Parallel Computational Algorithms for the Kinematics and Dynamics of Planar and Spatial Parallel Manipulators

C. M. Gosselin
Département de Génie Mécanique,
Université Laval,
Cité Universitaire, Québec, Québec,
Canada, G1K 7P4

This paper introduces a novel approach for the computation of the inverse dynamics of parallel manipulators. It is shown that, for this type of manipulator, the inverse kinematics and the inverse dynamics procedures can be easily parallelized. The result is a closed-form efficient algorithm using n processors, where n is the number of kinematic chains connecting the base to the end-effector. The dynamics computations are based on the Newton-Euler formalism. The parallel algorithm arises from a judicious choice of the coordinate frames attached to each of the legs, which allows the exploitation of the parallel nature of the mechanism itself. Examples of the application of the algorithm to a planar three-degree-of-freedom parallel manipulator and to a spatial six-degree-of-freedom parallel manipulator are presented.

1 Introduction

High-performance control of robotic manipulators calls for numerical algorithms capable of solving the inverse kinematics and the inverse dynamics in real time. This constraint has motivated an important part of the research in the field of manipulator dynamics over the last decade. Indeed, ever since the early work on this subject has been carried out (Luh et al., 1980; Hollerbach, 1980), several researchers have tackled the problem of deriving efficient algorithms for the solution of the inverse dynamics problem of serial manipulators (see for instance Angeles and Ma, 1988; Balafoutis et al., 1988, and many others). Additionally, some authors have investigated the possibility of using parallel algorithms in order to share the computational load among several processors (see for instance Hashimoto and Kimura 1989; Hashimoto et al., 1990; Fijany and Bejczy, 1991). For serial-type manipulators, algorithms achieving a very high degree of parallelism have been proposed (see for instance Hashimoto and Kimura 1989; Hashimoto et al., 1990).

Meanwhile, from a completely different perspective, it has been proposed (MacCallion and Pham, 1979; Hunt, 1978; Fichter, 1986; Merlet, 1987; Gosselin, 1988) to use manipulators with a parallel mechanical architecture to provide an alternative to current serial-type robotic manipulators which have their own limitations. Potential applications of parallel manipulators arise whenever there is a need for large structural stiffness or high-performance dynamics and when it is desirable to bring the actuators as close as possible to the base. Current applications of parallel devices include flight simulators (Dieudonne et al., 1972) and robotic applications requiring force control (Reboulet and Robert, 1985; Merlet, 1988; Kim and Tesar, 1990) or high-speed motion (Clavel, 1988; Pierrot et al., 1991; Gosselin and Hamel, 1994). The dynamics of parallel manipulators has been studied by a few authors (Sugimoto, 1987; Do and Yang, 1988; Reboulet and Berthomieu, 1991; Nguyen et al., 1991; Geng and Haynes, 1992; Guglielmetti, 1994). A complete dynamical model of a six-degree-of-freedom parallel manipulator has been presented in (Ma, 1991).

Contributed by the Dynamic Systems and Control Division for publication in the JOURNAL OF DYNAMIC SYSTEMS, MEASUREMENT, AND CONTROL. Manuscript received by the DSCD July 7, 1994; revised manuscript received January 5, 1995. Associate Technical Editor: B. Siciliano.

In this paper, it is proposed to exploit the mechanical architecture of parallel manipulators to obtain parallel computational algorithms for the inverse kinematics and inverse dynamics problems. Indeed, because of the parallel mechanical architecture, exact parallel algorithms can be derived which do not require any iterative procedure. It is shown that this formulation leads directly to practical implementations with one processor for each of the subchains connecting the platform to the base of the manipulator. Examples of application of this novel approach to planar and spatial parallel manipulators are then presented.

2 Inverse Kinematic Problem

A spatial six-degree-of-freedom parallel manipulator is represented schematically in Fig. 1. It consists of a base $A_1 \ldots A_6$ and a platform $B_1 \ldots B_6$ which are connected via 6 legs or kinematic chains. Each of the legs is attached to the base through a Hooke or Cardan joint and to the platform by a spherical joint. Moreover, each of the legs comprises an actuated prismatic joint which controls the length of the leg. All the other joints are unactuated. Globally, the mechanism has six degrees of freedom which allows the positioning and orientation of the platform arbitrarily with respect to the base.

A reference frame $\mathcal{R}(O, X, Y, Z)$ is fixed to the base and a coordinate frame $\mathcal{R}'(O', X', Y', Z')$ is attached to the platform. Furthermore, the position of the joints on the base—points A_i—are denoted by vectors $\mathbf{a}_i$, $i = 1, \ldots, 6$ and the position of the joints on the platform—points B_i—by vectors $\mathbf{b}_i$, $i = 1, \ldots, 6$. Vectors $\mathbf{a}_i$ are constant when expressed in frame $\mathcal{R}$ while vectors $\mathbf{b}_i$ are constant when expressed in frame $\mathcal{R}'$. Finally, let vector $\mathbf{r} = [r_x, r_y, r_z]^T$ denote the position of point O' with respect to point O expressed in frame $\mathcal{R}$ and let $\mathbf{Q}$ be the matrix representing the rotation from frame $\mathcal{R}$ to frame $\mathcal{R}'$. From Fig. 1, one can write:

$$[\mathbf{b}_i]_R = [\mathbf{r}]_R + \mathbf{Q}[\mathbf{b}_i]_{R'}, \quad i = 1, \ldots, 6 \qquad (1)$$

where the index outside the square brackets indicates the reference frame in which a vector is expressed. Subtracting vector $\mathbf{a}_i$ from Eq. (1), one obtains

$$[\mathbf{b}_i - \mathbf{a}_i]_R = [\mathbf{r}]_R + \mathbf{Q}[\mathbf{b}_i]_{R'} - [\mathbf{a}_i]_R, \quad i = 1, \ldots, 6 \qquad (2)$$

