System Identification and Control of Valkyrie through SVA–Based Regressor Computation

Shishir Kolathaya[1], Benjamin J. Morris[1], Ryan W. Sinnet[1] and Aaron D. Ames[2]

***Abstract*— This paper demonstrates simultaneous identification and control of the humanoid robot, Valkyrie, utilizing Spatial Vector Algebra (SVA). In particular, the inertia, Coriolis-centrifugal and gravity terms for the dynamics of a robot are computed using spatial inertia tensors. With the assumption that the link lengths or the distance between the joint axes are accurately known, it will be shown that inertial properties of a robot can be directly evaluated from the inertia tensor. An algorithm is proposed to evaluate the regressor, yielding a run time of $O(n^2)$. The efficiency of this algorithm yields a means for online system identification via the SVA–based regressor and, as a byproduct, a method for accurate model-based control. Experimental validation of the proposed method is provided through its implementation in three case studies: offline identification of a double pendulum and a 4-DOF robotic leg, and online identification and control of a 4-DOF robotic arm.**

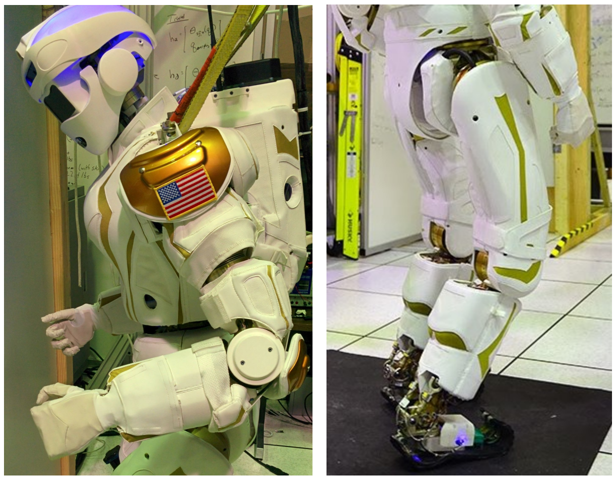


Fig. 1: Figures showing the 4-DOF robotic arm on the left and the 4-DOF leg on the right of the Valkyrie robot on which identification was conducted.

I. Introduction

The field of system identification has been a subject of much attention in the field of robotic systems, since the 1980's [8], [2], [17], [12]. One of the most important reasons is that achieving good tracking performance in robotic systems, specifically exponential convergence, without knowing the complete model of the robot has not been shown. Asymptotic convergence in tracking without knowing the model parameters has been shown in [16], [4] by using a special property of the Lagrangian dynamics of the robot, that is, the linearity of the parameters in the dynamics. Exponential convergence specific to a particular task using machine learning was also shown in [5] by using the concept of persistence of excitation [11], which requires running a series of trials for the controller to learn.

As a deviation from the methods shown above, it can be argued that identification of physical systems is required to realize good tracking performance. Currently, many system identification procedures have been implemented mainly by either measuring the parameters of the robot part by part [7], or dynamically by using the acceleration, velocity and angles of the robot [12], [14]. [1] showed that the parameters that are identified as linear combinations can be consistently set to zero to determine the set of identifiable parameters. Specifically, since the parameters are affine, the equation of motion can be expressed as a matrix (regressor) multiplied by a vector of unknown parameters (base inertial parameters). [18] used a novel method of selectively designing a robot such that the resulting inertia distribution linearizes the manipulator dynamics. In other words, the inertia parameters are made affine in the equations of motion of the rigid body robot. [15] used a similar method, where the affineness of the parameters in the dynamics is leveraged to compute the regressor.

Computing the regressor is primarily done by solving for the dynamics for an n-DOF, b-body robot and collecting the unknown parameters into a vector. [9] approached this problem using an energy based approach by using the Lagrangian formulation of robot dynamics as a starting point. [9] also showed a second approach where the Newton-Euler recursion method is reformulated using vector analysis–type techniques. [1] also used Newton-Euler equations in which the the acceleration data are obtained through a least squares estimation and the applied torques and forces are substituted to evaluate the regressor.

This paper uses the method adopted from [13] to evaluate the regressor, i.e., use Spatial Vector Algebra (SVA) to compute the regressors, an elegant of representing the Newton-Euler equations. If the distance between the axes are known prior to the experiment, the parameters to be identified are effectively the contents of the *spatial inertia tensor*. These tensors are computed by shifting the inertial elements to the joint axes. The contents of the tensor are also similar to the D-H parameters of each link in [12]. Given a n-DOF robot,

*This research is supported by NASA grant NNX11AN06H, NSF grants CNS-0953823 and CNS-1136104, and NHARP award 00512-0184-2009.

[1] R. Sinnet and B. Morris are with the department of Mechanical Engineering, Texas A & M University, 3128 TAMU, College Station, USA `shishirny,rsinnet,bmorris@tamu.edu`

[2] S. Kolathaya and Prof. A. D. Ames are with department of Mechanical Engineering, Georgia Institute of Technology, 85 5th St, Atlanta, USA `shishirny,ames@gatech.edu`

this novel technique specifically lists the parameters to be identified directly from the spatial inertia tensor. Contents of this tensor are not the minimum representation, and therefore will not be unique. But, it will be shown that these non-unique parameters obtained are sufficient for realizing the model based controller, computed torque, on the robot. This method is demonstrated on a double pendulum as well as the leg and arm of the Valkyrie robot (Figure 1). Online identification is done on the pendulum and the leg, and online model based control combined with identification is done on the arm.

We start with a brief introduction to spatial vectors in Section II which is extracted from [3]. Representation of kinetic energy, spatial momentum, forces and rigid body transformations in terms of spatial vectors are also explained. Section III shows how to use Spatial Vector Algebra (SVA) to extract the base inertial elements and the regressor of the robot conveniently. The derivation of this regressor and the base inertial elements are explained in detail in Section IV. The resulting algorithm to compute the regressor is explained in the same section and applications to control are considered in Section V. This is finally implemented and parameters are identified for three models in Section VI.

II. Spatial Vector Algebra for a Rigid Body

This section will introduce the concept of spatial vectors and Spatial Vector Algebra. This is primarily derived from [3] and many of the equations in this section are variants of the equations found in the same.

