Structural Network Modeling and Control of Rigid Body Robots

S. S. Ge and C. C. Hang

***Abstract*—In this paper, two new parametric network models of robots are presented based on the systems' own functions. The complete dynamics of robots are given in a ready-to-used format which is constructed by finite dimensional static parametric networks for the inertia matrix and the potential energy (or the gravitational forces). As a result, dynamic models of robots can be automatically generated by software once given the number of degrees of freedom (DOF) and the sequence of the joint types, without knowing other parameters such as the lengths and the twist angles of the links. An existing adaptive controller is used as an example to show that some of the controllers can be easily modified such that adaptive controllers can be automatically generated. It is shown that all the closed-loop signals are bounded and tracking error goes to zero.**

***Index Terms*—Adaptive control of robots, network modeling, nonlinear systems.**

I. Introduction

Robotic manipulators are highly nonlinear and dynamically coupled systems. Increasingly sophisticated tools from nonlinear control theory have been developed for better tracking performance. Concurrent advances in microprocessor technology have made the implementation of complicated nonlinear control algorithms feasible from a practical point of view. Different control methods have been introduced in the literature such as computed torque control [1], [2], adaptive control [5], [6], variable structure control [3], [4], neural network control [11], [12], and so on.

Because of the underlying approximation nature of the neural networks, closed-loop stability was guaranteed only if the approximation errors of the neural networks were taken into account. A sliding mode control was often used to suppress the errors and provide closed-loop robustness. Other problems associated with neural network controllers include the difficulty in choosing the number of nodes, the number of layers, the choice of basis functions and the distribution of grids. No matter how large the size of the neural network is, modeling errors always exist.

The concept of structural network modeling using system's own functions has been proposed in order to eliminate the problems associated with the approximation nature of neural networks [11], [12]. In [11], structural network modeling of robots has been briefly discussed using Kronecker operator. In this paper, two different structural network models for robots are presented in a ready-to-use format using GL operator and GL matrix [12], [13]. The first is derived by applying the concept of structural networks to model the inertia matrix, $D(q)$, and the potential energy, $V(q)$, while the second is obtained by modeling $D(q)$ and the gravitational force vector $G(q)$ using structural networks. Subsequently, it is shown that adaptive controllers can be automatically generated once given the number of DOF and the sequence of joint types of a robot. With the present modeling techniques, adaptive controller design for robots becomes simply the tuning of control parameters (a turn key solution) because the network models can be constructed automatically and there is no need for dynamic modeling for controller design.

The paper is organized as follows. Network models of robots are presented in Section II. An adaptive controller is investigated in Section III and followed by the simulation tests in Section IV.

Manuscript received October 29, 1996; revised April 25, 1998. This paper was recommended for publication by Editor A. Goldenberg upon evaluation of the reviewer's comments.

The authors are with the Department of Electrical Engineering, National University of Singapore, Singapore 119260 (e-mail: elegesz@nus.edu.sg; dvchcc@nus.edu.sg).

Publisher Item Identifier S 1042-296X(98)06076-5.

II. Structural Network Modeling

For a general n DOF robot, the Lagrangian function is given by $L = K - V$ where the kinetic energy $K = \frac{1}{2}\dot{q}^T D(q)\dot{q}$, the gravitational potential energy $V = V(q)$, and $D(q) \in R^{n\times n}$ is the inertia matrix which is symmetric definite [7], [8]. Let $d_{kj}(q)$ be the kjth element of $D(q)$. The Lagrange–Euler equations can be expressed as

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

where the kjth element of $C(q, \dot{q})$ and the kth element of $G(q)$ are given by

$$c_{kj}(q, \dot{q}) = \sum_{i=1}^{n} c_{ijk}(q)\dot{q}_i \tag{2}$$

$$g_k(q) = \frac{\partial V(q)}{\partial q_k} \tag{3}$$

$$c_{ijk}(q) = \frac{1}{2}\left[\frac{\partial d_{kj}(q)}{\partial q_i} + \frac{\partial d_{ki}(q)}{\partial q_j} - \frac{\partial d_{ij}(q)}{\partial q_k}\right]. \tag{4}$$

Without losing generality, assume that the first n_1 joints are revolute and the rest $n_2 = n - n_1$ joints are prismatic. It can be seen that $d_{kj}(q)$ can be exactly parametrized by: $d_{kj}(q) = S_{kj}^T(q)P_{kj}$ where P_{kj} is the parameter vector and $S_{kj}(q) = [1, s_1^T(q), s_2^T(q), \cdots, s_{2n}^T(q)]^T$ with sub-vector $s_m(q)$ being defined by stacking the elements from the set [12]

$$\begin{aligned}\overline{\mathcal{H}}_{kj} = \Bigg\{ & \sin^{r_1} q_1 \cdots \sin^{r_{n_1}} q_{n_1} \cos^{k_1} q_1 \cdots \\ & \cos^{k_{n_1}} q_{n_1} q_{n_1+1}^{l_{n_1+1}} \cdots q_n^{l_n}, \forall r_i, k_i \\ & l_{n_1+j} \in [0 \le r_i, k_i, l_j, \le 2, \sum_{i=1}^{n_1}(r_i + k_i) \\ & + \sum_{j=n_1+1}^{n} l_j = m, i = 1, \cdots, n_1 \\ & j = n_1 + 1, \cdots, n, 0 < m \le 2n] \Bigg\}.\end{aligned} \tag{5}$$

