Task Time Optimization of a Robot Manipulator using Artificial Neural Network and Genetic Algorithm

Akash Dutt Dubey
Dept. of Computer Engineering
IIT (BHU), Varanasi, India

R. B. Mishra
Dept. of Computer Engineering
IIT (BHU), Varanasi, India

A. K. Jha
Dept.of Mech. Engineering
IIT (BHU), Varanasi, India

ABSTRACT

In this paper we have proposed an evolutionary method to optimize the task time of robot manipulators. Tasks can be planned in joint space with respect to robot joints or in Cartesian space with respect to robot end effector under kinodynamic constraints. Genetic algorithm is implemented to optimize the parameters associated with the selected motion trajectory profile. These optimized results were then taken as the training data to train an artificial neural network which is used to obtain task time, velocity, accelerations and torques required by each motor to perform a given task. The method adopted in this study can be applied to any serial redundant or non-redundant manipulator that has rigid links and known kinematic and dynamic models with free motions or motions along specified paths with obstacle avoidance. The robot kinematic and dynamic models and the optimization method are developed in MATLAB.

Keywords

Pick and Place, Artificial Neural Network, Genetic Algorithm, Robot Manipulator, End effector, Mobile robot.

1. INTRODUCTION

The robotic arm is used to pick and place object at the specified locations in a given hemispherical 3-D space. The pick and place operation finds it applications in the field of pharmaceutical industry, electronics industry, food industry and consumer goods industry. For industrial profitability, manipulators that have the ability to perform these operations in the shortest possible cycle time are required. Till now, two main parallel mechanism families have been used in industry to perform such motions at high speed (more than 10m/s) and high acceleration (10g).

The industrial robots are often required to be highly productive, of high quality and adaptability as the result of competitive manufacturing. Since high productivity can be obtained by finding a minimum time planning strategies for the manipulators, the execution time of a specific task attains high value for commercial robot manipulators.

However, the highly non-linear multi-input dynamics of robot manipulators complicates the process of finding true minimum time [1]. Khan and Roth [2] used the linearization technique to introduce the time-optimal control of robotic manipulators for the first time. The first efficient algorithm for finding the optimal trajectory was developed independently in 1985 by Bobrow et al. and Shin and McKay [3, 4] based on the possibility of parameterising the path with a single scalar variable. Pfeiffer &Johanni and Shiller [5, 6] later on modified and simplified this algorithm. A time optimal control in an unknown environment was proposed by Shiller and Dubowsky [7] using a numerical approximation.

The time optimal motion along the path was formally proven to be bang-bang in acceleration by Chen & Desrochers [8]. Sontag and Sussmann [9, 10] have done more indepth investigation of the properties of the optimal control using Pontryagin minimum principle. Fourquet[11] classified the singular trajectories to simplify the results of the time optimal control. More details on time optimal control can also be found in many references such as [12, 13].

On the other hand, design optimization based on simulation is becoming extremely important due to the vast development in hardware and simulation methods. Simulating the robot performance allows different candidates of robot motion profiles to be evaluated a large number of times before undergoing the real manufacturing process. Then, these candidates can be optimized based on the simulation results using an optimization method. Basically there are two groups of optimization methods used in engineering applications, the gradient-based and non-gradient-based methods.

The gradient based methods have a fast asymptotic convergence and an accurate approximation but they require accurate calculations of derivatives which are a cumbersome task in many engineering optimization problems [14]. To calculate optimal values of design parameters the non-linear programming [15] or stochastic optimization technique [16] can be used.

In this paper, we have designed and developed a robot manipulator which is used for pick and place task in joint space and Cartesian space. This robot manipulator can be activated by entering coordinates in the PC which are then transferred to the onboard microcontroller and thereby trigger motion of the arm [17].

Further, Genetic algorithm method is applied to optimize the task time of serial robot manipulators for particular applications. The considered parameters are candidates of the selected motion trajectory. The kinematic and dynamic models of the manipulator are symbolically derived to allow the use of different candidates in the simulation program. The objective function is set to be a balance between the total task time and the quadratic average of joint torques.

After obtaining the optimized parameters from the genetic algorithm method proposed by Hatim and Jha [18] , we have trained an artificial neural network with the optimized data and created a neural network that gives the optimized task time, $E_a$and E_b as the output in response to the input data of initial and the final positions of the four joints of robot manipulator in the joint space.

The paper is divided into the following sections: Section 2 discusses the previous works that have been used to optimize the task time duration and other parameters using genetic algorithms and artificial neural networks. Section 3 discusses the design of the proposed 4 degree of freedom robotic manipulator and the kinematics of the robot manipulator.

Section 4 describes the implementation of the genetic algorithm and artificial neural network on our work. Section 5 shows the test results of the task time optimization using the genetic algorithm and artificial neural network. Section 6 concludes the work and discusses the future aspects of the work.

