Fuzzy improved adaptive neuro-NMPC for on-line path tracking and obstacle avoidance of redundant robotic manipulators

Ashkan M. Z. Jasour Mohammad Farrokhi

Department of Electrical Engineering,
Center of Excellence for Power System Automation and Operation
IRAN University of Science and Technology,
TEHRAN, IRAN
amjasour@ee.iust.ac.ir farrokhi@iust.ac.ir

Abstract: This paper presents a Nonlinear Model Predictive Control (NMPC) for redundant robotic manipulators. Using NMPC, the end-effector of the robotic manipulator tracks a predefined geometry path in Cartesian space in such a way that no collision with obstacles in the workspace and no singular configurations for the robot occurs. Nonlinear dynamic of the robot, including actuators dynamic, is also considered. Moreover, the on-line tuning of the weights in NMPC is performed using the fuzzy logic. The proposed method automatically adjusts the weights in the cost function in order to obtain good performance. Furthermore, using neural networks for model prediction, no prior knowledge about system parameters is necessary and system robustness against changes in its parameters is achieved. Numerical simulations of a 4DOF redundant spatial manipulator actuated by DC servomotors shows effectiveness of the proposed method.

Keywords: robotic manipulator; path tracking, obstacle avoidance; model predictive control; fuzzy logic; adaptive.

1 Introduction

Today, robotic manipulators are increasingly used in many tasks such as industry, medicine and space. One of the main reasons for the development of manipulators is to replace human in doing long and repetitive operations and unhealthy tasks. In particular, these robots are needed to track a predefined path in such a way that no collision with obstacles in the environment occurs. High degrees of freedom for redundant manipulators lead to an infinite number of possible joint positions for the same pose of the end-effector. Hence, for a given end-effector path in Cartesian space, the robot can track it in many different configurations, among which the collision free and singular free tracking must be selected. Finding feasible path for joints of redundant manipulators for a given end-effector path is called the redundancy resolution (Conkur, 2005). Redundancy resolution and obstacle avoidance are already considered in papers. With gradient projection technique, redundancy can be solved considering obstacle avoidance (Chen et al., 2002). In task-priority redundancy resolution technique, tasks are performed with the order of priority. Path tracking is given the first priority and obstacle avoidance or singularity avoidance is given the second priority (Chiacchio et. al, 1991; Nakamuro, 1991). This technique provides local optimal solution that is suitable for real-time redundancy control but not for large number of tasks. The generalized inverse Jacobin technique and extended Jacobin technique, which are used for redundancy solution, are time consuming (Boddy and Taylor, 1993; Chevallereau and Khalil, 1988; Yoshikawa, 1993). Optimization techniques, which minimize a cost function subject to constraints, like the end-effector path tracking and the obstacle avoidance, are not suitable for on-line applications (Nakamuro, 1991).

In this paper, Nonlinear Model Predictive Control (NMPC) method is presented for redundancy resolution considering obstacles and singularity avoidance. Although Model Predictive Control (MPC) is not a new control method, works related to robot manipulators using MPC are limited. The related works are about joint space control and end-effector coordinating. The linear MPC is used by Kim and Han (2004), Valle, Tadeo, and Alvarez (2002) and Vivas and Mosquera (2005). On the other hand, NMPC is used by Hedjar et al. (2002; 2005), Poignet and Gautier (2000) and Wroblewski (2004) for joint space control of manipulators. In all these MPC methods, manipulator joints follow trajectories defined for each joint. Therefore, redundancy resolution as well as obstacle and singularity avoidance were not considered before.

In this paper, redundancy resolution, obstacle and singularity avoidance as well as constraints of the robot are considered simultaneously. The proposed method can handle large number of tasks simultaneously despite the task-priority redundancy resolution technique. The generalized inverse Jacobin technique, the extended Jacobin method and optimization techniques can be implemented in real-time. Moreover, changes in the workspace of the robot can be taken into account. In addition, NMPC eliminates the need for path planning and obtains directly the control law to navigate the robot from one pint to another or to track a predefined path in the workspace. In other words, using NMPC, the input voltages of DC servomotors of joints, considering the input limitations of motors, are obtained in such a way that the end-effector of a redundant manipulator tracks a given path in Cartesian space considering obstacles and singularity avoidance. Moreover, fuzzy logic is implemented in order to improve performance of NMPC. That is, using a fuzzy system, an automatic mechanism for the on-line tuning of the weights for the path tracking and obstacle avoidance terms in the cost function is proposed. In other words, a fuzzy system is used to calculate optimal values for the weights in the cost function that is a challenging issue in MPC. On the other hand, in previous works related to fuzzy MPC, fuzzy systems were employed for system identification (Ibarrola, Pinzolas, and Cano, 2005; Karer et al., 2007; Mendonca, Sousa and Costa, 2007; Yeh et al., 2006).

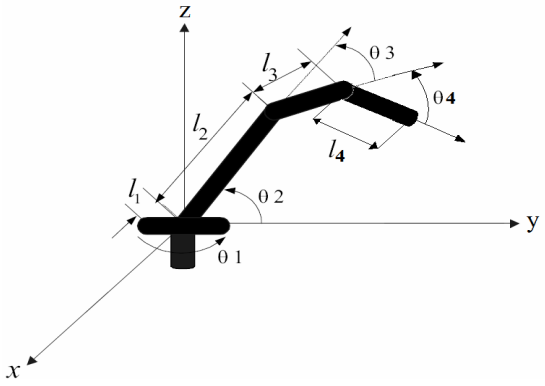

Figure 1 Schematic of a 4DOF spatial manipulator

Table 1 Denavit-Hartenberg parameters of robot Figure1

Link	α^o	a	d	θ^o
1	90	0	0	θ_1
2	0	l_2	0	θ_2
3	0	l_3	0	θ_3
4	0	l_4	0	θ_4

Some researchers employ adaptive control methods when the robot dynamic is uncertain. Adaptive control methods using Neural Networks (NNs) have been employed before to control highly uncertain and nonlinear systems (Huang et al., 2007; Hayakawa, Haddad, and Hovakimyan, 2008; Mnif, Gastli, and Jallouli, 2007). In this paper, NNs are used as the model predictor in NMPC. An important advantage of NNs is that no prior knowledge about system parameters is required. Moreover, due to adaptability of NNs, system robustness against changes in its parameters is obtained.

This paper is organized as follows: Section 2 presents nonlinear dynamic of a 4DOF spatial redundant manipulator including the actuators dynamic. Section 3 describes the nonlinear predictive control. Section 4 shows how to implement NMPC for path tracking and obstacle avoidance of a 4DOF manipulator. Section 5 presents the proposed modified NMPC using fuzzy logic. In Section 6, neural networks are used to model the system. Conclusions are drawn in Section 7.

2 Robot Manipulator Dynamic

Schematic diagram of a 4DOF spatial redundant robot manipulator is shown in Figure 1. According to Denavit-Hartenberg parameters (Lewis, Abdallah, and Dawson, 2004) of this robot in Table 1, the position of the end-effector in Cartesian space can be calculated in terms of joint angles as

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} c_1(l_4 c_{234} + l_3 c_{23} + l_2 c_2) \\ s_1(l_4 c_{234} + l_3 c_{23} + l_2 c_2) \\ l_4 s_{234} + l_3 s_{23} + l_2 s_2 \end{bmatrix}. \tag{1}$$

The dynamic model of robot manipulators can be obtained using the Lagrangian method as follows (Lewis, Abdallah, and Dawson, 2004; Yoshikawa, 1990):

$$\mathbf{M}(\boldsymbol{\theta})\ddot{\boldsymbol{\theta}} + \mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) + \mathbf{D}(\dot{\boldsymbol{\theta}}) + \mathbf{G}(\boldsymbol{\theta}) = \boldsymbol{\tau} \tag{2}$$

where $\boldsymbol{\theta} \in R^n$ is the vector containing the joint angles, $\mathbf{M}(\boldsymbol{\theta}) \in R^{n \times n}$ is the symmetric and positive definite inertia matrix, $\mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \in R^n$ is the centrifugal and coriolis force vector, $\mathbf{D}(\dot{\boldsymbol{\theta}}) \in R^n$ is the vector representing friction of the joints, $\mathbf{G}(\boldsymbol{\theta}) \in R^n$ is the gravity vector, $\boldsymbol{\tau} \in R^n$ is the torque vector of joints, and n is the degree of freedom, which is equivalent to four for the robot considered in this paper. These vectors and matrix are given in Appendix. Friction for the i^{th} joint can be written as (Lewis, Abdallah, and Dawson, 2004)

$$D(i) = D_V \dot{\theta}_i + D_d \operatorname{sgn}(\dot{\theta}_i) \tag{3}$$

where D_V and D_d are the coefficients for the viscous friction and the dynamic friction, respectively.

