M4. This gives us a final color (W4.2, 00 000) which corresponds to a part completely processed. Now the part can be unloaded by robot R2 or R3 from M4 and eventually leaves the cell. It is left to the reader to trace the token until it is unloaded from the cell.

V. Conclusion

In this paper we have introduced a new architecture to model a large class of flexible manufacturing systems using colored Petri nets. Advantages of this new architecture are:

1) cell model and part process information are separated, thus eliminating the need to update the CPN model every time there is a change in the part types manufactured in the system;
2) alternate sequencing of operations is allowed during processing;
3) machine assignments for operations are made dynamically during processing.

It is important to note that the model of the FMC created captures all possible operation sequences in the cell. Therefore, the structural analysis of the Petri net for static deadlock prevention policies is compromised. However, the important issue of deadlock can be addressed using deadlock avoidance policies as in [5].

References

[1] J. Ezpeleta and J. M. Colom, "Automatic synthesis of colored Petri nets for the control of FMS," *IEEE Trans. Robot. Automat.*, vol. 13, pp. 327–337, June 1997.
[2] M. D. Byrne and P. Chutima, "Real time operational control of an FMS with full routing flexibility," *Int. J. Prod. Econ.*, vol. 51, pp. 109–113, 1997.
[3] R. David and H. Alla, *Petri Nets and Grafcet.* Englewood Cliffs, NJ: Prentice-Hall, 1992.
[4] A. A. Desrochers and R. Y. Al-Jaar, *Application of Petri Nets in Manufacturing Systems*. Piscataway, NJ: IEEE Press, 1995.
[5] N. Viswanadham, Y. Narahari, and T. L. Johnson, "Deadlock prevention and deadlock avoidance in flexible manufacturing systems using Petri net models," *IEEE Trans. Robot. Automat.*, vol. 6, pp. 713–723, Dec. 1990.

Coordinate-Invariant Algorithms for Robot Dynamics

Scott R. Ploen and Frank C. Park

***Abstract*—In this article, we present, using methods from the theory of Lie groups and Lie algebras, a coordinate-invariant formulation of the dynamics of open kinematic chains. We first reformulate the recursive dynamics algorithm originally given in [8] for open chains in terms of standard linear operators on the Lie algebra of the Special Euclidean group. Using straight forward algebraic manipulations, we then recast the resulting algorithm into a set of closed-form dynamic equations; this transformation allows one to move easily between $O(n)$ recursive algorithms advantageous for computation, and closed-form equations advantageous for symbolic manipulation and analysis. The transformation also illuminates how the choice of link reference frames affects the computational structure. We then reformulate Featherstone's articulated body inertia algorithm [3] using this same geometric framework, and rederive Rodriguez *et al.*'s [11]–[13] square factorization of the mass matrix and its inverse. An efficient $O(n)$ recursive algorithm for forward dynamics is also extracted from the inverse factorization. The resulting equations lead to a succinct high-level description of robot dynamics in both joint and operational space coordinates that minimizes symbolic complexity without sacrificing computational efficiency, and provides the basis for a dynamics formulation that does not require link reference frames in the description of the forward kinematics.**

***Index Terms*— Lie algebra, Lie group, multibody dynamics, robot dynamics.**

I. Introduction

From the point of view of classical mechanics, deriving the equations of motion of a rigid-link manipulator is usually regarded as a straightforward procedure: once a suitable set of generalized coordinates and reference frames have been chosen, what remains is to apply either Lagrange's or Newton and Euler's equations to obtain the differential equations of motion. Anyone who has derived the dynamics of an actual manipulator, however, will have experienced firsthand the enormous complexity of the ensuing equations. Past research in robot dynamics has been driven primarily by a desire to reduce this complexity—there is now extensive literature on algorithms for efficiently computing the dynamics, usually in a recursive fashion (see, e.g., [1], [3], [5]).

Aside from computational considerations, however, many advanced applications, particularly in robot control and planning, require an explicit closed-form representation of the dynamic equations (e.g., [6]). Recent applications suggest that a useful dynamics formulation should maintain the balance between computational efficiency and the ease with which it can be manipulated at a high level. Further, it should be flexible enough to admit a degree of coordinate independence, viz., a given problem should not be bound to any

Manuscript received April 2, 1996; revised March 24, 1997. This paper was recommended for publication by Associate Editor Y. F. Zheng and Editor A. Goldenberg upon evaluation of the reviewers' comments. This work was supported by the National Science Foundation under Award CMS-9403019, a U.S. Department of Education GAANN Fellowship, the Engineering Research Center for Advanced Control and Instrumentation, and the Institute for Advanced Machinery Design, Seoul National University.

S. R. Ploen is currently with the Guidance and Control Analysis Group, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91009 USA (e-mail: sploen@grover.jpl.nasa.gov).

F. C. Park is with the School of Mechanical and Aerospace Engineering, Seoul National University, Seoul, Korea (e-mail: fcp@plaza.snu.ac.kr).

Publisher Item Identifier S 1042-296X(99)09534-8.

specific choice of reference frames and/or local coordinates to carry out the kinematic and dynamic analysis.

Motivated in part by these considerations, Featherstone [3] has developed a recursive dynamics formulation using the machinery of classical screw theory, while Rodriguez, Jain, and Kreutz-Delgado have developed the spatial operator algebra formulation of dynamics ([11]–[13]) by identifying structural similarities in the dynamic equations for open chains and the equations for discrete-time Kalman filtering. Khatib [4] has also proposed the operational space paradigm as a means of managing the complexity in dynamics-based control and planning tasks. In Park, Bobrow, and Ploen [8], the dynamic equations for an open chain manipulator are formulated in both recursive Newton–Euler and Lagrangian form using methods from the theory of Lie groups and Lie algebras.