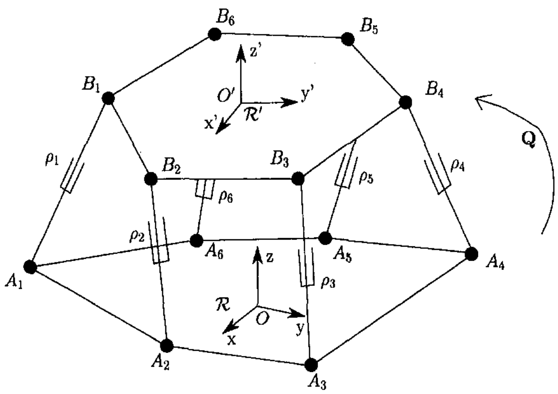

Fig. 1 Spatial six-degree-of-freedom parallel manipulator

where the left-hand side represents, in fact, a vector connecting point A_i to point B_i, along the ith leg. Hence, taking the Euclidean norm of both sides of this equation leads to:

$$\rho_i = \|[\mathbf{b}_i - \mathbf{a}_i]_R\|$$
$$= \|[\mathbf{r}]_R + \mathbf{Q}[\mathbf{b}_i]_{R'} - [\mathbf{a}_i]_R\|, \quad i = 1, \ldots, 6 \qquad (3)$$

Therefore, for a given position and orientation of the platform, the length of each of the legs is easily computed. This constitutes the solution of the inverse kinematics problem for the platform, which leads to a unique branch. This result has been obtained by many authors. However, it is pointed out here that each of the six equations obtained for the inverse kinematics are independent from one another and can be computed in parallel. This result is quite different from what is obtained for serial manipulators, where the joint coordinates have to be computed in sequence when a closed-form solution is used. This result will be exploited next.

2.1 Parallel Algorithm for the Inverse Kinematics. As mentioned above, the six equations obtained for the inverse kinematics—Eqs. (3)—are independent from one another. In other words, each of the joint coordinates can be computed directly from the original data, i.e., from the Cartesian coordinates of the platform. Indeed, Eqs. (3) can be expanded, which leads to

$$\rho_i = \sqrt{U_i^2 + V_i^2 + W_i^2}, \quad i = 1, \ldots, 6 \qquad (4)$$

where

$$U_i = r_x + q_{11}b_i^x + q_{12}b_i^y + q_{13}b_i^z - a_i^x \qquad (5)$$

$$V_i = r_y + q_{21}b_i^x + q_{22}b_i^y + q_{23}b_i^z - a_i^y \qquad (6)$$

$$W_i = r_z + q_{31}b_i^x + q_{32}b_i^y + q_{33}b_i^z - a_i^z \qquad (7)$$

in which q_{ij} denotes the ij component of matrix $\mathbf{Q}$, and in which the components of vectors $\mathbf{a}_i$ and $\mathbf{b}_i$ have been defined such that:

$$[\mathbf{a}_i]_R = [a_i^x, a_i^y, a_i^z]^T, \quad [\mathbf{b}_i]_{R'} = [b_i^x, b_i^y, b_i^z]^T, \quad i = 1, \ldots 6 \qquad (8)$$

where quantities $(a_i^x, a_i^y, a_i^z, b_i^x, b_i^y, b_i^z)$ are functions of the geometry of the base and the platform. From the above equations, it is clear that each of the joint coordinates can be computed directly from the geometry of the manipulator and the Cartesian coordinates. Hence, we can use one processor for each of the legs, i.e., for each of the joint coordinates. The idea of the parallelism in the computational scheme can also be extended to the computation of the Jacobian matrices. This is presented in the next section.

2.2 Parallel Algorithm for the Evaluation of the Jacobian Matrices. When Eqs. (3) are differentiated with respect to time, a set of linear equations relating the joint rates to the Cartesian velocities are obtained. Following the formalism proposed in (Gosselin and Angeles, 1990) for parallel manipulators, two Jacobian matrices $\mathbf{J}$ and $\mathbf{K}$ are obtained and the velocity equation is written as

$$\mathbf{J}\dot{\boldsymbol{\rho}} + \mathbf{K}\mathbf{t} = 0 \qquad (9)$$

where $\mathbf{t}$ is the six-dimensional twist of the platform and $\dot{\boldsymbol{\rho}}$ is the vector of joint velocities. These vectors are defined as

$$\mathbf{t} = [\boldsymbol{\omega}^T, \dot{\mathbf{r}}^T]^T, \quad \dot{\boldsymbol{\rho}} = [\dot{\rho}_1, \ldots, \dot{\rho}_6]^T \qquad (10)$$

in which $\boldsymbol{\omega}$ is the angular velbcity of the platform and $\dot{\mathbf{r}}$ is the linear velocity of point O'. From (Gosselin and Angeles, 1990), the aforementioned Jacobian matrices can be written as:

$$\mathbf{J} = \mathrm{diag}(\rho_1, \ldots, \rho_6) \qquad (11)$$

and

$$\mathbf{K} = \begin{bmatrix} \mathbf{k}_1^T \\ \mathbf{k}_2^T \\ \vdots \\ \mathbf{k}_6^T \end{bmatrix} \qquad (12)$$

with

$$\mathbf{k}_i = [\{(\mathbf{Q}[\mathbf{b}_i]_{R'}) \times \mathbf{s}_i\}^T, \mathbf{s}_i^T]^T, \quad i = 1, \ldots, 6 \qquad (13)$$

where $\mathbf{s}_i$ is the vector connecting point A_i to point B_i, i.e.,

$$\mathbf{s}_i = [U_i, V_i, W_i]^T, \quad i = 1, \ldots, 6 \qquad (14)$$

Matrix $\mathbf{J}$ is readily available from the solution of the inverse kinematic problem since its non-vanishing components are the joint coordinates themselves. Moreover, it is clear from the expressions given above that each of the rows of matrix $\mathbf{K}$ can be computed independently, just as in the case of the inverse kinematics. Quantities U_i, V_i and W_i, which have already been evaluated, are used again here.

