2) The determined calibration plane can in turn be used to detect errors associated with the range images of a 3-D LIDAR.
3) Both a polynomial model and a polynomial-ARV model have been proposed in this new method to compensate for the range image errors.

The experimental results indicate that the compensation based on the polynomial model can reduce the range image errors from 163 counts to 18 counts. The polynomial-ARV model can further reduce the range image error by more than seven counts by considering the dynamic characteristics imbedded in the range image errors.

REFERENCES


Regressors Formulation of Robot Dynamics: Computation and Applications

W.-S. Lu and Q.-H. Meng

Abstract—Two approaches to the evaluation of the manipulator regressor of a general n-degree-of-freedom (DOF) robot are presented. The first method is an "energy-based" approach using Lagrangian formulation of robot dynamics as a starting point. A key fact used in deriving the solution is that the manipulator Lagrangian is linewise additive. The second approach generates an iterative algorithm for efficient numerical evaluation of the regressor. It is obtained by reformulating the Newton–Euler recursion using vector analysis type techniques. In addition, a modified Sbolme–LI algorithm for adaptive motion control is presented and is then applied in a simulation study to a 4-DOF PUMA-type robot, where the manipulator regressor is evaluated using the iterative algorithm proposed.

I. INTRODUCTION

The manipulator regressor, often denoted by $y(q, \dot{q}, \ddot{q})$, is a key quantity in derivation as well as implementation of the many established adaptive motion and force control algorithms [1], [2]. This is because its availability enables one to express the dynamics of a robot arm as $Y\dot{\theta} = \tau$ with $\theta \in \mathbb{R}^n$ representing the manipulator parameters, thus a Lyanpunov approach may lead to a linear law for updating the parameters. Studies on this linear parameter-dependence issue from an identification point of view can be found in [3]–[5] among others.

In principle, the regressor can be obtained by using a two-step approach. The first step is to formulate the manipulator dynamics as

$$H(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau.$$  (1)

This can be accomplished, for example, by using the Newton–Euler or Lagrange formulation, see [6, ch. 6] for details of these formulations and their computational complexity. Having done this, the second step of the approach defines a parameter vector $\theta$ and then works on every entry on the left-hand side of (1) to extract vector $\theta$, leading (1) to the regressor formulation $Y\theta = \tau$. So we see computationally that this is an indirect approach that requires formulating (1) plus a parameter extraction procedure. As the entries of $\theta$ are, in general, spread over all the entries of $H(q)$, $C(q, \dot{q})$, and $G(q)$, the second step is also computationally complicated.

In this paper, we propose two methods that compute the regressor of a general n-degree-of-freedom (DOF) robot without using (1). Our first method provides a closed-form solution, which is obtained by extracting parameter $\theta$ from link Lagrangians during the Lagrangian formulation; our second method gives a recursive-type solution, which is obtained by extracting parameter $\theta$ from joint velocities, accelerations, forces, and torques during the Newton–Euler formulation. As opposed to the conventional two-step approach, in which one derives (1) with the entries $\theta$ spread widely over the terms and then extracts these parameters term by term, the proposed

Manuscript received May 15, 1990; revised May 4, 1992. A portion of this work was presented at the Canadian Conference on Electrical and Computer Engineering, Ottawa, Canada, September 1990, and at the IEEE Pacific Rim Conference, Victoria, BC, Canada, May 1991. This work was supported in part by the Networks of Excellence Program of the Government of Canada and in part by PRECARN Associates Inc.

W.-S. Lu is with the Department of Electrical and Computer Engineering, University of Victoria, Victoria, BC, V8W 3P6 Canada.

Q.-H. Meng is with the Department of Electrical Engineering, Lakehead University, ON, Canada.

IEEE Log Number 9207364.
methods perform the parameter extraction and dynamics formulation simultaneously and, therefore, more efficiently. Furthermore, unlike the two-step approach, which defines parameter vector \( \theta \) by trial-and-error until \( \theta \) can be extracted from every term on the left-hand side of (1) (see the example in Section III-C for details), the formation of \( \theta \) in the proposed methods has an explicit rule to follow. It is found that, if for a specified link the mass, mass center, and inertia tensor are the parameters to be extracted, then this portion of vector \( \theta \) has dimension 16 although the number of physical parameters related to the link is 10. In addition, the so-called filtered regressor adopted in several adaptive motion control algorithms [7], [8] can readily be obtained from the proposed formula.

The paper is organized as follows. Preliminaries on the manipulator regressor and its relation to link Lagrangians are discussed in Section II. In Section III, we present a closed-form solution to the regressor evaluation problem. Special cases for robots having point-mass links (or load) and links with regular geometry are addressed. In Section IV, we propose an iterative algorithm for evaluating the manipulator regressor. The proposed methods are applied to obtain the regressor of a 2-DOF robot grasping a non-point-mass object. In Section V, a modified version of the Slotine–Li algorithm for adaptive motion control [5] is described, and its stability is shown via a Lyapunov approach. The proposed algorithm is then applied in a simulation study to a simplified 4-DOF PUMA-type robot, where the major steps of the control algorithm are expressed in terms of the regressor, which is evaluated using the proposed recursive algorithm.

II. PRELIMINARIES

A. \( Y_\alpha(q, \dot{q}, \ddot{q}) \)–Regressor Associated with Unknown Parameters

It is known [1], [5] that (1) can be written in the form of

\[
Y(q, \dot{q}, \ddot{q})\theta = \tau, \tag{2}
\]

where \( Y(q, \dot{q}, \ddot{q}) = Hq + C\dot{q} + G \) is the manipulator regressor and \( \theta \in R^{n+1} \) is the vector formed by the dynamic parameters of the manipulator in a certain manner. Denoting

\[
\theta = \left[ \begin{array}{c} \theta_k \\ \theta_u \end{array} \right], \tag{3}
\]

with \( \theta_k \in R^{n+1} \) and \( \theta_u \in R^2 \) representing the known and unknown parameters, respectively, and

\[
Y = [Y_k, Y_u], \tag{4}
\]

with \( Y_k \in R^{n \times (n+1) + n \times n+1} \) and \( Y_u \in R^{n \times 2} \), (1) and (2) imply that

\[
\dot{\ddot{q}} + \dddot{q} + \dddot{q} + G = Y_k \dot{\theta}_k + Y_u \dot{\theta}_u. \tag{5}
\]

If \( \dot{\theta}_u \) is an estimate of \( \theta_u \), then

\[
\dot{\ddot{q}} + \dddot{q} + \dddot{q} = Y_k \dot{\theta}_k + Y_u \dot{\theta}_u, \tag{6}
\]

where \( \dot{\ddot{q}} \) and \( \dddot{q} \) assume the same forms as \( H, C, \) and \( g \), respectively, with \( \dot{\theta}_u \) replaced by \( \dot{\theta}_u \). It follows that

\[
\dot{\ddot{q}} + \dddot{q} + \dddot{q} = Y_u \dot{\theta}_u, \tag{7}
\]

where \( (\dot{\ddot{q}} + \dddot{q}) = (\dddot{q}) \). It is (7) that plays a role in the establishment of the many stable algorithms for adaptive control of robots.