2. Mechanism

2.1 Genetic Algorithms:

Genetic algorithm approaches (GAs) were introduced by Goldberg. The basic idea of Genetic Algorithms is the mechanics of natural genetics and natural selection. Each optimization parameter is coded into a gene as for example a real number or a string of bits. The corresponding genes for all parameters form a chromosome, which describes each individual. A chromosome can be represented as an array consisting of real numbers, a binary string or a set of components of a database, depending on the specific problem. Each individual represents a possible solution, and a set of individuals form a population. In the genetic algorithm, the individual which has the highest fitness value has the highest probability of getting selected for generation of next individuals. The generation of the next individuals is done using Crossover, where genes from different parents are combined to produce a child. Then there is also the possibility that a mutation might occur. Finally, the children fitness is compared to the parent fitness, and if the children fitness value is more than the parents', the children are inserted into the population to form a new generation. Applications of GAs in the field of robot trajectory planning have been carried out by several researchers. Parker et. al. [19] presented a genetic algorithm approach which allows additional constraint to be easily specified. Davidor [20] generated the robot trajectory of a predefined end-effector robot by finding the inverse kinematics using Genetic Algorithm. A new method for optimum motion planning based on an genetic algorithm was proposed by Yun and Xi [21] which incorporates kinematics constraints, dynamics constraints as well as control constraints. In 1996, Hirakawa and Kawamura [22] proposed a combination of B-spline trajectory generation and steepest gradient optimization to design an optimal motion planning for redundant manipulators. McAvoy et al. [23] used genetic algorithms for optimal point-to-point motion planning for kinematically redundant manipulators to satisfy both the initial conditions and some other specified criteria. Tian and Collins [24] proposed a genetic algorithm using a floating point representation to search for optimal end-effector trajectory for a redundant manipulator.

2.2 Artificial Neural Networks (ANN)

An artificial neural network (ANN) is an intelligent computing method that utilizes the concept of structure and functional aspects of biological neural networks. A neural network is an interconnected group of artificial neurons which processes information using a connectionist approach to computation. ANN can be considered as an adaptive system which whose final outputs change according to the internal and external information that are fed to the neural network during the training. Modern neural networks are mainly used to find the relationship between the inputs and the outputs as non-linear statistical data modeling tools and identify the patterns in the data. In 1969 Marvin Minsky and Seymour Papert[25] published a book in which some of the limitations of the Perceptron model were discussed. Paul Werbos[26] worked to improve the earlier Perceptron model and created the now popular back-propagation network. Hopfield's approach [27] was not simply to create models but to develop technologies that could be applied to real life problems.

The artificial neural networks were first used by Kawato, Uno, Isobe and Suzuki in 1988 [28] when physiological information and previous models of neural network were used to drive the motors of a robot manipulator. Ciliz and Isik [29] developed a four layer feed forward neural network which mapped the position of the system to the optimal control actions. Safaric and Jezernic [30] applied two neural networks on a two DOF Scara type robot which were joint space neural network and task space neural network. Zhao-Hui Jiang Ishida [31] proposed a control scheme where two parallel subsystems were applied of which one neural network controller which calculated force and the torque required by the robot manipulator.

3. Design of the Proposed Mobile Robot Model

Our proposed mobile robotic manipulator has been designed for the purpose of the pick and place task [32]. The robot manipulator has total four links. All the actuators used in this project, are servo motors. Servo motors are very light weight, accurate and provide large standing torque even at low voltage ratings. They require a specific PWM signal for operation. By varying the duty cycle of the PWM signal the shaft of the motors can be rotated accordingly.

The motor used in the robotic arm are the four mega torque quarter scale servo which provide extreme torque of 20kg.cm at 4.8V and 25kg.cm at 6V. The robotic arm has been made of aluminum sheets which are light weight and provide optimum support to the motors.

The base of the robot manipulator has been made by using fiber glasses which has the dimensions of 30x22 cms and have been separated by a distance of 4.6 cms. There are four wheels which are connected to the base to provide the mobility to the manipulator have a diameter of 3 cms. The distance between the two wheels on one side is approximately 10 cms. These wheels are connected to High Torque DC Geared Motor 300RPM, massive torque of 30Kgcm to make the wheels rotate providing the mobility to the manipulator.

The robotic manipulator has a general purpose robotic gripper which has the ability to be used with 2 servo motors for gripper open/close and wrist rotate. This robotic arm can grip and lift up to 200 gm of load in form of small objects. All the servo motors attached to the robotic arm and the dc motors attached to the wheels are driven by the Rhino control board controller which contains ATMega16 microcontroller, used to control any servo based robot. The details of the links of the robot arms are as follows:

Table I. Parameters of the mobile robot manipulator.