If the acceleration inversion is also required, Eq. (9) is differentiated with respect to time. This leads to

$$\mathbf{J}\ddot{\boldsymbol{\rho}} + \dot{\mathbf{J}}\dot{\boldsymbol{\rho}} + \mathbf{K}\dot{\mathbf{t}} + \dot{\mathbf{K}}\mathbf{t} = \mathbf{0} \qquad (15)$$

where $\dot{\mathbf{t}}$ is the generalized acceleration of the platform and $\ddot{\boldsymbol{\rho}}$ is the vector of joint accelerations. These vectors are defined as

$$\dot{\mathbf{t}} = [\dot{\boldsymbol{\omega}}^T, \ddot{\mathbf{r}}^T]^T, \quad \ddot{\boldsymbol{\rho}} = [\ddot{\rho}_1, \ldots, \ddot{\rho}_6]^T \qquad (16)$$

in which $\dot{\boldsymbol{\omega}}$ is the angular acceleration of the platform and $\ddot{\mathbf{r}}$ is the acceleration of point O'. In Eq. (15), the derivatives of matrices $\mathbf{J}$ and $\mathbf{K}$ need to be computed. They are given as

$$\dot{\mathbf{J}} = \mathrm{diag}(\dot{\rho}_1, \ldots, \dot{\rho}_6) \qquad (17)$$

and

$$\dot{\mathbf{K}} = \begin{bmatrix} \dot{\mathbf{k}}_1^T \\ \dot{\mathbf{k}}_2^T \\ \vdots \\ \dot{\mathbf{k}}_6^T \end{bmatrix} \qquad (18)$$

with

$$\dot{\mathbf{k}}_i = [\{(\dot{\mathbf{Q}}[\mathbf{b}_i]_{R'}) \times \mathbf{s}_i + (\mathbf{Q}[\mathbf{b}_i]_{R'}) \times \dot{\mathbf{s}}_i\}^T, \dot{\mathbf{s}}_i^T]^T \qquad (19)$$

where $\dot{\mathbf{s}}_i$ and $\dot{\mathbf{Q}}$ are the time derivatives of vector $\mathbf{s}_i$ and matrix $\mathbf{Q}$, respectively. Matrix $\dot{\mathbf{J}}$ is readily obtained from the velocity inversion. Moreover, the computation of matrix $\dot{\mathbf{K}}$ is easily parallelized since each of the rows is related to the corresponding leg and can be computed independently from the other ones. The resulting parallel computational scheme—including the solution of the inverse kinematics and the computation of the Jacobian matrices and their time derivatives—is represented schematically in Fig. 2. One processor can be used for each of the legs. Because of the mechanical architecture of the manipulator, the nature of the problem is completely parallel. Therefore, each of the six processors is assigned exactly one sixth of the total amount of computations which would have to be performed in a serial implementation.

3 Inverse Dynamics

The solution of the inverse dynamics problem consists in computing the actuator forces or torques required to perform a prescribed motion of the manipulator. It requires knowledge of the mass and inertial properties of each of the moving links of the manipulator.

For serial manipulators, once the motion of each of the links is determined, the recursive application of the Newton-Euler equations on each of the links from the end-effector to the base leads to an efficient serial algorithm (Luh et al., 1980). The serial nature of the algorithm obtained is directly related to the serial architecture of the kinematic chain involved. However, more detailed analyses lead to parallel algorithms whose complexity is of the order of log (n) (Hashimoto and Kimura, 1989) where n is the number of degrees of freedom of the serial manipulator.

For parallel manipulators, the situation is quite different since the end-effector—the platform—is not the end link of a kinematic chain. However, as opposed to the results obtained above for the inverse kinematics, it is not obvious from the outset that the parallel structure of the mechanism can be exploited in the derivation of parallel algorithms for the inverse dynamics.

In order to formulate the problem systematically, the free-body diagrams of the platform and of the two parts of one of the legs, which are shown in Fig. 3, are first considered. In Fig. $3(a)$, the platform is represented with all the forces acting on it. Point G_P denotes the mass center of the platform. Since each of the legs is attached to the platform via a spherical joint, no torque can be applied to the platform by the legs. Moreover, for reasons that will become clearer in the next subsection, each of the contact forces between the legs and the platform is decomposed as one force acting in the direction of the leg, noted $\mathbf{f}_i$ and one force acting in a plane orthogonal to the direction of the leg, noted $\mathbf{g}_i$ (Reboulet and Berthomieu, 1991). Since the direction of the leg is known from the kinematic analysis, force $\mathbf{f}_i$ contains only one unknown—its magnitude—and force $\mathbf{g}_i$ contains two unknowns. Hence, there are 18 unknowns in the free-body diagram of Fig. $3(a)$ which means that this diagram cannot be solved directly. In Fig. $3(b)$, the free-body diagram of the upper part of the ith leg is represented. Point G_{i1} denotes the mass center of this link. Forces $\mathbf{f}_i$ and $\mathbf{g}_i$ are still present, in virtue of Newton's third law. Moreover, the forces and moments applied to the upper part of the leg by the lower part of the leg are denoted as follows: force $\mathbf{d}_i$ is the force acting in a plane orthogonal to the leg, force $\mathbf{h}_i$ is the force acting in the direction of the leg—the force applied by the actuator—moment $\mathbf{n}_i$ is the moment acting in a plane orthogonal to the leg and moment $\mathbf{o}_i$ is the moment acting in the direction of the leg. Again, the force and moment acting in the plane orthogonal to the leg contain two unknowns while the force and moment acting in the direction of the leg contain only one unknown, i.e., their magnitude. Finally, the free-body diagram of the lower part of the leg is represented in Fig. $3(c)$. Point G_{i2} is the mass center of this link. Since the link is connected to the base via a Hooke joint, only a moment in the direction of the leg can be applied by the base link to the lower part of the leg. This moment is denoted $\mathbf{l}_i$. Furthermore, the force applied by the base to the lower part of the leg is decomposed into two forces: force $\mathbf{c}_i$ is the force acting in the direction of the leg and force $\mathbf{e}_i$ is the force acting in a plane orthogonal to the direction of the leg.

