A vision-based generic dynamic model of PKMs and its experimental validation on the Quattro parallel robot.

Erol Ozgür, Redwan Dahmouche, Nicolas Andreff, Philippe Martinet

2014 IEEE/ASME International Conference on
Advanced Intelligent Mechatronics (AIM)
Besançon, France, July 8-11, 2014

A vision-based generic dynamic model of PKMs and its experimental validation on the Quattro parallel robot

Erol Özgür[1], Redwan Dahmouche[2], Nicolas Andreff[3], Philippe Martinet[4]

Abstract— **In this paper, we present a dynamic modeling method for parallel kinematic manipulators. This method is built upon the postures (i.e., 3D orientation vectors, lengths and self rotations) of the kinematic elements which form the kinematic chains of the robot. Regarding the structure of the above method, computer vision emerges as a good option to obtain the postures of these kinematic elements. This method is then validated experimentally on the commercial Adept Quattro parallel robot.**

I. INTRODUCTION

Unlike serial manipulators, parallel kinematic manipulators (PKMs) are designed with two main performance objectives in mind: *high speed* and *precision* [1]. To manipulate precisely such a PKM at high speed, it is well known that PKM should be controlled at dynamic level which requires us to develop a dynamic model. If kinematic control is used instead of dynamic control at high speed, then the unconsidered physical effects (e.g., inertial forces) reduce precision and may destabilize the control.

Therefore, in this paper, we want to contribute on exploitability of these performance objectives of PKMs by proposing a *vision-based* and *generic* dynamic model.

The first question to ask here is: why do we choose a vision-based approach? First of all, vision is particularly relevant to the control of large class of PKMs. It was shown that the end-effector pose control is more appropriate for PKMs than joint configuration control [2], [3], [4], [5]. Indeed, joint configuration of a PKM does not fully describe the robot configuration. A well-known example for this is the Gough-Stewart platform which may have 40 real moving-platform configurations for a given single joint configuration [6]. For this reason, the forward kinematic model (FKM) of PKMs is complicated to compute. The precision of the estimated end-effector pose through FKM depends on the completeness of the robot geometry modeling and the identification accuracy of the model whereas the moving platform configuration describes the complete geometry of the robot without ambiguity.

Vision appears then to be a relevant solution for PKM modeling and control since it allows the moving platform pose to be sensed. Vision also allows for relative positioning of the robot in a dynamic environment. This makes the control of a robot robust to uncertainties of the dynamic environment and of the robot kinematic models. The only drawback for a standard vision system is that it is too slow (50-60 Hz) to feed back the dynamic control. This discourages many not to use it in dynamic control. Fortunately we now have a solution to this problem [7].

The second question to ask is: as a consequence of vision can we come up with a generic model for PKMs? Most of the PKMs have very complex and different geometries so that desired performance objectives are satisfied. This makes usually their models robot-specific. Also if one uses only conventional motorized joint information, which is poor for PKMs because of existing many other passive joints, to write the models of these complex geometries, then models of PKMs inflate, become slow, hard to understand and to implement. This inevitably urges one to offer simplifications [8] and to omit some of the modeling errors, thus giving simplified and fast [9] but less accurate new models for control. Therefore, modeling of PKMs, based on joint sensing, becomes inefficient when the geometric complexity of the robot increases. We see once more that joint configuration information is not relevant for modeling. Then we ask ourselves, what if we had used visual information? Could it be possible to write one simple, intuitive, and generic model valid for all PKMs? Again, fortunately we have a positive answer to this question [10].

In short, the main contribution of this paper is to unify vision and dynamic modeling for a new vision-based generic dynamic model, and to experimentally show the correctness of the proposed model on the Adept Quattro parallel robot.

This paper goes on as follows: Section II presents the dynamic model of the Quattro parallel robot which is based on the generic method proposed in [10]; Section III shows how computer vision can compute the dynamic states of the previously proposed generic dynamic model; Section IV describes the experimental validation process of the proposed vision-based generic dynamic model and gives the experimental results for the Adept Quattro parallel robot; and Section V concludes the paper.

II. DYNAMIC MODEL OF THE QUATTRO ROBOT

The generic dynamic modeling method proposed in [10] is inspired from the ideas from Khalil [3], Tsai [11], and mainly Kane [12], [13]. This generic method has the following two original characteristics: (i) It uses lines to model the *concrete* moving bodies rather than the *abstract* joint axes. This enhances the visual perception of a robot, such that a human brain can almost vividly imagine the robot's motion

This work was supported by project ANR-VIRAGO.

