# Control of Redundant Manipulators in Non-Stationary Environments Using Neural Networks and Model Predictive Control

**Ashkan M. Jasour, Mohammad Farrokhi**

*Department of Electrical Engineering*
*Iran University of Science and Technology*
*a.m.jasour@ee.iust.ac.ir, farrokhi@iust.ac.ir*

**Abstract:** In this paper, a nonlinear model predictive control (NMPC) is designed for redundant robotic manipulators in non-stationary environments. Using NMPC, the end-effector of the robot could track predefined desired path and reaches a moving target in the Cartesian space, while at the same time avoids collision with moving obstacles and singular configurations in the workspace. To avoid collisions with moving obstacles and capturing moving target, the future position of obstacles and the moving target in 3D space is predicted using artificial neural networks. Using online training of the neural network, no knowledge about obstacles and motion of the moving object is required. The nonlinear dynamic of the robot including actuators dynamic is also considered. Numerical simulations performed on a 4DOF redundant spatial manipulator actuated by DC servomotors, shows effectiveness of the proposed method.

*Keywords:* robotic manipulator, predictive control, manoeuvring target, obstacle avoidance, neural networks.

## 1. INTRODUCTION

Robotic manipulators are increasingly used in many tasks such as industry, medicine, space, and humanoid robots. Therefore, efficient control strategy for manipulators plays a key role in various applications. In some applications, manipulators are needed to track a predefined path in three-dimensional space or move to desired position in such a way that no collision with obstacles in the environment occurs. Therefore, feasible paths for each joint of the robot must be determined in such a way that the required criteria are met. In some cases, these criteria cannot be met simultaneously. In these cases, implementation of redundant manipulators is necessary. High degrees of freedom for redundant manipulators lead to infinite number of possible joint positions for the same pose of the end-effector. Hence, for a given end-effector coordination in the Cartesian space, the robot can reach it in many different configurations, among which, the collision free and singular free situations must be selected. Finding feasible paths for joints of redundant manipulators for a given end-effector coordination is called redundancy resolution (Conkur, 2005). Redundancy resolution and obstacle avoidance are already considered in papers. Using the gradient projection technique, redundancy is solved considering the obstacle avoidance (Chen *et al.*, 2002). In task-priority redundancy resolution technique, the tasks are performed based on the order of priority, where path tracking is given the first priority and the obstacle avoidance or the singularity avoidance is given the second priority (Chiacchio, *et al.*, 1991; Nakamuro, 1991). This technique yields locally 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; Yoshikawa, 1993). Optimization techniques, which minimize a cost function subject to constraints, like the end-effector path tracking and obstacle avoidance, are not suitable for on-line applications (Nakamuro, 1991). The space-time and velocity adjusting methods are proposed in (Lee, 1989; Bagchi and Hatwal, 1992) on the problem of obstacle avoidance. Choi and Kim have used the sum of inverse of predicted collision distances between manipulator links and static obstacles to avoid collisions (2000). Mbede *et al.* have employed the artificial potential field to solve this problem (2000). Saramago and Steffen have solved the problem of collision with moving obstacles using the distances between the potentially colliding parts and the motion is represented using translation and rotational matrices (2000).

Works related to the motion of robot manipulators and obstacle avoidance using intelligent methods are limited as compared to the classical methods. Bagchi and Hatwal have proposed a method based on fuzzy logic for motion planning of a robot manipulator amongst unknown moving obstacles (1992). In their method, sensory data are analyzed through a fuzzy controller, which estimates whether a collision is imminent, and if so, employs a geometric approach to compute the joint movements necessary to avoid the collision. Zhang and Wang have used neural networks for inverse kinematic problem for obstacle avoidance in kinematically redundant manipulators (2004).

This paper presents a solution for obstacle avoidance with moving obstacles and at the same time capturing a moving target using intelligent methods. To this end, the future position of obstacles and the moving target in 3D space is predicted using artificial neural networks. Using online training of the neural network, no knowledge about obstacles and motion of the moving object is required.

For the control of the manipulator, nonlinear model predictive control (NMPC) method is presented for redundancy resolution considering moving obstacles and singularity avoidance. Although the model predictive control

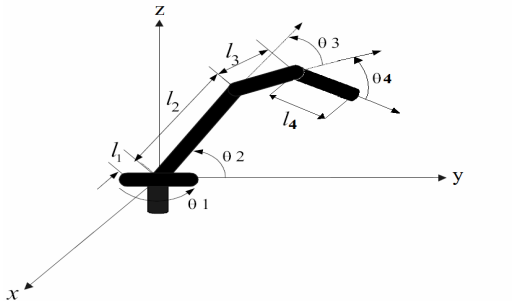


Fig. 1. Schematic diagram of a 4DOF spatial manipulator.