This article develops a general coordinate-invariant mathematical framework for rigid-body dynamics, based on the same set of geometric tools first examined in [8], from which a complete range of dynamics algorithms (including those mentioned above) can be formulated in a uniform and mathematically consistent manner. The main elements of our geometric framework are introduced by reformulating the recursive dynamics algorithm for open chains in terms of standard linear operators on the Lie algebra of the Special Euclidean group. Using simple algebraic manipulations, the resulting algorithm is then recast into a set of closed-form dynamic equations; this transformation allows one to move easily between $O(n)$ recursive algorithms advantageous for computation, and closed-form equations advantageous for symbolic manipulation and analysis. Moreover, we explicitly show the invariance of the formulation to choice of link reference frames—the effect of choice of reference frames on the structure of the recursive computations now becomes transparent.

As a demonstration of the generality and flexibility of our geometric language for robot dynamics, we reformulate Featherstone's articulated body inertia algorithm [3] within our geometric framework, and re-derive Rodriguez *et al.*'s [11] square factorization of the mass matrix and its inverse without invoking results from estimation theory. Along the same lines, we also re-derive their spatial operator algebra-based $O(n)$ recursive forward dynamics algorithm, this time expressed entirely in terms of coordinate-invariant Lie algebraic operators. The operational space control formulation is also reformulated from the geometric perspective and is discussed in [10].

One of the difficulties with traditional dynamics formulations is the use of what are generally *ad hoc* definitions, conventions, and notation, in particular the derivation of specialized formulas which more often than not turn out to be standard results from linear algebra. Our geometric framework allows one to tap into the vast body of standard and well-known results from linear algebra and Lie theory. For example, one of the important main results of the spatial operator algebra formulation, the square factorization of the mass matrix and its inverse, turns out to be a straightforward consequence of the Matrix Inversion Lemma (or the Sherman–Morrison–Woodbury Formula). Above all, the geometric framework provides a common and unified mathematical language in which to express the ideas introduced by Silver *et al.* and other researchers in a concise, coordinate-invariant manner, as well as a powerful means of formulating dynamics algorithms for a wide range of applications.

II. The Equations of Motion

A. Recursive Newton–Euler Algorithm

Due to space limitations, the reader should consult [7]–[9] for a detailed discussion of the special Euclidean group SE(3), its corresponding Lie algebra se(3), and their associated adjoint representations.

We now briefly review the recursive formulation of robot dynamics as given in [8]. The idea behind the recursive formulation is a two-step iterative process where in the outward iteration the kinematics of each link are propagated from base to tip, and in the inward iteration the kinetics are propagated from tip to base. We make the following definitions (here all quantities are expressed in the corresponding link frame coordinates). Let $V_i \in \Re^{6\times 1}$ be the generalized velocity of link i, $F_i \in \Re^{6\times 1}$ the total generalized force transmitted from link $i-1$ to link i through joint i with its first three components corresponding to the moment vector, and τ_i the applied torque at joint i. Also, let $f_{i-1,i} = M_i e^{S_i q_i}$ denote the position and orientation of the link i frame relative to the link $i-1$ frame with $M_i \in SE(3)$ and $S_i = (\omega_i, 0) \in se(3)$. Here ω_i denotes a unit vector along the axis of rotation of joint i. (In this article we assume that single degree-of-freedom joints—revolute or prismatic joints—connect the links in the multibody chain.) Further, $J_i \in \Re^{6\times 6}$ is defined as

$$J_i = \begin{bmatrix} I_i - m_i[r_i]^2 & m_i[r_i] \\ -m_i[r_i] & m_i I_{3\times 3} \end{bmatrix}$$

where m_i is the mass of link i, r_i is the vector in link i coordinates from the origin of the link i frame to the center of mass of link i, I_i is the inertia tensor of link i about the center of mass, and $I_{3\times 3}$ denotes the 3×3 identity matrix.

The recursive equations can now be expressed in terms of our geometric definitions and notation as follows.

1) **Initialization**

$$Given: \quad V_0, \dot{V}_0, F_{n+1}.$$

2) **Forward recursion: for $i = 1$ to n do**

$$f_{i-1,i} = M_i e^{S_i q_i} \tag{1}$$

$$V_i = Ad_{f_{i-1,i}^{-1}}(V_{i-1}) + S_i \dot{q}_i \tag{2}$$

$$\dot{V}_i = S_i \ddot{q}_i + Ad_{f_{i-1,i}^{-1}}(\dot{V}_{i-1}) - ad_{S_i \dot{q}_i} Ad_{f_{i-1,i}^{-1}}(V_{i-1}). \tag{3}$$

3) **Backward recursion: for $i = n$ to 1 do**

$$F_i = Ad^*_{f_{i,i+1}^{-1}}(F_{i+1}) + J_i \dot{V}_i - ad^*_{V_i}(J_i V_i) \tag{4}$$

$$\tau_i = S_i^T F_i. \tag{5}$$

Here V_0 and $\dot{V}_0$ denote the generalized velocity and acceleration of the base respectively, and F_{n+1} denotes the force acting at the tip of the open chain. In the sequel we will assume that $V_0 = 0$ and that $\dot{V}_0 = (0, g)$ where $g \in \Re^{3-1}$ denotes the gravity vector in appropriate units and direction.