B. Relation of \( Y(q, \dot{q}, \ddot{q}) \) to the Lagrangians of Manipulator Links and Load

Consider an \( n \)-DOF manipulator grasping firmly a non-point-mass load. Denote by \( k(i) \) and \( u(i) \) \( 1 \leq i \leq n \) the kinetic and potential energy of link \( i \), and by \( k(n+1) \) and \( u(n+1) \) the kinetic and potential energy of the load, respectively. If the load is treated as link \( n + 1 \) and the link Lagrangian of link \( i \) is defined as

\[
L^{(i)} = k^{(i)} - u^{(i)}, \quad 1 \leq i \leq n + 1, \tag{8}
\]

then the manipulator Lagrangian is

\[
L = \sum_{i=1}^{n+1} L^{(i)}. \tag{9}
\]

If follows that the manipulator Lagrangian is linkwise additive; that is, if a new link is added to the robot, its manipulator Lagrangian is then equal to the original \( L \) plus the Lagrangian of the new link. This property turns out to be a key fact in the subsequent derivation of the regressor as it allows one to separate the parameters of a specific link from the parameters of other links. From (2), (9), and the Lagrange's equation of motion

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{q}} \right) - \frac{\partial L}{\partial q} = \tau, \tag{10}
\]

we obtain

\[
\sum_{i=1}^{n+1} \left[ \frac{d}{dt} \left( \frac{\partial L^{(i)}}{\partial \dot{q}} \right) - \frac{\partial L^{(i)}}{\partial q} \right] = Y(q, \dot{q}, \ddot{q})\theta, \tag{11}
\]

Now if \( \theta \) is partitioned as

\[
\theta = \left[ \begin{array}{c} \theta^{(1)} \\ \theta^{(2)} \\ \vdots \\ \theta^{(n+1)} \end{array} \right], \tag{12}
\]

with \( \theta^{(i)} \) representing the dynamic parameters of link \( i \), that is, its mass, mass center, and inertia tensor, and if \( Y \) is partitioned into \( n + 1 \) blocks

\[
Y = \begin{bmatrix} Y^{(1)} & Y^{(2)} & \cdots & Y^{(n+1)} \end{bmatrix}, \tag{13}
\]

with dimensions consistent to (12), then

\[
\frac{d}{dt} \left( \frac{\partial L^{(i)}}{\partial \dot{q}} \right) - \frac{\partial L^{(i)}}{\partial q} = Y^{(i)} \theta^{(i)}, \quad 1 \leq i \leq n + 1. \tag{14}
\]

In the next section, (14) will be used to obtain a formula of \( L^{(i)} \).

Once \( Y \) is computed, \( Y_k \) and \( Y_u \) in (4) can readily be found as follows. Vector \( \theta \) in (12) can be regrouped as in (3), where \( \theta_u \) is formed by collecting the entries, each of which involves at least one of the unknown parameters, and \( \theta_k \) is simply the complement of \( \theta_u \) in \( \theta \). Obviously, this regrouping can be done by premultiplying \( \theta \) by an elementary transformation matrix \( T \), that is,

\[
T\theta = \left[ \begin{array}{c} \theta_k \\ \theta_u \end{array} \right]_{(r_1 + r_2) \times 1}. \tag{15}
\]

From

\[
Y \theta = YT^T T \theta = YT^T \left[ \begin{array}{c} \theta_k \\ \theta_u \end{array} \right] \tag{16}
\]

it follows that

\[
Y_k = \text{first } r_1 \text{ columns of } YT^T, \tag{17}
\]

and

\[
Y_u = \text{last } r_2 \text{ columns of } YT^T. \tag{18}
\]
where the kinetic energy of link $i$ is given by
\[
k^{(i)} = \frac{1}{2} m \| \dot{v}_{ci} \|^2 + \frac{1}{2} \omega_i^T \]
(23)
with
\[
\dot{v}_{ci} = J_o \dot{q} + J_w \dot{q} \times \dot{R} p_{ci},
\]
(24)
and
\[
\omega_i = \dot{R} J_w \dot{q},
\]
(25)
and the potential energy of the link is given by
\[
u^{(i)} = -m \cdot \dot{p}_{ci} \cdot g
\]
(26)
with $\dot{g}$ the gravity vector and $p_{ci}$ the position vector from the origin of frame $\{0\}$ to the link's mass center, as is shown in Fig. 1.

From (23)-(25) it follows that
\[
\frac{\partial k^{(i)}}{\partial \dot{q}} = \frac{\partial \nu^{(i)}}{\partial \dot{q}} + J_o^T R I_o \dot{R} J_w \dot{q},
\]
(27)
with $\nu^{(i)}$ given by (24), (126) can be used to express the first term in (27) as $W_1(q, \dot{q}) \theta_1$ with $W_1(q, \dot{q})$ defined by (127) and $\theta_1$ defined by (125). Furthermore, let
\[
d = \dot{R} J_w \dot{q} = [d_1, d_2, d_3]^T.
\]
(28)

Formula (107) gives
\[
J_o^T R I_o \dot{R} J_w \dot{q} = J_o^T R B(d) \theta_3 \equiv W_2(q, \dot{q}) \theta_3,
\]
(29)
where $\theta_3$ is defined by (106) and $B(d)$ is given by (108). Equation (27) can now be written as
\[
\frac{\partial k^{(i)}}{\partial \dot{q}} = [W_1(q, \dot{q}) W_2(q, \dot{q})] \theta^{(i)}
\]
(30)
with
\[
\theta^{(i)} = \begin{bmatrix} \theta_2 \\ \theta_3 \end{bmatrix}.
\]
(31)

By (23)-(25) the second term in (22) becomes
\[
-\frac{\partial k^{(i)}}{\partial \dot{q}} = -m \frac{\partial v_{ci}^T}{\partial \dot{q}} v_{ci} - \frac{\partial (\dot{q}^T J_o^T R) \dot{R} J_w \dot{q}}{\partial \dot{q}}.
\]
(32)
By repeatedly using (19) and (20), it is found that
\[
-\frac{\partial v_{ci}}{\partial \dot{q}} = [Y_{11}(q, \dot{q}) Y_{12}(q, \dot{q}) Y_{13}(q, \dot{q}) Y_{14}(q, \dot{q})] \theta_3
\equiv Y_1(q, \dot{q}) \theta_3,
\]
(33)
where
\[
Y_{11}(q, \dot{q}) = -D_o^T J_o R
\]
(34)
\[
Y_{12}(q, \dot{q}) = [-D_o \times J_o \dot{q} + J_w \dot{q} \times D_o R] J_o^T R - S
\]
(35)
\[
Y_{13}(q, \dot{q}) = \left[ \frac{\partial J_o}{\partial q_1} J_o \dot{q} \cdots \frac{\partial J_o}{\partial q_n} J_o \dot{q} \right]^T \dot{q}
\]
(36)
\[
D_o = \left[ \frac{\partial J_o}{\partial q_1} J_o \dot{q} \cdots \frac{\partial J_o}{\partial q_n} J_o \dot{q} \right], \quad D_w = \left[ \frac{\partial J_w}{\partial q_1} J_w \dot{q} \cdots \frac{\partial J_w}{\partial q_n} J_w \dot{q} \right]
\]
(37)
\[
S = \left[ \frac{\partial (j_o R)}{\partial q_1} (J_o \dot{q} \times J_o \dot{q}) \cdots \frac{\partial (j_o R)}{\partial q_n} (J_o \dot{q} \times J_o \dot{q}) \right]
\]
(38)
and $Y_{14}(q, \dot{q})$ is determined by
\[
\begin{bmatrix}
\dot{p}_{ci}^T E_i p_{ci} \\
\vdots \\
\dot{p}_{ci}^T E_n p_{ci}
\end{bmatrix} = Y_{14}(q, \dot{q}) \theta_4,
\]
(39)
where \( \theta_4 \) is defined by (112), and
\[
E_j = \frac{1}{2} \dot{R} \dot{L}_w \dot{q}^T \frac{\partial J_o}{\partial q} R - \frac{1}{2} \dot{R} \frac{\partial J_o}{\partial q} R + J_o \ddot{q}^T J_o^T \ddot{R}.
\] (40)

