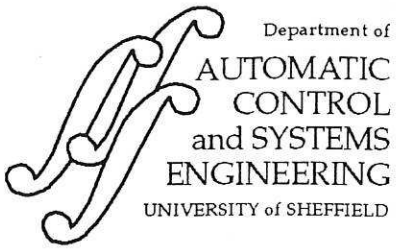


TOWARDS A COMPARATIVE STUDY OF NEURAL NETWORKS IN INVERSE MODEL LEARNING AND COMPENSATION APPLIED TO DYNAMIC ROBOT CONTROL

M. W. CHEN[1], A.M.S. ZALZALA[1] AND N.E. SHARKEY[2]

[1]*Robotics Research Group*
Department of Automatic Control and Systems Engineering

[2]*Department of Computer Science*

The University of Sheffield
Mappin Street, Sheffield S1 3JD, United Kingdom

Research Report #639
August 1996

Tel : +44 (0)114 2825250
Fax : +44 (0)114 2731729
EMail : rrg@sheffield.ac.uk
Robotics Research Group

Towards a Comparative Study of Neural Networks in Inverse Model Learning and Compensation Applied to Dynamic Robot Control

M W Chen[+], A M S Zalzala[+], and N E Sharkey[++]

+: Dept. of Automatic Control & Systems Engineering, Sheffield University.

++: Dept. of Computer Science, Sheffield University.

Abstract

This report deals with the applications of neural networks in inverse model learning and compensation to the mobile manipulator dynamic trajectory tracking and control. The mobile base is subject to a non-holonomic constraint and the base and onboard manipulator cause disturbances to each other. Compensational neural network controllers are proposed to learn to reach a sequence of targets with given times, to track dynamic trajectories under a non-holonomic constraint and torque limit constraint, and to compensate for uncertainties in the non-holonomic base and the manipulator and the disturbances between the base and the manipulator. Both multi-layered perceptron networks and radial basis function networks are considered in the report. Comparison was made between neural network controllers with and without model information. It is shown through various simulations the proposed neural network compensation schemes perform better than traditional controllers.

1. Introduction

A mobile manipulator is a combination of a robot arm and a vehicle with infinite working space in the horizontal plane. In the case of simultaneous locomotion and manipulation, the robot hand in principle will have the dynamics of the robot arm and the working space of the vehicle[1]. A typical task of a mobile manipulator includes the following two steps: first, move the vehicle in an unstructured environment towards a target either following a prescribed trajectory or going in a collision-free path, second, in the vicinity of the target position, move the vehicle and the onboard arm simultaneously so that the endeffector can follow a trajectory accurately to perform a task, for example doing welding or cutting.

Neural network based controllers have received much attention in recent years[2]-[7]. This type of controller exploits the possibilities of neural networks for learning non-linear functions as well as for solving certain type of problems where massive parallel computation is required. The learning capability of NNs is used to make the controller learn a certain nonlinear function, representing direct dynamics, inverse dynamics or any other mappings in a process.

This report deals with neural networks in inverse model learning and compensation to the motion control of a mobile manipulator in the following three aspects:

1. Sensor guided motion control of a vehicle in an unstructured environment.
2. Dynamic control of a mobile platform with a non-holonomic constraint.
3. Compensation control to the dynamic interactions between a vehicle and its onboard manipulator.

The control of mobile robots by means of sensory information is a fundamental problem in robotics. The autonomous, unsupervised control of a mobile robot in a novel environment is particularly challenging. Several researchers[8][9] have attacked some of the problems in mobile robot control using techniques from engineering and artificial intelligence. Problems that have been researched vary from sensor fusion to path planning and intelligent navigation. Here we focus on low-level control, that is, moving the vehicle to a single or a sequence of stationary targets by using simple movements. More specific description is that a target and the time to reach the target is given and the

wheels produce corresponding torques to move the vehicle the target in the given time. A neural network is applied to learn the relationship between the distance and angle difference to a target and the output torques of the vehicle. We will show an adaptive neural network architecture can carry out simple reaching movements robustly and efficiently.

Motion planning and control of mobile robots has been an active topic in robotics in the past several years[10][11]. Nevertheless, much less is known about the dynamic control of platform/manipulator system with a non-holonomic constraint on the platform and the developments in this area are very recent. Some researchers[12]-[14] have proposed some methods to the motion control of mobile manipulators. But they do not consider the non-holonomic constraint or are not robust enough to uncertainties or disturbances.

In many cases, a constrained robot is required to follow a trajectory which may not be followed accurately without violating the wheel torque limits and/or the non-holonomic constraint. But if the wheel input torques are compensated effectively, a good trajectory, which is very close to the desired one, can be generated. In some other cases, a mobile robot with a non-holonomic constraint is required to go through several positions at given times. The requirement is to minimise the position and orientation errors at these positions. In this report, a neural network compensator is proposed to learn the relationship between the given trajectory and the compensation torques, and then the compensation torques are added to the feedforward torques produced by the dynamic equations to control the motion of the mobile robot.

The kinetic laws of mechanics imply that the motion of any member of a multibody system - as is also the mobile manipulator - causes disturbant forces and torques to act upon other system members. erroneous displacements result from this influence, if the joint and the platform controllers fail to react appropriately. In this report, two separate neural network controllers are designed for the platform and the manipulator, respectively. Compensation forces are produced by these neural networks to cancel the disturbant forces caused by the other member of the system.

Various simulation results are given in this report to show the effectiveness of the neural network controllers.

2. Dynamic model of a mobile manipulator

A mobile platform with an onboard manipulator as shown in Figure 1 is considered in this report. The manipulator has one rotational link and two planar links. The platform has two driving wheels(the centre ones) and four passive ones(the corner ones). The two driving wheels are independently driven by two motors.

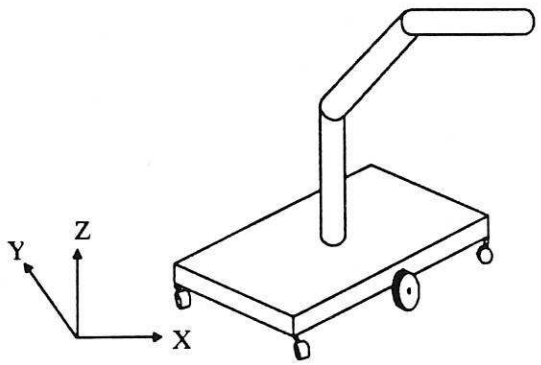


Figure 1 A mobile platform with an onboard manipulator

Considering the inertial reference frame in the (X, Y) plane and choosing a point P along the axis of the driving wheels on the mobile platform as shown in Figure 2, the mobile platform at point P can be described by three variables(x, y, θ_p), where (x, y) denotes the Cartesian position and θ_p describes the heading angle, respectively . For the manipulator, the joint angles of the three links are θ_1, θ_2 and θ_3.

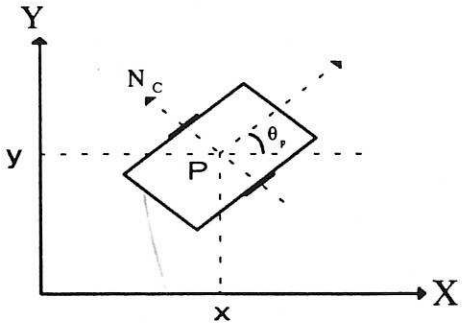


Figure 2 Top view of the mobile platform

Define a vector of coordinates

$$q = (q_1, q_2, q_3, q_4, q_5, q_6)^T \tag{1}$$

where q_1, q_2 and q_3 denote the platform position(x, y) and the heading angle θ_p, respectively, and q_4, q_5 and q_6 denote the manipulator joint angles θ_1, θ_2 and θ_3, respectively.

The platform is subject to the following non-holonomic constraints:

$$-\dot{x}\sin\theta_p + \dot{y}\cos\theta_p = 0 \tag{2}$$

i.e., the platform can not move in the direction normal to the axis of symmetry. Note that the position (x, y) and the heading angle θ_p of the platform are not independent of each other due to the non-holonomic constraint.

The non-holonomic constraint (1) can be rewritten as a matrix form as:

$$A\dot{q}_p = 0 \tag{3}$$

where $A = \begin{bmatrix} -\sin\theta_p & \cos\theta_p & 0 \end{bmatrix}$, $\dot{q}_p = \begin{bmatrix} \dot{x} & \dot{y} & \dot{\theta}_p \end{bmatrix}^T$.

The Newton-Euler method is used to derive dynamic equations of the mobile manipulator. In order to apply the iteration equations, the platform is visualised as having three joints, two prismatic joints(along x and y directions, respectively) and one rotational joint(along the vertical direction). The coordinate system is then built up based on the fictitious joints and the real manipulator joints. After the complete coordinate system is built up, the Newton-Euler iteration equations are applied to obtain the platform fictitious joint torques and the manipulator joint torques.

Considering first the mobile platform alone with no onboard manipulator, and referring to Figure 2, the dynamic equations of the platform written in Cartesian coordinates, after the inclusion of the forces due to the non-holonomic constraint, are

$$m_p\ddot{x} = \frac{\cos\theta_p}{r}(\tau_r + \tau_l) - N_c \sin\theta_p \tag{4}$$

$$m_p\ddot{y} = \frac{\sin\theta_p}{r}(\tau_r + \tau_l) + N_c \cos\theta_p \tag{5}$$

$$I\ddot{\theta}_p = \frac{R}{2r}(\tau_r - \tau_l) \tag{6}$$

where τ_r, τ_l are the external torque inputs acting on the right and left driving wheels; m_p is the mass of the platform; r is the radius of the wheels; R is the width of the platform; I is the rotational inertia of the platform; N_c is the centripetal force due to the non-holonomic constraint, as shown in Figure 2. Here, τ_r and τ_l are subject to lower and upper torque limit constraints:

$$\tau_{low} \le \tau_r, \tau_l \le \tau_{upp} \tag{7}$$

where τ_{low} and τ_{upp} are the lower and upper torque limits of the two wheels.