B. Global Matrix Representation of the Newton-Euler Algorithm

By expanding the individual equations (2)-(5) for $i = 1, 2, \cdots, n$ it can be shown that the recursive Newton–Euler algorithm admits the following global matrix representation:

$$V = GS\dot{q} + GP_0 V_0 \tag{6}$$

$$\dot{V} = GS\ddot{q} + Gad_{S\dot{q}}\Gamma V + Gad_{S\dot{q}} P_0 V_0 + GP_0 \dot{V}_0 \tag{7}$$

$$F = G^T J \dot{V} + G^T ad^*_V J V + G^T P_t^T F_{n+1} \tag{8}$$

$$\tau = S^T F \tag{9}$$

where

$$V = \text{column } [V_1, V_2, \cdots, V_n] \in \Re^{6n\times 1}$$
$$F = \text{column } [F_1, F_2, \cdots, F_n] \in \Re^{6n\times 1}$$
$$\dot{q} = \text{column } [\dot{q}_1, \dot{q}_2, \cdots, \dot{q}_n] \in \Re^{n\times 1}$$
$$\tau = \text{column } [\tau_1, \tau_2, \cdots, \tau_n] \in \Re^{n\times 1}$$
$$P_0 = \text{column } [Ad_{f_{0,1}^{-1}}, 0, \cdots, 0] \in \Re^{6n\times 6}$$
$$P_t^T = \text{column } [0, 0, \cdots, Ad^*_{f^{-1}_{n,n+1}}] \in \Re^{6n\times 6}$$
$$S = \text{diag } [S_1, S_2, \cdots, S_n] \in \Re^{6n\times n}$$
$$J = \text{diag } [J_1, J_2, \cdots, J_n] \in \Re^{6n\times 6n}$$
$$ad_{S\dot{q}} = \text{diag } [-ad_{S_1\dot{q}_1}, \cdots, -ad_{S_n\dot{q}_n}] \in \Re^{6n\times 6n}$$
$$ad^*_V = \text{diag } [-ad^*_{V_1}, \cdots, -ad^*_{V_n}] \in \Re^{6n\times 6n.}$$

Also, $\Gamma \in \Re^{6n\times 6n}$ is given by

$$\Gamma = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ Ad_{f_{1,2}^{-1}} & 0 & \cdots & 0 & 0 \\ 0 & Ad_{f_{2,3}^{-1}} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & Ad_{f_{n-1,n}^{-1}} & 0 \end{bmatrix}$$

and it can be verified by direct multiplication that $G = (I - \Gamma)^{-1}$ is

$$G = \begin{bmatrix} I_{6\times 6} & 0 & 0 & \cdots & 0 \\ Ad_{f_{1,2}^{-1}} & I_{6\times 6} & 0 & \cdots & 0 \\ Ad_{f_{1,3}^{-1}} & Ad_{f_{2,3}^{-1}} & I_{6\times 6} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ Ad_{f_{1,n}^{-1}} & Ad_{f_{2,n}^{-1}} & Ad_{f_{3,n}^{-1}} & \cdots & I_{6\times 6} \end{bmatrix}$$

where $I_{6\times 6}$ denotes the 6×6 identity matrix. Note that the eigenvalues of Γ are identically zero: As a result Γ is a nilpotent matrix (i.e., $\Gamma^n = 0$), and it is easy to verify that $G = (I - \Gamma)^{-1} = I + \Gamma + \cdots + \Gamma^{n-1}$. Also note that $G_{i,j} = G_{i,i-1}G_{i-1,i-2}\cdots G_{j-1,j}$ for $i > j$.

Combining (6)–(9), the equations of motion for an open chain manipulator can be expressed as

$$M(q)\ddot{q} + C(q, \dot{q}) + \phi(q) + J_t^T(q) f_t = \tau \tag{10}$$

where

$$M(q) = S^T G^T J G S \tag{11}$$
$$C(q, \dot{q}) = S^T G^T (J G a d_{S\dot{q}} \Gamma + a d^*_V J) G S \dot{q} \tag{12}$$
$$\phi(q) = S^T G^T J G P_0 \dot{V}_0 \tag{13}$$
$$J_t(q) = P_t G S. \tag{14}$$

Note that in (10) we have introduced the notation f_t for F_{n+1}.

The above matrix factorization of the equations of motion is an explicit representation of the $O(n)$ recursive dynamics algorithm and provides a demonstration of the equivalence of the Newton-Euler and Lagrangian formulations of robot dynamics [14]. One of the most useful structural features of the above equations is the transparent manner in which the robot parameters appear; for example, in the factorization of the mass matrix all the inertial parameters are contained in the constant block-diagonal matrix J, while S is a constant matrix containing only the kinematic parameters, and G is the only matrix dependent on the q_i.

C. Coordinate Invariance of the Dynamic Equations

Let $Q \in \Re^{6n\times 6n}$ be defined as

$$Q = \text{diag}[Ad_{Q_1}, Ad_{Q_2}, \cdots, Ad_{Q_n}],$$

with each Q_i an element of $SE(3)$, and Ad_{Q_i} a 6×6 matrix of the form discussed in [8]. Suppose S, G, and J now undergo the following transformation:

$$A = QS \tag{15}$$
$$L = QGQ^{-1} \tag{16}$$
$$D = Q^{-T} J Q^{-1}. \tag{17}$$

Note that if $S_i \in se(3)$ is expressed as a 6×1 vector then from (15) $A_i = Ad_{Q_i}(S_i)$; if S_i is expressed instead as a 4×4 matrix then $A_i = Q_i S_i Q_i^{-1}$. It is not difficult to see that any Q as defined above will preserve the structure of S, G, and J, i.e., A and L have the same block-matrix structure as S and G, etc.