(MPC) is not a new control method, works related to manipulator robots using the MPC is limited. Most of the related works are about joint space control and end-effector coordinating. Using MPC, the joints of robotic manipulator track the trajectories defined for each joint. Therefore, redundancy resolution as well as obstacle and singular avoidance have not been considered. The linear MPC is used in (Valle *et al.*, 2002; Kim and Han, 2004; Vivas and Mosquera, 2005) and NMPC is employed in (Wroblewski, 2004; Hedjar *et al.*, 2005; Poignet and Gautier, 2000) for joint space control of manipulators.

In our previous work, the Fuzzy NMPC and Adaptive NMPC were introduced for online path tracking and obstacle avoidance of redundant manipulator robots (Jasour and Farrokhi, 2009a; 2009b; 2010). In this paper, using NMPC, the input voltages of DC servomotors of joints are obtained in such a way that the end-effector of a redundant manipulator tracks a predefined path and reaches a moving target in the Cartesian space considering obstacles and singularity avoidance.

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 model predictive control. Section 4 explains prediction of moving obstacles and the target position using neural networks. Section 5 implements the neuro model of the moving obstacle and the NMPC to control a 4DOF manipulator. Simulation results are presented in Section 6. Conclusions are drawn in Section 7.

## 2. DYNAMIC OF ROBOTIC MANIPULATOR

Schematic diagram of a 4DOF spatial redundant manipulator robot is shown in Fig. 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 the robot manipulator including dynamics of the armature-controlled DC servomotors that drive links can be obtained using the Lagrangian method as (Hedjar *et al.*, 2002; Lewis *et al.*, 2004)

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

where $\boldsymbol{\theta} \in R^n$ is the angular position of joints, $\mathbf{M}(\boldsymbol{\theta}) \in R^{n\times n}$ is the symmetric and positive definite robot inertia matrix, $\mathbf{C}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}) \in R^n$ is the centrifugal and Coriolis robot force vector, $\mathbf{G}(\boldsymbol{\theta}) \in R^n$ is the robot gravity vector, $\mathbf{D}(\dot{\boldsymbol{\theta}})\,\theta \in R^n$ is the vector for joints friction of the links, $\mathbf{K}_T \in R^{n\times n}$ is the diagonal matrix of the motor torque constant, $\mathbf{J}_m \in R^{n\times n}$ is the diagonal matrix of the motor moment inertia, $\mathbf{B}_m \in R^{n\times n}$ is the diagonal matrix of motor torsional damping coefficients, $\mathbf{V}_t \in R^n$ is the vector of motor armature input voltages, $\mathbf{R}_a \in R^{n\times n}$ is the diagonal matrix of motor armature resistances, and $\mathbf{K}_E \in R^{n\times n}$ is the diagonal matrix of the motor back electromotive force (EMF) coefficients, $\Gamma \in R^{n\times n}$ is a diagonal positive definite matrix representing the gear ratios for $n$ joints and $n$ is the degree of freedom, which is equivalent to four for the robot considered in this paper. The above matrix and vectors are given in Appendix. According to Eq. (2), the armature input voltages are considering as control efforts, respectively.

## 3. MODEL PREDICTIVE CONTROL

Unlike classical control methods, where control actions are calculated based on the past output(s) of the system, the MPC is a model-based optimal controller, which uses predictions of system outputs to calculate the control law (Allgower *et al.*, 2004; Findeisen *et al.*, 2003).

Based on measurements obtained at the sampling time $k$, the controller predicts the output of the system over the prediction horizon $N_P$ in future instances using the system model and determines the input over the control horizon $N_C \le N_P$ such that a predefined cost function is minimized. To incorporate the feedback, only the first member of the obtained input is applied to the system until the next sampling time (Allgower *et al.*, 2004). Using new measurements 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 follow:

$$u = \arg\min_{u}(J(k)) \tag{3}$$

subject to the following constraints:

$$\begin{aligned} & x(k \mid k) = x_0 \\ & u(k+j \mid k)) = u(k+N_C \mid k)),\ j \ge N_C \\ & x(k+j+1 \mid k) = f_d(x(k+j \mid k), u(k+j \mid k)) \\ & y(k+j+1 \mid k) = h_d(x(k+j+1 \mid k)) \\ & x_{\min} \le x(k+j+1 \mid k) \le x_{\max} \\ & u_{\min} \le u(k+j \mid k) \le u_{\max} \end{aligned} \tag{4}$$

where $j \in [0\ ,\ N_P\text{-}1]$, $x$ and $u$ are states and input of the system, $x_0$ is the initial condition, and $f_d$ and $h_d$ are the model of the system used for prediction. The notation $a(m|n)$ indicates the value of $a$ at instant $m$ predicted at instant $n$. Intervals $[x_{\min}\ ,\ x_{\max}]$ 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. MPC schemes that are based on the nonlinear model of the system or consider a non-quadratic cost function and nonlinear constrains on inputs and states are called Nonlinear MPC (Findeisen *et al.*, 2003). The optimization problem (3) must be solved at each sampling time $k$, yielding a sequence of optimal control law as $\{u^*(k \mid k), ..., u^*(k+N_C \mid k)\}$. For optimization, the SQP method is used in this paper (Fletcher, 1987).

