Dynamic Control of the Quattro Robot by the Leg Edges

Erol Ozgür, Nicolas Bouton, Nicolas Andreff, Philippe Martinet

Dynamic Control of the Quattro Robot by the Leg Edges

Erol Özgür, Nicolas Bouton, Nicolas Andreff, Philippe Martinet

***Abstract*— This paper discusses variable selection for the efficient dynamic control of the Quattro parallel robot through an inverse dynamic model expressed by means of leg orientations. A selection is made within a group of variables where each can imply the state of the robot. Besides, in this work, steering a parallel robot dynamically using its self-projection onto the image plane (where the edges of the lower-legs are exploited in control) is proposed and validated for the first time. In the light of the realistic control simulation, the formative points of better control of the Quattro robot are figured out.**

I. INTRODUCTION

In an industrial automated plant, robots are the key machines for the typical applications such as pick-and-place, high-speed machining, etc., and nowadays the parallel robots meet the demands of production bands well since they are faster and potentially have better stiffness and accuracy than the serial robots [1].

However, they are structurally very complex to be controlled precisely so as to exploit their full potential. There have been many studies in this area in the last two decades (see [2] for a literature review) which have offered conceptually generic solutions to the problem. The solution of this difficult problem was simply looked for in the following conventional scenario, where it has usually been treated in separate modules, namely *sensing*, *modeling* and *control*, respectively.

In sensing, since the robots are directly equipped with motor encoders that measure the articular positions, many researchers tried to develop models by hovering around this shallow information. As a result, the models were very long, time consuming (difficult to warrant the real-time constraints) and hard to understand. Therefore, inevitably, this urges one to offer simplifications [3] and to turn a blind eye to some of the modeling errors in the mechanism, thus giving simplified and fast [4] but less accurate new models for control. Consequently, this pushes the community to look for sophisticated control algorithms in order to compensate for the loss of accuracy.

It can easily be noticed from the above scenario that each module has poorly evolved having the negative effects of the others on the way to the precise control of a parallel robot. So, what would be a better approach which directly considers the objective? If one looks over the whole problem from a wide perspective, the answer can be seen in the selection of an "*efficient variable set*" [5]. This variable set should let the modules be right in step with one another, namely, make them converge to a *unified single system* that governs the parallel robot right on target. In this paper, the *control* module has been explored further in detail so as to incorporate it smoothly into and complete the desired *unified system*, where the integration of *sensing* and *modeling* parts have already been discussed in our previous works [6], [7], [8]. The expected contributions of this work are: (i) to propose a dynamic control through the leg contour pixels (edgels) of a parallel robot for potentially better performance, (ii) to avoid using an artificial pattern in dynamic tracking, while the parallel robot turns itself into a pattern, (iii) to increase the robustness of dynamic control since the image-space is resistant to errors, and (iv) to give intuitions that might shed light on a complete image-based (2D) dynamic modeling and control of parallel robots.

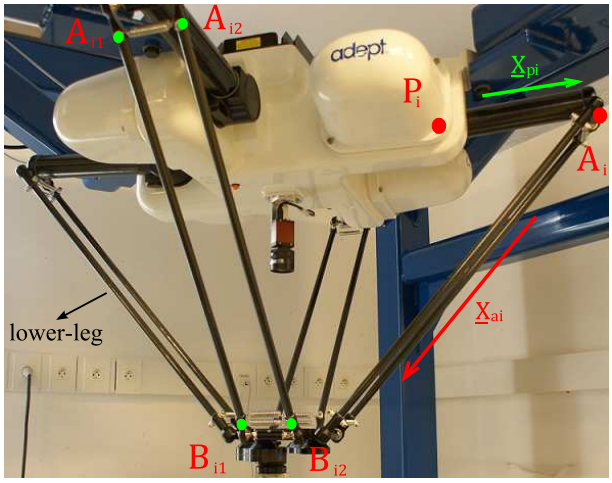

Fig. 1. The Quattro parallel robot with a base-mounted camera.

The remainder of this paper is organized as follows: Section II gives background information on the geometry and the notation of the Quattro robot (Fig. 1) and briefly talks about the leg orientations based inverse dynamic model. In Section III, we discuss both the choice of the sensor-signal and the control-law used to steer the robot efficiently. In Section IV, we give place to comparative results and their annotations. Finally, Section V concludes the paper with some remarks and highlights the future research directions.

II. BACKGROUND

A. The Quattro

The Quattro is composed of four identical kinematic legs, that carry the articulated moving platform. Each of the 4 kinematic legs is actuated from the base by a revolute motor,

This work is supported by the ANR-VIRAGO project.

The authors are with LASMEA-CNRS, Clermont Universités, 24 Avenue des Landais 63177 Aubière Cedex, Clermont-Ferrand, France. N. Andreff is also with FEMTO-ST Institute, Université de Franche-Comté, Besançon, France. firstname.lastname@lasmea.univ-bpclermont.fr

located at $\mathbf{P}_i$, and has two consecutive bodies (an upper-leg and a lower-leg) linked with each other at $\mathbf{A}_i$. (see Fig. 1). The lower-legs are connected to the articulated moving platform at $\mathbf{B}_i$. Notation:

- $i = 1,2,3,4$ denotes the kinematic legs and $j = \{l,r\}$ denotes the left and right edges of the slim and cylindrical shaped lower-leg rods $\{[\mathbf{A}_{i1}\mathbf{B}_{i1}], [\mathbf{A}_{i2}\mathbf{B}_{i2}]\}$.
- Vectors are denoted with boldface characters and, in addition, unit vectors are underlined.
- $\mathbf{V}_e = [\dot{x},\dot{y},\dot{z}]^T$ and ω_z are, in turn, the translational velocity and the angular velocity around the fixed axis $\underline{\mathbf{z}}_e$ of the end-effector $\mathbf{E}$. Thus, the Cartesian pose velocity of the end-effector frame can be represented by:

$$\dot{\chi} = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} & \omega_z \end{bmatrix}^T$$

B. Dynamics

In previous works [6], [7], an algebraic expression for the inverse dynamic of the Quattro robot was derived through the following procedure:

- *Khalil* [9], intuition of splitting the parallel robot into subsystems: kinematic legs and a moving platform.
- *Kane* [10] and *Newton-Euler*, to calculate the generalized forces on the decomposed parts of the parallel robot.
- *d'Alembert/principle of virtual work*, to combine all the works and write the final equations of motion.

yielding the inverse dynamic model (IDM) by means of the lower-leg orientations $\underline{\mathbf{x}}_{ai}$ as below:

$$\Gamma = IDM(\ddot{\underline{\mathbf{x}}}_{ai}, \dot{\underline{\mathbf{x}}}_{ai}, \underline{\mathbf{x}}_{ai}) = \mathbf{A}(\underline{\mathbf{x}}_{ai})\ddot{\underline{\mathbf{x}}}_{ai} + \mathbf{h}(\underline{\mathbf{x}}_{ai}, \dot{\underline{\mathbf{x}}}_{ai}) \quad (1)$$

where $\Gamma \in \Re^{4\times 1}$ is the motor torque vector. The reason for writing the inverse dynamic model as a function of such variables is not only its simple closed-form expression but is also related to sensor-based control issues.

III. SENSOR-BASED DYNAMIC CONTROL

A. Background

The standard method for dynamic control is the so-called computed-torque control (CTC) [11] which linearizes and decouples the control. Its well-known form, adapted to serial mechanisms, is derived from the Lagrange formulation of the dynamic model:

$$\Gamma = \mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q},\dot{\mathbf{q}}) \triangleq IDM(\mathbf{q},\dot{\mathbf{q}},\ddot{\mathbf{q}}) \quad (2)$$

Under the assumption that $\hat{\mathbf{A}}$ and $\hat{\mathbf{h}}$ are correct estimates of $\mathbf{A}(\mathbf{q})$ and $\mathbf{h}(\mathbf{q},\dot{\mathbf{q}})$, one can build a control torque of the form:

$$\Gamma = \hat{\mathbf{A}}\,\mathbf{u}_q + \hat{\mathbf{h}} \quad (3)$$

where $\mathbf{u}_q$ is an auxiliary control vector, equivalent to an acceleration in the joint-space. Indeed, inserting such a control in the direct dynamic model yields a closed-loop equation of the form:

$$\ddot{\mathbf{q}} = \underbrace{\mathbf{A}(\mathbf{q})^{-1}\hat{\mathbf{A}}}_{\approx \mathbf{I}}\,\mathbf{u}_q + \mathbf{A}(\mathbf{q})^{-1}(\underbrace{\hat{\mathbf{h}} - \mathbf{h}(\mathbf{q},\dot{\mathbf{q}})}_{\approx \mathbf{0}}) \quad (4)$$

which is a second-order linear-system and can therefore be controlled by any linear controller, the latter being often chosen as a proportional-derivative plus feedforward (PD+FF) controller in the joint-space.

Now, for parallel kinematic manipulators, this control is hardly used because it is computationally heavy (partly because one has to solve for the forward kinematic problem at each iteration). It is therefore often discarded to the profit of simplified controllers (the worst being a PID controller in the joint-space) which can be shown [12] to have poor properties as far as linearization and decoupling are concerned.

The essential drawback of this joint-space computed-torque control is its reliance on a dynamic model expressed in joint-space. An alternative to this is to set up a Cartesian-space computed-torque control according to many recommendations for expressing the dynamic model in the Cartesian-space [13], [14]:

$$\Gamma = \widehat{\mathbf{A}(\chi)}\,\mathbf{u}_\chi + \widehat{\mathbf{h}(\chi,\dot{\chi})} \quad (5)$$

where $\mathbf{u}_\chi$ is now equivalent to a Cartesian acceleration and $(\chi,\dot{\chi})$ are the Cartesian pose and velocity of the end-effector to be estimated. This is often considered as a second-order linear-model and controlled with a PD+FF [11], [12], [14], although there exits a formal non-linear control based on the Lie group structure of $SE(3)$ [15]. This control is a state feedback in the case where the inverse kinematic problem has a single solution.

However, despite the fact that the dynamic model is lighter, the estimation of χ and $\dot{\chi}$ is not easy and, consequently, one would like to set up a sensor-based computed-torque control:

$$\Gamma = \widehat{\mathbf{A}(\mathbf{s})}\,\mathbf{u}_s + \widehat{\mathbf{h}(\mathbf{s},\dot{\mathbf{s}})} \quad (6)$$

where $\mathbf{u}_s$ is now equivalent to the second-order time-derivative of the sensor signal $\mathbf{s}$. This control encompasses the former two since a joint-space computed-torque control can be seen as a sensor-based control where the sensor signal is given by the joint encoders and a Cartesian-space control as a sensor-based control where the sensor-signal is the end-effector pose. In fact, the theoretical condition for the validity of a sensor-based control is that there exists a diffeomorphism (i.e. a differentiable bijective mapping) between the sensor-space and the state-space of the system [16]. That is where it hurts in parallel kinematics, especially when one considers only the actuator positions for sensing: the mapping is neither bijective (several solutions to the forward kinematic problem) nor differentiable (singularities of any type). Then, which sensor-signal shall be used?