With \( d \) defined by (28) and (107), the second term in (32) can be written as
\[
- \frac{\partial (\dot{q}^T J_o^T \ddot{R})}{\partial \dot{q}} \delta I_o \delta R L_w = Y_2(q, \dot{q}) \theta_4
\] (41)

where
\[
Y_2(q, \dot{q}) = - \left[ \frac{\partial (\delta R L_w)}{\partial q_1} \cdots \frac{\partial (\delta R L_w)}{\partial q_n} \dot{q} \right] B(d).
\] (42)

To compute the last term in (22), note that
\[
\dot{q}^T \frac{\partial \phi^T R}{\partial \dot{q}} = \left( \frac{\partial \phi^T R}{\partial \dot{q}} \right)^T \dot{q}
\]
which, in conjunction with (24) and (99), leads to
\[
- m_R R \frac{\partial \phi^T R}{\partial \dot{q}} = q^T \dot{Y}_2(q) \theta_2,
\]
where \( \theta_2 \) is defined by (101) and
\[
\begin{align*}
Y_2(q) &= [Y_{21}(q) \quad Y_{22}(q)] \\
Y_{21}(q) &= -J_o \dot{q}^T g \\
Y_{22}(q) &= (J_o \times \dot{q})^T g R.
\end{align*}
\]

Using (14), (22), (30), (32), (33), (41), and (43), we obtain
\[
Y^{(i)} = [W_1(q, \dot{q}) + \dot{Y}_1(q, \dot{q}) \quad W_2(q, \dot{q}) + Y_2(q, \dot{q})]
\] (47)

where
\[
\dot{Y}_1(q, \dot{q}) = [Y_{11} + Y_{31} \quad Y_{12} + Y_{32} \quad Y_{13} + Y_{34}].
\] (48)

Remarks

1) From (47) and (31) it is observed that, although there are only ten physical parameters involved in a link or load, in general, the dimension of \( \theta^{(i)} \) is 16. This parameter redundancy, presented in vector \( \theta_4 \) which is defined by (125), appears to be necessary to reformulate the dynamics so that a linear appearance of vector \( \theta^{(i)} \) in the dynamics is achieved. Conversely, if the link (load) has a regular geometry, the dimension of \( \theta^{(i)} \) will very likely be reduced, leading to a simplified solution. Additional discussion on this issue will be given in the next subsection.

2) Another feature of (47) is that the formula as it stands is suitable to serve as a starting point to derive a closed-form solution for the so-called filtered regressed system that has been used in several globally stable adaptive control algorithms [7, 8]. This is due to the fact that in (47), \( Y^{(i)} \) depends implicitly on \( \dot{q} \) through the time derivative of \( W_1 \) and \( W_2 \).

B. Special Cases

1) Point-Mass Link (or Load): If the link (or load) can be treated as a point mass, then \( \theta^{(i)} = \theta_3 \), which is defined by (125). Consequently, \( Y^{(i)} \) is given by
\[
Y^{(i)} = W_1(q, \dot{q}) + \dot{Y}_1(q, \dot{q}).
\] (49)

2) When Axes of \( c_i \) are the Principal Axes of the Link: If link \( i \) has a regular geometry, such that the axes of \( c_i \) coincide with the principal axes of the link, that is,
\[
c_i = \text{diag}(I_{xx}, I_{yy}, I_{zz})
\] (50)

and if the mass center of the link lies on the \( x \) or \( z \) axis of frame \( c_i \), then \( \theta^{(i)} \) is reduced to a six-dimensional vector of the form shown in (51), at the bottom of the page, and expressions for matrices \( W_1, W_2, Y_1, \) and \( Y_2 \) can be simplified considerably.

For \( p_{ci} = [p_x, 0, 0]^T \), matrix \( W_1(q, \dot{q}) \in \mathbb{R}^{3 \times 3} \) becomes
\[
W_1(q, \dot{q}) = \begin{bmatrix}
J_{xx} \dot{J}_{xx} - J_{xy} \dot{J}_{xy} - J_{xz} \dot{J}_{xz} & J_{yy} \dot{J}_{yy} - J_{yx} \dot{J}_{yx} - J_{yz} \dot{J}_{yz} & J_{zz} \dot{J}_{zz} - J_{zx} \dot{J}_{zx} - J_{zy} \dot{J}_{zy} \\
\end{bmatrix}^T R
\] (52)

where \( \dot{r}_1 \) denotes the first column of \( \dot{q} R \), and \( b \) is defined by \( b = [b_1 \cdots b_n]^T \) with
\[
b_{j} = \begin{bmatrix}
\dot{r}_1^T J_{xj} \dot{J}_{xj} & \dot{r}_1^T J_{yj} \dot{J}_{yj} & \dot{r}_1^T J_{zj} \dot{J}_{zj}
\end{bmatrix}^T, \quad 1 \leq j \leq n.
\] (53)

Matrix \( W_2(q, \dot{q}) \) defined by (29) becomes a \( 3 \times 3 \) matrix with \( B(d) \in \mathbb{R}^{3 \times 3} \) given by
\[
B(d) = \text{diag}(J_{xx}, J_{yy}, J_{zz})
\] (54)

that is, the \( 3 \times 3 \) diagonal matrix with \( \delta R L_w \) as entries along its main diagonal. Matrix \( Y_1 \) becomes
\[
Y_1 = [Y_{11} + Y_{31} \quad Y_{12} + Y_{32} \quad Y_{13} + Y_{34}]
\] (55)

where
\[
\begin{align*}
y_{11} &= -\left( J_{xx} \times \dot{g} \right)^T \dot{r}_1 \\
y_{12} &= -(J_{yy} \times \dot{g})^T \dot{r}_1 \\
y_{13} &= -(J_{zz} \times \dot{g})^T \dot{r}_1 \\
y_{14} &= [e_{11} \cdots e_{1n}]^T
\end{align*}
\] (56)

and \( Y_2 \in \mathbb{R}^{n \times 3} \) is given by (42) with \( B(d) \) given by (54).

For \( p_{ci} = [0, 0, 0] \), a simplified formula can also be established that is almost identical to (52)-(59) except that \( \dot{r}_1 \) should be replaced by \( \dot{r}_3 \)—the third column of \( \dot{q} R \).

C. Example

Consider the 2-DOF planar arm shown in Fig. 2, where the length of link \( i \) is denoted by \( l_i \). It is assumed that the links are of uniform mass type, with each mass center at the origin of its link frame. The robot handles a rectangular bar with uniform material density. The parameters of the load include mass \( m_2 \), mass center \( p_{c2} \), and inertia tensor \( I_{c2} \), where
\[
p_{c2} = [p_x \ 0 \ 0] \quad \text{and} \quad I_{c2} = \text{diag}(I_{xx}, I_{yy}, I_{zz}).
\] (51)

For comparison, the manipulator regressor will be evaluated using the conventional approach described in Section I and the approach proposed in Section III.
Conventional Approach: As mentioned earlier, there are two steps that need to be carried out in this indirect approach.