The element, 1, in $S_{kj}(q)$ is to take care of the constant parameter of $d_{kj}(q)$. A reduction in the number of elements in $\overline{\mathcal{H}}_{kj}$ is possible by using the fact that $\sin^2 q_i + \cos^2 q_i = 1$ and among others. It is not going to be elaborated further since it is not the main contribution here. Let $\underline{\mathcal{H}}_{kj}$ be the minimum subset of $\overline{\mathcal{H}}_{kj}$ such that $d_{kj}(q)$ is exactly parametrized. Then the function $d_{kj}(q)$ is over parametrized (the number of parameters is more than necessary) by set $\mathcal{H}_{kj}$ when $\underline{\mathcal{H}}_{kj} \subseteq \mathcal{H}_{kj} \subseteq \overline{\mathcal{H}}_{kj}$. From (4), we have $c_{ijk}(q) = \frac{1}{2}[w_{kji}^T(q)P_{kj} + w_{kij}^T(q)P_{ki} - w_{ijk}^T(q)P_{ij}]$ where

$$w_{ijk}(q) = \left[\frac{\partial S_{ij}(q)}{\partial q_k}\right] = [0, w_{s1}^T(q), w_{s2}^T(q), \cdots, w_{sn}^T(q)]^T \tag{6}$$

with $w_{si}(q) = (\partial s_i(q)/\partial q_k)$. By applying the same idea, we have $V(q) = S_0^T(q)P_0$ with P_0 being the parameter vector of interest, and

$$S_0(q) = [s_1^T(q), s_2^T(q), \cdots, s_n^T(q)]^T \tag{7}$$

where $s_m(q) \in \overline{\mathcal{H}}_0 = \overline{\mathcal{H}}_{11}$ which leads to $g_k(q) = (\partial V(q)/\partial q_k) = w_{0k*}^T(q)P_0$ with

$$w_{0k}(q) = \left[\frac{\partial S_0(q)}{\partial q_k}\right] = [w_{s1}^T(q), w_{s2}^T(q), \cdots, w_{sn}^T(q)]^T \tag{8}$$

and $w_{si}(q) = (\partial s_i(q)/\partial q_k)$. Because of the partial derivatives, the 1 has been omitted in (7).

Therefore, once $S_{kj}(q)$ and $S_0(q)$ are pre-determined as networks of known functions to parametrize all possible variations, the corresponding network-based dynamics for robots can be described by **Network Model I**

$$D(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = \tau \tag{9}$$

where

$$d_{kj}(q) = S_{kj}^T P_{kj} \tag{10}$$

$$c_{kj}(q, \dot{q}) = \tfrac{1}{2}\sum_{i=1}^{n}\left[w_{kji}^T(q)P_{kj} + w_{kij}^T(q)P_{ki} - w_{ijk}^T(q)P_{ij}\right]\dot{q}_i \tag{11}$$

$$g_k(q) = w_{0k}^T(q)P_0. \tag{12}$$

The full dynamics of robots can be constructed using the networks for $D(q)$ and $V(q)$, because $C(q, \dot{q})$ can be constructed based on the parameters of $D(q)$.

The above parametric description for $G(q)$ is deduced from the network for the potential energy $V(q)$. Actually, the proposed network modeling approach can be applied to $G(q)$ directly. Let $S_{0k}(q)$ be formed by all the elements in the set $\mathcal{H}_{0k} \subseteq \overline{\mathcal{H}}_{0k} = \overline{\mathcal{H}}_{11}$, and P_{0k} be the corresponding parameter vector such that $g_k(q) = S_{0k}^T(q)P_{0k}$, then we have **Network Model II** accordingly with the kjth elements of $D(q)$ and $C(q, \dot{q})$ as defined by (10) and (11), respectively.

Property 1: As matrix $D(q)$ is symmetric, i.e., $d_{ij}(q) = d_{ji}(q)$, we have, $S_{ij}(q) = S_{ji}(q)$, $P_{ij} = P_{ji}$, $w_{ijk}(q) = w_{jik}(q)$, $\{w_{ij*}(q)\} = \{w_{ji*}(q)\}$ if the same network is used for the corresponding off-diagonal elements. Using GL matrices and its operator [12], [13], we have

$$D(q) = [\{S(q)\}^T \bullet \{P\}] = [\{P\}^T \bullet \{S(q)\}]. \tag{13}$$

Property 2: For clarity, define $\{w_{ij*}(q)\} = \{w_{ij1}(q), w_{ij2}(q), \cdots, w_{ijn}(q)\}$, $\{w_{i*j}(q)\} = \{w_{i1j}(q), w_{i2j}(q), \cdots, w_{inj}(q)\}$, $\{w_{*ij}(q)\} = \{w_{1ij}(q), w_{2ij}(q), \cdots, w_{nij}(q)\}$, $\{P_i\}^T = \{P_{i1}^T, P_{i2}^T, \cdots, P_{in}^T\}$, $P_{ij}^T \bullet \{w_{ij*}\} = [P_{ij}^T w_{ij1}, P_{ij}^T w_{ij2}, \cdots, P_{ij}^T w_{ijn}]^T$. Then, from (11), the matrix $C(q, \dot{q})$ can be expressed conveniently in three terms as

