T}_e$ are known by calibration and ${}^c\mathbf{T}_p{}^{-1}$ is measured by vision.

Consequently, the proposed control is extremely simple and has a fundamental property: joint values do not appear in the control equations (12)-(17). Thus, the mechanical design of parallel robot can be simplified.

Fig. 8. A Gough-Stewart platform observed by a camera.

IV. Experimental validation

In the previous derivation, we did not make any assumption on which parallel robot was to be controlled, *i.e.* on the expression of the inverse kinematic model. In this section, the approach is experimentally validated on a Gough-Stewart platform in eye-to-hand configuration (Figure 8).

A. Inverse kinematic model

It has 6 legs of varying length $\mathbf{q}_i, i \in 1..6$, attached to the base by spherical joints located in points $\mathbf{A}_i$ and to the moving platform (end-effector) by spherical joints located in points $\mathbf{B}_i$. The inverse kinematic model of such an hexapod expressed in the end-effector frame is given by [1]

$$\forall i \in 1..6, \quad \mathbf{q}_i^2 = \overrightarrow{{}^e\mathbf{A}_i\,{}^e\mathbf{B}_i}^T\,\overrightarrow{{}^e\mathbf{A}_i\,{}^e\mathbf{B}_i} \tag{18}$$

expressing that q_i is the length of vector $\overrightarrow{\mathbf{A}_i\mathbf{B}_i}$. Introducing ${}^e\underline{\mathbf{u}}_i$ the unit vector pointing from $\mathbf{A}_i$ to $\mathbf{B}_i$, we can rewrite (18) as

$$q_i\,{}^e\underline{\mathbf{u}}_i = {}^e\mathbf{B}_i - {}^e\mathbf{R}_b\,{}^b\mathbf{A}_i - {}^e\mathbf{t}_b \tag{19}$$

from which one obtains the differential inverse kinematic model

$$\dot{\mathbf{q}} = {}^e\mathbf{J}_e^{inv}\,{}^e\tau_e \tag{20}$$

with

$${}^e\mathbf{J}_e^{inv} = \begin{pmatrix} {}^e\underline{\mathbf{u}}_1^T & {}^e\mathbf{B}_1 \times {}^e\underline{\mathbf{u}}_1^T \\ \vdots & \vdots \\ {}^e\underline{\mathbf{u}}_6^T & {}^e\mathbf{B}_6 \times {}^e\underline{\mathbf{u}}_6^T \end{pmatrix} \tag{21}$$

where the ${}^b\mathbf{A}_i$ and the ${}^e\mathbf{B}_i$ are constant calibration parameters.

B. Experimental results

The proposed approach was implemented using an open source visual servoing library [28] on a tailored commercial DeltaLab platform. It has to be noticed that this library simplified much of the development since everything but the integration of the platform (15)-(21) was already implemented. Hence, off-the-shelf software comes in support to the assessment claimed in the title.

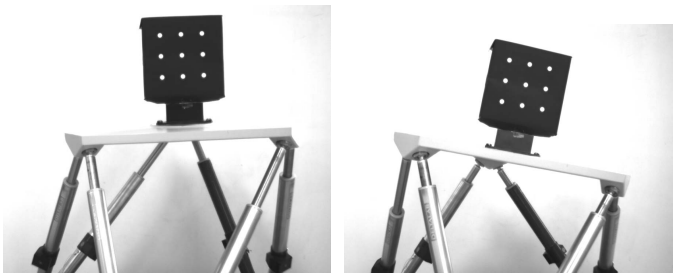

Fig. 9. Initial (left) and desired (right) position of the end-effector, seen from the camera.

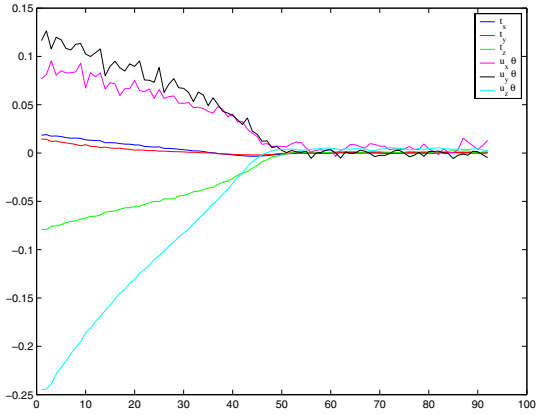

Fig. 10. Evolution of the Cartesian error.