Differentiating (2), multiplying (4) and (5) by $-\sin\theta_p$ and $\cos\theta_p$ respectively, and adding, we obtain

$$N_C = m_p(\dot{x}\cos\theta_p + \dot{y}\sin\theta_p)\dot{\theta}_p \tag{8}$$

Rewriting Equations (4)-(6) with the constraint included, the following equation is obtained for the platform without an onboard manipulator:

$$\begin{bmatrix} m_p & 0 & 0 \\ 0 & m_p & 0 \\ 0 & 0 & I \end{bmatrix}\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{\theta}_p \end{bmatrix} + \begin{bmatrix} m_p \sin\theta_p(\dot{x}\cos\theta_p + \dot{y}\sin\theta_p)\dot{\theta}_p \\ -m_p \cos\theta_p(\dot{x}\cos\theta_p + \dot{y}\sin\theta_p)\dot{\theta}_p \\ 0 \end{bmatrix} = \begin{bmatrix} \frac{\cos\theta_p}{r} & \frac{\cos\theta_p}{r} \\ \frac{\sin\theta_p}{r} & \frac{\sin\theta_p}{r} \\ \frac{R}{r} & \frac{R}{r} \end{bmatrix}\begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix} \tag{9}$$

or in a concise form as:

$$M_p\ddot{q}_p + K(q_p,\dot{q}_p) = b\tau_p \tag{10}$$

where M_p is the platform inertia matrix, K is the term with the centripetal force and b is torque coefficient matrix. Note the b matrix is not square matrix. However the inertia matrix is found to be positive definite and symmetric.

From Equation(9), it can be seen that the platform has three output variables, but only has two driving torques. Furthermore, the three variables are constrained by the non-holonomic constraint and they are not independent on each other.

An important effect of the non-holonomic constraint is that an error in any of output variables x , y and θ_p requires to adjust both τ_r and τ_l and hence affects the other two outputs. Or in another way, a change in any of the two wheel input torques will affect all the 3 variables x , y and θ_p, while in a manipulator, this kind of coupling is much weaker. This strong coupling makes the control of a platform with a non-holonomic constraint more difficult than that of a manipulator.

For the case when the manipulator is on board the platform, the dynamic equations for the composite platform/manipulator system are obtained in a similar fashion. The dynamic equations for the platform X, Y directions are given by

$$m_p\ddot{x} = \frac{\cos\theta_p(\tau_r + \tau_l)}{r} - \cos\theta_p f_{hm} + \sin\theta_p f_{am} - N_c\sin\theta_p \tag{11}$$

$$m_p\ddot{y} = \frac{\sin\theta_p(\tau_r + \tau_l)}{r} - \sin\theta_p f_{hm} - \cos\theta_p f_{am} + N_c\cos\theta_p \tag{12}$$

where f_{hm}, f_{am} denotes the projections of the joint force from the first manipulator link onto the heading direction and axis direction of the platform, respectively.

Differentiating Equation(2), multiplying Equations(11) and (12) by $-\sin\theta_p$ and $\cos\theta_p$ respectively, and adding, we obtain

$$N_c = m_p(\dot{x}\cos\theta_p + \dot{y}\sin\theta_p)\dot{\theta}_p + f_{ma} \tag{13}$$

The composite dynamic equations of the mobile manipulator system can be expressed as

$$\overline{M}(q)\ddot{q} + \overline{K}(q,\dot{q}) = \overline{b}\tau \tag{14}$$

where

$$\overline{M}(q) = \begin{bmatrix} M_{mm} & M_{mp} \\ M_{pm} & M_{pp} \end{bmatrix},\ \overline{K}(q,\dot{q}) = \begin{bmatrix} K_m + K_{mp} \\ K_p + K_{pm} \end{bmatrix},\ \overline{b} = \begin{bmatrix} I_m & 0 \\ 0 & b_p \end{bmatrix},\ q = \begin{bmatrix} q_m \\ q_p \end{bmatrix},\ \tau = \begin{bmatrix} \tau_m \\ \tau_p \end{bmatrix};$$

$$q_m = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 \end{bmatrix}^T,\ q_p = \begin{bmatrix} x & y & \theta_p \end{bmatrix}^T;$$

$\tau_m = [\tau_1 \quad \tau_2 \quad \tau_3]^T$, $\tau_p = \begin{bmatrix} \tau_r \\ \tau_l \end{bmatrix}$;

I_m is an $m \times m$ identity matrix, m is the degree of freedom of the manipulator.

The disturbant couplings between the platform and the onboard manipulator are:

$$D_{pm} = M_{pm}\ddot{q}_m + K_{pm} \tag{15}$$

$$D_{mp} = M_{mp}\ddot{q}_p + K_{mp} \tag{16}$$

where the disturbant force D_{pm} from the manipulator to the platform consists of the inertial term $M_{pm}\ddot{q}_m$ and the term K_{pm} with centrifugal and Coriolis components in case of the manipulator motion; the disturbant force D_{mp} from the platform to the manipulator includes the inertia term $M_{mp}\ddot{q}_p$ and the term K_{mp} with centrifugal and Coriolis components in case of the platform motion. Precompensation of D_{pm} and D_{mp} leads to the consequence that the platform controller acts only on the platform only and the manipulator controller acts only on the manipulator only.

3. Neural networks for sensor-based motion control of a mobile robot

The control of mobile robots by means of sensory information is a fundamental problem in robotics. A typical example of sensor-based motion control is to guide the robot through a sequence of targets with some sort of sensing. For instance, the robot might be instructed to pick up a series of objects, or to navigate through a building or other environment by indicating a series of way points.

In this problem our work is based on a mobile robot equipped with a full set of sensing system. The robot has separated translational and rotational driving wheels and it is subject to the non-holonomic constraints mentioned above, which is that the robot can not move in the direction normal to the heading angle of the robot. The full sensing system can always give the current position and direction of the robot. Therefore, once a target position is given, the displacement between the current position and the target position, and the angle from the current position to the target can be obtained directly. A neural network is proposed to learn the mappings between the displacement and the corresponding forces and torques required by a mobile robot to reach the target in a given time. The neural network structure is given as in Figure 3.

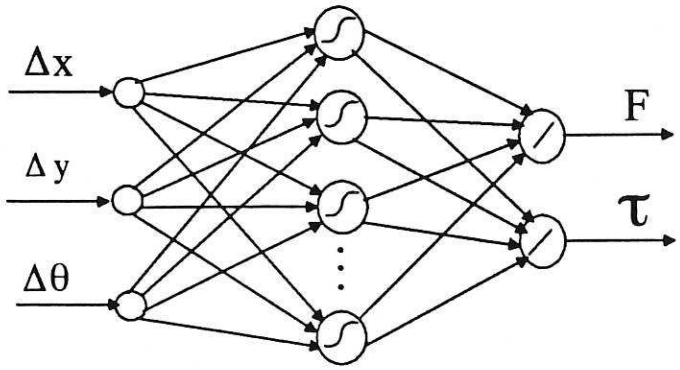


Figure 3 A multi-layered perceptron neural network to learn torque-to-displacement mappings

It is a multi-layer perceptron with one hidden layer. The number of inputs, number of nodes in the hidden layer and the number of outputs are 3, 5 and 2, respectively. The inputs to the neural network are the displacement of x, y and θ, namely Δx, Δy and $\Delta\theta$. The outputs are the translational force and rotational torque of the mobile robot.

The robot is trained in two ways. One is through simulation and the other is through real navigation

In the simulated training process, a simulated model is used to produce the motion of the robot. A large number of random targets and the given times to reach the targets are given to the neural network. The neural network is trained to learn the forces and the torques needed for the mobile robot to reach the targets in the given time. Because the desired output forces and torques of the robot are not known, an indirect training method is used as shown in Figure 4. First, The displacement inputs are put at the neural network input layer. Then the output torques from the neural network are executed on the simulated model of the robot. Next, the output displacement from the simulated model is compared with the given displacement. And finally the errors are sent to the neural network to update the weights of the network.

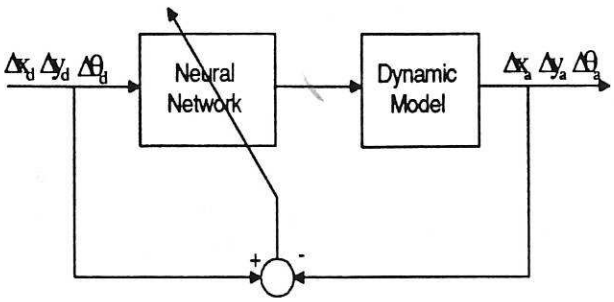


Figure 4 The neural network indirect learning structure

During the process of training the above neural network through the real robot in the real environment, first the translational force and rotational torque are put to the real robot, and then the corresponding motion is measured and recorded. The measured distances and rotating angles are paired with the given forces and torques to train the multi-layed perceptron network with the backpropagation learning paradigm. The forces and the torques are used as the desired outputs of the neural network and thus the network is trained in a direct way.

After the neural network is trained, it is applied to the simulated model and the real robot to reach arbitrary targets or a sequence of targets in a trajectory.

The above neural network can be easily trained to move the robot to reach targets without time constraint. The network outputs are changed to translational and rotational velocities, while the inputs are the same.

4. Dynamic trajectory following by neural network compensators

4.1 Structure of the neural network compensation control system

From Equation (3)-(5), we can obtain the following two equations:

$$\tau_r + \tau_l = m_p r(\ddot{x}\cos\theta_p + \ddot{y}\sin\theta_p) \tag{17}$$

$$\tau_r - \tau_l = \frac{2rI}{R}\ddot{\theta}_p \tag{18}$$

or

$$\tau_r = \frac{1}{2}(\frac{2rI}{R}\ddot{\theta}_p + m_p r(\ddot{x}\cos\theta_p + \ddot{y}\sin\theta_p)) \tag{19}$$

$$\tau_l = \frac{1}{2}(-\frac{2rI}{R}\ddot{\theta}_p + m_p r(\ddot{x}\cos\theta_p + \ddot{y}\sin\theta_p)) \tag{20}$$

From Equations(17)(18), it can be seen that $\tau_r - \tau_l$ directly affects $\ddot{\theta}_p$, and $\tau_r + \tau_l$ directly affects $\ddot{x}$, $\ddot{y}$ and $\ddot{\theta}_p$. And Equations (19)(20) show that the input variable affecting τ_r and τ_l are θ_p, $\ddot{x}$, $\ddot{y}$ and

$\ddot{\theta}_p$. Therefore it is not difficult to assume that in order to control the orientation of the platform, the difference between the right wheel toque and the left wheel toque is required to be adjusted, while to control the x and y direction motion, the sum of these two torques needs to be changed first. Based on this assumption, the outputs in the neural networks proposed for platform motion control are chosen as $\tau_r + \tau_l$ and $\tau_r - \tau_l$. The inputs to the neural network are chosen as θ_p, $\ddot{x}$, $\ddot{y}$ and $\ddot{\theta}_p$. For generation of compensation torques $\Delta\tau_r$ and $\Delta\tau_l$, the network outputs are chosen as $\Delta\tau_r + \Delta\tau_l$ and $\Delta\tau_r - \Delta\tau_l$. After the neural network output layer, an additional layer is used to obtain the torque τ_r and τ_l from $\tau_r + \tau_l$ and $\tau_r - \tau_l$ or to obtain $\Delta\tau_r$ and $\Delta\tau_l$ from $\Delta\tau_r + \Delta\tau_l$ and $\Delta\tau_r - \Delta\tau_l$.