Step 1—Establishing the Robot Dynamics: Following [14], we compute

\[
H(q) = \sum_{i=1}^{3} m_i (J_i^{(1)})^T \ddot{J}_i^{(1)} + (J_i^{(2)})^T \ddot{q} I \omega_i^{(i)},
\]

where \( J_i^{(1)} \) and \( J_i^{(2)} \) form the link Jacobian for link \( i \) that relates the joint velocity to the velocity of the frame \( \{c_i\} \) which is obtained by translating \( \{i\} \) to the mass center of link \( i \). In our case, \( \{c_1\} = \{1\}, \{c_2\} = \{2\}, \) and \( a_1 = a_2 = 0 \), so \( J_1^{(1)} = J_2^{(1)} = J_3^{(1)} \), \( J_2^{(2)} = J_3^{(2)} \), where

\[
J_1^{(1)} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad J_2^{(2)} = \begin{bmatrix} -l_1 s_1 \theta_1 \\ -l_1 c_1 \theta_1 \end{bmatrix}.
\]

Since the linear velocity of \( \{c_2\} \) is

\[
v_{c_2} = [ -l_2 s_1 \theta_1 - (l_2 + p_x) s_2 (\dot{\theta}_1 + \dot{\theta}_2),
- l_1 c_1 \theta_1 + (l_2 + p_x) c_2 (\dot{\theta}_1 + \dot{\theta}_2),]
\]

we obtain

\[
J_2^{(3)} = \begin{bmatrix} -l_1 s_1 - (l_2 + p_x) s_2 \\ l_1 c_1 + (l_2 + p_x) c_2 \end{bmatrix}.
\]

Furthermore, notice that

\[
J_2^{(3)} \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}
\]

hence the equation at the bottom of the page results. Denoting \( H(q) = (H_3, \ldots, H_1) \), it follows from [14] that the term \( C \dot{q} \) in (1) is

\[
C(q, \dot{q}) \dot{q} = \begin{bmatrix} \sum_{j=1}^{2} \left( \frac{\partial H_{1j}}{\partial q_1} - \frac{\partial H_{1j}}{\partial q_1} \right) \dot{q}_1 \\ \sum_{j=1}^{2} \left( \frac{\partial H_{2j}}{\partial q_1} - \frac{\partial H_{2j}}{\partial q_1} \right) \dot{q}_1 \\ \sum_{j=1}^{2} \left( \frac{\partial H_{3j}}{\partial q_1} - \frac{\partial H_{3j}}{\partial q_1} \right) \dot{q}_1 \end{bmatrix}
= \begin{bmatrix} -l_1 (l_2 + p_x) \theta_1 m_2 \theta_1 (2 \dot{\theta}_1 + \dot{\theta}_2) \\ \theta_1 (l_2 + p_x) m_2 \theta_1 (2 \dot{\theta}_1 + \dot{\theta}_2) \end{bmatrix}.
\]

Finally, we compute the total potential energy of the system as

\[
U = m_2 g l_1 \theta_1 + m_3 g [l_2 s_2 (l_2 + p_x) \theta_2] \]

hence

\[
G = \begin{bmatrix} \frac{\partial U}{\partial q_1} \\ \frac{\partial U}{\partial q_2} \end{bmatrix} = \begin{bmatrix} (m_2 + m_3) l_1 g c_1 + m_3 g (l_2 + p_x) c_1 c_2 \\ m_3 g (l_2 + p_x) c_1 c_2 \end{bmatrix}
\]

Step 2—Parameter Extraction: As mentioned in Section 1, this step begins with defining a parameter vector \( \theta \). Suppose that the parameters of interest are link mass \( m_i \) (note that the link mass \( m_1 \) plays no role in the dynamics), and the parameters associated with the load, that is, \( m_3, p_x, \) and \( I_z \). From what we have done in Step 1, it is observed that every entry of \( H, C, \) and \( G \) involves some of these parameters. In addition, in a number of entries, parameters present themselves in a nonlinear manner such as \( m_3 p_x \). It takes a while to figure out that

\[
\theta = \begin{bmatrix} m_2 & m_3 & m_3 p_x & m_3 p_x^2 & I_z \end{bmatrix}^T
\]

defines a parameter vector with minimum dimension such that for every entry of \( H, C, \) and \( G \), the parameters of interest can all be extracted. Note that one does not need to include \( p_x \) as a single component in \( \theta \), as parameter \( p_x \) in all the terms involved always presents itself together with \( m_3 \). Obviously, such a parameter vector cannot be defined adequately before a careful inspection of all entries is completed. For a robot with more non-point-mass joints, this step will become quite involved and time-consuming.

With \( \theta \) as defined earlier, we can now extract it from the entries of \( H, C, \) and \( G \) as

\[
H_{11} = \begin{bmatrix} l_1^2 + 2 l_1 l_2 c_2 + l_2^2 + 2 (l_1 c_2 + l_2) \theta_1 \theta_2 + 1 & 0 \\ 0 & l_2 \theta_1 + l_1 c_2 \theta_2 + 2 l_2 \theta_1 \theta_2 + 1 \end{bmatrix} \equiv h_{11} \theta,
\]

\[
H_{12} = \begin{bmatrix} 0 & l_1 l_2 c_2 + 2 l_2 \theta_1 \theta_2 + 1 \\ l_1 c_2 \theta_2 + 2 l_2 \theta_1 \theta_2 + 1 \end{bmatrix} \equiv h_{12} \theta,
\]

\[
H_{22} = \begin{bmatrix} l_2^2 + 2 l_2 \theta_1 \theta_2 + 1 \theta_1 \theta_2 + 2 l_2 \theta_1 \theta_2 + 1 \theta_1 \theta_2 + 1 \theta_1 \theta_2 + 1 \theta_1 \theta_2 \end{bmatrix} \equiv h_{22} \theta,
\]

\[
C(q, \dot{q}) \dot{q} = \begin{bmatrix} 0 & -l_1 (l_2 + p_x) (2 \dot{\theta}_1 + \dot{\theta}_2) \\ 0 & l_1 (l_2 + p_x) \theta_1 (2 \dot{\theta}_1 + \dot{\theta}_2) \\ 0 & l_1 (l_2 + p_x) \theta_1 (2 \dot{\theta}_1 + \dot{\theta}_2) \\ 0 & l_1 (l_2 + p_x) \theta_1 (2 \dot{\theta}_1 + \dot{\theta}_2) \end{bmatrix}
\]

\[
G = \begin{bmatrix} l_1 c_1 g & l_1 c_1 c_2 g & l_1 c_2 g & l_1 c_2 g & l_2 c_2 g \end{bmatrix} \Theta \equiv Y_c \theta.
\]

Therefore, the manipulator regressor associated with \( \theta \) is given by (61), at the bottom of the next page.
Proposed Approach: The problem of computing $Y^{(3)}(\dot{q}, \ddot{q}, \dddot{q})$ falls obviously within the special case addressed in Section III-B.2. Using

$$J_2(q) = \begin{bmatrix} -l_1 s_1 - l_2 s_2 & -l_2 s_1 \\ l_1 c_1 + l_2 c_2 & l_2 c_1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

and

$$J_3(q) = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

(52) becomes the equation shown at the bottom of the previous page. Since $B(d) = \text{diag}(0, 0, \dot{q}_1 + \dot{q}_2)$, (29) gives

$$W_2(q, \dot{q}) = \begin{bmatrix} 0 & \dot{q}_1 + \dot{q}_2 \\ 0 & 0 \end{bmatrix}.$$  

By (34), (45), and (56)–(59), we obtain $\dot{Y}_1$ with

$$Y_{11} + \dot{Y}_{21} = \begin{bmatrix} (l_1 c_1 + l_2 c_1) \dot{q}_1 \\ l_1 s_2 \dot{q}_1 (\dot{q}_1 + \dot{q}_2) + l_2 c_2 \dot{q}_2 \end{bmatrix},$$

$$\ddot{Y}_{11} + \ddot{Y}_{21} = \begin{bmatrix} l_1 s_2 \ddot{q}_1 (\dot{q}_1 + \dot{q}_2) + l_2 c_2 \ddot{q}_2 \\ \ddot{q}_1 + \ddot{q}_2 \end{bmatrix},$$

$$\dddot{Y}_{11} = \begin{bmatrix} l_1 s_2 \dddot{q}_1 (\dot{q}_1 + \dot{q}_2) + l_2 c_2 \dddot{q}_2 \\ \dddot{q}_1 + \dddot{q}_2 \end{bmatrix}.$$

