

# 國立交通大學

## 電控工程研究所

### 博 士 論 文

飛彈防衛系統之任務分配及軌跡規劃

Design of Task Assignment and Path Evolution for Missile Defense System (MDS)



研 究 生：洪堃能

指 導 教 授：王啟旭 教授

中 華 民 國 一 〇 二 年 六 月

飛彈防衛系統之任務分配及軌跡規劃

Design of Task Assignment and Path Evolution for Missile Defense System (MDS)

研究生：洪堃能

Student : Kun-Neng Hung

指導教授：王啟旭

Advisor : Chi-Hsu Wang



Electrical Control Engineering

June 2013

Hsinchu, Taiwan, Republic of China

中華民國一〇二年六月

# 飛彈防衛系統之任務分配及軌跡規劃

研究生：洪堃能

指導教授：王啟旭 博士

國立交通大學電控工程研究所博士班

## 摘 要

針對非線性系統的控制問題，本論文發展兩個嶄新的控制架構。首先探討多自主系統(MAS, multi-agent system)，其為一包含多個自主運作的群體系統，能接受來自其他機構或者中央的命令來採取行動，並在我們所設定的條件內達成任務。飛彈防衛系統(MDS, missile defense system)即為一多自主系統的延伸架構，環境內有來自敵方的多個攻擊飛彈的威脅、有限多個據點將遭受戰損與有限多個防衛飛彈發射並各自攔截攻擊飛彈等，均是本論文第一部份的探討重點。在飛彈的任務規劃部份，傳統的任務分配使用窮舉法來做多對多的配對，雖然最終能找到滿足條件的最佳配對，但是當對象數量增加時所花費的計算量更是龐大，因此本論文提出自組織映射(SOM, self-organizing map)，其優勢在於降低配對的計算量並可根據多個據點的最小總戰損為目標來進行多個防衛飛彈與多個攻擊飛彈之間的配對，如此不僅可以加速配對時間也可以滿足所設定的最小總戰損條件。在控制器的部份，本論文根據飛彈導引法則為基礎建構提出智慧型模糊類神經網路(FNN, fuzzy neural network)控制器架構，相較於CMAC(cerebellar model articulation controller)在飛彈導引的運算時間過長及失誤距離過大，更進一

步改善飛彈導引系統的即時性；本論文所提出的控制器架構可經由李亞普諾夫穩定性證明來保證系統的穩定性，同時藉由參數學習的機制來克服系統的非線性。最後由模擬成果可明顯看出自組織映射在任務分配與模糊類神經網路控制器在飛彈導引控制的結合度極高，因此可以完整建構出一飛彈防衛系統。本論文之第二部份為高階霍普菲爾神經網路(HOHNN, high-order Hopfield-base neural network)應用於動態系統之鑑別。高階霍普菲爾神經網路中的函數型連結網路(FLN, functional link net)能提供額外的輸入給網路之各神經元。相對於傳統的霍普菲爾網路(HNN, Hopfield neural network)，本論文所提出的函數型連結網路具有系統化階次的數學表示法，具有較快的收斂速度及較少的計算負載。另外，函數型連結網路之權重更新，亦可藉由李亞普諾夫穩定理論來保證在非線性即時系統的鑑別收斂。針對各種基於霍普菲爾神經網路架構的比較，可由模擬結果及計算量分析顯示我們所提出的高階霍普菲爾神經網路在未知動態系統的鑑別具有較高的效能。

# Design of Task Assignment and Path Evolution for Missile Defense System (MDS)

Student: Kun-Neng Hung

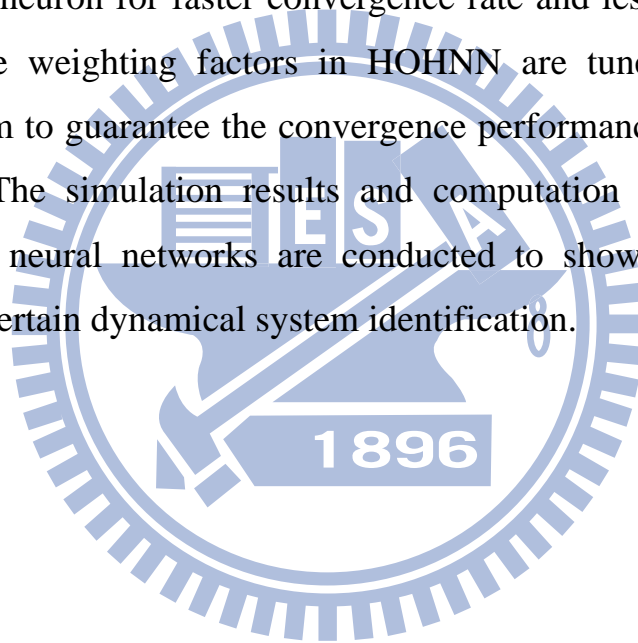
Advisor: Dr. Chi-Hsu Wang

Institute of Electrical Control Engineering  
National Chiao Tung University

## ABSTRACT

In this dissertation, two novel control schemes are proposed to solve the control problems of nonlinear systems. The first is the multi-agent system (MAS) consists of multiple autonomous systems which can activate, interact, and communicate with each others or from central command, and eventually complete some missions under the desired conditions. Missile defense system (MDS) is a suitable application of MAS: threat from multiple attacking missiles, some limited assets are under attack, and multiple defense missiles (or agents) are launched to intercept the associated attacking missiles (or targets), and a fuzzy neural network (FNN) controller with self-organizing map (SOM) for MAS are investigated in the first part of this thesis. The presented approaches are better than traditional exhausted method which can find the optimal solution though time-consuming when the processing data increases. The advantage of SOM is the less computational load under the condition of minimal total asset damages. Therefore, SOM can be adopted to not only dispatch the agents toward the targets, but also lower the computational load under the desired condition. Based on the missile guidance law, the proposed FNN can deal with the problems of large computational load and miss distance by the cerebellar model articulation controller (CMAC). Finally, the proposed SOM-based FNN

controller adopted in the highly nonlinear MDS can be guaranteed stable and the parameters can be updated via Lyapunov stability criterion. From the experimental results, it can be demonstrated the possibility of applying the proposed intelligent control method in MDS. In the second part of the thesis, the high-order Hopfield-based neural network (HOHNN) is proposed to the dynamical system identification. The functional link net (FLN) in HOHNN has additional inputs for each neuron. In comparison with the traditional Hopfield neural network (HNN), the compact structure of FLN with a systematic order mathematical representation combined into the proposed HOHNN has additional inputs for each neuron for faster convergence rate and less computational load. In addition, the weighting factors in HOHNN are tuned via the Lyapunov stability theorem to guarantee the convergence performance of real-time system identification. The simulation results and computation analysis for different Hopfield-based neural networks are conducted to show the effectiveness of HOHNN in uncertain dynamical system identification.



## Acknowledgement

在博士班求學六年多來，學生承蒙指導教授 王啟旭 博士辛勤、殷切地細心指導，無論在學業上的教導與鼓勵，或是待人處事的諄諄教誨，均使學生受益良多，由莘莘學子茁壯而成為巍巍大樹。在博士論文的撰寫期間，因為吾師及口試委員們給予諸多的教導與指正，才能讓本論文得以順利完成。在此，向諸位師長致上誠摯的謝意與敬意。

在學生修課期間也要感謝實驗室全體學長對學生在論文與研究上的教導。另外；感謝實驗室的全體學弟時常關心並提供生活上的支持，使得博士班的研究生活更顯得多采多姿，充滿美好的回憶。

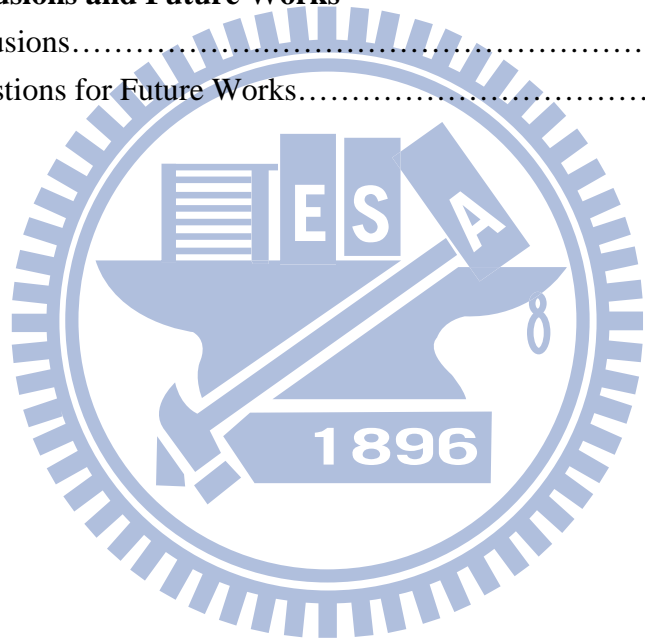
最後還要感謝多年來辛苦栽培我的家人以及所有曾經給予我關懷與協助的貴人，使我得以順利完成此階段之學業，願您們能與我一同分享這份成果的喜悅。

# Table of Contents

Abstract in Chinese .....	i
Abstract in English .....	iii
Acknowledgement .....	v
Table of Content .....	vi
List of Figures .....	viii
List of Tables .....	x
Nomenclatures .....	xi
<b>1. Introduction</b>	
1.1 Background and Motivation.....	1
1.2 Major Works.....	8
1.3 Dissertation Overview.....	9
<b>2. Dynamic Task Assignment with Path Control for Multi-Agent System using Intelligent Adaptive SOM-based Fuzzy Neural Network</b>	
2.1 Background and Motivation.....	10
2.2 Problem Formulation.....	10
2.3 The Self-Organizing Map (SOM).....	12
2.3.1 Description of SOM.....	12
2.3.2 Major Works.....	12
2.4 Design of Fuzzy Neural Network (FNN) Controller.....	15
2.4.1 Description of FNN.....	15
2.4.2 Major Works.....	18
2.5 The Lyapunov Stability Analysis.....	22
2.6 Illustrated Examples.....	24
2.7 Conclusions.....	34
<b>3. Toward a New Task Assignment and Path Evolution (TAPE) for Missile Defense System (MDS) using Intelligent Adaptive SOM with Fuzzy Neural Networks</b>	
3.1 Background and Motivation.....	36
3.2 Problem Formulation.....	36
3.3 The Three-Dimensional CLOS Guidance Model.....	38
3.4 One-To-One Agent-Target Path Evolution using FNN.....	42
3.5 The Design of SOM for Task Assignment of MDS.....	48
3.6 Illustrated Examples.....	56



3.7	Conclusions.....	67
<b>4.</b>	<b>Dynamical System Identification using High-Order Hopfield-based Neural Network (HOHNN)</b>	
4.1	Background and Motivation.....	69
4.2	High-Order Hopfield-based Neural Network (HOHNN) Models.....	69
4.2.1	Description of HOHNN.....	71
4.2.2	Major Works.....	72
4.3	The Function Approximation using HOHNN.....	74
4.4	The Lyapunov Tuning of HOHNN for Identification.....	78
4.5	Robust Learning Analysis.....	80
4.6	Illustrated Examples.....	85
4.7	Conclusions.....	92
<b>5.</b>	<b>Conclusions and Future Works</b>	
5.1	Conclusions.....	94
5.2	Suggestions for Future Works.....	95
	<b>References</b>	96
	<b>Vita</b>	104
	<b>Publication List</b>	105



## List of Figures

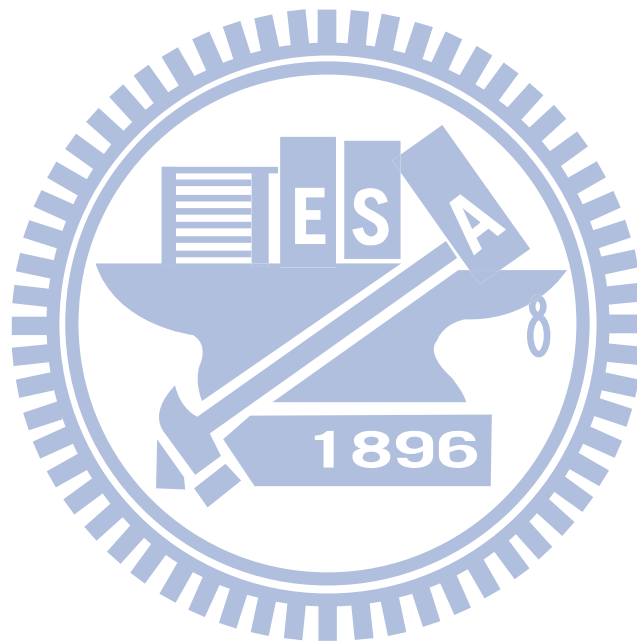
- Fig. 1-1 The relationships between TAPE, MAS, and MDS.
- Fig. 2-1 The closed-loop adaptive SOM-based FNN controller for MAS.
- Fig. 2-2 The structure of self-organizing map (SOM).
- Fig. 2-3 The architecture of fuzzy neural network (FNN).
- Fig. 2-4 The configuration of  $\mathbf{u}_{fnn}$  and  $\mathbf{u}_m$  for the  $i^{\text{th}}$  agent.
- Fig. 2-5 The closed-loop SOM-based FNN for MAS dynamic task assignment and path control.
- Fig. 2-6 The trajectories of 6 agents with 6 targets.
- Fig. 2-7 The errors and change of errors of MAS.
- Fig. 2-8 The trajectories of 8 agents with 8 targets.
- Fig. 2-9 The errors and change of errors of MAS.
- Fig. 2-10 The trajectories of 8 agents with 8 moving targets.
- Fig. 2-11 The errors and change of errors of MAS.
- Fig. 2-12 The trajectories of 64 agents with 64 targets.
- Fig. 2-13 The errors and change of errors of MAS.
- Fig. 3-1 The overall concept of adaptive SOM with FNN controller for MDS.
- Fig. 3-2 Three-dimensional agent-target pursuit diagram.
- Fig. 3-3 Definition of tracking error.
- Fig. 3-4 The configuration of  $\mathbf{u}_{fnn}$  and  $\mathbf{u}_m$  for single agent.
- Fig. 3-5 The structure of self-organizing map (SOM).
- Fig. 3-6 The self-organizing map (SOM) example in MDS.
- Fig. 3-7 The closed-loop configuration of SOM with FNN controller for MDS TAPE.
- Fig. 3-8 Block diagram representation of estimation algorithm for guidance information.
- Fig. 3-9 The closed-loop configuration of FNN controller for missile guidance.
- Fig. 3-10 The initial positions and angles of targets in the  $X_T$ - $Y_T$  plane.
- Fig. 3-11 The errors and change of errors of FNN controller in 25 battle scenario simulations.
- Fig. 3-12 The control inputs of FNN controller in 25 battle scenario simulations.
- Fig. 3-13 The trajectories of 9 agents with 9 targets.
- Fig. 3-14 The errors and change of errors of MAS.
- Fig. 3-15 The trajectories of 3 agents with 9 targets.
- Fig. 3-16 The errors and change of errors of MAS.
- Fig. 3-17 The trajectories of 4 agents with 12 targets.
- Fig. 3-18 The errors and change of errors of MAS.
- Fig. 4-1 The architecture of Hopfield neural network.
- Fig. 4-2 A compact structure of functional link net.
- Fig. 4-3 The HOHNN structure for a single neuron.

- Fig. 4-4 The closed-loop configuration of HOHNN for function approximation.
- Fig. 4-5 The uncontrolled regular-order Chen system plotted in the (a)  $x_3$ - $x_2$ - $x_1$  space, and (b) time series within 50 seconds.
- Fig. 4-6 The overall identification diagram of regular-order Chen system.
- Fig. 4-7 The system approximation comparisons between RHONN, HNN, FHOHNN, and HOHNN (a) for  $t = 0 \sim 15$  seconds, and the detailed drawing for (b)  $t = 0 \sim 0.5$  seconds and (c)  $t = 9.9 \sim 10.4$  seconds.
- Fig. 4-8 The error comparisons of system approximation between RHONN, HNN, FHOHNN, and HOHNN (a) for  $t = 0 \sim 15$  seconds, and the detailed drawing for (b)  $t = 0 \sim 0.5$  seconds and (c)  $t = 9.9 \sim 10.4$  seconds.
- Fig. 4-9 The mean square error comparisons of system approximation between RHONN, HNN, FHOHNN, and HOHNN (a) for  $t = 0 \sim 15$  seconds, and the detailed drawing for (b)  $t = 0 \sim 0.5$  seconds and (c)  $t = 9.9 \sim 10.4$  seconds.



## List of Tables

Table 2-1	Total path length comparisons for the three cases.
Table 2-2	Comparisons of computational loads in an iteration for all the cases.
Table 3-1	Comparisons of computational loads for the task assignment using exhaustive methods and the proposed SOM.
Table 3-2	The 25 guidance results (MD: Miss Distance; CL: Computational Load).
Table 4-1	Increase number of input pattern components with enhancement.
Table 4-2	Execution time for all the Hopfield-based neural networks.



# Nomenclatures

## < Chapter 3 >

$\psi_t$	Yaw angle of target.
$\theta_t$	Pitch angle of target.
$\psi_a$	Yaw angle of agent.
$\theta_a$	Pitch angle of agent.
$\phi_{ac}$	Roll angle command of agent.
$\sigma_t$	Azimuth angle of line-of-sight (LOS) to target.
$\gamma_t$	Elevation angle of LOS to target.
$\sigma_a$	Azimuth angle of LOS to agent.
$\gamma_a$	Elevation angle of LOS to agent.
$\Delta\sigma$	$\sigma_a - \sigma_t$ .
$\Delta\gamma$	$\gamma_a - \gamma_t$ .
$g$	Gravity acceleration.
$a_x$	Axial acceleration of agent.
$a_{yc}$	Yaw acceleration command of agent.
$a_{zc}$	Pitch acceleration command of agent.
$R_a$	Agent range from ground tracker.
$R_t$	Target range from ground tracker.
$v_a$	Agent velocity.
$v_t$	Target velocity.
$a_t$	Target acceleration.
$s\theta$	$\sin(\theta)$ .
$c\theta$	$\cos(\theta)$ .
$(X_I, Y_I, Z_I)$	Agent inertial frame.
$(X_A, Y_A, Z_A)$	Body frame.
$(X_L, Y_L, Z_L)$	LOS frame.
$(x_t, y_t, z_t)$	Target position in inertial frame.
$(x_a, y_a, z_a)$	Agent position in inertial frame.

