

# Controlling Multi-Link Manipulators by Fuzzy Selection of Dynamic Models

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

\*Faculty of Engineering Systems and Technology,  
Graduate School of Science and Engineering,

\*\*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, kiguchi, izumi}@me.saga-u.ac.jp

## Abstract

*A method for the identification of complex non-linear dynamics of a multi-link robot manipulator using Runge-Kutta-Gill Neural Networks (RKGNNs) in the absence of input torque information is proposed. The RKGNNs constructed using shape adaptive radial basis functions (RBF) are trained using an evolutionary algorithm. Due to the fact that the main function network is divided into sub-networks to represent detailed properties of the dynamics of a manipulator, the neural networks have greater information processing capacity and they can be tested for properties such as positive definiteness of the inertia matrix. Dynamics of a three-link manipulator are identified using only input-output position and their velocity data, and promising control results are obtained to prove the ability of the proposed method in capturing highly nonlinear dynamics of a multi-link manipulator in an effective manner.*

## 1 Introduction

The ability to grasp the full dynamics of manipulators accurately is of much importance in judging the robustness of model-based control strategies such as computed torque controllers. The problem often encountered in the identification of dynamics of multi-link industrial manipulators is the lack of knowledge of necessary data such as input joint torques or the gains of the servo controllers. Identification of inertia parameters using the least-squares algorithm has been investigated in [1]. Yet these methods need the input joint torque information which needs special hardware changes in an industrial manipulator. Identification of dynamics of manipulators using ANNs has been studied in [2], [?]. These methods suffer from several drawbacks: one is the tendency of backpropagation algo-

rithms to converge to local minima, another is that in some of these methods the trained NN depends on the sampling time width of the training data and the other is that these methods do not make use of the inherent structure of the dynamics of manipulators. Moreover the simulation results are shown only for two link manipulators, where the nonlinearities of the dynamics are relatively simple compared with the actual industrial manipulators. Therefore the ability of these methods to accurately grasp the dynamics of a manipulator with high degrees-of-freedom is questionable.

The proposed method synthesizes a function NN consisting of sub-networks, so that the sub-networks represent the components of the dynamics of the manipulator [3], to catch full dynamics of a manipulator. Shape adaptive RBFNNs have been used for each sub-network. The proposed method uses 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 [4]. 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 width unlike in direct mapping NNs. Weights of the NNs are optimized using a new evolutionary algorithm [5].

Simulation studies are carried out for a three-link robot arm. It is assumed that the servo controller can be represented by a Proportional Derivative (P-D) controller and the reference input of joint angle, its velocity, and their output data obtained from the P-D controller are used for the system identification studies. The RKGNNs are trained only using these data for two trajectories and the identified model is used to build two computed torque controllers for the respective trajectories and the manipulator is controlled

for a third trajectory using a fuzzy membership based model selection method. Promising results are obtained to prove the ability of the proposed method in capturing highly nonlinear dynamics of a multi-link manipulator in an effective manner.

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 study is given. In section 3, the RKGNNs and the structure of the function NNs are explained. In section 4, the features of the evolutionary algorithm used for this study are elaborated. In section 5, the controller design procedure is explained, section 6 gives a discussion of the results, and some concluding remarks are given in section 7.

## 2 Identifying the Dynamics of a Manipulator

### 2.1 Manipulator dynamic model and its properties

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  $\boldsymbol{\tau} \in \mathbb{R}^n$  is the joint torque vector,  $\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \ddot{\boldsymbol{\theta}} \in \mathbb{R}^n$  are the joint angle, angular velocity, and angular acceleration vectors respectively,  $M(\boldsymbol{\theta}) \in \mathbb{R}^{n \times n}$  is the inertia matrix,  $V(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \in \mathbb{R}^n$  is the centrifugal and Coriolis force vector,  $G(\boldsymbol{\theta}) \in \mathbb{R}^n$  is the gravity force vector, and  $F(\boldsymbol{\theta}) \in \mathbb{R}^n$  is the friction vector, where  $n$  is the number of joints of the manipulator [3].

It is well known that the rigid dynamics of equation (1) has the following properties.