Upon substitution of (15)–(17) into the equations of motion (11)–(14), we find

$$M(q) = A^T L^T D L A \tag{18}$$
$$C(q, \dot{q}) = A^T L^T (D L \overline{ad}_{S\dot{q}} \overline{\Gamma} + \overline{ad}^*_V D) L A \dot{q} \tag{19}$$
$$\phi(q) = A^T L^T D L \overline{P}_0 \dot{V}_0 \tag{20}$$
$$J_t(q) = \overline{P}_t L A \tag{21}$$

where

$$\overline{\Gamma} = Q \Gamma Q^{-1} \tag{22}$$
$$\overline{ad}_{S\dot{q}} = Q a d_{S\dot{q}} Q^{-1} \tag{23}$$
$$\overline{ad}^*_V = Q^{-T} a d^*_V Q^T \tag{24}$$
$$\overline{P}_0 = Q P_0 \tag{25}$$
$$\overline{P}_t^T = Q^{-T} P_t^T. \tag{26}$$

The expression for $\overline{\Gamma}$ is obtained via the following identity: $I - L^{-1} = Q(I - G^{-1})Q^{-1} = Q\Gamma Q^{-1}$.

Upon comparing (18)–(21) to (11)–(14) it is apparent that the structure of the equations of motion is unchanged under the coordinate transformation defined by Q. This invariance is a result of the fact that M, C, ϕ and J_t are direct tensor products of known tensor quantities (recall that direct products of tensors are themselves tensors). According to the transformation rules given above, under a change of coordinates A and P_0 transform as vector quantities, or type (0,1) tensors, P_t transforms as a covector, or a type (1,0) tensor, D transforms as an inner product acting on vectors, or a type (2,0) tensor, and L, Γ, $ad_{S\dot{q}}$ and $\overline{ad}^*_V$ transform as linear operators, or (1,1) tensors. For a well-written discussion of tensor analysis see [2].

Physically, different choices of Q correspond to different sets of local link reference frames in which to express the kinematic and dynamic parameters of the robot. As a concrete example consider a change of local link reference frames defined by $Q = \text{diag}[Ad_{Q_1}, Ad_{Q_2}, \cdots, Ad_{Q_n}]$ where $Q_i = (I, -r_i)$ and r_i denotes the vector in link i coordinates from the origin of the link i frame to the center of mass of link i. Under the coordinate transformation defined by Q the equations of motion of each link are expressed with respect to local link references frames attached at the center of mass of each link. As expected, it is easily shown that the generalized inertia matrix for each link with respect to the center of mass reference frames is given by $D_i = Ad^{-T}_{Q_i} J_i Ad^{-1}_{Q_i} = \text{diag}[I_i, m_i I_{3\times 3}]$.

The cancellation of Q upon substituting (15)–(17) into the equations of motion (11)–(14) shows that the equations of motion as expressed above are not bound to any specific assignment of local link reference frames for representing the physical parameters of the robot. Moreover, each choice of admissible Q leads to a dynamics

formulation that can be computed recursively. A general coordinate invariant recursive algorithm is obtained by substituting (15)–(17) and (22)–(26) into the global matrix representation of the Newton–Euler algorithm (6)–(9)

$$\overline{V} = LA\dot{q} + L\overline{P}_0 V_0 \tag{27}$$

$$\dot{\overline{V}} = LA\ddot{q} + L\overline{ad}_{S\dot{q}}\overline{FV} + L\overline{ad}_{S\dot{q}}\overline{P}_0 V_0 + L\overline{P}_0\dot{V}_0 \tag{28}$$

$$\overline{F} = L^T D\dot{\overline{V}} + L^T\overline{ad}_V^* D\overline{V} + L^T\overline{P}_t^T F_{n+1} \tag{29}$$

$$\tau = A^T\overline{F} \tag{30}$$

where

$$\overline{V} = QV \tag{31}$$

$$\dot{\overline{V}} = Q\dot{V} \tag{32}$$

$$\overline{F} = Q^{-T}F. \tag{33}$$

Upon direct expansion of (27)–(30), it can be shown that they are equivalent to the following recursive algorithm:

1) **Initialization**

$$Given: \quad V_0, \dot{V}_0, F_{n+1}$$

2) **Forward recursion: for $i = 1$ to n do**

$$f_{i-1,i} = Q_{i-1}M_i Q_i^{-1} e^{A_i q_i} \tag{34}$$

$$\overline{V}_i = Ad_{f_{i-1,i}^{-1}}(\overline{V}_{i-1}) + A_i\dot{q}_i \tag{35}$$

$$\dot{\overline{V}}_i = A_i\ddot{q}_i + Ad_{f_{i-1,i}^{-1}}(\dot{\overline{V}}_{i-1}) - ad_{A_i\dot{q}_i}Ad_{f_{i-1,i}^{-1}}(\overline{V}_{i-1}) \tag{36}$$

3) **Backward recursion: for $i = n$ to 1 do**

$$\overline{F}_i = Ad^*_{f_{i,i+1}^{-1}}(\overline{F}_{i+1}) + D_i\dot{\overline{V}}_i - ad^*_{\overline{V}_i}(D_i\overline{V}_i) \tag{37}$$

$$\tau_i = A_i^T\overline{F}_i. \tag{38}$$

Note that the above equations have exactly the same form as the recursive Newton-Euler equations with S_i replaced by A_i, M_i replaced by $Q_{i-1}M_iQ_i^{-1}$, J_i replaced by D_i and all generalized velocities, accelerations, and forces replaced by their components in the new set of local link reference frames defined by Q.