# Chapter 1

## Introduction

### 1.1 Background and Motivation

The multi-agent system (MAS) has attracted increasing interests among researchers working on a wide variety of topics due to its broad applications in autonomous underwater vehicles, unmanned aerial helicopters, mobile multi-robot formations, and so on [1–8]. Some studies have reported on consensus control and synchronization [1–8], including the leaderless and modified leader-follower architecture in the presence of actuator faults. The main part of consensus control is that the agents communicate with each other via an adjacent graph and behaves in a similar manner to an equilibrium state. However, the dynamic task assignment between agents and targets was not considered in the studies, and few intelligent control techniques have been applied to find the agent trajectories. The task assignment of MAS is to decide the dispatching for agents toward the corresponding targets according to some exhaustive criteria, or to control a group of agents so that they can move to their designated target locations with the coordination and cooperation of each robot. Many researches have been proposed in the task assignment domain, like the intelligent transportation systems (ITS) [9–11], transport logistics [12], taxi dispatching [13], and so on. In [9–14], the most important factor considered in MAS is the total computational loads (or balance of group resources) which means the total real-time travel period has to be minimized. However, the considered targets in [9–14] were static, and dynamic task assignments with intelligent path control algorithms for MAS have not been developed yet. The traditional exhaustive method used to find the minimal total damaging cost is time-consuming, that is, the solution can be obtained, and the number of computation steps is  $D!/(D-N)!$ ,  $D \geq N$  when there are  $N$  agents and  $D$  targets. In practice the task assignment can not accept such an

inefficient method since the computational load will get heavier as  $N$  becomes larger. There are various algorithms proposed for the task assignment problem, such as the genetic algorithm [15], agent based algorithm [16], dynamic Tabu search algorithm [17], and graph matching algorithm [18]. However, these algorithms mainly focus on the target assignment problem without considering the nonlinear control and path planning of robots. In comparison with leader-following tracking problem, there are no information interchanges among the agents in our MAS because one agent just has to know whether its corresponding target has been matched to another agent. Moreover, it is convenient and efficient for task assignments that we do not need to consider the information delays between neighbors or the other agents due to the communication constraints. The missile defense system (MDS) provides a sort of protective shield against a limited missile attack. The incoming missiles are launched to attack limited assets which have their own significances. Once an asset is unfortunately destroyed by some incoming missiles, it will cause the corresponding damaging cost which can be regarded as the value of asset. Because the number of assets that are under attacking by unknown number of incoming missiles, the allocation of defending missiles becomes important to ensure the damaging cost are minimized (or the surviving assets are maximized). A dynamic programming for interceptor allocation problem in theater missile defense (TMD) [43] has been discussed to develop reliable defense guidelines. However, the approach in [43] is too complex to be implemented in practice. In this thesis, we will treat the MDS as an important application of MAS [1–8]. The defending missiles and incoming missiles in MDS represent as agents and targets in MAS, respectively. Therefore, a task assignment considered in MAS can easily deal with the allocation problem in MDS.

The self-organizing map (SOM) first proposed in [19] has the winner-takes-all property that activates the winner among a group of neuron based on competitive learning. The SOM has been studied for various areas such as web content mining [20], pattern classification [21], and image processing [22], etc. In [23], SOM has been proposed as a useful dynamic task

assignment method for a multi-robot system, in which the positions of agents are updated by the SOM with neighborhood update rule. Once the best match is found by SOM, the agents start to move their initial targets, using a simple incremental path planning mechanism with dynamic weighting factors. However, the weighting factors should be updated in the next iteration via the learning neighborhood vectors to account for the random indexed targets. Furthermore, the SOM consists of distributed and competitive methods that can efficiently and dynamically dispatch the agents to the respective targets. In comparison with traditional SOM algorithm, the SOM proposed in [24] eliminated the time consuming tuning in neighborhood function to reduce the computational load in task assignment. A simple incremental path planning [23] has been adopted to allow the agents to move toward the chosen targets; however, the high nonlinearities and uncertainties of the agents (such as robots) have never been considered. The neural network control technique has been adopted for various systems in recent works [25–30]. The important element is the parameterized neural network, which can approximate the unknown system dynamics after the learning process. In the past decades, the fuzzy logic and neural network have grown into a popular research topic [31–36]. The fuzzy neural network (FNN) has the advantages of fuzzy systems and neural network, because of the combination of the fuzzy reasoning capability and the neural network on-line learning capability [31]. The FNN has been adopted widely for the control and identification application of complex dynamical systems [32, 33].

In control purpose, the path evolution (or planning) adopted by FNN can smoothly and robustly be constructed when the nonlinear dynamics and uncertainties of agents are considered in MDS. The adaptive SOM with FNN controller constructed under the command line-of-sight (CLOS) missile guidance law [44–46] consists of FNN controller and monitoring controller. The principle of CLOS missile guidance law is to force the agent to fly as closely as possible along the instantaneous line-of-sight (LOS) between the ground tracker and the target. If the missile can continuously stay on the LOS, it will eventually hit the target. The



CLOS missile guidance law has been regarded as a low-cost guidance concept because it emphasizes placement of avionics on the launch platform, as opposed to mounting on the expendable weapons [44]. Many different guidance laws have been developed over the years, and with the advent of highly maneuverable targets, research on improved guidance laws is still active [47–49]. However, their methods have resulted in complicated controllers and some of the guidance laws require knowledge of the maneuvering model of the targets. These are limitations for guidance of missiles, therefore, the guidance system should be robust enough to reject disturbances, and the escaping model of the target should not be considered. A cerebellar model articulation controller (CMAC) in [46] has been developed under CLOS for missile guidance law. However, the CMAC structure is too complex to be implemented in real-time environment, and the enormous weight space and limited modeling capability in CMAC can be further improved using the proposed FNN controller with fewer mappings and layers. Moreover, the multi-agent-multi-target cases with task assignment have not been discussed in [44–49]. The surviving assets [43] in MDS are the most important role for the task assignment to efficiently make an interception list for agents to intercept targets. As long as the interception list is done by SOM, the FNN controller will be adopted to force the agents to intercept their corresponding targets. Finally, the overall task assignment and path evolution (or planning) (TAPE) in MAS can be achieved to be adopted in MDS. Figure 1-1 illustrates the relationships between TAPE, MAS, and MDS. The MDS (outermost circle) contains various spatial fields such as sea-based, space-based, and high altitude anti-ballistic missile systems. In order to cope with task assignment and path evolution (TAPE), the proposed MAS in this paper is considered as the application of multi-agent-multi-target space-based missile guidance.

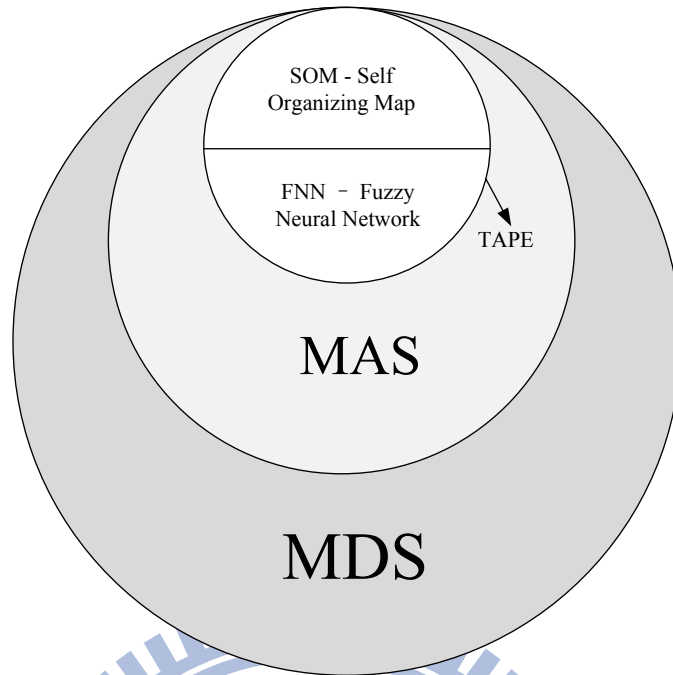


Fig. 1-1. The relationships between TAPE, MAS, and MDS.

The Hopfield neural network (HNN) proposed in 1982 [50] is an auto-associative learning network that consists of a set of neurons with a multiple-loop feedback structure in which the number of feedback loops is equal to the number of neurons. The HNN is a type of recurrent neural network (RNN) that has important capabilities that are not found in feed-forward neural networks (FNNs), such as attractor dynamics and the ability to store information for later use [51]. The abilities of HNN to deal with time-varying input or output through their own natural temporal operations [52–54] are particularly interesting. Researchers have devoted much attention to applying neural networks to identify and model nonlinear dynamical systems. Neural networks are suitable for identifying nonlinear dynamical systems because of their learning and memorization capabilities. In recent years, research on HNN has been conducted for pattern recognition [55], adaptive control [56], and crossbar switching problem [57]. The effectiveness of functional link net (FLN) in classification was first proposed in 1989 [58, 59] and has been combined with the HNN to create the high-order neural networks (HONNs). The extra input patterns of a FLN are formed

by either a functional expansion representation, tensor representation, or a combination of these two representations. The FLN may be conveniently used for functional approximation and pattern classification with faster convergence rate and less computational load than a multilayer perceptron (MLP) (or the static neural network) from many considerable interests in exploring the applications of functional link model to deal with nonlinearities and uncertainties [60–63]. A radial basis function neural network (RBFNN) was combined with a random-vector FLN to improve the recognition of English language script. In [61], FLN with higher-order statistics was introduced for signal enhancement. A nonlinear adaptive filter with pipelined Chebyshev functional link artificial recurrent neural network in [62] used a modification real-time recurrent learning algorithm for nonlinear colored signal prediction. In [63], a functional link net-adaptive neuro fuzzy system was adopted as a controller for robot path tracking purposes. The advantages of FLN shown in [64–66] indicated that not only is the efficiency of supervised learning greatly improved, but a flat net without hidden layer is sufficiently capable to do the job. However, the functional link artificial neural networks in [67] did not include the HNN. Furthermore, the weighting factors were tuned by back-propagation algorithm, which cannot guarantee the convergence of tuning results, especially in real-time applications.

The applications of MLP for identification were studied in [68]. However, the fact that RNN involves dynamic elements with lower connectivity and fewer weighting factors results in the easier learning process of RNN than that of MLP [38, 69]. Thus, a further extension of RNN to yield HOHNN for dynamical system identification is proposed in this thesis. HOHNN is basically formed by HNN and a compact functional link structure with a systematic order mathematical representation. The application of HOHNN in the identification is explored to show the advantages of extra inputs for each neuron. In [70], the initial efforts were only reflected toward the effectiveness of the nonlinear system identification via HOHNN.

From the above discussions of novel neural network structures, there are two motivations in this thesis. The first motivation is the task assignment for the agents to be forced in a bounded start region to their corresponding targets in a one-to-one mapping. If any target is to be inserted at any time instant, the new best-matching pairs will be dynamically found by SOM. Furthermore, the adaptive process of SOM was replaced by the proposed FNN control mechanism, in which the path planning of agents can be smoothly and robustly constructed. In order to effectively lead the high-order nonlinear agents to their targets, a FNN controller and monitoring controller are adopted in the adaptive process of SOM. Therefore, a new intelligent adaptive algorithm based on SOM is proposed in this thesis to deal with the dynamic task assignment and path control problem. In comparison with the traditional SOM, the total resources can also be minimized by this new approach, in which only the winner vectors are considered. The intelligent adaptive SOM-based FNN controller is operated in conjunction with the traditional SOM in order to find the best paths allowing all agents to go to their final targets. The Lyapunov adaptive process is adopted to update the weighting factors, which is very different from the simple update of weighting factors for path planning in traditional SOM. A new monitoring controller is also designed to work with FNN controller; thus, the agents can be forced to go to their corresponding targets within the constraints of nonlinear dynamics and uncertainties of the agents (or robots). The second motivation in this thesis is the improvement of Hopfield-based neural networks, in which a compact functional link structure with a systematic order mathematical representation in FLN can perform satisfactory results among the other Hopfield-based neural network. A Lyapunov-based tuning theorem is also proposed to find the optimal weighting factor matrix of HOHNN to achieve favorable approximation error, which can be attenuated to arbitrary specified level. The robust learning analysis is also discussed to improve the convergence performance. Finally, the simulation results and computation analysis for different Hopfield-based neural networks are conducted to show the effectiveness of HOHNN in uncertain dynamical system identification.

More detailed discussions and comparisons are proposed in this thesis.

## *1.2 Major Works*

In this dissertation, a SOM-based FNN controller with a monitoring controller is adopted for the task assignment and path evolution (or planning). The proposed FNN controller utilized for missile guidance is to mimic an ideal controller, and the monitoring controller is designed to compensate the tracking error between the FNN controller and the ideal controller. The parameters of FNN are tuned based on the Lyapunov stability criterion to achieve a favorable performance. In comparison with CMAC, the FNN controller behaves less miss distance and computational load in one-to-one agent-target missile guidance. In comparison with the result in [46] using CMAC, our control mechanism is much simplified with nearly the same accuracy. In the simulation results, it can be seen that the proposed FNN controller can not only effectively be adopted in the scenario, but also the computational load of proposed FNN controller is better than that by using CMAC [46]. Furthermore in the MAS, the SOM has the advantage of the dimension reduction from inputs and the efficient dispatching between agents and targets under the desired condition. From the results, the FNN controller combined with SOM can not only deal with the agent-target matching in two-dimensional space, but also force a group of agents to different number of targets in the real-time MDS environment. The objective of this new MDS system is to minimize the total damaging cost after executing the TAPE system, which is an immediate application of MAS. Excellent simulation results are obtained under three scenarios via TAPE to achieve successful MDS.

In addition, for the system identification, the other purpose of the dissertation is to develop a new Hopfield-based neural network in which the FLN structure with systematic mathematical representation can efficiently perform the nonlinear dynamical system identification. The proposed HOHNN can be guaranteed stable by the Lyapunov stability

criterion and its weighting factors can be adjusted to minimize the approximation error by the robust learning analysis. Finally, the proposed scheme is applied to identify the regular-order Chen system to illustrate its effectiveness. The illustrated examples demonstrate that the proposed HOHNN can obtain better identification performance than the other Hopfield-based neural networks.

### *1.3 Dissertation Overview*

Four MAS systems are illustrated to achieve successful results using this new approach. It is noted that this new proposed approach can also handle the insertion of random targets at any time instant and moving targets, which is illustrated in Cases 2 and 3 of Section VI.

This thesis is organized as follows. The problem of formulation of MAS and SOM algorithm for task assignment is first defined in Chapter 2. Chapter 3 presents the task assignment and path evolution (or planning) for MDS. The proposed FNN controller and Lyapunov stability analysis are also provided in Chapter 2 and Chapter 3. In Chapter 4, the HOHNN is proposed for nonlinear dynamical system identification. Finally, the discussions and future works of the proposed approach are given in Chapter 5.

The major contributions of this thesis are the successful developments of the following:

- 1) an adaptive fuzzy neural network (FNN) control system in which the Lyapunov stability theorem is used for on-line tuning of the missile guidance design parameters.
- 2) a monitoring controller is used to compensate the residual of the tracking error.
- 3) an online dispatching in multi-agent system (MAS) under the desired condition is adopted for the task assignment problem.
- 4) a battle scenario environment of the missile defense system (MDS) is constructed.
- 5) a novel high-order Hopfield-based neural network (HOHNN) is proposed for nonlinear dynamical system identification.

# Chapter 2

## Dynamic Task Assignment with Path Control for Multi-Agent System using Intelligent Adaptive SOM-based Fuzzy Neural Network

### 2.1 Background and Motivation

The traditional self-organizing map (SOM) aims to exclusively search the real-time shortest paths for all agents, thus allowing them to go to their targets. After this traditional task assignment, the weighting factors of our new SOM-based fuzzy neural network (FNN) controller are activated to force the agents toward their corresponding targets. The FNN controller is the main controller combining the fuzzy rules with the neural network. A monitoring controller is also designed to reduce the error between FNN controller and ideal controller. Using the Lyapunov constraints, the weighting factors for the proposed SOM-based FNN controller are updated to guarantee the stability of the path control system.

### 2.2 Problem Formulation

Consider a group of  $N$  agents in the  $M$ -dimensional workspace, it is desired to first perform task assignment by self-organizing map (SOM), after which the path control is activated so that all the agents are capable of going to their targets under the agent dynamics constraints. The dynamics for the  $i^{\text{th}}$  agent can be described by [37]

$$\mathbf{M}_i \ddot{\mathbf{a}}_i + \mathbf{f}_i = \mathbf{u}_i, \quad 1 \leq i \leq N \quad (2-1)$$

where  $\mathbf{a}_i \in \mathfrak{R}^M$  is the position;  $\mathbf{M}_i \in \mathfrak{R}^{M \times M}$  is the mass or inertia matrix;  $\mathbf{f}_i \in \mathfrak{R}^M$  represents the centripetal, Coriolis, gravitational effects and additive disturbances; and  $\mathbf{u}_i \in \mathfrak{R}^M$  represents the control input. We assume that

$$\mathbf{f}_i = \mathbf{F}_i^k + \mathbf{F}_i^u \quad (2-2)$$

where  $\mathbf{F}_i^k$  and  $\mathbf{F}_i^u$  represent the known and the unknown vectors of the  $i^{\text{th}}$  agent, respectively. We also assume that the unknown vector is bounded by a known bound  $\bar{F}_i$ . In other words, let

$$\|\mathbf{F}_i^u\| \leq \bar{F}_i \quad (2-3)$$

for all the  $N$  agents. Moreover, it is assumed that,  $\mathbf{M}_i$  is nonsingular and its lower and upper bounds are bounded by a known bound. In other words, the matrices  $\mathbf{M}_i$  satisfy

$$\mathbf{M}_{Li} \|\boldsymbol{\alpha}\|^2 \leq \boldsymbol{\alpha}^T \mathbf{M}_i(\mathbf{x}_i) \boldsymbol{\alpha} \leq \mathbf{M}_{Ui} \|\boldsymbol{\alpha}\|^2 \quad (2-4)$$

where  $\mathbf{M}_{Li} > 0$  and  $\mathbf{M}_{Ui} < \infty$  are the known lower and upper bounds of the  $i^{\text{th}}$  agent, respectively, and  $\boldsymbol{\alpha} \in \mathcal{R}^M$  is an arbitrary vector. Assume that the initial positions  $\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_N\}$  of agents are located randomly in a given bounded space, and the initial positions  $\mathbf{T} = \{\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_D\}$  of targets are distributed randomly in the same  $M$ -dimensional workspace. Then the main control objective is to find the best-matching pairs iteratively by SOM, such that agents can find their relatively shorter paths to the final chosen targets. Therefore, the planning paths for all agents may have initial chattering (or transient) effects; nevertheless, these disappear once the best-match time  $t_b$  is reached. The stability of the closed-loop system can be guaranteed by adaptively adjusting the weighting factors for the proposed SOM-based FNN controller with the aid of a monitoring controller. Define the control inputs  $\mathbf{U} = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N\}$  for all the agents, the overall concept proposed in this chapter can be illustrated in the following Fig. 2-1.