## 4. NEURO-PREDICTIVE MODEL OF DYNAMIC ENVIRONMENT

The dynamic environments contain movable objects such as moving obstacles and/or targets. In order to implement the MPC to control the manipulator in dynamic environment, a prediction model for environment must be obtained such that the future position of the moving objects is predicted over the prediction horizon.

In this paper, a neural model using the multilayer perceptron (MLP) is used for this purpose. One of the main advantages of using NNs for prediction is that by employing the online training scheme, no prior knowledge about the motion of obstacles is needed. The prediction is based on the coordinate of the moving object at the past sampling times; the predictor output is the coordinate of the moving object at the next sampling time. Using this structure, a one-step-ahead prediction of the moving object position is obtained. However, in the predictive control, multi-step predictions over the prediction horizon are needed. By applying one-step prediction recursively, the multi-step prediction can be achieved. In this case, the outputs of the Neural Network (NN) are considered as inputs for the next step. To acquire high accuracy for the one-step-ahead prediction, the NN structure must be selected carefully. The structure of the employed MLP is shows in Fig. 2. Transfer functions for the hidden and output layer neurons are of hyperbolic tangent and linear types, respectively. To predict the obstacle position, the obstacle coordinates at the current and at the past 2 samples are fed as inputs to the NN; the output of the NN is the obstacle coordinates at the next sampling time (Fig. 2). The same method is employed for predicting the coordinates of the moving target in 3D space.

Training the NNs is performed online using the recursive least squares algorithm (Fletcher, 1987). Using this algorithm at each sampling time $k$, summation of the matching errors for all input-output pairs up to the $k^{\text{th}}$ sample is minimized. However, in this paper, summation of errors for the last $N$ input-output pairs is considered for minimization. In this way, the training time can remain constant and $N$ can be selected in such a way that training time over every sampling time does not exceed the sampling rate.

## 5. CONTROL OF MANIPULATOR USING NMPC

The purpose of the manipulator control in a dynamic environment is to obtain a control law such that the end-effector tracks a predefined path or reaches a moving target in the Cartesian space and at the same time avoids collisions with moving obstacles or singular configurations in the work space. To this end, the NMPC method is implemented in this paper. The block diagram of NMPC is shown in Fig. 3. As explained in Section 3, in order to obtain the control law, an appropriate cost function should be defined for the NMPC algorithm.

For the path tracking and object capturing, the cost function must have direct relation with the distance between the end-effector and the desired coordinates. On the other hand, for obstacles avoidance the cost function must have an inverse relationship with the distance between the obstacle and the manipulator links. In this paper the cost function is defined as

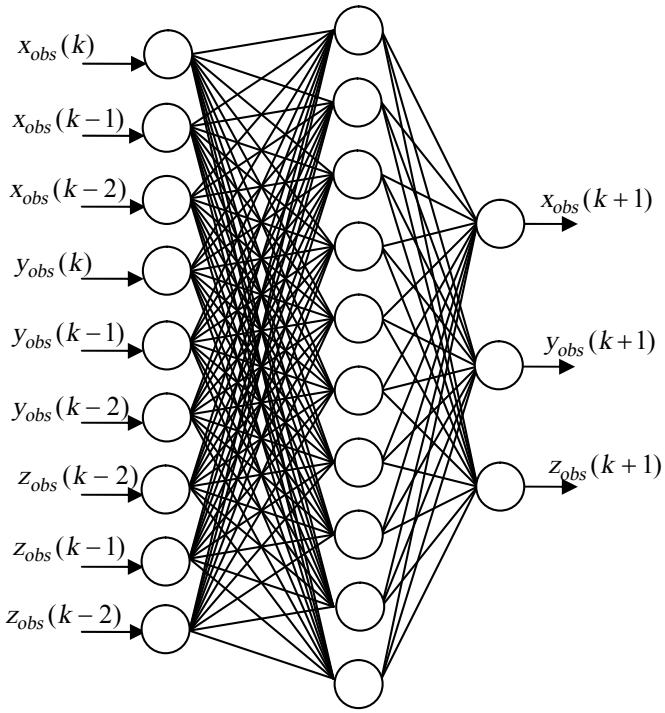



Fig. 2. Neural network structure.

