

# Dynamic Identification of an Industrial Seven-Link Manipulator by Evolving Runge–Kutta–Gill RBF Networks

Thrishanta Nanayakkara\*, Keigo Watanabe\*\*, Kazuo Kiguchi\*\* and Kiyotaka Izumi\*\*\*

\*Graduate School of Science and Engineering,

Faculty of Engineering Systems and Technology,

\*\*Department of Advanced Systems Control Engineering,

Graduate School of Science and Engineering,

\*\*\*Department of Mechanical Engineering,

Faculty of Science and Engineering,

Saga University, 1-Honjomachi, Saga 840-8502, Japan

†E-mail: watanabe@me.saga-u.ac.jp

## Abstract

This paper proposes a method for identification of dynamics of a multi-link robot manipulator using Runge-Kutta-Gill neural networks (RKGNNs). Shape adaptive radial basis function (RBF) NNs have been employed with an evolutionary algorithm to optimize the shape parameters and the weights of the RKGNN. Unlike in conventional methods, the proposed method can even be used without input torque information because a torque network is part of the functional network. The method has been employed for dynamic identification of a seven-link industrial manipulator called PA-10, manufactured by the Mitsubishi Heavy Industries Ltd.

**Keywords:** Runge-Kutta-Gill neural networks, Radial basis functions, Multi-link robot manipulators, Evolutionary optimization.

## 1 Introduction

Stability and accuracy of model based controllers such as computed torque algorithms demands for effective identification of dynamics of manipulators [1]. In most of the industrial robot manipulators, the parameters of the dynamics are vaguely known or incompletely known to design such control algorithms. Identification of inertia parameters using the least squares algorithm has been investigated in [2]. These methods have the problems of finding a globally optimum solution and the difficulty of identifying a complex system with high degrees-of-freedom due to the backpropagation algorithm to find weights of NNs. Moreover these methods use input torque information for the calculations that can be expensive on one hand and on the other hand, needs special hardware changes.

Therefore the motivation of this research has been to develop a method that is capable of identifying the

dynamics of a complex multi-link robot manipulator with basic reference inputs of joint angle and its velocity, and their output data.

The proposed method uses Runge–Kutta–Gill neural networks (RKGNNs), which is an extension version of Runge–Kutta neural networks (RKNNs), for the identification of the dynamics, because RKNNs are capable of identifying the changing rates of the states accurately due to the state space interpolation between one sampling interval [3]. Furthermore, this method only needs one state information to predict the next state and the trained NN of the function is independent of the sampling time interval unlike in direct mapping neural networks (DMNN). The function NN has sub networks to represent the components of the dynamics of the manipulator [1]. Radial basis function (RBF) NNs have been used for each sub-network. Due to the complex structure of the dynamics of a manipulator, the number of weights of the total NN system demands for efficient optimization techniques to train the networks. Therefore optimization of weights using a new evolutionary optimization algorithm [4] has been proposed in this paper. Due to the property of monotonically reducing standard deviation of mutation, the convergence rate is fast in the evolutionary algorithm. In the proposed algorithm, sustaining adequate standard deviation of mutation throughout the optimization process guarantees the convergence to a global optimality.

PA-10 manipulator with seven-links has been subjected to the dynamics identification study. Experimental data of basic reference input position, its velocity and their output data have been used to train the RKGNNs.

Rest of this paper is organized as follows: In section 2, a brief introduction to the basic components of the dynamics of a multi-link robot manipulator and the problems in a practical dynamics identification s-

tudy is explained. In section 3, the RKGNNs and the structure of the function NNs are explained. In section 4, the procedure of training the NNs for PA-10 manipulator is explained. In section 5, the results are discussed.

## 2 Dynamics of a manipulator

Dynamics of a manipulator is given by

$$\boldsymbol{\tau} = M(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + V(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + G(\boldsymbol{\theta}) + F(\boldsymbol{\theta}) \quad (1)$$

where  $\boldsymbol{\tau}$  is the joint torque vector,  $\boldsymbol{\theta}$  is the joint angle vector,  $\dot{\boldsymbol{\theta}}$ ,  $\ddot{\boldsymbol{\theta}}$  are joint angular velocity and angular acceleration vectors respectively,  $M(\boldsymbol{\theta})$  is the inertia matrix,  $V(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})$  is the Coriolis and centrifugal force vector,  $G(\boldsymbol{\theta})$  is the gravity force vector, and  $F(\boldsymbol{\theta})$  is the friction vector [1].