$$C(q, \dot{q}) = \tfrac{1}{2}[C_1(q, \dot{q}) + C_2(q, \dot{q}) - C_3(q, \dot{q})] \tag{14}$$

where

$$\begin{aligned} C_1(q, \dot{q}) &= \left[\{P\}^T \bullet \{W_1(q, \dot{q})\}\right] \\ &= \begin{bmatrix} \{P_1\}^T \bullet \{W_{11}(q, \dot{q})\} \\ \cdots \\ \{P_n\}^T \bullet \{W_{1n}(q, \dot{q})\} \end{bmatrix} \\ &= \begin{bmatrix} P_{11}^T \bullet \{w_{11*}(q)\}\dot{q} & \cdots & P_{1n}^T \bullet \{w_{1n*}(q)\}\dot{q} \\ \cdots & \cdots & \cdots \\ P_{n1}^T \bullet \{w_{n1*}(q)\}\dot{q} & \cdots & P_{nn}^T \bullet \{w_{nn*}(q)\}\dot{q} \end{bmatrix} \end{aligned} \tag{15}$$

$$\begin{aligned} C_2(q, \dot{q}) &= \left[\{P\}^T \odot \{W_2(q, \dot{q})\}\right] \\ &= \begin{bmatrix} \{P_1\}^T \bullet \{w_{1*1}(q)\}\dot{q} & \cdots & \{P_1\}^T \bullet \{w_{1*n}(q)\}\dot{q} \\ \cdots & \cdots & \cdots \\ \{P_n\}^T \bullet \{w_{n*1}(q)\}\dot{q} & \cdots & \{P_n\}^T \bullet \{w_{n*n}(q)\}\dot{q} \end{bmatrix} \end{aligned} \tag{16}$$

$$\begin{aligned} C_3(q, \dot{q}) &= \left[\{P\}^T \otimes \{W_3(q, \dot{q})\}\right] \\ &= \begin{bmatrix} \{P_1\}^T \bullet \{w_{*11}(q)\}\dot{q} & \cdots & \{P_n\}^T \bullet \{w_{*n1}(q)\}\dot{q} \\ \cdots & \cdots & \cdots \\ \{P_1\}^T \bullet \{w_{*1n}(q)\}\dot{q} & \cdots & \{P_n\}^T \bullet \{w_{*nn}(q)\}\dot{q} \end{bmatrix} \end{aligned} \tag{17}$$

with

$$\{W_1(q, \dot{q})\} = \begin{Bmatrix} \{w_{11*}(q)\}\dot{q} & \cdots & \{w_{1n*}(q)\}\dot{q} \\ \cdots & \cdots & \cdots \\ \{w_{n1*}(q)\}\dot{q} & \cdots & \{w_{nn*}(q)\}\dot{q} \end{Bmatrix} = \begin{Bmatrix} \{W_{11}(q, \dot{q})\} \\ \cdots \\ \{W_{1n}(q, \dot{q})\} \end{Bmatrix}$$

$$\{W_2(q, \dot{q})\} = \begin{Bmatrix} \{w_{1*1}(q)\}\dot{q} & \cdots & \{w_{1*n}(q)\}\dot{q} \\ \cdots & \cdots & \cdots \\ \{w_{n*1}(q)\}\dot{q} & \cdots & \{w_{n*n}(q)\}\dot{q} \end{Bmatrix}$$

$$\{W_3(q, \dot{q})\} = \begin{Bmatrix} \{w_{*11}(q)\}^T\dot{q} & \cdots & \{w_{*n1}(q)\}^T\dot{q} \\ \cdots & \cdots & \cdots \\ \{w_{*1n}(q)\}^T\dot{q} & \cdots & \{w_{*nn}(q)\}^T\dot{q} \end{Bmatrix}$$

$$\{W_{1i}(q, \dot{q})\} = \{\{w_{i1*}(q)\}\dot{q}, \quad \cdots, \quad \{w_{in*}(q)\}\dot{q}\}$$

and $\odot$, $\otimes$ are different operators as defined above. Thus, $C(q, \dot{q})$ can be expressed as

$$C(q, \dot{q}) = \tfrac{1}{2}\left[\{P\}^T W_s(q, \dot{q})\right] \tag{18}$$

where

$$W_s(q, \dot{q}) = \bullet\{W_1(q, \dot{q})\} + \odot\{W_2(q, \dot{q})\} - \otimes\{W_3(q, \dot{q})\}. \tag{19}$$

Even though $W_s(q, \dot{q})$ is not well defined, $[\{P\}^T W_s(q, \dot{q})]$, accordingly, $C(q, \dot{q})$ are well defined by computing $\bullet$, $\odot$ and $\otimes$ first. From Property 1, we know that $\{w_{*jk}(q)\} = \{w_{j*k}(q)\}$, which subsequently leads to

$$C_3(q, \dot{q}) = \left[\{P\}^T \odot \{W_2(q, \dot{q})\}\right]^T = C_2^T(q, \dot{q}). \tag{20}$$

Property 3: Matrix $N := \dot{D}(q) - 2C(q, \dot{q})$ is skew-symmetric by definition. A new insight into $N^T = -N$ is given here owing to the network representation for $D(q)$. Because $\dot{D}(q) = [\{P\}^T \bullet \{\dot{S}(q)\}] = [\{P\}^T \bullet \{W_1(q, \dot{q})\}] = C_1(q, \dot{q})$, we have $N = \dot{D}(q) - 2C(q, \dot{q}) = C_3(q, \dot{q}) - C_2(q, \dot{q})$. Thus, it is easy to verify that $N^T = -N$ because $C_2^T(q, \dot{q}) = C_3(q, \dot{q})$.