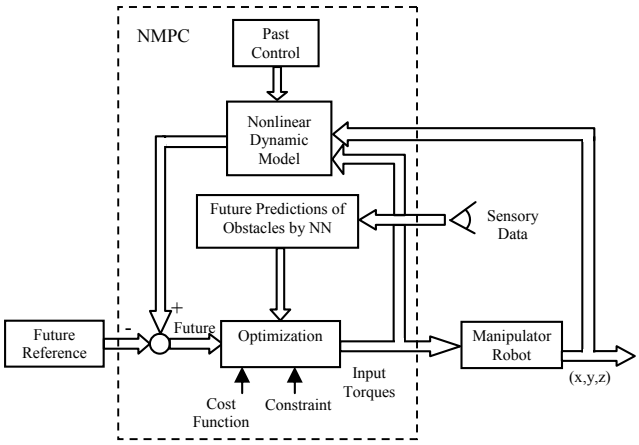



Fig. 3. Block diagram of NMPC

$$J=\sum_{j=1}^{N_P} Q\left(\frac{e^{D_P(k+j|k)}-e^{D_{P\min}}}{e^{D_{P\max}}-e^{D_{P\min}}}\right)+R\left(\frac{e^{-D_O(k+j|k)}-e^{-D_{O\max}}}{e^{-D_{O\min}}-e^{-D_{O\max}}}\right), \quad (5)$$

where $D_P$ is the Euclidean distance between the end-effector and the desired position related to predefined path (or moving target), $D_O$ 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 $n$. Intervals $[D_{P\min},\ D_{P\max}]$ and $[D_{O\min},\ D_{O\max}]$ are the range of variations for $D_P$ and $D_O$, respectively. Using the cost function (5), the two terms for the path tracking and the obstacle avoidance are normalized to [0 1].

The NMPC in this paper uses the nonlinear dynamic model of the manipulator in the optimization of the cost function. Substituting $(\theta(k+1)-\theta(k))/T$ for $\dot{\theta}$ in dynamic Eq. (2), a one-step-ahead prediction for joints angles can be expressed as

$$\theta(k+1) = f_d(\theta(k), V_t(k)), \quad (6)$$

where $k$ is the sampling time and $T$ is the sampling rate. Using the forward kinematics (1), a one-step-ahead prediction of the end-effector position can be obtained. However, in the predictive control, this equation is used for multi-step predictions over the prediction horizon by recursively applying the one-step-ahead prediction.

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

$$V_{t\min} \le V_t \le V_{t\max}, \quad (7)$$

where $V_{t\min}$ and $V_{t\max}$ stand for the lower and the upper bound of input voltages of servo DC motors, respectively.

In a singular configuration, theoretically the joint velocities become infinite. Therefore, the following constrain must be taken into account:

$$\dot{\theta}_{\min} \le \dot{\theta} \le \dot{\theta}_{\max}, \quad (8)$$

where $\dot{\theta}_{\min}$ and $\dot{\theta}_{\max}$ are the lower and the upper bound of the joints velocity, respectively.

By incorporating constrains (7) and (8) into the cost function, the optimization problem will be solved.

## 6. SIMULATION RESULTS

Parameters of the robotic manipulator and DC servomotors are given in Tables I and II, respectively. The sampling rate $T$ is equal to 0.5 sec. Furthermore, based on the robot parameters, it is assumed that $\dot{\theta}_{\min}$ and $\dot{\theta}_{\max}$ are equal to $-400$ and 400 degree/s, respectively. It is assumed that the velocity and acceleration of the moving objects are within an acceptable range as compared to the manipulator joints limitations. NN consists of three layers: 9 neurons in the input layer, 10 neurons in the hidden layer, and 3 neurons in the output layer. To show the prediction accuracy of the NN, a highly nonlinear motion for the target and obstacles is considered here. For the cost function (5), according to the length of manipulator links given in Table 1, $D_{P\min}$ and $D_{P\max}$ are equal to 0 and 2.4 meter, respectively and $D_{O\min}$ and $D_{O\max}$ are equal to 0 and 1.2 meter, respectively.

In the first part of simulations, a rectangular path in the Cartesian space with static and moving obstacles inside the work space is considered (Fig. 4). The motion equation of the moving obstacle is defined as

$$\begin{aligned} R_1(t) &= 0.1\times\cos(0.06\times\pi\times t) + 0.5 \\ R_2(t) &= 0.2\times\cos(0.01\times\pi\times t) \\ x_{obs}(t) &= R_1(t)\times\cos(0.01\times\pi\times t) \\ y_{obs}(t) &= R_1(t)\times\sin(0.01\times\pi\times t) \\ z_{obs}(t) &= -0.1 + R_2(t)\times\sin(0.05\times\pi\times t) \end{aligned} \quad (9)$$