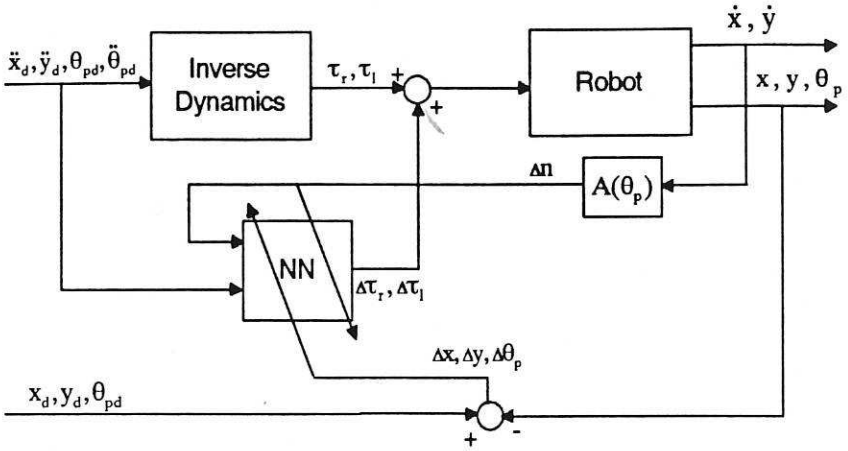


Figure 5 Structure of the neural network control for the platform

Based on the above analysis, a neural network compensation control system is proposed in this section as shown in Figure 5. The inputs of the control systems are the desired x, y and θ_p, and their accelerations. The inverse dynamic equations are applied as feedforward to produce the computed wheel torques and the inputs to the inverse dynamics and the neural network are θ_p, $\ddot{x}$, $\ddot{y}$ and $\ddot{\theta}_p$. The neural network is used as a compensator to generate the compensation torques. When the given trajectory is feasible and there are no model errors and disturbances, the robot will follow the desired trajectory accurately. When the given trajectory is not feasible or there is tracking errors, the neural network starts to learn on-line and produce compensation torques to reduce the tracking errors. The second function of the neural network is that when the robot slips, i.e. the non-holonomic constraint is violated and a corresponding constraint error($\Delta n = A\dot{q}$) occurs, then this error is fedback to the neural network for training and producing corresponding compensation torques to reduce the slippage and the tracking errors.

Two types of neural networks are considered in this report. They are multi-layered perceptron networks and radial basis functions networks.

4.2 Multi-Layered Perceptron neural network compensator

The multi-layered perceptron(MLP) neural network compensator is a four-layer perceptron with two hidden layers, four input nodes and two output nodes. The four inputs are θ_p, $\ddot{x}$, $\ddot{y}$ and $\ddot{\theta}_p$. The two outputs are $\Delta\tau_r + \Delta\tau_l$ and $\Delta\tau_r - \Delta\tau_l$. The errors Δx, Δy, $\Delta\theta_p$ and Δn are applied to update the weights of the network. The network is trained using well-known back propagation method.

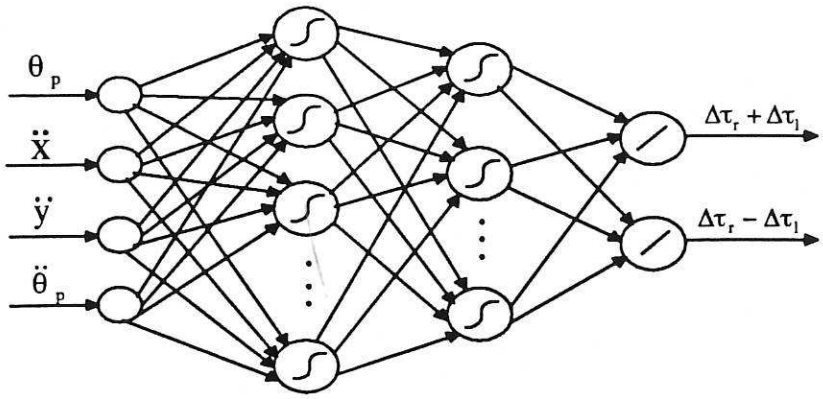


Figure 6 The MLP structure for platform compensation control

The structure of the neural network is given in Figure 6, where bias connections are not shown. The neural network is trained to learn the relationship between $\Delta\tau_r + \Delta\tau_l$ and the inputs and the relationship between $\Delta\tau_r - \Delta\tau_l$ and the inputs. The wheel compensation torques $\Delta\tau_r$ and $\Delta\tau_l$ are then indirectly calculated from the neural network outputs $\Delta\tau_r + \Delta\tau_l$ and $\Delta\tau_r - \Delta\tau_l$.

The trajectory tracking error is defined as:

$$e = \sqrt{(x_d - x)^2 + (y_d - y)^2 + (\theta_{pd} - \theta_p)^2}\,; \tag{21}$$

As the desired outputs of the network are not available, the errors used for updating the weights are chosen according to the inverse dynamic equations of the platform. Here, the errors used to update the weights connecting to the two network outputs $\Delta\tau_r + \Delta\tau_l$ and $\Delta\tau_r - \Delta\tau_l$ are chosen respectively as:

$$\delta_1 = K_x(x_d - x) + K_y(y_d - y) \tag{22}$$

$$\delta_2 = K_{\theta_p}(\theta_{pd} - \theta_p) \tag{23}$$

The weights are updated according to the standard back-propagation algorithm.

4.3 Radial Basis Functions neural network compensator

In this subsection, a Gaussian radial basis functions(RBF) neural network is applied to develop the wheel compensation torques. The RBF is a standard one hidden layer network with fixed centres. Figure 7 shows the structure of the RBF. The errors used for updating weights are similar to that in MLP.

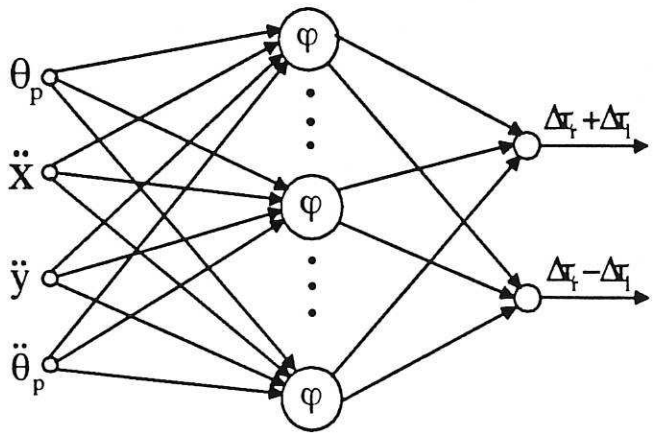


Figure 7 The RBF structure for the platform compensation control

5. Compensation control to the dynamic interactions between the platform and the onboard manipulator

5.1 Dynamic control of a manipulator by using a neural network

The neural network compensation control for an n-link manipulator is proposed as shown in Figure 8. The NN model is used to model the inverse dynamics of each joint, for nonlinear compensation of the manipulator. Control input torque to each joint consists of the compensation torque from the neural network and the feedback torque from the PD control. The neural networks are the standard MLP trained by back-propagation algorithm.

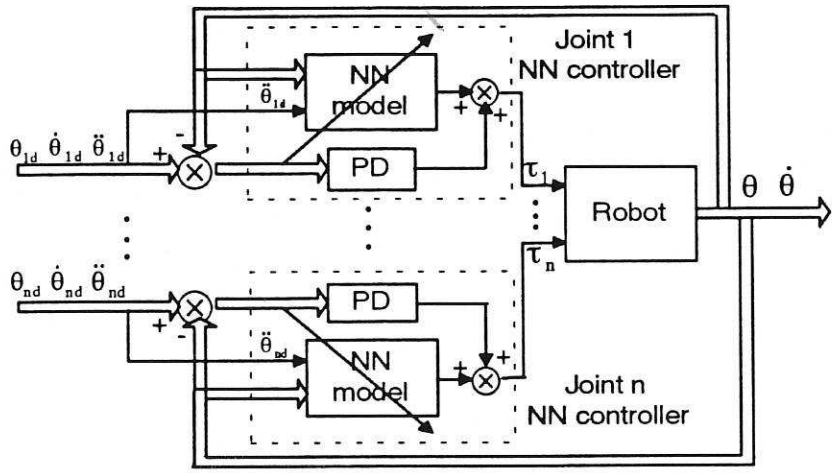


Figure 8 Structure of the neural network control scheme for an n-link manipulator

5.2 Compensation control to the dynamic interactions between the platform and the onboard manipulator

A compensation control structure is proposed as shown in Figure 9. The system includes two sub-control systems, one for the manipulator, one for the platform. The manipulator and the platform are coupled due to the disturbant forces. The coupling disturbant forces are compensated by the neural network controller in each sub-control system.

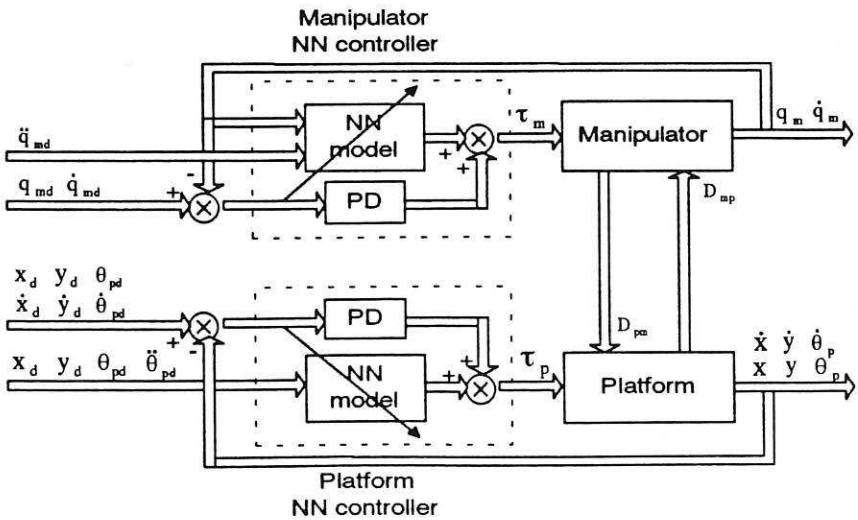


Figure 9 Structure of the decoupling control to a platform/manipulator system

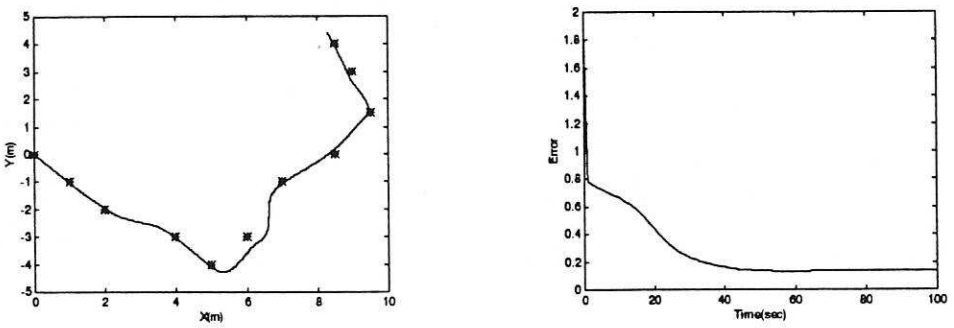
6. Simulations

6.1 Reaching a sequence of targets by a mobile robot(B12)

In this simulation, a mobile robot(B12) is asked to go through a sequence of targets in the given times. The robot stops once a target has been reached and then start to move to the next target. The time between any two consecutive stops is given as 0.5 seconds. Two kinds of simulations are carried out. The first one is by neural network learning without any model information. The other one is by neural network learning with model information. The neural networks are multi-layered perceptrons with only one hidden-layer.