Property 4: The network models can be expressed in linear-in-the-parameters dynamics explicitly. Let $\ddot{q}_r, \dot{q}_r \in R^n$, the linear-in-the-parameters dynamics for Network Model I can be written as

$$[\{P\}^T \bullet \{S(q)\}]\ddot{q}_r + \tfrac{1}{2}\left[\{P\}^T W_s(q, \dot{q})\right]\dot{q}_r + W_0^T(q)P_0 = \tau \tag{21}$$

with P_0 being the parameter vector for $V(q)$ and

$$W_0^T(q) = \begin{bmatrix} w_{01}(q) \\ w_{02}(q) \\ \cdots \\ w_{0n}(q) \end{bmatrix}.$$

The linear-in-the-parameters dynamics for Network Model II can be expressed as

$$\begin{aligned} &[\{P\}^T \bullet \{S(q)\}]\ddot{q}_r + \tfrac{1}{2}\left[\{P\}^T W_s(q, \dot{q})\right]\dot{q}_r \\ &\quad + [\{S_0(q)\}^T \bullet \{P_0\}] = \tau \end{aligned} \tag{22}$$

where $\{S_0(q)\}$ and $\{P_0\}$ are defined and re-defined as

$$S_0(q) = \begin{Bmatrix} S_{01}(q) \\ \cdots \\ S_{0n}(q) \end{Bmatrix}, \; P_0 = \begin{Bmatrix} P_{01} \\ \cdots \\ P_{0n}^T \end{Bmatrix}. \tag{23}$$

Equations (21) and (22) are in a ready-to-use format for controller implementation.

Remark 1: For articulated robots, $S_{kj}(q)$ and $S_0(q)$ [or $S_{0k}(q)$] are described by the complete set of

$$\overline{\mathcal{H}}_{kj} := \Big\{ \sin^{r_1} q_1 \cdots \sin^{r_n} q_{n_1} \cos^{k_1} q_1 \cdots \cos^{k_{n_1}} q_n \;\; \forall r_i,\, k_i \in [0 \le r_i,\, k_i, \le 2,\, \sum_{i=1}^{n} (r_i + k_i) = m,\, 0 \le m \le 2n, \quad i = 1, \cdots, n] \Big\}. \tag{24}$$

The most attractive feature of the above set is that each element is bounded. Therefore the networks are of bounded "basis" functions in neural network terminology.

Remark 2: For prismatic robots, $S_{kj}(q)$ and $S_0(q)$ [or, $S_{0k}(q)$] are described by the complete set: $\overline{\mathcal{H}}_{kj} := \{q_1^{l_1} \cdots q_n^{l_n},\, \forall l_i \in [0 \le l_i, \le 2,\, \sum_{i=1}^{n}(l_i) = m,\, 0 \le m \le 2n,\, i = 1, \cdots, n]\}$. In this case, each element in the above set is not necessarily bounded. This, however, is not going to cause a problem in controller design for the closed-loop stability is guaranteed.

Remark 3: The present modeling approach can also be used to solve the nonlinear friction modeling problems by simply constructing a function space large enough to include all possible nonlinearities.

III. Controller Design

For illustration, an adaptive controller design based on the Network Model I (21) is briefly presented. Similar results can be obtained for Network Model II. Let $q_d,\, \dot{q}_d,\, \ddot{q}_d$ be the desired position, velocity, and acceleration of the desired trajectory. Define the tracking error as $e = q_d - q$, and modified reference velocity as $\dot{q}_r = \dot{q}_d + \Lambda e$ where $\Lambda = \mathrm{diag}[\lambda_i] > 0$. The dynamic tracking error measure is defined as $r := \dot{q}_r - \dot{q} = \dot{e} + \Lambda e$, which guarantees that if $r \to 0$ as $t \to \infty$, then e and $\dot{e} \to 0$. Let $(\hat{.})$ be the estimate of (.), define $(\tilde{.}) = (.) - (\hat{.})$, $\hat{D}(q) := [\{\hat{P}\}^T \bullet \{S(q)\}]$, $\hat{C}(q, \dot{q}) := \frac{1}{2}\Big[\{\hat{P}\}^T W_s(q, \dot{q})\Big]$, and $\hat{G}(q) := W_0^T(q)\hat{P}_0$.