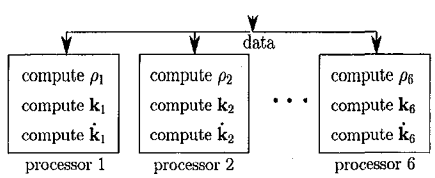

Fig. 2 Parallel computational algorithm for the inverse kinematics, velocity, and acceleration inversion

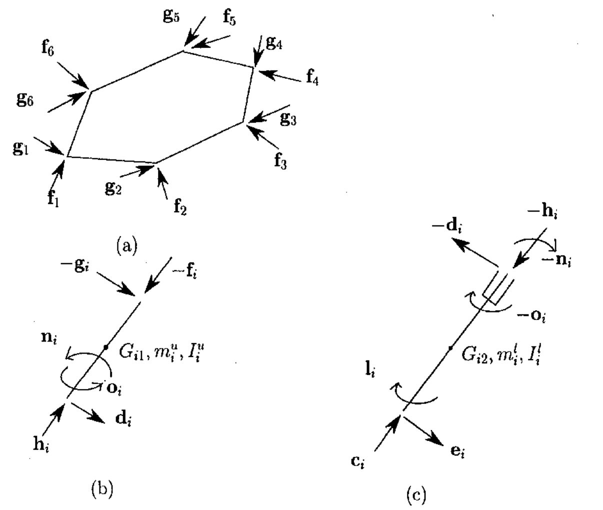

Fig. 3 Free-body diagram of the platform and of the two parts of one of the legs

Hence, the inverse dynamics problem can be formulated globally using 13 free-body diagrams involving a total of 78 scalar unknowns which are nothing but the components of the forces and moments defined above. Six of these components are the actuator forces and they constitute the output of the inverse dynamics procedure. Applying Newton and Euler's equations to each of the free-body diagrams, 78 linear equations are obtained, which is consistent with the fact that the problem at hand is dynamically determined. However, the system of equations obtained is not completely coupled since only a subset of the unknown variables appear in each of the equations. In fact, the maximum number of unknowns appearing in one same equation is 18, which occurs for the equations involving the free-body diagram of the platform. This suggests that the solution of the problem could probably be broken down into a set of simpler problems. Moreover, if it is possible to obtain a set of decoupled problems, then a parallel algorithm could be derived. This is discussed in the next subsection.

3.1 Parallel Algorithm for the Inverse Dynamics. As mentioned above, the parallelization of the inverse dynamics procedure is not as straightforward as in the case of the inverse kinematics. Indeed, the free-body diagram of the platform cannot be solved directly from the outset. However, a closer look at the free-body diagrams of Fig. $3(b)$ and (c) reveals that some of the unknowns appearing on these diagrams can be solved for directly and independently from the other legs if the unknown forces and torques are judiciously decomposed. This decomposition is the one presented above, i.e., each of the unknown forces and torques associated with one particular leg is decomposed into a vector in the direction of the leg and a vector acting in a plane orthogonal to the leg. The application of the Newton-Euler equations in the plane orthogonal to the direction of one given leg to both the upper and the lower part of the leg then leads to a system of 8 equations in 8 scalar unknowns which can be solved independently from the other legs. These equations are obtained as follows: first, the free-body diagram of the upper part of the leg is considered. The forces and moments are summed in the plane orthogonal to the direction of the leg, which leads to

$$\mathbf{d}_i - \mathbf{g}_i = m_i^u \mathbf{a}_i^{u\perp} \quad (20)$$

and

$$\mathbf{r}_{i1} \times \mathbf{d}_i - \mathbf{r}_{i2} \times \mathbf{g}_i + \mathbf{n}_i = \mathbf{I}_i^u \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \mathbf{I}_i^u \boldsymbol{\omega}_i \quad (21)$$

where $\mathbf{a}_i^{u\perp}$ is the acceleration of the center of mass of the link in the plane orthogonal to the direction of the link, m_i^u is the mass of the upper part of the leg, $\mathbf{I}_i^u$ is the inertia tensor of this link, $\mathbf{r}_{i1}$ is the vector connecting point G_{i1}—the center of the mass of the link—to the point of application of $\mathbf{d}_i$, $\mathbf{r}_{i2}$ is the vector connecting point G_{i1} to the point of application of $\mathbf{g}_i$ and $\boldsymbol{\omega}_i$ and $\dot{\boldsymbol{\omega}}_i$ are the angular velocity and acceleration of the link, respectively. These last two vectors, as well as vector $\mathbf{a}_i^{u\perp}$, are obtained from the kinematic analysis. The same equations are applied to the lower part of the leg. This leads to

$$-\mathbf{d}_i + \mathbf{e}_i = m_i^l \mathbf{a}_i^{l\perp} \quad (22)$$

and

$$-\mathbf{r}_{i3} \times \mathbf{d}_i + \mathbf{r}_{i4} \times \mathbf{e}_i - \mathbf{n}_i = \mathbf{I}_i^l \dot{\boldsymbol{\omega}}_i + \boldsymbol{\omega}_i \times \mathbf{I}_i^l \boldsymbol{\omega}_i \quad (23)$$

where $\mathbf{a}_i^{l\perp}$ is the acceleration of the center of mass of the link in the plane orthogonal to the direction of the link, m_i^l is the mass of the lower part of the leg, $\mathbf{I}_i^l$ is the inertia tensor of this link $\mathbf{r}_{i3}$ is the vector connecting point G_{i2}—the center of mass of the link—to the point of application of $\mathbf{d}_i$, $\mathbf{r}_{i4}$ is the vector connecting point G_{i2} to the point of application of $\mathbf{e}_i$ and $\boldsymbol{\omega}_i$ and $\dot{\boldsymbol{\omega}}_i$ are the angular velocity and acceleration of the link, respectively. These last two vectors, as well as vector $\mathbf{a}_i^{l\perp}$, are obtained from the kinematic analysis.