	Link Length	Link Masses	Link Center of Mass [x,y,z]
L1	11.4 cm.	145 gm.	[1.0, 1.0, 1.0]
L2	11.7 cm.	201 gm.	[4.5, 2.0, 2.0]
L3	11.3 cm.	234 gm.	[5.6, 3.0, 3.0]
L4	1 cm.	32 gm.	[0.5, 0.5, 0.5]

3.1 Kinematics of the robot manipulator:

For serial robot manipulator with n degrees of freedom (DoF), the kinematic model can be extracted from the wellknown D-H convention [33]. The governing equations of motions can

be derived using Newton-Euler method or Lagrange-Euler method [1]. The equations of motion without considering the friction effects and motor inertia effects appear in the form:

$$\tau_i(t) = \sum_j M_{ij}\ddot{q}_j + \sum_j \sum_k h_{ijk}\dot{q}_j\dot{q}_k + G_i$$

where $\tau_i(t)$ is the joint torque vector; M_{ij} is the symmetric inertia matrix; h_{ijk} are the centrifugal and Coriolis force coefficients; G_i is the gravity force vector; and q(t), q(t), q (t)are the joint position, velocity, acceleration vectors, respectively. The robot is required to move from the initial point P_i to a destination point P_f, both can be specified in Cartesian space or joint space coordinates. The task time is calculated as T and the torques to be exerted at the robot joints at each time step are $\tau_i(t)$. The cost function implemented in this study is a balance between the task time T and the quadratic average of actuator efforts $\tau_i(t)$ [14], i.e.

$$\min F_{obj} = uT + \frac{1-u}{2}\int_0^T \sum_{i=1}^{n}\left(\frac{\tau_i(t)}{\tau_i^{max}}\right)^2 dt$$

Where u is a weighing factor ranges from [0 to 1] subjected to the following constraints:

$$q(t=0) = q^{ini}, q(t=T) = q^{fin}$$

$$\dot{q}(t=0) = 0,\ \dot{q}(t=T) = 0$$

$$\ddot{q}(t=0) = 0, \ddot{q}(t=T) = 0$$

$$|q_i(t)| \le q_i^{max}, |\dot{q}_i(t)| \le \dot{q}_i^{max}$$

$$|\ddot{q}_i(t)| \le \ddot{q}_i^{max}, |\tau_i(t)| \le \tau_i^{max}$$

Calculations of τ_i are straight forward at each time step using these equations.

3.2 Calculations of Task Time:

To facilitate the stochastic optimization method the normalization of the time scale is introduced where the problem is solved for fixed final time [13], i.e. for q (x), $0 \le x \le 1$. Equation (1) becomes:

$$\tau_i(x) = \frac{1}{T^2}\left[\sum_j M_{ij}(q(x))\acute{q}_j(x) + \sum_j\sum_k h_{ijk}(\dot{q}(x))q_j(x)\dot{q}(x)\right] + G_i(q(x))$$

This process allows us to translate the geometric, kinematic, and dynamic constraints into bounds on admissible values of the optimal transfer time T_q of the parameter (x), for instance,

$$T_v = \max_{i=1,\dots n}\left[\max_{x\in[0,1]}\frac{[\dot{q}(x)]}{\dot{q}_i^{max}}\right]$$

$$T_a = \max_{i=1,\dots n}\left[\max_{x\in[0,1]}\frac{[\ddot{q}(x)]}{\ddot{q}_i^{max}}\right]$$

Where T_v and T_a are the minimum task time due to bounds of joints velocities and accelerations, respectively. $T_L$and T_R are the left and right time bounds due to torques and calculated based on the values of $\tau a_i(x)$, $\tau b_i(x)$, and $H_i(x)$given in Table 1, where

$$\tau a_i = -\tau_i^{max} - G_i(x)$$

$$\tau ba_i = \tau_i^{max} - G_i(x)$$

$$H_i(x) = \sum_j M_{ij}(q(x))\ddot{q}_j(x) + \sum_j\sum_k h_{ijk}(q(x))\dot{q}_j(x)\dot{q}_k(x)$$

$$\text{and } \acute{q} = \frac{\partial q}{\partial x}, \ddot{q} = \frac{\partial\partial q}{\partial x}$$

The minimum task time T_q of the selected trajectory q(x) should satisfy

$$Tq \ge Tv, \ge Ta, \text{ and } T_q \in [T_l, T_r]$$

Applying this method inside an optimization algorithm allows us to test different trajectory candidates and efficiently scan the solution space for the optimization objective.