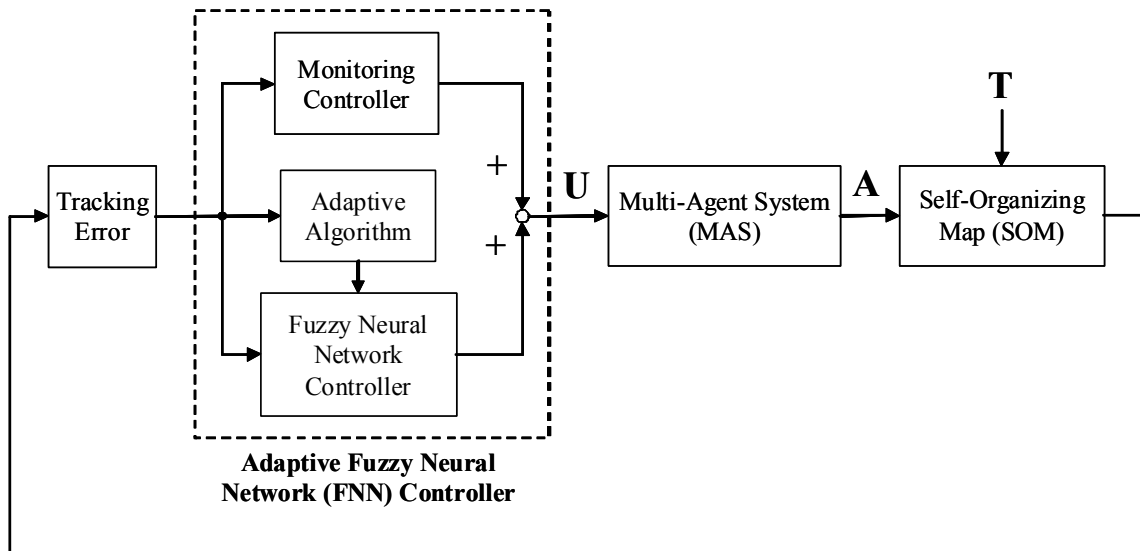


Fig. 2-1. The closed-loop adaptive SOM-based FNN controller for MAS.

## 2.3 The Self-Organizing Map (SOM)

### 2.3.1 Description of SOM

The principal goal of SOM is to transform an input pattern of arbitrary dimension into a one- or two-dimensional discrete map as well as to perform this transformation adaptively in a topologically ordered fashion [19, 38]. The SOM is suitable for dealing with the dynamic task assignment because the dimension of the targets can be simplified, and mapped to the relatively corresponding agents. Furthermore, the SOM can iteratively search the best-matching pairs if the targets and agents are dynamically inserted into the workspace. The overall MAS system can be considered a self-organizing system which can adjust its basic structure when its environment changes.

### 2.3.2 Major Works

The algorithm of the SOM proceeds first by initializing the synaptic weights in the network, such that it can be done by assigning them in random indexed patterns. Thus, no

prior index is imposed on the feature map. Once the network has been properly initialized, there are three essential processes involved in the formation of the SOM, as follows.

**Random Indexed Process:**

In order to prevent the dependence of an agent on the initial workspace configuration and the input data order, all targets are placed in random indexed patterns in iterations after the first sampling time  $T_S$ . For each input pattern shown in Fig. 2-2, the random indexed input vectors chosen from the positions of targets are denoted as

$$\mathbf{R} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_d, \dots, \mathbf{r}_D\}, \quad t > T_S \quad (2-5)$$

where  $T_S$  is the sampling time. As long as an iteration starts, the target vector is transformed to the random indexed input vector. The neurons in the network compute their respective values of a discriminant function. This discriminant function then provides the basis for competition among the neurons. The particular neuron with the largest value of discriminant function is declared the winner of the competition. The synaptic weight vector of each neuron in the network has the same dimension as the input space. Let the synaptic weight vector corresponding to the input  $\mathbf{r}_d$  be denoted by

$$\mathbf{P}_d = [\mathbf{p}_{1,d} \quad \mathbf{p}_{2,d} \quad \dots \quad \mathbf{p}_{N,d}]^T. \quad (2-6)$$

To find the best match for the input vector  $\mathbf{r}_d$  with the synaptic weight vectors  $\mathbf{p}_{i,d}$ , we compare the inner products  $\mathbf{p}_{i,d}^T \mathbf{r}_d$  to the  $N$  agents and select the largest. Based on maximizing the inner product  $\mathbf{p}_{i,d}^T \mathbf{r}_d$ , the best-matching criterion is mathematically equivalent to minimizing the Euclidean distance. If we use the index  $i_w$  to identify the neuron that best matches the input vector  $\mathbf{r}_d$ , we may then determine the index of winner neuron  $i_w$ , which satisfies the following condition

$$i_w = i(\mathbf{r}_d) = \arg \min_i \|\mathbf{r}_d - \mathbf{p}_{i,d}\|, \quad i = 1, 2, \dots, N \quad (2-7)$$

that sums up the essence of the random indexed process among the neurons. Depending on the application of interest, the response of the network could either be the index of the winning neuron or the synaptic weight vector closest to the input vector in a Euclidean sense.

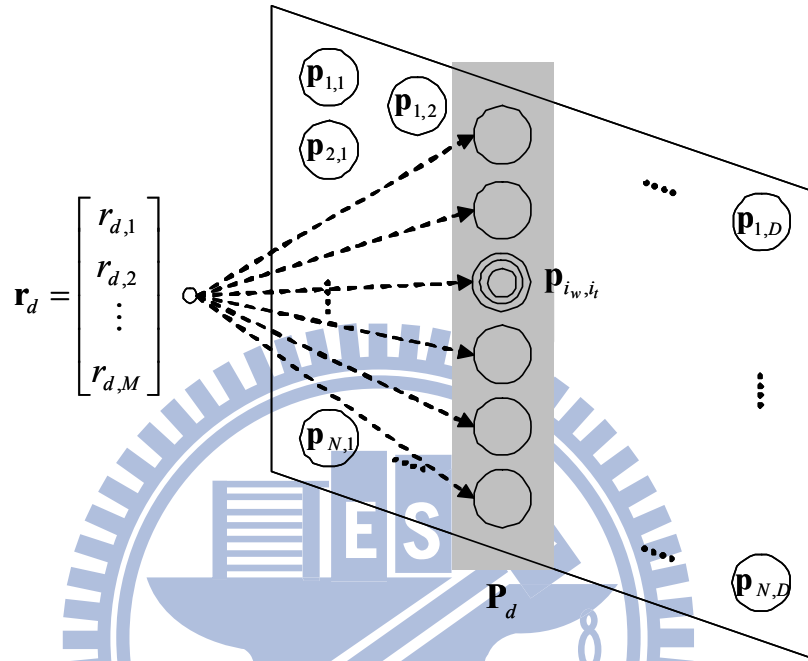


Fig. 2-2. The structure of self-organizing map (SOM).

**Competitive Process:**

The winner neuron determines the spatial location of a topological neighborhood of excited neurons. In traditional SOM, the winning neuron locates the center of a topological neighborhood of cooperating neurons. In this chapter, the neighborhood of the winner is neglected since the agents move toward their corresponding targets without any cooperative process. For a given target as an input, the output neurons compete to be the winner according to a specified criterion described as

$$[i_w, i_t] = \min \{D_{i,d}, i = 1, 2, \dots, N; d = 1, 2, \dots, D; \{i, d\} \notin \Omega\} \quad (2-8)$$

where  $[i_w, i_t]$  denotes that the pair in which the  $i_t^{\text{th}}$  target from the  $i_w^{\text{th}}$  agent is the winner, and  $\Omega$  is the set of neurons in which the winner has been chosen in an iteration. The distance  $D_{i,r}$  is given as

$$D_{i,d} = \|\mathbf{r}_d - \mathbf{a}_i\|. \quad (2-9)$$

As long as the  $N$  winners are found in an iteration, the index of agents  $\mathbf{A}$  are re-allocated to obtain a new  $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_N\}$  corresponding to the targets to be used in the adaptive process. From the above two processes, the computational load for finding the best-matching pairs can be obtained as  $O(N^2)$ . In comparison with traditional SOM method, the new adaptive SOM method eliminates the time consuming tuning in neighborhood function and is able to reduce the computational load in the task assignment of MAS.

#### **Adaptive Process:**

The last process enables the excited neurons to increase the individual values of their discriminant functions in relation to the input patterns through suitable adjustments applied to their synaptic weights. In the competitive process, we define the group vector consisting of the winner agents defined as  $\mathbf{W}$ . This group vector is then utilized to obtain the error matrix to update the weights of the winner. In comparison with incremental adjustment in traditional SOM, the proposed adaptive FNN controller can handle the overall path control for high-order nonlinear agents. This updating method is explained in the following section.

## *2.4 Design of Fuzzy Neural Network (FNN) Controller*

### *2.4.1 Description of FNN*

The FNN architecture in this thesis shown in Fig. 2-3 is a fully linked layer, in which the input layer accepts the input variables, the fuzzification layer calculates the Gaussian membership function and represents the fuzzy rules, and the output layer sums the output of the fuzzification layers. The fuzzy system in internal FNN is trained by the neural network

adaptive algorithm. Therefore, the fuzzy inference system and artificial neural network can complementarily operate for the controlling of nonlinear dynamical systems. For each layer in the following figure, the superscripted number represents each layer and the subscripted number represents the neuron in this layer. The detailed net input and net output are represented as follows.

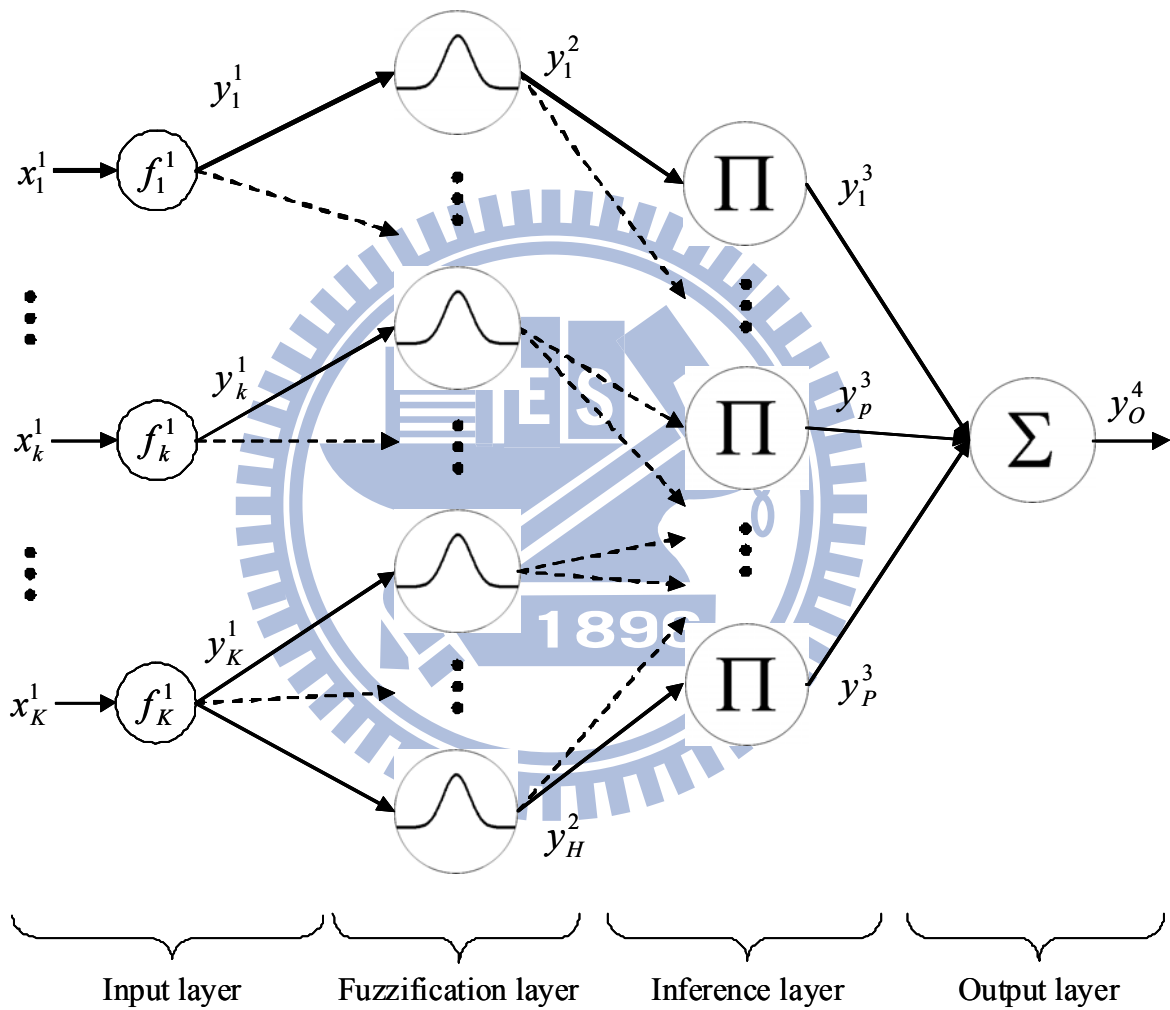


Fig. 2-3. The architecture of fuzzy neural network (FNN).

**Input layer:**

An input vector is fed into the input layer of the  $i^{\text{th}}$  agent. The net input and output of the input layer are presented as follows.

$$net_k^1 = x_k^1 \quad (2-10)$$

$$y_k^1 = f_k^1(net_k^1) = net_k^1 \quad (2-11)$$

where  $net_k^1$  represents the net input in this layer, and  $x_k^1$  and  $y_k^1$  are defined as the  $k^{\text{th}}$  input and output to the node of input layer, respectively. Each node in this layer represents an input linguistic variable. The presentations of notation in following layers are similar to those defined in this layer.

### Fuzzification layer:

Each node performs the fuzzification operation and acts as an element for membership degree calculation, in which the Gaussian function is adopted as the membership function of the IF-parts of the fuzzy rules given by

$$net_h^2 = -\frac{(x_k^2 - m_{kh})^2}{v_{kh}^2} \quad (2-12)$$

$$y_h^2 = f_h^2(net_h^2) = \exp(net_h^2) \quad (2-13)$$

where  $m_{kh}$  and  $v_{kh}$  are referred to as the mean and the standard deviation of the Gaussian function, respectively.

### Inference layer:

Let  $A_1^h, A_2^h, \dots, A_K^h$ , and  $B^h$  denote the fuzzy sets characterized by their corresponding membership function in (2-12) and (2-13) in the function layer, the  $h^{\text{th}}$  fuzzy rule can be defined as

Rule  $h$ : IF  $x_1^2$  is  $A_1^h$  AND  $x_2^2$  is  $A_2^h$  AND  $\dots$ ,  $x_K^2$  is  $A_K^h$  THEN  $x_h^3$  is  $B^h$ .

The inference layer implements the fuzzy AND aggregation operation which is chosen as the simple PRODUCT operation. Each node multiplies the incoming signals and outputs the result of this product as

$$net_p^3 = \prod_{h=1}^H w_{hp}^3 x_h^3 \quad (2-14)$$

$$y_p^3 = f_p^3(net_p^3) = \frac{net_p^3}{\sum_{p=1}^P net_p^3} \quad (2-15)$$

where  $w_{hp}^3$  represents the rule weight of the  $h^{\text{th}}$  fired rule between the function layer and the inference layer.

### Output layer:

Each node multiplies the incoming signals and outputs the result of this product as follows

$$net_o^4 = \sum_{p=1}^P w_{po}^4 x_p^4 \quad (2-16)$$

$$y_o^4 = f_o^4(net_o^4) = net_o^4 \quad (2-17)$$

where  $w_{po}^4$  represents the output action strength of the  $o^{\text{th}}$  output associated with the  $p^{\text{th}}$  rule.

Finally, the overall representation is given by

$$y_o = y_o^4 = \sum_{p=1}^P w_{po}^4 \prod_{h=1}^H \exp(-(x_k^1 - m_{kh})^2 / (v_{kh})^2). \quad (2-18)$$

In summary, the FNN output can be presented as

$$y_o = \sum_{p=1}^P w_{po}^4 \theta_p(x_k^1, m_{kh}, v_{kh}) \quad (2-19)$$

where

$$\theta_p(x_k^1, m_{kh}, v_{kh}) = \prod_{h=1}^H \exp(-(x_k^1 - m_{kh})^2 / (v_{kh})^2). \quad (2-20)$$

The above (2-20) represents the firing weight of the  $p^{\text{th}}$  neuron in the rule layer. The output in the output layer of FNN is adopted as the main controller to the MAS.