Rigid body motions and forces are normally described in two separate entities: 3D linear vectors and 3D angular vectors. Computing linear and rotational dynamics separately has been the normal practice in describing the equations of motion of bodies. But, if the two 3D vectors are combined together to form a 6D vector, a new vector space can be described for such systems. This vector space, of course has different rules and regulations when performing the standard mathematical operations. The 6D vectors are formed formally by what are called the Plücker coordinates.

A rigid body with origin located at point O, linear velocity v, and angular velocity ω, about an axis passing through O is shown in Figure 2. The spatial velocity of the rigid body

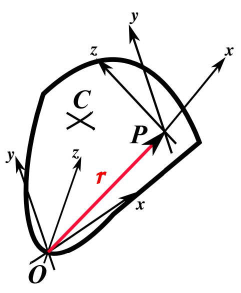


Fig. 2: Figure showing the rigid body with the origin O, an arbitrary point P and the center of mass located at C.

can be represented as:

$$\hat{v}_O = \begin{bmatrix} \omega_x, \omega_y, \omega_z, v_x, v_y, v_z \end{bmatrix}^T \tag{1}$$

Similarly, the spatial force which consists of the linear force acting at point O and the moment about the axis passing through O can be represented as:

$$\hat{f}_O = \begin{bmatrix} \tau_x, \tau_y, \tau_z, f_x, f_y, f_z \end{bmatrix}^T. \tag{2}$$

The order of angular and translational vectors considered is not important. Spatial vectors can also be considered with translational vector considered first and followed by the rotational vector.

It is important to note that the spatial vectors and forces are independent of the origin considered and depend solely on the bases chosen. The reason behind using spatial vectors is that both rotations and translations can be represented in one vector. In addition, the properties of these spatial vectors are different and they have a different algebra. More details about the algebra of spatial vectors can be found in [3].

Coordinate Transforms. It is important to have coordinate transformations in order to realize rotations and translations of coordinate frames of robotic systems using spatial vectors.

Translation. Spatial forces and velocities have the following forms of translation from point O to an arbitrary point P.

$$\hat{v}_P = \begin{bmatrix} 1 & 0 \\ -r\times & 1 \end{bmatrix} \hat{v}_O, \quad \hat{f}_P = \begin{bmatrix} 1 & -r\times \\ 0 & 1 \end{bmatrix} \hat{f}_O \tag{3}$$

These can be derived based on the fact that the linear velocity $v_P = v_O + \omega \times r$ and torque $\tau_P = \tau_O + f \times r$. Here r is the position vector directed from O to P, i.e., $\vec{OP}$ (see Figure 2). $r\times$ denotes the matrix equivalent of the cross product.

Rotation. Spatial forces and velocities have the following forms of rotation about a point O:

$$\hat{v}_P = \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix} \hat{v}_O, \quad \hat{f}_P = \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix} \hat{f}_O \tag{4}$$

where ν indicates the rotation about the axes x, y or z or two of them or even all of them at once. If both rotations and translations are involved then the following relationship is obtained:

$$\hat{v}_P = \begin{bmatrix} \nu & 0 \\ -\nu r\times & \nu \end{bmatrix} \hat{v}_O, \quad \hat{f}_P = \begin{bmatrix} \nu & -\nu r\times \\ 0 & \nu \end{bmatrix} \hat{f}_O. \tag{5}$$

Note that the translation operation from O to P is done first, and the the rotation about point P is carried out. If it is required that the rotation be done first, then the rotation matrix from (4) is multiplied with v_O, and then the translation is carried out. In doing this, it is important to remember that the vector r also gets rotated by ν. The resulting coordinate transformation will look like:

$$\hat{v}_P = \begin{bmatrix} \nu & 0 \\ -(\nu r)\times\ \nu & \nu \end{bmatrix} \hat{v}_O,$$

$$\hat{f}_P = \begin{bmatrix} \nu & -(\nu r)\times\ \nu \\ 0 & \nu \end{bmatrix} \hat{f}_O, \tag{6}$$

where (νr) is the vector r rotated by the matrix ν. Note that $(\nu r)\times = \nu r \times \nu^{-1}$. When this is substituted in (6) we effectively get (5). This result will be useful in reconstructing the spatial inertia tensors in order to conveniently evaluate the regressor algorithm.

Momentum of a Rigid Body. If the body has mass m, rotational inertia $\bar{I}_C$ about its center of mass, the following spatial momentum is described:

$$\hat{h}_C = \begin{bmatrix} \bar{I}_C\omega \\ mv_C \end{bmatrix} = \underbrace{\begin{bmatrix} \bar{I}_C & 0 \\ 0 & m1 \end{bmatrix}}_{I_C} \hat{v}_C, \tag{7}$$

which is the product of the spatial inertia I_C and the spatial velocity $\hat{v}_C$. The momentum defined in (7) was w.r.t. the center of mass. To compute the momentum about an arbitrary point O, we have to do the transformation. So the transformation from point O to the center of mass C is given by:

$$\hat{h}_C = \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix} \begin{bmatrix} 1 & -r\times \\ 0 & 1 \end{bmatrix} \hat{h}_O, \tag{8}$$

which is obtained since $mv_O = mv_C + mr{\times}\omega_C$. v_O, ω_O represent the spatial vectors at point O, and v_C, ω_C represent the spatial vectors at point C. r is new position vector from point O to the center of mass C (instead of P). Expressing $\hat{h}_O$ in terms of $\hat{h}_C$, the transformation matrices get inverted:

$$\hat{h}_O = \begin{bmatrix} 1 & r\times \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix}^{-1} \hat{h}_C, \tag{9}$$

and using (7) in (8) and substituting for $\hat{v}_C$, we have:

$$\begin{aligned} \hat{h}_O &= \begin{bmatrix} \nu^{-1} & r{\times}\,\nu^{-1} \\ 0 & \nu^{-1} \end{bmatrix} I_C \hat{v}_C \qquad (10) \\ &= \begin{bmatrix} \nu^{-1} & r{\times}\,\nu^{-1} \\ 0 & \nu^{-1} \end{bmatrix} I_C \begin{bmatrix} \nu & 0 \\ -\nu r\times & \nu \end{bmatrix} \hat{v}_O. \end{aligned}$$