The dynamics of the armature-controlled DC servomotors that drive the links can be expressed in the following form (Lewis, Abdallah, and Dawson, 2004):

$$\begin{aligned} \boldsymbol{\tau}_e &= \mathbf{K}_T \mathbf{i}_a \\ \boldsymbol{\tau}_e &= \mathbf{J}_m \ddot{\boldsymbol{\theta}}_m + \mathbf{B}_m \dot{\boldsymbol{\theta}}_m + \boldsymbol{\tau}_m \\ \mathbf{v}_t &= \mathbf{R}_a \mathbf{i}_a + \mathbf{L}_a \dot{\mathbf{i}}_a + \mathbf{K}_E \dot{\boldsymbol{\theta}}_m \end{aligned} \tag{4}$$

where $\boldsymbol{\tau}_e \in R^n$ is the vector of electromagnetic torques, $\mathbf{K}_T \in R^{n\times n}$ is the diagonal matrix of the motor torque constant, $\mathbf{i}_a \in R^n$ is the vector of armature currents, $\mathbf{J}_m \in R^{n\times n}$ is the diagonal matrix of the moment inertia, $\mathbf{B}_m \in R^{n\times n}$ is the diagonal matrix of torsional damping coefficients, $\boldsymbol{\theta}_m, \dot{\boldsymbol{\theta}}_m, \ddot{\boldsymbol{\theta}}_m \in R^n$ denote the vectors of motor shaft positions, velocities and accelerations, respectively, $\boldsymbol{\tau}_m \in R^n$ is the vector of load torques, $\mathbf{v}_t \in R^n$ is the vector of armature input voltages, $\mathbf{R}_a \in R^{n\times n}$ is the diagonal matrix of armature resistances, $\mathbf{L}_a \in R^{n\times n}$ is the diagonal matrix of armature inductances, and $\mathbf{K}_E \in R^{n\times n}$ is the diagonal matrix of back electromotive forces (EMF) coefficients.

In order to use the DC servomotors for moving the n^{th} link of the robot manipulator, the relationship between the robot joint and the motor-shaft can be represented as

$$\boldsymbol{\Gamma} = diag\left[\frac{\boldsymbol{\theta}_i}{\boldsymbol{\theta}_{m_i}}\right] = diag\left[\frac{\boldsymbol{\tau}_{m_i}}{\boldsymbol{\tau}_i}\right], \tag{5}$$

where $\boldsymbol{\Gamma} \in R^{n\times n}$ is a diagonal positive definite matrix of the gear ratios for the n^{th} joint. Since the armature inductance is very small and negligible, Eq. (4) can be expressed as (Lewis, Abdallah, and Dawson, 2004)

$$\mathbf{J}_m\ddot{\boldsymbol{\theta}}_m + \left(\mathbf{B}_m + \mathbf{K}_E\mathbf{K}_T\mathbf{R}_a^{-1}\right)\dot{\boldsymbol{\theta}}_m + \boldsymbol{\tau}_m = \mathbf{K}_T\mathbf{R}_a^{-1}\mathbf{v}_t. \tag{6}$$

Using Eq. (5) to eliminate $\boldsymbol{\theta}_m$ and $\boldsymbol{\tau}_m$ in Eq. (6) and then substituting for $\boldsymbol{\tau}$ from Eq. (2), the governed equation of *n*-link robot manipulator including actuator dynamics can be obtained as

$$(\mathbf{J}_m + \boldsymbol{\Gamma}^2\mathbf{M})\ddot{\boldsymbol{\theta}} + \left(\mathbf{B}_m + \mathbf{K}_E\mathbf{K}_T\mathbf{R}_a^{-1}\right)\dot{\boldsymbol{\theta}} + \boldsymbol{\Gamma}^2(\mathbf{C}+\mathbf{G}+\mathbf{D}) = \boldsymbol{\Gamma}\mathbf{K}_T\mathbf{R}_a^{-1}\mathbf{v}_t. \tag{7}$$

According to Eq. (7), armature input voltages are considering as the control effort. The detailed parameters of the robot manipulator and DC servomotors are given in Tables 2 and 3, respectively.

Table 2 Manipulator parameters

Link	**1**	**2**	**3**	**4**
***L* (m)**	1	0.5	0.4	0.3
***M* (kg)**	1	0.5	0.4	0.3

Table 3 Parameters of DC servomotors

Motor	**1**	**2**	**3**	**4**
R_a	6.51	6.51	6.51	6.51
K_E	0.7	0.7	0.7	0.7
K_T	0.5	0.5	0.5	0.5
B_m	64×10^{-4}	64×10^{-4}	64×10^{-4}	64×10^{-4}
J_m	0.2	0.2	0.2	0.2
γ	1:100	1:100	1:10	1:10
v_t	24	24	24	24

3 Model Predictive Control

Unlike classical control schemes where the control actions are taken based on the past output of the system, the MPC is a model-based optimal controller, which uses predictions of the system output to calculate the control law (Allgower and Findeisen, 2004; Findeisen, 2003).

At every sampling time k, based on measurements obtained at time k, the controller predicts the output of the system over prediction horizon N_P in future using model of the system and determines the input over the control horizon $N_C \leq N_P$ such that a predefined cost function is minimized.

To incorporate feedback, only the first member of the obtained input is applied to the system until the next sampling time (Allgower and Findeisen, 2004).
Using the new measurement at the next sampling time, the whole procedure of *prediction and optimization* is repeated.
From the theoretical point of view, the MPC algorithm can be expressed as

$$u = \arg_{u} \min(J(k)) \tag{8}$$

subject to

$$\begin{aligned}
&\mathbf{x}(k \mid k) = \mathbf{x}_0 \\
&u(k+j \mid k)) = u(k+N_C \mid k)) \ , \ \ j \geq N_C \\
&\mathbf{x}(k+j+1 \mid k) = \ f_d(\mathbf{x}(k+j \mid k), u(k+j \mid k)) \\
&y(k+j+1 \mid k) = h_d(\mathbf{x}(k+j+1 \mid k)) \\
&\mathbf{x}_{\min} \leq \mathbf{x}(k+j+1 \mid k) \leq \mathbf{x}_{\max} \\
&u_{\min} \leq u(k+j \mid k) \leq u_{\max}
\end{aligned} \tag{9}$$

where $j \in [0, N_P - 1]$, $\mathbf{x}$ and u are states and input of the system, $\mathbf{x}_0$ is the vector of initial conditions, f_d and h_d are the model of the system used for prediction, and N_P and N_C are the prediction horizon over the output of the system and the control law, respectively. The notation $a(m|n)$ indicates the value of a at instant m predicted at instant n. Moreover, $[\mathbf{x}_{\min}, \mathbf{x}_{\max}]$ and $[u_{\min}, u_{\max}]$ stand for the lower and the upper bound of states and input, respectively. The cost function J is defined in terms of the predicted and the desired output of the system over the prediction horizon.
The MPC schemes that are based on the nonlinear model of the system or consider non-quadratic cost function and nonlinear constrains on inputs and states are called Nonlinear MPC (NMPC) (Allgower and Findeisen, 2004).
The optimization problem in (8) and (9) must be solved at every sampling time k, yielding a sequence of optimal control law as $\{u^*(k \mid K), \ldots, u^*(k+N_C-1)\}$.

For the optimization problem, the Sequential Quadratic Programming (SQP) method has been employed. This method closely mimics the Newton's method for the constrained optimization just as is performed for the unconstrained optimization. In this method, an approximation is made recursively of the Hessian of the Lagrangian function using a quasi-Newton updating method. This is then used to generate a QP sub-problem whose solution is used to form a search direction for a line search procedure. The SQP method is one of the fastest and most efficient methods to solve the constraint nonlinear optimization problem. For more details, the reader may refer to Fletcher (1987).