[1]Author is with Pascal Institute, IFMA, Clermont-Ferrand, France name.surname@ifma.fr

[2,3]Authors are with FEMTO-ST, Besançon, France name.surname@femto-st.fr

[4]Author is with Ecole Centrale de Nantes, IRCCYN, Nantes, France name.surname@irccyn.ec-nantes.fr

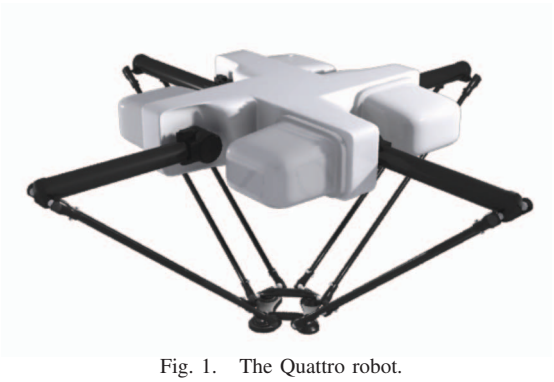

Fig. 1. The Quattro robot.

by just reading the equations; (ii) Equations of motion use simple linear vector algebra and are in compact form. This eases the codability and the fast solvability of equations on the computer. Even for the most complex robots, the inverse dynamic model can be worked out with pen and paper. Here, we show its application on the Quattro parallel robot.

A. The Quattro Robot

The Quattro is composed of four identical kinematic legs which carry the articulated nacelle. See Fig. 1. Each of the 4 kinematic legs is actuated from the base by a revolute motor located at $\mathbf{P}_i$. A kinematic leg has two consecutive kinematic elements: an upper-leg $[\mathbf{P}_i\mathbf{A}_i]$ and a lower-leg $[\mathbf{A}_i\mathbf{B}_i]$. Lower-leg and upper-leg are attached to each other at $\mathbf{A}_i$. At the top, the upper-legs are connected to the motors, while at the bottom, the lower-legs are connected to the articulated nacelle. See Fig. 2. The articulated nacelle is designed with four kinematic elements [14]: the two lateral kinematic elements (either $[\mathbf{B}_1\mathbf{B}_2]$, $[\mathbf{B}_3\mathbf{B}_4]$ or $[\mathbf{C}_1\mathbf{C}_2]$, $[\mathbf{C}_3\mathbf{C}_4]$) and the two central kinematic elements ($[\mathbf{C}_3\mathbf{C}_2]$, $[\mathbf{C}_4\mathbf{C}_1]$) linking lateral ones with revolute joints. See Fig. 3. The configurations of the kinematic elements (KE) are defined by the unit direction vectors of the bodies rather than the non-linear joint coordinates. The static state of the robot is therefore totally and redundantly defined by the unit vectors ($\underline{\mathbf{x}}$). The end-effector pose (X) is composed of the origin ($\mathbf{E}$)

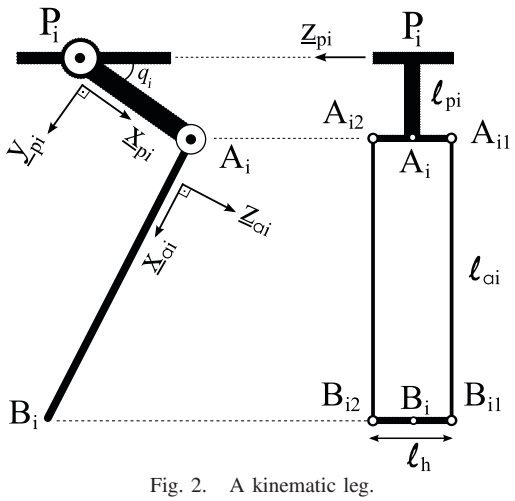

Fig. 2. A kinematic leg.

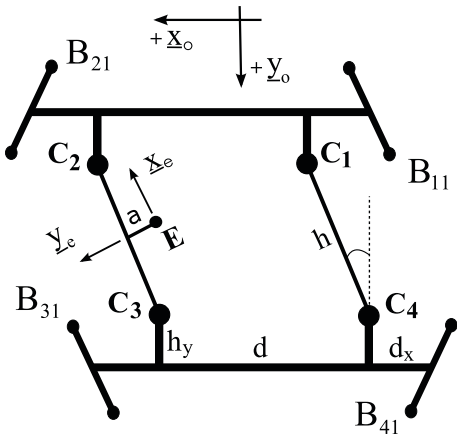

Fig. 3. Nacelle.

of the end-effector frame and the orientation of the $[\mathbf{C}_3\mathbf{C}_2]$ moving platform ($\underline{\mathbf{x}}_e$):

$$\mathrm{X} \triangleq \begin{bmatrix} \mathbf{E} \\ \underline{\mathbf{x}}_e \end{bmatrix} \tag{1}$$

B. Kinematics of the Quattro Robot

A point on a kinematic leg can be reached by successive additions of the scaled direction vectors of the kinematic elements. For example, an attachment point $\mathbf{B}_i$ of the kinematic leg to the nacelle can be expressed as follows:

$$\mathbf{B}_i = \mathbf{P}_i + \ell_{pi}\underline{\mathbf{x}}_{pi} + \ell_{ai}\underline{\mathbf{x}}_{ai} \tag{2}$$

We can then write straightforward the linear velocity and acceleration of this point with respect to the fixed robot base frame as below:

$$\dot{\mathbf{B}}_i = \ell_{pi}\dot{\underline{\mathbf{x}}}_{pi} + \ell_{ai}\dot{\underline{\mathbf{x}}}_{ai}, \qquad \ddot{\mathbf{B}}_i = \ell_{pi}\ddot{\underline{\mathbf{x}}}_{pi} + \ell_{ai}\ddot{\underline{\mathbf{x}}}_{ai} \tag{3}$$