Consider the general controller of the form

$$\tau = \hat{D}(q)\ddot{q}_r + \hat{C}(q, \dot{q})\dot{q}_r + \hat{G}(q) + Kr + K_i \int_0^t r \, d\tau. \tag{25}$$

From (9), (21), and (25), we have an error equation of the form

$$\begin{aligned} D(q)\dot{r} &+ C(q, \dot{q})r + Kr + K_i \int_0^t r\, d\tau \\ &= \tilde{D}(q)\ddot{q}_r + \tfrac{1}{2}\Big(\tilde{C}_1(q, \dot{q}) + \tilde{C}_2(q, \dot{q}) - \tilde{C}_3(q, \dot{q})\Big)\dot{q}_r + \tilde{G}(q) \\ &= \{\tilde{P}\}^T \bullet \{S(q)\}\ddot{q}_r + \tfrac{1}{2}\Big[\{\tilde{P}\}^T W_s(q, \dot{q})\Big]\dot{q}_r + W_0^T(q)\tilde{P}_0. \end{aligned} \tag{26}$$

Let I_0 be the set of integers, and $n_{ij} \in I_0$, $i = 1, \cdots, n$, $j = 1, \cdots, n$, $\Gamma_i = \Gamma_i^T = [\gamma_{i1} \quad \gamma_{i2} \cdots \gamma_{in}]$, $\gamma_{ij} \in \mathcal{R}^{m \times n_{ij}}$, $m = \sum_{j=1}^{n} n_{ij}$, $j = 1, 2, \cdots, n$, then the following equation is well defined: $\Gamma_i \bullet \{S_i\} = \{\Gamma_i\} \bullet \{S_i\} := [\gamma_{i1}S_{i1},\, \gamma_{i2}S_{i2},\, \cdots,\, \gamma_{in}S_{in}] \in R^{m \times n}$.

The stability property of the system (26) is given by the following theorem:

Theorem 1: If $K > 0$, $K_i = K_i^T \ge 0$, the closed-loop system (26) is asymptotically stable, i.e., e and $\dot{e} \to 0$ as $t \to \infty$ under the following parameter adaptation

$$\begin{aligned} \dot{\hat{P}}_i = \Gamma_i \Big\{ &\bullet \{S_i(q)\}\ddot{q}_r r_i + \bullet\{W_{1i}(q, \dot{q})\}\dot{q}_r \frac{r_i}{2} \\ &+ \sum_{k=1}^{n} \bullet\{w_{i*k}(q)\}\dot{q} \frac{\dot{q}_{rk} r_i - \dot{q} r_k \dot{q}_{ri}}{2} \Big\} \end{aligned} \tag{27}$$

$$\dot{\hat{P}}_0 = \Gamma_0(q)W_0 r \tag{28}$$

where $\Gamma_0 = \Gamma_0^T > 0$ and $\Gamma_i = \Gamma_i^T > 0$. In addition, all the closed-loop signals, $\hat{P}_i$, $\hat{P}_0$, and τ are bounded.

Proof: See [12].

Remark 1: If Γ_i is defined as $\Gamma_i = \mathrm{diag}[\Gamma_{ij}]$ with Γ_{ij}, $1 \le j \le n$ being dimensional compatible matrix blocks, then, the adaptation law (27) can be written as

$$\begin{aligned} \dot{\hat{P}}_{ij} = \Gamma_{ij} S_{ij}(q)\ddot{q}_{rj} r_i &+ \{\Gamma_{ij}\} \bullet \{w_{ij*}(q)\}\dot{q}\dot{q}_{rj}\frac{r_i}{2} \\ &+ \sum_{k=1}^{n} \Gamma_{ij} w_{ijk}(q)\dot{q}_j \frac{\dot{q}_{rk} r_i - r_k \dot{q}_{ri}}{2}. \end{aligned} \tag{29}$$

Remark 2: Since the network models are valid for all kinds of robots having the same number of DOF and the same sequence of joint types, the modeling approach is independent of the physical structures of the systems. A controller designed for one robot can be used to control other robots having the same numbers of DOF and the same sequence of joint types, thus the period of control system development is shortened. Because of the systematic modeling approach presented, a program can be written to automatically generate the codes when the user is prompted to key in the number of DOF and the sequence of joint types of a given robot. There is no need for the tedious and error prone process of dynamic modeling as commonly being done, especially for higher degrees-of-freedom robots. Subsequently, executable codes can be generated (and down loaded to DSP motion control cards) for real-time control as closed-loop stability is guaranteed.

IV. Simulation Tests

For the simulation studies, we consider a planar two-link manipulator, whose dynamics are described (9) with $d_{11} = m_1 + m_2 + 2m_3 \cos q_2$, $d_{12} = d_{21} = m_2 + m_3 \cos q_2$, $d_{22} = m_2$, $c_{11} = -m_3\dot{q}_2 \sin q_2$, $c_{12} = -m_3(\dot{q}_1 + \dot{q}_2)\sin q_2$, $c_{21} = m_3\dot{q}_1 \sin q_2$, $c_{22} = 0.0$, and $g_1 = m_4 \cos q_1 + m_5 \cos(q_1 + q_2)$, $g_2 = m_5 \cos(q_1 + q_2)$ where the m_i's are parameters of interest. To illustrate the steps in controller design for robots, construction of the networks is given for the two degrees-of-freedom robot. In fact, the system is completely described by the set

$$\begin{aligned} \overline{H}_{kj} = \{&\sin q_i,\, \cos q_i \sin q_i,\, \sin q_j \sin q_i \\ &\cos q_j,\, \cos q_i \cos q_j,\, i, j = 1, 2\}. \end{aligned}$$