4 Path Tracking and Obstacle Avoidance Using NMPC

The purpose of the path tracking and the obstacle avoidance for robot manipulators is to obtain a control law such that the end-effector tracks a given geometry path in Cartesian space; and at the same time to avoid collisions between the end-effector, links, and obstacles on the workspace. To achieve this purpose, the NMPC is implemented in this section. Block diagram of NMPC is shown in Figure 2.

According to the NMPC algorithm, an appropriate cost function must be determined in order to obtain the control law.

For path tracking, the cost function must have direct relationship with the tracking error (i.e. the error between the end-effector coordination and the given path in Cartesian space). On the other hand, for obstacles avoidance, the cost function must have inverse relationship with the distance between the obstacle and the manipulator. Therefore, a proper cost function is introduced as

$$J = \sum_{j=1}^{N_P} D_P(k+j \mid k)\, Q\, D_P(k+j \mid k) + \frac{R}{D_O(k+j \mid k) D_O(k+j \mid k)} \tag{10}$$

where D_P is the Euclidean distance between the end-effector and the geometry path in Cartesian space, D_O is the minimum Euclidean distance between the manipulator and obstacles, and $Q \geq 0$ and $R \geq 0$ are the weighting parameters. Notation $a(m|n)$ indicates the value of a at the instant m predicted at instant n.

According to Eq. (10), the path tracking term of the cost function is described as the distance between the end-effector and the desired path. However, the obstacle avoidance term is described as the inverse of the distance between the manipulator and obstacles. Hence, it is important to notice that the distance is bounded in the workspace, but the inverse of the distance is not bounded. Therefore, the combination of these two inconsistent terms as a cost function is not appropriate for an optimization problem. To handle this problem properly, these two terms are normalized to [0, 1] using a nonlinear map. Based on these, the modified cost function is proposed as

$$J = \sum_{j=1}^{N_P} Q\left(\frac{e^{D_P(k+j|k)} - e^{D_{P\min}}}{e^{D_{P\max}} - e^{D_{P\min}}}\right) + R\left(\frac{e^{-D_O(k+j|k)} - e^{-D_{O\max}}}{e^{-D_{O\min}} - e^{-D_{O\max}}}\right) \tag{11}$$

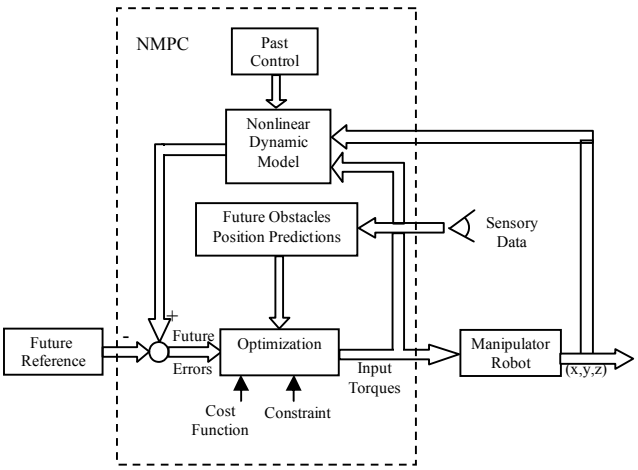

Figure 2 Block diagram of NMPC

where $[D_{P\min}, D_{P\max}]$ and $[D_{O\min}, D_{O\max}]$ represent the range of variations for D_P and D_O, respectively. It should be noted that D_P is the Euclidean distance between the end-effector and the geometry path in Cartesian space. Therefore, for the worst case (i.e. $D_{P\max}$) the maximum distance between the end-effector and the desired path is equal to the diameter of the work space, which is equal to 2.4 m for the manipulator in Table 2. On the other hand, D_O is the minimum Euclidean distance between the manipulator and obstacles. Hence, $D_{O\min}$ is equal to the radius of the work space, which is equal to 1.2 m for the manipulator in Table 2.

For NMPC, a nonlinear dynamic model of the manipulator is used for the optimization of the cost function in this paper.

Substituting $\left(\theta_i(k+1)-\theta_i(k)\right)/T$ for $\dot{\theta}_i$ $(i=1,\ldots,n)$ in the dynamic Eq. (7), a one-step-ahead prediction for joints angles can be expressed as

$$\boldsymbol{\theta}(k+1)=\mathbf{f}_d\left(\boldsymbol{\theta}(k),\mathbf{v}_t(k)\right) \tag{12}$$

where k is the discrete sampling time and T is the sampling rate, which is equal to 0.5 sec. in simulations. Using forward kinematics as in Eq. (1), a one-step-ahead prediction of the end-effector position can be obtained. However, in the predictive control, it is used for multi-step predictions over the prediction horizon by applying it recursively.

Next, constrains in the optimization problem is considered. Considering the fact that the amplitude of input voltages is limited, one of the constrains is

$$\mathbf{v}_{t\min} \le \mathbf{v}_t \le \mathbf{v}_{t\max} \tag{13}$$

where $\mathbf{v}_{t\min}$ and $\mathbf{v}_{t\max}$ stand for the lower and the upper bound of input voltages applied to servo DC motors, respectively (e.g. –24 V and 24 V, respectively as Table 3 shows).

Next, considering singular configurations, the joint velocities theoretically become infinite. Therefore, the following constrain must be taken into account to avoid singularities:

$$\dot{\boldsymbol{\theta}}_{\min} \le \dot{\boldsymbol{\theta}} \le \dot{\boldsymbol{\theta}}_{\max} \tag{14}$$

where $\dot{\boldsymbol{\theta}}_{\min}$ and $\dot{\boldsymbol{\theta}}_{\max}$ are the lower and the upper bound of the joints velocity, respectively (e.g. –400 and 400 deg/s, respectively, considering the manipulator and motors parameters given in Tables 2 and 3).

By incorporating constrains (13) and (14) into the cost function (11), the optimization problem can be solved. Simulation results for a rectangular path in Cartesian space with obstacles inside the workspace are shown in Figures 3 to 7. In this case, $N_P = 5$, $N_C = 1$, $Q = 10$, and R = 0.8. As Figure 3 shows, the end-effector follows the desired path very closely with negligible tracking error. Figure 4 indicates that the proposed controller produces relatively smooth trajectory for joints positions. Figure 6 shows that the input voltages applied to servomotors go to saturation for a short when the links start to move from the down-ward position to the un-ward position. According to the 3D view in Figure 7, when the manipulator approaches the obstacle, the NMPC changes the joints positions in such a way that no collision occurs while tracking of the desired path of the end effector takes place.

Figures 8 to 10 show the case where the obstacle is located on the path. In this case, $N_P = 5$ and $N_C = 1$. However, the best results are obtained for $Q = 10$ and $R = 1.3$. That is, when the coordinates of obstacles are changed, the weights in the cost function must be customized accordingly. As Figure 8 shows, in this case, the manipulator avoids the obstacle but follows the desired path with more tracking error. This is mainly due to the fact that weighting factors Q and R need to be changed

adaptively while the end-effector approaches the obstacle or moves away from it. In other words, when the end-effector is far away form the obstacle, Q must be increased and R must be decreased; and vise versa.

In the next section, a fuzzy method is proposed to adaptively tune Q and R as the robot follows the desired path and tries to avoid obstacles.