B. Choosing a sensor signal

The choice of the sensor is influenced by the state of the technology, but more relevantly, since the latter is ever improving, by the control algorithm and, in turn, the control algorithm depends on the model it is built upon. In correlation with the proposed methodology, this comes down to examining the variables in models and to determining how

to get them efficiently. There are three kinds of variables involved in the models:

- Variables related to the actuators: $\{\underline{\mathbf{x}}_{pi}, \dot{\underline{\mathbf{x}}}_{pi}\}$
- Variables related to the end-effector: $\{\underline{\mathbf{x}}_{e}, \dot{\underline{\mathbf{x}}}_{e}\}$
- Redundant variables: $\{\underline{\mathbf{x}}_{ai}, \dot{\underline{\mathbf{x}}}_{ai}\}$

Which kind, if any, will best conform to an efficient control? Of course, if one can sense all of these variables, the problem is completely solved, except for itching calibration and data coherence issues. For the same reason, a combination of two of the three kinds is temporarily left out of the discussion here (although there might lie the practical optimum).

The variables related to the actuators definitely have to be discarded, since they face up the forward kinematic problem, which is not only a non-linear but also a square problem. The variables related to the end-effector are not necessarily the answer due to either their technological cost (laser) or algorithmic cost (pose estimation in computer vision). The latter case is nevertheless better than the forward kinematic problem since the non-linear problem of pose estimation is not square but over-constrained, which makes it numerically more robust, and since it relies on optics rather than on mechanical parts.

Focusing on the use of redundant variables only and having inspirations of metrological redundancy [17], we have found out that they provide us with a linear formulation of the whole problem. Indeed, once all the variables are known, the proposed models make only use of linear algebra. The variables related to the actuators and the end-effector can linearly be expressed from the known variables related to the lower-legs and the known constant parameters [6].

Consequently, as soon as one can sense the redundant variables, one can derive a control using only linear algebra. And so, the only remaining question is how to measure those redundant variables. Our answer is, unsurprisingly, to observe by *vision* the associated mechanical elements in the kinematic legs, preferably as revolute cylinders, as we did it in kinematics [8].

C. Sensor-based computed-torque control law

Since we discuss the control law in different variable spaces than the linearized-dynamics in (1) where $\ddot{\underline{\mathbf{x}}}_{ai} = \mathbf{u}$, it will be appropriate to define a pseudo-system $\ddot{\mathbf{s}} = \boldsymbol{\omega}$. Then, the error function, f_e, which is expressed in terms of current $\mathbf{s}$ and desired $\mathbf{s}^*$ sensor signals, can simply be denoted as follows:

$$\mathbf{e} = f_e(\mathbf{s}^*, \mathbf{s}) = \mathbf{s}^* - \mathbf{s} \tag{7}$$

and assuming a second-order diffeomorphism between $\underline{\mathbf{x}}_{ai}$ and $\mathbf{s}$ is:

$$\dot{\underline{\mathbf{x}}}_{ai} = \mathbf{L}_s\, \dot{\mathbf{s}} \quad \Rightarrow \quad \ddot{\underline{\mathbf{x}}}_{ai} = \dot{\mathbf{L}}_s \dot{\mathbf{s}} + \mathbf{L}_s\, \ddot{\mathbf{s}} \tag{8}$$

where $\mathbf{L}_s$ is the differential kinematic model between the sensor signal and a lower-leg unit orientation vector. One can write the control $\mathbf{u}$ using (8) and pseudo-control $\boldsymbol{\omega}$ as below:

$$\mathbf{u} = f_u(\mathbf{L}_s,\ \dot{\mathbf{L}}_s, \dot{\mathbf{s}}, \boldsymbol{\omega}) = \dot{\mathbf{L}}_s \dot{\mathbf{s}} + \mathbf{L}_s\, \boldsymbol{\omega} \tag{9}$$

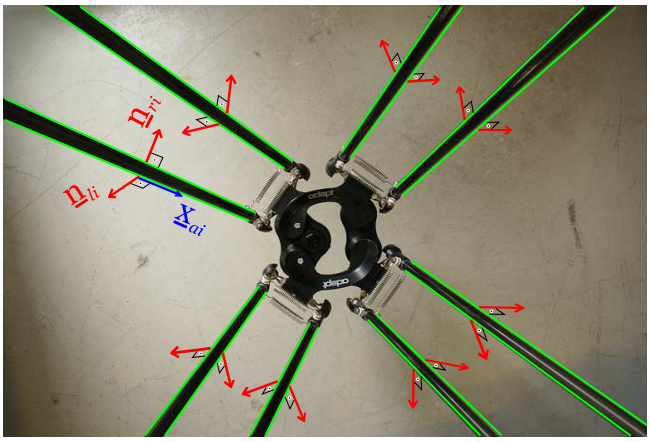

Fig. 2. An image of lower-legs with their edgels (green), orientation vector (blue) and projection-line vectors (red) from the base-mounted camera of the Quattro robot in Fig. 1.