## 2.4.2 Major Works

The control problem of the MAS is to control the position of the winner target so that they can move to the desired target. The error matrix of MAS is defined as

$$\mathbf{e} = \mathbf{R} - \mathbf{W} = \{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_i, \dots, \mathbf{e}_N\}. \quad (2-21)$$

Considering the  $i^{\text{th}}$  agent, the tracking error vector defined as

$$\mathbf{e}_{s_i} = [\mathbf{e}_i \quad \dot{\mathbf{e}}_i]^T \quad (2-22)$$

represents the input vector fed into the input node of FNN controller. For the ease of notation, the tracking error vector for single agent is denoted as  $\mathbf{e}_s$ . Assuming all the system dynamics are well known and that there exists an ideal controller for  $i^{\text{th}}$  agent and  $d^{\text{th}}$  target based on the optimal control design, we then arrive at [39]:

$$\mathbf{u}_{id} = \mathbf{M}_i \ddot{\mathbf{r}}_d + \mathbf{f}_i + \mathbf{M}_i (k_1 \dot{\mathbf{e}}_i + k_2 \mathbf{e}_i), \quad 1 \leq i \leq N, \quad 1 \leq d \leq D. \quad (2-23)$$

Applying (2-23) into (2-1), the following error dynamics in two-dimensional workspace can be given

$$\dot{\mathbf{e}}_s = \mathbf{K} \mathbf{e}_s \quad (2-24)$$

where

$$\mathbf{K} = \begin{bmatrix} 0 & 1 \\ -k_2 & -k_1 \end{bmatrix}$$

is a Hurwitz matrix by choosing proper  $k_1$  and  $k_2$ . However, the ideal controller  $\mathbf{u}_{id}$  is difficult to implement in practice since the system dynamics is highly nonlinear and sometimes unavailable. Therefore, in order to control the output state efficiently, the control law is assumed to take the following form:

$$\mathbf{u}_i = \mathbf{u}_{fnn} + \mathbf{u}_m \quad (2-25)$$

where  $\mathbf{u}_{fnn}$  is a FNN controller, and  $\mathbf{u}_m$  is a monitoring controller. The FNN control  $\mathbf{u}_{fnn}$  is the main tracking controller used to imitate the ideal controller in (2-23), and the monitoring controller  $\mathbf{u}_m$  is designed to recover the residual approximation error. The monitoring controller, which is similar to a hitting controller in a traditional sliding mode controller, is



derived in the sense of Lyapunov theorem to cope with all system uncertainties to guarantee the stability of the system. Fig. 2-4 illustrates the concept of (2-25) in our new approach. The FNN structure shown in Figs. 2-3 and 2-4 has been considered. For simplicity, the following  $\mathbf{m}$  and  $\mathbf{v}$  vectors are defined to collect all parameters in the hidden layer of Fig. 2-3 given as

$$\mathbf{m} = [m_{11} \cdots m_{K1} \quad m_{12} \cdots m_{K2} \quad \cdots \quad m_{1H} \cdots m_{KH}]^T \quad (2-26)$$

$$\mathbf{v} = [v_{11} \cdots v_{K1} \quad v_{12} \cdots v_{K2} \quad \cdots \quad v_{1H} \cdots v_{KH}]^T \quad (2-27)$$

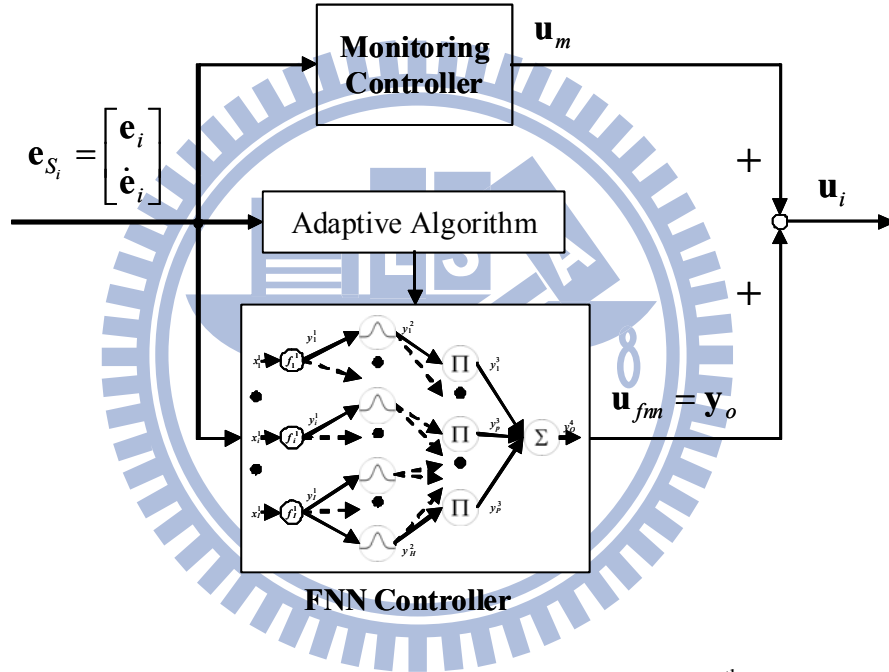


Fig. 2-4. The configuration of  $\mathbf{u}_{fm}$  and  $\mathbf{u}_m$  for the  $i^{\text{th}}$  agent.

Then, the output of FNN can be represented in vector form as

$$\mathbf{y}_o = \mathbf{w}_e^T \boldsymbol{\theta}(\mathbf{x}, \mathbf{m}, \mathbf{v}) \quad (2-28)$$

where  $\mathbf{y}_o = y_o^A$ ,  $\mathbf{w}_e = [w_{e,1} \quad w_{e,2} \quad \cdots \quad w_{e,H}]^T$ , and  $\boldsymbol{\theta} = [\theta_1 \quad \theta_2 \quad \cdots \quad \theta_H]^T$ . By the universal approximation theorem, there exists an ideal  $\mathbf{y}_o^* = \mathbf{y}_o(\mathbf{x}, \mathbf{w}_e^*, \mathbf{m}^*, \mathbf{v}^*)$  such that [40, 41]

$$\mathbf{y}_o = \mathbf{y}_o^* + \mathbf{E} = \mathbf{w}_e^{*T} \boldsymbol{\theta}(\mathbf{x}, \mathbf{m}^*, \mathbf{v}^*) + \mathbf{E} \quad (2-29)$$

where  $\mathbf{E}$  denotes the approximation error and  $\mathbf{w}_e^*$ ,  $\mathbf{m}^*$ , and  $\mathbf{v}^*$  are the optimal parameter vectors of  $\mathbf{w}_e$ ,  $\mathbf{m}$ , and  $\mathbf{v}$ , respectively. In fact, the optimal parameter vectors needed to best approximate a given nonlinear function are difficult to determine. Thus, an estimate function is defined as

$$\hat{\mathbf{y}}_o = \mathbf{y}_o(\mathbf{x}, \hat{\mathbf{w}}_e, \hat{\mathbf{m}}, \hat{\mathbf{v}}) = \hat{\mathbf{w}}_e^T \boldsymbol{\theta}(\mathbf{x}, \hat{\mathbf{m}}, \hat{\mathbf{v}}) \quad (2-30)$$

where  $\hat{\mathbf{w}}_e$ ,  $\hat{\mathbf{m}}$ , and  $\hat{\mathbf{v}}$  are the estimates of  $\mathbf{w}_e^*$ ,  $\mathbf{m}^*$ , and  $\mathbf{v}^*$ , respectively. For notational convenience, we denote  $\boldsymbol{\theta}^* = \boldsymbol{\theta}(\mathbf{x}, \mathbf{m}^*, \mathbf{v}^*)$  and  $\hat{\boldsymbol{\theta}} = \boldsymbol{\theta}(\mathbf{x}, \hat{\mathbf{m}}, \hat{\mathbf{v}})$ . Then, we define the estimation error as

$$\begin{aligned} \tilde{\mathbf{y}}_o &= \mathbf{y}_o - \hat{\mathbf{y}}_o \\ &= \mathbf{y}_o^* - \hat{\mathbf{y}}_o + \mathbf{E} \\ &= \mathbf{w}_e^{*T} \boldsymbol{\theta}^* - \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \mathbf{E} \\ &= (\hat{\mathbf{w}}_e^T + \tilde{\mathbf{w}}_e^T)(\hat{\boldsymbol{\theta}} + \tilde{\boldsymbol{\theta}}) - \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \mathbf{E} \\ &= \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \hat{\mathbf{w}}_e^T \tilde{\boldsymbol{\theta}} + \tilde{\mathbf{w}}_e^T \tilde{\boldsymbol{\theta}} - \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \mathbf{E} \\ &= \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \hat{\mathbf{w}}_e^T \tilde{\boldsymbol{\theta}} + \tilde{\mathbf{w}}_e^T \tilde{\boldsymbol{\theta}} + \mathbf{E} \end{aligned} \quad (2-31)$$

where  $\tilde{\mathbf{w}}_e = \mathbf{w}_e^* - \hat{\mathbf{w}}_e$  and  $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}^* - \hat{\boldsymbol{\theta}}$ . In the following, some tuning laws are derived to on-line tune the parameters of the FNN to achieve favorable estimation. To achieve this goal, we use the linearization technique to transform the nonlinear Gaussian functions into partially linear form so that the Lyapunov theorem extension can be applied [40] as follows

$$\tilde{\boldsymbol{\theta}} = \begin{bmatrix} \tilde{\theta}_1 \\ \vdots \\ \tilde{\theta}_H \end{bmatrix} = \begin{bmatrix} \frac{\partial \theta_1}{\partial \mathbf{m}} \\ \vdots \\ \frac{\partial \theta_H}{\partial \mathbf{m}} \end{bmatrix}_{\mathbf{m}=\hat{\mathbf{m}}} \tilde{\mathbf{m}} + \begin{bmatrix} \frac{\partial \theta_1}{\partial \mathbf{v}} \\ \vdots \\ \frac{\partial \theta_H}{\partial \mathbf{v}} \end{bmatrix}_{\mathbf{v}=\hat{\mathbf{v}}} \tilde{\mathbf{v}} + \mathbf{H} \equiv \boldsymbol{\theta}_m^T \tilde{\mathbf{m}} + \boldsymbol{\theta}_v^T \tilde{\mathbf{v}} + \mathbf{H} \quad (2-32)$$

where  $\tilde{\mathbf{m}} = \mathbf{m}^* - \hat{\mathbf{m}}$ ,  $\tilde{\mathbf{v}} = \mathbf{v}^* - \hat{\mathbf{v}}$ ,  $\boldsymbol{\theta}_m = \left[ \frac{\partial \theta_1}{\partial \mathbf{m}} \quad \dots \quad \frac{\partial \theta_H}{\partial \mathbf{m}} \right]_{\mathbf{m}=\hat{\mathbf{m}}}$ ,  $\boldsymbol{\theta}_v = \left[ \frac{\partial \theta_1}{\partial \mathbf{v}} \quad \dots \quad \frac{\partial \theta_H}{\partial \mathbf{v}} \right]_{\mathbf{v}=\hat{\mathbf{v}}}$ ,

and  $\mathbf{H}$  is the higher-order term, and  $\frac{\partial \theta_h}{\partial \mathbf{m}}$  and  $\frac{\partial \theta_h}{\partial \mathbf{v}}$  are defined respectively as

$$\begin{bmatrix} \frac{\partial \theta_h}{\partial \mathbf{m}} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{(h-1) \times k} & \frac{\partial \theta_h}{\partial m_{1h}} & \dots & \frac{\partial \theta_h}{\partial m_{kh}} & \mathbf{0}_{(H-h) \times k} \end{bmatrix} \quad (2-33)$$

$$\begin{bmatrix} \frac{\partial \theta_h}{\partial \mathbf{v}} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{(h-1) \times k} & \frac{\partial \theta_h}{\partial v_{1h}} & \dots & \frac{\partial \theta_h}{\partial v_{kh}} & \mathbf{0}_{(H-h) \times k} \end{bmatrix}. \quad (2-34)$$

Substituting (2-32) into (2-31) gives

$$\begin{aligned} \tilde{\mathbf{y}}_o &= \tilde{\mathbf{w}}_e^T \tilde{\boldsymbol{\theta}} + \hat{\mathbf{w}}_e^T (\boldsymbol{\theta}_m^T \tilde{\mathbf{m}} + \boldsymbol{\theta}_v^T \tilde{\mathbf{v}} + \mathbf{H}) + \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \mathbf{E} \\ &= \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \tilde{\mathbf{m}}^T \boldsymbol{\theta}_m \hat{\mathbf{w}}_e + \tilde{\mathbf{v}}^T \boldsymbol{\theta}_v \hat{\mathbf{w}}_e + \mathbf{d} \end{aligned} \quad (2-35)$$

where  $\hat{\mathbf{w}}_e^T \boldsymbol{\theta}_m^T \tilde{\mathbf{m}} = \tilde{\mathbf{m}}^T \boldsymbol{\theta}_m \hat{\mathbf{w}}_e$  and  $\hat{\mathbf{w}}_e^T \boldsymbol{\theta}_v^T \tilde{\mathbf{v}} = \tilde{\mathbf{v}}^T \boldsymbol{\theta}_v \hat{\mathbf{w}}_e$  are used since they are scalars, and the uncertain term  $\mathbf{d} = \hat{\mathbf{w}}_e^T \mathbf{H} + \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \mathbf{E}$  is assumed to be bounded by  $\|\mathbf{d}\| \leq \Delta$ . Since the uncertainty bound  $\Delta$  is difficult to determine, it is on-line estimated in the following section.

## 2.5 The Lyapunov Stability Analysis

The proposed control system is comprised of an FNN identifier and an optimal controller defined in (2-25), in which  $\mathbf{u}_{fnn}$  is used to mimic the ideal controller  $\mathbf{u}_{id}$ , and the compensation tangent controller  $\mathbf{u}_m$  is used to compensate for the difference between the FNN controller and the ideal controller. Substituting (2-25) into (2-1) and using (2-23), the error dynamic equation becomes

$$\dot{\mathbf{e}}_s = \mathbf{K} \mathbf{e}_s + \mathbf{B} (\mathbf{u}_{id} - \mathbf{u}_{fnn} - \mathbf{u}_m) = \mathbf{K} \mathbf{e}_s + \mathbf{B} (\tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \tilde{\mathbf{m}}^T \boldsymbol{\theta}_m \hat{\mathbf{w}}_e + \tilde{\mathbf{v}}^T \boldsymbol{\theta}_v \hat{\mathbf{w}}_e + \mathbf{d} - \mathbf{u}_m) \quad (2-36)$$

where  $\mathbf{B} = \mathbf{M}_i^{-1}$  is a bounded matrix. Since  $\mathbf{K}$  is a Hurwitz matrix, given a symmetric positive-definite matrix  $\mathbf{Q} \in \mathfrak{R}^{2 \times 2}$ , there exists a symmetric positive-definite matrix  $\mathbf{P} \in \mathfrak{R}^{2 \times 2}$ , such that the following Lyapunov equation [39, 42]

$$\mathbf{K}^T \mathbf{P} + \mathbf{P} \mathbf{K} = -\mathbf{Q} \quad (2-37)$$

is satisfied.

**Theorem 2-1:** Consider the dynamic nonlinear system represented by (2-1) with the control law in (2-25), where the FNN identifier is designed as (2-30). Then, the weighting vectors  $\hat{\mathbf{w}}_e$ ,  $\hat{\mathbf{m}}$ , and  $\hat{\mathbf{v}}$  will remain bounded, and the performance errors will approach zero. The parameters are updated by the following learning rules:

$$\dot{\hat{\mathbf{w}}}_e = -\dot{\tilde{\mathbf{w}}}_e = \eta_w \mathbf{e}_s^T \mathbf{PB}\hat{\boldsymbol{\theta}} \quad (2-38)$$

$$\dot{\hat{\mathbf{m}}} = -\dot{\tilde{\mathbf{m}}} = \eta_m \mathbf{e}_s^T \mathbf{PB}\boldsymbol{\theta}_m \hat{\mathbf{w}}_e \quad (2-39)$$

$$\dot{\hat{\mathbf{v}}} = -\dot{\tilde{\mathbf{v}}} = \eta_v \mathbf{e}_s^T \mathbf{PB}\boldsymbol{\theta}_v \hat{\mathbf{w}}_e \quad (2-40)$$

$$\mathbf{u}_m = \Delta \tanh(\mathbf{e}_s^T \mathbf{PB}) \quad (2-41)$$

where  $\eta_w$ ,  $\eta_m$ , and  $\eta_v$  are the positive real values. Then, the stability of the FNN control system can be guaranteed.