Angular velocity ω and acceleration $\dot{\omega}$ of a kinematic element of the Quattro robot can be written with respect to the fixed robot base frame by means of only the direction vector and the derivatives of the direction vector of the kinematic element:

$$\omega \triangleq \underline{\mathbf{x}} \times \dot{\underline{\mathbf{x}}}, \qquad \dot{\omega} \triangleq \underline{\mathbf{x}} \times \ddot{\underline{\mathbf{x}}} \tag{4}$$

C. Kinematic Constraints of the Quattro Robot

From the closed-loop kinematics of the robot, it is possible to relate the velocity of a kinematic element to the velocity of the end-effector pose through a kinematic constraint matrix:

$$\dot{\underline{\mathbf{x}}} = M\dot{\mathrm{X}} \tag{5}$$

where $M \in \Re^{3\times 6}$ is the kinematic constraint matrix which relates the motion of a kinematic element to the motion of the end-effector. The reader can look at [15] for more details.

D. Motion Basis of the Quattro Robot

Remember that the state of the robot is defined by the unit direction vectors of the kinematic elements:

$$\{\underline{\mathbf{x}}_{pi}, \underline{\mathbf{x}}_{ai}, \underline{\mathbf{x}}_{bi}\}, \qquad i = 1,\ldots,4 \tag{6}$$

where $\underline{\mathbf{x}}_{bi} = \delta_i \underline{\mathbf{x}}_e$ where $\delta_1 = \delta_3 = 0$, $\delta_2 = -1$, $\delta_4 = 1$. Therefore, we choose the motion basis of the Quattro robot as the time derivatives of these state variables:

$$\{\dot{\underline{\mathbf{x}}}_{pi}, \dot{\underline{\mathbf{x}}}_{ai}, \dot{\underline{\mathbf{x}}}_{bi}\}, \qquad i = 1,\ldots,4 \tag{7}$$

E. Kinematic Coordinates of the Quattro Robot

The kinematic coordinates express the velocity of the whole robot in the previously chosen motion basis. That is to say, velocity of every kinematic element is encoded in the kinematic coordinates. Kinematic coordinates are computed as partial derivatives of kinematic equations with respect to the motion basis of the robot. Table I tabulates the linear kinematic coordinates of the mass centers of the kinematic elements (see Fig. 4), and the angular kinematic coordinates of the kinematic elements.

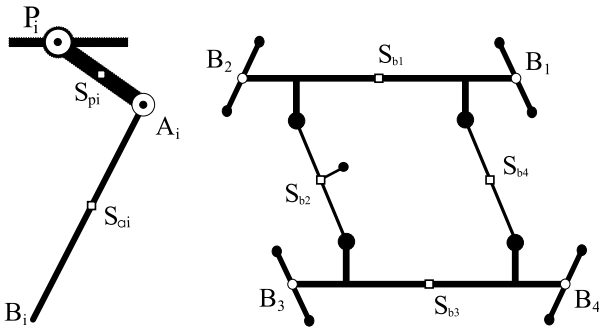

Fig. 4. Mass centers.

TABLE I

THE (TRANSPOSED) KINEMATIC COORDINATES OF THE QUATTRO ROBOT

	$\partial\dot{\mathbf{S}}_{pi}$	$\partial\omega_{pi}$	$\partial\dot{\mathbf{S}}_{ai}$	$\partial\omega_{ai}$	$\partial\dot{\mathbf{S}}_{bi}$	$\partial\omega_{bi}$
$\partial\dot{\underline{\mathbf{x}}}_{pi}$	$\frac{1}{2}\ell_{pi} I_3$	$[\underline{\mathbf{x}}_{pi}]^T_\times$	$\ell_{pi} I_3$	$\mathbf{0}$	$\ell_{pi} I_3$	$\mathbf{0}$
$\partial\dot{\underline{\mathbf{x}}}_{ai}$	$\mathbf{0}$	$\mathbf{0}$	$\frac{1}{2}\ell_{ai} I_3$	$[\underline{\mathbf{x}}_{ai}]^T_\times$	$\ell_{ai} I_3$	$\mathbf{0}$
$\partial\dot{\underline{\mathbf{x}}}_{bi}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\delta_i^2 \frac{1}{2} h I_3$	$[\underline{\mathbf{x}}_{bi}]^T_\times$

F. Dynamic Coordinates of the Quattro Robot

Dynamic coordinates (i.e., generalized forces) express the forces which contribute to the motion of the robot. Thus, non-contributing forces are eliminated. Fig. 5 and Table II explain the possible active and consequently reactive local forces may appear on a kinematic leg of the Quattro robot.