III. Square Factorization of the Mass Matrix

The factorization of the mass matrix given in (11) is not a square factorization in the sense that S is not a square matrix. As a result it is not possible to use this factorization to invert the mass matrix explicitly. Rodriguez *et al.* [13] have derived a square factorization of the mass matrix and its inverse using results from estimation theory. In this section, we determine an alternative square factorization of M and M^{-1} using our earlier Lie algebraic results, and explicitly show how this factorization transforms under a change of coordinates.

Featherstone [3] has shown that the open chain equations of motion can alternately be formulated recursively in the following manner:

$$F_i = \hat{J}_i\dot{V}_i + b_i \qquad i = n, \cdots, 1 \tag{39}$$

where $\hat{J}_i$ is the *articulated body inertia* of link i, and $b_i = b_i(V_i, V_{i+1}, S_{i+1}, \hat{J}_{i+1}, \tau_{i+1})$ is the *bias force* associated with link i.

Upon expressing the quantities appearing in Featherstone's articulated body inertia algorithm in terms of our geometric definitions and notation it can be shown [9] that Featherstone's $\hat{J}_i$ is related to the J_i from the generalized Newton–Euler algorithm as follows.

1) **Initialization**

$$\hat{J}_n = J_n.$$

2) **Backward recursion: for $i = n - 1$ to 1 do**

$$\hat{J}_i = J_i + Ad^*_{f_{i,i+1}^{-1}}\hat{J}_{i+1}Ad_{f_{i,i+1}^{-1}} - \frac{Ad^*_{f_{i,i+1}^{-1}}\hat{J}_{i+1}S_{i+1}S_{i+1}^T\hat{J}_{i+1}Ad_{f_{i,i+1}^{-1}}}{S_{i+1}^T\hat{J}_{i+1}S_{i+1}}. \tag{40}$$

In terms of our earlier definitions, the above recursion is equivalent to the following matrix equation:

$$J = \hat{J} - \Gamma^T\hat{J}\Gamma + \Gamma^T\hat{J}S(S^T\hat{J}S)^{-1}S^T\hat{J}\Gamma \tag{41}$$

where the symmetric matrix $\hat{J}$ is defined as $\hat{J} = \text{diag}[\hat{J}_1, \hat{J}_2, \cdots, \hat{J}_n] \in \Re^{6n\times 6n}$.

A square factorization of M results if J is expressed as a function of $\hat{J}$ in the factorization $M = S^TG^TJGS$:

Proposition 1: The mass matrix M can be expressed in terms of the $n \times n$ factors $[I + S^TG^T\Pi]$ and Ω as follows:

$$M = [I + S^TG^T\Pi]\Omega[I + S^TG^T\Pi]^T$$

where $\Omega = S^T\hat{J}S \in \Re^{n\times n}$, $\Phi = \Omega^{-1}S^T\hat{J} \in \Re^{n\times 6n}$, $\Pi = \Gamma^T\Phi^T \in \Re^{6n\times n}$, and I is the 3×3 identity matrix.

Proof: Substituting (41) into S^TG^TJGS and using the identity $\Gamma G = G - I$ yields

$$M = S^TG^T\hat{J}S + S^T\hat{J}GS - S^T\hat{J}S + S^TG^T\Gamma^T\hat{J}S(S^T\hat{J}S)^{-1}S^T\hat{J}\Gamma GS.$$

Adding and subtracting $S^T\hat{J}S$ to the above equation and rearranging yields

$$M = S^T\hat{J}S + S^T(G^T - I)\hat{J}S + S^T\hat{J}^T(G - I)S + S^TG^T\Gamma^T\hat{J}S(S^T\hat{J}S)^{-1}S^T\hat{J}\Gamma GS.$$

Using the identity $G - I = \Gamma G$ results in

$$M = S^T\hat{J}S + S^TG^T\Gamma^T\hat{J}S + S^T\hat{J}^T\Gamma GS + S^TG^T\Gamma^T\hat{J}S(S^T\hat{J}S)^{-1}S^T\hat{J}\Gamma GS.$$

Upon post-multiplying the second term by $(S^T\hat{J}S)^{-1}$ $(S^T\hat{J}S)$, pre-multiplying the third term by $(S^T\hat{J}S)$ $(S^T\hat{J}S)^{-T}$, and noting that $(S^T\hat{J}S)^{-1}$ from the last term also equals $(S^T\hat{J}S)^{-1}$ $(S^T\hat{J}S)$ $(S^T\hat{J}S)^{-T}$, the result follows after an elementary calculation. □

Note that $\Omega = Diag[\Omega_1, \Omega_2, \cdots, \Omega_n] \in \Re^{n\times n}$, $\Phi = Diag[\Phi_1, \Phi_2, \cdots, \Phi_n] \in \Re^{n\times 6n}$, where $\Omega_k = S_k^T\hat{J}_kS_k$ and $\Phi_k = S_k^T\hat{J}_k\Omega_k^{-1} = S_k^T\hat{J}_k/S_k^T\hat{J}_kS_k$. Here we have used the fact that Ω_k^{-1} exists since $\hat{J}_k > 0$ (See [3] for the details). Further, $\Pi \in \Re^{6n\times n}$ has the following structure:

$$\Pi = \begin{bmatrix} 0 & \Pi_{1,2} & 0 & \cdots & 0 \\ 0 & 0 & \Pi_{2,3} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \Pi_{n-1,n} \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

where $\Pi_{k,k+1} = Ad^*_{f_{k,k+1}^{-1}}\hat{J}_{k+1}S_{k+1}\Omega_{k+1}^{-1}$.