It can be seen in [1] that the derivation of this dynamic equation for a manipulator with higher degrees of freedom such as PA-10 manipulator, is very complex and time consuming. Even if the dynamic equation is known for such a manipulator, many crucial parameters such as link inertia are unknown or partially known. The identification of these unknown parameters involves data collection including the input torque sensor information if conventional least-squares method [2] or DMNN approach is employed. Yet in most practical situations, the torque sensors are not built-in with the manipulator. Therefore if torque sensors are to be used, some hardware changes are required that makes additional trouble and expense.

The proposed method attempts to identify the dynamics involving only the reference input of joint angle and its velocity, and their output information, using the RKGNNs trained by an evolutionary algorithm. This needs rearranging of the dynamic equation in (1) to form an ordinary differential equation (ODE) of the form

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}). \quad (2)$$

Given the dynamics of the manipulator as in equation (1), the state vector  $\boldsymbol{x}$  can be defined by

$$\boldsymbol{x} = [\boldsymbol{\theta} \ \dot{\boldsymbol{\theta}}]^T. \quad (3)$$

Then the ODE given by (2) can be formed using  $\boldsymbol{x}$  and the dynamics defined by equation (1) as

$$\begin{bmatrix} \dot{\boldsymbol{\theta}} \\ \ddot{\boldsymbol{\theta}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta} \\ \dot{\boldsymbol{\theta}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ M^{-1}(\boldsymbol{\theta})\{\boldsymbol{\tau} - V(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dots\} \end{bmatrix}$$

The right-hand side of the above ODE gives the rate of change of the states given by the left-hand side. Therefore for this family of ODEs, RKGNNs may very accurately identify the right hand side.

## 3 RKGNNs and the structure of the function networks.

**Figure 1** shows the basic RKGNN algorithm. From Fig. 1, it can be seen that the prediction of the next state involves one previous state information and the sampling interval is external to the function NN model. One other feature is that all four NN models given by  $N_f$  in the RKGNN are the same. Therefore finding the optimum weights for one  $N_f$  is all that is required to identify the system. The RKGNN algorithm contains the following steps.

Step 1:

$$\mathbf{k}_0 = hN_f(\boldsymbol{x}(i))$$

$$\boldsymbol{x}^{(1)}(i+1) = \boldsymbol{x}(i) + \frac{1}{2}\mathbf{k}_0$$

$$\mathbf{q}_1 = \mathbf{k}_0$$

Step 2:

$$\mathbf{k}_1 = hN_f(\boldsymbol{x}^{(1)}(i+1))$$

$$\boldsymbol{x}^{(2)}(i+1) = \boldsymbol{x}^{(1)}(i+1) + \alpha(\mathbf{k}_1 - \mathbf{q}_1)$$

$$\mathbf{q}_2 = \beta\mathbf{q}_1 + e\mathbf{k}_1$$

Step 3:

$$\mathbf{k}_2 = hN_f(\boldsymbol{x}^{(2)}(i+1))$$

$$\boldsymbol{x}^{(3)}(i+1) = \boldsymbol{x}^{(2)}(i+1) + d(\mathbf{k}_2 - \mathbf{q}_2)$$

$$\mathbf{q}_3 = \gamma\mathbf{q}_2 + g\mathbf{k}_2$$

Step 4:

$$\mathbf{k}_3 = hN_f(\boldsymbol{x}^{(3)}(i+1))$$

$$\boldsymbol{x}^{(4)}(i+1) = \boldsymbol{x}^{(3)}(i+1) + \frac{1}{6}\mathbf{k}_3 - \frac{1}{3}\mathbf{q}_3$$

$$\boldsymbol{x}(i+1) \equiv \boldsymbol{x}^{(4)}(i+1)$$

In this case,  $h$  is the time step width,  $d = (\sqrt{2} + 1)/\sqrt{2}$ ,  $e = (2 - \sqrt{2})$ ,  $g = (2 + \sqrt{2})$ ,  $\alpha = (\sqrt{2} - 1)/\sqrt{2}$ ,  $\beta = -2 + 3/\sqrt{2}$ , and  $\gamma = -(2 + 3/\sqrt{2})$ .

The function NN  $N_f(\cdot)$  as shown in **Fig. 2** contains components for the identification of each component of the dynamic equation as shown in equation (1). Since information about the controller is unknown, a torque network is constructed that will grasp the dynamics of the controller.