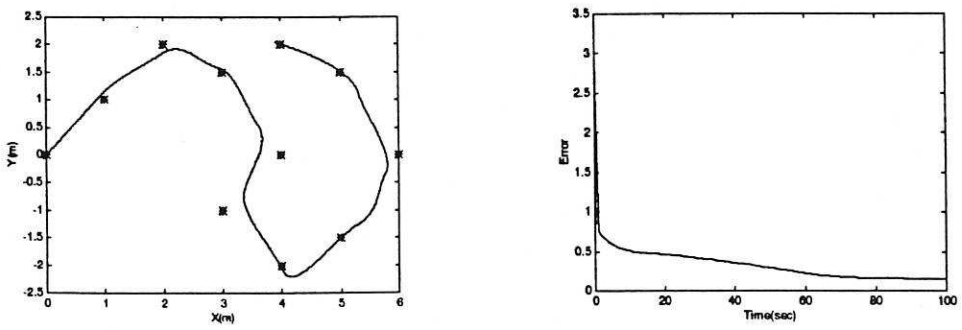
6.1.1 Neural network learning without any model information

Figure 10 and Figure 11 shown the simulation results of two examples. Without any model information, there are large tracking errors at some targets. The right figures shown the error norms during the neural network learning processes. The final error norms of example 1 and example 2 are 0.14 and 0.16, respectively.



(a) (b)

Figure 10 Simulation results of example 1 without any model information (a) the actual path (b) the error norms during the NN learning process



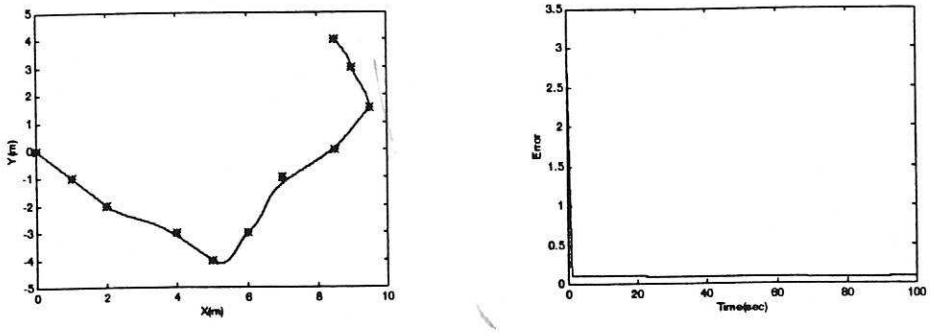
(a) (b)

Figure 11 Simulation results of example 2 without any model information (a) the actual path (b) the error norms during the NN learning process

6.1.2 Neural network learning with model information

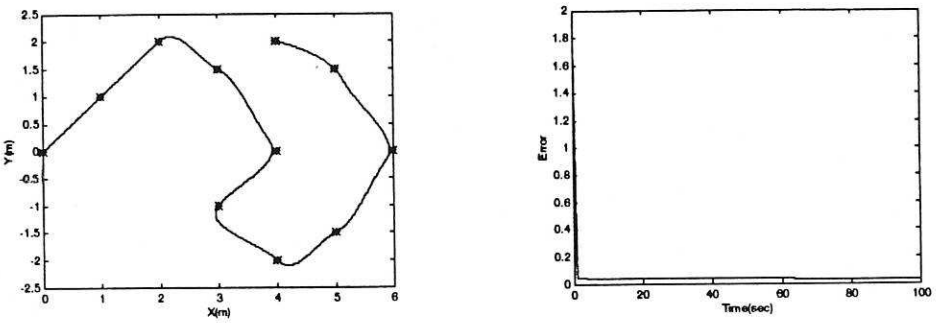
Figure 12 and Figure 13 shown the simulation results of two same examples as above. With model information, the tracking errors are much smaller. And also the neural networks learn much faster.

The right figures shown the error norms during the neural network learning processes. The final error norms of the above two examples are 0.05 and 0.04, respectively.



(a) (b)

Figure 12 Simulation results of example 1 with model information (a) the actual path (b) the error norms during the NN learning process



(a) (b)

Figure 13 Simulation results of example 2 with model information (a) the actual path (b) the error norms during the NN learning process

6.1.3 Simulations by SIMULINK neural networks

Two simulation models as shown in the Figure 14 and Figure 15 are built by using MATLAB SIMULINK to simulate the motion of the robot. The models include two parts--the neural network and the dynamic model of the mobile robot(B12). The neural networks are trained beforehand and their weights are imported from the saved weight matrix.

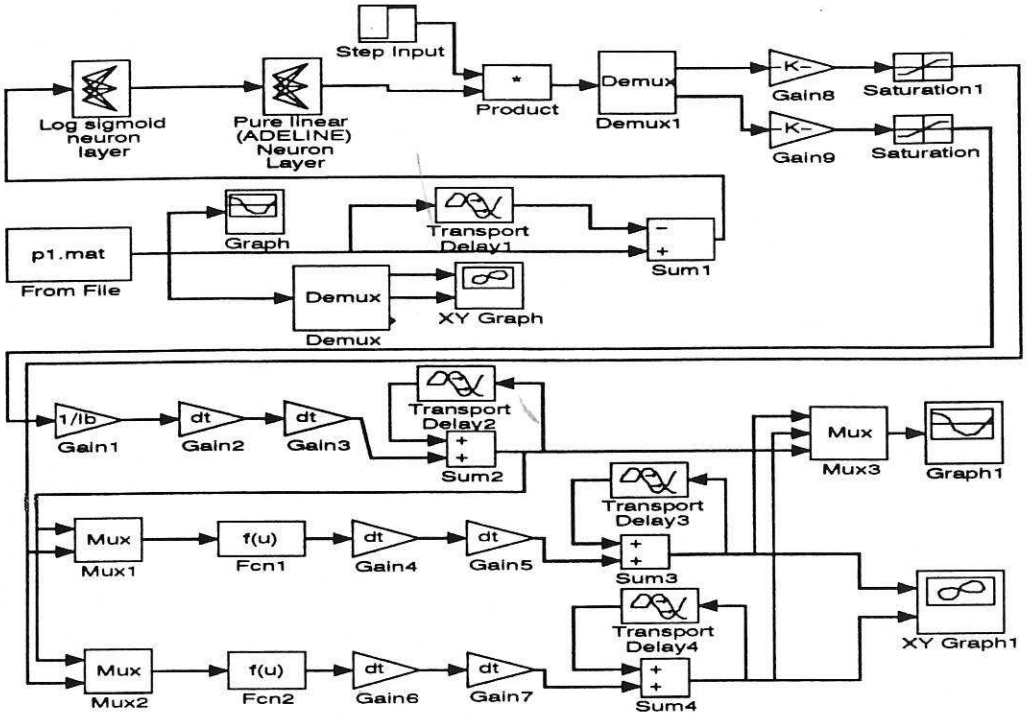


Figure 14 The SIMULINK model for reaching arbitrary targets by a NN without model information

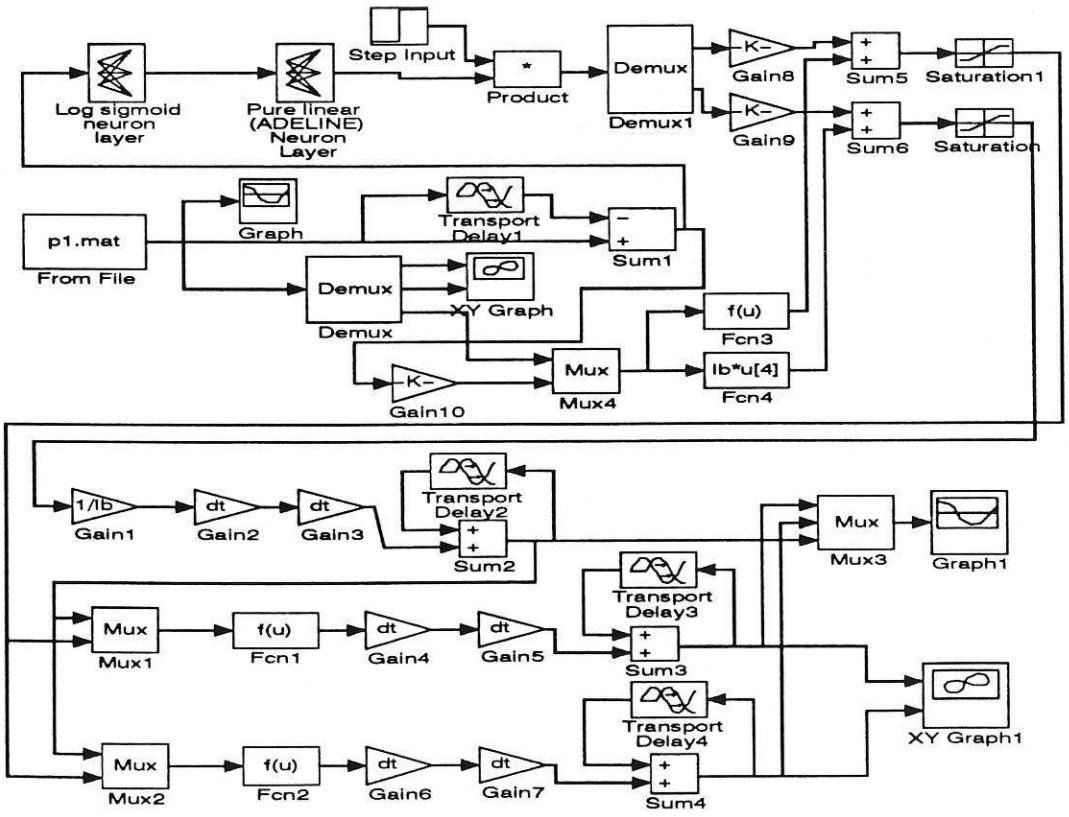
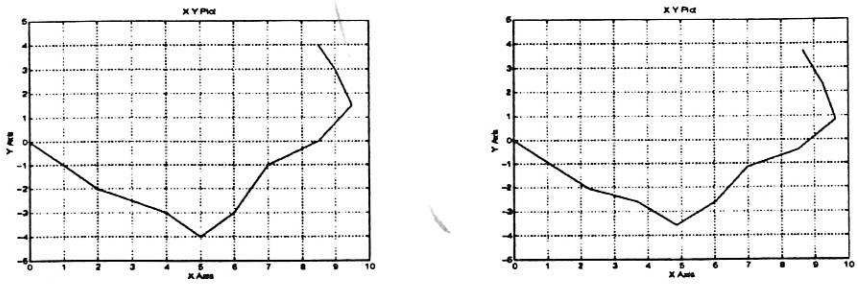