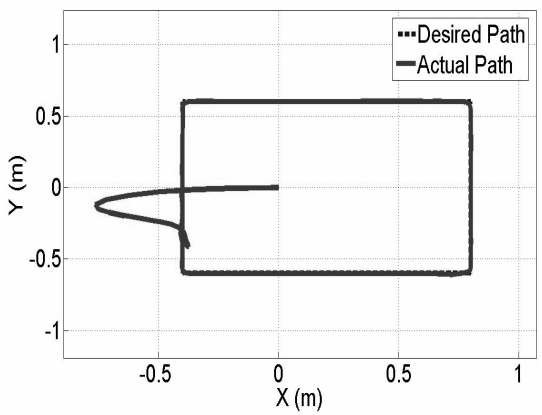

Figure 3 Desired and actual end-effector path

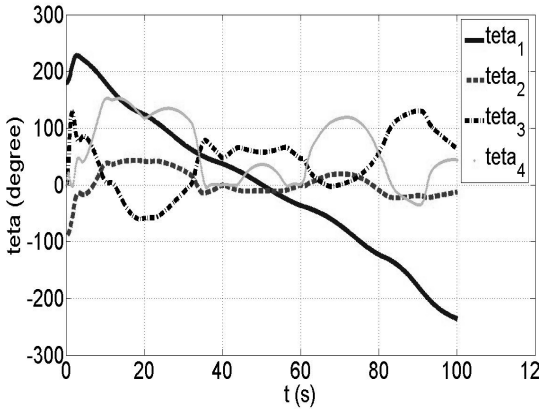

Figure 4 Positions of manipulator joints

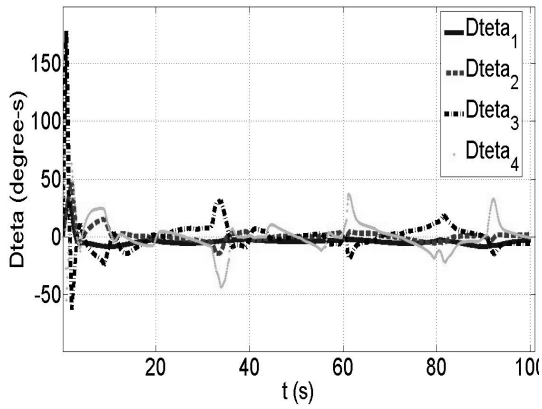

Figure 5 Joints velocities

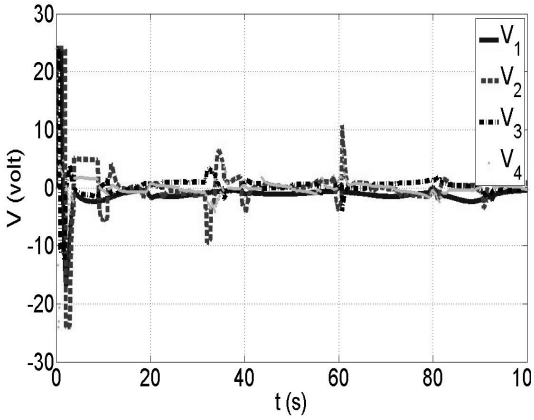

Figure 6 Input voltages of servo DC motors

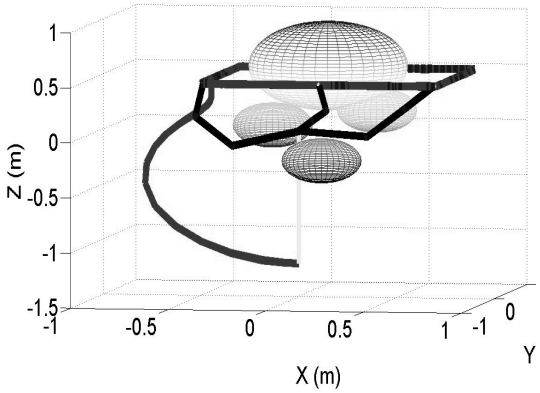

Figure 7 Path following of a 4DOF manipulator with obstacles on the workspace

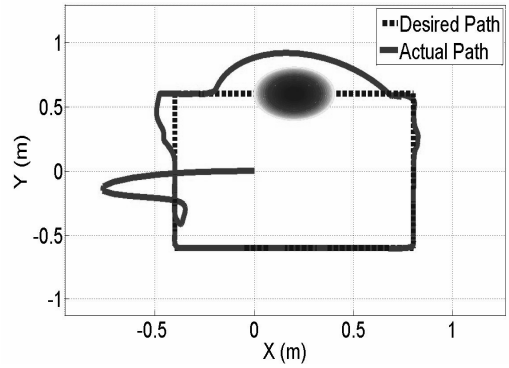

Figure 8 Desired and actual end-effector path

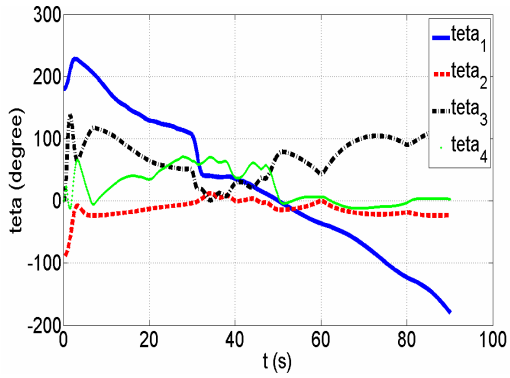

Figure 9 Positions of manipulator joints

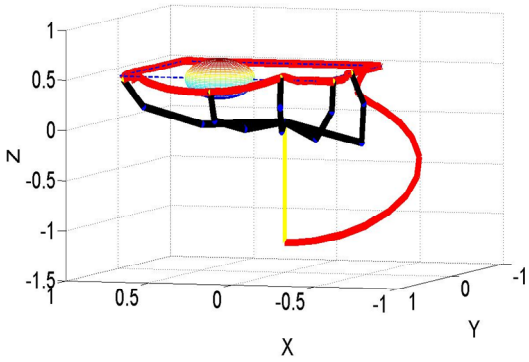

Figure 10 Path following with obstacle on the desired path

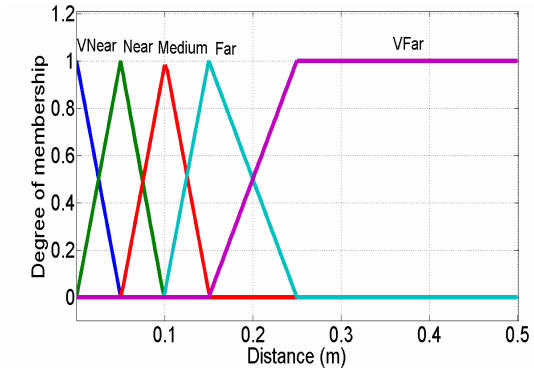

Figure 11 Membership functions for distance

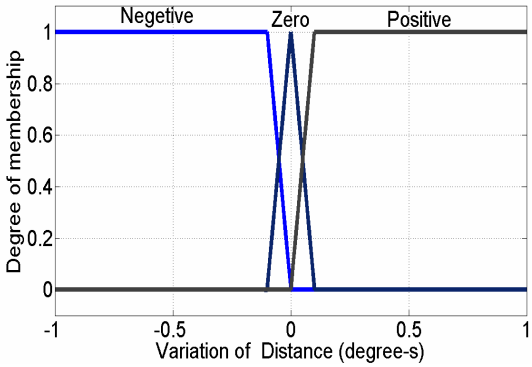

Figure 12 Membership functions for distance variations

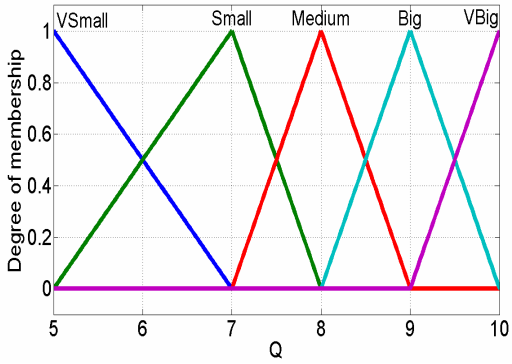

Figure 13 Membership functions for weight *Q*

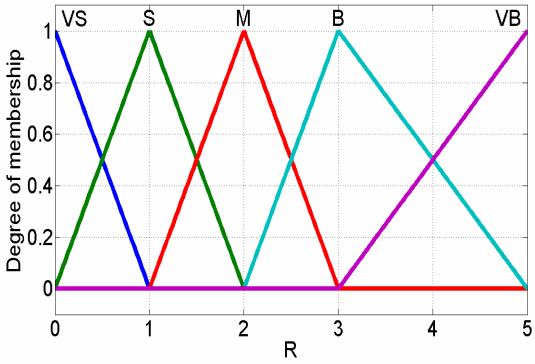

Figure 14 Membership functions for weight *R*

Table 4 Fuzzy operations

And	Implication	Aggregation	Defuzzification
min	Prod	max	lom†

† largest (absolute value) of the maximum

Table 5 Fuzzy rules

If D_O is Very Far & $\dot{D}_O$ is Positive Then Q is Very Big & R is Very Small
If D_O is Very Far & $\dot{D}_O$ is Zero Then Q is Very Big & R is Very Small
If D_O is Very Far & $\dot{D}_O$ is Negative Then Q is Very Big & R is Very Small
If D_O is Far & $\dot{D}_O$ is Positive Then Q is Very Big & R is Very Small
If D_O is Far & $\dot{D}_O$ is Zero Then Q is Very Big & R is Very Small
If D_O is Far & $\dot{D}_O$ is Negative Then Q is Big & R is Small
If D_O is Medium & $\dot{D}_O$ is Positive Then Q is Big & R is Very Small
If D_O is Medium & $\dot{D}_O$ is Zero Then Q is Big & R is Small
If D_O is Medium & $\dot{D}_O$ is Negative Then Q is Medium & R is Medium
If D_O is Near & $\dot{D}_O$ is Positive Then Q is Big & R is Small
If D_O is Near & $\dot{D}_O$ is Zero Then Q is Medium & R is Big
If D_O is Near & $\dot{D}_O$ is Negative Then Q is Small & R is Big
If D_O is Very Near & $\dot{D}_O$ is Positive Then Q is Medium & R is Medium
If D_O is Very Near & $\dot{D}_O$ is Zero Then Q is Very small & R is Very Big
If D_O is Very Near & $\dot{D}_O$ is Negative Then Q is Very Small & R is Very Big

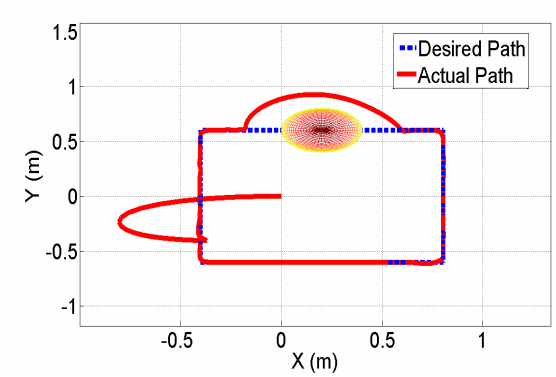

Figure 15 Desired and actual end-effector path

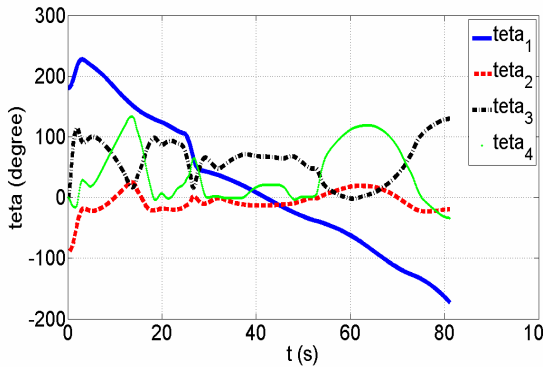

Figure 16 Positions of manipulator joints

5 Path Tracking and Obstacle Avoidance Using Fuzzy NMPC

In the previous section, it was observed that for different paths and different positions of obstacles, the weights Q and R must be changed and fine-tuned in order to produce satisfactory results. To provide a proper solution to this problem, fuzzy logic is employed in this paper for the on-line tuning of these weights. The proposed fuzzy systems use the minimum distance between the manipulator and the obstacle and the rate of change of this distance as their inputs. The outputs of fuzzy systems are the weighting factors Q and R. Notice that a fuzzy system has only one output; hence, two fuzzy systems are needed to estimate these weights.

To design these fuzzy systems, a boundary around each obstacle is considered in such a way that the control algorithm does not care about obstacles unless the end-effector or any links of the manipulator enters this boundary region.

Parameters of fuzzy systems (the membership functions and fuzzy IF-THEN rules) are defined in such a way that when the manipulator is outside the obstacle regions, R is equal to zero and when the manipulator is inside this region, R is increased and Q is decreased accordingly. Fuzzy rules, membership functions, and fuzzy operations are shown in Figures 11 to 14 and in Tables 4 and 5.