Furthermore, by (42) it is found that $Y_2(q, \dot{q})$ is a $2 \times 3$ matrix. Since $Y_2$ and the first two columns of $W_2$ are all zero, $\theta^{(3)}$ can be redefined as

$$\theta^{(3)} = [m_3 m_4 p_x m_5^2 I_{1x}]^T$$

and $Y^{(3)}$ is now obtained as (63), shown at the bottom of the page. To compute $Y^{(1)}$ and $Y^{(2)}$, note that each link is of point-mass involving only one parameter—its link mass. The link Jacobians for link 1 and 2 are given by (60), and (49) gives

$$Y^{(1)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

$$Y^{(2)} = \begin{bmatrix} l_1 \dddot{q}_1 + l_1 c_1 \dot{q}_1 \\ 0 \end{bmatrix}.$$  

Since the link mass $m_1$ does not play a role in the dynamics, $Y_1$ (the zero column) can be dropped in $Y$, that is

$$Y = [Y^{(2)} Y^{(3)}].$$  

With $\theta = [m_2 m_3 m_4 p_x m_5^2 I_{1x}]^T$.

On comparing these two approaches and their results, namely (63)–(65) and (61), it is observed that the proposed approach finds the regressor by directly and, therefore, more efficiently manipulating the link Jacobians. As our method is evolved from Lagrange's equation of motion, it may be viewed as a counterpart of the Lagrangian formulation for the evaluation of regressor dynamics.

IV. Iterative Computation of $Y(q, \dot{q}, \ddot{q})$

We begin our analysis by reformulating the Newton–Euler recursion [9] using the techniques developed in Appendix I. For the sake of simplicity, all joints are assumed to be revolute. Notation adopted below will be consistent with those used in [6, chap. 6].

A. Outward Iterations ($i: 0 \rightarrow n$)

By (104), the outward iterations [6, p. 200] imply that

$$i^{+1} F_{i+1} = A_{i+1} \theta^{(i+1)}$$

where $\theta^{(i+1)}$ is defined by $\theta_2$ in (101) with the understanding that $m_{i+1}$ is the mass of link $i + 1$, $F_{i+1} = [p_x p_y p_z]^T$, and

$$A_{i+1} = [i^{+1} \dot{t}_{i+1} H_{i+1}]$$

$$H_{i+1} = \Omega(i^{+1} \dot{t}_{i+1}) + i^{+1} \omega_{i+1}^T I$$

$$i^{+1} U_{i+1} = i^{+1} \omega_{i+1} + i^{+1} t_{i+1}^T.$$

By (107), we obtain

$$i^{+1} N_{i+1} = E_{i+1} \theta^{(i+1)}$$

By (108) and (109), we obtain

$$i^{+1} F_{i+1} = A_{i+1} \theta^{(i+1)}$$

where $\theta^{(i+1)}$ is defined by $\theta_2$ in (101) with the understanding that $m_{i+1}$ is the mass of link $i + 1$, $F_{i+1} = [p_x p_y p_z]^T$, and

$$A_{i+1} = [i^{+1} \dot{t}_{i+1} H_{i+1}]$$

$$H_{i+1} = \Omega(i^{+1} \dot{t}_{i+1}) + i^{+1} \omega_{i+1}^T I$$

$$i^{+1} U_{i+1} = i^{+1} \omega_{i+1} + i^{+1} t_{i+1}^T.$$

By (107), we obtain

$$i^{+1} N_{i+1} = E_{i+1} \theta^{(i+1)}$$

where $\theta^{(i+1)}$ is defined by $\theta_2$ in (101) with the understanding that $m_{i+1}$ is the mass of link $i + 1$, $F_{i+1} = [p_x p_y p_z]^T$, and

$$A_{i+1} = [i^{+1} \dot{t}_{i+1} H_{i+1}]$$

$$H_{i+1} = \Omega(i^{+1} \dot{t}_{i+1}) + i^{+1} \omega_{i+1}^T I$$

$$i^{+1} U_{i+1} = i^{+1} \omega_{i+1} + i^{+1} t_{i+1}^T.$$
where $\theta^{(n+1)}_2$ is defined by (106) with $I_{xx}, \ldots, I_{zz}$ from $i_1 A_{n+1}$, and
\[
E_{n+1} = B^{(n+1)} \omega_{n+1} + \Omega^{(n+1)} \omega_{n+1} B^{(n+1)} \omega_{n+1}.
\] (70)

**B. Inward Iterations (i: n + 1 → 1)**

If no contact occurs between the load and its environment, we can begin the inward iteration with zero boundary conditions. By (66) and (69), the inward iterations [6, p. 200] imply that
\[
\begin{align*}
E_{n+1} & = A_{n+1} e^{(n+1)}_2 + \Omega^{(n+1)} e^{(n+1)}_2 B^{(n+1)} \omega_{n+1} \\
\omega_{n+1} & = \tilde{h}_n \Phi_1 + \Phi_1 \Phi_3 \omega_{n+1} + \Phi_1 \Phi_3 \Phi_1 \Phi_3 \omega_{n+1} + \cdots
\end{align*}
\] (71)

where
\[
\begin{align*}
& m_{n+1} = m_{n+1} + 1 \\
& \Phi_1 = \left[ \begin{array}{c}
\Phi_1 \\
\Phi_2 \\
\Phi_3 \\
\Phi_4 \\
\Phi_5
\end{array} \right]
\end{align*}
\] (72)

with $m_{n+1}$ the mass of link $n+1$ (i.e., the load) and $H_{n+1}$ is defined in (68). From (109)-(116) it follows that
\[
\begin{align*}
& m_{n+1} \omega_{n+1} + 1 \\
& \Phi_1 = -\Omega^{(n+1)} e^{(n+1)}_2 \omega_{n+1}
\end{align*}
\] (73)

with $\Phi_1$ defined by (114). Hence,
\[
\begin{align*}
& \Phi_1 = \left[ \begin{array}{c}
\Phi_1 \\
\Phi_2 \\
\Phi_3 \\
\Phi_4 \\
\Phi_5
\end{array} \right]
\end{align*}
\] (74)

where
\[
\Pi_{n+1} = [\Phi_1 \Phi_2 \Pi_{n+1}] \Phi_5 = [0 0 0] \tilde{A}_{n+1}, \quad \Phi_5 = [0 0 0] \tilde{A}_{n+1}
\] (75)

To obtain the iterative relation of $\Pi_{n+1}$ to $\Pi$, for $i = n, n-1, \ldots, 1$, first we write the vector $\theta$ in (12) as
\[
\theta = \begin{bmatrix}
\theta^{(n+1)}_2 \\
\theta^{(n+1)}_3 \\
\theta^{(n+1)}_4 \\
\theta^{(n+1)}_5 \\
\theta^{(n+1)}_6
\end{bmatrix}
\] (76)

and write (71) and (74) as

\[
\begin{align*}
E_{n+1} & = A_{n+1} \theta \\
\omega_{n+1} & = \tilde{h}_n \Phi_1 + \Phi_1 \Phi_3 \omega_{n+1} + \Phi_1 \Phi_3 \Phi_1 \Phi_3 \omega_{n+1} + \cdots
\end{align*}
\] (77)

with
\[
\begin{align*}
& \tilde{h}_n = [0 0 \cdots 0] \tilde{A}_{n+1} \theta \\
& \Phi_1 = [0 0 \cdots 0] \tilde{A}_{n+1} \theta
\end{align*}
\] (78)