where f_u is a function of a differential kinematic model $\mathbf{L}_s$ and its derivative (computed numerically), of the derivative of the sensor signal $\mathbf{s}$ and the pseudo-control $\boldsymbol{\omega}$, respectively. Afterwards, one only needs to measure / compute the $\{\mathbf{s}, \dot{\mathbf{s}}, \mathbf{L}_s, \dot{\mathbf{L}}_s\}$. One should prove as well that linearized-dynamics ($\ddot{\underline{\mathbf{x}}}_{ai} = \mathbf{u}$) is equivalent to pseudo-system ($\ddot{\mathbf{s}} = \boldsymbol{\omega}$) in order to bring the error down to zero. Subsequently, one can rewrite the linearized-dynamics using the right sides of the last two expressions in (8) and (9) as follows:

$$\dot{\mathbf{L}}_s \dot{\mathbf{s}} + \mathbf{L}_s\, \ddot{\mathbf{s}} = \dot{\mathbf{L}}_s \dot{\mathbf{s}} + \mathbf{L}_s\, \boldsymbol{\omega} \tag{10}$$

and this will boil down to the state of pseudo-system ($\ddot{\mathbf{s}} = \boldsymbol{\omega}$), on condition that *the good approximations of the models* exist and $(\ddot{\mathbf{s}} - \boldsymbol{\omega})$ does not lie in the null-space of $\mathbf{L}_s$. Finally, the pseudo-control $\boldsymbol{\omega}$ can be set up as below:

$$\boldsymbol{\omega} = K_P\ \mathbf{e} + K_D\ \dot{\mathbf{e}} + \ddot{\mathbf{s}}^* \tag{11}$$

where K_P and K_D are the proportional and derivative positive controller gains, respectively. This yields a second-order convergence in $\mathbf{s}$. In the following three subsections, we will be deriving the control laws with different sensor-signals for comparative purposes of dynamic control.

1) Image-space computed-torque control (IS-CTC): On the image plane, the left and right fitted line equations in pixel-units ${}^{p}\underline{\mathbf{n}}_{ji} \in \Re^{3\times 1}$ of the edgels of a projected lower-leg rod (see Fig. 2) of the Quattro parallel robot are exploited as sensor signals in control (see Fig. 3). The error vector for each leg, $\mathbf{e}_i \in \Re^{6\times 1}$, is written over them as follows:

$$\mathbf{e}_i = \begin{bmatrix} \mathbf{e}_{li} \\ \mathbf{e}_{ri} \end{bmatrix} = \begin{bmatrix} {}^{p}\underline{\mathbf{n}}^*_{li} - {}^{p}\underline{\mathbf{n}}_{li} \\ {}^{p}\underline{\mathbf{n}}^*_{ri} - {}^{p}\underline{\mathbf{n}}_{ri} \end{bmatrix} \tag{12}$$

where $\{{}^{p}\underline{\mathbf{n}}^*_{li}, {}^{p}\underline{\mathbf{n}}^*_{ri}\}$ are the left and right desired projection-lines of a lower-leg rod. One interesting advantageous side of the representation of a projection-line is that ${}^{c}\underline{\mathbf{n}}_{ji}$, meanwhile, corresponds to the unit vector orthogonal to the interpretation plane which is defined by the $3D$ line $\mathscr{L}_{ji}$ lying along the surface of the cylindrical rod and the center of projection (see Fig. 2) [8]. The transformations between the expressions of the projection-line in pixel coordinates ${}^{p}\underline{\mathbf{n}}$ and in camera

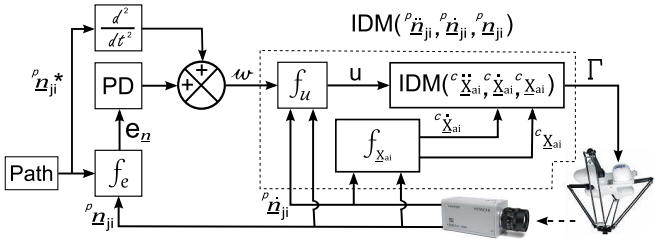

Fig. 3. Image-space computed-torque control (IS-CTC).

frame coordinates ${}^{c}\underline{\mathbf{n}}$ are given as below:

$$ {}^{c}\underline{\mathbf{n}} = \frac{\mathbf{K}^{T}\,{}^{p}\underline{\mathbf{n}}}{\|\mathbf{K}^{T}\,{}^{p}\underline{\mathbf{n}}\|}, \quad {}^{p}\underline{\mathbf{n}} = \frac{\mathbf{K}^{-T}\,{}^{c}\underline{\mathbf{n}}}{\|\mathbf{K}^{-T}\,{}^{c}\underline{\mathbf{n}}\|} \tag{13} $$

where $\mathbf{K}$ is the intrinsic camera matrix. The geometry of a cylindrical lower-leg rod allows one to calculate its direction as well (see Fig. 2) through its projection-lines as follows:

$$ {}^{c}\underline{\mathbf{x}}_{ai} = \frac{{}^{c}\underline{\mathbf{n}}_{li} \times {}^{c}\underline{\mathbf{n}}_{ri}}{\|\,{}^{c}\underline{\mathbf{n}}_{li} \times {}^{c}\underline{\mathbf{n}}_{ri}\,\|} \tag{14} $$