where $t$ is the time and $[x_{obs}(t), y_{obs}(t), z_{obs}(t)]$ are the obstacle coordinates in 3D space (Fig. 5). Simulation results for the case of $N_p = 5$, $N_C = 2$, and $N = 100$ are shown in Figs. 6 to 9. Fig. 6 shows that the end-effector tracks the predefined path in the workspace. According to Figs. 7, the manipulator tracks the path in a way that no collision with obstacles occurs. Fig. 8 shows the ability of NNs to predict coordinates of the obstacle at $k = 4$, $k = 40$ and the maximum prediction error over the prediction horizon at each sampling time. As this figure shows, the NNs are able to learn quickly the behaviour of the moving obstacle. Fig. 9 shows that the training time for NNs at every sampling time is less than the sampling rate $T$=0.5 s, yielding a method that can be easily implemented in practice.

In the second part of simulations, a moving target in the Cartesian space along with the static and moving obstacles inside the work space is considered (Fig. 10). The motion equation of the moving obstacle is defined as:

$$\begin{aligned} R_3(t) &= 0.2\times\cos(0.06\times\pi\times t) + 0.5 \\ x_t(t) &= R_3(t)\times\cos(0.01\times\pi\times t) \\ y_t(t) &= R_3(t)\times\sin(0.01\times\pi\times t) \\ z_t(t) &= 0.5 + 0.2\times\cos(0.01\times t) \end{aligned} \quad (10)$$

where $t$ is the time and $[x_t(t), y_t(t), z_t(t)]$ are the target coordinates in 3D space (Fig. 11). Simulation results for the case of $N_p = 5$, $N_C = 2$, and $N = 100$ are shown in Figs. 12 to 16. Fig. 12 shows that the end-effector reaches the target in a reasonable time. Figs. 13, 14 and 15 show positions, velocities, and input voltages of manipulator joints, respectively. According to Fig. 16, manipulator robot captures the moving target in a way that no collision with obstacles occurs.

## 7. CONCLUSION

In this paper, a neuro MPC method was proposed for obstacle avoidance and catching moving targets by robotic manipulators in non-stationary environments. For this reason, two terms were introduced in the cost function, one for the tracking problem and the other one for the obstacle avoidance. Moreover, by introducing constrains to the joints velocities, singularities were avoided. To avoid robot collision with moving obstacles and to capture moving targets, the future position of obstacles and moving targets in 3D space were predicted using artificial neural networks. Moreover, by using the online training scheme of NNS, no prior knowledge about obstacles and target motion is needed.

**Table 1. Parameters of Manipulator**

| Link | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| $L$ (m) | 1 | 0.5 | 0.4 | 0.3 |
| $M$ (kg) | 1 | 0.5 | 0.4 | 0.3 |

**TABLE 2. PARAMETERS OF DC SERVO MOTORS**

| 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 |
| $B_m$ | $64\times10^{-4}$ | $64\times10^{-4}$ | $64\times10^{-4}$ | $64\times10^{-4}$ |
| $J_m$ | 0.2 | 0.2 | 0.2 | 0.2 |
| R | 1:100 | 1:100 | 1:10 | 1:10 |
| $V_t$ | 24 | 24 | 24 | 24 |

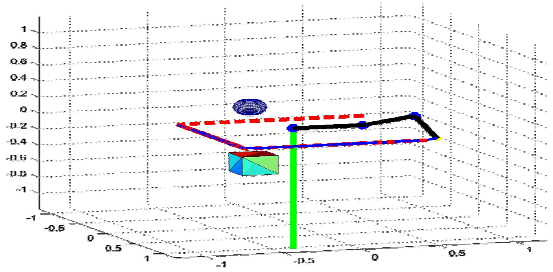


Fig. 4. desired rectangular path with static (cube) and dynamic (sphere) obstacle in work space.

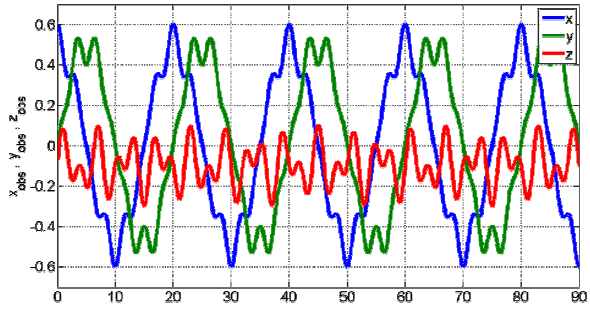



Fig. 5. Coordinates of moving obstacle

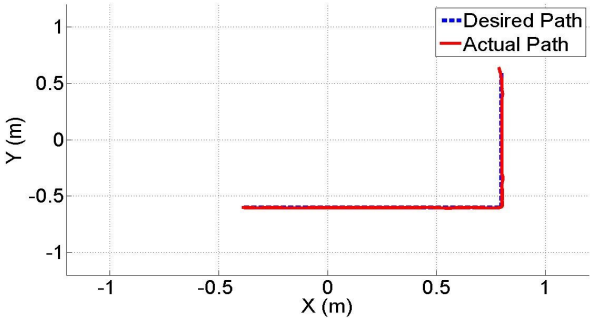



Fig. 6. Desired and actual end-effector path.