The details of the component networks of inertia, velocity including the Coriolis and centrifugal terms, and gravity are presented in [5]. The construction of

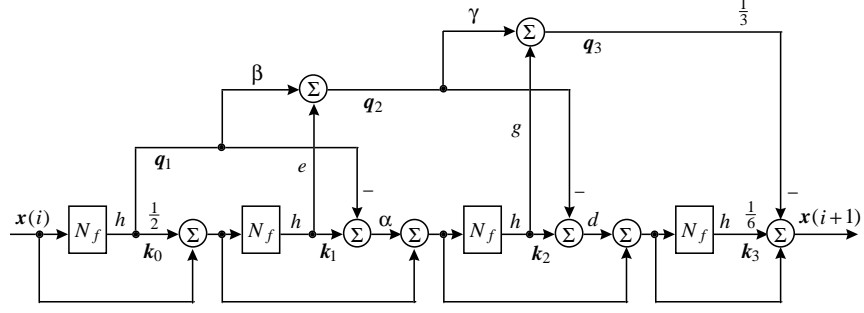


Figure 1: Structure of the fourth-order RKGNN by the function neural networks  $N_f$ .

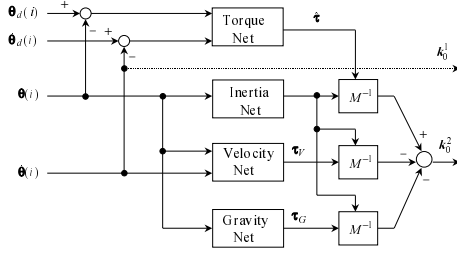


Figure 2: Basic concept of the function NN to identify each component of the dynamic equation.

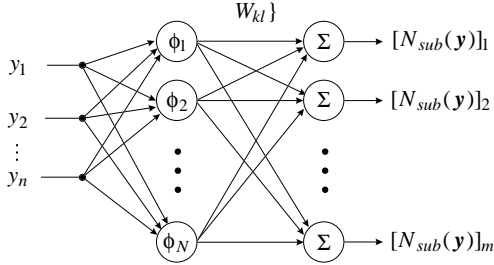


Figure 3: Basic structure of the RBF NN.

sub-networks of each of these component networks involve construction of shape adaptive RBF NNs as can be seen in **Fig. 3**.

The input-output mapping relationship of the RBF in this application is given by

$$\phi_l(\mathbf{y}; \bar{\mathbf{y}}_l, \sigma_l) = \exp\{-[(\mathbf{y} - \bar{\mathbf{y}}_l)^T(\mathbf{y} - \bar{\mathbf{y}}_l)/\sigma_l^2]\} \quad (4)$$

where  $\mathbf{y} \in \mathbb{R}^n$  and  $\bar{\mathbf{y}}_l \in \mathbb{R}^n$  are the input vector and the vector of centers of the radial basis function,  $\sigma_l^2$  is the variance of the  $l$ th RBF. Therefore the output at the  $k$ th output node of the sub-network  $N_{sub}(\cdot)$  is

given by

$$[N_{sub}(\mathbf{y})]_k = \sum_{l=1}^N W_{kl} \phi_l(\mathbf{y}; \bar{\mathbf{y}}_l, \sigma_l), k = 1, \dots, m \quad (5)$$

where  $N$  is the number of RBFs that construct the sub-network and  $W_{kl}$  is the weight from  $l$ th RBF to  $k$ th output node.

## 4 Identification of dynamics of PA-10 manipulator

The PA-10 manipulator consists of seven links with seven degrees of freedom (three rotation axes and four pivot axes) as illustrated in **Fig. 4**. The main specification of the PA-10 manipulator are as follows: the length of the manipulator is 1345 mm, the weight of the manipulator is 35 kgf, the maximum combined speed with all axes is 1.55 m/s, the payload weight is 10 kgf, and output torque is 9.8 Nm. The input reference angle, its velocity and their output data of all seven joints were experimentally obtained. Then these experimental data were used to train the RKGNNs to identify the dynamics of the manipulator. The weights of the RKGNN and the RBF parameters,  $\bar{\mathbf{y}}_l$  and  $\sigma_l$ , were found using an evolutionary algorithm so as to minimize the cost function given by

$$J = \frac{1}{t_N} \sum_{i=1}^{t_N} \|\theta_d(i) - \theta(i)\| + \|\dot{\theta}_d(i) - \dot{\theta}(i)\| + P \quad (6)$$

where  $t_N$  is the total time span of the trajectory,  $P$  is a penalty. If the minimum of the principle minors of the inertia matrix  $M(\theta)$  estimated by the network is negative,  $P$  is given the negative value of the minimum of the principle minors of the inertia matrix  $M(\theta)$  estimated by the network. Otherwise  $P$  is set to zero.