Figure 15 The SIMULINK model for reaching arbitrary targets by a NN with model information

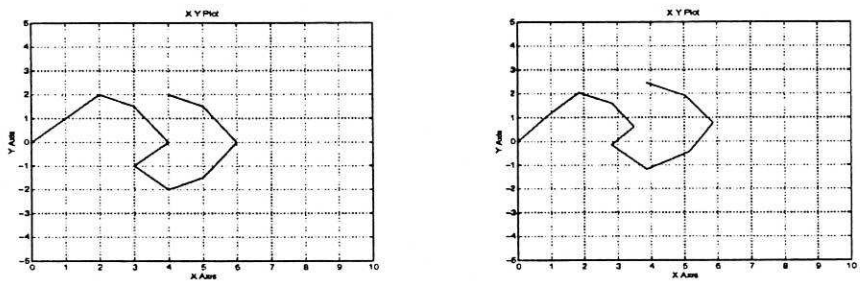
Case 1 SIMULINK simulation results without model information

The following two figures(Figure 16 and Figure 17) shown that by the neural network control without any model information the robot has large tracking errors at some target positions.



(a) (b)

Figure 16 Simulation results of example 1 by SIMULINK without model information (a) the given path connecting a sequence of targets (b) the actual path

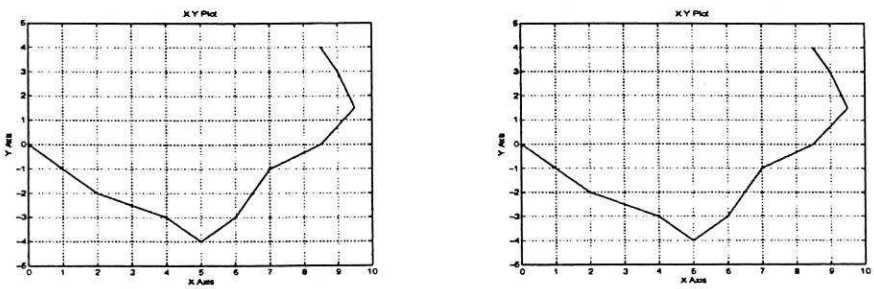


(a) (b)

Figure 17 Simulation results of example 2 by SIMULINK without model information (a) the given path connecting a sequence of targets (b) the actual path

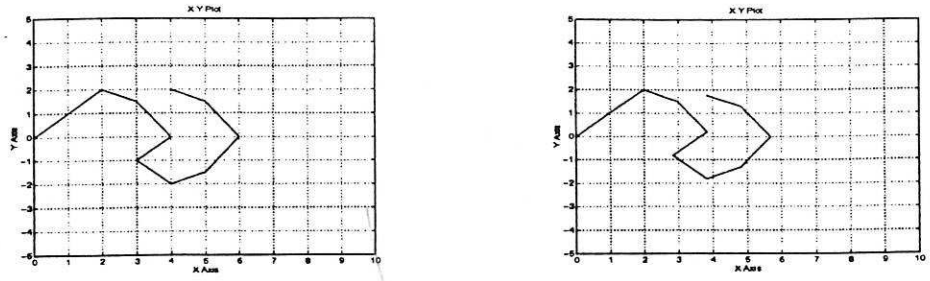
Case 2 SIMULINK simulation results by neural network with model information

The following two figures shown that with model information a neural network controller has better tracking results than without any model information.



(a) (b)

Figure 18 Simulation results of example 1 by SIMULINK with model information (a) the given path connecting a sequence of targets (b) the actual path



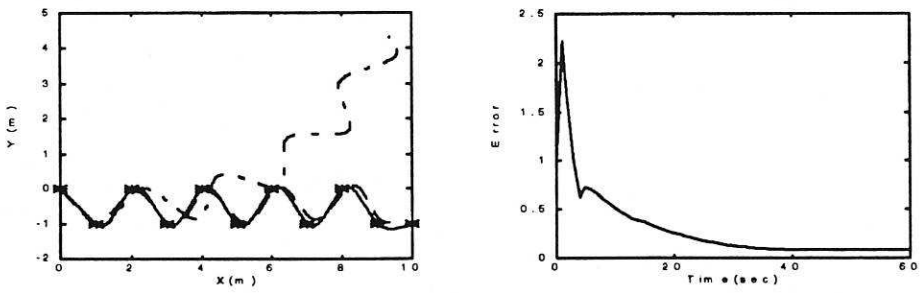
(a) (b)

Figure 19 Simulation results of example 2 by SIMULINK with model information (a) the given path connecting a sequence of targets (b) the actual path

6.2 Dynamic trajectory following by a constrained platform

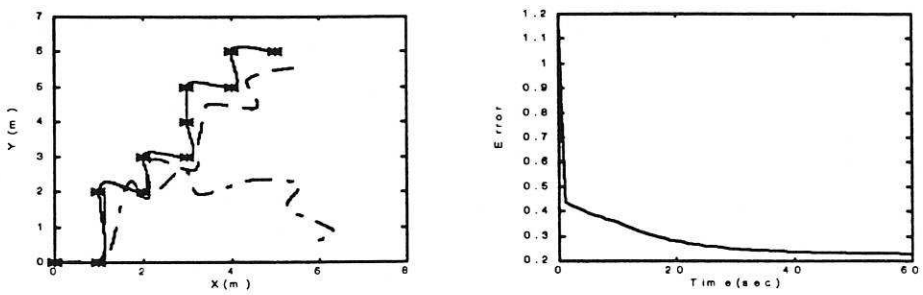
Case 1

A number of points are given and the robot is required to go through these points at specified time. The criterion is to minimise the position and orientation errors at these points. The trajectories between these points are created by splines. The compensation torques by generated by the NN are added to the torques computed from the inverse dynamics to control the motion of the platform. Figure 20 and 21 show the simulation results of two examples of such trajectory tracking by MLP compensation. Figure 22 and Figure 23 show the simulation results of the two examples by RBF compensation.



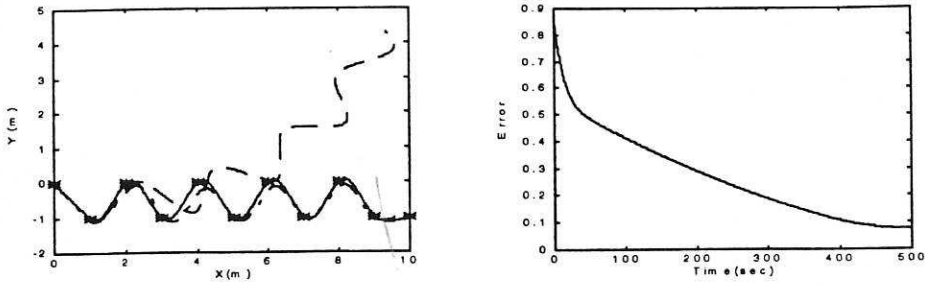
(a) (b)

Figure 20 (a) the desired trajectory(solid, *: given positions), the trajectory by computed torques(dashed/dotted) and the trajectory by MLP plus feedforward control(dashed) (b) error norms during the MLP learning process.



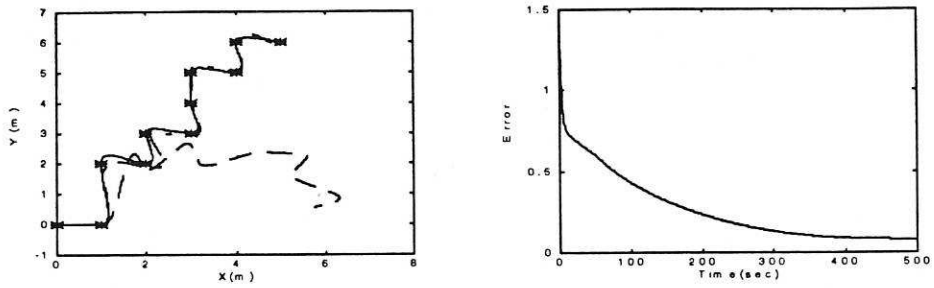
(a) (b)

Figure 21 (a) the desired trajectory(solid, *: given positions), the trajectory by computed torques(dashed/dotted) and the trajectory by MLP plus feedforward control(dashed) (b) error norms during the MLP learning process.



(a) (b)

Figure 22 (a) the desired trajectory(solid, *: given positions), the trajectory by computed torques(dashed/dotted) and the trajectory by RBF plus feedforward control(dashed) (b) error norms during the MLP learning process.



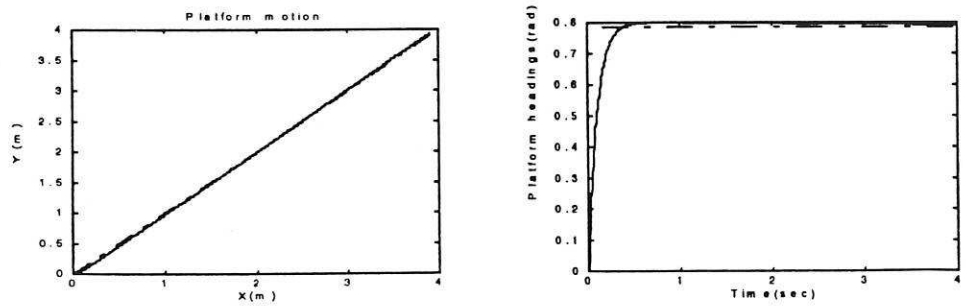
(a) (b)

Figure 23 (a) the desired trajectory(solid, *: given positions), the trajectory by computed torques(dashed/dotted) and the trajectory by RBF plus feedforward control(dashed) (b) error norms during the MLP learning process.