In the reported experiment, the robot is asked to reach the desired position from the initial configuration that are displayed in Figure 9. Thus, the robot covers a large amount of its workspace.

Figure 10 shows that the errors converges to 0 as expected, from an initial error to a final one displayed in Table II. Notice that the error curves are not exponentials since an adaptive gain strategy was used to compensate for Coulomb friction near convergence without generating high image velocities at the beginning. Figure 11 shows that convergence is also normally reached in the joint space.

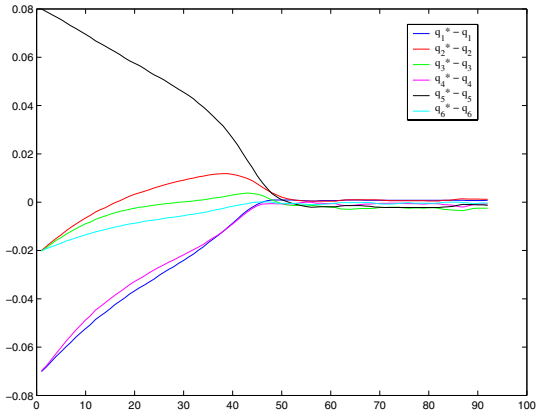

Fig. 11. Evolution of the joint errors.

	Position error (cm)	Orientation error (deg)
Initial Errors	8.23	16.157
Final Errors	0.141	0.342

TABLE II

INITIAL AND FINAL ERRORS

V. DISCUSSION

Using 3D pose visual servoing was demonstrated in this paper a straightforward approach for controlling the end-effector pose of a parallel robot. It is straightforward since it is fully coherent with the need for estimating the end-effector pose to feed the parallel robot models, but also since it can be easily implemented with off-the-shelf hardware and software. Indeed, one can now easily find camera to pattern pose estimation libraries (for instance, OpenCV for a free one) that deliver a rigid transformation which is similar to the robot end-effector pose with respect to its base frame up to two rigid transformations. Moreover, there even exist libraries for visual servoing that implement everything from frame grabbing and visual tracking to control, where one only has to plug in the robot inverse kinematic model and joint control.

Now, as soon as the compulsory camera to pattern pose is estimated, one has more than needed to perform one's preferred visual servoing control scheme, such as image-based or hybrid-based visual servoing, to impose one's desired behavior to the end-effector.

One should notice also the proposed method does not require any non-linear optimization problem to solve and even better does not need either any numerical matrix inversion since the differential inverse kinematic model and the interaction matrix inverse have analytical expressions.

Moreover, nowhere in the proposed approach, joint values were needed. This means that a camera is the only sensor needed for controlling a parallel robot and hence that one may simplify in the future the mechatronics design, manufacturing and assembly of parallel robots by suppressing joint encoders. Nevertheless, this is, in the present state of technology, limited to robots that have compatible velocity and accuracy with vision (about 100 Hz control loop frequency and accuracy within 1/100000 of the field of view): large scale telescopes, car assembly robots, for instance. However, if one can afford it, laser tracker [29] also delivers a sensor-to-target pose (equivalent to the camera-to-pattern one) with higher performances which fits into the proposed framework.

REFERENCES

[1] J.P. Merlet. *Parallel robots*. Kluwer Academic Publishers, 2000.

[2] J-P. Merlet. An algorithm for the forward kinematics of general 6 d.o.f. parallel manipulators. Technical Report 1331, INRIA, November 1990.

[3] M. Husty. An algorithm for solving the direct kinematics of general Gough-Stewart platforms. *Mech. Mach. Theory*, 31(4):365–380, 1996.

[4] P. Dietmaier. The Stewart-Gough platform of general geometry can have 40 real postures. In J. Lenarčič and M. L. Husty, editors, *Advances in Robot Kinematics: Analysis and Control*, pages 1–10. Kluwer, 1998.

[5] V.E. Gough and S.G. Whitehall. Universal tyre test machine. In *Proceedings of the FISITA* 9^{th} *International Technical Congress*, pages 117–137, 1962.

[6] D. Stewart. A platform with six degrees of freedom. In *Proc. IMechE (London)*, volume 180, pages 371–386, 1965.

[7] D. Daney. Self calibration of Gough platform using leg mobility constraints. In *Proc. of the* 10^{th} *World Congress on the theory of machine and mechanisms*, pages 104–109, Oulu, Finland, 1999.

[8] G. Gogu. Fully-isotropic T3R1-type parallel manipulator. In J. Lenarčič and C. Galletti, editors, *On Advances in Robot Kinematics*, pages 265–272. Kluwer Academic Publishers, 2004.