It becomes quite clear now that Eqs. (20)–(23) form a system of 8 scalar equations in 8 unknowns, the unknowns being the components of $\mathbf{g}_i$, $\mathbf{d}_i$, $\mathbf{e}_i$, and $\mathbf{n}_i$. Moreover, this system can readily be solved, regardless of the other legs. The solution of this system leads to the determination of the components of vector $\mathbf{g}_i$. Hence, when this operation is performed for each of the legs—in parallel—only 6 unknown quantities will remain on the free-body diagram of the platform, i.e., the forces in the direction of each of the legs. These can be solved for using the Newton-Euler equations on the platform. Finally, Newton's equation is applied to each of the upper parts of the legs, in the direction of the leg. This allows the computation of $\mathbf{h}_i$, the actuator force, and thereby completes the procedure.

The solution of the inverse dynamics derived above is almost completely parallel. Only one of the steps—the application of the Newton-Euler equations to the platform—must be performed on one single processor. In fact, this part of the algorithm, which amounts to the inversion of a linear system of algebraic equations, could also be parallelized. However, for the sake of simplicity and to minimize the communication between the processors, it is executed on a single processor here. The algorithm is represented schematically in Fig. 4. As in the case of the inverse kinematics, one processor is associated with each of the legs which leads to a very important improvement in efficiency. This will be clearly demonstrated in the next section where the application of the parallel algorithms derived above to two examples of parallel manipulators is discussed.

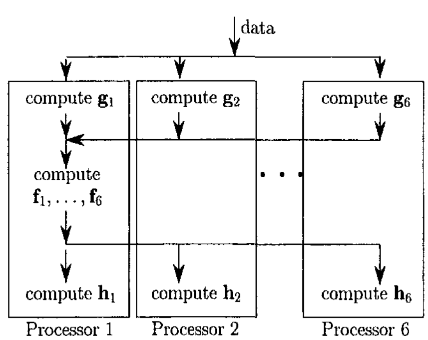

Fig. 4 Parallel computational algorithm for the inverse dynamics

4 Examples

4.1 Planar Three-Degree-of-Freedom Parallel Manipulator. First, a simple example presenting the inverse dynamics of a planar three-degree-of-freedom parallel manipulator is treated. This manipulator has been studied in (Gosselin and Angeles, 1988) and a serial algorithm for the inverse dynamics has been proposed in (Angeles and Ben-Zvi, 1988). The manipulator is represented in Fig. 5. Three prismatic joints are used to position and orient the triangular platform on the plane. All the revolute joints are unactuated. The free-body diagrams of the platform and of the two parts of one of the legs are shown in Fig. 6.

The inverse kinematics of the manipulator is first studied. If the position and the orientation of the platform are known, the positions of points B_i, $i = 1, 2, 3$ are readily computed. The position of point C—the reference point on the platform—will be noted $C(x, y)$ and the orientation of the platform with respect to a fixed reference will be given by angle ϕ. The inverse kinematics can then be written as

$$\rho_i = \sqrt{U_i^2 + V_i^2}, \quad i = 1, 2, 3 \quad (24)$$

with

$$U_i = x + b_i^x \cos\phi - b_i^y \sin\phi - a_i^x \quad (25)$$

$$V_i = y + b_i^x \sin\phi + b_i^y \cos\phi - a_i^y \quad (26)$$

where (b_i^x, b_i^y) are the components of the position vector of point B_i in a frame attached to the platform, (a_i^x, a_i^y) are the components of the position vector of point A_i in a frame attached to the base and ρ_i is the length of the leg, i.e., the ith joint coordinate. Equation (24) can be evaluated independently for each of the legs, thereby leading to a parallel algorithm. Each of the three processors involved will have to perform 6 multiplications, 7 additions, one square root, one sine and one cosine. The computational load is equally distributed among the processors.

The velocity equation can be written as

$$\mathbf{J}\dot{\boldsymbol{\rho}} + \mathbf{K}\dot{\mathbf{r}} = \mathbf{0} \quad (27)$$

where $\dot{\mathbf{r}} = [\dot{x}, \dot{y}, \dot{\phi}]^T$ is the Cartesian velocity vector of the platform and $\dot{\boldsymbol{\rho}}$ is the vector of joint velocities. Matrix $\mathbf{J}$ is a diagonal matrix whose nonzero elements are the joint coordinates and hence requires no computations. On the other hand, the ith row of matrix $\mathbf{K}$, noted $\mathbf{k}_i$, can be written as

$$\mathbf{k}_i = [U_i, V_i, W_i], \quad i = 1, 2, 3 \quad (28)$$

with

$$W_i = -U_i(b_i^x \sin\phi + b_i^y \cos\phi) + V_i(b_i^x \cos\phi - b_i^y \sin\phi) \quad (29)$$

Hence, the computation of each row of the Jacobian matrix requires 6 multiplications and 3 additions to be performed in parallel by each of the 3 processors. The acceleration inversion is similar and is not discussed here for the sake of conciseness.