*Property 1—Boundedness of the Inertia Matrix:* The inertia matrix  $M(\boldsymbol{\theta})$  is symmetric and positive definite and satisfies the following inequalities:

$$m_1 \|y\|^2 \leq y^T M(\boldsymbol{\theta}) y \leq m_2 \|y\|^2, \quad \forall y \in \mathbb{R}^n, \quad (2)$$

where  $m_1$  and  $m_2$  are known positive constants and  $\|\cdot\|$  denotes the standard Euclidean norm.

*Property 2—Skew Symmetry:* The inertia and centrifugal-Coriolis matrices have the following property:

$$y^T \left( \frac{1}{2} \dot{M}(\boldsymbol{\theta}) - C(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \right) y = 0, \quad \forall y \in \mathbb{R}^n, \quad (3)$$

where  $\dot{M}(\boldsymbol{\theta})$  is the time derivative of the inertia matrix and  $V(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) = C(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}})\dot{\boldsymbol{\theta}}$ .

### 2.2 The new concept for effective dynamics identification

The proposed method attempts to identify the dynamics involving only the reference input of the joint

angle, its velocity, and their output information, using the RKGNNs trained by an evolutionary algorithm. This needs rearranging of the dynamic equation (1) to form an ordinary differential equation (ODE) of the form

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}). \quad (4)$$

Given the dynamics of the manipulator as in equation (1), the state vector  $\boldsymbol{x}$  can be defined by  $\boldsymbol{x} = [\boldsymbol{\theta} \ \dot{\boldsymbol{\theta}}]^T$ . Then the ODE given by (4) can be formed using  $\boldsymbol{x}$  and the dynamics defined by equation (1) as

$$\begin{bmatrix} \dot{\boldsymbol{\theta}} \\ \ddot{\boldsymbol{\theta}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\theta} \\ \dot{\boldsymbol{\theta}} \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ M^{-1}(\boldsymbol{\theta})\{\boldsymbol{\tau} - V(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \dots\} \end{bmatrix} \quad (5)$$

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 RKGNN and the Structure of Function NNs

### 3.1 RKGNNs for the dynamics identification of manipulators

The basic structure of a RKGNN is depicted by **Fig. 1**. From **Fig. 1**, it can be seen that the prediction of the next state involves one previous state information and the sampling width 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:

$$\begin{aligned} \mathbf{k}_0 &= hN_f(\mathbf{x}(i)), \\ \mathbf{x}^{(1)}(i+1) &= \mathbf{x}(i) + \frac{1}{2}\mathbf{k}_0, \\ \mathbf{q}_1 &= \mathbf{k}_0, \end{aligned}$$

Step 2:

$$\begin{aligned} \mathbf{k}_1 &= hN_f(\mathbf{x}^{(1)}(i+1)), \\ \mathbf{x}^{(2)}(i+1) &= \mathbf{x}^{(1)}(i+1) + \alpha(\mathbf{k}_1 - \mathbf{q}_1), \\ \mathbf{q}_2 &= \beta\mathbf{q}_1 + e\mathbf{k}_1, \end{aligned}$$

Step 3:

$$\begin{aligned} \mathbf{k}_2 &= hN_f(\mathbf{x}^{(2)}(i+1)), \\ \mathbf{x}^{(3)}(i+1) &= \mathbf{x}^{(2)}(i+1) + d(\mathbf{k}_2 - \mathbf{q}_2), \\ \mathbf{q}_3 &= \gamma\mathbf{q}_2 + g\mathbf{k}_2, \end{aligned}$$

Step 4:

$$\begin{aligned} \mathbf{k}_3 &= hN_f(\mathbf{x}^{(3)}(i+1)), \\ \mathbf{x}^{(4)}(i+1) &= \mathbf{x}^{(3)}(i+1) + \frac{1}{6}\mathbf{k}_3 - \frac{1}{3}\mathbf{q}_3, \\ \mathbf{x}(i+1) &\equiv \mathbf{x}^{(4)}(i+1), \end{aligned}$$

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  $\gamma = -(2 + 3/\sqrt{2})$ .

Since information about the controller is unknown, a torque network is constructed that will grasp the dynamics of the controller.