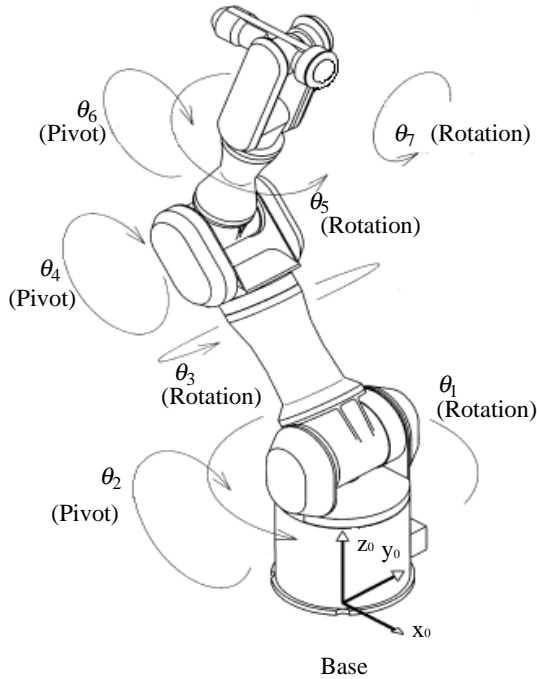


Figure 4: Mechanical structure of the PA-10 manipulator.

The reason to add this penalty is to make sure that the estimated inertia matrix is symmetric and positive definite. The conditions of the usage of evolutionary algorithm were as follows: 100% mutation and 20% crossover was employed. The tournament size was 15 and the number of individuals were 100. One individual consists of 3080 elements. The algorithm was run for 200 generations.

## 5 Results and discussion

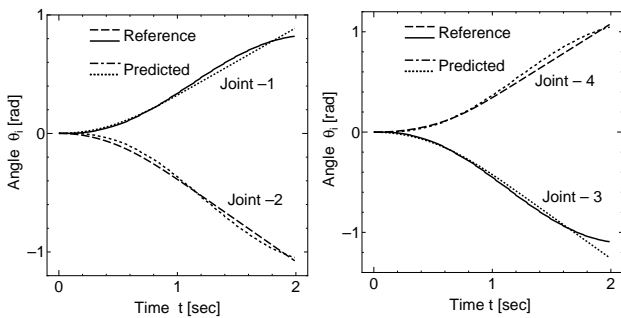


Figure 5: The reference and predicted joint angular trajectories. On the left is that for joint 1 and 2, and on the right is that for joint 3 and 4.

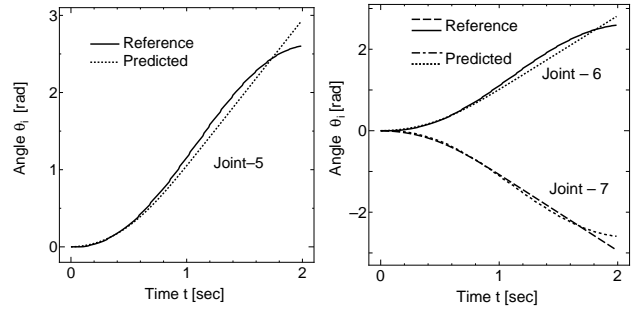


Figure 6: The reference and predicted angular trajectories. On the left is that for joint 5, and on the right is that for joint 6 and 7.

The RKGNN has been developed and it was applied for identifying the complex dynamics of a seven-link industrial manipulator called PA-10, only using the input angle, its velocity and their output data. It is very useful, because it reduces the burden of deriving the complex dynamic equations for such manipulators. The adoption of RKGNNs makes the long term prediction of the states very accurate due to its property of interpolation of states within one sampling interval and estimating the rate of change of states accurately. This method is advantageous in the practical situation where input torque information is unknown.

## References

- [1] J. J. Craig, *Introduction to Robotics: Mechanics and control*, 2nd edn, Reading, MA: Addison-Wesley, 1989.
- [2] C. H. An, C. G. Atkeson, J. M. Hollerbach, "Estimation of Internal Parameters of Rigid Body Links of Manipulators," *Artificial Intelligence Memo 887, MIT Artificial Intelligence Laboratory*, 1986.
- [3] Y.-J. Wang and C.-T. Lin, "Runge-Kutta Neural Network for Identification of Dynamical Systems in High Accuracy," *IEEE Trans. on Neural Networks*, vol. 9, no. 2, pp. 294–307, 1998.
- [4] T. Nanayakkara, K. Watanabe and K. Izumi, "Evolving in Dynamic Environments Through Adaptive Chaotic Mutation," in *Proc. of Fourth International Symposium on Artificial Life and Robotics*, vol. 2, 1999, pp. 520–523.
- [5] T. Nanayakkara, K. Watanabe, and K. Izumi, "Evolving Runge-Kutta-Gill RBF Networks to Estimate the Dynamics of a Multi-Link Manipulator," in *Proc. of IEEE International Conference on Systems, Man, and Cybernetics*, vol. 2, 1999, pp. 770–775.