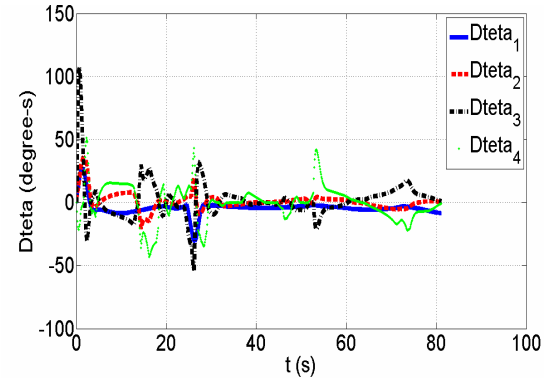

Figure 17 Velocities of manipulator joints

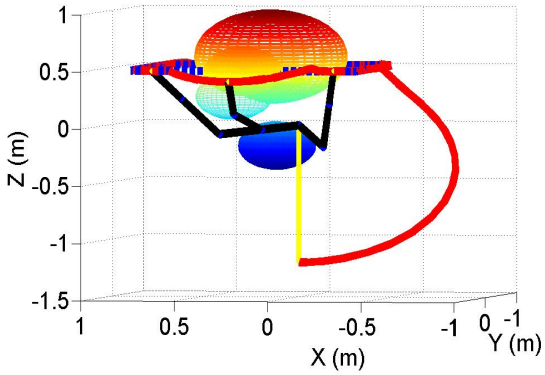

Figure 18 Path following with obstacles in workspace

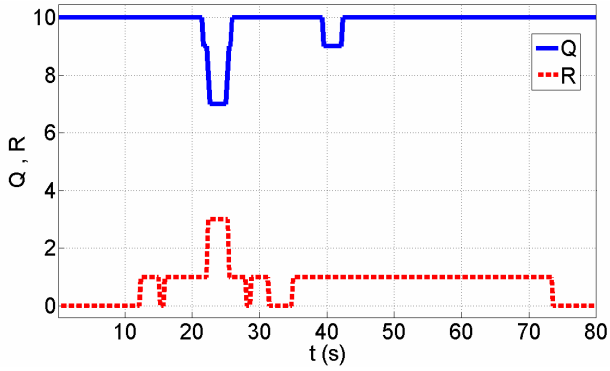

Fig. 19 Tuning of optimization weights Q and R in cost function

Using the proposed fuzzy system, when the distance between the manipulator and obstacles is more than 0.2 m, $R = 0$ and $Q = 10$. For distances less than 0.2 m, $5 \leq Q < 10$ and $0 < R \leq 5$.

Simulation results of the proposed fuzzy NMPC are shown in Figures 15 to 19. As these figures show, the manipulator can follow the desired path with better accuracy as compared with the previous section. Moreover, Figure 19 shows that the fuzzy system effectively changes the weighting parameters Q and R in the optimization process for better path following and obstacle avoidance.

6 Model Prediction Using Artificial Neural Networks

As it was mentioned before, a model of the system is needed for predictions in MPC. In previous sections, the nonlinear dynamic equations of the manipulator were employed for model prediction. In this section, a Multilayer Perceptron (MLP) is used for modeling of the nonlinear dynamic equations of the manipulator in MPC. One of the advantages of using neural networks (NNs) for the model prediction is that no prior knowledge about system parameters is needed. The prediction is based on the input voltages, angular positions, and angular velocities of joints at the current sampling time. The output of the predictor is the angular position of the corresponding joint at the next sampling time. Using this structure, a one-step-ahead prediction of the joints position is obtained. However, in the predictive control, multi-step prediction over the prediction horizon is needed. By recursive application of the one-step prediction, the multi-step prediction can be achieved. In this case, the outputs of the NN are considered as inputs for the next step. For higher accuracy, one NN is employed for every link of the manipulator to predict the position of the corresponding joint angle instead of using one neural network for the whole robot. The employed structure of the MLP is shows in Figure 20. The NN consists of three

layers: 9 neurons in the input layer, 10 neurons in the hidden layer, and 1 neuron in the output layer. The activation functions for the hidden and the output layers are of tangent hyperbolic and linear types, respectively. The inputs to the NN for the i^{th} link are angular positions and angular velocities of joints and input voltages for the i^{th} joint at the current sampling time. Therefore, the input vector applied to the NNs is

$$[v_{ti}(k)\ \ \theta_1(k)\ \ \theta_2(k)\ \ \theta_3(k)\ \ \theta_4(k)\ \ \Delta\theta_1(k)\ \ \Delta\theta_2(k)\ \ \Delta\theta_3(k)\ \ \Delta\theta_4(k)]$$

The output is $\theta_i(k+1)$ which refers to the position of the i^{th} joint at the next sampling time. The training method is the error back-propagation algorithm. Trainings are performed in two steps: offline and online. The offline training is needed only for some familiarity of NNs with nonlinear properties of the robot manipulator. This is mainly because the weights of NNs are initialized randomly and if NNs with random weights are used for online training, early collision with obstacles might occur. Next, using the online training for the model prediction, system robustness against changes in system parameters (e.g. masses and frictions) will be obtained. This is mainly due to the adaptability property of NNs to cope with changes in the system parameters.