where $\theta^{(n+1)}_2$ is placed in a position consistent with that of $\theta^{(n+1)}_2$ in (76), and $\Pi_{n+1}$ is placed in a position consistent with that of $\theta^{(n+1)}_6$. By (77) and (66), the inward iterations [6, p. 200] for $1 \leq i \leq n$ give
\[
\begin{align*}
& \theta^{(i+1)}_2 = i_{i+1} R \tilde{A}_{i+1} \theta + A_i \theta^{(i)}_2 \\
& \tilde{A}_{i+1} = i_{1+1} R \tilde{A}_{i+1} + [0 0 \cdots 0] A_i \theta^{(i)}_2
\end{align*}
\] (80)

with $A_i$ placed in a position consistent with that of $\theta^{(i)}_2$ in (76). Furthermore, by (69), (79), (72), (102), and (81), the inward iterations give
\[
\begin{align*}
& \theta^{(i+1)}_2 = i_{i+1} R \tilde{A}_{i+1} \theta + A_i \theta^{(i)}_2 \\
& \tilde{A}_{i+1} = i_{1+1} R \tilde{A}_{i+1} + [0 0 \cdots 0] A_i \theta^{(i)}_2
\end{align*}
\] (81)

with $A_i$ placed in a position consistent with that of $\theta^{(i)}_2$ (which is the last three components of $\theta^{(i)}_2$ and $\theta^{(i)}_6$, and $E_i$ placed in a position consistent with that of $\theta^{(i)}_6$, respectively.

We now obtain the manipulator dynamics as $\tau = Y(q, \dot{q}, \ddot{q}) \theta$, where
\[
Y(q, \dot{q}, \ddot{q}) = \begin{bmatrix}
\tilde{\Pi}_1 \\
\vdots \\
\tilde{\Pi}_n
\end{bmatrix}
\] (82)

with $\tilde{\Pi}_i$ defined by (84) for $1 \leq i \leq n$. Obviously, if joint $i$ is not revolute but prismatic, then the ith row of $Y$ in (85) should be $\tilde{A}_i$. In summary, the computation of $Y(q, \dot{q}, \ddot{q})$ in (85) can be accomplished by following the steps listed below.

**Algorithm for Computation of $Y(q, \dot{q}, \ddot{q})$**

**Step 1:** Compute $A_{n+1}$ using (67) and (68).

**Step 2:** Compute $E_{n+1}$ using (69) and (70).

**Step 3:** Compute $\Phi_{n+1}$ using (73).

**Step 4:** Form $\tilde{A}_{n+1}$ and $\tilde{\Pi}_{n+1}$ using (78) and (75), (80), respectively.

**Step 5:** For $i = n, n-1, \ldots, 1$,

- Compute $A_i$ using (67) and (68).
- Compute $E_i$ using (69) and (70).
- Compute $\Phi_i$ using (73).
- Compute $\tilde{A}_i$ using (82).
- Compute $\tilde{\Pi}_i$ using (84).

**Step 6:** Form $Y(q, \dot{q}, \ddot{q})$ where

\[
\begin{bmatrix}
\tilde{\Pi}_1 \\
\vdots \\
\tilde{\Pi}_n
\end{bmatrix}
\] (85)

The $i$th row of $Y(q, \dot{q}, \ddot{q})$ is

\[
\begin{bmatrix}
\tilde{\Pi}_i \\
\vdots \\
\tilde{\Pi}_n
\end{bmatrix}
\] (86)

if joint $i$ is revolute

\[
\begin{bmatrix}
\tilde{\Pi}_i \\
\vdots \\
\tilde{\Pi}_n
\end{bmatrix}
\] (87)

if joint $i$ is prismatic.

**C. Example**

We consider the robot used in Section III and perform the iterative algorithm to compute its regressor as follows.

Following the steps given in Section IV-B, we compute
\[
A_3 = \begin{bmatrix}
\beta_1 & \beta_2 & \beta_3 \\
\beta_1 & \beta_1 & \beta_1 \\
\beta_1 & \beta_1 & \beta_1 \\
\beta_1 & \beta_1 & \beta_1
\end{bmatrix}
\] (88)

where
\[
\begin{align*}
& \beta_1 = 1 + \eta(q_2 + \phi_2)^2 + \lambda(q_1 + \phi_2)^2 + \eta 2 \eta_2 \\
& \beta_2 = 1 + \eta(q_1 + \phi_2)^2 + \lambda(q_1 + \phi_2)^2 + \eta_2 2 \eta_2
\end{align*}
\] (89)

Because of the regular geometry of the load, we have $p_c = p_c = 0$, hence $\theta^{(3)}_2$ can be regarded as a two-dimensional vector
\[
\theta^{(3)}_2 = [m_3 m_3 p_c] T.
\] (90)

Consequently, the last two columns of matrix $A_3$ are not needed in the computation and $A_3$ can be redefined as
\[
A_3 = \begin{bmatrix}
\beta_1 & \beta_1 & \beta_1 \\
\beta_2 & \beta_1 & \beta_1 \\
0 & 0 & 0
\end{bmatrix}
\] (91)
Similarly, we compute
\[ E_3 = \begin{bmatrix} 0 \\ 0 \\ \dot{q}_3 + \dot{q}_2 \end{bmatrix} \]
with
\[ \theta_3 = I_{3x}. \]
Next note that
\[ \begin{bmatrix} \theta_3^{(3)} \\ \theta_4^{(3)} \end{bmatrix} = \begin{bmatrix} m_3p_x \\ m_3p_z \end{bmatrix} \]
hence, we only need the first and fourth columns of \( \Phi_3 \). By (73) it is found that
\[ \Phi_3 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ \beta_2 & \dot{q}_1 + \dot{q}_2 \end{bmatrix}. \]
Using (78), (75), and (80), we have
\[ \dot{A}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]
and
\[ \dot{\Pi}_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \]
Furthermore, using (84), it is found that
\[ \dot{\Pi}_2 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]
and
\[ \dot{\Pi}_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]
where
\[ \pi_{22} = \beta_2 \dot{q}_1 + \dot{q}_3 + c_2 \dot{q}_2. \]
\[ \pi_{23} = \beta_2 \dot{q}_1 + \dot{q}_2. \]
\[ \pi_{3} = (l_2 + c_2 \dot{q}_1 + \dot{q}_2) - l_1 \dot{q}_1 + \dot{q}_2. \]
The parameter vector associated with \( \Pi \) is
\[ \theta = \begin{bmatrix} m_2 \\ m_3 \\ m_3p_x \\ m_3p_z \\ I_{1x}. \end{bmatrix} \]
Using (85), the corresponding regressor \( Y(q, \dot{q}, \ddot{q}) \) is given by
\[ Y = \begin{bmatrix} \pi_{22} \\ \pi_{23} \\ \pi_{3} \\ 0 \end{bmatrix} \begin{bmatrix} \dot{q}_1 + \dot{q}_2 \\ \dot{q}_1 + \dot{q}_2 \\ \dot{q}_1 + \dot{q}_2 \end{bmatrix}. \]
Note that the \( Y \) obtained is identical to the one given by (65).

V. STABLE ADAPTIVE MOTION CONTROL AND ITS IMPLEMENTATION USING THE REGRESSOR

In the first part of this section, a modified version of the Slotine–Li adaptive scheme [5] is presented, and its global stability is shown through a Lyapunov approach. Unlike the algorithm in [5], the modified version enables us to use directly the regressor \( Y \) in controller implementation. Another adaptive motion control algorithm that uses \( Y \) directly was proposed by Craig et al. [10]. Contrary to the algorithm of [10], however, the approach proposed below does not require the use of joint acceleration measurements \( \ddot{q} \) and inversion of the estimated mass matrix. In the second part of this section, we present a case study that simulates the two adaptive motion control algorithms where the regressors encountered are evaluated using the iterative algorithm described in Section IV.