Hence, one can derive the differential relation between the orientation vector of a lower-leg rod and its projection-lines in pixel coordinates by differentiating (14) and the first expression in (13). Afterwards, the following expression comes up:

$$ {}^{c}\dot{\underline{\mathbf{x}}}_{ai} = \mathbf{L}_{n_i} \begin{bmatrix} {}^{p}\dot{\underline{\mathbf{n}}}_{li} \\ {}^{p}\dot{\underline{\mathbf{n}}}_{ri} \end{bmatrix} \tag{15} $$

where $\mathbf{L}_{n_i} \in \Re^{3\times 6}$ is the interaction matrix between the velocities of a lower-leg rod $3D$ direction and its projection-lines:

$$ \mathbf{L}_{n_i} = [\mathbf{f}_i]_v \begin{bmatrix} [{}^{c}\underline{\mathbf{n}}_{ri}]_{\times}^{T}[\mathbf{k}_{li}]_v\,\mathbf{K}^{T} & [{}^{c}\underline{\mathbf{n}}_{li}]_{\times}[\mathbf{k}_{ri}]_v\,\mathbf{K}^{T} \end{bmatrix} \tag{16} $$

and where $\mathbf{f}_i$ and $\mathbf{k}_{ji}$ are:

$$ \mathbf{f}_i = {}^{c}\underline{\mathbf{n}}_{li} \times {}^{c}\underline{\mathbf{n}}_{ri}, \qquad \mathbf{k}_{ji} = \mathbf{K}^{T}\,{}^{p}\underline{\mathbf{n}}_{ji} \tag{17} $$

The $[\cdot]_{\times}$ represents the skew-symmetric matrix associated to the vector cross-product and $[\cdot]_v \in \Re^{3\times 3}$ denotes the differential tensor matrix for a given vector which will be scaled down to a unit vector by its norm:

$$ \frac{d}{dt}\left(\frac{\mathbf{v}}{\|\mathbf{v}\|}\right) = [\mathbf{v}]_v\,\dot{\mathbf{v}} = \frac{1}{\|\mathbf{v}\|}\left(\mathbf{I}_3 - \frac{\mathbf{v}\mathbf{v}^{T}}{\|\mathbf{v}\|^2}\right)\dot{\mathbf{v}} \tag{18} $$

Then, the corresponding control $\mathbf{u}_i$ is derived from (9) as follows:

$$ \mathbf{u}_i = \dot{\mathbf{L}}_{n_i} \begin{bmatrix} {}^{p}\dot{\underline{\mathbf{n}}}_{li} \\ {}^{p}\dot{\underline{\mathbf{n}}}_{ri} \end{bmatrix} + \mathbf{L}_{n_i}\,\boldsymbol{\omega}_i \tag{19} $$

where $\boldsymbol{\omega}_i = \begin{bmatrix} {}^{p}\ddot{\underline{\mathbf{n}}}_{li}{}^{T} & {}^{p}\ddot{\underline{\mathbf{n}}}_{ri}{}^{T} \end{bmatrix}^{T}$ is obtained with (12) and (11). In Figs. 3, 4 and 5, the function $f_{\underline{\mathbf{x}}_{ai}}$ calculates the directions of the lower-legs and their first-order derivatives via (14) and (15) in order to be used in the IDM.

2) Leg orientations space computed-torque control (LS-CTC): Since the $3D$ directions of lower-legs stand in almost at the heart of the IDM, the sensor-signal is chosen as $\{\underline{\mathbf{x}}_{ai}|_{i=1}^{4}\}$ for control (see Fig. 4) and the error is directly regulated over them in order to have an efficient performance:

$$ \mathbf{e}_i = {}^{c}\underline{\mathbf{x}}^{*}_{ai} - {}^{c}\underline{\mathbf{x}}_{ai} \tag{20} $$

where ${}^{c}\underline{\mathbf{x}}^{*}_{ai}$ is the desired i^{th} lower-leg orientation and the $\mathbf{e}_i \in \Re^{3\times 1}$ is the error vector for the i^{th} lower-leg. Afterwards, the auxiliary control law, $\mathbf{u}_i \in \Re^{3\times 1}$, can simply be calculated through (20) and (11) ($\boldsymbol{\omega}_i = \mathbf{u}_i$). One can directly use it as the final control signal since the inverse dynamic model is represented in the lower-legs orientations space.

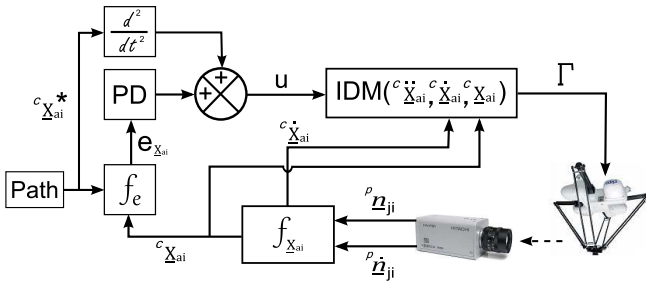

Fig. 4. Leg orientations space computed-torque control (LS-CTC).

3) Cartesian-space computed-torque control (CS-CTC): In Cartesian space, the end-effector pose $\boldsymbol{\chi}$ of the Quattro, *which is calculated through the linear function $f_{\boldsymbol{\chi}}$ as in* [6] *by using the projection-lines of the lower-legs* rather than through a non-linear pose estimation from a set of points on a grid, is used as a sensor-signal in control (see Fig. 5). Then, the error is defined as difference in poses as follows:

$$ \mathbf{e} = {}^{c}\boldsymbol{\chi}^{*} - {}^{c}\boldsymbol{\chi} \tag{21} $$

where ${}^{c}\boldsymbol{\chi}^{*} \in \Re^{4\times 1}$ is the desired end-effector pose. Then, the control signal $\mathbf{u}_i$ can be found from (9) as below:

$$ \mathbf{u}_i = \dot{\mathbf{L}}_{\chi_i}\,{}^{c}\dot{\boldsymbol{\chi}} + \mathbf{L}_{\chi_i}\,\boldsymbol{\omega} \tag{22} $$

where $\boldsymbol{\omega} = {}^{c}\ddot{\boldsymbol{\chi}}$ is computed from (21) and (11), and $\mathbf{L}_{\chi_i} \in \Re^{3\times 4}$ is the inverse differential kinematic model between the end-effector pose and a lower-leg $3D$ direction vector [18].