For simulations, the same rectangular path in Cartesian space with obstacles inside the workspace is considered. Simulation results for the case of $T = 0.5$ sec., $N_p = 5$ and $N_C = 1$ are shown in Figures 21 to 26.

Next, in order to show the robustness of the proposed method against changes in system parameters, the mass of link 4 is increased by 20 Kg at $t = 20$ sec. Simulation results for this case are shown in Figures 26 and 27. As Figure 26 shows, the controller can adapt itself very quickly to the changes in the system parameters in a relatively short time without any collision with obstacles. Although 20 kg change in the mass of the 4^{th} link might be unrealistic, nevertheless, the aim was to show the ability of the proposed method even against huge changes in system parameters.

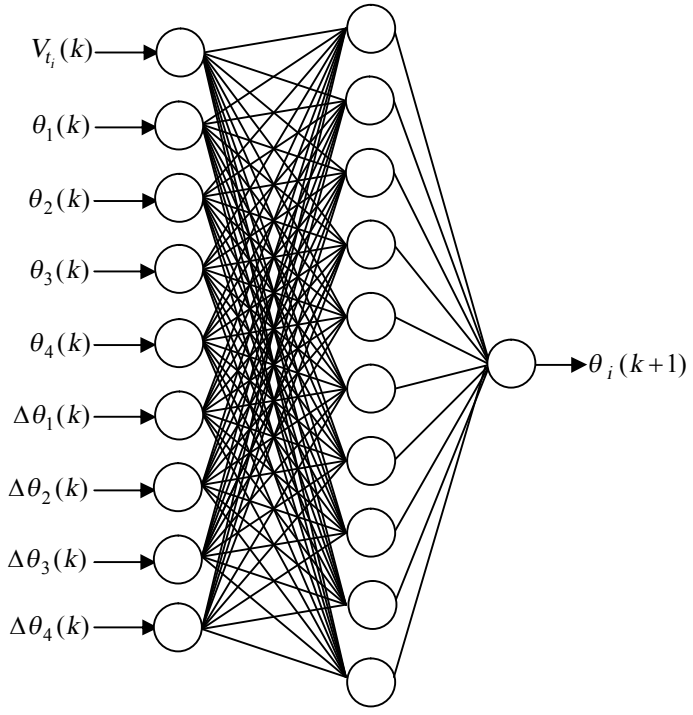

Figure 20 Neural network structure

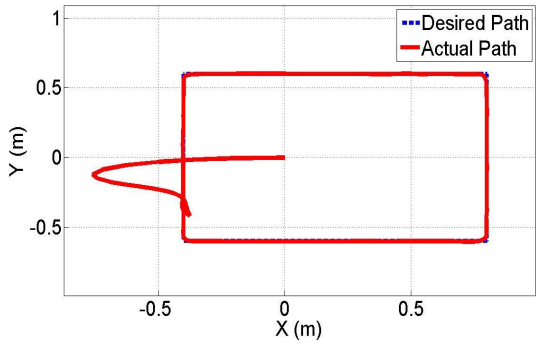

Figure 21 Desired and actual end-effector path

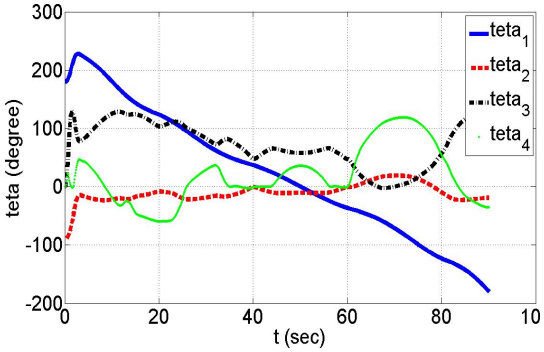

Figure 22 Positions of manipulator joints

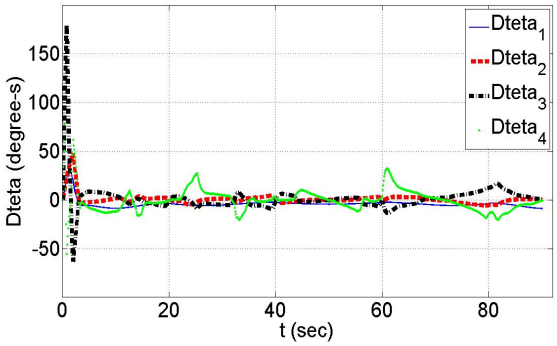

Figure 23 Velocities of manipulator joints

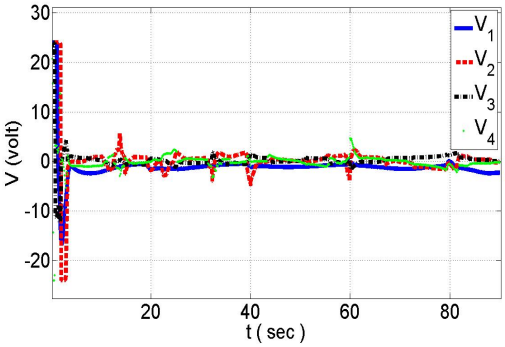

Figure 24 Input voltages of servo DC motors

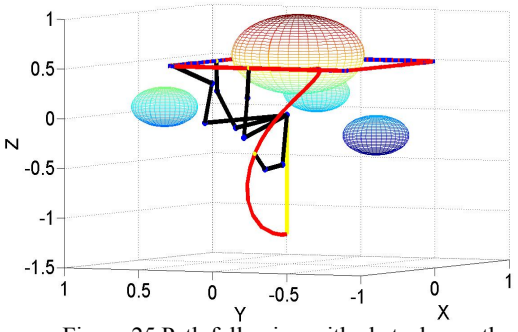

Figure 25 Path following with obstacles on the workspace

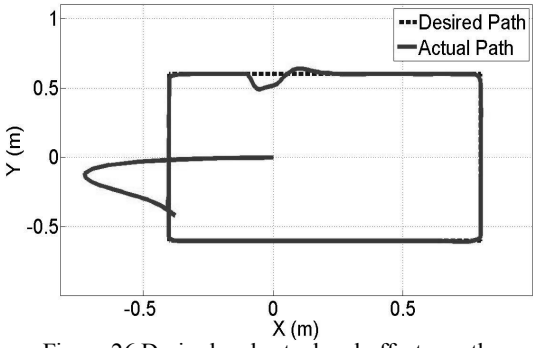

Figure 26 Desired and actual end-effector path

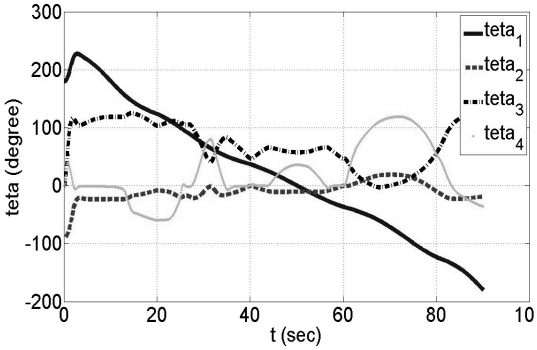

Figure 27 Positions of manipulator joints

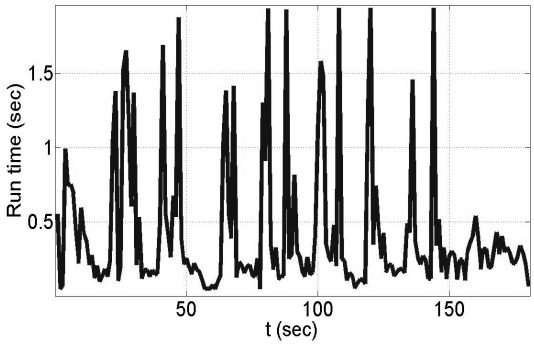

Figure 28 Simulation run time

Finally, possibility of real-time application of the proposed method is considered. Figure 28 shows the simulation runtime. As this figure shows, the runtime is mostly less than the sampling rate. However, it should be noted that: 1) simulations are performed in MATLAB software, which requires more time than a lower level computer program such as C++, 2) in simulations, solving the robot equations also incurs some computational time, which is not required in practice, 3) some aspects of computations, such training of NNs, can be processed in parallel. Hence, the simulation run time can be much less than that of shown in Figure 28. Therefore, in practice, the proposed method can be successfully applied to robot manipulators using conventional digital computers.