TABLE II

THE LOCAL FORCES AND TORQUES OF THE QUATTRO ROBOT

	Active		*Friction*		*Inertia**	
	Motor	*Gravity*	*Motor*	*Joint*	*Motor*	*KE*
Forces (pi)	$\mathbf{0}$	$\mathbf{f}_{\mathbf{g}(pi)}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{f}^*_{pi}$
Torques (pi)	$\tau_{\underline{\mathbf{x}}_{pi}\,\underline{\mathbf{z}}_{pi}}$	$\mathbf{0}$	$\bar{\tau}_{\underline{\mathbf{x}}_{pi}}$	$\mathbf{0}$	$\tau^*_{\underline{\mathbf{x}}_{pi}}$	τ^*_{pi}
Forces (ai)	$\mathbf{0}$	$\mathbf{f}_{\mathbf{g}(ai)}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{f}^*_{ai}$
Torques (ai)	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\bar{\tau}_{\underline{\mathbf{x}}_{ai}}$	$\mathbf{0}$	τ^*_{ai}
Forces (bi)	$\mathbf{0}$	$\mathbf{f}_{\mathbf{g}(bi)}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{f}^*_{bi}$
Torques (bi)	$\mathbf{0}$	$\mathbf{0}$	$\mathbf{0}$	$\bar{\tau}_{\underline{\mathbf{x}}_{bi}}$	$\mathbf{0}$	τ^*_{bi}

These local forces contain contributing and non-contributing parts for the motion of the robot. Dynamic coordinates are computed through the matrix-wise multiplication

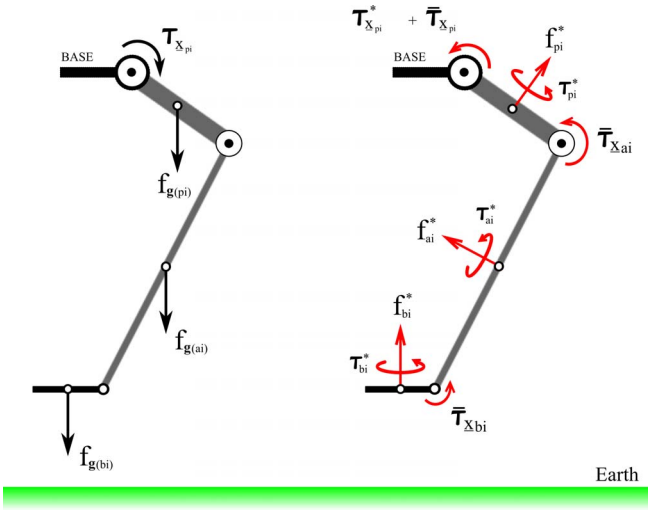

Fig. 5. Local forces and torques which act on one of the identical kinematic legs. Left figure shows the active forces and torques of a kinematic leg. Right figure shows the reactive inertial and frictional forces/torques which balance the active forces/torques.

of the Tables I and II:

$$\mathrm{F}_i = \Pi_i\, \Sigma_i, \qquad i = 1,...,4. \tag{8}$$

where $\mathrm{F}_i = [\mathrm{F}^T_{\underline{x}\,pi}, \mathrm{F}^T_{\underline{x}\,ai}, \mathrm{F}^T_{\underline{x}\,bi}]^T$ is the vector of dynamic coordinates of the i^{th} kinematic leg; Π_i is the 3×6 matrix whose elements are the cells of the Table I, where each cell is a 3×3; and Σ_i is the 6×1 vector obtained by row-wise summation of forces and torques in Table II.

G. Dynamic Constraints of the Quattro Robot

The dynamic constraints of the Quattro robot can be written from the d'Alembert's principle of virtual work as follows:

$$M_p^T\, \mathrm{F}_p + M_a^T\, \mathrm{F}_a + M_b^T\, \mathrm{F}_b = \mathbf{0}_{6\times 1} \tag{9}$$

where $\mathrm{F}_p \in \Re^{12\times 1}$, $\mathrm{F}_a \in \Re^{12\times 1}$ and $\mathrm{F}_b \in \Re^{12\times 1}$ are the stacked vectors of the dynamic coordinates:

$$\mathrm{F}_p = \begin{bmatrix} \mathrm{F}_{\underline{x}p1} \\ \vdots \\ \mathrm{F}_{\underline{x}p4} \end{bmatrix}, \quad \mathrm{F}_a = \begin{bmatrix} \mathrm{F}_{\underline{x}a1} \\ \vdots \\ \mathrm{F}_{\underline{x}a4} \end{bmatrix}, \quad \mathrm{F}_b = \begin{bmatrix} \mathrm{F}_{\underline{x}b1} \\ \vdots \\ \mathrm{F}_{\underline{x}b4} \end{bmatrix} \tag{10}$$

and where $M_j \in \Re^{12\times 6}$ is a matrix composed of the stacked constraint matrices of the corresponding kinematic elements for $j \in \{p, a, b\}$. These constraint matrices come from (5).