Under the coordinate transformation defined by Q it can be shown that

$$M = [I + A^TL^T\overline{\Pi}]\Omega[I + A^TL^T\overline{\Pi}]^T \tag{42}$$

where $\overline{\Pi} = Q^{-T}\Pi$, $\Omega = A^T\hat{\overline{J}}A$, $\hat{\overline{J}} = Q^{-T}\hat{J}Q^{-1}$. Equation (42) provides a coordinate-invariant expression for the square decomposition of M.

IV. Inversion of the Mass Matrix

The square factorization of M immediately leads to a similar square factorization for M^{-1}.

Proposition 2: The inverse mass matrix M^{-1} is given by

$$M^{-1} = [I - S^T Y \Pi]^T \Omega^{-1} [I - S^T Y \Pi]$$

where $Y = (I - X^T)^{-1} \in \Re^{6n \times 6n}$ and $X^T = \Gamma^T (I - \hat{J} S (S^T \hat{J} S)^{-1} S^T) = \Gamma^T (I - \Phi^T S^T)$.

Proof: Applying the well-known matrix inversion lemma (also known as the Sherman–Morrison–Woodbury Formula)

$$(A + BCD)^{-1} = A^{-1} - A^{-1} B (DA^{-1}B + C^{-1})^{-1} DA^{-1}$$

to $[I + S^T G^T \Pi]^{-1}$ and recalling $G^{-T} = (I - \Gamma^T)$, the result follows after a routine calculation. □

The above proof is similar to that given in Rodriguez *et al.* [11]. Note that $X = (I - S\Phi)\Gamma \in \Re^{6n \times 6n}$ has the following structure:

$$X = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 \\ X_{2,1} & 0 & \cdots & 0 & 0 \\ 0 & X_{3,2} & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & X_{n,n-1} & 0 \end{bmatrix}$$

where

$$X_{k+1,k} = \left(I - \frac{S_{k+1} S_{k+1}^T \hat{J}_{k+1}}{S_{k+1}^T \hat{J}_{k+1} S_{k+1}} \right) Ad_{f_{k,k+1}^{-1}}.$$

Clearly X has the same structure as Γ and is also a nilpotent matrix. As a result, $Y = I + X^T + \cdots + (X^T)^{n-1}$ and has the same structure as G^T, i.e.,

$$Y = \begin{bmatrix} I & Y_{1,2} & Y_{1,3} & \cdots & Y_{1,n} \\ 0 & I & Y_{2,3} & \cdots & Y_{2,n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & Y_{n-1,n} \\ 0 & 0 & 0 & \cdots & I \end{bmatrix}.$$

It follows that $Y_{i,j} = Y_{i,i+1} \cdots Y_{j-1,j}$ for $i < j$.

Under the coordinate transformation defined by Q it can be shown that

$$M^{-1} = [I - A^T \overline{Y}\, \overline{\Pi}]^T \Omega^{-1} [I - A^T \overline{Y \Pi}] \tag{43}$$

where $\overline{Y} = (I - \overline{X}^T)^{-1}$, $\overline{X}^T = Q^{-T} X^T Q^T$. Equation (43) provides a coordinate-invariant representation of the inverse mass matrix.

V. Relation to the Spatial Operator Algebra

Our derivation of the square factorization of the mass matrix and its inverse presented above is similar to that of Rodriguez *et al.* [11]–[13]. However, while the spatial operator approach is based on analogies from optimal estimation and filtering theory, our results are derived via basic principles and concepts from mechanics and geometry. One of the advantages of this geometric approach is that the resulting computational algorithms are independent of any assignment of link reference frames in the description of the forward kinematics. To assist the reader in understanding the connection between our geometric formulation and the algebraic formulation of Rodriguez *et al.*, the correspondence between our Lie group based operators and the spatial chance operators of Rodriguez *et al.* are summarized in Table I.

It follows from the Table I that Γ^T corresponds to ε_ϕ, X corresponds to ε_ψ^*, J corresponds with M, S corresponds to H^*, etc. It should be pointed out that these correspondences are qualitative in nature, as Rodriguez *et al.* do not follow the standard link numbering convention in robotics: in the spatial operator algebra the links are numbered $n, n-1, \cdots, 1$ outward rather than the usual $1, 2, \cdots, n$. Also, the spatial operators given in [11] are expressed as linear operators in basis free notation rather than as matrices with respect to local body-fixed reference frames.

TABLE I
Relationship Between Spatial Operators and Lie Group Framework

Lie Group Operator	Spatial Operator	Size
$G^T = (I - \Gamma^T)^{-1}$	$\phi = (I - \varepsilon_\phi)^{-1}$	$6n \times 6n$
$\hat{J} = X^T \hat{J} X + J$	$P = \varepsilon_\psi P \varepsilon_\psi^* + M$	$6n \times 6n$
$\Omega = S^T \hat{J} S$	$D = HPH^*$	$n \times n$
$\Phi^T = \hat{J} S \Omega^{-1}$	$G = PH^* D^{-1}$	$6n \times n$
$\Pi = \Gamma^T \Phi^T$	$K = \varepsilon_\phi G$	$6n \times n$
$X^T = \Gamma^T (I - \Phi^T S^T)$	$\varepsilon_\psi = \varepsilon_\phi (I - GH)$	$6n \times 6n$
$Y = (I - X^T)^{-1}$	$\psi = (I - \varepsilon_\psi)^{-1}$	$6n \times 6n$

VI. Algorithms for Forward Dynamics

The recursive Newton–Euler equations given in (1)–(5) are well suited for solving the inverse dynamics problem; viz., given the motion of the system $(q, \dot{q}, \ddot{q})$, determine the joint torques τ. In applications such as simulation it is also necessary to solve the forward dynamics problem; i.e., given the initial state of the system $(q_0, \dot{q}_0)$, the applied generalized forces acting at the tip f_t, and the applied torques τ, determine the accelerations of the system $\ddot{q}$.