7 Conclusion

For the problem of path tracking and obstacle avoidance for robotic arms, the NMPC method was proposed in this paper. For this reason, two terms were introduced in the cost function, one for the tracking problem and the other for the obstacle avoidance. Moreover, by introducing constrains to the joints velocities, singularities were avoided. Furthermore, on-line tuning of the weighting factors in NMPC was achieved using fuzzy logic. The proposed fuzzy system automatically adjusts the path tracking and obstacle avoidance weights in the cost function for better performance. Using the tuning mechanism, obstacles do not affect performance of the manipulator unless they enter a predefined boundary region around obstacles. Moreover, NNs are employed for coping with uncertainties in system parameters. In this case, no prior knowledge about manipulator parameters is needed. Moreover, it was shown that the proposed method can be implemented in real-time applications.

References

Allgower, F., Findeisen, R., and Nagy, Z. (2004) 'Nonlinear model predictive control: from theory to application', *J. Chin. Inst. Chem. Engrs*, Vol. 35, No. 3, pp. 299–315.

Boddy, C. and Taylor, J. (1993) 'Whole arm reactive collision avoidance control of kinematically redundant manipulators', *IEEE Int. Conf. on Robotics and Automation*, Vol. 3, pp. 382–387, Atlanta, Georgia, USA.

Chevallereau, C. and Khalil, W. (1988) 'A new method for the solution of the inverse kinematics of redundant robots', *Proceedings of IEEE Int. Conf. on Robotics and Automation*, pp. 37–42, Washington, DC, USA.

Conkur, E. (2005) 'Path planning using potential fields for highly redundant manipulators', *Robotics and Autonomous Systems*, Vol. 52, pp. 209–228.

Chen, J., Liu, J., Lee, W., and Liang, T. (2002) 'On-line multi-criteria based collision-free posture generation of redundant manipulator in constrained workspace', *Robotica*, Vol. 20, pp. 625–636.

Chiacchio, P., Chiaverini, S., Sciavicco, L., and Siciliano, B. (1991) 'Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy', *International Journal of Robotics Research*, Vol. 10, pp. 410–426.

Findeisen, R., Imsland, L., Allgower, F., and Foss, B. (2003) 'State and output feedback nonlinear model predictive control: an overview', *European Journal of Control*, Vol. 9, pp. 190–206.

Fletcher, R. (1987) *Practical Methods of Optimization*, New York: John Wiley & Sons.

Hedjar, R., Toumi, R., Boucher, P., Dumur, D., and Tebbani, S. (2005) 'Finite horizon non linear predictive control with integral action of rigid link manipulators', *IEEE Conference on Control Applications, Int. J. Appl.* Math. Comput. Sci., Vol. 15, No. 4, pp. 101–113, Août, Canada.

Hedjar, R., Toumi, R., Boucher, P., and Dumur, D. (2002) 'Feedback nonlinear predictive control of rigid link robot manipulators', *American Control Conference*, Anchorage, Alaska.

Huang, S., KiongTan, K., Lee, T., and Putra, A. (2007) 'Adaptive control of mechanical systems using neural networks', *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, Vol 37, Issue 5, pp. 897–903.

Hayakawa, T. Haddad, W., and Hovakimyan, N. (2008) 'Neural network adaptive control for a class of nonlinear uncertain dynamical systems with asymptotic stability guarantees', *IEEE Transactions on Neural Networks*, Vol. 19, Issue 1, pp. 80–89.

Ibarrola, J., Pinzolas, M.and Cano, J., A (2005) 'Neurofuzzy scheme to on-line identification in an adaptive-predictive control', *Neural Computation and Application*, Vol.15, pp. 41–48.

Karer, G., Music, G., Skrianc, I.and Zupancic, B., (2007) 'Hybrid Fuzzy Modelling for Model Predictive Control', *Journal of Intelligent and Robotic Systems*, Vol. 50, Issue 3, pp. 297–319.

Kim, J. and Han, M. (2004) 'Adaptive robust optimal predictive control of robot manipulators', *30th Annual Conf. of the IEEE Industrial Electronics Society*, Busan, Korea.

Lewis, F., Abdallah, C., and Dawson, D. (2004) *Control of Robot Manipulators Theory and Practice*, New York: Marcel Dekker Inc.

Mnif, F., Gastli, A., and Jallouli, M. (2007) 'Adaptive ANN-based control for constrained robot manipulators', *Int. J. of Intelligent Systems Technologies and Applications* Vol. 2, No.1, pp 77-99.

Mendonca, L., Sousa, J. and SadaCosta, J., (2007) 'Fault tolerant control using fuzzy MPC', *6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Vol. 6, part 1, Tsinghua, P.R. China.

Nakamuro, Y. (1991) *Advanced Robotics Redundancy and Optimization*, New York: Addison-Wesley.

Poignet, Ph. and Gautier, M. (2000) 'Nonlinear model predictive control of a robot manipulator', *6th Int. Workshop on Advanced Motion Control*, pp. 401-406, Nagoya, Japan.

Valle, F., Tadeo, F., and Alvarez, T. (2002) 'Predictive control of robotic manipulators', *Proceedings of IEEE Int. Conf. Control Applications*, pp. 203-208, Glasgow, Scotland, UK.

Vivas, A., Mosquera, V. (2005) 'Predictive functional control of a PUMA robot', *ACSE Conf.*, CICC, Cairo, Egypt.

Wroblewski, W. (2004) 'Implementation of a model predictive control algorithm for a 6dof Manipulator-simulation results', *4th Int. Workshop on Robot Motion and Control*, Puszczykowo, Poland.

Yeh, S., Ji, D., Yoo, W., and Won, S., (2006) 'Efficient Fuzzy-MPC for Nonlinear systems: rule rejection', *International Joint Conference SICE-ICASE*, Busan, Oct, 2006.

Yoshikawa, T. (1993) 'Analysis and control of robot manipulators with redundancy', *First International Symposium in Robotic Research*, MIT Press, Cambridge, MA, pp. 439-446.

Yoshikawa, T. (1990) *Foundation of Robotics, Analysis and Control*, Boston: the MIT Press.