H. Inverse Dynamics of the Quattro Robot

With some algebraic manipulations, starting from (9), we can work out a linear representation of the dynamic constraints as shown below:

$$A(\underline{\mathbf{x}})\, \Gamma + \mathbf{b}(\ddot{\underline{\mathbf{x}}}, \dot{\underline{\mathbf{x}}}, \underline{\mathbf{x}}) = \mathbf{0} \tag{11}$$

where $A \in \Re^{6\times 4}$ and $\mathbf{b} \in \Re^{6\times 1}$ are the configuration matrix and the contributing efforts vector of the passive kinematic elements of the robot, respectively. We can write straightforward the inverse dynamics of the Quattro robot from (11) as follows:

$$\Gamma = -A^{\dagger}(\underline{\mathbf{x}})\, \mathbf{b}(\ddot{\underline{\mathbf{x}}}, \dot{\underline{\mathbf{x}}}, \underline{\mathbf{x}}) \tag{12}$$

where Γ is the vector of motor torques, and it is computed by means of the direction vectors of the kinematic elements and their time derivatives as a linear solution of (11).

III. COMPUTER VISION FOR KINEMATIC ELEMENTS' POSTURES

We use a computer vision method which is explained in [16] to estimate the pose and velocity of the end-effector of the robot at high rates ($\approx 400Hz$). The computer vision method is based on the sequential acquisition approach of a calibrated camera. In this method a rigid object is abstracted as a set of 3D points, and it is observed by grabbing single sub-images successively where each sub-image contains a single visual point at a time. Afterwards, virtual visual servoing (VVS) [17] minimizes an error defined between the visual points observed sequentially from the moving rigid object and their corresponding points calculated at successive instants from a virtual model of this rigid object. Thus, VVS makes virtual model converge to the current pose and velocity of the real object.

A. Projection Model

The projection model of a set of 3D points $\mathbf{P}_i$ resulting in a visually deformed shape of the object in the image is thus given by:

$$\left[\begin{array}{c} m_i \\ 1 \end{array}\right] \times [\mathbf{K}|\mathbf{0}]\, {}^c\mathbf{T}_o\, \delta\mathbf{T}_i \left[\begin{array}{c} {}^o\mathbf{P}_i \\ 1 \end{array}\right] = \mathbf{0} \qquad \forall i = 1...n \quad (13)$$

where n is the number of 2D-3D correspondences, ${}^o\mathbf{P}_i$ are the points expressed in the object reference frame, m_i is the projected point in image coordinates, ${}^c\mathbf{T}_o$ is the homogeneous transformation matrix between the object and camera frames at a reference time t_{ref}, and $\delta\mathbf{T}_i$ is 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 distortion is not shown here but is compensated for.

B. Constant Acceleration Motion Model

In dynamic control process of a robot, control inputs are acceleration signals. We assume that between two control inputs the robot moves with a given acceleration under constant acceleration motion model. Therefore, the displacement $\delta\mathbf{T}_i$ in the projection model of a moving rigid object (i.e., end-effector) should also evolve under constant acceleration motion model. Thus the velocity screw $\xi = (\upsilon, \omega)$ of the moving rigid object at time t_i becomes:

$$\xi(t_i) = \xi(t_{ref}) + \int_{t_{ref}}^{t_i} \dot{\xi}\, dt \quad (14)$$

and subsequently the displacement $\delta\mathbf{T}_i$ is written as follows:

$$\delta\mathbf{T}_i = \int_{t_{ref}}^{t_i} r\left(\xi(t_{ref}) + \int_{t_{ref}}^{t_i} \dot{\xi}\, dt\right) dt \quad (15)$$

where r is the reshaping operator which transforms kinematic screw into a 4×4 matrix:

$$r\left(\begin{array}{c} \upsilon \\ \omega \end{array}\right) = \left[\begin{array}{cc} [\omega]_\times & \upsilon \\ \mathbf{0} & 0 \end{array}\right] \quad (16)$$

C. Virtual Visual Servoing Control Law

The estimation method is based on minimization of the reprojection errors built upon (13):

$$\mathbf{e}_u = \mathbf{m} - \mathbf{m}^* \quad (17)$$

where $\mathbf{m} \in \Re^{2n\times 1}$ is the stacked vector of the calculated image coordinates of the n reprojected points from the model of the moving rigid object at successive instants, and $\mathbf{m}^* \in \Re^{2n\times 1}$ is the stacked vector of the measured image coordinates of the n visual points observed sequentially from itself of the moving rigid object by a calibrated camera.

One can obtain an exponential decrease of the error by imposing:

$$\dot{\mathbf{e}}_u = -\lambda\, \mathbf{e}_u \quad (18)$$

The derivative of the reprojection error with respect to the virtual time u of VVS [7] is given by:

$$\frac{d\,\mathbf{e}_u}{du} = \frac{d\,\mathbf{m}}{du} = \mathbf{L}\left[\begin{array}{c} \xi_u \\ \dot{\xi}_u \end{array}\right] \quad (19)$$

where $\mathbf{L} \in \Re^{2n\times 12}$ is a matrix which relates the object velocity ξ_u and acceleration $\dot{\xi}_u$ to the image velocity of the set of image points $\mathbf{m}$. The subscript u indicates that the velocity and acceleration twists are virtual. They do not correspond to twists related to the actual object but to twists related to the virtual object model which evolves along the virtual time u to converge to the same state as the real object.