Recall from (10) that the accelerations of the system are given by

$$\ddot{q} = M^{-1}(\tau - C - \phi - J_t^T f_t). \tag{44}$$

Using (18)–(21) and (43) it can be shown that

$$\ddot{q} = [I - A^T \overline{Y}\, \overline{\Pi}]^T \Omega^{-1} [I - A^T \overline{Y}\, \overline{\Pi}] \hat{\tau} \tag{45}$$

where

$$\hat{\tau} = \tau - A^T L^T [DL(\tilde{a} + \overline{P}_0 \dot{V}_0) + \tilde{b} + \overline{P}_t^T f_t]. \tag{46}$$

Here $\tilde{a} = \overline{ad}_{S\dot{q}} \overline{\Gamma} L A \dot{q}$ and $\tilde{b} = \overline{ad}_V^* D L A \dot{q}$.

Rodriguez *et al.* show that their square factorization of the mass matrix leads to a recursive algorithm for the forward dynamics [12]. Similarly, the following coordinate invariant $O(n)$ algorithm for robot forward dynamics is embedded within the structure of (45) and (46) (See [9] for more details):

The forward kinematic maps $f_{k,k-1}$ can be computed recursively as follows.

1) **Initialization**

$$f_{0,1} = M_1 Q_1^{-1} e^{A_1 q_1} \tag{47}$$

$$f_{n,n+1} \to Q_n f_{n,n+1}. \tag{48}$$

2) **Forward recursion: for $k = 2$ to n do**

$$f_{k-1,k} = Q_{k-1} M_k Q_k^{-1} e^{A_k q_k}. \tag{49}$$

It is straightforward to show by direct expansion that the expressions for $\tilde{a}$ and $\tilde{b}$ are equivalent to the following $O(n)$ recursive algorithm.

1) **Initialization**

$$S_0 = 0, \quad \hat{a}_0 = 0, \quad \overline{V}_0 = 0. \tag{50}$$

2) **Forward recursion: for $k = 1$ to n do**

$$\hat{a}_k = Ad_{f_{k-1,k}^{-1}} (A_{k-1} \dot{q}_{k-1} + \hat{a}_{k-1}) \tag{51}$$

$$\overline{V}_k = Ad_{f_{k-1,k}^{-1}} \overline{V}_{k-1} + A_k \dot{q}_k \tag{52}$$

$$\tilde{a}_k = -\overline{ad}_{S_k \dot{q}_k} \hat{a}_k \tag{53}$$

$$\tilde{b}_k = -\overline{ad}_{V_k}^* D_k \overline{V}_k. \tag{54}$$

Once $\tilde{a}$ and $\tilde{b}$ have been computed it can be shown by direct expansion that (46) is equivalent to the following recursive algorithm.

1) **Initialization**

$$\alpha_0 = 0, \tilde{a}_1 \rightarrow \tilde{a}_1 + Ad_{f_{0,1}^{-1}} \dot{V}_0. \tag{55}$$

2) **Forward recursion: for $k = 1$ to n do**

$$\alpha_k = Ad_{f_{k-1,k}^{-1}} \alpha_{k-1} + \tilde{a}_k \tag{56}$$

$$\overline{\alpha}_k = D_k \alpha_k + \tilde{b}_k. \tag{57}$$

3) **Initialization**

$$P_{n+1} = 0, \overline{\alpha}_n \rightarrow \overline{\alpha}_n + Ad^*_{f_{n,n+1}^{-1}} f_t. \tag{58}$$

4) **Backward recursion: for $k = n$ to 1 do**

$$P_k = Ad^*_{f_{k,k+1}^{-1}} P_{k+1} + \overline{\alpha}_k \tag{59}$$

$$\hat{P}_k = A_k^T P_k \tag{60}$$

$$\hat{\tau}_k = \tau_k - \hat{P}_k. \tag{61}$$

Once $\hat{\tau}$ has been computed it can be shown by direct expansion that (45) is equivalent to the following recursive algorithm.