The inverse dynamics procedure is similar to the general algorithm presented in Section 3.1. A system of 4 scalar equations in 4 unknowns—the forces orthogonal to the direction of the leg and the torque at the prismatic joint—can be written

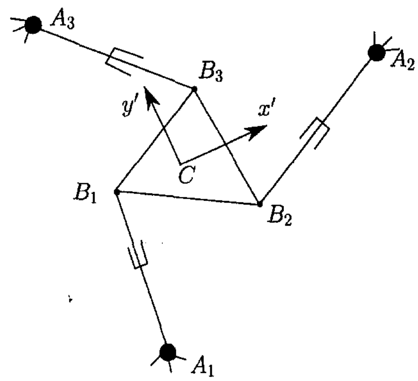

Fig. 5 Planar three-degree-of-freedom parallel manipulator

for each of the legs. Moreover, this system can be solved in closed-form and its solution for the force at the revolute joint connecting the leg to the platform takes on the following form

$$d_i = \frac{(I_l + I_u)\dot{\omega}_i + m_u l_u(\mathbf{a}_{iu})^{\perp} - m_l l_l(\mathbf{a}_{il})^{\perp}}{\rho_i} \quad (30)$$

$$g_i = d_i - m_u(\mathbf{a}_{iu})^{\perp} \quad (31)$$

where d_i is the component of the interaction force between the upper and lower parts of the ith leg in a direction orthogonal to the leg, g_i is the interaction force between the platform and the upper part of the ith leg in a direction orthogonal to the leg, I_l and I_u are the moments of inertia of the lower and upper part of the leg, $\dot{\omega}_i$ is the angular acceleration of the leg, m_u is the mass of the upper part of the leg, m_l is the mass of the lower part of the leg, $(\mathbf{a}_{iu})^{\perp}$ is the component of the acceleration of the mass center of the upper part of the leg in a direction orthogonal to the leg, $(\mathbf{a}_{il})^{\perp}$ is the component of the acceleration of the mass center of the lower part of the leg in a direction orthogonal to the leg, l_u is the distance from the mass center of the upper part of the leg and the revolute joint attached to the platform, and l_l is the distance between the mass center of the lower part of the leg and the revolute joint attached to the base. The above computations are performed in parallel by each of the three processors and they require only 7 multiplications and 4 additions (assuming that the accelerations of the mass centers and the angular accelerations are available).

Then, the 3 Newton-Euler equations are applied to the platform and a 3×3 linear system of equation is obtained in terms of the components of the forces applied by the legs to the platform in the direction of the legs. In general, this system does not contain any zero and must be solved with a general procedure. As in the case of the spatial manipulator, this is the only non-parallel part of the algorithm. One of the processors can be assigned this task, which will require 11 multiplications and 6 additions.

Finally, from the forces computed above, the joint force, h_i can be computed as follows

$$h_i = m_u(\mathbf{a}_{iu})^{\parallel} - f_i \quad (32)$$

where $(\mathbf{a}_{iu})^{\parallel}$ is the acceleration of the mass center of the upper part of the leg in the direction of the leg and f_i is the force applied by the upper part of the leg to the platform, in the direction of the leg. This last operation is performed in parallel, for each of the legs and it requires only one multiplication and one addition.

Globally, the time required to compute the inverse dynamics of the planar three-degree-of-freedom parallel manipulator can then be reduced to the time required to perform 19 multiplications and 11 additions on one processor. A serial implementation would yield 35 multiplications and 21 additions. It is recalled that this does not include some kinematic computations that would be necessary to obtain the acceleration of the mass center of each of the links. These computations, which are required in both the serial and the parallel implementation, can also be easily parallelized. This would increase the gain in computation time provided by the parallel algorithm.

4.2 Spatial Six-Degree-of-Freedom Parallel Manipulator. The computational complexity of the algorithm will now be discussed for the general case of a spatial six-degree-of-freedom parallel manipulator such as the one described in Sections 2 and 3 and represented in Fig. 1.

4.2.1 Inverse Kinematics. The inverse kinematics is parallelized using the procedure presented in Section 2.1. The computation of the joint variables is performed according to Eq. (4). This requires 12 multiplications, 14 additions and one square root per processor. Then, the evaluation of each row of the Jacobian matrix requires 18 multiplications and 9 additions, considering that some of the quantities needed have already been computed in the inverse kinematics procedure. Finally, the computation of one row of the derivative of the Jacobian matrix requires 48 multiplications and 24 additions. In total, the kinematic procedure will require 78 multiplications, 47 additions and one square root, 6 times less than in a serial implementation since the computational load is equally distributed among the processors.

4.2.2 Inverse Dynamics. The first part of the inverse dynamics algorithm consists in the solution of Eqs. (20–23) which, it is recalled, form a system of 8 linear scalar equations in 8 unknowns, the unknowns being the components of $\mathbf{g}_i$, $\mathbf{d}_i$, $\mathbf{e}_i$ and $\mathbf{n}_i$ (there are only 2 unknowns per vector since we know the plane in which these vectors act). However, this system is not completely general in the sense that not all the unknowns appear in each of the equations. Therefore, the solution can be simplified substantially. First of all, Eqs. (21) and (23) are summed, which leads to the elimination of vector $\mathbf{n}_i$. Then, an expression for $\mathbf{e}_i$ is obtained from Eq. (22) and is substituted in the equation resulting from the previous operation. This leads to the following

$$P_i + Q_i = R_2 + R_4 \quad (33)$$

with

$$P_i = \mathbf{r}_{i1} \times \mathbf{d}_i - \mathbf{r}_{i2} \times \mathbf{g}_i \quad (34)$$

$$Q_i = -\mathbf{r}_{i3} \times \mathbf{d}_i + \mathbf{r}_{i4} \times (\mathbf{d}_i + m_i^l \mathbf{a}_i^{l\perp}) \quad (35)$$

and where R_2 and R_4 stand for the right-hand side of Eqs. (21) and (23), respectively. Finally, an expression for vector $\mathbf{d}_i$ is obtained from Eq. (20) and substituted in Eq. (33) which leads to an equation in $\mathbf{g}_i$ only, i.e.,

$$(\mathbf{r}_{i1} - \mathbf{r}_{i2} - \mathbf{r}_{i3} + \mathbf{r}_{i4}) \times \mathbf{g}_i = Z \quad (36)$$