3.3 Trajectory Planning:

The motion of a robot can be classified into motion along predefined paths like those in the seam welding or coating applications or free point to point motions such as those of pick and place applications. In both cases the trajectory describes the positions, velocities, and accelerations of either the end effector of a robot manipulator in the Cartesian space or the robot joints in the joint space for each time step during the task. The planned trajectory must satisfy the geometric, kinematic, and dynamic constraints. In free point to point motions the shape of the trajectory can be freely selected while meeting the constraints on the beginning and end of the task with other constraints on maximum velocities and accelerations. There are considerable research works in the area of robot trajectory planning and many planners have been suggested such as cubic trajectory, quintic trajectory, trapezoidal trajectory, smooth trapezoidal trajectory, or cubic spline etc. The equations governing the smoothed trapezoidal velocity profile (STVP) are given next as an example of the suggested trajectories. STVP will be used in the optimization algorithm and the numerical example discussed next; however, other trajectories can be satisfactorily applied to the proposed method.

$$q(x) = \begin{cases} q^{ini} + \frac{(q^{fin}-q^{ini})}{(1+x_b-x_a)}\left(\frac{2x^3}{x_a^2} - \frac{x^4}{x_a^3}\right) & 0 \le x \le x_a \\ q^{ini} + \frac{(q^{fin}-q^{ini})(2x-x_a)}{(1+x_b-x_a)} & x_a \le x \le x_b \\ q^{ini} + \frac{(q^{fin}-q^{ini})}{(1+x_b-x_a)}\left(\frac{(x-x_b)^4}{(1-x_b)^3} - 2\frac{(x-x_b)^3}{(1-x_b)^2} + (2x-x_b)\right) & x_b \le x \le 1 \end{cases}$$

The variables in this trajectory are x_a and x_b which define the shape and characteristics of the trajectory profile while x is the current time in the scaled period

4. Implementation:

The parameters for the elitist GA used in our example are listed in Table II. In this application a repair algorithm is developed to make sure that only valid chromosomes are generated after crossover, i.e. the constraints and bounds are fulfilled before valuating the chromosome. If an individual

violates any constraint, it is modified in a random fashion until it is no longer violating any constraints.

Table II. Genetic Algorithm Parameters

Population size	10
Chromosome length	10
Generations	100
Fitness selection	Rank
Crossover	0.7
Mutation probability	0.1

We applied the proposed genetic algorithm to our designed robot manipulator. The parameters that constitute robot kinematics are given in Table I. The motors' weights and inertias are neglected.

The proposed artificial neural network uses the optimized data generated by the genetic algorithm proposed in this paper and gives the optimal time, torques, velocities and accelerations for the given task. The neural network algorithm used is the conjugate gradient back propagation with Fletcher-Reeves updates. The training of the robot was done by a neural network whose structure is defined as follows: number of inputs: 8, number of inputs hidden layers: 3, number of inputs: 8, number of output layers: 3, number of neurons in hidden layers: (33, 23 and 9), activation function for the three layers: poslin, tansig, tansig, purelin. The proposed neural network was trained by a training set of 400 data while the testing was performed on 100 datasets. Explanation of the Inputs: The inputs in this neural network are the initial position and the final position of the four motor of the robot manipulator expressed in Cartesian space. The input positions are expressed in terms of radians where the first joint can move from an angle of 0 Degree to 180 degrees, second joint can move from an angle of 0 Degree to 120 degrees, third joint can move from an angle of 0 Degree to 120 degrees and fourth joint can move from an angle of 0 Degree to 180 degrees.

Explanation of the outputs: The number of outputs of this neural network is 3. The first output denotes the optimal time that will be needed by the robotic manipulator to move for an initial position to a final position. The other two ouput denote X_a and X_b, the points where the joints attain their maximum velocity for the longest possible period.

5. Results:

In our experiment, we moved the robotic manipulator from the initial position of [0 0 0 0] to [1.9 2 .9 1.8] and tracked down the time, velocity, torque and accelerations of the four joints which were needed to complete the task. In first case, when we used un-optimized results, the time taken by the robotic manipulator in completing this task was 2.12 seconds while in the second case, when we used the proposed genetic algorithm, the task time was reduced to 1.0046 seconds. Thus we found out that optimizing the task-time of the robotic manipulator had a significant effect on the parameters of the robotic manipulator which are used to drive the manipulator from one place to the other for a task.

Table IV shows the corresponding motion profile using STVP (position, velocity, and acceleration). In the first column, the graphs show the results which were generated without using genetic algorithms while in the second column, the graphs show the trend of the velocity, acceleration, torque and positions of the four joints with respect to the optimized time which is generated by the genetic algorithm.

The joint torques with respect to time is depicted in Table IV. It was also observed that time between x_a and x_b when time T is optimized is longer than un-optimized case. This allows the manipulator to move in its maximum velocity for the longest possible period which eventually leads to minimizing the task time. If the task is given in Cartesian space then the inverse kinematic problem has to be solved to convert to the joint space coordinates before starting the optimization loop.