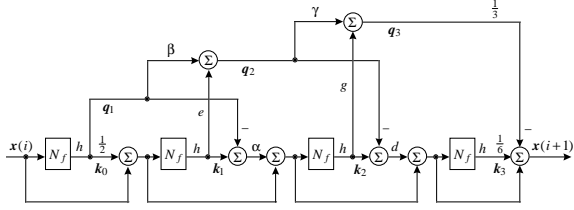


Figure 1: Alternative structure of the 4th-order RKGNN by the function NNs  $N_f$

It is very important to note that in the proposed method, the function NN  $N_f(\cdot)$  as shown in **Fig. 2** contains components for the identification of each component of the dynamic equation (1). The construction of sub-networks of each of these component networks involves the construction of shape adaptive RBFNNs.

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

$$\phi_l(\mathbf{y}; \bar{\mathbf{y}}_l, \sigma_l) = \exp\{-[(\mathbf{y} - \bar{\mathbf{y}}_l)^T(\mathbf{y} - \bar{\mathbf{y}}_l)/\sigma_l^2]\} \quad (6)$$

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

$$[N_{sub}(\mathbf{y})]_k = \sum_{l=1}^N W_{kl}\phi_l(\mathbf{y}; \bar{\mathbf{y}}_l, \sigma_l), k = 1, \dots, m \quad (7)$$

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.

## 3.2 The anatomy of the function NN

### 3.2.1 Torque network

The structure of the torque network is depicted by **Fig. 3**, where two sub-networks for the identification

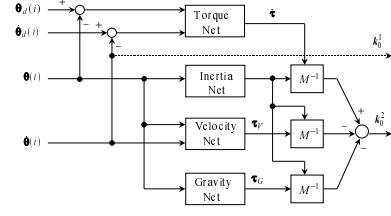


Figure 2: Basic concept of the function NN to identify each component of the dynamic equation.

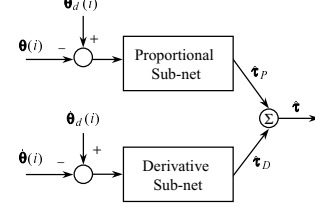


Figure 3: Torque network for identifying the proportional and derivative gains in a P-D servo controller.

of proportional and derivative gains of the actual controller are employed. The angular error vector given by  $(\boldsymbol{\theta}_d - \boldsymbol{\theta})$  and the angular velocity error vector given by  $(\dot{\boldsymbol{\theta}}_d - \dot{\boldsymbol{\theta}})$  are given as inputs to the proportional and derivative networks respectively to produce a suitable torque vector for the robot manipulator, where  $\boldsymbol{\theta}_d$  is the desired angular trajectory and  $\dot{\boldsymbol{\theta}}_d$  is the desired angular velocity. If detailed information of the controller is known, the structure of the network can be altered accordingly.

### 3.2.2 Inertia network

The structure of the sub-networks of the inertia network is designed such that they identify the diagonal and upper triangular components of the inertia matrix of the dynamic equation. This helps to make sure that the inertia matrix is symmetric and positive definite. The block diagram of the inertia sub-networks can be

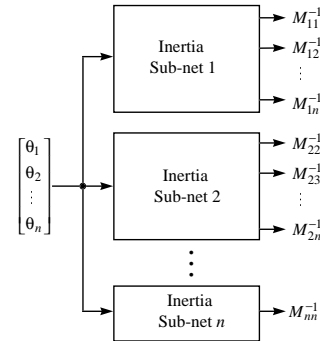


Figure 4: Inertia network considering the symmetry of the inertia matrix.

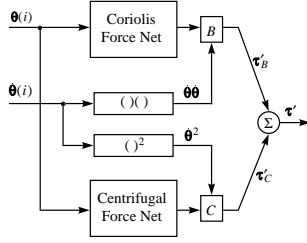


Figure 5: Velocity network consisting of the centrifugal and Coriolis force networks.

seen in **Fig. 4**.

### 3.2.3 Velocity network

The velocity network set which includes centrifugal and Coriolis components are shown in **Figs. 5**. Basically the output of the velocity network is given by

$$V(\theta, \dot{\theta}) = B(\theta)[\dot{\theta}\dot{\theta}] + C(\theta)[\dot{\theta}^2] \quad (8)$$

where  $B(\theta)$  is a matrix of size  $n \times n(n-1)/2$  and

$$[\dot{\theta}\dot{\theta}] = [\dot{\theta}_1\dot{\theta}_2 \quad \dot{\theta}_1\dot{\theta}_3 \quad \dots \quad \dot{\theta}_{n-1}\dot{\theta}_n]^T. \quad (9)$$

$C(\theta)$  is a matrix of size  $n \times n$ , and

$$[\dot{\theta}^2] = [\dot{\theta}_1^2 \quad \dot{\theta}_2^2 \quad \dots \quad \dot{\theta}_n^2]^T. \quad (10)$$

### 3.2.4 Gravity network

The gravity network consists of one RBF network with  $n$  RBFs and  $n$  outputs.