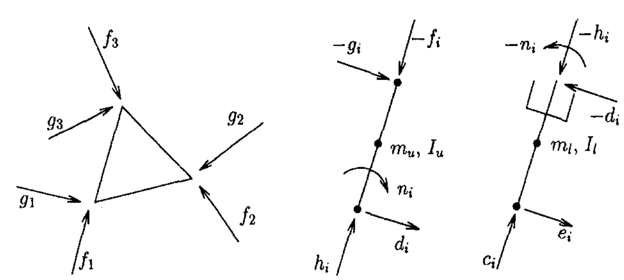

Fig. 6 Free-body diagram of the platform and of the two parts of one of the legs of the planar three-degree-of-freedom parallel manipulator

with

$$Z = R_2 + R_4 - \mathbf{r}_{i1} \times m_i^u \mathbf{a}_i^{u\perp} + \mathbf{r}_{i3} \times m_i^u \mathbf{a}_i^{u\perp} - \mathbf{r}_{i4} \times (m_i^u \mathbf{a}_i^{u\perp} + m_i^l \mathbf{a}_i^{l\perp}) \quad (37)$$

Equation (36) can be solved for the components of $\mathbf{g}_i$, the force applied by leg i to the platform in the plane orthogonal to the direction of the leg. It is pointed out that Eq. (36) could also have been obtained from the application of the Euler equation to the whole leg, considering a fixed point, i.e., the lower end of the leg.

If the vectors are expressed in a local frame, with one of its axes in the direction of the leg, then the computations will be simplified. However, this entails the transformation of all the velocities and accelerations in this frame. Such a transformation requires 9 multiplications and 6 additions and must be performed for vectors $\boldsymbol{\omega}$, $\dot{\boldsymbol{\omega}}$, $\mathbf{a}_i^u$, and $\mathbf{a}_i^l$. Then, Eq. (36) can be used directly to compute the two non-vanishing components of $\mathbf{g}_i$ in the local frame. This requires a total of 64 multiplications and 39 additions. Hence, the first part of the procedure, which is executed in parallel by each of the 6 processors requires a total of 100 multiplications and 63 additions per processor.

As mentioned in Section 3.1, the second part of the algorithm cannot be directly parallelized and must be assigned to one of the processors. It consists in the application of the Newton-Euler equations to the platform which leads to a system of 6 linear equations in 6 unknowns, i.e., the components of the forces applied to the platform by each of the legs in the direction of the leg. In general, this system of equations will not contain any zero. Therefore, its solution will require approximately 120 multiplications and 110 additions, including the preparation of the right hand side of the linear system and the fact that the forces $\mathbf{g}_i$, $i = 1, \ldots, 6$, must be transformed into the reference frame attached to the platform.

Finally, the last part of the procedure can be computed in parallel on each of the six processors. This allows the computation of the actuator force $\mathbf{h}_i$ as

$$\mathbf{h}_i = \mathbf{f}_i + m_i^u \mathbf{a}_i^{u\parallel} \quad (38)$$

In the above equation, vectors $\mathbf{f}_i$ and $\mathbf{a}_i^{u\parallel}$ must be transformed into the reference frame attached to the leg. The equation becomes then scalar since 2 of the 3 components of each of the vectors will be equal to zero. This last step will therefore require 19 multiplications and 13 additions per processor.

Hence, the time to complete the inverse dynamics procedure, assuming that the kinematic computations have been performed, is equivalent to the time required by 240 multiplications and 190 additions on one single processor. The same algorithm implemented on a single processor would require 834 multiplications and 566 additions, whereas the direct solution of the original system of 78 equations in 78 unknowns would require approximately 150000 multiplications and additions. The gain is, therefore, substantial. Moreover, the algorithm derived here—regardless of the fact that it can be parallelized—leads to a much more stable procedure since the largest linear system of simultaneous equations encountered is a system of 6 equations in 6 unknowns.

In a real implementation, the communication overhead introduced by the fact that several processors are used would have to be considered. Experimental results illustrating this effect are given in (Guglielmetti 1994). However, since only 6 processors are used here (for the six-degree-of-freedom manipulator) and that they can execute exactly the same code, it is expected that this overhead would be limited.

5 Conclusion

The inverse kinematics and the inverse dynamics of parallel manipulators have been studied in this paper. The objective was to derive parallel algorithms which would make use of several processors in order to improve the numerical efficiency. It has been shown that, for this type of manipulator, the inverse kinematics and the inverse dynamics procedures can be easily parallelized. The parallelization of the inverse kinematics and of the computation of the Jacobian is straightforward while the key to the parallelization of the dynamics was the definition of judicious local reference frames. An efficient closed-form algorithm using n processors, where n is the number of kinematic chains connecting the base to the end-effector, has been presented. Examples of the application of the algorithm to a planar three-degree-of-freedom manipulator and to a spatial six-degree-of-freedom parallel manipulators have been presented. In each case, the algorithm leads to a very significant improvement of the computational efficiency and of the robustness.

6 Acknowledgments

This work was completed under research grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) and from the Fonds pour la Formation des Chercheurs et l'Aide à la Recherche (FCAR) du Québec. The author would also like to thank Mr. Pierre Billot and Mr. Michel Lambert for their help in the implementation of the algorithms.

References

Angeles, J., and Ben-Zvi, A., 1988, "Dynamic Modeling of a Planar Three-Degree-of-Freedom Parallel Manipulator," *Proceedings of the 2nd International Symposium on Robotics and Manufacturing,* Albuquerque, pp. 341–349.

Angeles, J., and Ma, O., 1988, "Dynamic Simulation of n-axis Serial Robotic Manipulators Using a Natural Orthogonal Complement," *The International Journal of Robotics Research,* Vol. 7, No. 5, pp. 32–47.