If the task in Cartesian space coordinates includes a required orientation of the robot's end effector then this has also to be treated before conversion to the joint space coordinates using for example a rotation about a vector in space an angle of alpha. Then, the selected trajectory can be applied to alpha. In case the path is predefined through some via points, then these via points can be treated as constraints and other freely selected points can be adjusted to minimize the cost function in the same way.

After obtaining the optimized time through genetic algorithms, the proposed neural network was applied and the optimized time was used as the testing data for the neural network. Based on the optimal time period given by the neural network, the results generated by the applied genetic algorithm indicate that the accuracy of the neural network is approximately 97% as compared to the optimal results that has been produced by the genetic algorithms. The table V denotes the initial position and final position of the robot manipulator, the optimized time generated by the genetic algorithm, the time generated by the artificial neural network and the accuracy of the data generated by the artificial neural network as compared to the optimized data generated by genetic algorithm. Figure 1 shows the graph of the velocities, accelerations and the torques that have been generated by our proposed artificial neural network for a given task.

6. Conclusion and Future Works:

In this study a four DOF robotic manipulator has been optimized by a genetic algorithm and artificial neural network under kinematic and dynamic constraints. The required path along which robot moves can be described in joint space with respect to robot joints or in Cartesian space with respect to robot end effector. The smoothed trapezoidal velocity profile is given in this paper as an example of free motion trajectory between two points; however, other profiles of free motion or motion along via points can be applied to the optimization method as well. The result obtained from this method are the task time, velocity, accelerations and the torques that can be applied to each link of the robot manipulator to optimize the task. The method discussed in this paper can also be applied to optimize robot design parameters along with task time for a given task.

Table III. The result and variation of the Velocity, Acceleration and Torques after applying Genetic Algorithm on the Robotic Manipulator

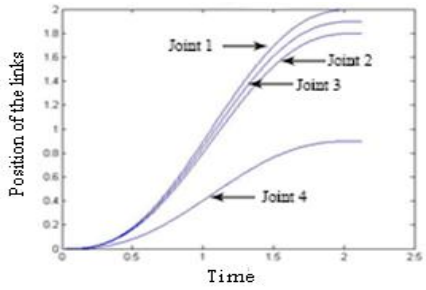

Fig 2.1. Position vs Time Graph

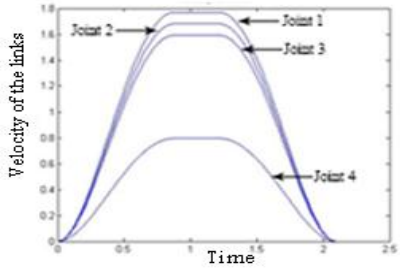

Fig 2.2. Velocity vs Time Graph

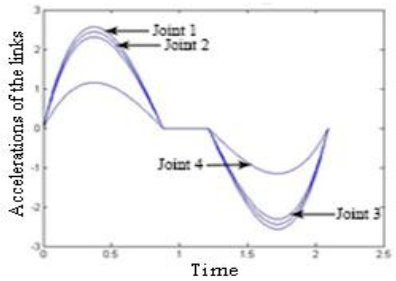

Fig 2.3. Acceleration vs Time Graph

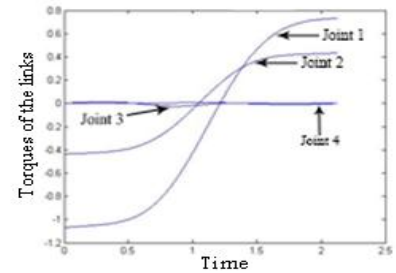

Fig 2.4. Torque vs Time Graph

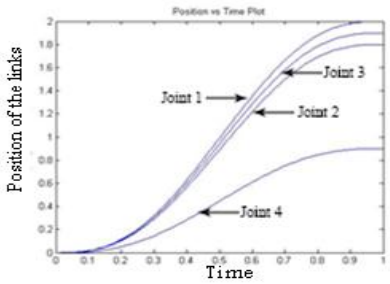

Fig 2.5. Position vs Time Graph

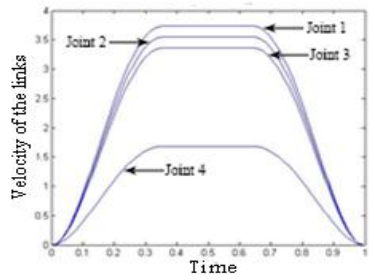

Fig 2.6. Velocity vs Time Graph

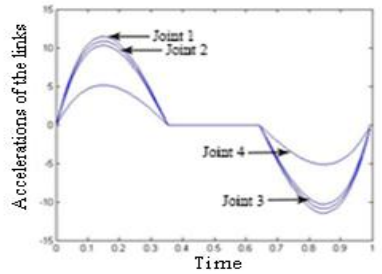

Fig 2.7. Acceleration vs Time Graph

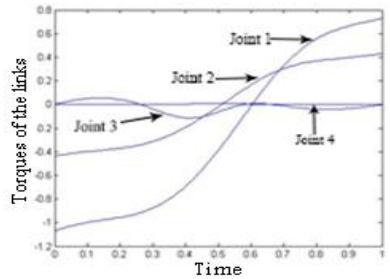

Fig 2.8. Torque vs Time Graph

Table IV. Comparison and Accuracy of the result generated by Artificial Neural Network and Genetic Algorithm