IV. RESULTS

The vision-based control simulations are conducted on the ADAMS/Simulink platform. The simulation frequency is 500 Hz. A 0.2 m diameter circle motion with 2 m/s maximum velocity and $4G$ maximum acceleration is planned out such that it spans XY, XZ and YZ planes. The simulations are executed in three different control-spaces and results are compared. Table I gathers the accuracy obtained under various types of noises for each control law, where the accuracy of the control laws is assessed in terms of mean (bold) and standard deviation (italic) values of the tracking errors that are evaluated in the Cartesian space. Firstly, $100\mu m$ of uncertainty is injected on the 3D coordinates of the extremity points $\{\mathbf{A}_{i1}, \mathbf{B}_{i1}\}$ of the lower-legs so as to imitate the effects of clearances in passive joints, assembly errors and etc. (This noise has a great impact since it directly changes the orientations of the lower-legs and a good calibration is a must in case of ignorance that kind of mechanical errors.) Afterwards, for the sensory noise, the locations of the edgels of a lower-leg are orthogonally perturbed (with respect to its noiseless projection-line) in between $[-2,+2]$ pixels which will make the new fitted line take a slight deflection off the

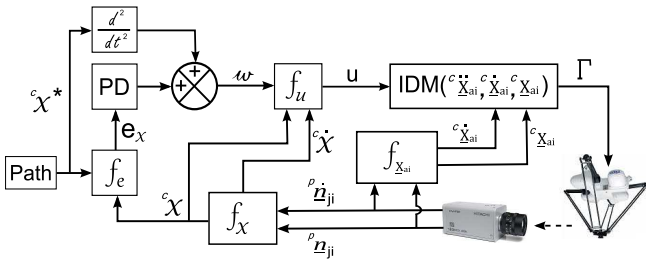

Fig. 5. Cartesian-space computed-torque control (CS-CTC).

previous noiseless one. We also performed (see last row in Table I) a computed-torque control with a feedback pose estimated by direct observation of the end-effector (EE-CTC) instead of the lower-legs and the feedback χ is corrupted with a $\{100\,\mu m, 0.01^\circ\}$ noise, corresponding to state-of-the-art accuracy of high-speed vision. Figure 6 shows the reference circle motion in time. The results for the fourth row of the Table I are plotted in Figs. 7 to 10. Figures 7, 8, and 9 depict the traces of the performed trajectories and the applied torques. Figure 10 shows the Euclidean and rotational distances to the reference trajectory, respectively. Observing the results in Table I, one can immediately conclude that CS-CTC performs better and IS-CTC performs worse than the others. It is surprising to have that result while our expectations are put on the IS-CTC since the control variable ${}^p\underline{\mathbf{n}}$ is directly defined in the very sensor-space. However, differences on the orders of magnitudes of the errors are not so decisive to promote one over the others. Going into details of the results, one can end up that: IS-CTC and LS-CTC seem robust only to the errors in the sensor space (line fitting easily smooths out the sensor noise), while being sensitive to the mechanical errors. They are slightly better in rotation but slightly worse in translation. It seems that, the closer the control-variable to the operational space of the robot is, the better the results are. Moreover, the superior robustness of the CS-CTC to the both types of noise (mechanical and sensory) can be explained by the fact that the pose is calculated from the projection-lines of the lower-legs. This imposes explicitly the closed-loop kinematic constraint that is helping to smooth out the 3D errors. In the applied torques the CS-CTC performs better as well, while the others are more oscillatory and peaky. Let us finally remark that the EE-CTC is worse than any other proposed controls, which confirms that observing the lower-legs is probably one good way to enhanced accuracy. Note that those results were achieved with a PD+FF controller under the assumption of a perfect decoupling and linearizing of the dynamics. In practice, due to noise, this assumption might not be valid and the actual performance of the system should be improved by making call to advanced control.

V. CONCLUSIONS

In this paper, for a competent control performance of a parallel robot, the variable-spaces have been explored regarding a specific IDM expressed in leg orientations. The prevailing results are brought by the CS-CTC. This outcome suggests that: (i) the chosen variable set should be as close

TABLE I

TRACKING ERRORS EVALUATED IN CS $(\mu m, deg)$.

	IS	$(\underline{\mathbf{n}})$	LS	$(\underline{\mathbf{x}}_{ai})$	CS	(χ)
no noise	**408**	**0.23°**	**356**	**0.23°**	**359**	**0.23°**
	239	*0.19°*	*190*	*0.16°*	*182*	*0.16°*
$100\,\mu m$	**674**	**0.32°**	**652**	**0.37°**	**553**	**0.36°**
	447	*0.26°*	*385*	*0.26°*	*269*	*0.25°*
$\pm 2\,pixels$	**553**	**0.22°**	**522**	**0.22°**	**529**	**0.34°**
	428	*0.18°*	*371*	*0.16°*	*236*	*0.23°*
$100\,\mu m$	**881**	**0.28°**	**899**	**0.29°**	**560**	**0.36°**
$\pm 2\,pixels$	*647*	*0.23°*	*703*	*0.24°*	*264*	*0.25°*
$100\,\mu m$	-		-		**862**	**0.56°**
0.01°					*400*	*0.38°*