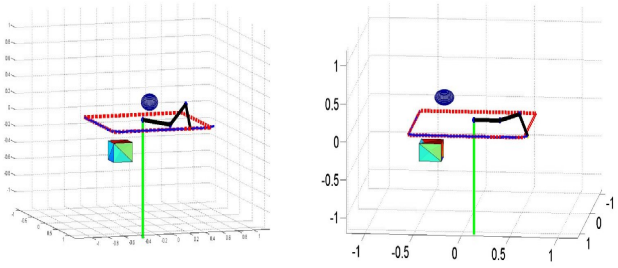
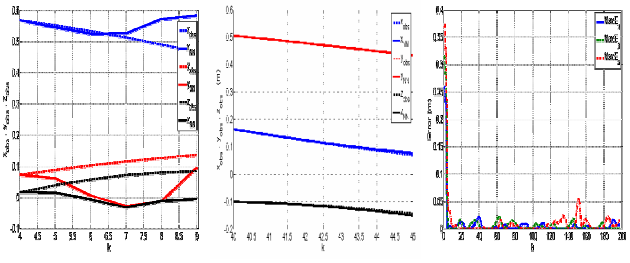


Fig. 7. Path tracking and obstacle avoidance.



(a) (b) (c)

Fig. 8. (a) and (b) NN prediction over prediction horizon at sampling time $k = 4$ and $k = 40$, (c) maximum prediction error of NN at each sampling time.

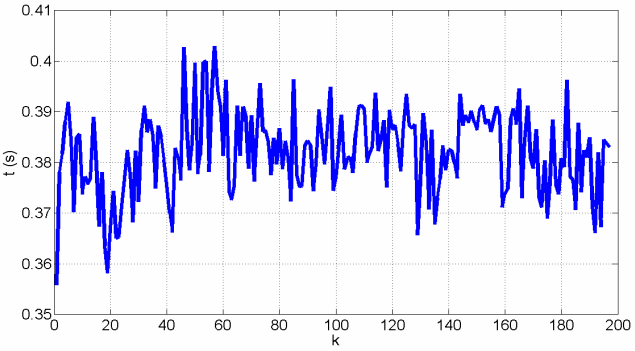



Fig. 9. training time for NNS at every sampling time.

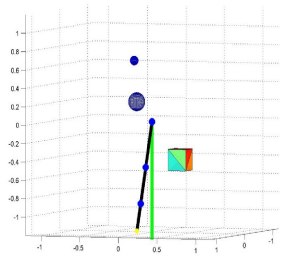


Fig. 10. moving target (small moving sphere), static (cube) & dynamic (sphere) obstacles.

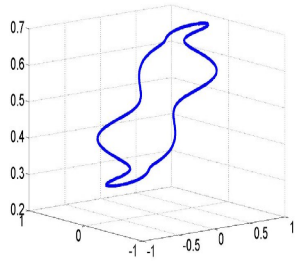


Fig. 11. coordinates of target.

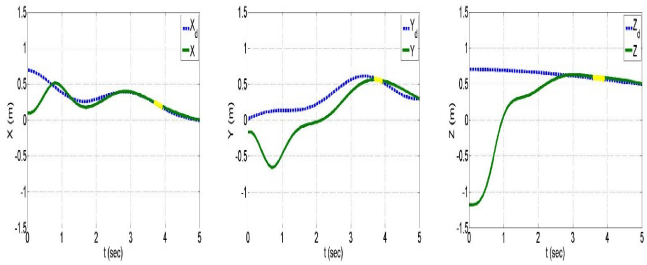



Fig. 12. path of moving target and end-effector.

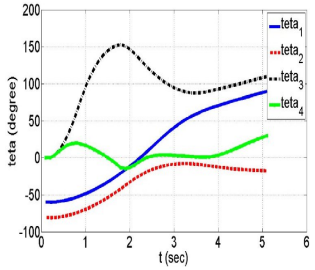



Fig. 13. Positions of joints.

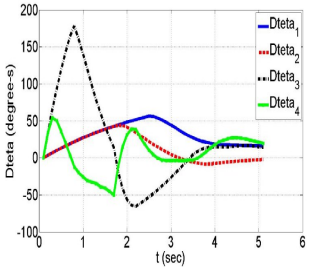



Fig. 14. Velocities of joints.