Serial No.	Initial Position of the four motors in Cartesian space	Final Position of the four motors in Cartesian space	Optimized time using GA(in seconds)	Task Time generated by proposed ANN (in seconds)	Percentage accuracy of result produced by ANN
1.	[0.0 0.0 0.0 0.0]	[2.7 2.0 1.9 2.1]	1.5013	1.5533	96.53%
2.	[0.0 0.0 0.0 0.0]	[1.5 1.5 1.5 1.5]	0.7058	0.7172	98.38%
3.	[2.0 1.5 1.6 1.7]	[0.0 0.0 0.0 1.2]	1.2789	1.2954	98.73%
4.	[0.0 0.0 0.0 0.0]	[0.5 1.0 1.5 2.0]	0.5210	0.5334	97.61%
5.	[1.0 1.0 1.0 1.0]	[2.0 2.0 2.0 2.0]	0.7979	0.7816	97.95%
6.	[0.8 0.5 0.2 0.1]	[2.5 2.0 1.5 2.1]	1.2073	1.1906	98.61%
7.	[1.1 1.1 0.0 2.1]	[2.0 2.0 0.5 2.0]	1.1589	1.1065	95.45%
8.	[1.5 1.5 1.5 1.5]	[2.5 2.0 2.0 2.5]	0.8196	0.8249	99.53%
9.	[0.5 0.5 0.5 0.5]	[2.0 2.1 2.2 2.3]	0.9867	0.9824	99.56%
10.	[0.2 0.3 0.4 0.5]	[1.8 1.9 2.0 2.1]	1.0945	1.0889	99.45%
11.	[2.0 1.5 1.5 2.0]	[0.0 0.0 0.0 0.0]	2.4789	2.4549	99.03%
12.	[2.0 1.5 1.5 2.0]	[1.00.5 1.1 1.2]	0.9871	0.9639	97.64%
13.	[2.1 1.9 1.8 1.7]	[0.5 0.6 0.7 0.8]	1.0781	1.0118	93.85%
14.	[2.1 1.9 1.6 1.5]	[0.1 0.2 0.3 0.4]	2.1970	2.2587	93.83%
15.	[0.3 0.4 0.5 0.6]	[2.0 2.1 2.1 2.5]	0.9458	0.9065	95.84%
16.	[0.00.5 1.1 1.8]	[2.0 0.0 0.0 3.1]	2.1426	2.0123	93.91%
17.	[1.0 0.8 0.7 0.6]	[1.2 1.9 2.1 3.1]	0.3158	0.2540	80.43%
18	[0.0 0.1 0.2 0.3]	[1.9 2.0 2.1 2.2]	0.9967	1.0656	93.08%
19.	[0.0 1.0 0.1 0.0]	[2.9 1.0 1.1 1.0]	1.5074	1.2626	83.76%
20.	[1.0 2.4 1.1 0.5]	[2.3 0.3 2.1 0.0]	1.0541	1.1330	92.51%
21	[2.9 1.5 0.1 0.5]	[0.3 0.3 2.1 2.9]	1.9987	2.0801	95.92%
22.	[0.9 1.0 1.1 1.5]	[2.3 1.8 2.1 2.9]	1.0050	0.9743	96.94%
23.	[0.5 0.6 0.7 0.8]	[2.7 1.5 1.6 1.7]	0.8995	0.9280	96.83%
24.	[1.1 1.9 1.7 1.6]	[2.4 0.5 1.7 0.1]	0.8004	0.7724	96.50%
25.	[0.7 0.6 0.5 0.4]	[1.9 2.0 1.7 2.0]	0.9471	0.9001	95.03%
26.	[0.1 0.2 0.3 0.4]	[1.8 1.9 2.0 2.1]	1.1247	1.0472	93.10%
27.	[2.1 1.9 0.5 1.6]	[0.0 0.5 1.9 0.1]	1.0159	1.0028	98.71%