*Proof:*

Let the Lyapunov-like function candidate be

$$V = \frac{1}{2} \mathbf{e}_s^T \mathbf{P} \mathbf{e}_s + \frac{1}{2\eta_w} \tilde{\mathbf{w}}_e^T \tilde{\mathbf{w}}_e + \frac{1}{2\eta_m} \tilde{\mathbf{m}}^T \tilde{\mathbf{m}} + \frac{1}{2\eta_v} \tilde{\mathbf{v}}^T \tilde{\mathbf{v}}. \quad (2-42)$$

Taking the derivative of  $V$  in (2-42) with respect to time and using (2-36) and (2-37), yields

$$\begin{aligned} \dot{V} &= \frac{1}{2} \dot{\mathbf{e}}_s^T \mathbf{P} \mathbf{e}_s + \frac{1}{2} \mathbf{e}_s^T \mathbf{P} \dot{\mathbf{e}}_s + \frac{1}{\eta_w} \tilde{\mathbf{w}}_e^T \dot{\tilde{\mathbf{w}}}_e + \frac{1}{\eta_m} \tilde{\mathbf{m}}^T \dot{\tilde{\mathbf{m}}} + \frac{1}{\eta_v} \tilde{\mathbf{v}}^T \dot{\tilde{\mathbf{v}}} \\ &= \frac{1}{2} \mathbf{e}_s^T (\mathbf{K}^T \mathbf{P} + \mathbf{P} \mathbf{K}) \mathbf{e}_s + \mathbf{e}_s^T \mathbf{PB} (\tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\theta}} + \tilde{\mathbf{m}}^T \boldsymbol{\theta}_m \hat{\mathbf{w}}_e + \tilde{\mathbf{v}}^T \boldsymbol{\theta}_v \hat{\mathbf{w}}_e) \\ &\quad + \mathbf{e}_s^T \mathbf{PB} (\mathbf{d} - \mathbf{u}_m) + \frac{1}{\eta_w} \tilde{\mathbf{w}}_e^T \dot{\tilde{\mathbf{w}}}_e + \frac{1}{\eta_m} \tilde{\mathbf{m}}^T \dot{\tilde{\mathbf{m}}} + \frac{1}{\eta_v} \tilde{\mathbf{v}}^T \dot{\tilde{\mathbf{v}}} \\ &= -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \tilde{\mathbf{w}}_e^T (\mathbf{e}_s^T \mathbf{PB}\hat{\boldsymbol{\theta}} + \frac{1}{\eta_w} \dot{\tilde{\mathbf{w}}}_e) + \tilde{\mathbf{m}}^T (\mathbf{e}_s^T \mathbf{PB}\boldsymbol{\theta}_m \hat{\mathbf{w}}_e + \frac{1}{\eta_m} \dot{\tilde{\mathbf{m}}}) \\ &\quad + \tilde{\mathbf{v}}^T (\mathbf{e}_s^T \mathbf{PB}\boldsymbol{\theta}_v \hat{\mathbf{w}}_e + \frac{1}{\eta_v} \dot{\tilde{\mathbf{v}}}) + \mathbf{e}_s^T \mathbf{PB} (\mathbf{d} - \mathbf{u}_m) \end{aligned} \quad (2-43)$$

Substituting the learning rules (2-38)–(2-41) into (2-43), (2-43) becomes

$$\begin{aligned}
\dot{V} &= -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \mathbf{e}_s^T \mathbf{P} \mathbf{B} (\mathbf{d} - \mathbf{u}_m) \\
&\leq -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \left| \mathbf{e}_s^T \mathbf{P} \mathbf{B} \right| \|\mathbf{d}\| - \Delta \mathbf{e}_s^T \mathbf{P} \mathbf{B} \tanh(\mathbf{e}_s^T \mathbf{P} \mathbf{B}) \\
&\leq -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \Delta \left| \mathbf{e}_s^T \mathbf{P} \mathbf{B} \right| \\
&\leq -\frac{1}{2} \|\mathbf{e}_s\| \lambda_{\min}(\mathbf{Q}) \|\mathbf{e}_s\| + \|\mathbf{e}_s\| \|\mathbf{P}\| \|\mathbf{B}\| \Delta \\
&\leq -\frac{1}{2} (\lambda_{\min}(\mathbf{Q}) - 1) \|\mathbf{e}_s\|^2 + \|\mathbf{P} \mathbf{B}\|^2 \|\Delta\|
\end{aligned} \tag{2-44}$$

where  $\lambda_{\min}(\mathbf{Q})$  is the minimum eigenvalue of  $\mathbf{Q}$ . Since  $\lambda_{\min}(\mathbf{Q})$  can be chosen as  $\lambda_{\min}(\mathbf{Q}) > 1$ , then (2-44) reveals that

$$\int_0^t \|\mathbf{e}_s\|^2 dt \leq \frac{2}{\lambda_{\min}(\mathbf{Q}) - 1} (|V(0)| + |V(t)|) + \frac{2}{\lambda_{\min}(\mathbf{Q}) - 1} \|\mathbf{P} \mathbf{B}\|^2 \int_0^t \|\Delta\| dt \tag{2-45}$$

for all  $t \geq 0$ . Furthermore, if  $\Delta$  is squared integratable, then from (2-45),  $\mathbf{e}_s \in L_2$  has been proven [39]. In addition, the right hand side of (2-45) is bounded, that is,  $\dot{\mathbf{e}}_s \in L_\infty$ . Using Barbalat's Lemma [39], we can prove that  $\lim_{t \rightarrow \infty} \|\mathbf{e}_s\| = 0$  when  $\int_0^t \|\Delta\| dt < \infty$ . The stability of the overall approximation scheme is guaranteed based on the above results and the Lyapunov stability theorem. Based on (2-38)–(2-40), the adaptive law of weighting factors in an element form can be obtained. Thus, the Lyapunov stability theorem is guaranteed under the optimal approximation model with no modeling error. **Q.E.D.**

## 2.6 Illustrated Examples

In this section, four numerical simulation cases are presented in order to illustrate the effectiveness of the proposed new intelligent SOM-based FNN controller discussed in previous section. For ease of plotting, we only consider agents and targets in a bounded two-dimensional space; however, qualitatively, the results are expected to be the same for higher dimensions. We consider agents with point-mass dynamics with unknown mass and additive sinusoidal disturbances. In other words, we consider the model

$$\mathbf{M}_i \ddot{\mathbf{a}}_i + \mathbf{f}_i = \mathbf{u}_i, \quad 1 \leq i \leq N. \tag{2-46}$$

Without loss of generality, we assume that unity mass  $\mathbf{M}_i = \mathbf{1}$  are for all the  $N$  agents, and that there exists the following unknown uncertainties.

**Case 2-1:**  $N = D = 6$

$$\begin{cases} \mathbf{f}_1 = \begin{bmatrix} e^{-t} \\ \sin(t^2) \end{bmatrix}, \mathbf{f}_2 = \begin{bmatrix} e^{-2t} \\ \cos(t^2) \end{bmatrix}, \mathbf{f}_3 = \begin{bmatrix} \cos(t^2) \\ e^{-3t} \end{bmatrix}, \\ \mathbf{f}_4 = \begin{bmatrix} -\sin(t^2) \\ e^{-3t} \cos(t) \end{bmatrix}, \mathbf{f}_5 = \begin{bmatrix} -\cos(t) \sin(t^2) \\ e^{-3t} \sin(t) \end{bmatrix}, \text{ and } \mathbf{f}_6 = \begin{bmatrix} e^{-5t} \sin(t) \\ \sin(t) \cos(t^2) \end{bmatrix} \end{cases} \quad (2-47)$$

**Cases 2-2 and 2-3:**  $N = D = 8$

$$\begin{cases} \mathbf{f}_1 \sim \mathbf{f}_6 \text{ are the same as Case 1,} \\ \mathbf{f}_7 = \begin{bmatrix} e^{-3t} \cos(t) \\ e^{-5t} \sin(t) \end{bmatrix}, \text{ and } \mathbf{f}_8 = \begin{bmatrix} e^{-2t} \sin(t) \\ e^{-5t} \cos(t) \end{bmatrix} \end{cases} \quad (2-48)$$

**Case 2-4:**  $N = D = 64$

$$\mathbf{f}_i = \sin(t^2) e^{-t} \boldsymbol{\lambda}, 1 \leq i \leq N \quad (2-49)$$

where  $\boldsymbol{\lambda} \in \mathcal{R}^{M \times N}$  is a random vector with all the elements bounded in  $[-1, 1]$ . Note that it satisfies the bounded assumption  $\|\mathbf{f}_i\| \leq 1, 1 \leq i \leq 64$ . As controller parameters in the simulations below, we choose

$$\mathbf{Q} = \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}, \mathbf{P} = \begin{bmatrix} 2.2 & 0.2 \\ 0.2 & 0.24 \end{bmatrix},$$

and  $k_1 = k_2 = 5$ . The weighting factors  $\eta_w = 5$  and  $\eta_m = \eta_v = 0.2$  are chosen. Figure 2-5 shows the closed-loop configuration of SOM-based FNN for MAS dynamic task assignment and path control.

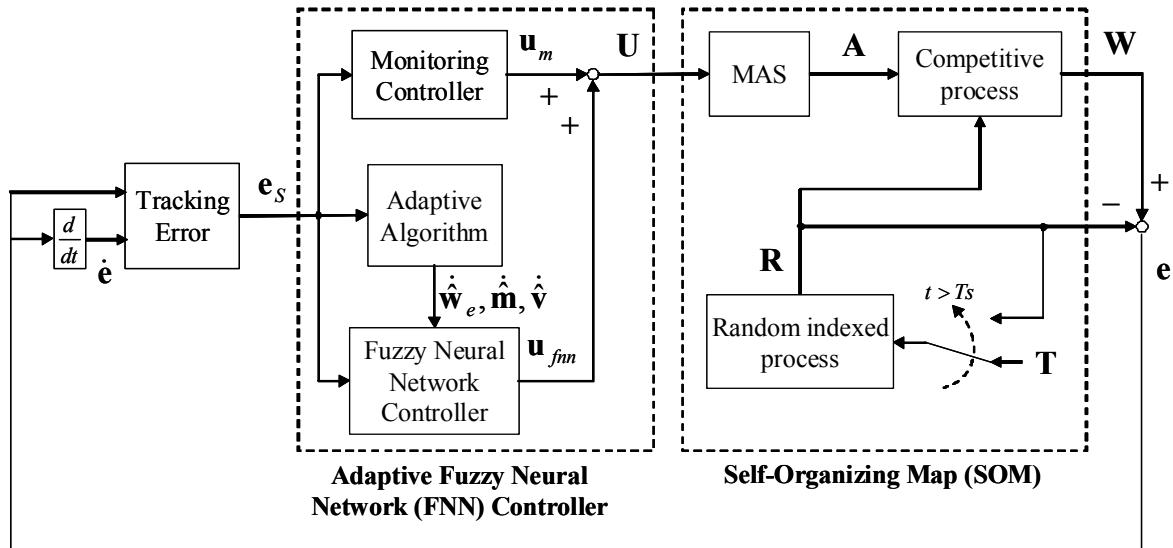


Fig. 2-5. The closed-loop SOM-based FNN for MAS dynamic task assignment and path control.

The best-match time  $t_b$  is added and shown in the simulation results. In the following figures, the agents marked as 'X' are randomly located in the grey circle, and the points marked as '◇' are the positions of targets which are the input to the SOM for finding the winner agents. When the number of agents and targets are different, our proposed approach can also be applied and implemented in the MAS. In this chapter, the agent-target matching pairs of SOM are assumed to be completed before the control inputs are fed to the MAS for path control. In other words, time delay is not considered in the proposed SOM. Moreover, we assume that the number of agents and targets are the same at any time moment in the following cases in order to construct the agent-target matching pair. Therefore, if there is one target inserted into the MAS, there should be one more agent inserted.

### Case 2-1: Static random targets

Consider 6 agents and 6 targets in the same two-dimensional (2D) workspace shown in Fig. 2-6. The targets find their matching agents that are then forced to their corresponding targets via SOM-based FNN controller. In order to ensure and check whether or not the best-matching pair is chosen, the dash line shows the best-matching pairs for all the agents

and targets. It is apparent that the transient chattering effects in all paths disappeared after the time is larger than  $t_b$ . In Fig. 2-7, it can be seen that  $t_b = 1.91$  seconds is the best-match time; after  $t_b$ , the errors and change of errors will no longer chatter because the best-matching pairs are found. Finally, the best match is completed and the tracking error converges to a satisfactory small value.

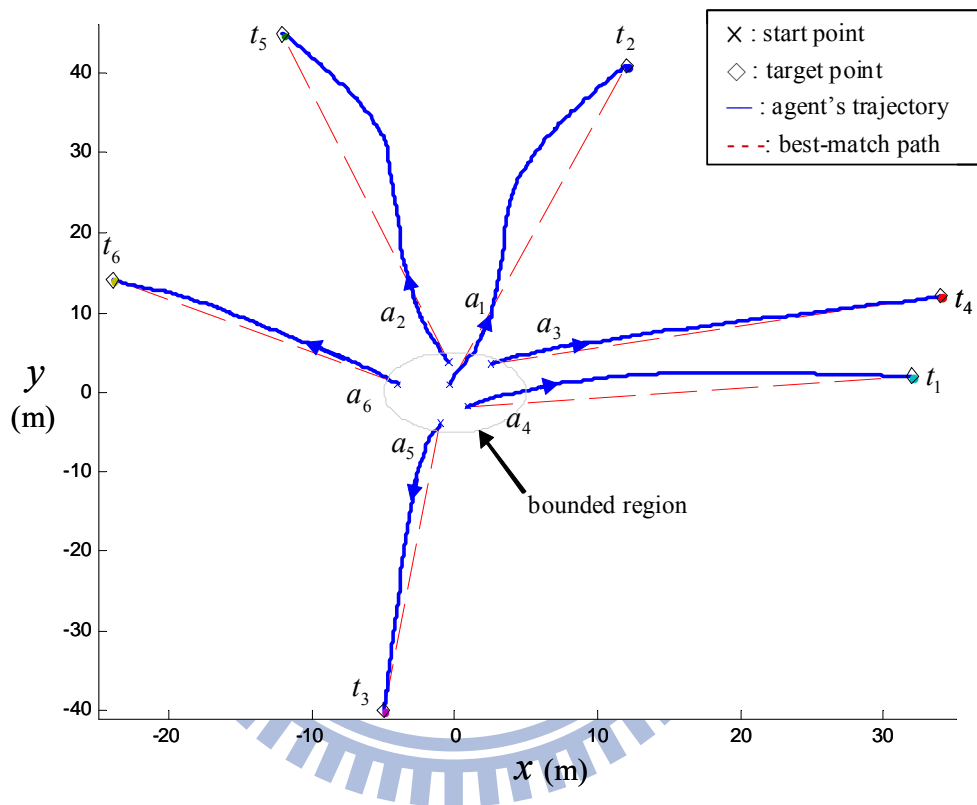


Fig. 2-6. The trajectories of 6 agents with 6 targets.



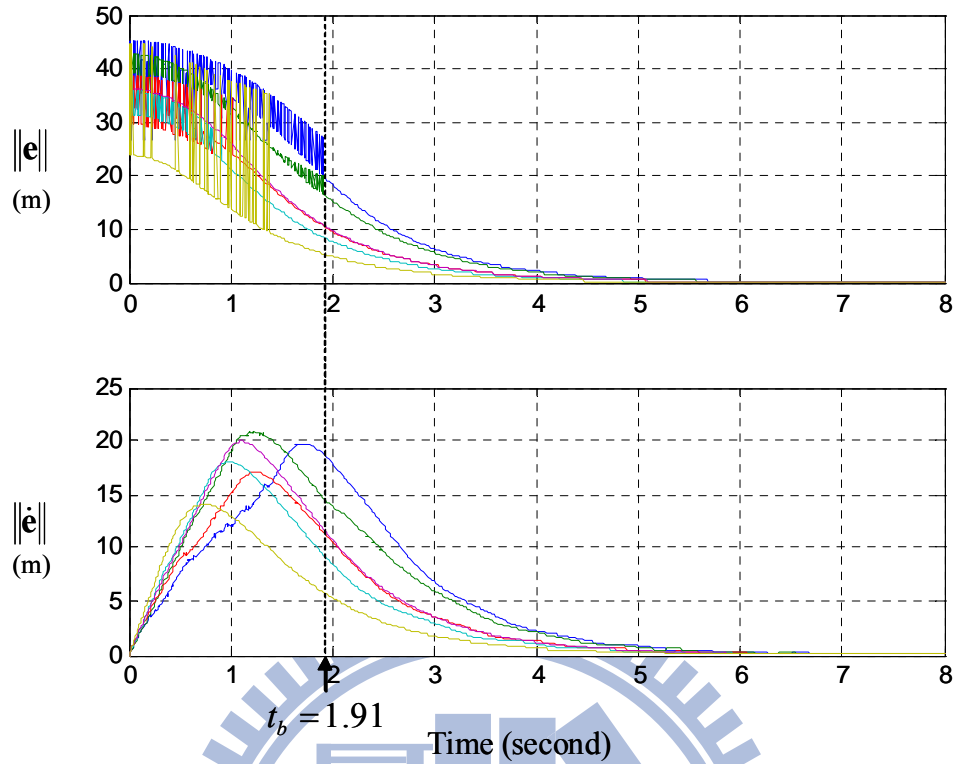


Fig. 2-7. The errors and change of errors of MAS.

### Case 2-2: Insertion of random targets

Consider 6 agents and 6 targets in the same two-dimensional (2D) workspace shown in Fig. 2-8. Define the insertion time  $t_i$  for the MAS are the time point when the additional random targets are inserted into the workspace. In this case, the 7<sup>th</sup> and 8<sup>th</sup> targets and their corresponding agents are inserted when  $t_i = 2$  seconds and  $t_i = 4$  seconds. As soon as one target is inserted, there will be an agent produced at a random location in the given bounded region, and the best match will be automatically completed by the adaptive SOM for all the new agent-target pairs as shown in Fig. 2-8. In Fig. 2-9, it can be seen that the best match is reached after  $t_b = 4$  seconds even if the chattering transient peak happened twice at the two insertion times.