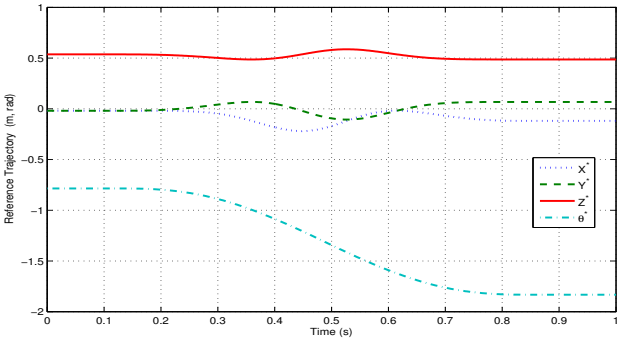

Fig. 6. CS reference trajectory expressed in the camera frame.

as possible to the sensor space, (ii) the models should be able to be directly / compactly expressed by that variable set and (iii) the control-variable should be in the operational space of the robot and has to be linearly calculated by the chosen variable set. In this work, the only but quite challenging *assumption* for the moment is that vision can perceive the leg edges and their velocities at high speed and control rates. However, it seems feasible in the close future, since the sensing technology is fast by this point [19]. Thereafter, the impact of the noisy $\underline{\mathbf{n}}$ on IDM should be analyzed as well, namely as decoupling is concerned. We build also the following perspectives for the future: (i) Shall we be able to do identification / calibration from $\{\underline{\mathbf{n}}, \dot{\underline{\mathbf{n}}}\}$ and Γ ? and (ii) As stated earlier, shall the way towards optimum lie in the merging of the measurements in different spaces, such as $\mathbf{q}$ and χ (calculated from $\underline{\mathbf{n}}$) ? Finally, we conclude that this work bricks the last hole up theoretically in the control-oriented *unified system* and, once the real-time tracking of the edges is worked out, will let it be practically put to good use in parallel robots as a favourable option.

REFERENCES

[1] Jean P. Merlet, "Still a long way to go on the road for parallel mechanisms", *Proc. of the ASME 27th Biennial Mechanisms and Robotics Conference*, Montreal, Quebec, 2002.

[2] Jean P. Merlet, *Parallel Robots*, Kluwer Academic Publishers, 2000.

[3] A. Vivas, P. Poignet, F. Pierrot, "Predictive functional control of a parallel robot", *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS'03*, pp 2785-2790, Las Vegas, USA, October 2003.

[4] V. Nabat, S. Krut, O. Company, P. Poignet, F. Pierrot, "On the Design of a Fast Parallel Robot Based on Its Dynamic Model", *Springer-Verlag Berlin Heidelberg, Experimental Robotics*, January 30, 2008.

[5] Paul C. Mitiguy, Thomas R. Kane, "Motion Variables Leading to Efficient Equations of Motion", *Int. Journal of Robotics Research*, 15(5), pp 522-532, 1996.

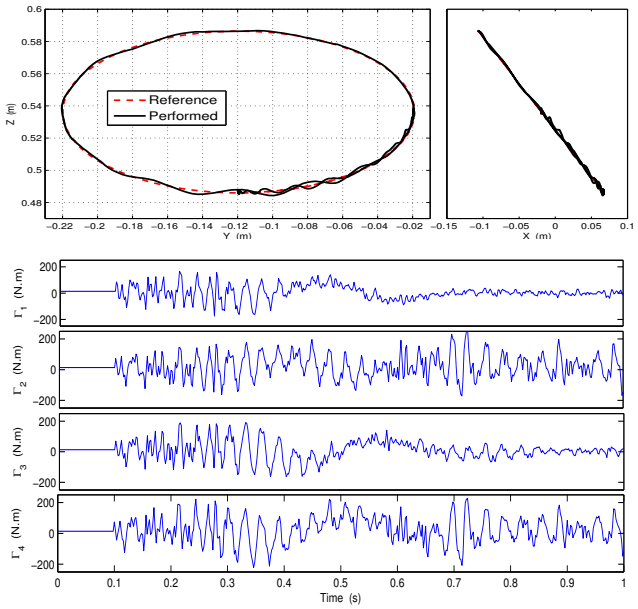

Fig. 7. Reference (red) and performed (black) superimposed trajectories in ZY and ZX planes (top), motor torques (bottom) for the IS-CTC.

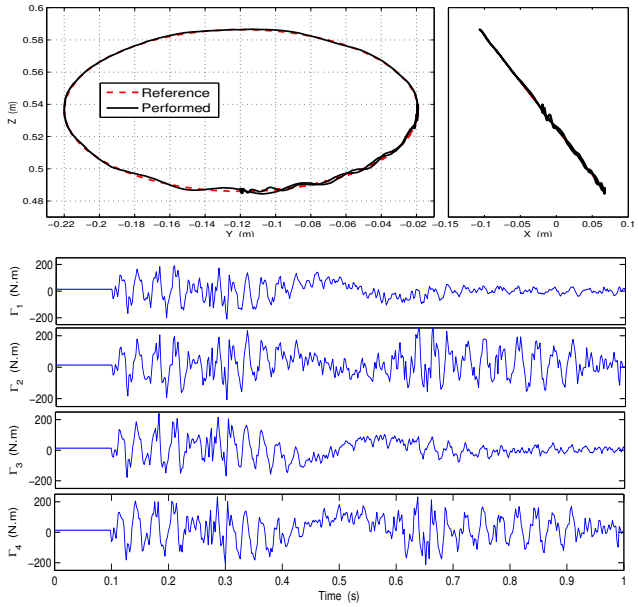

Fig. 8. Reference (red) and performed (black) superimposed trajectories in ZY and ZX planes (top), motor torques (bottom) for the LS-CTC.