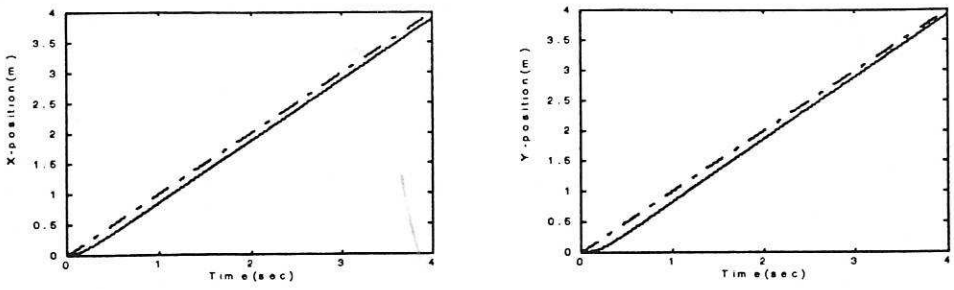
Case 2

A continuous trajectory is given with 30% of platform weight error. The task for the neural network compensator is to learn the relationship between the trajectory inputs and the compensation torques. The compensation torques are added to the computed torques to control the robot to follow the given trajectory as closely as possible. Figure 24 shows the tracking results by a PD control only.

The tracking process by an MLP compensation is shown in Figure 25, which is better than by PD control only.

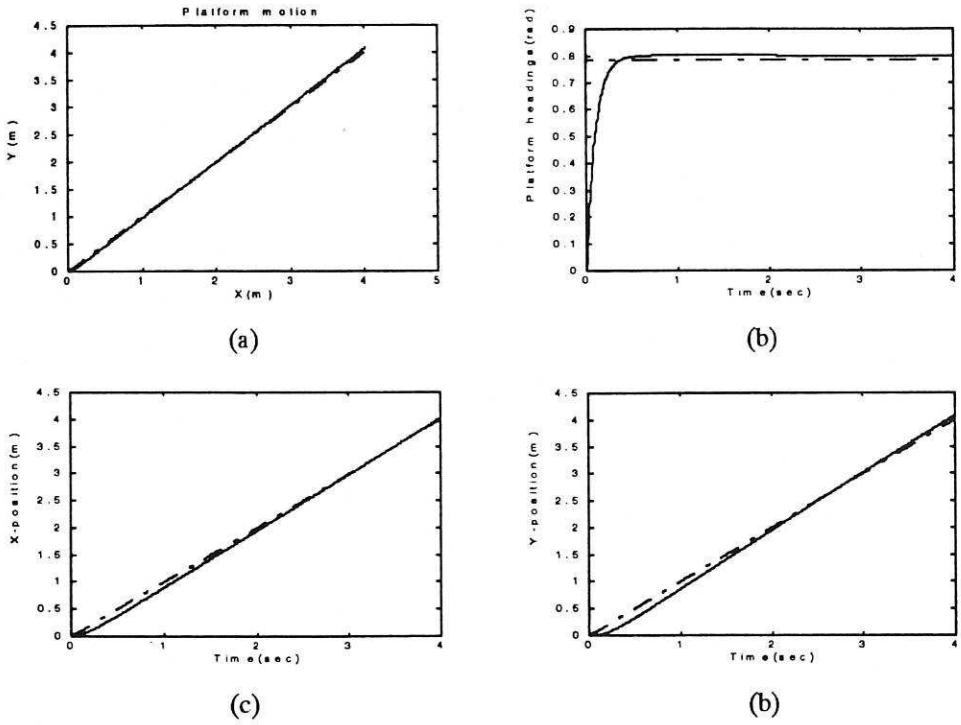


(a) (b)



(c) (d)

Figure 24 Trajectory tracking by PD controller only with 30% platform weight error(desired: dashed/dotted, actual: solid)



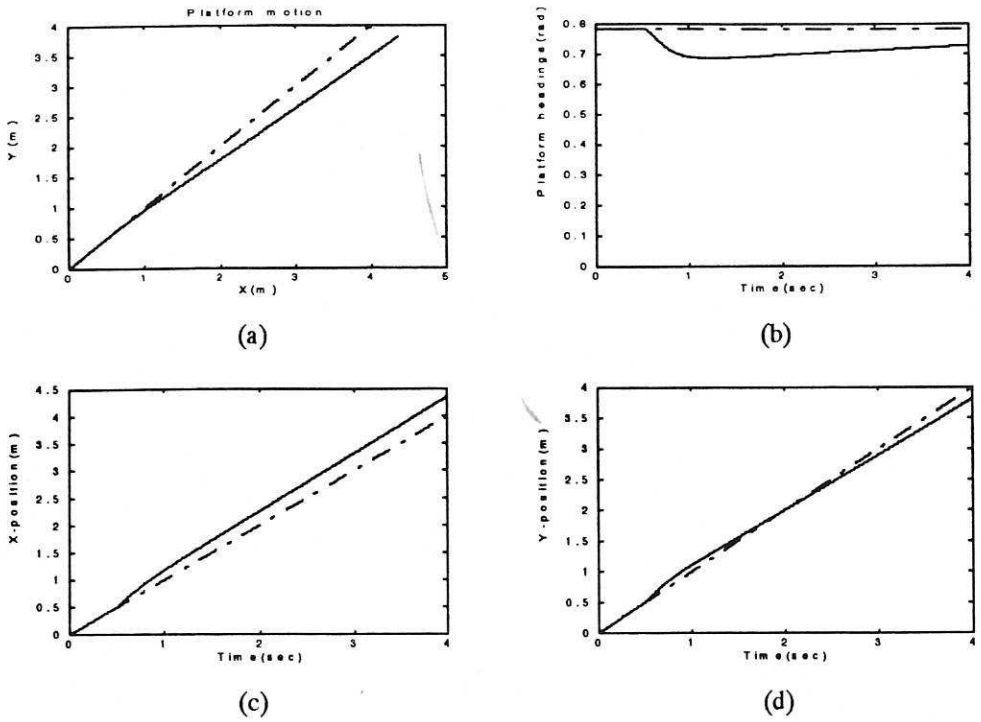
(a) (b)

(c) (b)

Figure 25 Trajectory tracking by MLP compensation with 30% platform weight error(desired: dashed/dotted, actual: solid)

Case 3

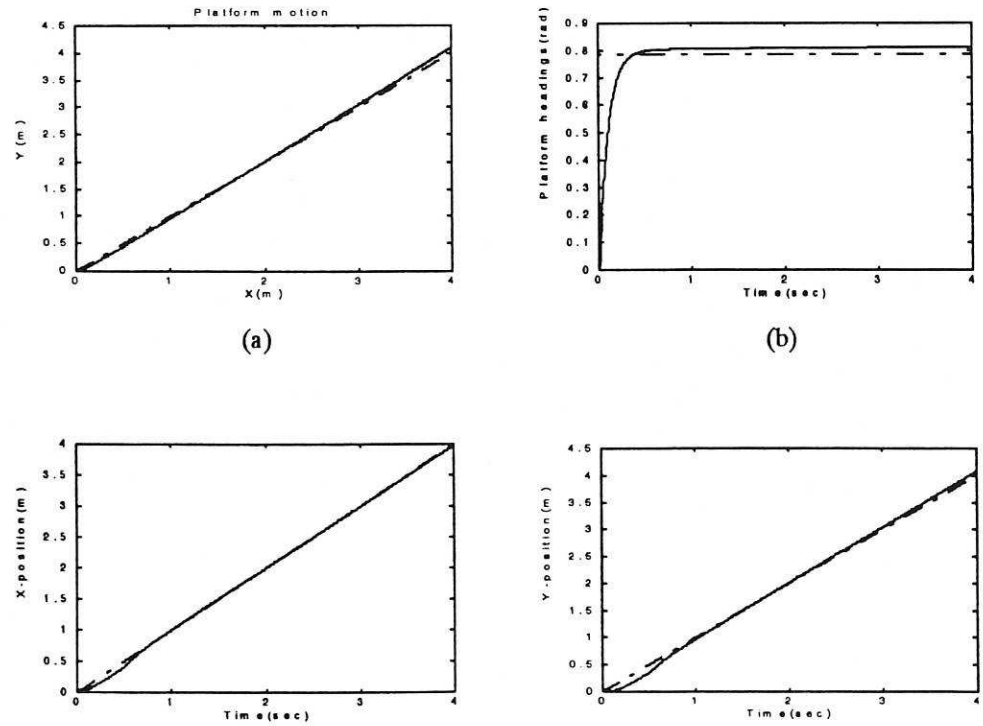
In this case, it is assumed that a slippage happened between 0.5s and 0.6s from the starting. During this period the non-holonomic constraint is violated. A constraint error is fedback to the neural network to speed the learning process. Figure 26 and 27 shows the results by the feedforward plus PD control and the results by a neural network plus PD control. As the results shown, feedforward plus PD control failed to control the platform back to the desired trajectory, while the neural network controller successfully bring the platform back to the desired trajectory very quickly



(a) (b)

(c) (d)

Figure 26 Trajectory tracking by feedforward plus PD control with a slippage between the time 0.5s and 0.6s from the starting(desired: dashed/dotted, actual: solid)



(a) (b)

(c) (d)

Figure 27 Trajectory tracking by MLP plus PD control with a slippage between the time 0.5s and 0.6s from the starting(desired: dashed/dotted, actual: solid)

6.3 Dynamic control of a 2-link manipulator

In this simulation, a 2 link manipulator is required to follow a given trajectory with the joints' trajectories specified. A 25% weight error of the second link is considered. The feedforward plus PD control failed to follow the desired trajectory as shown in Figure 28, while the neural network controller compensated the weight change very well as shown in Figure 29.