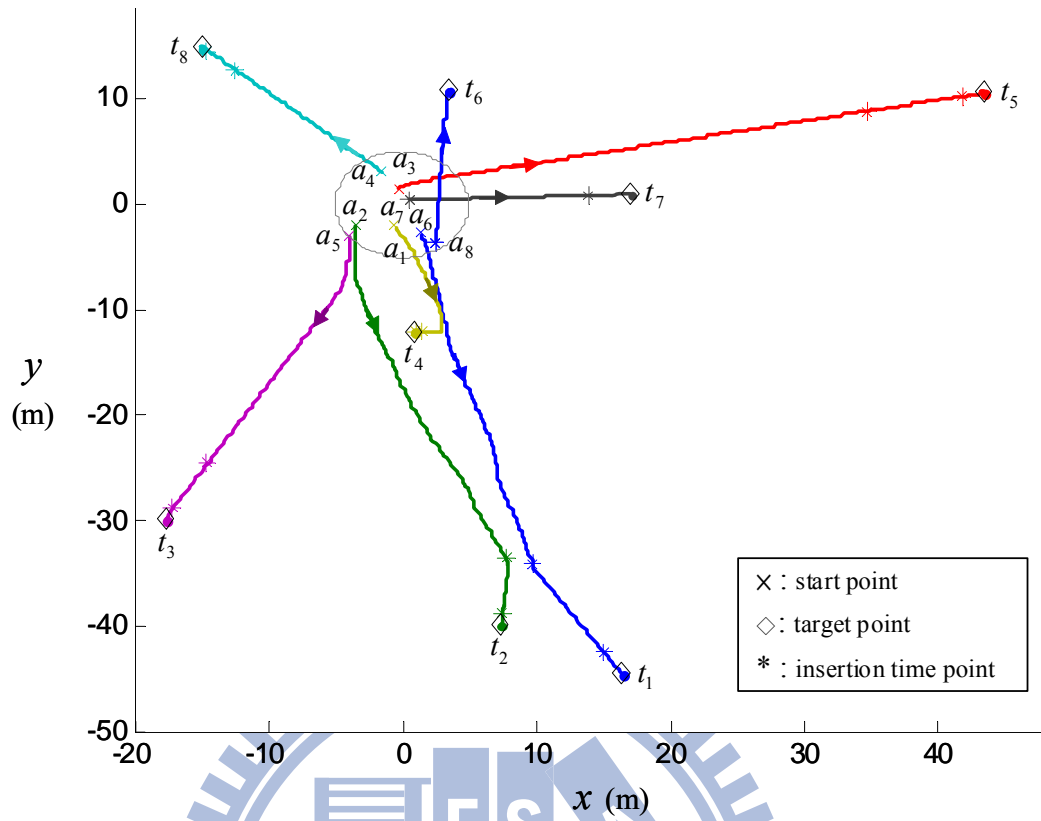


Fig. 2-8. The trajectories of 8 agents with 8 targets.

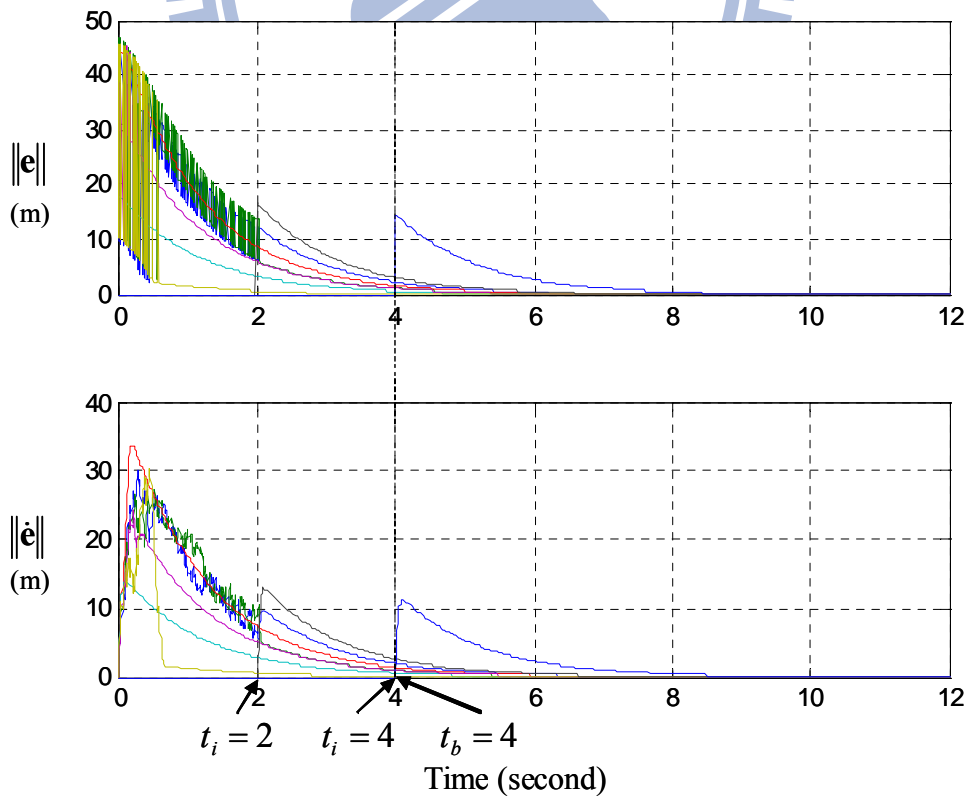


Fig. 2-9. The errors and change of errors of MAS.

On the other hand, the traditional exhaustive method has no obvious advantage on the dynamic task assignment when additional targets and agents are inserted into the workspace. However, in order to search the shortest total patch length, the computational load in the traditional exhaustive method will take from  $6!$  ( $= 720$ ) to  $8!$  ( $= 40,320$ ) computational loads each iteration according to the number of targets after the insertion time.

### **Case 2-3: Moving random targets**

Assume that the targets move a random bounded distance in  $[-2, 2]$  each  $T_m = 1$  seconds for the first five seconds which can be defined by the user. However, the targets have to be static before the simulation time to avoid the racing situation for the agents and targets. Consider 8 agents and 8 moving targets in the same two-dimensional (2D) workspace shown in Fig. 2-10. The red '◇' points indicate the initial target positions; the green '◇' points indicate the temporary positions every  $T_m$  seconds, and the incrementally move paths with directions are marked as gray dash lines and black arrows. Finally, the targets stop at the dark '◇' points which represent the static positions after four seconds. It can be obviously seen that no matter where the targets move, the agents instantly update their trajectories toward their corresponding targets.

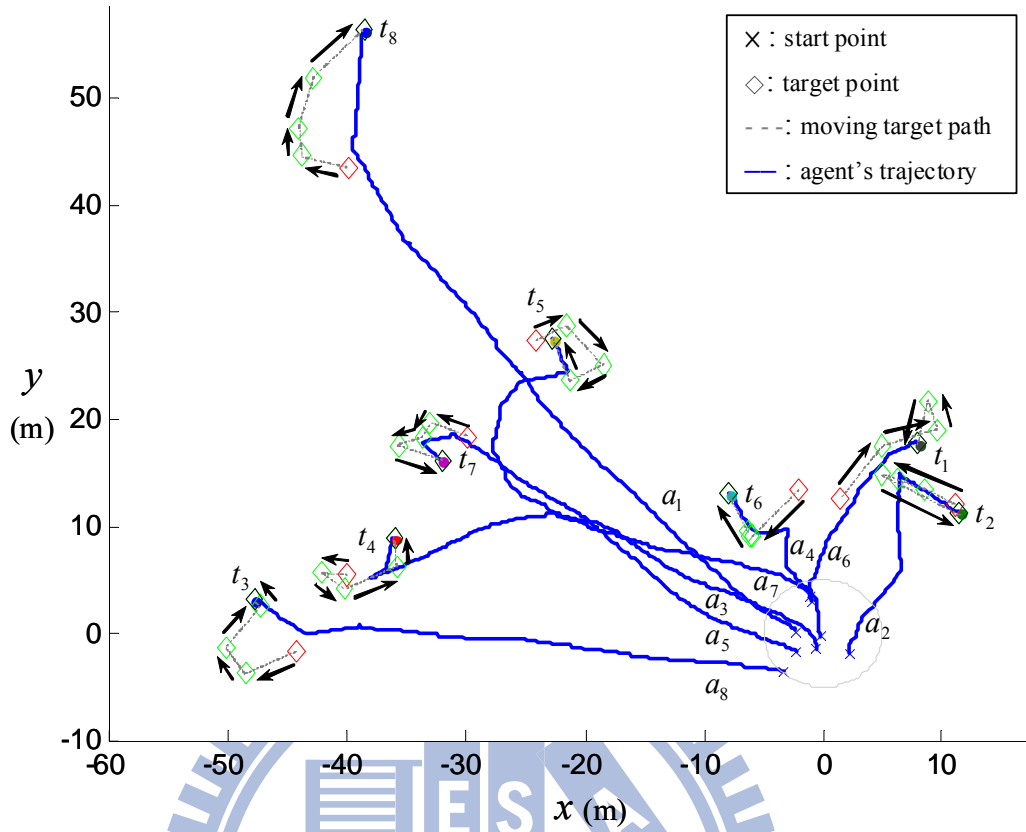


Fig. 2-10. The trajectories of 8 agents with 8 moving targets.

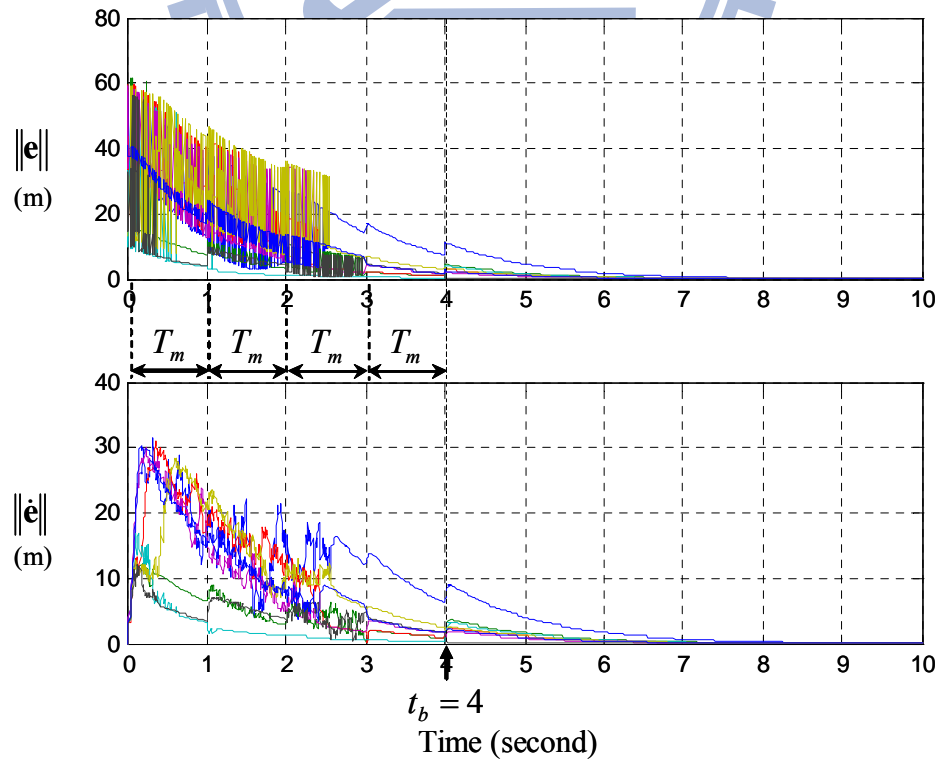


Fig. 2-11. The errors and change of errors of MAS.

As shown in Table 2-1, the shortest and longest total path lengths are both obtained by the exhaustive method. In comparison with the optimal match obtained by the exhaustive method, the total path length of the adaptive SOM-based FNN control method are slightly similar to those of the optimal match by the exhaustive method. However, the benefit for the shortest total path length of the exhaustive method is the loss of computational load if there is large number of random targets which will be indicated in Case 2-4.

Table 2-1 Total path length comparisons for the three cases.

	Optimal match by the exhaustive method (m)	Worst match by the exhaustive method (m)	SOM-based FNN (m)
Case 2-1	210.4025	242.3714	211.0147
Case 2-2	211.4181	741.2715	222.6199
Case 2-3	291.1972	754.3930	310.7617

#### Case 2-4: Large number of random targets

Consider 64 agents and 64 targets in the same 2D workspace shown in Fig. 2-12. The number of targets and their corresponding agents in this case is much larger than those in cases 2-1, 2-2 and 2-3. The total path length of MAS by the proposed adaptive SOM-based FNN control method is calculated as 2168.8 meters which displays a satisfactory tracking performance. Moreover, it also can be seen that the best match is reached after  $t_b = 7.38$  seconds in Fig. 2-13. In this case, it can be seen that the proposed SOM-based FNN controller is capable of effectively handling the best match even if huge number of targets and agents with nonlinear uncertainties is considered.

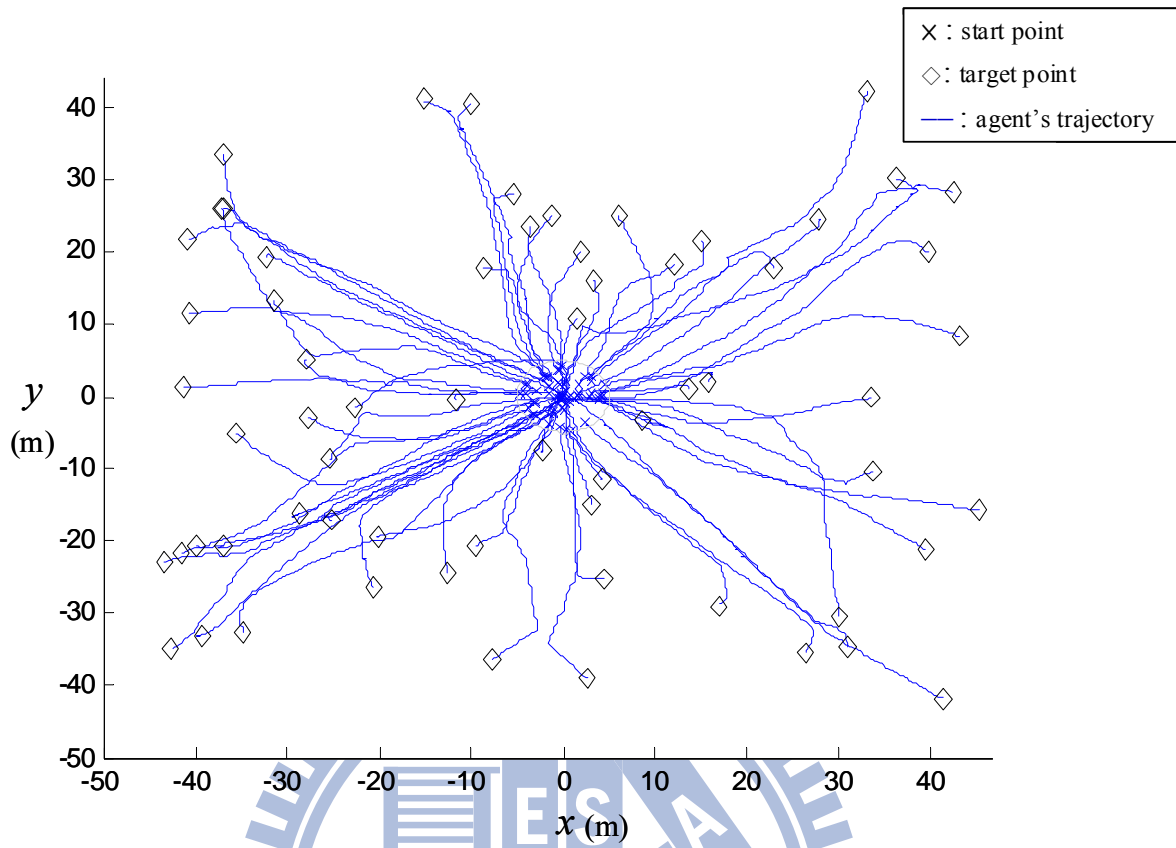


Fig. 2-12. The trajectories of 64 agents with 64 targets.

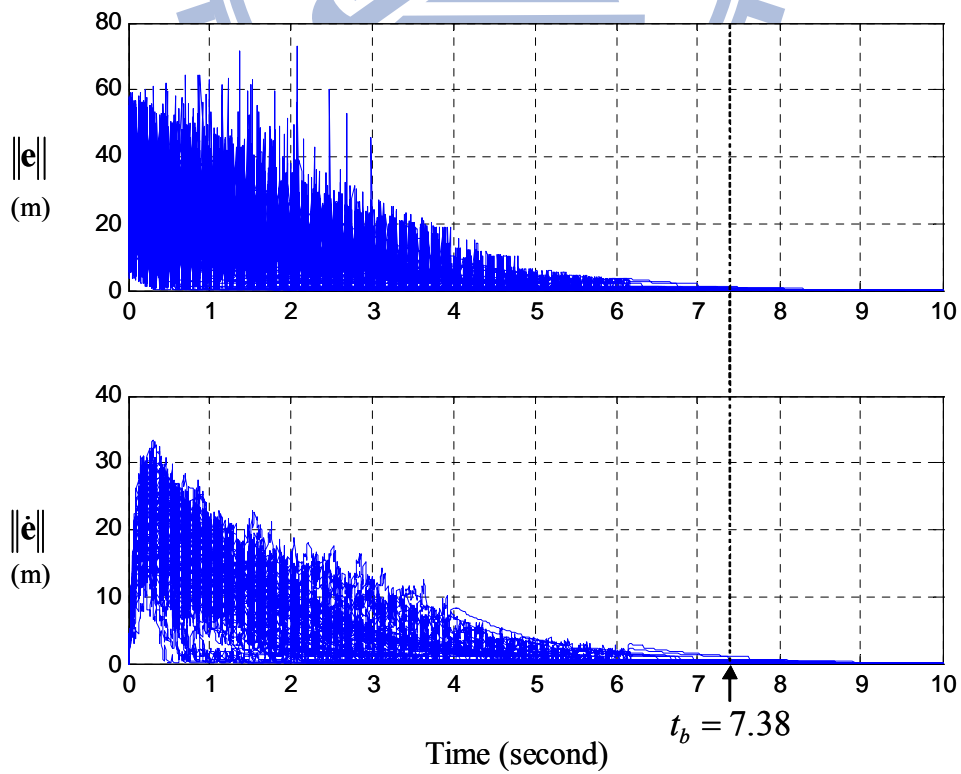


Fig. 2-13. The errors and change of errors of MAS.

From the simulation results, the adaptive SOM-based FNN control method is capable of handling dynamic task assignment of the agents and then smoothly control the agents toward the corresponding targets. Moreover, the computational load in the traditional exhaustive method is  $64!$  ( $= 1.2689 \times 10^{89}$ ) for each iteration in Case 2-4. It is only  $64^2$  ( $= 4096$ ) in our adaptive SOM-based FNN method. It can be seen that the advantage of our new adaptive SOM-based FNN is the tremendous saving of computational load, which is shown in the following Table 2-2.