If the center of mass of the rigid body is c, then at zero rotation angle, let $r = c$. In other words, let r be the position of the center of mass such that rotation of the coordinate frame results in negative rotation of r. In other words $r = \nu^{-1}c$. Applying the trick used in (6) and substituting for $r\times = (\nu^{-1}c)\times = \nu^{-1}c \times \nu$, we have the following result:

$$\hat{h}_O = \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix}^{-1} I_O \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix} \hat{v}_O, \tag{11}$$

where I_O:

$$I_O = \begin{bmatrix} 1 & c\times \\ 0 & 1 \end{bmatrix} I_C \begin{bmatrix} 1 & 0 \\ c{\times}^T & 1 \end{bmatrix}, \tag{12}$$

forms the spatial inertia tensor, which is purely a function of the parameters of the robot and independent of the orientation of the coordinate frame considered. This equation will be used for a general n-DOF b-body robot where the spatial inertia tensor is effectively utilized to compute the unknown parameters. Simplifying the spatial inertia tensor results in:

$$I_0 = \begin{bmatrix} \bar{I}_C + m\ c\times\ c{\times}^T & m\ c\times \\ m\ c{\times}^T & m1 \end{bmatrix}. \tag{13}$$

Having the expression for momentum, the kinetic energy can now be computed as:

$$\begin{aligned} T &= \frac{1}{2}\hat{h}^T\hat{v} = \frac{1}{2}\hat{h}_O^T\hat{v}_O \\ &= \frac{1}{2}\hat{v}_O^T \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix}^T I_O \begin{bmatrix} \nu & 0 \\ 0 & \nu \end{bmatrix} \hat{v}_O. \qquad (14) \end{aligned}$$

III. Lagrangian Dynamics for an n-DOF, b-Body Robot

Equation of motion of an n-DOF manipulator is explained in detail in this section.

A robot can be modeled as an n-link manipulator. Given the configuration space $\mathbb{Q} \subset \mathbb{R}^n$, with the coordinates $q \in \mathbb{Q}$, and the velocities $\dot{q} \in T_q\mathbb{Q}$, the Lagrangian of the n degree of freedom robot can be defined as:

$$L(q, \dot{q}) = \frac{1}{2}\dot{q}^T D(q)\dot{q} - V(q), \tag{15}$$

where $D(q) \in \mathbb{R}^{n\times n}$ is the mass matrix of the robot, $V(q) \in \mathbb{R}^n$ is the potential energy of the robot. Specifically. The equations of motion of the n-link robot can be derived as:

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) \quad = B\mathrm{U}, \tag{16}$$

where $C(q, \dot{q}) \in \mathbb{R}^{n\times n}$ is the matrix of coriolis and centrifugal forces and $G(q) \in \mathbb{R}^n$ is the gravity matrix, $\mathrm{U} \in \mathbb{R}^m$ is the torque input with m being the number of actuators, and $B \in \mathbb{R}^{n\times m}$ is the mapping from actuator torques to joint torques, often the identity map.

The first step is to determine the inertia $D(q)$, Coriolis-centrifugal $C(q, \dot{q})$ and gravity $G(q)$ matrices of the robot via spatial vectors. We define the body spatial velocity for each link or body i of the manipulator about the joint axis O_i:

$$v_{O_i} = J_i(q)\dot{q}, \tag{17}$$

$J_i(q)$ is the body Jacobian of link i for the i^{th} joint axis. Declare the spatial inertia tensor about the joint axis O_i for the i^{th} link or body as I_{O_i} which is obtained from (13):

$$I_{O_i} = \begin{bmatrix} \bar{I}_{C_i} + m_i\ c_i\times\ c_i{\times}^T & m_i\ c_i\times \\ m_i\ c_i{\times}^T & m_i 1 \end{bmatrix}, \tag{18}$$

which is primarily obtained from the momentum equation of (7). $\bar{I}_{C_i}$ is the inertia matrix taken w.r.t. the center of mass and m_i is the mass for link i. c_i is the center of mass location of the same link w.r.t. the joint axis O_i. Accordingly, the kinetic energy of the i^{th} link is given by using (17). Substituting for v_{O_i} in (17) results in:

$$T_i = \frac{1}{2}\dot{q}^T J_i^T \begin{bmatrix} \nu_i & 0 \\ 0 & \nu_i \end{bmatrix}^T I_{O_i} \begin{bmatrix} \nu_i & 0 \\ 0 & \nu_i \end{bmatrix} J_i\dot{q}, \tag{19}$$

where ν_i denotes the rotation of the i^{th} link w.r.t. the joint axis. The inertia matrix, $D(q)$, can thus be expressed from the total energy of the b bodies:

$$\begin{aligned} T &= \sum_{i=1}^{b} T_i \qquad (20)\\ &= \frac{1}{2}\dot{q}^T \underbrace{\left(\sum_{i=1}^{b} J_i^T \begin{bmatrix} \nu_i & 0 \\ 0 & \nu_i \end{bmatrix}^T I_{O_i} \begin{bmatrix} \nu_i & 0 \\ 0 & \nu_i \end{bmatrix} J_i\right)}_{D(q)} \dot{q}, \end{aligned}$$

where the inertia tensor I_{O_i} is obtained from (18). Note that the Jacobian J_i is purely a function of the joint angles and the link lengths. Therefore, assuming that the distances between the joint axes are known (which are easy to measure), all the other terms, namely, center of mass position, inertia, masses are in the spatial inertia tensor I_{O_i}. This fact will be utilized in later sections to compute the regressor.