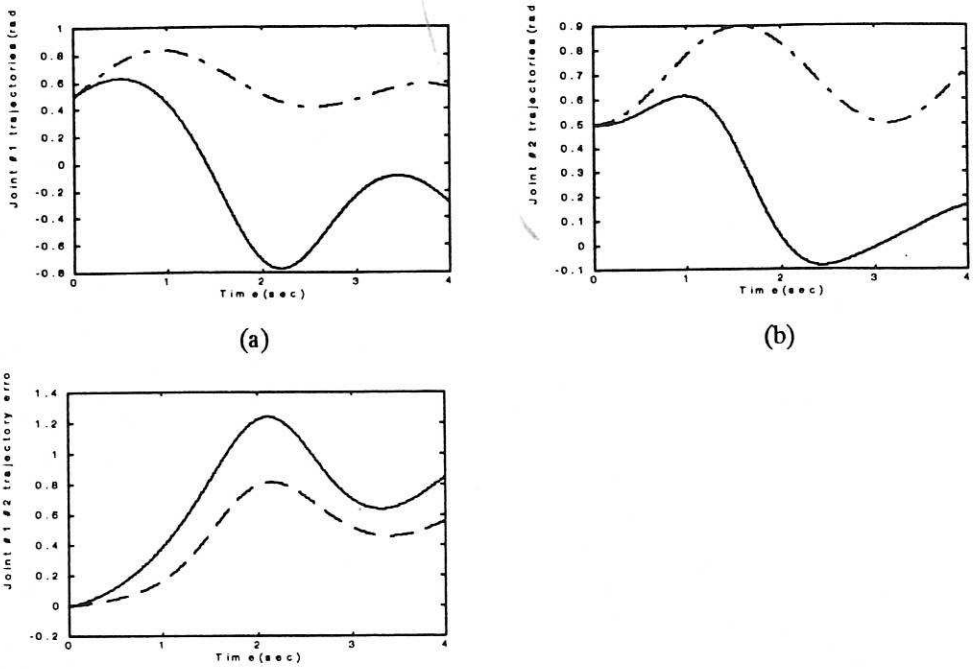
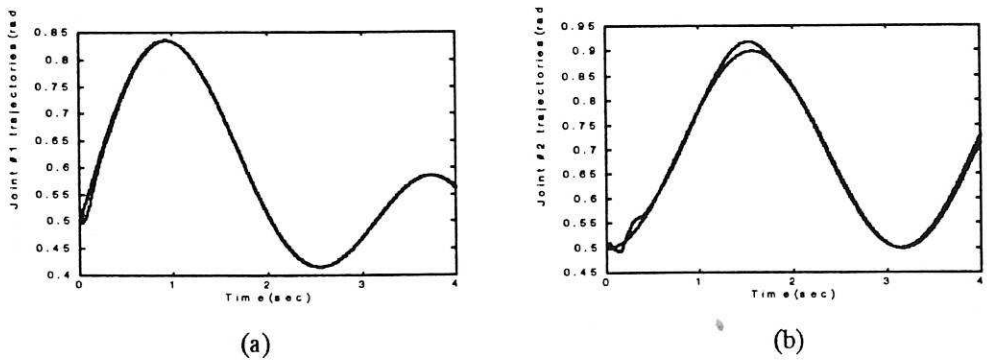
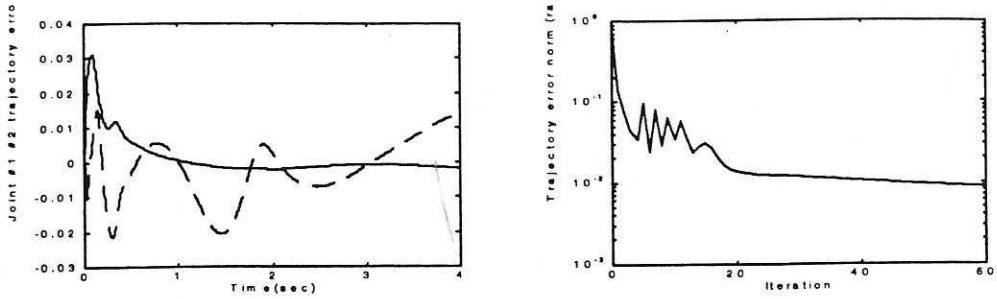


Figure 28 Manipulator trajectory tracking by feedforward plus PD control with a 25% weight error in the second link (a) trajectories of the first joint(desired: dashed/dotted, actual: solid) (b) trajectories of the second joint(desired: dashed/dotted, actual: solid) (c) tracking errors of the two joints(first joint: solid, second joint: dashed)





(c) (d)

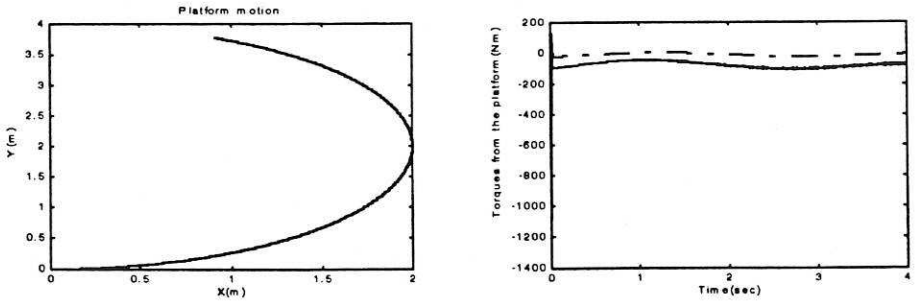
Figure 29 Manipulator trajectory tracking by MLP plus PD control with a 25% weight error in the second link (a) trajectories of the first joint(desired: dashed/dotted, actual: solid) (b) trajectories of the second joint(desired: dashed/dotted, actual: solid) (c) tracking errors of the two joints(first joint: solid, second joint: dashed) (d) error in the MLP learning process

6.4 Decoupling control of a platform/manipulator system

This simulation considers the neural network compensations against the disturbant forces from the manipulator to platform and from the platform to the manipulator.

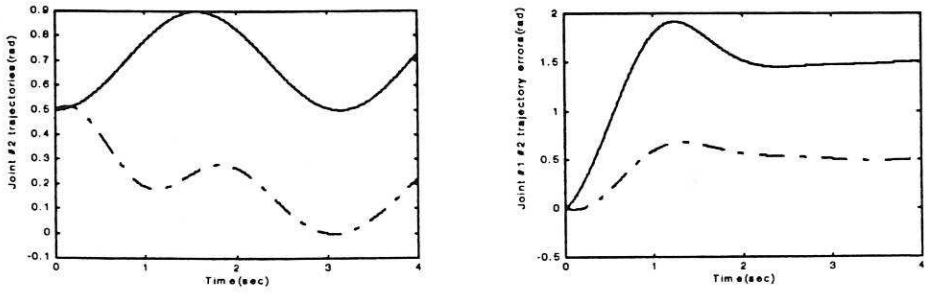
Case 1: Manipulator neural network compensation to the disturbance from the platform

Figure 30 shows the motion of the platform and its disturbant forces to the manipulator. Figure 31 is the simulation results from a PD controller only which does not work in this case. The results by the neural network compensation is shown in Figure 32. The MLP can compensate the disturbance quite well and the learning process converges very fast only with slight vibration in the beginning.

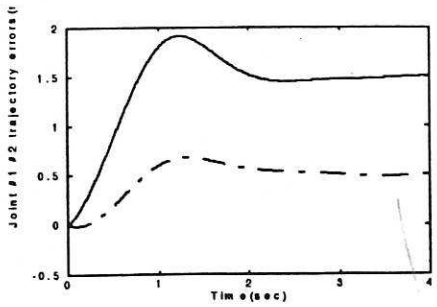


(a) (b)

Figure 30 (a) The given platform motion and (b) its disturbant forces to the manipulator

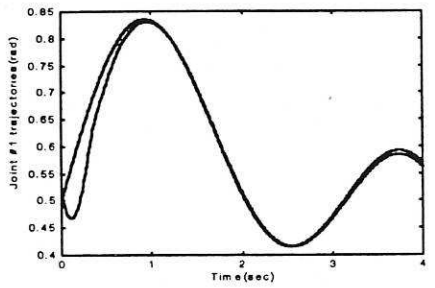


(a) (b)

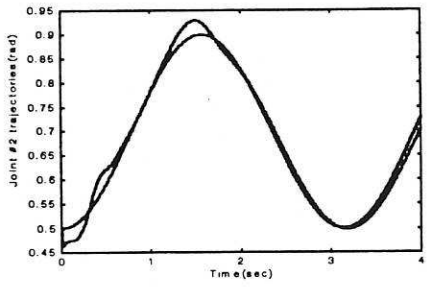


(c)

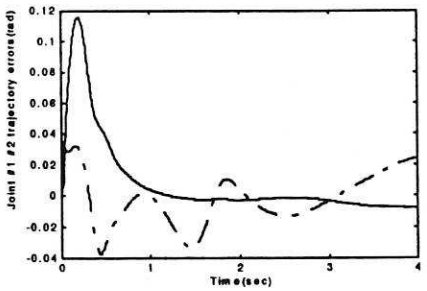
Figure 31 Manipulator trajectories and their errors by a PD controller in the case of platform motion (a) trajectories of the first joint(desired: dashed/dotted, actual: solid) (b) trajectories of the second joint(desired: dashed/dotted, actual: solid) (c) trajectories errors(first joint: solid, second joint: dashed/dotted)



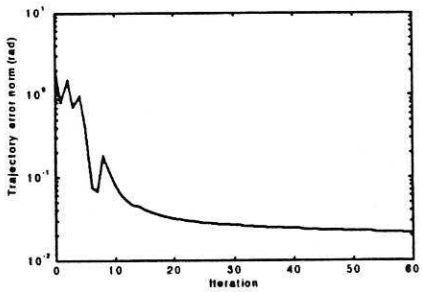
(a)



(b)



(c)

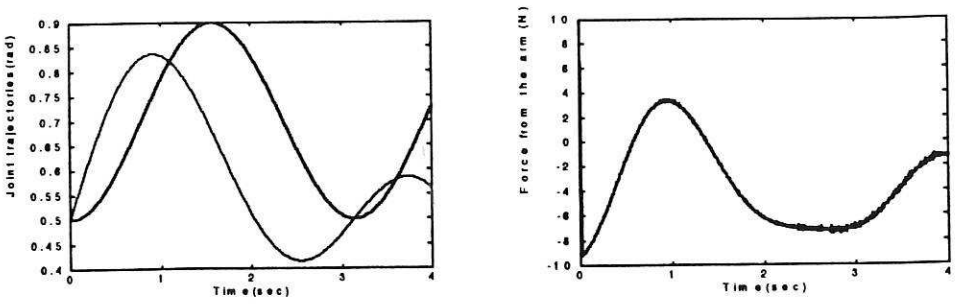


(d)

Figure 32 Manipulator trajectories and their errors by an MLP compensator in the case of platform motion (a) trajectories of the first joint(desired: dashed/dotted, actual: solid) (b) trajectories of the second joint(desired: dashed/dotted, actual: solid) (c) trajectories errors(first joint: solid, second joint: dashed/dotted) (d) error during the MLP learning process