In practice, however, only a subset of all the possible elements is enough to describe the system. For example, we may have $H_{kj} = \{\sin q_1,\, \cos q_2,\, \sin q_1 \sin q_2,\, \sin q_1 \cos q_2,\, \sin q_2 \cos q_1,\, \cos q_1 \cos q_2\}$ and choose $S_{11}(q) = [1, \cos q_1, \cos q_2]^T$, $S_{12}(q) = S_{21}(q) = [1, \sin q_1, \cos q_2]^T$, and $S_{22}(q) = [1, \cos q_2]^T$. They form the "basis" functions for $D(q) = [\{P\}^T \bullet \{S(q)\}]$ with $\{P\}$ being the parameters of interest: $P_{11} = [m_1 + m_2, 0, 2m_2]^T$, $P_{12} = P_{21} = [m_2, 0, m_2]^T$, and $P_{22} = [m_2, 0]^T$. Clearly, $D(q)$ is over parametrized because of the presence of the 0 elements in $\{P\}$. From (6),

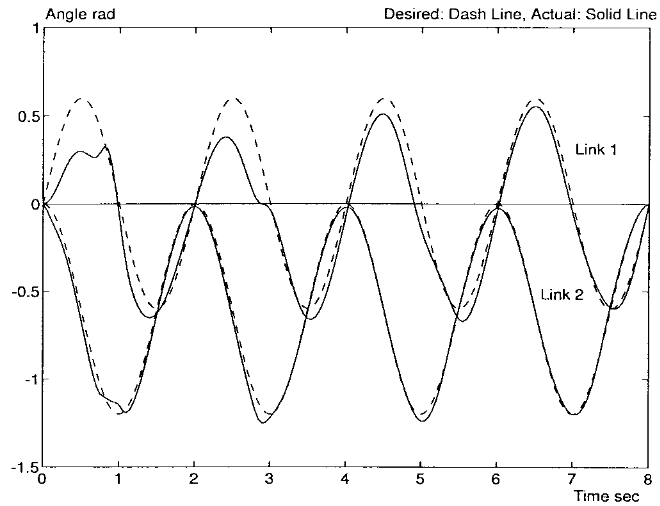

Fig. 1. Position tracking with adaptation.

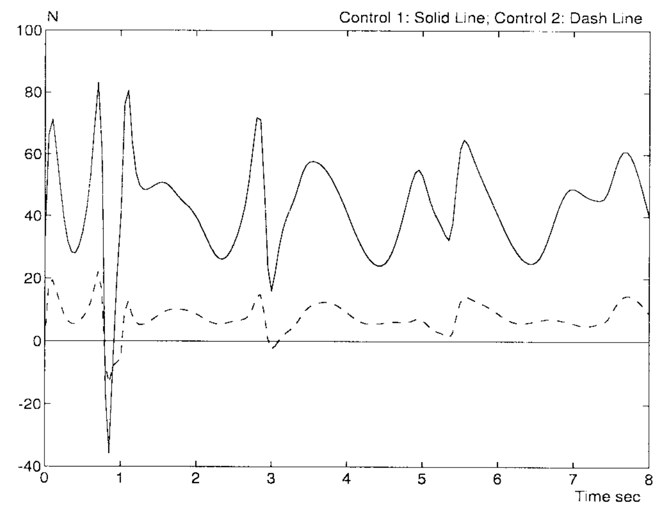

Fig. 2. Control with adaptation.

we have $w_{111}(q) = [0, -\sin q_1, 0]^T$, $w_{112}(q) = [0, 0, -\sin q_2]^T$, $w_{121}(q) = [0, \cos q_1, 0]^T$, $w_{122}(q) = [0, 0, -\sin q_2]^T$, $w_{211}(q) = w_{121}(q)$, $w_{212}(q) = w_{122}(q)$, $w_{221}(q) = [0, 0]^T$, $w_{222}(q) = [0, -\sin q_2]^T$. Subsequently $C(q, \dot{q})$ can be constructed from these $w_{ijk}(q)$ and $\{P\}$ as given by (11). For the network of the potential energy $V(q)$, we may choose

$$S_0(q) = [\sin q_1, \sin q_2, \sin q_1 \cos q_2, \cos q_1 \sin q_2]^T \tag{30}$$

we have $w_{01}(q) = [\cos q_1, 0, \cos q_1 \cos q_2, -\sin q_1 \sin q_2]^T$, $w_{02}(q) = [0, \cos q_2, -\sin q_1 \sin q_2, \cos q_1 \cos q_2]^T$ and $G(q)$ can be described by $G(q) = W_0^T(q) P_0$ with $P_0 = [m_4, 0, m_5, m_5]^T$. Once the networks are being built, controllers can be generated accordingly.

In the simulation results, $\hat{P}_{ijk}$ and $\hat{P}_{0k}$ stand for the kth elements of the vectors $\hat{P}_{ij}$ and $\hat{P}_0$. Suppose initially that we do not have any knowledge about the system, i.e., $\hat{P}_{11}(0) = \hat{P}_{12}(0) = \hat{P}_{21}(0) = 0$, $\hat{P}_{22}(0) = 0$, $\hat{P}_0(0) = 0$. The desired trajectory $q_d(t) = [0.6 \sin(\pi t), 0.6(\cos(\pi t) - 1)]^T$ which guarantees continuous bounded position, velocity and acceleration. The controller gains $K = \text{diag}[10.0]$, $\Lambda = \text{diag}[5.0]$, and the adaptive gains $\Gamma_0 = \text{diag}[10.0]$, $\Gamma_i = \text{diag}[2.0]$. The position tracking performance of the robot is as shown in Fig. 1 and the control inputs are shown in Fig. 2. The corresponding parameter adaptations are shown in Fig. 3.