$C(q, \dot{q})$ can also be derived as a linear function of the same elements of the tensor, by utilizing the Christoffel symbols: Γ_{ijk} is obtained from the inertia matrix $D(q)$:

$$\begin{aligned} \Gamma_{ijk} &= \frac{1}{2}\left(\frac{\partial D_{ij}(q)}{q_k} + \frac{\partial D_{ik}(q)}{q_j} - \frac{\partial D_{kj}(q)}{q_i}\right) \\ C_i(q, \dot{q})\dot{q} &= \sum_{j,k=1}^{b} \Gamma_{ijk}\dot{q}_j\dot{q}_k. \qquad (21) \end{aligned}$$

The potential energy function $V(q)$ can be computed as sum of the potential energies of the individual links:

$$V(q) = \sum_{i=1}^{b} V_i(q) = \sum_{i=1}^{b} m_i g h_i(q), \qquad (22)$$

where m_i is the mass of the individual links, g is the gravity and h_i is the vertical position of the center of mass for each link. Therefore, h_i is the sum of heights of the i^{th} joint axis, h_{O_i}, and the vertical height of the CoM w.r.t. the joint axis:

$$h_i(q) = h_{O_i}(q) + h_{C_i}(q), \qquad (23)$$

since the link lengths are assumed to be known, h_{O_i} in (23) is known. The height of the CoM h_{C_i} is the dot product of the center of mass location $c_i = [c_{i,1}, c_{i,2}, c_{i,3}]^T$ rotated by a transformation matrix, ν, and the vertical axis. Assuming that z axis is along the vertical axis, we have:

$$h_{C_i}(q) = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^T \nu(q) c_i. \qquad (24)$$

(24) and (23) can be substituted in (22) to obtain:

$$V(q) = \sum_{i=1}^{n} m_i g h_{O_i}(q) + \sum_{i=1}^{b} g \underbrace{\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}^T \nu(q)}_{\kappa} m_i c_i, \qquad (25)$$

where the unknown parameters are linear in the expression and are already present in the inertia tensor I_O. Therefore, in perspective, all the unknown parameters are effectively collected in I_O, which motivates the path that this paper takes to compute the regressor for any general n-DOF b-body system.

IV. The Regressor and the Parameters

Since the parameters are not perfectly known, the equation of motion, (16) computed with the given set of parameters will be henceforth have $\hat{}$ over the symbols. Therefore, D_a, C_a, G_a are the actual inertia, motor inertia, Coriolis and gravity matrices of the robot, and $\hat{D}, \hat{C}, \hat{G}$ are the assumed inertia, Coriolis and gravity matrices of the robot.

Consider the equation of motion of an n-link robot which is obtained from the Lagrangian (15) and is restated here as:

$$K(q, \dot{q}, \ddot{q}) = B\mathrm{U}, \qquad (26)$$

where $K = D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q)$ obtained from (16).

It is a well known fact that the parameters of a robot, like the inertia, masses, position of center of mass are affine in (26) (see [17]). Therefore, it is possible to write (16) in the form:

$$K = \mathrm{Y}(q, \dot{q}, \ddot{q})\Theta = \mathrm{U}, \qquad (27)$$

where $\mathrm{Y}(q, \dot{q}, \ddot{q})$ is called the regressor in [17], and $\Theta \in \mathbb{R}^{n_P}$ is called the set of base inertial elements (parameters). It is important to note that Θ need not be unique, and if the set is unique, then it is called the Base Parameter Set [10]. We can write, $\Theta = [\theta_1, \theta_2, \theta_3, \dots]^T$, where each θ_i is a function of the unknown parameters of the robot. n_P is the size of the parameter set. Accordingly, $K_a = \mathrm{Y}(q, \dot{q}, \ddot{q})\Theta_a$, and $\hat{K} = \mathrm{Y}(q, \dot{q}, \ddot{q})\hat{\Theta}$, where Θ_a is the actual set of base inertial parameters, and $\hat{\Theta}$ is the assumed set of base inertial parameters.

Determining the base inertial parameters (Θ). Since we know the inertia of link i, $\bar{I}_C \in \mathbb{R}^{3\times3}$ is symmetric, and the square of a skew-symmetric matrix results in a symmetric matrix, $\bar{I}_{C_i} + c_i \times c_i \times^T$ is symmetric. We can assign the inertial parameters of the i^{th} link $\Theta_i = [\theta_{i,1}, \theta_{i,2}, \dots]$ to the elements of the spatial inertia tensor I_{O_i} given in (18) in the following manner:

$$I_{O_i} = \begin{bmatrix} \theta_{i,1} & \theta_{i,2} & \theta_{i,3} & 0 & -\theta_{i,4} & \theta_{i,5} \\ \theta_{i,2} & \theta_{i,6} & \theta_{i,7} & \theta_{i,4} & 0 & -\theta_{i,8} \\ \theta_{i,3} & \theta_{i,7} & \theta_{i,9} & -\theta_{i,5} & \theta_{i,8} & 0 \\ 0 & \theta_{i,4} & -\theta_{i,5} & \theta_{i,10} & 0 & 0 \\ -\theta_{i,4} & 0 & \theta_{i,8} & 0 & \theta_{i,10} & 0 \\ \theta_{i,5} & -\theta_{i,8} & 0 & 0 & 0 & \theta_{i,10} \end{bmatrix},$$

where 10 parameters are obtained for each link. This is similar to how the parameters were categorized in [6], which uses the newton-euler method directly. The major difference is that (28) is obtained from the tensors and are used directly in online identification and control, which gets tedious without SVA. Obtaining $D(q)$ and $C(q, \dot{q})$ from I_{O_i} are straightforward from (20) and (21). Consider the gravity vector, which is the partial derivative w.r.t q of the potential energy, $V(q)$:

$$G(q) = \sum_{i=1}^{n} g\left(\frac{\partial h_{O_i}}{\partial q} m_i + \frac{\partial \kappa}{\partial q} c_i m_i\right) \qquad (28)$$

Therefore, even the gravity vector $G(q)$ is a linear function of the inertial parameters present in the tensor.

Algorithm 1 Regressor Pseudocode

```
for $i = 1$ to $n$ do
  for $j = 1$ to $10$ do
    $\theta_{i,j} = 0$
  end for
  Update $I_{O_i}$ with the value $\theta_{i,j}$
end for
for $j = 1$ to $n$ do
  for $j = 1$ to $10$ do
    $\theta_{i,j} = 1$
    Update $I_{O_i}$ with the value $\theta_{i,j}$
    $\mathrm{Y}_{i+j-1} = D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q)$
    $\theta_{i,j} = 0$
    Update $I_{O_i}$ with the value $\theta_{i,j}$
  end for
end for
```

Comparing with [1] and [12], the inertial parameters chosen were different than the one chosen here. Specifically, the inertial parameters in $\bar{I}_C$ form the base inertial parameters; whereas here the parameters in the matrix $\bar{I}_{C_i} + c_i \times c_i \times^T$ make the unknown parameter set. Besides, it is also possible to directly compute the regressor while evaluating the dynamics of the robot, by just picking the coefficient of every parameter θ_i one by one. In fact, this becomes the basis for a very simple algorithm shown in Algorithm 1. For a robot having b rigid bodies, the number of unknown parameters will be $10b$.