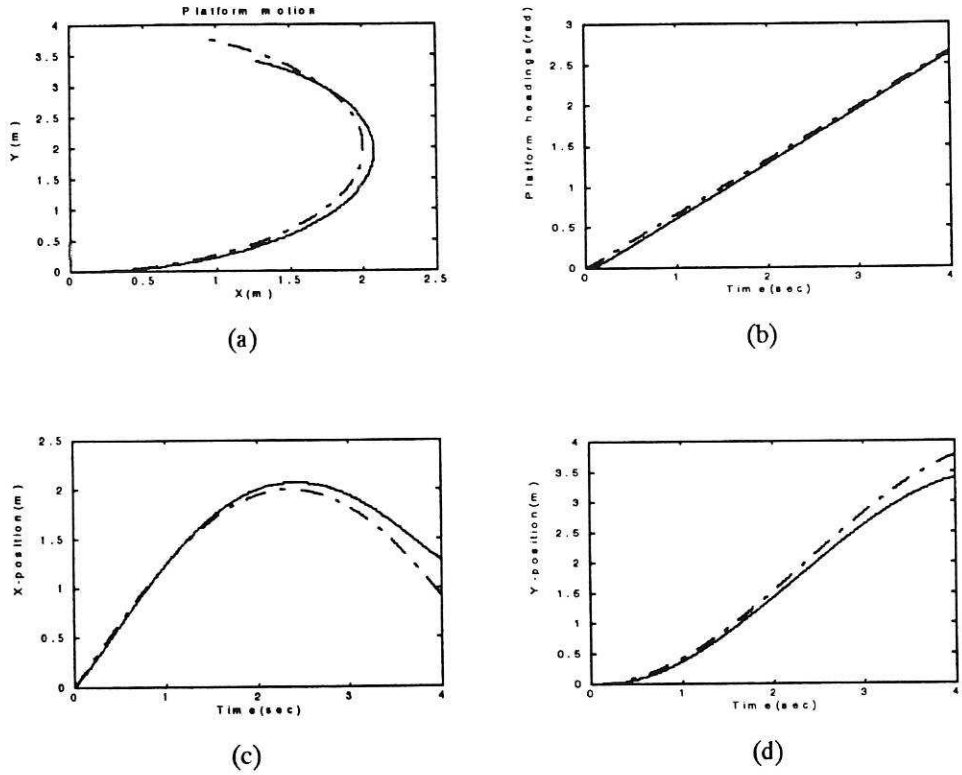
Case 2: Platform neural network compensation to the disturbance from the manipulator

Figure 33 shows the motion of the manipulator and its disturbant forces to the platform. Figure 34 is the simulation results from a PD controller only. The results by the neural network compensation is shown in Figure 35. The MLP can compensate the distubance quite well and the learning process converges steadily.



(a) (b)

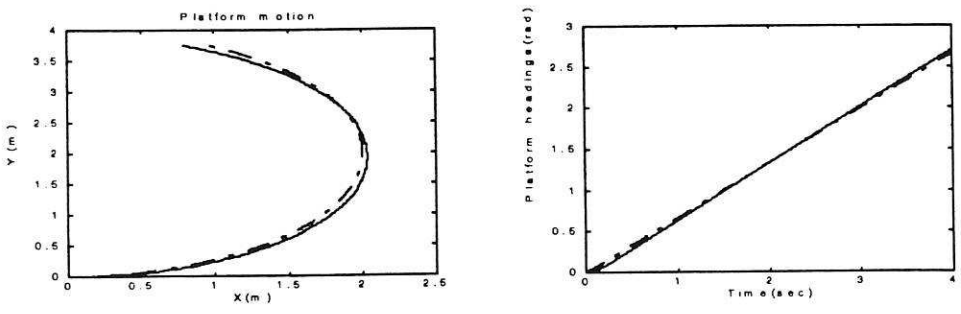
Figure 33 (a) The given manipulator joint motion(solid: first joint, dashed: second joint) and (b) its disturbant force to the platform



(a) (b)

(c) (d)

Figure 34 Platform trajectories by a PD controller in the case of manipulator motion(desired: dashed/dotted, actual: solid)



(a) (b)

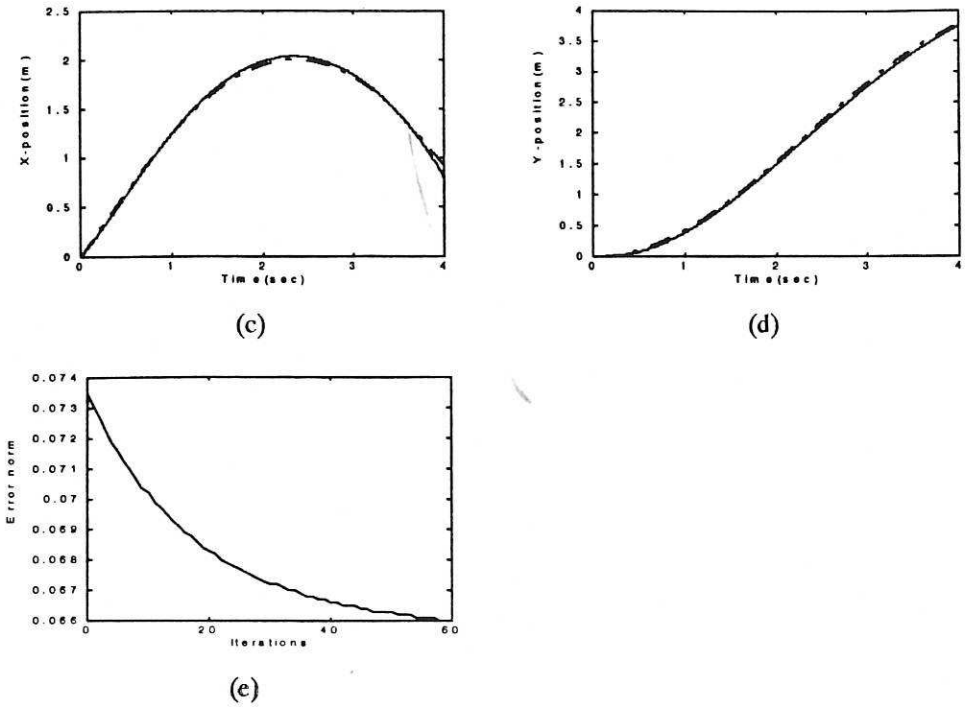


Figure 35 Platform trajectories by an MLP compensator in the case of manipulator motion (a)-(d) desired and actual trajectories (desired: dashed/dotted, actual: solid) (e) error during the MLP learning process

7. Conclusions

This report proposed a neural network compensation controller to a platform/manipulator system. The difficulty with the control of such a system is that the platform is subject to a non-holonomic constraint and there exist disturbances between the platform and manipulator motions. Traditional controllers(PID or adaptive controllers) can not achieve satisfactory control performance. Neural networks can virtually learn to compensate any tracking errors and have strong robustness. The presented simulation results well shown the advantages of the proposed neural network compensators. The multi-layered perceptron network is mainly considered in this report, but the radial basis functions network can also be applied to solve the similar problems as shown in two examples. Comparison was made between neural network controller with and without any model information. Due to dynamic requirements, without any model information the neural network controller has a larger tracking error than with model information. If no dynamic constraint is applied, the mappings between the output and the input of a robot motion controller become less complicated, and hence a neural network alone can learn the mappings and give good peformance.

References

[1] W. Miksch and D. Schroeder, "Performance functional based controller design for a mobile manipulator", Proceedings of the 1992 IEEE International Conference on robotics and Automation, Nice, France, May, pp. 227-232, 1992.

[2] K. J. Hunt, D. Sbarbaro, R. Zbikowski, and P. J. Gawthrop, "Neural Networks for Control Systems --- A Survey", Automatica, Vol.28, No.6, pp.1083-1112, 1992.

[3] R. Carrelli, E. F. Camacho, and D. Patino, "A neural network based feedforward adaptive controller for robots", IEEE Transactions on Systems, Man, and Cybernetics, Vol.25, No.9, September, 1995.

[4] D. Katic and M. Vukobratovic, "Connectionist Approaches to the control of manipulation robots at the executive hierarchical level: An Overview", Journal of Intelligent and Robotic Systems, Vol.10 pp.1-36, 1994.

[5] D. M. Katic and M. K. Vukobratovic, "Highly efficient robot dynamics learning by decomposed connectionist feedforward control structure", IEEE Transactions on Systems, Man, and Cybernetics, Vol. 25, No.1, pp.145-158, January 1995.

[6] N. Sadegh, "A perceptron network for functional identification and control of nonlinear systems", IEEE Transactions on Neural Networks, Vol. 4, No.6, November, 1993.

[7] S. M. Ziauddin and A. M. S. Zalzala, "Model-based compensation and comparison of neural network controllers for uncertainties of robotic arms", IEE Proceedings--Control Theory and Applications, Vol.142, No.5, pp.501-507, 1995.

[8] E. Zalama, P. Gaudiano, and J. L. Coronado, "A real-time, unsupervised neural network for the low_level control of a mobile robot in a nonstational environment", Neural Networks, Vol.8 No.1 pp.103-123, 1995.

[9] K. Saga, T. Sugasaka, M. Sekiguchi, S. Nagata, and K. Asakswa, "Mobile robot control by neural networks using self-supervised learning", IEEE Transactions on Industrial Electronics, Vol.39, No.6, December 1992.

[10] J. Borenstein and Y. Koren, "Motion control Analysis of a mobile robot", Journal of Dynamic Systems, Measurement, and Control, Vol.109, June, pp.73-79, 1987.

[11] N. Sarkar, X. Yun, and V. Kumar, "Control of Mechanical systems with rolling constraints: Application to dynamic control of mobile robots", The International Journal of Robotics Research, Vol.13, No.1, Februrary, pp.55-69, 1994.

[12] Y. Yamamoto and X. Yun, "Coordinating Locomotion and manipulation of a mobile manipulator", IEEE Transactions on Automatic Control, Vol.39, No.6, pp.1326-1332, June 1994.

[13] S. Jagannathan, F. L. Lewis, and K. Liu, "Motion control and obstacle avoidance of a mobile robot with an onboard manipulator", Journal of Intelligent Manufacturing, vol.5 pp.287-302, 1994.

[14] N. E. Sharkey, G. Manson, A. M. S. Zalzala, and J. N. H. Heemskek, "Experiments for the EPSRC project Neural Network Learning for Coordinated Robotic Control", Internal Report 0.1(draft), October, 1995.