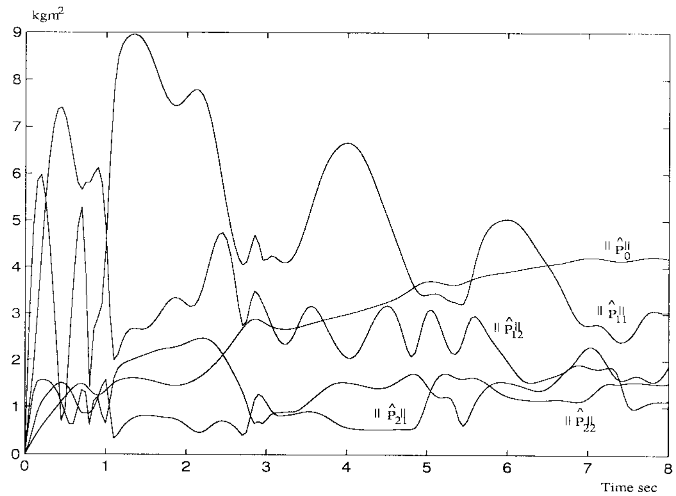

Fig. 3. Parameters $\|\hat{P}_{ij}\|$ and $\|\hat{P}_0\|/10$ with adaptation.

The simulation results thus demonstrate that the proposed adaptive network control can effectively handle changes in the dynamics of the system.

By extensive computer simulation, it was found that the control performance becomes better by repeatedly setting current $\hat{P}_{ij}(0)$ and $\hat{P}_0(0)$ to $\hat{P}_{ij}(t_d)$ and $\hat{P}_0(t_d)$ of last simulation. Therefore, for better control performance, the controller can be "trained" first by test runs before it is used for actual control operation. As the number of trials increases, the tracking performance becomes better and better, the parameter variations and control signals are smoother and smaller.

V. Conclusion

In this paper, a systematic approach for controller design was presented based on the fact that all the elements of the inertia matrix $D(q)$ and the potential energy $V(q)$ [or the gravitational forces $G(q)$] can be precisely parametrized by all possible combinations of known functions of positions. The well known structural properties for the new model, such as linear-in-the-parameters dynamics, were examined. Subsequently, they were used for controller design. It has been shown that the closed-loop system is asymptotically stable by appropriately adjusting the weights of the networks on-line.

References

[1] R. H. Middleton and G. C. Goodwin, "Adaptive computed torque control of rigid link manipulators," *Syst. Contr. Lett.*, vol. 10, pp. 9–16, 1988.
[2] M. W. Spong and R. Ortega, "On adaptive inverse dynamics control of rigid robots," *IEEE Trans. Automat. Contr.*, vol. 35, pp. 92–95, Jan. 1990.
[3] K.-K. D. Young, "A variable structure model following control design for robotics applications," *IEEE J. Robot. Automat.*, vol. 4, pp. 556–561, Oct. 1988.
[4] S. N. Singh, "Adaptive model following control of nonlinear robotic systems," *IEEE Trans. Automat. Contr.*, vol. AC-30, Nov. 1985.
[5] J.-J. E. Slotine and W. Li, "Adaptive manipulator control: A case study," *IEEE Trans. Automat. Contr.*, vol. 33, Nov. 1988.
[6] N. Sadegh and R. Horowitz, "An exponential stable adaptive control law for robotic manipulators," *IEEE Trans. Robot. Automat.*, vol. 6, Aug. 1990.
[7] K. S. Fu, R. C. Gonzalez, and C. S. G. Lee, *Robotics: Control, Sensing, Vision and Intelligence.* New York: McGraw-Hill, 1987.
[8] M. W. Spong and M. Vidyasagar, *Robot Dynamics and Control.* New York: Wiley, 1988.
[9] F. L. Lewis, C. T. Abdallah, and D. M. Dawson, *Control of Robot Manipulators.* New York: Macmillan, 1993.
[10] J. J. E. Slotine and W. Li, *Applied Nonlinear Control.* Englewood Cliffs, NJ: Prentice-Hall, 1991.

[11] F. L. Lewis, K. Liu, and A. Yesildirek, "Neural net robot controller with guaranteed tracking performance," *IEEE Trans. Neural Networks*, vol. 6, pp. 703–715, May 1995.
[12] S. S. Ge, T. H. Lee, and C. J. Harris, *Adaptive Neural Network Control of Robotic Manipulators.* Singapore: World Scientific, 1998.
[13] S. S. Ge and I. Postlethwaite, "Adaptive neural network controller design for flexible joint robots using singular perturbation technique," *Trans. Inst. Meas. Contr.*, vol. 17, no. 3, pp. 120–131, 1995.