[9] X. Zhao and S. Peng. Direct displacement analysis of parallel manipulators. *Journal of Robotics Systems*, 17(6):341–345, 2000.

[10] L. Baron and J. Angeles. The direct kinematics of parallel manipulators under joint-sensor redundancy. *IEEE Transactions on Robotics and Automation*, 16(1):1–8, février 2000.

[11] L. E. Weiss, A. C. Sanderson, and C. P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE Journal of Robotics and Automation*, RA-3(5):404–417, October 1987.

[12] B. Espiau, F. Chaumette, and P. Rives. A New Approach To Visual Servoing in Robotics. *IEEE Trans. on Robotics and Automation*, 8(3), June 1992.

[13] C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: the Task Function Approach*. Clarendron Press, Oxford University Press, Oxford, UK, 1990.

[14] B. Thuilot, P. Martinet, L. Cordesses, and J. Gallice. Position based visual servoing: keeping the object in the field of vision. *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA'02)*, pages 1624–1629, May 2002.

[15] M.L. Koreichi, S. Babaci, F. Chaumette, G. Fried, and J. Pontnau. Visual servo control of a parallel manipulator for assembly tasks. In *6th Int. Symposium on Intelligent Robotic Systems, SIRS'98*, pages 109–116, Edimburg, Scotland, July 1998.

[16] H. Kino, C.C. Cheah, S. Yabe, S. Kawamua, and S. Arimoto. A motion control scheme in task oriented coordinates and its robustness for parallel wire driven systems. In *Int. Conf. Advanced Robotics (ICAR'99)*, pages 545–550, Tokyo, Japan, Oct. 25-27 1999.

[17] P. Kallio, Q. Zhou, and H. N. Koivo. Three-dimensional position control of a parallel micromanipulator using visual servoing. In Bradley J. Nelson and Editors Jean-Marc Breguet, editors, *Microrobotics and Microassembly II, Proceedings of SPIE*, volume 4194, pages 103–111, Boston, USA, November 2000.

[18] F. Chaumette and E. Marchand. Recent result in visual servoing for robotics applications. *In Proceeding of the 8th ESA Workshop on Advanced Space Technologies for Robotics and Automation 'ASTRA 2004' ESTEC, Noordwijk, the Netherlands*, November 2004.

[19] P. Martinet. Comparison of visual servoing techniques: Experimental results. *Proceedings of the European Control Conference, ECC'99,paper 1059-4, Karlsruhe, Germany*, August 1999.

[20] N. Andreff, B. Espiau, and R. Horaud. Visual servoing from lines. *Int. Journal of Robotics Research*, 21(8):679–700, August 2002.

[21] F. Chaumette. Image moments: a general and useful set of features for visual servoing. *IEEE Trans. on Robotics*, 20(4):713–723, August 2004.

[22] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. In *In D. Kriegman, G. Hager, A.S Morse, editeurs, the Conflunce of Vision and Control*, pages 66–78, 1998.

[23] E. Malis and P. Rives. Robustness of image-based visual servoing with respect to depth distribution errors. In *IEEE International Conference on Robotics and Automation (ICRA'03)*, Taipei, Taiwan, September 2003.

[24] E. Malis, F. Chaumette, and S. Boudet. 2 1/2 d visual servoing. *IEEE Tran. On Robotics and Automation*, 15(2):238–250, April 1999.

[25] W. Wilson, C. Hulls, and G. Bell. Relative end-effector control using cartesian position-based visual servoing. *IEEE Tran. On Robotics and Automation*, 12(5):684–696, October 1996.

[26] D. DeMenthon and L. Davis. Model-based object pose in 25 lines of code. *Lecture Notes in Computer Science*, pages 335–343, 1992.

[27] J.M Lavest, M. Viala, and M. Dhome. Do we really need an accurate calibration pattern to achieve a reliable camera calibration. In *Proceedings of the* 5^{th} *European Conference on Computer Vision*, pages 158–174, Freiburg, Allemagne, June 1998.

[28] E. Marchand, F. Spindler, and F. Chaumette. ViSP: a generic software platform for visual servoing. *IEEE Robotics and Automation Magazine*, 12(4), December 2005.

[29] M. Vincze, J.P. Prenninger, and H. Gander. A laser tracking system to measure position and orientation of robot end-effectors under motion. *International Journal of Robotics Research*, 13(4):305–314, 1994.