## 4 The Evolutionary Algorithm Used for Training the Networks

### 4.1 The representation

The individuals are defined based on real valued vectors in the form

$$\mathbf{a} = (\mathbf{x}, \boldsymbol{\sigma}) \quad (11)$$

to deal with continuous parameter optimization problems. Here  $\mathbf{x} \in \mathbb{R}^{n_x}$  is the object variable vector and  $\boldsymbol{\sigma} \in \mathbb{R}^{n_x}$  is the strategy parameter vector used for the mutation of individuals.

### 4.2 Recombination

In the proposed method, arithmetical crossover is adopted with a relatively small probability (e.g., 0.2).

### 4.3 Mechanism of mutation

A chaotic mutation mechanism was adopted [5].

## 4.4 Adaptive strategic moves based on the local knowledge

In this method, a short-term memory of  $n_p$  previous best individuals is kept. Then in each generation,  $\mu_p$  strategic moves are made as

$$x_{ij}^s(k+1) = U_{ij}(B_L, B_U)x_{ij}^*(k) - \sum_{r=1}^{n_p} x_{ij}^*(k-r) \quad (12)$$

where  $i = 1, \dots, \mu_p$ ,  $j = 1, \dots, n_x$ ,  $\mu_p$  is set to be a lower percentage of the population size  $\mu$ ,  $n_x$  is the total number of object variables in an individual,  $x_{ij}^s(k+1)$  is the  $j$ th element of the  $i$ th strategic estimate of the solution,  $x_{ij}^*(k)$  is the best solution of the current generation,  $x_{ij}^*(k-1), \dots, x_{ij}^*(k-n_p)$  are best solutions upto  $n_p$  previous generations.  $U_{ij}(B_L, B_U)$  is a uniform random number generator with lower bound  $B_L$  and upper bound  $B_U$ , resampled anew for each prediction.

Then depending on the fitness ranking of the population, the lowest  $\mu_p$  individuals are replaced by the new estimates.

## 5 Controller Design Procedure

### 5.1 Training the RKGNNs

The RKGNN was trained using the evolutionary algorithm described in section 4, with the objective function subjected to constraints [6]. Please note that in the evolutionary algorithm,  $B_U = B_L + 1$ . The constrained optimization method in [6] is augmented in this application to give relative weightage to each constraint as given by

$$J = \left( \frac{1}{t_N} \sum_{i=1}^{t_N} \|\boldsymbol{\theta}_d(i) - \boldsymbol{\theta}(i)\| + \|\dot{\boldsymbol{\theta}}_d(i) - \dot{\boldsymbol{\theta}}(i)\| \right) + \frac{s_t}{2} \left( \sum_{j=1}^3 \lambda_j [g_j^+(\boldsymbol{\theta})]^2 \right) \quad (13)$$

where  $t_N$  is the total time span of the trajectory,  $g_j^+(\boldsymbol{\theta}) = \max\{0, g_j(\boldsymbol{\theta})\}$  is the additional penalty function for the  $j$ th constraint satisfaction,  $s_t$  is a log penalty parameter such that  $s_t = 10 + \log(i+1)$ , and  $\lambda_j$  is the augmenting parameter for the  $j$ th constraint to depict the relative importance of the constraints. Three constraints are considered: one is that the inertia matrix estimated by the NN should be positive definite and the other two are the bounds of the inertia matrix. In the dynamics identification of a manipulator, the first constraint is very important to guarantee the stability of the model-based controller. Therefore the values of

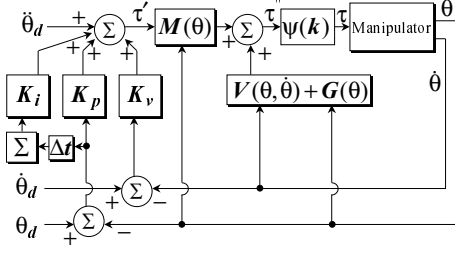


Figure 6: The structure of the computed torque controller.

the augmenting parameters selected were  $\lambda_1 = 1000$ ,  $\lambda_2 = 1$ , and  $\lambda_3 = 1$ .