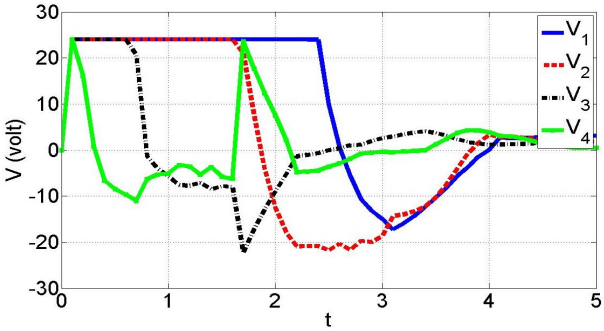



Fig. 15. Input voltages of servo DC motor.

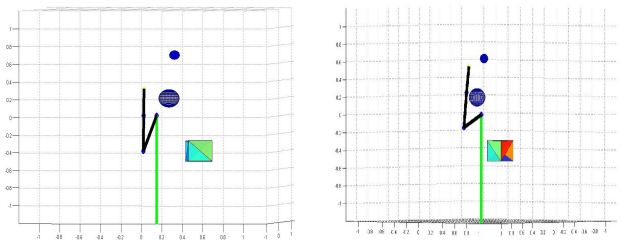


Fig. 16. Moving target capturing and obstacle avoidance.

## References


Allgower, F., Findeisen, R., and Nagy, Z. K. (2004). Nonlinear model predictive control: from theory to application. *J. Chin. Inst. Chem. Engrs*, 35, 299-315.

Bagchi, A. and Hatwal, H. (1992). Fuzzy Logic-based Techniques for Motion Planning of a Robot Manipulator Amongst Unknown Moving Obstacles, *Robotica*, 10, 563-574.

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

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*, 10, 410-426.

Chen, J.L., Liu, J.S., Lee, and W.C., Liang, T.C. (2002). On-line multi-criteria based collision-free posture generation of redundant manipulator in constrained workspace. *Robotica*, 20, 625–636.

Choi, S. and Kim, B.K. (2000). Obstacle avoidance control for redundant manipulators using collidability measure, *Robotica* 18, 143-151.

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

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

Fletcher, R. (1987). *Practical Methods of Optimization*, John Wiley & Sons, NY, USA.

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*, 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.

Jasour, A.M and Farrokhi, M. (2010). Fuzzy Improved Adaptive Neuro-NMPC for On-Line Path Tracking and Obstacle Avoidance of Redundant Robotic Manipulators. *International Journal of Automation and Control*, Vol. 4, No. 2, pp. 177 – 200.

Jasour, A.M and Farrokhi, M. (2009a). Path Tracking and Obstacle Avoidance of Redundant Robotic Arms Using Fuzzy NMPC, *Proc. American Control Conference*, Missouri, USA, June, pp.1353-1358.

Jasour, A.M and Farrokhi, M. (2009b). Adaptive Neuro-NMPC Control of Redundant Robotic Manipulators for Path Tracking and Obstacle Avoidance. *Proc. European Control Conference*, Budapest, Hungary, August, pp. 2181:2186.

Kim, J.K. and Han, M.C. (2004). Adaptive robust optimal predictive control of robot manipulators. *30th IEEE Industrial Electronics Conference*, Busan, Korea.

Lee, B.H. (1989). Constraint identification in time-varying obstacle avoidance for mechanical manipulators. *IEEE Trans. Syst. Man Cyber.*, 19, 140-143.

Lewis, F.L., Abdallah, C.T., and Dawson, D.N. (2004). *Control of Robot Manipulators Theory and Practice*, Marcel Dekker Inc., NY.

Mbede, J.B., Huang, X., and Wang, M. (2000). Fuzzy motion planning among dynamic obstacles using artificial potential fields for robot manipulators, *Robotics and Autonomous Systems*, 32 61–72.

Nakamuro, Y. (1991). *Advanced Robotics Redundancy and Optimization*. Addison-Wesley, Reading, MA, USA.

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

Saramagoa S.F.P. and Steffen, V. (2000). Optimal trajectory planning of robot manipulators in the presence of moving obstacles. *Mechanism and Machine Theory*, 35, 1079-1094.

Valle, F., Tadeo, F., and Alvarez, T. (2002). Predictive control of robotic manipulators", *IEEE Int. Conf. on Control Applications*, 203-208, Glasgow, Scotland, UK.

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

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

Yoshikawa, T. (1993). Analysis and control of robot manipulators with redundancy. *First International Symposium on Robotic Research*. MIT Press, Cambridge, MA, USA.

Zhang, Y. and Wang, J. (2004). Obstacle avoidance for kinematically redundant manipulators using a dual neural network. *IEEE Transactions on Systems, Man, and Cybernetics-Part B: Cybernetics*, 34, 752-759.