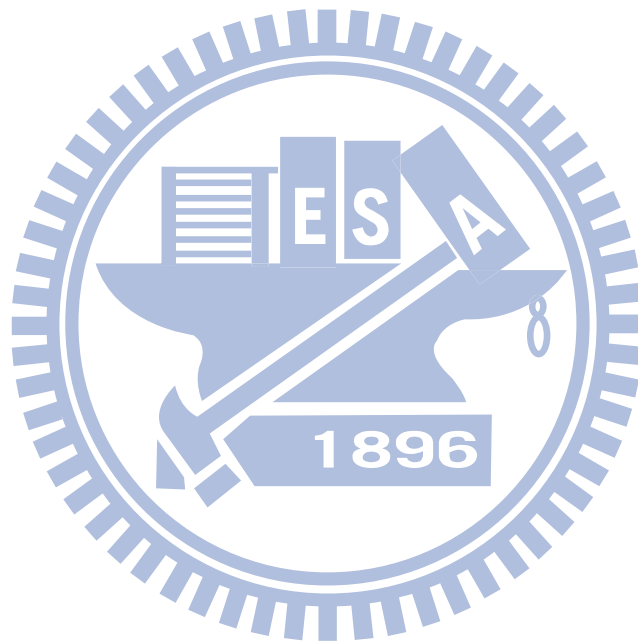
Table 2-2 Comparisons of computational loads in an iteration for all the cases.

	Optimal match by the exhaustive method	Worst match by the exhaustive method	SOM-based FNN
Case 2-1	720	720	36
Case 2-2	720 ~ 40320	720 ~ 40320	36 ~ 64
Case 2-3	40320	40320	64
Case 2-4	$1.2689 \times 10^{89}$	$1.2689 \times 10^{89}$	4096

## 2.7 Conclusions

In this chapter, a SOM-based FNN controller is adopted in the MAS to choose the best-matching pairs between agents and targets and perform path planning using intelligent adaptive methodology. Compared with the simple incremental path planning adopted in the traditional SOM to let the agents move toward the chosen targets, the high nonlinearities and uncertainties of the agents have been considered in this chapter. The intelligent adaptive SOM-based FNN controller is operated in conjunction with the traditional SOM to find the best paths allowing all agents to go to their final targets. The proposed main controller is the FNN controller, in which the fuzzy rule is combined into the neural network, and a new

monitoring controller is also designed to work with FNN controller. This forces the agents to go to their corresponding targets within the constraints of nonlinear dynamics and uncertainties of the agents. It is obvious that the weighting factors are updated via the Lyapunov stability constraints, a process which is very different from the simple update method used by the traditional SOM. From the simulation results, excellent path planning for all agents has been obtained via the intelligent adaptive SOM-based FNN controller.





# Chapter 3

## Toward a New Task Assignment and Path Evolution (TAPE) for Missile Defense System (MDS) using Intelligent Adaptive SOM with Fuzzy Neural Networks

### 3.1 Background and Motivation

In this thesis, we assume that there are limited  $N$  defending missiles and  $D$  incoming missiles in MDS. The  $D$  incoming missiles are launched to attack the limited assets which have their own significances. Once an asset is destroyed by some incoming missiles, it will lose its asset value (or the damaging cost). Because the number of assets that are under attack is unknown, the assignment of  $N$  defending missiles becomes important to minimize the total damaging costs (or maximize the total surviving assets). In the first part of this chapter, a one-to-one agent-target missile guidance law using fuzzy neural network is proposed in comparison with the cerebellar model articulation controller (CMAC) [46], however, the CMAC structure is too complex to be implemented in real-time environment, and the enormous weight space and limited modeling capability in CMAC can be further improved using the proposed FNN controller with fewer mappings and layers. In the second part of this chapter, an adaptive SOM with FNN controller is proposed for multi-agent-multi-target task assignment and missile guidance.

### 3.2 Problem Formulation

In multi-agent system (MAS) with a group of  $N$  agents in the three-dimensional workspace, we assume that the positions and angles of agents  $\mathbf{A} = \{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_N\}$  are initially in a user defined region, and the positions and angles of targets  $\mathbf{T} = \{\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_D\}$  are

initially distributed randomly in the same three-dimensional workspace. The MAS can rapidly and efficiently complete an assigned task via the control inputs  $\mathbf{U} = \{\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_N\}$  for all the agents. In the MAS, it is desired to first perform task assignment by self-organizing map (SOM), after which the path evolution is activated so that all the agents are capable of going to their corresponding targets under the agent dynamics constraints. The architecture of MAS can be extended to missile defense system (MDS), in which the defending interceptors and incoming missiles can be seen as agents and targets, respectively. Furthermore, we consider in the MDS that the assets  $\mathbf{S}$  with different asset values and will be attacked by targets. For any asset  $\mathbf{s}_i$  in  $\mathbf{S}$ , it contains its own asset value  $\bar{V}(\mathbf{s}_i)$  denoted by  $\bar{V}_i$ , which can be regarded as the damaging cost when the asset is attacked and then destroyed. Thus the overall damaging cost by all the targets can be denoted by  $\bar{\mathbf{V}} = \{\bar{V}_1, \bar{V}_2, \dots, \bar{V}_D\}$ . The main control objective is to find the minimal total damaging cost  $\sum_{d=1}^D \bar{V}_d$  to all the assets by SOM, which will be discussed in Section 3.5. After task assignment for all the agents, the fuzzy neural network (FNN) controller is adopted for the agents to intercept the targets. The overall concept proposed in this chapter can be illustrated in the following Fig. 3-1.

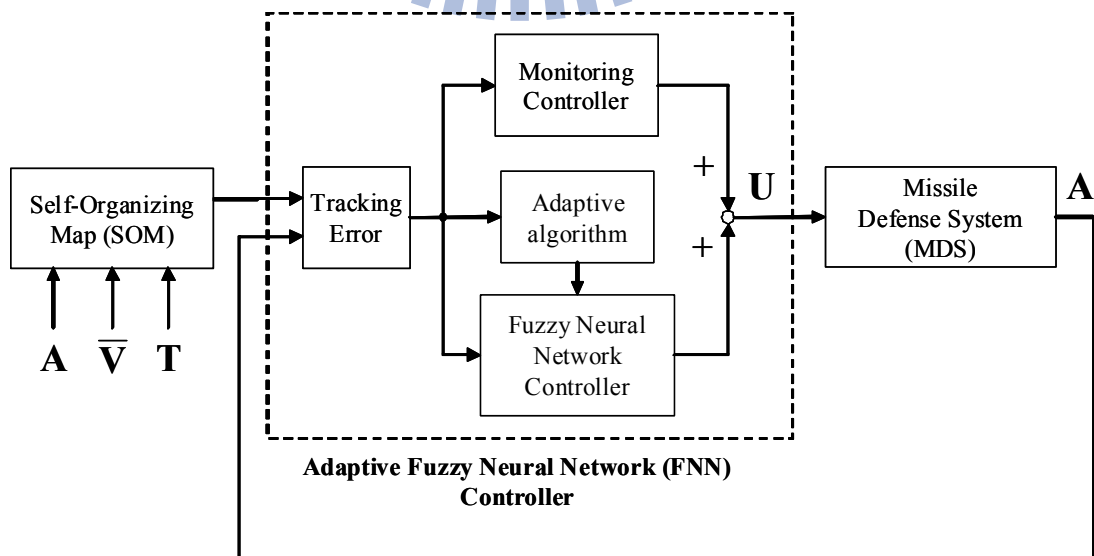


Fig. 3-1. The overall concept of adaptive SOM with FNN controller for MDS.

Before considering multi-agent-multi-target scenarios, a new intelligent algorithm for single agent-target command line-of-sight (CLOS) guidance law will be first proposed in the following sections.

### 3.3 The Three-Dimensional CLOS Guidance Model

The three-dimensional CLOS guidance problem in Fig. 3-2 is a well-known guidance model [44, 46] which can be formulated as a tracking problem for a time-varying nonlinear system. The three-dimensional CLOS guidance model in [44, 46] will be repeated here for convenience. The origin of the inertial frame is located at the ground tracker. The  $Z_I$  axis is vertical upward and the  $X_I$ - $Y_I$  plane is horizontal. The origin of the agent body frame is fixed at the agents' center of mass, with the  $X_A$  axis forward along the agent centerline. The dynamics of all the agents in the inertial frame can be represented [44] as

$$\begin{bmatrix} \ddot{x}_a \\ \ddot{y}_a \\ \ddot{z}_a \end{bmatrix} = \begin{bmatrix} c\theta_a c\psi_a & I_1 & I_2 \\ c\theta_a s\psi_a & I_3 & I_4 \\ s\theta_a & s\phi_{ac} c\theta_a & c\phi_{ac} c\theta_a \end{bmatrix} \begin{bmatrix} a_x \\ a_{yc} \\ a_{zc} \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} \quad (3-1)$$

$$\begin{bmatrix} \dot{\psi}_a \\ \dot{\theta}_a \end{bmatrix} = \begin{bmatrix} \frac{c\phi_{ac}}{v_a c\theta_a} & \frac{s\phi_{ac}}{v_a c\theta_a} \\ \frac{s\phi_{ac}}{v_a} & \frac{c\phi_{ac}}{v_a} \end{bmatrix} \begin{bmatrix} a_{yc} \\ a_{zc} \end{bmatrix} - \begin{bmatrix} 0 \\ \frac{gc\theta_a}{v_a} \end{bmatrix}$$

where

$$I_1 = -s\phi_{ac} s\theta_a c\psi_a - c\phi_{ac} s\psi_a, \quad I_2 = -c\phi_{ac} s\theta_a c\psi_a + s\phi_{ac} s\psi_a,$$

$$I_3 = -s\phi_{ac} s\theta_a s\psi_a + c\phi_{ac} c\psi_a, \quad \text{and} \quad I_4 = -c\phi_{ac} s\theta_a s\psi_a - s\phi_{ac} c\psi_a.$$

A tracking error is defined in order to convert the CLOS guidance problem into a tracking problem. The CLOS guidance involves guiding the agent along the line-of-sight (LOS) to the target. The LOS frame is shown in Fig. 3-3 in which the origin is located at the ground tracker. The  $X_L$  axis forwards along the LOS to the target, and the  $Y_L$  axis is horizontal to the left of the  $X_L$ - $Y_L$  plane. Then, the coordinates  $(R_p, e_1, e_2)$  indicated in Fig. 3-3 represent the agent

position in the LOS frame, and they are related to  $(x_a, y_a, z_a)$  through rotations as follows:

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} -s\sigma_t & c\sigma_t & 0 \\ -s\gamma_t c\sigma_t & -s\gamma_t s\sigma_t & c\gamma_t \end{bmatrix} \begin{bmatrix} x_a \\ y_a \\ z_a \end{bmatrix}. \quad (3-2)$$

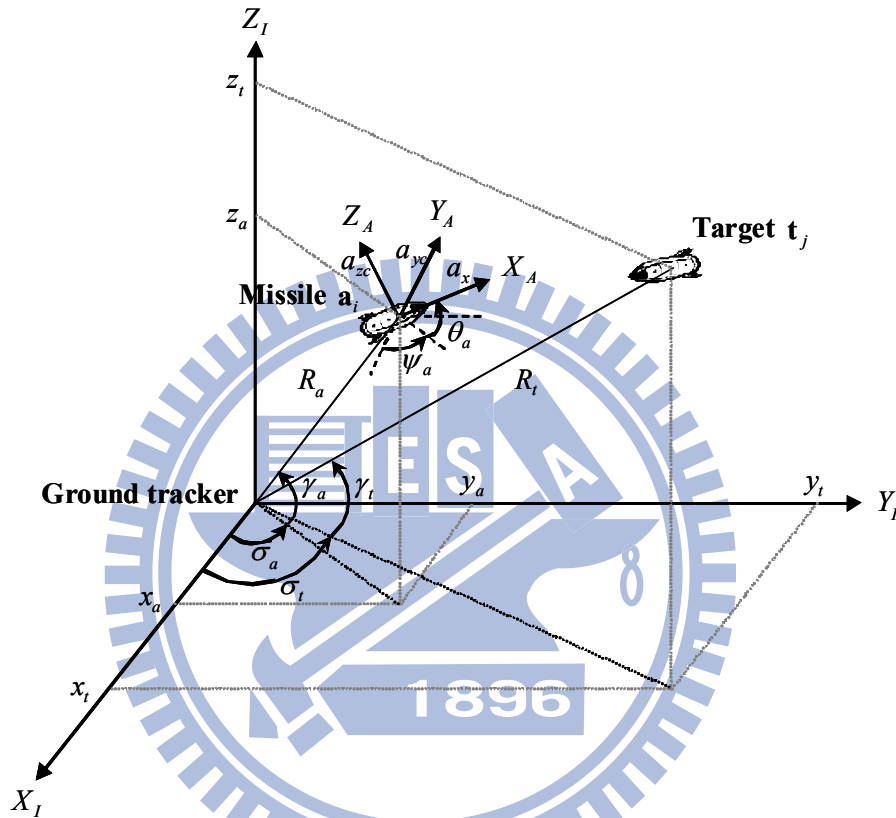


Fig. 3-2. Three-dimensional agent-target pursuit diagram [44, 46].

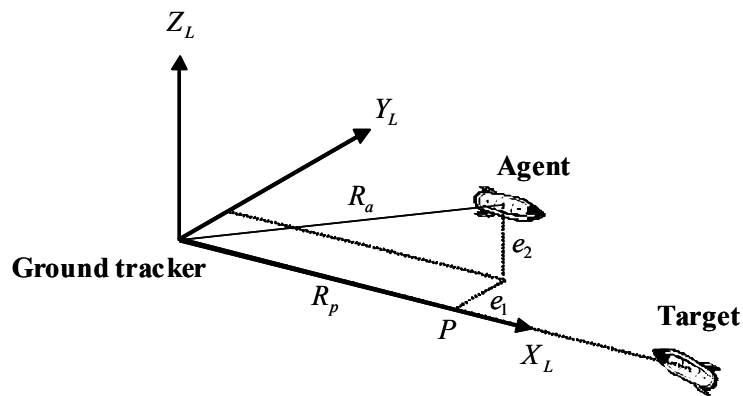


Fig. 3-3. Definition of tracking error [44, 46].

The tracking error is defined as  $\mathbf{e} = [e_1, e_2]^T$ . Since  $e_1$  and  $e_2$  can not be measured directly, these quantities must be computed indirectly using the polar position data of the agent available from the ground tracker as

$$\mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} R_a c(\Delta\gamma + \gamma_t) s \Delta\sigma \\ R_a s(\Delta\gamma + \gamma_t) c \gamma_t - R_a c(\Delta\gamma + \gamma_t) s \gamma_t c \Delta\sigma \end{bmatrix}. \quad (3-3)$$

Note that  $\|\mathbf{e}\|$  represents the distance from the agent to the LOS. Therefore, the agent will eventually hit the target if the tracking error is driven to zero before the target crosses the agent. The three-dimensional CLOS guidance problem has been formulated a tracking problem. Define

$$\begin{aligned} \mathbf{x} &= [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6 \ x_7 \ x_8]^T \\ &= [x_a \ y_a \ z_a \ \dot{x}_a \ \dot{y}_a \ \dot{z}_a \ \psi_a \ \theta_a]^T \\ \mathbf{u} &= [u_1 \ u_2]^T = [a_{yc} \ a_{zc}]^T. \end{aligned} \quad (3-4)$$

Using the previous notations, (3-1), (3-2), and (3-4) can be put into the following state-space form:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{f}(\mathbf{x}, t) + \sum_{i=1}^2 \mathbf{g}_i(\mathbf{x}) \mathbf{u}_i \\ \mathbf{e} &= \mathbf{h}(\mathbf{x}, t) \end{aligned} \quad (3-5)$$

where

$$\mathbf{f}(\mathbf{x}, t) = \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ a_x(t) c x_7 c x_8 \\ a_x(t) s x_7 c x_8 \\ a_x(t) s x_8 - g \\ 0 \\ g c x_8 \\ -\frac{0}{\sqrt{x_4^2 + x_5^2 + x_6^2}} \end{bmatrix}, \quad \mathbf{g}_1(\mathbf{x}) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -s \phi_{ac} s x_8 c x_7 - c \phi_{ac} s x_7 \\ -s \phi_{ac} s x_8 s x_7 + c \phi_{ac} c x_7 \\ s \phi_{ac} c x_8 \\ \frac{c \phi_{ac}}{\sqrt{x_4^2 + x_5^2 + x_6^2}} \\ s \phi_{ac} \\ \frac{s \phi_{ac}}{\sqrt{x_4^2 + x_5^2 + x_6^2}} \end{bmatrix},$$

$$\mathbf{g}_2(\mathbf{x}) = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -c\phi_{ac}sx_8cx_7 + s\phi_{ac}sx_7 \\ -c\phi_{ac}sx_8sx_7 - s\phi_{ac}cx_7 \\ \frac{c\phi_{ac}cx_8}{s\phi_{ac}} \\ \frac{\sqrt{x_4^2 + x_5^2 + x_6^2}cx_8}{c\phi_{ac}} \\ \frac{\sqrt{x_4^2 + x_5^2 + x_6^2}}{c\phi_{ac}} \end{bmatrix}, \text{ and } \mathbf{h}(\mathbf{x}, t) = \begin{bmatrix} h_1(\mathbf{x}, t) \\ h_2(\mathbf{x}, t) \end{bmatrix} = \begin{bmatrix} -x_1s\sigma_t + x_2c\sigma_t \\ -x_1s\gamma_t c\sigma_t - x_2s\gamma_t s\sigma_t + x_3c\gamma_t \end{bmatrix}.$$

The objective of CLOS guidance control is to find a control law to drive the tracking error  $\mathbf{e}(t)$  to zero. For the system shown in (3-5), define the vector fields  $X_j, j = 0, 1, 2$  by

$$X_0 = \frac{\partial}{\partial t} + \sum_{i=1}^8 f_i(\mathbf{x}, t) \frac{\partial}{\partial x_i} \quad (3-6)$$

$$X_j = \sum_{i=1}^8 g_{j,i}(\mathbf{x}) \frac{\partial}{\partial x_i}$$

where  $f_i(\mathbf{x}, t)$ ,  $g_{j,i}(\mathbf{x})$  and  $x_i$  are the  $i^{\text{th}}$  components of  $\mathbf{f}(\mathbf{x}, t)$ ,  $\mathbf{g}_j(\mathbf{x})$  and  $\mathbf{x}$ , respectively [44, 46].