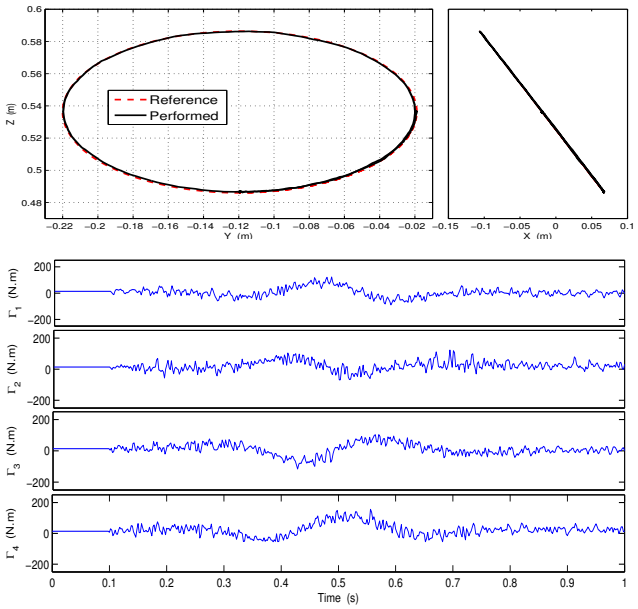

Fig. 9. Reference (red) and performed (black) superimposed trajectories in ZY and ZX planes (top), and motor torques (bottom) for the CS-CTC.

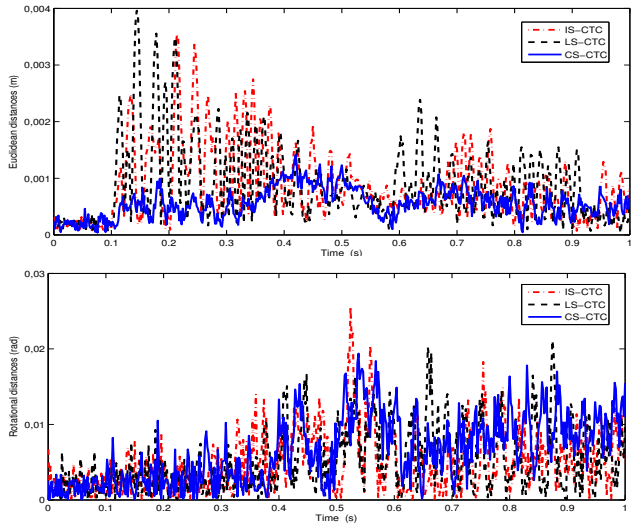

Fig. 10. Euclidean and rotational distances to the reference trajectory.

[6] E. Ozgur, N. Andreff and F. Martinet, "Vector-Based Dynamic Modeling and Control of the Quattro Parallel Robot by means of Leg Orientations", *IEEE Int. Conf. on Robotics and Automation, (ICRA'10)*, Alaska, USA, 2010.

[7] E. Ozgur, N. Andreff and F. Martinet, "On the adequation of dynamic modelling and control of parallel kinematic manipulators", *The 1st Joint International Conference on Multibody System Dynamics, (IMSD'10)*, Lappeenranta, Finland, 2010.

[8] N. Andreff, T. Dallej and F. Martinet, "Image-based visual servoing of Gough-Stewart parallel manipulators using legs observation.", *In Joint Issue of IJCV and IJRR on Vision and Robotics*, 2006.

[9] W. Khalil, O. Ibrahim, "General Solution for the Dynamic Modeling of Parallel Robots", *IEEE Int. Conf. on Robotics and Automation, (ICRA'04)*, New Orleans, LA, 2004.

[10] Kane, T.R., Levinson, D.A., "Dynamics: Theory and Applications", *McGraw Hill* New York, 1985.

[11] W. Khalil, E. Dombre, "Modeling, identification and control of robots", London, 2002.

[12] F. Paccot, N. Andreff, P. Martinet, " A review on dynamic control of parallel kinematic machines: theory and experiments.", *International Journal of Robotics Research* 28, 3, 395-416, Feb. 2009.

[13] B. Dasgupta and P. Choudury, "A general strategy based on the Newton-Euler approach for the dynamic formulation of parallel manipulators", *Mechanism and Machine Theory* 34, 801824, 1999.

[14] M. Callegari, M. Palpacelli, M. Principi, "Dynamics modelling and control of the 3-rcc translational platform.", *Mechatronics*, 16, 589-605, 2006.

[15] F. Bullo and R.M. Murray, "Proportinal Derivative (PD) Control on the Euclidean Group.", *CDS Technical Report 95-010*, 1995.

[16] C. Samson, M. Le Borgne, B. Espiau, "Robot Control: the Task Function Approach.", *Clarendon Press, Oxford University Press*, Oxford, UK, 1991.

[17] L. Baron, J. Angeles, "The on-line direct kinematics of parallel manipulators under joint-sensor redundancy", *Advances in Robot Kinematics*, pp 126-137, Kluwer, 1998.

[18] T. Dallej, N. Andreff and F. Martinet, "Visual Servoing of Par4 using Leg Observation", *The 32nd Annual Conf. of the IEEE Industrial Electronics Society, (IECON'06)*, pp 3782-3787, Paris, France, 2006.

[19] Y. Nakabo, M. Ishikawa, H. Toyoda, S. Mizuno, "1ms column parallel vision system and its application of high speed target tracking", *IEEE Int. Conf. on Robotics and Automation, (ICRA'00)*, 2000.