The virtual control law can be written from (17), (18) and (19) as follows:

$$\left[\begin{array}{c} \xi_u \\ \dot{\xi}_u \end{array}\right] = -\lambda\, \mathbf{L}^\dagger \left(\mathbf{m}({}^c\hat{\mathbf{T}}_o, \widehat{\xi}) - \mathbf{m}^*(t)\right), \quad \lambda > 0 \quad (20)$$

where ${}^c\hat{\mathbf{T}}_o$ and $\widehat{\xi}$ are the previous estimates of ${}^c\mathbf{T}_o$ and ξ. Note that $\mathbf{L}$, $\mathbf{m}$ and $\mathbf{m}^*$ are composed of rows that are evaluated at successive time instants. Equation (20) provides the pose and velocity correction vector. The object velocity is thus obtained by integrating the acceleration $\dot{\xi}_u$ and the pose is obtained by exploiting both velocity and acceleration parts of the virtual control law.

The virtual object model converges to the same state as the real object usually in one iteration. This single-iteration VVS and the sequential small sub-image acquisition approach increase the estimation speed to several hundreds Hz.

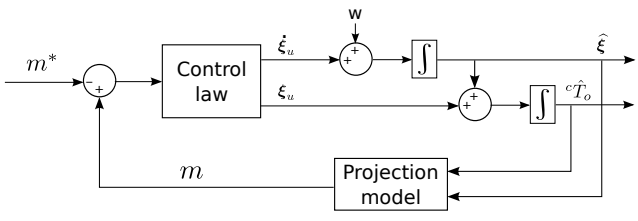

Fig. 6. Pose and velocity estimation with virtual visual servoing scheme.

These estimated pose and velocity variables can be transformed to the robot's kinematic elements' postures and to their velocities through the kinematic models of the robot.

Figure 6 shows the VVS scheme, and Fig. 7 shows the integration of VVS scheme into a computed torque control (CTC) scheme. As it is seen in Figs. 6 and 7, the computation of the pose velocity can be fed forward by the acceleration output **w** of the PID controller of the CTC scheme. **w** reflects the real acceleration of the robot between two control sampling instances. The VVS scheme and the robot run in parallel at the same frequency.

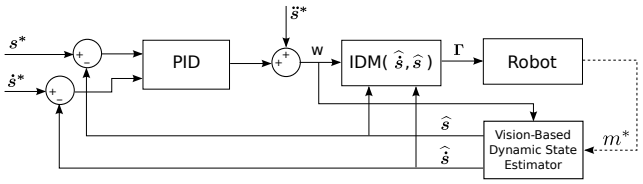

Fig. 7. Vision-based computed torque control scheme.

IV. EXPERIMENTS

We validated the presented vision-based generic dynamic modeling method on the Adept Quattro parallel robot (see Fig. 8) as follows: (i) we prepared a desired trajectory;

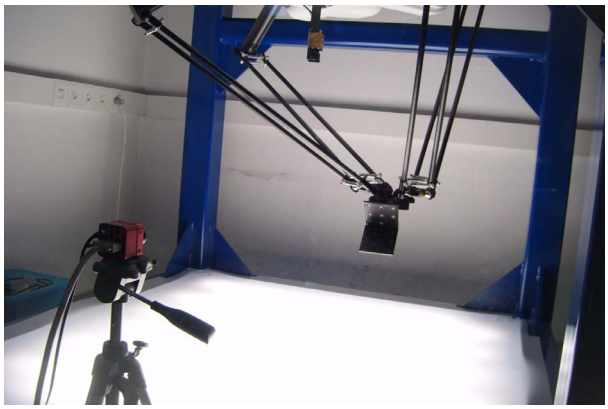

Fig. 8. Experimental setup.

(ii) we moved the Quattro parallel robot on this desired trajectory through the Adept's built-in controllers (i.e., joint-space PID control); (iii) we measured the control torques of the motors which are coming from the Adept's built-in controllers while the Quattro parallel robot was moving on the desired trajectory, and we recorded these torques as ground truth; (iv) and also at the same time we observed the motion (i.e., pose and velocity) of an artificial pattern which is fixed to the end-effector of the Quattro robot by a calibrated camera as explained in Section III; (v) afterwards we calculated the postures of the kinematic elements of the Quattro parallel robot and their velocities through the observed motion of the artificial pattern and by using the known kinematic models of the robot; (vi) then we were able to compute the motor torques through the presented generic dynamic modeling method; (vii) and we finally compared the measured ground-truth torques and the computed torques. Figure 9 shows these steps in a block diagram representation. The kinematic parameters and most of the dynamic parameters are taken from the robot data sheet and CAD files. The overall mass and the friction parameters in the motors were identified experimentally. Figures 10, 11 and 12 give the reference trajectory (red) and the estimated trajectory (blue dashed) by vision. The reference trajectory is an ellipse cycle which is repeated three times. The major and minor axes' lengths of the ellipse are $20\,cm$ and $5\,cm$, respectively. There is no rotational motion. On the trajectory, the maximum velocity of the end-effector is about $70\,cm/s$. Figure 13 superimposes the computed and measured torques versus time. Table III tabulates the normalized root mean square errors (NRMSE) of the computed torques with respect to the measured torques. NRMSE expresses the error (deviation) of the computed torques from the measured torques as a percentage. From Fig. 13 and Table III, we see that the computed torques are able to express well the dynamic behavior of the robot. The approximate 13% deviation from the measurements is obtained with the CAD values and a simple (i.e., manual) identification of friction. It thus qualitatively validates the proposed model (and the associated approach) for further implementation of the control. Quantitatively, we can expect to reduce the deviation to less than 10% using proper numerical identification. After that using our model for control should improve the performance of the robot, compared to the built-in controller (a simple PID) as suggested theoretically by [5] and experimentally by [7].