It is shown in [3] that it is possible to compute inverse dynamics of n-DOF robot in $O(n)$. This is achieved by deploying Newton-Euler recursive method. This is a standard technique used in numerical computation of the dynamics, which was used initially in the 1980's. Therefore, assuming that it is possible to compute the eom in just n recursive iterations, we propose Algorithm 1 which calls in the current state and acceleration of the robot and computes the regressor from the data. The number of iterations for evaluating the regressor is $10b$, which is proportional to the number of rigid bodies present in the robot. And the maximum number of degrees of freedom for each rigid body is 6, which implies that $b \leq n \leq 6b$. Accordingly in each iteration the equation of motion is computed which takes n iterations, and the resulting algorithmic complexity will be $O(n^2)$.

V. Estimation and Control

The regressor of the previous section has two main applications 1) to facilitate the identification of unknown model parameters and 2) to enable straightforward calculation of computed-torque controllers. This section states these problems in the general case, with specific examples to follow in Section VI.

Parameter Identification. The problem of parameter identification can be stated as follows: Suppose we are given a fully actuated robotic linkage with b rigid bodies, n degrees of freedom and unknown inertial parameters $\Theta_a \in R^{10b}$. Given s vectors of torque $\mathrm{U} = [u_1, u_2, \ldots, u_n]^T$, generalized configuration $q = [q_1, q_2, \ \ldots, q_n]^T$, generalized velocity data $\dot{q} = [\dot{q}_1, \dot{q}_2, \ldots, \dot{q}_n]^T$, and generalized acceleration $\ddot{q} = [\ddot{q}_1, \ddot{q}_2, \ldots, \ddot{q}_n]^T$, choose model parameters $\hat{\Theta} \in R^{10b}$ such that[1]

$$\hat{\Theta} = \underset{\Theta \in \mathbb{R}^{10b}}{\operatorname{argmin}} \quad \|\mathrm{U}_C - \mathrm{Y}_C \Theta\|_2 \tag{29}$$

where U_C is the collection of torque vector inputs and Y_C is the collection of regressor matrices for s samples of angle, velocity and acceleration data. U_C and Y_C are given as:

$$\mathrm{U}_C = \begin{bmatrix} \mathrm{U}[1] \\ \mathrm{U}[2] \\ \vdots \\ \mathrm{U}[s] \end{bmatrix}, \mathrm{Y}_C = \begin{bmatrix} \mathrm{Y}(q[1], \dot{q}[1], \ddot{q}[1]) \\ \mathrm{Y}(q[2], \dot{q}[2], \ddot{q}[2]) \\ \vdots \\ \mathrm{Y}(q[s], \dot{q}[s], \ddot{q}[s]) \end{bmatrix}$$

and

$$q[i] = \begin{bmatrix} q_1[i] \\ q_2[i] \\ \vdots \\ q_n[i] \end{bmatrix} \quad \dot{q}[i] = \begin{bmatrix} \dot{q}_1[i] \\ \dot{q}_2[i] \\ \vdots \\ \dot{q}_n[i] \end{bmatrix} \quad \ddot{q}[i] = \begin{bmatrix} \ddot{q}_1[i] \\ \ddot{q}_2[i] \\ \vdots \\ \ddot{q}_n[i] \end{bmatrix}.$$

A vector $\hat{\Theta}$ that minimizes $\|\mathrm{U}_C - \mathrm{Y}_C \hat{\Theta}\|_2$ can be found using the Moore-Penrose pseudoinverse:

$$\hat{\Theta} = \operatorname{pinv}(\mathrm{Y}_C)\mathrm{U}_C. \tag{30}$$

An estimated (or modeled) set of computed torques $\hat{\mathrm{U}}_C$ can be calculated using the parameter vector $\hat{\Theta}$,

$$\hat{\mathrm{U}}_C = \mathrm{Y}_C \hat{\Theta}. \tag{31}$$

Referring to the definition of $\hat{\mathrm{U}}_C$ above, a set of estimated torques can be found for each actuator. These estimates will be denoted $[\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_n]$. The coefficient of determination, R^2, can be used to describe the similarity between U_C and $\hat{\mathrm{U}}_C$ (or equivalently between $[u_1, u_2, \ldots, u_n]$ and $[\hat{u}_1, \hat{u}_2, \ldots, \hat{u}_n]$),

$$\begin{aligned} e &= \mathrm{U}_C - \mathrm{Y}_C \hat{\Theta} \\ R^2 &= 1 - (e^T e)/(\mathrm{U}_C^T \mathrm{U}_C). \end{aligned} \tag{32}$$

Parameter Identification with an Initial Guess. An initial guess or nominal parameter value can be incorporated into the parameter identification problem by modifying the cost function in (29).

$$\hat{\Theta} = \underset{\Theta \in \mathbb{R}^{10b}}{\operatorname{argmin}} \quad \alpha \|\mathrm{U}_C - \mathrm{Y}_C \Theta\|_2 + (1 - \alpha)\|\Theta - \Theta_0\|_2 \tag{33}$$

where $\Theta_0 \in R^{10b}$ is a nominal parameter vector, and $\alpha \in (0, 1)$ is a factor that can be used to vary the relative effects of the problem data (U_C and Y_C) and the initial guess (Θ_0).

[1]If necessary, the statement of the parameter identification problem (29) can be modified to include a requirement that $\Theta > 0$. The resulting problem will, in general, no longer have a closed form solution and could instead be solved using constrained quadratic programming.