A. Modified Slotine–Li Adaptive Control Algorithm

Following the discussion in Section II-A, we define
\[ \dot{q}_r = \dot{q}_d - \Lambda \ddot{q} \]
with \( \dot{q}_d \) is the desired velocity, \( \ddot{q} = q - q_d \), and \( \Lambda > 0 \), and note that
\[ \ddot{q} \dot{q}_r + C \ddot{q}_r + G = Y_n(q, \dot{q}, \ddot{q}) \dot{\theta}_u. \]
(87)
If a control torque vector \( \tau \) is assigned as
\[ \tau = \ddot{q} \dot{q}_r + C \ddot{q}_r + G - K \dot{s} \]
with \( \dot{s} = \dot{q} - \dot{q}_r \) and if \( \dot{\theta}_u \) is updated according to
\[ \dot{\theta}_u = -\Gamma Y^T_n(q, \dot{q}, \ddot{q}) \dot{\theta}_u, \]
then for any \( \dot{s}(0), \dot{\theta}_u(0) \), and a bounded \( \dot{s}(t) \), there exists a \( K \) such that the position and velocity tracking errors converge to zero. To show this, consider the Lyapunov function
\[ v = \frac{1}{2} \dot{s}^T H \dot{s} + \dot{\theta}_u^T \Gamma^{-1} \dot{\theta}_u \]
(90)
and compute its time derivative along trajectories of (1) as
\[ \dot{v} = \dot{s}^T (H \ddot{q} - H \dot{q}_r) + \dot{\theta}_u^T \Gamma^{-1} \ddot{\theta}_u + \frac{1}{2} \dot{s}^T \dot{H} \dot{s} \]
\[ = \dot{s}^T (\tau - C \ddot{q}_r - G - H \dot{q}_r) + \dot{\theta}_u^T \Gamma^{-1} \dot{\theta}_u \]
where the fact that \( \dot{H} = 2C \) is skew symmetric has been used. If control (88) and parameter update law (89) are employed, then
\[ \dot{v} = -s^T [K - C(q, \dot{q})] s \]
\[ = -s^T [K - \tilde{C}(\dot{q}, \ddot{q})] s \]
(91)
where \( \tilde{C}(\dot{q}, \ddot{q}) = [C(q, \dot{q}) + \tilde{C}(q, \dot{q})]/2 \) is a symmetric matrix. By (91) in conjunction with the same argument as was adopted in [5] and [11], it can be shown to be both \( \dot{s} \) and \( \dot{\theta}_u \) converge to zero.

Concerning the algorithm implementation, note that generating a new control torque vector requires that \( Y_n(q, \dot{q}, \ddot{q}) \) be evaluated to update parameter vector \( \dot{\theta}_u \) in (89), and then \( Y(q, \dot{q}, \ddot{q}) \) be computed to obtain \( \tau \) as (88) can be written in the form
\[ \tau = Y(q, \dot{q}, \ddot{q}) \begin{bmatrix} \theta_u \\ \dot{\theta}_u \end{bmatrix} - K \dot{s}. \]
(92)
As is noted in [5], if one chooses \( \Lambda = I \) in (86) and \( K = \lambda \dot{H}(q) \) in (88), then the time derivative of \( v \) in (90) along trajectories of (1) is given by
\[ \dot{v} = -s^T [\lambda H(q) - \tilde{C}(q, \dot{q})] s. \]
(93)
provided that the unknown parameters are updated according to
\[ \dot{\theta}_u = -Y^T_n(q, \dot{q}, \ddot{q}) (\dot{q} - \lambda \ddot{q}) s. \]
(94)
Since \( H(q) \) is uniformly positive definite, \( \dot{v} \) in (93) is negative if \( \lambda \) is sufficiently large. Note that the control torque in this case becomes
\[ \tau = Y(q, \dot{q}, \ddot{q}) \begin{bmatrix} \theta_u \\ \dot{\theta}_u \end{bmatrix} \]
(95)
where
\[ \dot{q}^* = \dot{q}_d - 2\lambda \ddot{q} - \lambda^2 \ddot{q}. \]
(96)
Obviously, (95) and (96) represent a quasi-computed-torque controller, which would be identical to the well-known computed-torque algorithm [6] if \( \dot{\theta}_u = \theta_u \).
B. Case Study

In what follows, we consider a 4-DOF manipulator, shown in Fig. 3, with a geometry identical to the first four links of a PUMA 500 robot. Its dynamic parameters are, however, simplified as follows: all four links are of point-mass with \( m_1 = m_2 = m_3 = 2 \) kg and \( m_4 = 0.5 \) kg. For the first three links, the mass center of each link is at its midpoint, and the mass center of link 4 is at its distal end. The robot carries a 0.5-kg point-mass load, tracking in the joint space a trajectory specified by

\[
\begin{bmatrix}
q_1(t) \\
q_2(t) \\
q_3(t) \\
q_4(t)
\end{bmatrix}
= \begin{bmatrix}
c_1 + a_1 \sin(\omega t) \\
c_2 + a_2 \cos(\omega t/2) \\
c_3 + a_3 \sin(\omega t/3) \\
c_4 + a_4 \cos(\omega t/4)
\end{bmatrix}
\]

with \( c_1 = 8\pi/9, c_2 = 2\pi/3, c_3 = 3\pi/4, c_4 = 15\pi/18, a_1 = 4\pi/9, a_2 = \pi/3, a_3 = \pi/4, \) and \( a_4 = 8\pi/18 \) for \( 0 \leq t \leq 2 \) s. Under these circumstances, the robot can be treated as if \( m_4 = 1 \) kg and carrying no load.

Now assume that the user does not know the load mass and makes an initial guess of \( m_4 = 0.5 \) kg. The initial robot configuration is set with 20% relative error in joint displacement for each joint. Both the modified Slotine-Li algorithm and the algorithm of Craig et al. [10] are applied to control the robot motion, and their tracking errors as well as parameter estimation errors are shown in Figs. 4 and 5. The regressors \( Y(q, \dot{q}, q^*), Y_0(q, \dot{q}, q - \lambda s) \) in the modified Slotine-Li algorithm, and the regressor \( Y_a(q, \dot{q}, \ddot{q}) \) in Craig's algorithm are evaluated using the iterative algorithm proposed in Section IV. From the figures, it is observed that it takes less than 0.3 s for the modified Slotine-Li algorithm and about 0.35 s for Craig's algorithm to see the tracking error converging to an acceptable tolerance. Both algorithms are able to identify the unknown mass of the load, but the modified Slotine-Li algorithm can do it quicker. As mentioned earlier, the control algorithm in Section V-A is computationally more efficient as compared with the algorithm of [10]. In addition, our simulation experience indicates that parameters \( \lambda \) and \( \Gamma \) in the modified Slotine-Li algorithm are less sensitive than their counterparts in Craig's algorithm, which is consistent with the observations made in a force control study [12].

An issue that is often encountered in simulation of robot dynamics is the computation of \( \dot{q} \) given \( q, \dot{q}, \) and \( \tau \). The steps listed below form a regressor version of an approach suggested in [13] for computing \( \dot{q} \).

Fig. 4. Simulation of modified Slotine-Li algorithm. (a) Actual and ideal joint-space trajectories. (b) Tracking errors. (c) Estimated \( m_4 \).