1) **Initialization**

$$\hat{z}_{n+1} = 0, \hat{\tau}_{n+1} = 0. \tag{62}$$

2) **Backward recursion: for $k = n$ to 1 do**

$$\hat{z}_k = \overline{Y}_{k,k+1} \hat{z}_{k+1} + \overline{\Pi}_{k,k+1} \hat{\tau}_{k+1} \tag{63}$$

$$c_k = \hat{\tau}_k - A_k^T \hat{z}_k \tag{64}$$

$$\hat{c}_k = \Omega_k^{-1} c_k. \tag{65}$$

3) **Initialization**

$$\lambda_0 = 0. \tag{66}$$

4) **Forward recursion: for $k = 1$ to n do**

$$\lambda_k = \overline{Y}^T_{k-1,k} \lambda_{k-1} + A_k \hat{c}_k \tag{67}$$

$$\ddot{q}_k = \hat{c}_k - \overline{\Pi}^T_{k-1,k} \lambda_{k-1}. \tag{68}$$

Here

$$\overline{Y}_{k,k+1} = \overline{X}^T_{k+1,k} = Ad^*_{f_{k,k+1}^{-1}} \left(I - \frac{\hat{\overline{J}}_{k+1} A_{k+1} A_{k+1}^T}{A_{k+1}^T \hat{\overline{J}}_{k+1} A_{k+1}} \right)$$

and

$$\overline{\Pi}_{k,k+1} = \frac{Ad^*_{f_{k,k+1}^{-1}} \hat{\overline{J}}_{k+1} A_{k+1}}{A_{k+1}^T \hat{\overline{J}}_{k+1} A_{k+1}}.$$

VII. Conclusion

In this article, we have presented a unified coordinate-invariant formulation of the dynamics of multibody open chain manipulators based on standard concepts from geometry and mechanics. One of the major benefits of our geometric formulation is that it provides a single unified framework to express ideas originally introduced by Silver, Featherstone and Rodriguez *et al.* in a clean, concise, and coordinate-invariant manner. We then showed that the resulting dynamic equations can be expressed recursively for applications requiring computationally efficient dynamics algorithms, or can be cast into closed-form for applications requiring high-level manipulation of the equations of motion. Moreover, the dynamic equations are formulated in a completely coordinate-invariant manner, and as a result are not bound to any specific set of local link reference frames in which to express the kinematic and dynamic parameters of the robot.

References

[1] C. A. Balafoutis and R. V. Patel, *Dynamic Analysis of Robotic Manipulators: A Cartesian Tensor Approach*. Boston, MA: Kluwer, 1991.

[2] B. A. Dubrovin, A. T. Fomenko, and S. P. Novikov, *Modern Geometry: Methods and Applications, Part 1*. New York: Springer-Verlag, 1984.

[3] R. Featherstone, *Robot Dynamics Algorithms*. Boston, MA: Kluwer, 1987.

[4] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Robot. Automat.*, vol. RA-3, pp. 43–53, Feb. 1987.

[5] J. Y. S. Luh, M. H. Walker, and R. P. Paul, "On-line computational scheme for mechanical manipulators," *ASME J. Dyn. Syst., Meas., Contr.*, vol. 102, pp. 69–76, 1980.

[6] B. J. . Martin and J. E. Bobrow, "Minimum-effort motions for open-chain manipulators with task-dependent end-effector constraints," *Int. J. Robot. Res.*, vol. 18, no. 2, pp. 213–224, 1999.

[7] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robot Manipulation*. Boca Raton, FL: CRC, 1993.

[8] F. C. Park, J. E. Bobrow, and S. R. Ploen, "A Lie group formulation of robot dynamics," *Int. J. Robot. Res.*, vol. 14, no. 6, pp. 609–618, 1995.

[9] S. R. Ploen, *Geometric Algorithms for the Dynamics and Control of Multibody Systems*, Ph.D. dissertation, Univ. California, Irvine, 1997, http://www.eng.uci.edu/~sploen/srpindex.html.

[10] S. R. Ploen, J. E. Bobrow, and F. C. Park, "Geometric algorithms for operational space dynamics and control," in *Proc. IEEE Int. Conf. Robot. Automat.*, Albuquerque, NM, 1997, pp. 1606–1611.

[11] G. Rodriguez, A. Jain, and K. Kreutz-Delgado, "A spatial operator algebra for manipulator modeling and control," *Int. J. Robot. Res.*, vol. 10, no. 4, pp. 371–381, 1991.

[12] ——, "Spatial operator algebra for multibody system dynamics," *J. Astronaut. Sci.*, vol. 40, no. 1, pp. 27–50, 1992.

[13] G. Rodriguez and K. Kreutz-Delgado, "Spatial operator factorization and inversion of the manipulator mass matrix," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 65–76, Feb. 1992.

[14] W. M. Silver, "On the equivalence of Lagrangian and Newton-Euler dynamics for manipulators," *Int. J. Robot. Res.*, vol. 1, no. 2, pp. 118–128, 1982.

A Vision-Based Method for the Circle Pose Determination With a Direct Geometric Interpretation

Zen Chen and Jen-Bin Huang

***Abstract*— A novel vision-based method for the circle pose determination is addressed. This method is based on two particular projected chords of a circle image. The first one is the projection of a circle chord which subtends the largest apex angle of the viewing cone for the circle image and the second one is the projection of a circle diameter whose backprojection plane bisects the above largest apex angle. This method is conceptually simple, since the circle center is the center point of the diameter chord and the circle orientation is given by the cross product of these two (directed) chords. We present theorems on the geometry of both the viewing cone and a reprojected circle image which are essential to the pose determination. We then give a pose determination method which is applicable under all viewing conditions. Experimental results illustrate the good performance of the method, when compared with other existing methods.**

***Index Terms*—Circular feature, elliptical image, largest apex angle, pose determination, principal axes, viewing cone.**

Manuscript received July 16, 1997; revised May 25, 1999. This paper was recommended for publication by Associate Editor T. Henderson and Editor V. Lumelsky upon evaluation of the reviewers' comments.

The authors are with the Department of Computer Science and Information Engineering, National Chiao Tung University, Hsinchu, Taiwan 30050, R.O.C. (e-mail: zchen@csie.nctu.edu.tw).

Publisher Item Identifier S 1042-296X(99)09234-4.