A least squares solution to (33) can be found that is similar in structure to the solution of (29),

$$\hat{\Theta} = \mathrm{pinv}(\widetilde{\mathrm{Y}_C})\widetilde{\mathrm{U}_C} \tag{34}$$

where

$$\widetilde{\mathrm{U}_C} = \begin{bmatrix} \alpha\,\mathrm{U}_C \\ (1-\alpha)\,\Theta_0 \end{bmatrix}, \ \widetilde{\mathrm{Y}_C} = \begin{bmatrix} \alpha\,\mathrm{Y}_C \\ (1-\alpha)\,\mathbf{I}_{10b} \end{bmatrix}. \tag{35}$$

Rank Properties of the Regressor. It is not necessary to consider all of the samples to compute the parameters because samples leading to low eigen values of the matrix Y_C form a computation burden. Furthermore, it is possible to obtain p^* samples which give the parameter estimate $\hat{\Theta} = \Theta^*$ from the optimization problem (29) such that:

$$\mathrm{Y}_C(q, \dot{q}, \ddot{q})\Theta^* = \mathrm{Y}_C(q, \dot{q}, \ddot{q})\Theta_a \tag{36}$$

Once a parameter vector is identified, the regressor can be used in a computed torque controller to bring about a desired acceleration of the joints. Let $\ddot{q}_{cmd} \in R^n$ represent a desired vector of joint accelerations, then a unique controller to induce these accelerations is given by:

$$\mathrm{U}_{cmd} = \mathrm{Y}(q, \dot{q}, \ddot{q}_{cmd})\hat{\Theta}. \tag{37}$$

The following lemma will introduce the relationship between the desired and actual acceleration of the robot.

Lemma 1: *For the fully actuated robot, i.e., $B = I_{n\times n}$, if $\hat{\Theta} = \Theta^*$ is evaluated from the optimization problem* (29) *with p^* samples, and if the control law used is* (37)*, then $\ddot{q} = \ddot{q}_{cmd}$.*

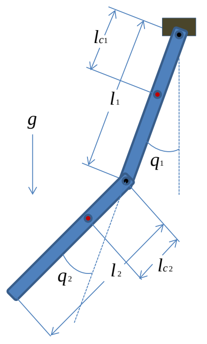
Note that the inertial parameters obtained from above will not yield true parameters of the robot, but will give the same value for the computed torque as described by Lemma 1. This property will be used in implementing online model based controllers and eliminate the identification of true parameters of the robot.

VI. Experimental Results

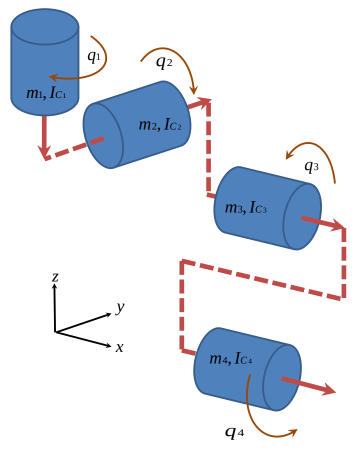
This section presents three experimental studies which use the regressor of Section IV to solve problems of identification and the control of identified systems. The first application will be in offline identification of a planar double pendulum, the second in offline identification of a 3D robotic leg of the Valkyrie robot (see Figure 1), and the third in online identification and control of a 3D robotic arm of the Valkyrie robot (see Figure 1).

Offline Identification of a Planar Double Pendulum. Consider a double pendulum, such as the one illustrated in Figure 3(a) with inertia tensors for both link 1 and 2 which are computed from (18). Note that since the pendulum considered is planar, the number of parameters used from the inertia tensor effectively being used is 3.

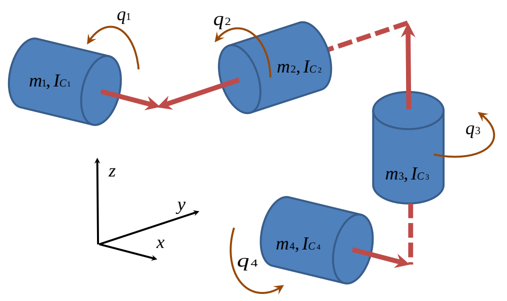
Figure 4 shows the outcome of an experiment where angles, velocities, accelerations, and torques were measured at each controller instant while the linkage executed a sinusoidal motion at each joint. An estimate of the parameter vector Θ was made without reference to an initial guess, and



(a) Illustration of the double pendulum.



(b) Figure showing the 4-link leg.



(c) Figure showing the robot arm.

Fig. 3: Robot models considered in this paper.

thus the estimation scheme of (30) was used. The tracking plots of Figure 4 and the scatter plots of Figure 5 illustrate that the offline identification procedure (using all collected data for a single parameter fit) produced a parameter set leading to very high correlation. As noted in Section IV, the regressor Y_C will not necessarily be full-rank, and thus any estimate $\hat{\Theta}$ should not be expected to converge to the actual parameters Θ_a, even for very large data sets.

Offline Identification of a 4-Link Robotic Leg. 4-DOF leg shown in Figure 3 and Figure 1 was identified offline. In the experiment a series of position, velocity, acceleration, and torques were measured at each of the four joints of the robot. An online estimate was made of the data, where the parameter identification was updated at a lower, decimated rate. The fit algorithm used here for online identification is very similar to (34) used in offline identification of the double pendulum. The two differences for online application are that 1) the data vectors U_C and Y_C grow throughout the experiment, and as such the quality of fit improves the longer the experiment is run, and 2) the initial parameter guess Θ_0 is used to ensure bounded behavior at the beginning of the experiment. Results of the fit are shown in Figure 6.

Online Identification of a 4-Link Robotic Arm. Online identification was conducted on a robotic arm with four degrees of freedom (see Figure 3(c) and Figure 1). To draw an analogue to human physiology, the joints q_1, q_2, q_3, and q_4 can be thought of as the shoulder extensor, the shoulder

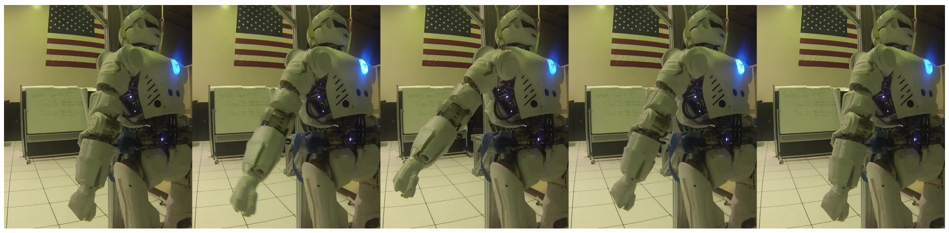


Fig. 8: Figure showing the tile of 4-DOF arm swing experiment used for identification.