### Evaluation of \( \dot{q} \) in Simulations

**Step 1:** Compute \( \dot{q} = Y(q, \dot{q}, 0) \Theta \equiv Y_0 \Theta \).

**Step 2:** Compute \( H(q)e_i + \dot{q} = Y(q, \dot{q}, e_i) \Theta \equiv Y_i \Theta \), for \( 1 \leq i \leq n \).

**Step 3:** Compute \( H(q) = [Y_1(q) \Theta \cdots (Y_n - Y_0) \Theta] \).

**Step 4:** \( \ddot{q} = H^{-1}(q)(\tau - Y_0 \Theta) \).

where \( e_i \) is the \( i \)-th column of the identity matrix. By replacing \( \Theta \) in the first three steps with \( [\theta_k^2, \theta_k^3]^T \), \( H(q) \) in Craig's algorithm can...
APPENDIX I
PARAMETER EXTRACTION VIA VECTOR ANALYSIS TECHNIQUES

Throughout this appendix we assume that $m$ is in $R^{1 \times 1}$, $p = [p_x p_y p_z]^T$, $d = [d_1 d_2 d_3]^T$, and $z = [x_1 x_2 x_3]^T$.

1) Consider

$$m(J_w \dot{q} \times Rp)^T g$$

where $J_w \in R^{3 \times n}$, $R \in R^{3 \times 3}$ and $g \in R^{3 \times 1}$ are given.

Denoting

$$\theta_1 = [mp_x \ mp_y \ mp_z]^T$$

and applying (19) to (97), we have

$$m(J_w \dot{q} \times Rp)^T g = \dot{q}^T Y_c \theta_1$$

where

$$Y_c = [g \times J_{w1} \ g \times J_{w2} \ldots \ g \times J_{wn}]^T R$$

is an $n \times 3$ known matrix.

2) Consider the vector given by

$$m[\dot{\omega} \times p + \omega \times (\omega \times p) + \dot{v}]$$

where $m$ and $p$ are parameters to be extracted. Define

$$\theta_2 = [m \ mp_x \ mp_y \ mp_z]$$

and note that

$$\dot{\omega} \times p = \Omega(\dot{\omega})p,$$

where $\Omega(\dot{\omega})$ is a skew symmetric matrix characterized by

$$\Omega(x) = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}.$$  

From (20) it follows that

$$\omega \times (\omega \times p) = (p^T \omega)\omega - (\omega^T \omega)p = [U - ||\omega||^2 I]p,$$

where $U = \omega^T \omega$. Hence

$$m[\dot{\omega} \times p + \omega \times (\omega \times p) + \dot{v}] = [\dot{\omega}^T \Omega(\dot{\omega}) + U - ||\omega||^2 I] \theta_2.$$  

3) Let

$$\begin{bmatrix} I_{xx} & -I_{xy} & -I_{xz} \\ -I_{yx} & I_{yy} & -I_{yz} \\ -I_{zx} & -I_{zy} & I_{zz} \end{bmatrix}$$

be the unknown inertia matrix. Define

$$\theta_3 = [I_{xx} \ I_{yy} \ I_{zz} \ I_{xy} \ I_{xz} \ I_{yx}].$$

It can readily be verified that

$$cI_d = B(d)\theta_3,$$

where

$$B(d) = \begin{bmatrix} d_1 & 0 & 0 & -d_2 & -d_3 & 0 \\ 0 & 0 & d_2 & 0 & -d_3 & 0 \\ 0 & d_3 & 0 & -d_1 & 0 & -d_2 \end{bmatrix}$$

VI. CONCLUSION

An attempt has been made to derive a closed-form solution as well as iterative algorithm for symbolic and numerical evaluation of the manipulator regressor. Similar to the Lagrangian and Newton–Euler formulations of robot dynamics, the closed-form solution obtained also be evaluated using the first three steps.
4) Consider the vector product

$$m_p \times (\vdot + H p)$$

(109)

where $H$ is a $3 \times 3$ known matrix. By (102), the first vector product in (109) is

$$m_p \times \vdot = -\Omega(\vdot)\theta_2.$$ 

If we write

$$H = \begin{bmatrix} h_1^T \\ h_2^T \\ h_3^T \end{bmatrix}$$

and define

$$\Psi = \begin{bmatrix} 0 & h_2^T & -h_3^T \\ -h_2^T & 0 & h_1^T \\ h_3^T & -h_1^T & 0 \end{bmatrix} \in \mathbb{R}^{9 \times 1}$$

then

$$m_p \times H p = \begin{bmatrix} h_2^T (mp_x p) - h_3^T (mp_y p) \\ h_1^T (mp_x p) - h_3^T (mp_y p) \\ h_2^T (mp_y p) - h_1^T (mp_x p) \end{bmatrix} = \Psi \begin{bmatrix} mp_x p \\ mp_y p \\ mp_y p \end{bmatrix} \in \mathbb{R}^{9 \times 1}.$$ 

By defining

$$\theta_4 = m[\begin{bmatrix} p_x & p_y & p_z & p_x p_y & p_x p_z & p_y p_z \end{bmatrix}]^T$$

(111) can be written as

$$m_p \times H p = \Psi \theta_4,$$

(113)

where

$$\Psi = [\begin{bmatrix} e_1 & e_5 & e_9 \\ e_2 + e_4 & e_3 & e_6 + e_8 \end{bmatrix}]$$

and $e_i$ is the $i$th column of the $9 \times 9$ identity matrix. Hence

$$m_p \times (\vdot + H p) = \Psi \theta_4,$$

with

$$\Phi = [-\Omega(\vdot) \ \Psi].$$

(116)

5) Finally, let us consider

$$\begin{bmatrix} \frac{\partial}{\partial q} (J_\vdot \hat{q} + J_\omega \hat{q} \times \hat{q} R p) \end{bmatrix}^T (J_\vdot \hat{q} + J_\omega \hat{q} \times \hat{q} R p).$$

(117)

By (102)

$$\frac{\partial}{\partial q} (J_\vdot \hat{q} + J_\omega \hat{q} \times \hat{q} R p) = -\Omega(\vdot R p) J_\omega.$$ 

Thus (117) can be written as

$$m [J_\vdot^T \Omega(\vdot R p) + J_\omega^T (J_\vdot \hat{q} \times \hat{q} R p) - J_\vdot^T \Omega(\vdot R p) J_\omega \hat{q} - J_\omega^T \Omega(\vdot R p) J_\vdot \hat{q} \times \hat{q} R p].$$

(119)

By (19):

$$J_\omega^T (J_\vdot \hat{q} \times \hat{q} R p) = (J_\vdot \times J_\omega \hat{q} \hat{q} R p)$$

the second term in (119) is equal to

$$(J_\vdot \times J_\omega \hat{q})^T R (mp)$$

where

$$J_\vdot \times J_\omega \hat{q} \equiv [J_{v1} \times J_{v1} \cdots J_{vn} \times J_{vn}].$$

From (111), (120), and (121), it follows that if we define a ten-dimensional vector $\theta_5$ as

$$\theta_5 = \begin{bmatrix} p_x & p_y & p_z & p_x p_y & p_x p_z & p_y p_z \end{bmatrix}^T,$$

(122) then (117) can be expressed as

$$m \begin{bmatrix} \frac{\partial}{\partial q} (J_\vdot \hat{q} + J_\omega \hat{q} \times \hat{q} R p) \end{bmatrix}^T (J_\vdot \hat{q} + J_\omega \hat{q} \times \hat{q} R p) = W_1(q, \hat{q}) \theta_5,$$

(125)

with

$$W_1(q, \hat{q}) = [J_\vdot^T J_\vdot \hat{q} + J_\omega^T J_\omega \hat{q} + J_\vdot^T J_\omega \hat{q} \times \hat{q} R p - D],$$

(126)

and $D$ is determined by

$$[p_x^T R J_\omega \hat{q}] J_\vdot^T R p = [D[p_x^T \ p_y^T \ p_z^T \ p_x p_y \ p_x p_z \ p_y p_z].$$

(129)

REFERENCES