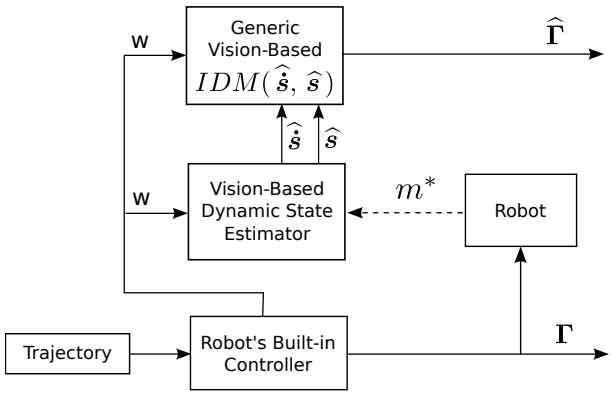

Fig. 9. Validation process of the vision-based generic dynamic model.

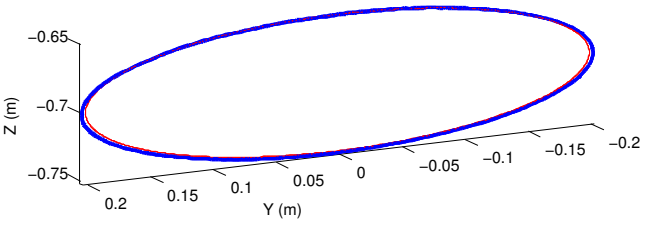

Fig. 10. Reference (red) and estimated (blue dashed) elliptic trajectory of the end-effector in 3D Cartesian space.

TABLE III

NRMSE OF COMPARED TORQUES OF MOTORS (%)

	motor 1	motor 2	motor 3	motor 4
NRMSE	13.39	11.93	12.97	11.38

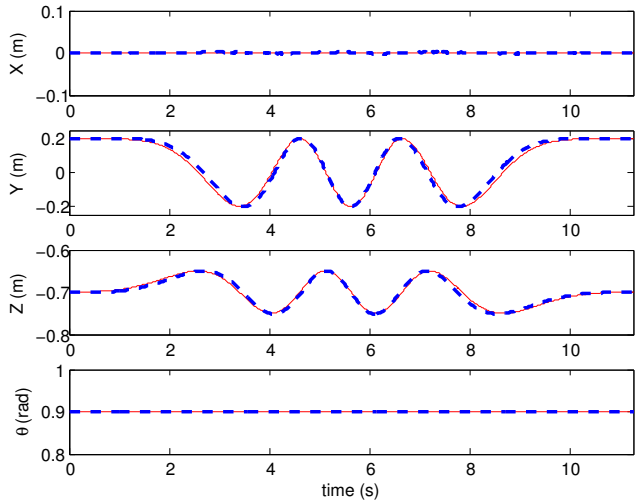

Fig. 11. Reference (red) and estimated (blue dashed) Cartesian end-effector poses of the Adept Quattro robot versus time.

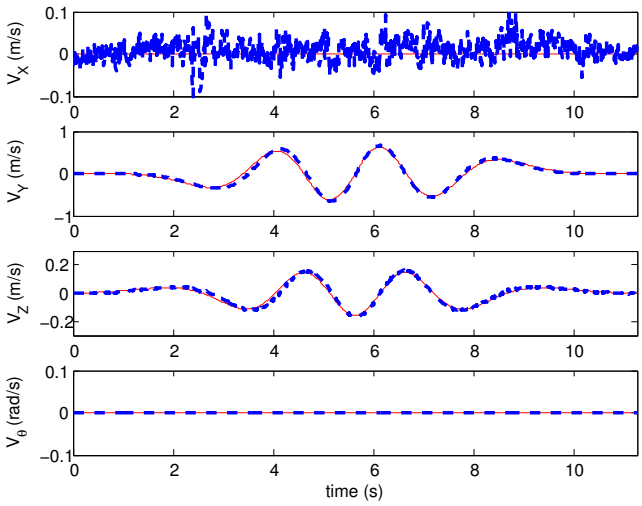

Fig. 12. Reference (red) and estimated (blue dashed) Cartesian end-effector velocities of the Adept Quattro robot versus time.