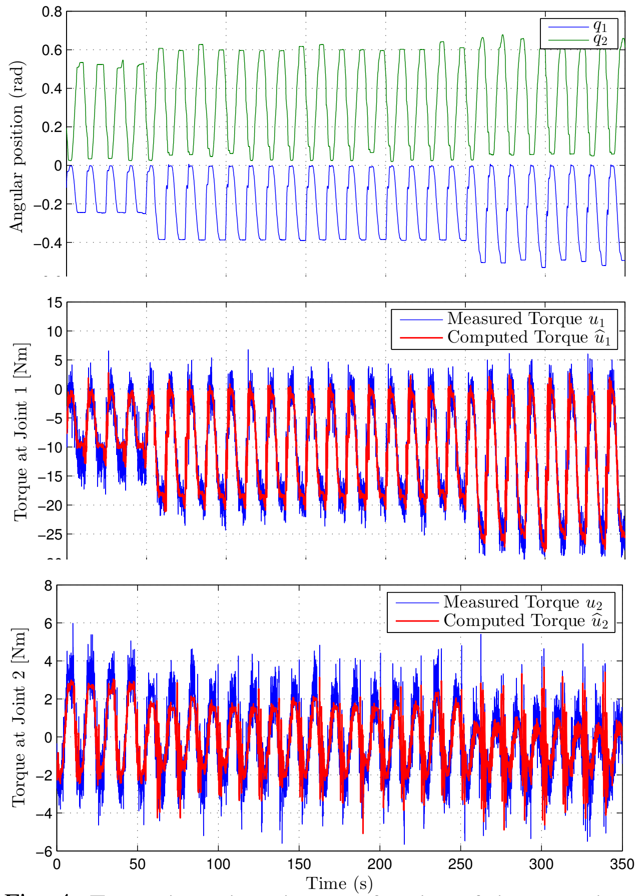


Fig. 4: **Top:** Joint trajectories as a function of time experienced by the experimental setup of Figure 3(a). **Middle, Bottom:** A comparison between experimentally measured torque vectors u_1, u_2 and the computed torque values $\hat{u}_1, \hat{u}_2$ corresponding to the identified system.

adductor/abductor, the upper arm pronator /supinator, and the elbow extensor, respectively. The joint angles and even velocities, which are necessary for computing the regressor, can be measured from encoders but accelerations must also be obtained and a common procedure is to filter accelerations – in this experiment we used an exponential moving average filter.

The experiment was run by controlling the arm to move from one position to another and back and so forth, following the trajectories in Figure 7(a). Tiles of the experiment are shown in Figure 8. The system identification procedure was brought online and was able to quickly identify the regressor

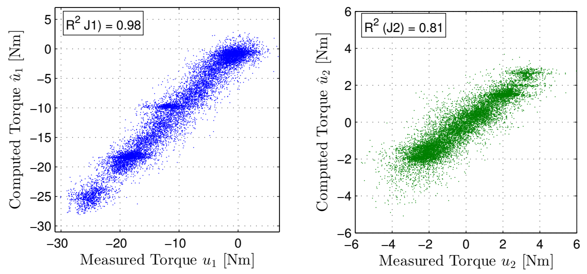


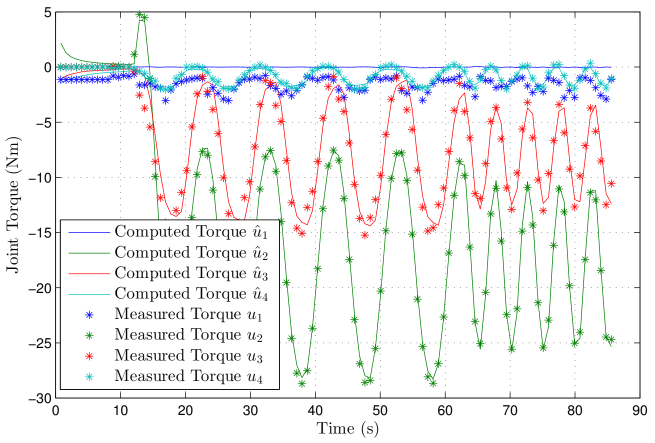
Fig. 5: Plotting measured torque vs. computed torque provides an illustration of quality of fit. The coefficient of determination, R^2, is shown for each joint.

parameters of the model. The initial guess was purposefully chosen to be incorrect to show the convergence properties (see Figure 7(b)) of the procedure as the nominal system model – that estimated from engineering software – was known with reasonable accuracy.

In order to avoid bad and potentially dangerous behavior, a threshold of .95 was set on the coefficient of determination, R^2, below which the nominal model would be used in place of the identified model. The validity of the identified model (and thereby its use in the controller) was also contingent upon the number of data points recorded. Specifically, it was required that the historical data buffers be full – in this experiment, 50 historical data were used in the buffer. The identification procedure was performed at 3 Hz and thus it took about 16–17 seconds for the buffer to be filled. It is quite apparent from Figure 7(c), as noted in the caption, when the controller began to use the identified model instead of the nominal model.

VII. Conclusions

Identification and control of a physical system, in particular an n-DOF robotic system with the implementation of an efficient computational mechanism was shown and demonstrated on three rigid body manipulators. The regressor involved in this implementation required a run time of $O(n^2)$ and computational errors resulting from this algorithm are solely due to the error in measurement of the states of the robots. This is a numerical method for computing the regressor and does not use symbolic expressions which are important for a robot like Valkyrie which has 44 degrees of freedom. Since SVA is required for computed torque control, evaluating the regressor through Algorithm 1 requires no extra computational overhead. In other words, this procedure