## Appendix

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

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

$$
\begin{aligned}
C(1,1) &= \begin{pmatrix} -\frac{1}{3}m_2 l_2^2 s_{22} - m_3 l_2^2 s_{22} - m_3 l_2 l_3 s_{223} - \frac{1}{3}m_3 l_3^2 s_{2233} - m_4 l_3^2 s_{2233} - m_4 l_2^2 s_{22}\\ -2m_4 l_2 l_3 s_{223} - m_4 l_4 l_3 s_{22334} - m_4 l_4 l_2 s_{2234} - \frac{1}{3}m_4 l_4^2 s_{223344}\end{pmatrix}\dot\theta_1\dot\theta_2\\
&\quad + \begin{pmatrix} -m_3 l_2 l_3 s_{23}c_2 - \frac{1}{3}m_3 l_3^2 s_{2233} - m_4 l_3^2 s_{2233} - 2m_4 l_3 l_2 s_{23}c_2 - m_4 l_3 l_4 s_{22334}\\ -m_4 l_2 l_4 s_{234}c_2 - \frac{1}{3}m_4 l_4^2 s_{223344}\end{pmatrix}\dot\theta_1\dot\theta_3\\
&\quad + \left(-m_4 l_4 l_3 s_{234}c_{23} - m_4 l_4 l_2 s_{234}c_2 - \frac{1}{3}m_4 l_4^2 s_{223344}\right)\dot\theta_1\dot\theta_4\\
C(2,1) &= \begin{pmatrix} \frac{1}{6}m_2 l_2^2 s_{22} + \frac{1}{2}m_3 l_2^2 s_{22} + \frac{1}{2}m_3 l_2 l_3 s_{223} + \frac{1}{6}m_3 l_3^2 s_{2233} + \frac{1}{2}m_4 l_3^2 s_{2233}\\ +\frac{1}{2}m_4 l_2^2 s_{22} + m_4 l_2 l_3 s_{223} + \frac{1}{2}m_4 l_4 l_3 s_{22334} + \frac{1}{2}m_4 l_2 l_4 s_{2234}\\ +\frac{1}{6}m_4 l_4^2 s_{223344}\end{pmatrix}\dot\theta_1^2\\
&\quad + \left(-\frac{1}{2}m_3 l_2 l_3 s_3 - m_4 l_3 l_2 s_3\right)\dot\theta_3^2 + \left(-\frac{1}{2}m_4 l_4 l_3 s_4\right)\dot\theta_4^2 + \left(-m_4 l_3 l_4 s_4\right)\dot\theta_2\dot\theta_4\\
&\quad + \left(-m_3 l_3 l_2 s_3 - 2m_4 l_3 l_2 s_3\right)\dot\theta_2\dot\theta_3 + \left(-m_4 l_3 l_4 s_4\right)\dot\theta_3\dot\theta_4\\
C(3,1) &= \begin{pmatrix} \frac{1}{2}m_3 l_3 l_2 s_{23}c_2 + \frac{1}{6}m_3 l_3^2 s_{2233} + \frac{1}{2}m_4 l_3^2 s_{2233} + m_4 l_2 l_3 s_{23}c_2 + \frac{1}{2}m_4 l_4 l_3 s_{22334}\\ +\frac{1}{2}m_4 l_4 l_2 s_{234}c_2 + \frac{1}{6}m_4 l_4^2 s_{223344}\end{pmatrix}\dot\theta_1^2\\
&\quad + \left(\frac{1}{2}m_3 l_2 l_3 s_3 + m_4 l_3 l_2 s_3\right)\dot\theta_2^2 + \left(-m_4 l_3 l_4 s_4\right)\dot\theta_2\dot\theta_4 + \left(-\frac{1}{2}m_4 l_3 l_4 s_4\right)\dot\theta_4^2\\
&\quad + \left(-m_4 l_3 l_4 s_4\right)\dot\theta_3\dot\theta_4\\
C(4,1) &= \left(\frac{1}{2}m_4 l_4 l_3 s_{234}c_{23} + \frac{1}{2}m_4 l_4 l_2 s_{234}c_2 + \frac{1}{6}m_4 l_4^2 s_{223344}\right)\dot\theta_1^2 + \left(\frac{1}{2}m_4 l_4 l_3 s_4\right)\dot\theta_2^2\\
&\quad + \left(\frac{1}{2}m_4 l_4 l_3 s_4\right)\dot\theta_3^2 + \left(m_4 l_4 l_3 s_4\right)\dot\theta_3\dot\theta_4
\end{aligned}
\qquad (A.3)
$$

where, $l_i$ and $m_i$ ($i=1,\ldots,4$) are the length and mass of the $i^{th}$ link, respectively, $\theta_i$ 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.