Theory of Two-Dimensional Transformations

Yutaka J. Kanayama and Gary W. Krahn

***Abstract*—This paper proposes a new "heterogeneous" two-dimensional (2-D) transformation group $\langle \mathcal{T}, \circ \rangle$ to solve motion analysis/planning problems in robotics. In this theory, we use a 3 × 1 matrix to represent a transformation as opposed to a 3 × 3 matrix in the homogeneous formulation. First, this theory is as capable as the homogeneous theory. Because of the minimal size, its implementation requires less memory space and less computation time, and it does not have the rotational matrix inconsistency problem. Furthermore, the raw rotation angle θ is more useful than the trigonometric values, cos θ and sin θ, in the homogeneous transformations. This paper also discusses how to apply the group $\langle \mathcal{T}, \circ \rangle$ to solve problems related to motion analysis/planning, trajectory generation, and others. This heterogeneous formulation has been successfully implemented in the MML software system for the autonomous mobile robot Yamabico-11 developed at the Naval Postgraduate School.**

***Index Terms*—Group theory, heterogeneous transformations, trajectory generation, transformation, two-dimensional transformation.**

I. Introduction

The three-dimensional (3-D) homogeneous transformation theory has been extensively used in the robotics field [1], [2]. Therefore, when we need a two-dimensional (2-D) transformation theory to deal with the problems in mobile robot motion control or computer graphics, a natural consequence is to apply the 2-D version of the 3-D homogeneous transformation formalism [3].

The general format of 2-D homogeneous transformations is

$$T = \begin{bmatrix} R_{11} & R_{12} & x \\ R_{21} & R_{22} & y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & x \\ \sin\theta & \cos\theta & y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where a transformation is represented by a 3 × 3 matrix. As opposed to this classical method, in this paper, we propose a new 2-D transformation group $\langle \mathcal{T}, \circ \rangle$ [4]. Each transformation in this theory is the 3 × 1 matrix $q = (x, y, \theta)^T$, where $x, y, \theta \in \Re$. Since a transformation in a plane has three degrees of freedom (two for translation and one for rotation), this 3 × 1 form is the minimal mathematical structure we need. Therefore, if this new transformation theory is as capable as the homogeneous theory, we can expect this theory to be an optimal one. The rotational information is contained in a 2 × 2 matrix in the homogeneous theory, however, in the proposed theory it is contained in one real number θ in the proposed theory. Since we do not use the "homogeneous" matrix operations in this new theory, we call this new formulation *heterogeneous*.

This paper reports four major results on the heterogeneous 2-D transformation theory:

1) a formulation that is as capable as the homogeneous transformation;
2) this formulation requires less memory space and less computation time than the homogeneous formulation and it does not have the rotation matrix consistency problem;
3) the explicit angle representation θ carries more information;
4) there are several applications of this theory in autonomous vehicle motion control.

More detailed discussions are given in Section III.

In the motion planning research, the concept of "configuration," (x, y, θ) has been widely used [5]. It should be noted that this concept is distinct from that of the heterogeneous transformations. A configuration describes the "static" positioning of a rigid body where $(x, y, 0)$ and $(x, y, 2\pi)$ are equivalent. For a transformation of a rigid body, however, there is a clear distinction between a no-rotation and a 2π-rotation.

In the 3-D transformations, to avoid its singular point, the quaternian algebra was introduced [6], [7]. In the 2-D transformations, however, there is no singular point in both the homogeneous and heterogeneous formulations and we do not need any special consideration in this respect.

The authors have implemented this heterogeneous transformation formulation as a part of the MML software system for the autonomous robot "Yamabico-11" at the Naval Postgraduate School [8], [9]. The simple function set (the inverse and composition functions) solves numerous motion planning/analyzing problems including the examples described in Section IV.

II. Heterogeneous Transformation Theory

Let $\Re$ denote the set of all real numbers. A *transformation*, q, is defined by

$$q \equiv \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} \tag{2}$$

where $x, y, \theta \in \Re$. The set of all transformations is denoted by $\mathcal{T}$. For example, $(2, 1, \pi/6)^T$, $(2, 4, \pi/4)^T \in \mathcal{T}$ (M^T denotes the transposition of the matrix M). A transformation q is interpreted as a 2-D coordinate transformation from one Cartesian coordinate system to another. Furthermore, q is interpreted as a composition of a translational transformation (x, y) and a rotational transformation θ.

Definition: The transformation group $\langle \mathcal{T}, \circ \rangle$ consists of the set $\mathcal{T}$ of transformations, where

$$\mathcal{T} = \{(x, y, \theta)^T | x, y, \theta \in \Re\}$$

and the binary operator (*composition function*), $\circ$, is defined as follows: Let $q_1 = (x_1, y_1, \theta_1)^T$, $q_2 = (x_2, y_2, \theta_2)^T \in \langle \mathcal{T}, \circ \rangle$, then

$$q_1 \circ q_2 \equiv \begin{pmatrix} x_1 + x_2 \cos\theta_1 - y_2 \sin\theta_1 \\ y_1 + x_2 \sin\theta_1 + y_2 \cos\theta_1 \\ \theta_1 + \theta_2 \end{pmatrix}. \tag{3}$$

Manuscript received July 19, 1994; revised June 1, 1998. This paper was recommended for publication by Associate Editors Y. Nakamura, V. Kumar, and A. De Luca and Editor A. Goldenberg upon evaluation of the reviewers' comments.

Y. Kanayama is with the Computer Science Department, Naval Postgraduate School, Monterey, CA 93943 USA (e-mail: kanayama@cs.nps. navy.mil).
G. Krahn is with the Department of Mathematical Sciences, United States Military Academy, West Point, NY 10996 USA (e-mail: ag2609@usma2. usma.edu).

Publisher Item Identifier S 1042-296X(98)07319-4.