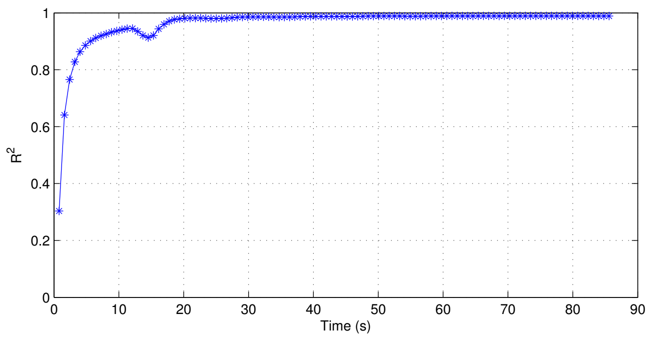
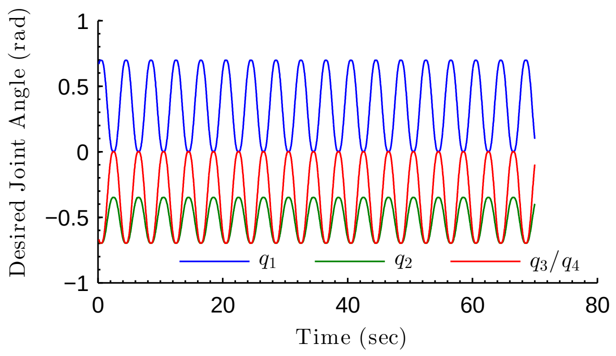
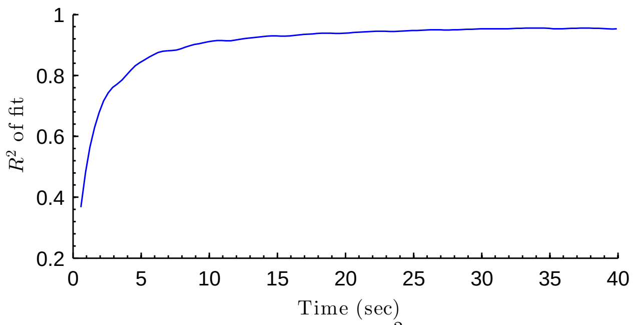


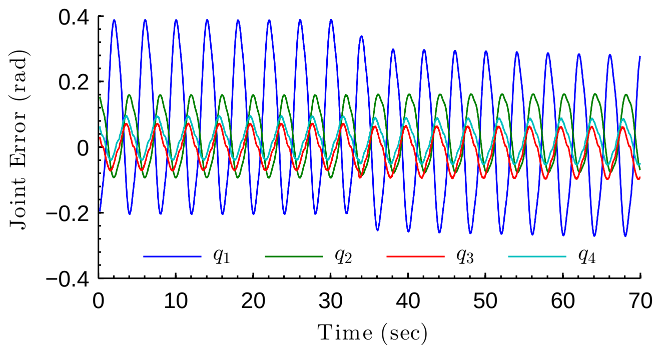
Fig. 6: The plots above show the quality of fit improves during an online identification exercise of the robot leg pictured in Figure 3(b). The top figure shows the instantaneous computed torque as a function of time, along with the measured data that were used for the parameter fit. Note that both the R^2 plot at bottom and the torque tracking plot at top both indicate that the output of the identification algorithm leads to noticeably higher quality of fit as the length of the available data vectors increases.



(a) The arm was made to exhibit a tick tock behavior where the joints followed the trajectories shown.



(b) The coefficient of determination, R^2, can be seen to converge to right below unity. With perfect sensing and no dependence on an incorrect initial guess, this quantity will converge precisely to unity. Due to noise, the evolution is non-monotonic.



(c) The peak-to-peak amplitudes of the joint errors appear essentially unchanged when the controller switches to using the identified model around 35 seconds, but the bias is clearly shifted closer to zero.

Fig. 7: Experimental results for a 4-DOF robotic arm.

can be directly integrated with in the Rigid Body Dynamics Library [3].

References

[1] Christopher G Atkeson, Chae H An, and John M Hollerbach. Estimation of inertial parameters of manipulator loads and links. *Intl. J. Robotics Research*, 5(3):101–119, 1986.

[2] Björn Bukkems, Dragan Kostić, Bram de Jager, and Maarten Steinbuch. Online identification of a robot using batch adaptive control. *Proc. 13th IFAC Symp. System Identification SYSID*, pages 953–958, 2003.

[3] Roy Featherstone. *Rigid Body Dynamics Algorithms*. Springer Science+Business Media, LLC, 2008.

[4] Fathi Ghorbel, John Y Hung, and Mark W Spong. Adaptive control of flexible-joint manipulators. *Control Systems Magazine, IEEE*, 9(7):9–13, 1989.

[5] Roberto Horowitz, William Messner, and John B Moore. Exponential convergence of a learning controller for robot manipulators. *IEEE Trans. Automatic Control*, 36(7):890–894, 1991.

[6] Pradeep K Khosla. Categorization of parameters in the dynamic robot model. *Robotics and Automation, IEEE Transactions on*, 5(3):261–268, 1989.

[7] Pradeep K Khosla and Takeo Kanade. Parameter identification of robot dynamics. In *24th IEEE Intl. Conf. Decision and Control*, volume 24, pages 1754–1760. IEEE, 1985.

[8] Krzysztof Kozlowski. *Modelling and identification in robotics*. Springer-Verlag, New York, 1998.

[9] W-S Lu and Q-H Meng. Regressor formulation of robot dynamics: computation and applications. *IEEE Trans. Robotics and Automation*, 9(3):323–333, 1993.

[10] H. Mayeda, K. Yoshida, and K. Oshun. Base parameters of manipulator dynamic models. *IEEE Trans. Robotics and Automation*, 6(3):312–321, 1990.

[11] John B Moore, Roberto Rorowitz, and William Messner. Functional persistence of excitation and observability. In *American Control Conference*, pages 308–314. IEEE, 1990.

[12] Marcelo H. Ang Jr. Ngoc Dung Vuong. Dynamic model identification for industrial robots. *Journal of Applied Sciences*, 6(5):51–68, 2009.

[13] Günter Niemeyer and Jean-Jacques E Slotine. Performance in adaptive manipulator control. *The International Journal of Robotics Research*,

10(2):149–161, 1991.

[14] Nicola Pedrocchi, Enrico Villagrossi, Federico Vicentini, and Lorenzo Molinari Tosatti. On robot dynamic model identification through sub-workspace evolved trajectories for optimal torque estimation. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 2370–2376. IEEE, 2013.

[15] Shih-Ying Sheu and Michael W Walker. Identifying the independent inertial parameter space of robot manipulators. *Intl. J. Robotics Research*, 10(6):668–683, 1991.

[16] Jean-Jacques E Slotine and Weiping Li. On the adaptive control of robot manipulators. *The International Journal of Robotics Research*, 6(3):49–59, 1987.

[17] Mark W Spong, Seth Hutchinson, and Mathukumalli Vidyasagar. *Robot modeling and control*. John Wiley & Sons, Hoboken, NJ, 2006.

[18] DCH Yang and SW Tzeng. Simplification and linearization of manipulator dynamics by the design of inertia distribution. *Intl. J. Robotics Research*, 5(3):120–128, 1986.