The constraints are given by

$$g_1(\theta) : -\Gamma(\theta) \leq 0 \quad (14)$$

$$g_2(\theta) : m_1 \|y\|^2 - y^T M(\theta)y \leq 0, \quad \forall y \in \mathbb{R}^n \quad (15)$$

$$g_2(\theta) : y^T M(\theta)y - m_2 \|y\|^2 \leq 0, \quad \forall y \in \mathbb{R}^n \quad (16)$$

where  $\Gamma(\theta)$  is the minimum principle minor of the inertia matrix,  $m_1 = 1.23$  and  $m_2 = 1.45$ . The trained function NN was used to control the manipulator using the computed torque controller as shown in **Fig. 6**, where  $\theta_d$ ,  $\dot{\theta}_d$ , and  $\ddot{\theta}_d$  are the desired angle, angular velocity, and angular accelerations of the three joints, and  $\theta$ ,  $\dot{\theta}$ , and  $\ddot{\theta}$  are their respective following data. The addition of the discrete time dependent function  $\psi(k) = (1 - \exp(-0.03k))$ , where  $k$  denotes the sampling point, is important here. This function allows to damp the initial transients of the input torque produced by the controller due to the fact that the identified dynamics do not perfectly match the actual dynamics of the manipulator.

Computed torque controllers were developed for two trajectories marked 1 and 2 as shown in **Fig. 7**.

Then for a new reference trajectory, fuzzy membership based dynamic model selection method was adopted to control the manipulator.

Note that in the torque network, the number of RBFs used were 3 in each sub-network; in the inertia network, the number of RBFs used were 10, in each sub-network to identify the dynamics as accurately as possible; in the velocity network, the number of RBFs used were 3 in each sub-network; and the gravity network consists of 3 RBFs.

## 5.2 Fuzzy membership based dynamic model selection to control the manipulator

Define the premise variables for the identified trajectories at the  $k$ th sampling time step as,

$$\begin{aligned} z_i(k) &= [z_{i1}(k) \ z_{i2}(k) \ \cdots \ z_{ip}(k)] \\ &= [\theta_{i1} \ \theta_{i2} \ \theta_{i3} \ \dot{\theta}_{i1} \ \dot{\theta}_{i2} \ \dot{\theta}_{i3}] \end{aligned} \quad (17)$$

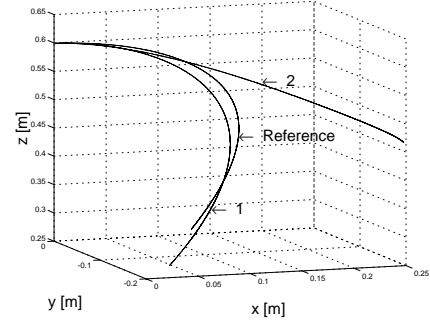


Figure 7: The trajectories used for training and the new reference trajectory.

and that for the new reference trajectory be given as

$$\begin{aligned} z_r(k) &= [z_{r1}(k) \ z_{r2}(k) \ \cdots \ z_{rp}(k)] \\ &= [\theta_{r1} \ \theta_{r2} \ \theta_{r3} \ \dot{\theta}_{r1} \ \dot{\theta}_{r2} \ \dot{\theta}_{r3}] \end{aligned} \quad (18)$$

where  $i=1, 2, p=6$ , and  $r$  denotes the reference trajectory.

Let the grade of membership of  $z_r(k)$  in the fuzzy set  $M_{ij}$ ,  $i = \{1, 2\}$ , be given as

$$M_{ij}(z_r(k)) = \exp(\ln(0.5)(z_{ij}(k) - z_{rj}(k))^2 \gamma L_j^2) \quad (19)$$

where  $L_j$  is the distance between the  $j$ th premise variables of two adjacent trajectories for which the dynamics are identified,  $\gamma$  is a constant set to be 0.66, that governs the width of fuzzy sets.