28.	[1.1 1.2 1.9 2.0]	[0.0 2.1 2.0 0.5]	0.9136	0.8651	94.69%
29.	[0.7 0.8 2.0 2.1]	[1.5 1.5 1.5 1.5]	0.7784	0.7925	98.18%
30.	[0.1 0.8 0.9 1.4]	[1.4 1.8 0.0 1.5]	0.7851	0.7412	94.40%
31.	[2.0 2.0 2.0 2.0]	[0.5 0.5 0.5 0.5]	1.2914	1.0231	79.22%
32.	[1.4 1.9 1.7 1.6]	[2.1 1.0 1.5 0.0]	0.8176	0.8033	98.25%
33.	[2.0 1.5 1.5 2.0]	[0.4 0.9 1.0 0.1]	0.9612	0.9441	98.22%
34.	[0.3 0.4 0.5 0.6]	[2.1 2.0 1.9 1.8]	1.1584	1.0619	91.66%
35.	[1.5 1.4 1.6 1.1]	[0.0 1.9 2.1 0.4]	1.1008	1.0769	97.82%
36.	[2.4 1.7 1.6 1.5]	[0.1 0.2 0.3 0.4]	1.4007	1.4603	95.74%
37.	[0.4 0.6 0.8 1.0]	[2.4 1.9 1.4 2.1]	0.8984	0.8554	95.21%
38.	[0.7 0.8 1.5 1.6]	[2.1 1.5 1.7 2.9]	0.9983	0.9707	97.23%
39.	[2.1 2.0 1.9 1.8]	[0.3 0.2 1.0 0.1]	1.2971	1.2765	98.41%

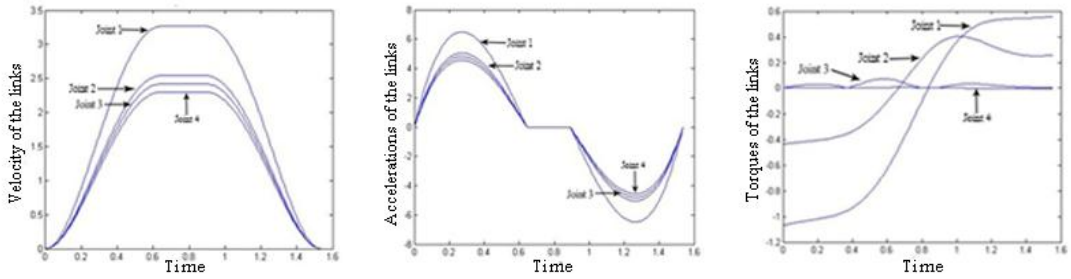

Figure1. The Velocity, Acceleration and Torques graphs after applying the neural network on the Robotic Manipulator for moving the manipulator from initial position of [0 0 0 0] to [2.7 2.0 1.9 2.1]

7. REFERENCES

[1] Angeles J. 2003. Fundamentals of Robotic Mechanical Systems, Theory, Methods, and Algorithms, 2nd ed., Springer-Verlag Inc.

[2] Kahn, E. and Roth, B. The near-minimum-time control of open-loop articulated kinematic chains. ASME Journal of Dynamic Systems, Measurements, and Control, 1971, 39(3) pp. 164-172.

[3] Bobrow, J.E. and Dubowsky, S., J. S. Gibson. Time-optimal control of robotic manipulators along specified paths, International Journal of Robotics Research, 1985, 4(3) pp. 3-17.

[4] Shin, K. G. and McKay, N. D. Minimum-Time Control of Robotic Manipulators with Geometric Path Constraints. IEEE Transactions on Automatic Control, 1985, 30(6): Pg. 531-541.

[5] Pfeiffer, F. and R. Johanni, A concept for manipulator trajectory planning, 1985, IEEE Journal of Robotics and Automation, 3(2), pp. 115-123.

[6] Shiller Z.. On singular time-optimal control along specified paths. IEEE Transactions on Robotics and Automation, 1994, 10(4), pp. 561-566.

[7] Shiller, Z.and Dubowsky, S. On computing the global time-optimal motions of robotic manipulators in the presence of obstacles. IEEE Transactions on Robotics and Automation, 1991, 7(6), pp. 785-797.

[8] Chen, Y. and Desrochers. A. A. 1989. Structure of minimum-time control law for robotic manipulators with constrained paths, In Proceedings of 1989 IEEE International Conference on Robotics and Automation, Scottsdale, AZ, pp. 971-976.

[9] Sontag, E.D. and Sussmann, H. J. 1985. Remarks on the time-optimal control of two-link manipulators. In Proceedings of 24th IEEE International Conference on Decision and Control, Ft. Lauderdale, F L, pp. 1646-1652.

[10] Sontag E.D. and Sussmann, H. J. 1986. Time-optimal control of manipulators, In: Proceedings of IEEE

International Conference on Robotics and Automation, San Francisco, CA, pp. 1692-1697.