Balafoutis, C. A., Misra, P., and Patel, R. V., 1988, "A Cartesian Tensor Approach for Fast Computation of Manipulator Dynamics," *Proceedings of the IEEE International Conference on Robotics and Automation,* Philadelphia, Vol. 3, pp. 1348–1353.

Clavel, R., 1988, "DELTA, a Fast Robot with Parallel Geometry," *Proc. of the Int. Symp. on Industrial Rob.,* Lausanne, Switzerland, pp. 91–100.

Dieudonne, J. E., Parrish, R. V., and Bardusch, R. E., 1972, "An Actuator Extension Transformation for a Motion Simulator and an Inverse Transformation Applying Newton-Raphson's method," NASA Technical Report TN D-7067.

Do, W. Q. D., and Yang, D. C. H., 1988, "Inverse Dynamics and Simulation of a Platform Type of Robot," *Journal of Robotic Systems,* Vol. 5, No. 3, pp. 209–227.

Fichter, E. F., 1986, "A Stewart Platform-Based Manipulator: General Theory and Practical Construction," *The International Journal of Robotics Research,* Vol. 5, No. 2, pp. 157–182.

Fijany, A., and Bejczy, A. K., 1991, "Parallel Computation of Manipulator Inverse Dynamics," *Journal of Robotic Systems,* Vol. 8, No. 5, pp. 599–635.

Geng, Z., and Haynes, L. S., 1992, "On the Dynamic Model and Kinematic Analysis of a Class of Stewart Platforms," *Journal of Robotics and Autonomous Systems,* Vol. 9, pp. 237–254.

Gosselin, C., 1988, "Kinematic Analysis, Optimization and Programming of Parallel Robotic Manipulators," Ph.D. thesis, Dept. of Mechanical Engineering, McGill University, Montréal.

Gosselin, C., and Angeles, J., 1988, "The Optimum Kinematic Design of a Planar Three-Degree-of-Freedom Parallel Manipulator," ASME *Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 110, No. 1, pp. 35–41.

Gosselin, C., and Angeles, J., 1990, "Singularity Analysis of Closed-Loop Kinematic Chains," *IEEE Transactions on Robotics and Automation,* Vol. 6, No. 3, pp. 281–290.

Gosselin, C. M., and Hamel, J.-F., 1994, "The Agile Eye: A High-Performance Three-Degree-of-Freedom Camera-Orienting Device," *Proceedings of the IEEE International Conference on Robotics and Automation,* San Diego, Mai, pp. 781–786.

Guglielmetti, P., 1994, "Model-Based Control of Fast Parallel Robots: A Global Approach in Operational Space," Ph.D. thesis, EPFL, Lausanne.

Hashimoto, K., and Kimura, H., 1989, "A New Parallel Algorithm for Inverse Dynamics," *The International Journal of Robotics Research,* Vol. 8, No. 1, pp. 63–76.

Hashimoto, K., Ohashi, K., and Kimura, H., 1990, "An Implementation of a Parallel Algorithm for Real-Time Model-Based Control on a Network of Microprocessors," *The International Journal of Robotics Research,* Vol. 9, No. 6, pp. 37–47.

Hollerbach, J. M., 1980, "A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity," *IEEE Transactions on Systems, Man and Cybernetics,* Vol. SMC-10, No. 11, pp. 730–736.

Hunt, K. H., 1978, *Kinematic Geometry of Mechanisms,* Clarendon Press.

Kim, W. K., and Tesar, D., 1990, "Study on Structural Design of Force-

Reflecting Manual Controllers," *Proceedings of the ASME Mechanisms Conference,* Chicago, Vol. 24, pp. 481–488.

Luh, J. Y. S., Walker, M. W., and Paul, R. P. C., 1980, "On-Line Computational Scheme for Mechanical Manipulators," ASME JOURNAL OF DYNAMIC SYSTEMS, MEASUREMENTS, AND CONTROL, Vol. 102, pp. 103–110.

Ma, O., 1991, "Mechanical Analysis of Parallel Manipulators with Simulation, Design and Control Applications," Ph.D. thesis, McGill University, June.

MacCallion, H., and Pham, D. T., 1979, "The Analysis of a Six-Degree-of-Freedom Work Station for Mechanised Assembly," *Proceedings of the 5th World Congress on the Theory of Machines and Mechanisms,* Montréal.

Merlet, J.-P., 1987, "Parallel Manipulators, Part I: Theory, Design, Kinematics, Dynamics and Control," Technical Report #646 INRIA, France.

Merlet, J.-P., 1988, "Force-Feedback Control of Parallel Manipulators," *Proceedings of the IEEE International Conference on Robotics and Automation,* Philadelphia, Vol. 3, pp. 1484–1489.

Nguyen, C. C., Antrazi, S. S., Zhou, Z. L., and Campbell, C. E., 1991, "Experimental Study of Motion Control and Trajectory Planning for a Stewart Platform Robot Manipulator," *Proceedings of the IEEE International Conference on Robotics and Automation,* Sacramento, Vol. 2, pp. 1873–1878.

Pierrot, F., Dauchez, P., and Fournier, A., 1991, "Hexa: A Fast Six-Degree-of-Freedom Fully Parallel Robot," *Proceedings of the Fifth International Conference on Advanced Robotics,* Pisa, Vol. 2, pp. 1158–1163.

Reboulet, C., and Berthomieu, T., 1991, "Dynamic Models of a Six-Degree-of-Freedom Parallel Manipulator," *Proceedings of the Fifth International Conference on Advanced Robotics,* Pisa, Vol. 2, pp. 1153–1157.

Reboulet, C., and Robert, A., 1985, "Hybrid Control of a Manipulator with an Active Compliant Wrist," 3rd ISRR, Gouvieux, France, pp. 76–80.

Sugimoto, K., 1987, "Kinematic and Dynamic Analysis of Parallel Manipulators by Means of Motor Algebra," ASME *Journal of Mechanisms, Transmissions and Automation in Design,* Vol. 109, No. 1, pp. 3–7.