Then the control rule for the general case is given by,

$$\begin{aligned} \text{IF } i_{\max} &= \max[w_1(z(k)) \ \cdots \ w_q(z(k))] \\ \text{THEN } \tau &= [\tau]_{i_{\max}} \end{aligned}$$

where

$$w_i(z(k)) = \prod_{j=1}^p M_{ij}(z(k)) \quad (20)$$

Here,  $q$  denotes the number of preidentified models

## 6 Results

**Figure 7** shows the original two trajectories used for training the RKGNNs, marked 1 and 2, and the reference trajectory used to control the manipulator. The manipulator was controlled using fuzzy membership based selection of an appropriate dynamic model from these two identified NNs at any given time. **Figure 8** shows the trajectory-1 and the control result obtained using the computed torque controller as shown in **Fig. 6**. **Figure 9** shows the trajectory-2 and the control result obtained using the computed torque controller as shown in **Fig. 6**. **Figure 10** shows the reference trajectory used to control the manipulator and

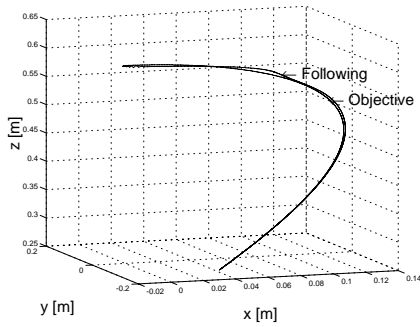


Figure 8: Control result for the trajectory-1.

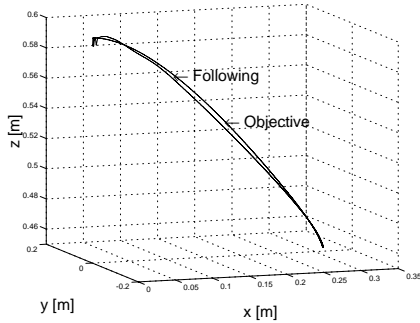


Figure 9: Control result for the trajectory-2.

the fuzzy membership based control result. It can be seen that the dynamics identified using only the reference input of joint angle, its velocity, and their output data, can be effectively applied to control a manipulator for a given desired trajectory, using the proposed method. This demonstrates the effectiveness of the combination of the RKGNNs, the architecture of the function NN composed of RBFs, and the evolutionary algorithm used to identify the dynamics of a multi-link manipulator.

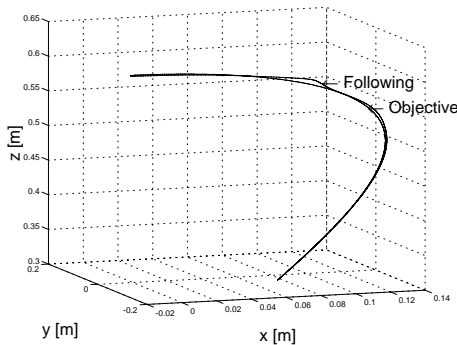


Figure 10: Control result for the new reference trajectory.

## 7 Conclusions

The Runge–Kutta–Gill neural network (RKGNN) has been developed and it was applied for identifying the complex dynamics of manipulators with higher degrees-of-freedom. 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.

Simulation studies were carried out to identify the dynamics of a three-link manipulator subject to dynamic constraints using only reference-input angle and angular velocity and their output data. The identified dynamics for two reference trajectories were used to control the manipulator for a new reference trajectory using maximum fuzzy membership based model selection method. Promising results have been obtained to prove the effectiveness of the proposed method to identify the highly non-linear dynamics of a multi-link manipulator.

## References

- [1] 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.
- [2] C.-J. Wu and C.-H. Huang, “Back-Propagation Neural Networks for Identification and Control of a Direct Drive Robot,” *J. of Intelligent and Robotic Systems*, vol. 16, no. 1, 1996, pp. 45–64.
- [3] J. J. Craig, *Introduction to Robotics: Mechanics and control*, 2nd edn, Reading, MA: Addison-Wesley, 1989.
- [4] 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, 1998, pp. 294-307.
- [5] T. Nanayakkara, K. Watanabe and K. Izumi, “Evolving in Dynamic Environments Through Adaptive Chaotic Mutation,” in *Procs. of Fourth International Symposium on Artificial Life and Robotics*, Oita, Japan, Jan 19–22, 1999, vol. 2, pp. 520–523.
- [6] J.-H. Kim, and Hyun Myung “Evolutionary Programming Techniques for Constrained Optimization Problems,” *IEEE Trans. on Evolutionary Computation*, vol. 1, no. 2, 1997, pp. 129–140.