V. CONCLUSIONS

We presented a new vision-based generic dynamic modeling method for parallel kinematic manipulators, and we validated its correctness by experiments on the Adept Quattro parallel robot. We proposed a way to bring advantages of computer vision to dynamic modeling of PKMs. Our next step will be to put into practice the computed torque control of PKMs based on this vision-based generic dynamic model.

As a future work, we plan to use the computer vision method which is based on direct observation of the articulated kinematic chains of the robot [18] rather than observing a pattern attached to the end-effector of the robot. This eliminates the use of an artificial pattern and thus the need for pattern to end-effector calibration. Another future research path is to adopt the proposed modeling method for efficient kinematic and dynamic identification.

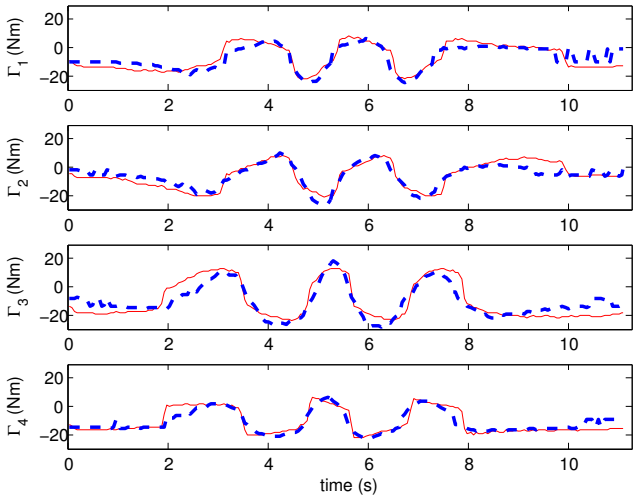

Fig. 13. Superimposed measured (red) and computed (blue dashed) torques versus time.

REFERENCES

[1] J.P. Merlet, "Parallel Robots", Dordrecht: Springer, 2009.

[2] B. Dasgupta, P. Choudhury, "A general strategy based on the newton-euler approach for the dynamic formulation of parallels manipulators", Mechanism and Machine Theory, vol. 34, pp. 801-824, 1999.

[3] W. Khalil, O. Ibrahim, "General solution for the dynamic modeling of parallel robots", In IEEE Int. Conf. on Robotics and Automation, New Orleans, LA, pp. 3665-3670, 2004.

[4] M. Callegari, M. Palpacelli, M. Principi, "Dynamics modelling and control of the 3-RCC translational platform", Mechatronics, vol. 16, pp. 589-605, 2006.

[5] F. Paccot, N. Andreff, P. Martinet, "A review on the dynamic control of parallel kinematic machines: theory and experiments", Int. J. Robotics Research, vol. 28, pp. 395-416, 2009.

[6] P. Dietmaier, "The StewartGough platform of general geometry can have 40 real postures", In Advances in Robot Kinematics: Analysis and Control. Dordrecht: Kluwer Academic Publishers, pp. 1-10, 1998.

[7] R. Dahmouche, N. Andreff, Y. Mezouar, O. Ait-Aider, P. Martinet, "Dynamic visual servoing from sequential regions of interest acquisition", Int. J. Robotics Research, vol. 31, no. 4, pp. 520-537, 2012.

[8] A. Vivas, P. Poignet, F. Pierrot, "Predictive functional control for a parallel robot", IEEE Int. Conf. on Intelligent Robots and Systems, pp. 2785-2790, October 2003.

[9] 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 2008.

[10] E. Ozgur, N. Andreff, P. Martinet, "Linear Dynamic Modeling of Parallel Kinematic Manipulators from Observable Kinematic Elements", Mechanism and Machine Theory, vol. 69 , pp. 73-89 , 2013.

[11] L. W. Tsai, "Robot analysis: The mechanics of serial and parallel manipulators", John Wiley & Sons, Inc., 1999.

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

[13] P. Mitiguy, T. R. Kane, "Motion variables leading to efficient equations of motion", Int. J. Robotics Research, pp. 522-532, October 1996.

[14] V. Nabat, M.O. Rodrigues, O. Company, S. Kurt, F. Pierrot "Par4: very high speed parallel robot for pick-and-place", IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Alberta, Canada, 2005.

[15] E. Ozgur, "From Lines to Dynamics of Parallel Robots", PhD Thesis, Université Blaise Pascal, Clermont-Ferrand, France, 2012.

[16] R. Dahmouche, N. Andreff, Y. Mezouar, P. Martinet, "3D Pose and Velocity Visual Tracking Based on Sequential Region of Interest Acquisition", IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, USA, 2009.

[17] E. Marchand, F. Chaumette, "Virtual visual servoing: a framework for real-time augmented reality", In G. Drettakis and H.-P. Seidel, editors, EUROGRAPHICS 2002 Conference Proceeding, vol. 21, no. 3, of Computer Graphics Forum, pp. 289-298, Saarebrcken, Germany, September 2002.

[18] E. Ozgur, R. Dahmouche, N. Andreff, P. Martinet, "High Speed Parallel Kinematic Manipulator State Estimation from Legs Observation", IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, Japan, 2013.