APPENDIX

$$
\begin{aligned}
M(1,1) &= \tfrac{2}{3} m_1 l_1^2 + \left(\tfrac{1}{3} m_2 l_2^2 + m_3 l_2^2 + m_4 l_2^2\right) c_2^2 + \left(\tfrac{1}{3} m_3 l_3^2 + m_4 l_3^2\right) c_{23}^2 + \tfrac{1}{3} m_4 l_4^2 c_{234}^2 \\
&\quad + \left(m_3 l_3 l_2 + 2 m_4 l_3 l_2\right) c_{23} c_2 + m_4 l_4 l_3 c_{234} c_{23} + m_4 l_4 l_2 c_{234} c_2 \\
M(2,2) &= \tfrac{1}{3} m_2 l_2^2 + m_3 l_2^2 + \tfrac{1}{3} m_3 l_3^2 + m_4 l_3^2 + m_4 l_2^2 + \tfrac{1}{3} m_4 l_4^2 + m_4 l_4 l_2 \; + \; m_4 l_3 l_4 c_4 + \left(m_3 l_3 l_2 + \; 2 m_4 l_2 l_3\right) c_3 \\
M(3,3) &= \tfrac{1}{3} m_3 l_3^2 + m_4 l_3^2 + \tfrac{1}{3} m_4 l_4^2 + \; m_4 l_3 l_4 c_4 \\
M(4,4) &= \tfrac{1}{3} m_4 l_4^2 \\
M(2,3) &= M(3,2) = \tfrac{1}{3} m_3 l_3^2 + m_4 l_3^2 + \tfrac{1}{2} m_4 l_4 l_2 + \tfrac{1}{3} m_4 l_4^2 + \left(\tfrac{1}{2} m_3 l_3 l_2 + m_4 l_2 l_3\right) c_3 + m_4 l_4 l_3 \; c_4 \\
M(2,4) &= M(4,2) = \tfrac{1}{2} m_4 l_4 l_2 + \tfrac{1}{3} m_4 l_4^2 + \tfrac{1}{2} m_4 l_4 l_3 c_4 \\
M(3,4) &= M(4,3) = \tfrac{1}{3} m_4 l_4^2 + \tfrac{1}{2} m_4 l_3 l_4 c_4 \\
M(1,2) &= M(2,1) = M(1,3) = M(3,1) = M(1,4) = M(4,1) = 0
\end{aligned}
\tag{A.1}
$$

$$
\begin{aligned}
G(1,1) &= 0 \\
G(1,2) &= \left(\tfrac{1}{2} m_2 g l_2 + m_3 g l_2 + m_4 g l_2\right) c_2 + \left(\tfrac{1}{2} m_3 g l_3 + m_4 g l_3\right) c_{23} + \tfrac{1}{2} m_4 g l_4 c_{234} \\
G(3,1) &= \left(\tfrac{1}{2} m_3 g l_3 + m_4 g l_3\right) c_{23} + \tfrac{1}{2} m_4 g l_4 c_{234} \\
G(4,1) &= \tfrac{1}{2} m_4 g l_4 c_{234}
\end{aligned}
\tag{A.2}
$$

$$
\begin{aligned}
C(1,1) &= \begin{pmatrix} -\tfrac{1}{3}m_2 l_2^2 \mathrm{s}_{22} - m_3 l_2^2 \mathrm{s}_{22} - \tfrac{1}{2}m_3 l_2 l_3 \mathrm{s}_{223} - \tfrac{1}{3}m_3 l_3^2 \mathrm{s}_{2233} - \tfrac{1}{2}m_4 l_3^2 \mathrm{s}_{2233} - m_4 l_2^2 \mathrm{s}_{22} \\ -2m_4 l_2 l_3 \mathrm{s}_{223} - \tfrac{1}{2}m_4 l_4 l_3 \mathrm{s}_{22334} - m_4 l_4 l_2 \mathrm{s}_{2234} - \tfrac{1}{3}m_4 l_4^2 \mathrm{s}_{223344} \end{pmatrix} \dot{\theta}_1 \dot{\theta}_2 \\
&+ \begin{pmatrix} -m_3 l_2 l_3 \mathrm{s}_{23} \mathrm{c}_2 - \tfrac{1}{3}m_3 l_3^2 \mathrm{s}_{2233} - \tfrac{1}{2}m_4 l_3^2 \mathrm{s}_{2233} - 2m_4 l_3 l_2 \mathrm{s}_{23} \mathrm{c}_2 - m_4 l_3 l_4 \mathrm{s}_{22334} \\ -m_4 l_2 l_4 \mathrm{s}_{234} \mathrm{c}_2 - \tfrac{1}{3}m_4 l_4^2 \mathrm{s}_{223344} \end{pmatrix} \dot{\theta}_1 \dot{\theta}_3 \\
&- \left(+m_4 l_4 l_3 \mathrm{s}_{234} c_{23} + m_4 l_4 l_2 \mathrm{s}_{234} c_2 + \tfrac{1}{3}m_4 l_4^2 \mathrm{s}_{223344} \right) \dot{\theta}_1 \dot{\theta}_4 \\
C(2,1) &= \begin{pmatrix} \tfrac{1}{6}m_2 l_2^2 \mathrm{s}_{22} + \tfrac{1}{2}m_3 l_2^2 \mathrm{s}_{22} + \tfrac{1}{2}m_3 l_2 l_3 \mathrm{s}_{223} + \tfrac{1}{6}m_3 l_3^2 \mathrm{s}_{2233} + \tfrac{1}{2}m_4 l_3^2 \mathrm{s}_{2233} + \tfrac{1}{2}m_4 l_2^2 \mathrm{s}_{22} \\ +m_4 l_2 l_3 \mathrm{s}_{223} + \tfrac{1}{2}m_4 l_4 l_3 \mathrm{s}_{22334} + \tfrac{1}{2}m_4 l_2 l_4 \mathrm{s}_{2234} + \tfrac{1}{6}m_4 l_4^{\,2} \mathrm{S}_{223344} \end{pmatrix} \dot{\theta}_1^2 \\
&+ \left(-\tfrac{1}{2}m_3 l_2 l_3 \mathrm{s}_3 - m_4 l_3 l_2 \mathrm{s}_3 \right) \dot{\theta}_3^2 + \left(-\tfrac{1}{2}m_4 l_4 l_3 \mathrm{s}_4 \right) \dot{\theta}_4^2 + \left(-m_4 l_3 l_4 \mathrm{s}_4 \right) \dot{\theta}_2 \dot{\theta}_4 \\
&+ \left(-m_3 l_3 l_2 \mathrm{s}_3 - 2m_4 l_3 l_2 \mathrm{s}_3 \right) \dot{\theta}_2 \dot{\theta}_3 + \left(-m_4 l_3 l_4 \mathrm{s}_4 \right) \dot{\theta}_3 \dot{\theta}_4 \\
C(3,1) &= \begin{pmatrix} \tfrac{1}{2}m_3 l_3 l_2 \mathrm{s}_{23} c_2 + \tfrac{1}{6}m_3 l_3^2 \mathrm{s}_{2233} + \tfrac{1}{2}m_4 l_3^2 \mathrm{s}_{2233} + m_4 l_2 l_3 \mathrm{s}_{23} c_2 + \tfrac{1}{2}m_4 l_4 l_3 \mathrm{s}_{22334} \\ +\tfrac{1}{2}m_4 l_4 l_2 \mathrm{s}_{234} c_2 + \tfrac{1}{6}m_4 l_4^2 \mathrm{s}_{223344} \end{pmatrix} \dot{\theta}_1^2 \\
&+ \left(\tfrac{1}{3}m_3 l_2 l_3 \mathrm{s}_3 + m_4 l_3 l_2 \mathrm{s}_3 \right) \dot{\theta}_2^2 + \left(-m_4 l_3 l_4 \mathrm{s}_4 \right) \dot{\theta}_2 \dot{\theta}_4 - \left(\tfrac{1}{2}m_4 l_3 l_4 \mathrm{s}_4 \right) \dot{\theta}_4^2 + \left(-m_4 l_3 l_4 \mathrm{S}_4 \right) \dot{\theta}_3 \dot{\theta}_4 \\
C(4,1) &= \left(\tfrac{1}{2}m_4 l_4 l_3 \mathrm{s}_{234} c_{23} + \tfrac{1}{2}m_4 l_4 l_2 \mathrm{s}_{234} \mathrm{c}_2 + \tfrac{1}{6}m_4 l_4^2 \mathrm{s}_{223344} \right) \dot{\theta}_1^2 + \left(\tfrac{1}{2}m_4 l_4 l_3 \mathrm{s}_4 \right) \dot{\theta}_2^2 \\
&+ \left(\tfrac{1}{2}m_4 l_4 l_3 \mathrm{s}_4 \right) \dot{\theta}_3^2 + \left(m_4 l_4 l_3 \mathrm{s}_4 \right) \dot{\theta}_3 \dot{\theta}_4
\end{aligned}
\tag{A.3}
$$

where, l_i and m_i ($i=1,\ldots,4$) are the length and mass of the i^{th} link, respectively, θ_i and $\dot{\theta}_i$ are the angular position and the angular velocity of the i^{th} joint, respectively, and $c_i = \cos(\theta_i)$, $s_i = \sin(\theta_i)$, $c_{ij} = \cos(\theta_i + \theta_j)$, $s_{ij} = \sin(\theta_i + \theta_j)$, and so forth.