[11] Fourquet, J. Y. 1993. Optimal control theory and complexity of the time optimal problem for rigid manipulators. In Proceedings of the 1993IEEE/YRSJ International Conference on Intelligent Robots and Systems, Yokohama, Japan.

[12] Siciliano, B., Scilavicco, L., Villani, L. and Oriolo, G. 2009. Robotics Modeling, Planning and Control, Springer-Verlag Inc.

[13] Haddad, M., Chettibi, T., Khalil, W.and Lehtihet, H. 2007. Trajectory Generation, in Robot Manipulators, Modeling, Performance, Analysis and Control, W. Khalil and E. Dombre, Eds, 1st ed., London: ISTE Ltd.

[14] Pettersson, M. and Olvander J. 2007. Adaptive complex method for efficient design optimization. In Proceedings of ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Las Vegas, Nevada, USA, vol. 6: 33rd Design Automation Conference, Parts A and B, pp. 265-272.

[15] Stryk, O. V. and Buliesch, R. Direct and indirect methods for trajectory optimization, Annals of Operations Research, 1993, 37(1), pp. 357-373.

[16] Bobrow, J. E., Martin, B. J., Sohi, G., Wang, E. C., Park, F. C. and Kim, J. Optimal robot motions for physical criteria, Journal of Robotic Systems, 2001, 18(12), pp. 785-795.

[17] Rana, A. S. and Zalazala A. M. S. Near time optimal collision free motion planning of robotic manipulators using an evolutionary algorithm, Robotica, 1996, 14, pp. 621-632.

[18] Hatem A. Al-Dois, Jha, A. K. and Mishra, R. B. Task-based design optimization of serial robot manipulators. Engineering Optimization. August, 2012.

[19] Deb, K. 2001. Multi-objective Objective Optimization using Evolutionary algorithms, Wiley and Sons Ltd (2001)

[20] Parker, J., Khoogar, K. A. R, and Goldberg, D.E. 1989. Inverse kinematics of redundant robots using genetic algorithms. In: Proceedings of IEEE International Conference on Robotics and Automation, vol. 1, pp. 271-276.

[21] Yun, W.M. and Xi, Y.G. Optimum motion planning in joint space for robots using genetic algorithms, Robotics and Autonomous Systems, 1996, 18(4), pp. 373-393.

[22] Hirakawa, A. R. and Kawamura, A. Trajectory generation for redundant manipulators under optimization of consumed electrical energy. In the 31st IAS Annual Meeting IAS 96, 1996, vol 3, pp. 1626-1632.

[23] McAvoy, B., Sangolola, B. and Szabad, Z. Optimal trajectory generation for redundant planar manipulator. In proceedings of IEEE International Conference on Systems, Man, and Cybernetics, 2000, Nashville, pp. 3241-3246.

[24] Tian, L. and Collins, C. Motion planning for redundant manipulators using a floating point genetic algorithm. Journal of Intelligent and Robotic Systems: Theory and Applications, 2003, 38(3-4), pp. 297-312.

[25] Minsky, M. and Papert S. 1969.Perceptrons. MIT Press.

[26] Werbos, P. Beyond regression: New tools for prediction and analysis in the behavioural sciences. 1974. PhD thesis, Harvard University, Cambridge, MA.

[27] Hopfield, J.J. 1982. Neural networks and physical systems with emergent collective computational properties. In proceedings of the National Academy of Sciences of the USA, 79:2554 – 2588.

[28] Kawato M, Uno Y, Isobe M, Suzuki R. Hierarchical neural network model for voluntary movement with application to robotics. IEEE Control Systems Magazine, 1988, 8, 8-16.

[29] Ciliz, M. K. and Isik, C. Trajectory following control of robotic manipulators using neural networks. In proceedings of the 5th IEEE International Symposium on Intelligent Control, Philadelphia, Pennsylvania, 1990, pp. 536-540.

[30] Šafarič, R., Jezernik, K. 1994. Trajectory tracking neural network controller for a robot mechanism and Lyapunov theory of stability. Conference Proceedings IROS, p. 626-633.

[31] Zhao, H., Ishida, T Trajectory tracking control of industrial robot manipulators using a neural network controller. IEEE International Conference on Systems, Man and Cybernetics, 2007, ISIC. 978-1-4244-0991-4: 2390 – 2395.

[32] Dubey, A.D., Mishra, R.B. Jha, A.K. Design and Path Planning of a Mobile Robot. International Journal pf Computer Science and technology (IJCST), 2012, 3(2), VER 1, 0976-8491, Pg. 73-77.

[33] Denavit, J. and Hartenberg, R.S. A Kinematic notation for lower-pair mechanisms based on matrices. ASME Journal of Applied Mechanisms, 1995, vol 77, pp. 215-221.