Direct computation yields

$$\begin{aligned} X_0 h_1 &= -\dot{\sigma}_t x_1 c\sigma_t - \dot{\sigma}_t x_2 s\sigma_t - x_4 s\sigma_t + x_5 c\sigma_t \\ &= -\dot{\sigma}_t R_p c\gamma_t + \dot{\sigma}_t e_2 - x_4 s\sigma_t + x_5 c\sigma_t \\ X_1 X_0 h_1 &= -s\phi_{ac}sx_8s(x_7 - \sigma_t) + c\phi_{ac}c(x_7 - \sigma_t) \\ X_2 X_0 h_1 &= -c\phi_{ac}sx_8s(x_7 - \sigma_t) - s\phi_{ac}c(x_7 - \sigma_t) \\ X_0 h_2 &= (\dot{\sigma}_t s\sigma_t s\gamma_t - \dot{\gamma}_t c\gamma_t c\sigma_t)x_1 - (\dot{\gamma}_t c\gamma_t s\sigma_t + \dot{\sigma}_t c\sigma_t s\gamma_t)x_2 \\ &\quad - \dot{\gamma}_t x_3 s\gamma_t - x_4 c\sigma_t s\gamma_t - x_5 s\sigma_t s\gamma_t + x_6 c\gamma_t \\ &= -\dot{\sigma}_t e_1 s\gamma_t - \dot{\gamma}_t R_p - x_4 c\sigma_t s\gamma_t - x_5 s\sigma_t s\gamma_t + x_6 c\gamma_t \\ X_1 X_0 h_2 &= \{c\gamma_t cx_8 + s\gamma_t sx_8 c(x_7 - \sigma_t)\}s\phi_{ac} + c\phi_{ac}s\gamma_t s(x_7 - \sigma_t) \\ X_2 X_0 h_2 &= \{c\gamma_t cx_8 + s\gamma_t sx_8 c(x_7 - \sigma_t)\}c\phi_{ac} - s\phi_{ac}s\gamma_t s(x_7 - \sigma_t) \\ X_0^2 h_1 &= (2\dot{\sigma}_t \dot{\gamma}_t s\gamma_t - \ddot{\sigma}_t c\gamma_t)R_p + \dot{\sigma}_t^2 e_1 + (2\dot{\sigma}_t \dot{\gamma}_t c\gamma_t + \ddot{\sigma}_t s\gamma_t)e_2 \\ &\quad - 2\dot{\sigma}_t \dot{R}_p c\gamma_t + 2\dot{\sigma}_t \dot{e}_2 s\gamma_t + a_x(t)cx_8s(x_7 - \sigma_t) \\ X_0^2 h_2 &= -(\ddot{\gamma}_t + \dot{\sigma}_t^2 s\gamma_t c\gamma_t)R_p - \ddot{\sigma}_t e_1 s\gamma_t + (\dot{\sigma}_t^2 |s\gamma_t|^2 + \dot{\gamma}_t^2)e_2 \\ &\quad - 2\dot{\gamma}_t \dot{R}_p - 2\dot{\sigma}_t \dot{e}_1 s\gamma_t + \{sx_8 c\gamma_t - cx_8 s\gamma_t c(x_7 - \sigma_t)\}a_x(t) \\ &\quad - g c\gamma_t \\ \dot{R}_p &= -(\dot{\sigma}_t s\sigma_t c\gamma_t - \dot{\gamma}_t s\gamma_t c\sigma_t)x_1 - (\dot{\gamma}_t s\gamma_t s\sigma_t - \dot{\sigma}_t c\sigma_t c\gamma_t)x_2 \\ &\quad + \dot{\gamma}_t x_3 c\gamma_t + x_4 c\gamma_t c\sigma_t + x_5 c\gamma_t s\sigma_t + x_6 s\gamma_t \end{aligned} \quad (3-7)$$

After manipulations, the tracking error in (3-5) can be shown concisely into the following form:

$$\begin{aligned} \begin{bmatrix} \ddot{e}_1 \\ \ddot{e}_2 \end{bmatrix} &= \begin{bmatrix} F_1(\mathbf{x}, t) \\ F_2(\mathbf{x}, t) \end{bmatrix} + \begin{bmatrix} G_{11}(\mathbf{x}, t) & G_{12}(\mathbf{x}, t) \\ G_{21}(\mathbf{x}, t) & G_{22}(\mathbf{x}, t) \end{bmatrix} \mathbf{u}(t) \\ &= \mathbf{F}(\mathbf{x}, t) + \mathbf{G}(\mathbf{x}, t) \mathbf{u}(t) \end{aligned} \quad (3-8)$$

where  $F_1(\mathbf{x}, t) = X_0^2 h_1$ ,  $F_2(\mathbf{x}, t) = X_0^2 h_2$ ,  $G_{11}(\mathbf{x}, t) = X_1 X_0 h_1$ ,  $G_{12}(\mathbf{x}, t) = X_2 X_0 h_1$ ,  $G_{21}(\mathbf{x}, t) = X_1 X_0 h_2$ , and  $G_{22}(\mathbf{x}, t) = X_2 X_0 h_2$ .

### 3.4 One-To-One Agent-Target Path Evolution using FNN

A new intelligent FNN controller to realize the single agent-target command line-of-sight (CLOS) guidance law will be discussed in this section. In comparison with the cerebellar model articulation controller (CMAC) structure in [46], the proposed FNN controller is with fewer mappings and layers and the enormous weight space and limited modeling capability in CMAC can be improved using FNN. The tracking error obtained in (3-8) can be further formed as the tracking error vector and be input to the input layer of FNN. The output in the output layer of FNN is adopted as the main controller to the MAS to evolves the positions of the winner targets to their corresponding desired targets. Assuming all the system dynamics are well known and that there exists an ideal controller for a single agent based on the feedback linearization control design, we then arrive from (3-8):

$$\mathbf{u}_{id} = \mathbf{G}^{-1}(\mathbf{x}, t) [-\mathbf{F}(\mathbf{x}, t) - \mathbf{K}_1 \dot{\mathbf{e}} - \mathbf{K}_2 \mathbf{e}]. \quad (3-9)$$

Applying (3-9) into (3-8), the following error dynamics for a single agent can be given

$$\ddot{\mathbf{e}} + \mathbf{K}_1 \dot{\mathbf{e}} + \mathbf{K}_2 \mathbf{e} = \mathbf{0}_{2 \times 1} \quad (3-10)$$

where

$$\mathbf{K}_1 = \begin{bmatrix} 0 & 1 \\ -k_{21} & -k_{11} \end{bmatrix}, \quad \mathbf{K}_2 = \begin{bmatrix} 0 & 1 \\ -k_{22} & -k_{12} \end{bmatrix}$$

are Hurwitz matrices by choosing proper  $k_{11}$ ,  $k_{21}$ ,  $k_{12}$  and  $k_{22}$ . However, the ideal controller  $\mathbf{u}_{id}$  is difficult to implement in practice since the system dynamics is highly nonlinear and sometimes unavailable. Therefore, in order to control the output state efficiently, the control law is assumed to take the following form:

$$\mathbf{u} = \mathbf{u}_{fnn} + \mathbf{u}_m \quad (3-11)$$

where  $\mathbf{u}_{fnn}$  is a FNN controller, and  $\mathbf{u}_m$  is a monitoring controller. The FNN control  $\mathbf{u}_{fnn}$  is the main tracking controller used to imitate the ideal controller in (3-9), and the monitoring controller  $\mathbf{u}_m$  is designed to recover the residual approximation error. The monitoring controller, which is similar to a hitting controller in a traditional sliding mode controller, is derived in the sense of Lyapunov theorem to cope with all system uncertainties to guarantee the stability of the system. The control input  $\mathbf{u}$  in (3-11) is used for the input of agent in (3-8). Figure 3-4 illustrates the concept of (3-11) in our new approach. The tracking error vector  $\mathbf{e}_s$  and neural network output  $\mathbf{y}_o$  in Fig. 3-4 will later be defined as the input and output of the FNN controller, respectively. The limiter in Fig. 3-4 is the maneuvering limiter of the agent to perform a practical behavior for simulations.

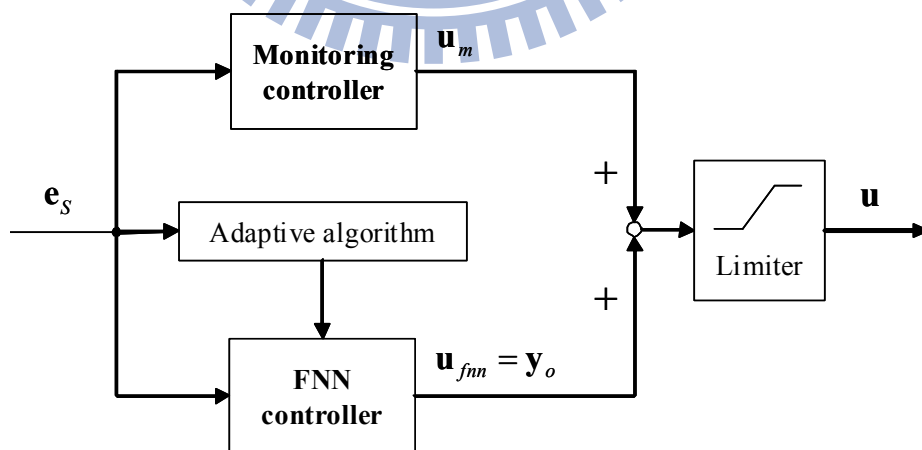


Fig. 3-4. The configuration of  $\mathbf{u}_{fnn}$  and  $\mathbf{u}_m$  for single agent.

The fully linked FNN architecture shown in Fig. 2-3 is also adopted in this section. Repeat



from (2-19) and (2-20), the FNN output can be presented as

$$y_o = \sum_{p=1}^P w_{po}^4 \zeta_p(\bar{x}_k, m_{kh}, v_{kh}) = \prod_{h=1}^H \exp(-(\bar{x}_k - m_{kh})^2 / (v_{kh})^2) \quad (3-22)$$

where  $\bar{x}_k$  denotes the  $k^{\text{th}}$  input to the node of input layer for distinguishing the state variables of agent  $\mathbf{x}$  in MAS. The above (3-22) represents the firing weight of the  $p^{\text{th}}$  neuron in the rule layer. For simplicity, the following  $\mathbf{m}$  and  $\mathbf{v}$  vectors are defined to collect all parameters in the hidden layer of Fig. 2-3 given as

$$\mathbf{m} = [m_{11} \cdots m_{K1} \quad m_{12} \cdots m_{K2} \quad \cdots \quad m_{1H} \cdots m_{KH}]^T \quad (3-23)$$

$$\mathbf{v} = [v_{11} \cdots v_{K1} \quad v_{12} \cdots v_{K2} \quad \cdots \quad v_{1H} \cdots v_{KH}]^T \quad (3-24)$$

Then, the output of the FNN can be represented in a vector form as

$$\mathbf{y}_o = \mathbf{w}_e^T \boldsymbol{\zeta}(\bar{\mathbf{x}}, \mathbf{m}, \mathbf{v}) \quad (3-25)$$

where  $\mathbf{y}_o = \bar{y}_o^4$ ,  $\mathbf{w}_e = [w_{e,1} \quad w_{e,2} \quad \cdots \quad w_{e,H}]^T$ , and  $\boldsymbol{\zeta} = [\zeta_1 \quad \zeta_2 \quad \cdots \quad \zeta_H]^T$ . By the universal approximation theorem, there exists an ideal  $\mathbf{y}_o^* = \mathbf{y}_o(\bar{\mathbf{x}}, \mathbf{w}_e^*, \mathbf{m}^*, \mathbf{v}^*)$  such that [40, 41]

$$\mathbf{y}_o = \mathbf{y}_o^* + \mathbf{E} = \mathbf{w}_e^{*T} \boldsymbol{\zeta}(\bar{\mathbf{x}}, \mathbf{m}^*, \mathbf{v}^*) + \mathbf{E} \quad (3-26)$$

where  $\mathbf{E}$  denotes the approximation error and  $\mathbf{w}_e^*$ ,  $\mathbf{m}^*$ , and  $\mathbf{v}^*$  are the optimal parameter vectors of  $\mathbf{w}_e$ ,  $\mathbf{m}$ , and  $\mathbf{v}$ , respectively. In fact, the optimal parameter vectors needed to best approximate a given nonlinear function are difficult to determine. Thus, an estimate function is defined as

$$\hat{\mathbf{y}}_o = \mathbf{y}_o(\bar{\mathbf{x}}, \hat{\mathbf{w}}_e, \hat{\mathbf{m}}, \hat{\mathbf{v}}) = \hat{\mathbf{w}}_e^T \boldsymbol{\zeta}(\bar{\mathbf{x}}, \hat{\mathbf{m}}, \hat{\mathbf{v}}) \quad (3-27)$$

where  $\hat{\mathbf{w}}_e$ ,  $\hat{\mathbf{m}}$ , and  $\hat{\mathbf{v}}$  are the estimates of  $\mathbf{w}_e^*$ ,  $\mathbf{m}^*$ , and  $\mathbf{v}^*$ , respectively. For notational convenience, we denote  $\boldsymbol{\zeta}^* = \boldsymbol{\zeta}(\bar{\mathbf{x}}, \mathbf{m}^*, \mathbf{v}^*)$  and  $\hat{\boldsymbol{\zeta}} = \boldsymbol{\zeta}(\bar{\mathbf{x}}, \hat{\mathbf{m}}, \hat{\mathbf{v}})$ . Then, we define the estimation error as

$$\begin{aligned}
\tilde{\mathbf{y}}_o &= \mathbf{y}_o - \hat{\mathbf{y}}_o \\
&= \mathbf{y}_o^* - \hat{\mathbf{y}}_o + \mathbf{E} \\
&= \mathbf{w}_e^{*T} \boldsymbol{\zeta}^* - \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \mathbf{E} \\
&= (\hat{\mathbf{w}}_e^T + \tilde{\mathbf{w}}_e^T)(\hat{\boldsymbol{\zeta}} + \tilde{\boldsymbol{\zeta}}) - \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \mathbf{E} \\
&= \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \hat{\mathbf{w}}_e^T \tilde{\boldsymbol{\zeta}} + \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \tilde{\mathbf{w}}_e^T \tilde{\boldsymbol{\zeta}} - \hat{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \mathbf{E} \\
&= \hat{\mathbf{w}}_e^T \tilde{\boldsymbol{\zeta}} + \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \tilde{\mathbf{w}}_e^T \tilde{\boldsymbol{\zeta}} + \mathbf{E}
\end{aligned} \tag{3-28}$$

where  $\tilde{\mathbf{w}}_e = \mathbf{w}_e^* - \hat{\mathbf{w}}_e$  and  $\tilde{\boldsymbol{\zeta}} = \boldsymbol{\zeta}^* - \hat{\boldsymbol{\zeta}}$ . In the following, some tuning laws are derived to on-line tune the parameters of the FNN to achieve favorable estimation. To achieve this goal, we use the linearization technique to transform the nonlinear Gaussian functions into partially linear form so that the Lyapunov theorem extension can be applied [40] as follows

$$\tilde{\boldsymbol{\zeta}} = \begin{bmatrix} \tilde{\zeta}_1 \\ \vdots \\ \tilde{\zeta}_H \end{bmatrix} = \begin{bmatrix} \frac{\partial \zeta_1}{\partial \mathbf{m}} \\ \vdots \\ \frac{\partial \zeta_H}{\partial \mathbf{m}} \end{bmatrix}_{\mathbf{m}=\hat{\mathbf{m}}} \tilde{\mathbf{m}} + \begin{bmatrix} \frac{\partial \zeta_1}{\partial \mathbf{v}} \\ \vdots \\ \frac{\partial \zeta_H}{\partial \mathbf{v}} \end{bmatrix}_{\mathbf{v}=\hat{\mathbf{v}}} \tilde{\mathbf{v}} + \mathbf{H} \equiv \boldsymbol{\zeta}_m^T \tilde{\mathbf{m}} + \boldsymbol{\zeta}_v^T \tilde{\mathbf{v}} + \mathbf{H} \tag{3-29}$$

where  $\tilde{\mathbf{m}} = \mathbf{m}^* - \hat{\mathbf{m}}$ ,  $\tilde{\mathbf{v}} = \mathbf{v}^* - \hat{\mathbf{v}}$ ,  $\boldsymbol{\zeta}_m = \begin{bmatrix} \frac{\partial \zeta_1}{\partial \mathbf{m}} & \dots & \frac{\partial \zeta_H}{\partial \mathbf{m}} \end{bmatrix}_{\mathbf{m}=\hat{\mathbf{m}}}$ ,  $\boldsymbol{\zeta}_v = \begin{bmatrix} \frac{\partial \zeta_1}{\partial \mathbf{v}} & \dots & \frac{\partial \zeta_H}{\partial \mathbf{v}} \end{bmatrix}_{\mathbf{v}=\hat{\mathbf{v}}}$ ,

and  $\mathbf{H}$  is the higher-order term, and  $\frac{\partial \zeta_h}{\partial \mathbf{m}}$  and  $\frac{\partial \zeta_h}{\partial \mathbf{v}}$  are defined respectively as

$$\begin{bmatrix} \frac{\partial \zeta_h}{\partial \mathbf{m}} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{(h-1) \times k} & \frac{\partial \zeta_h}{\partial m_{1h}} & \dots & \frac{\partial \zeta_h}{\partial m_{kh}} & \mathbf{0}_{(H-h) \times k} \end{bmatrix} \tag{3-30}$$

$$\begin{bmatrix} \frac{\partial \zeta_h}{\partial \mathbf{v}} \end{bmatrix} = \begin{bmatrix} \mathbf{0}_{(h-1) \times k} & \frac{\partial \zeta_h}{\partial v_{1h}} & \dots & \frac{\partial \zeta_h}{\partial v_{kh}} & \mathbf{0}_{(H-h) \times k} \end{bmatrix} \tag{3-31}$$

Substituting (3-29) into (3-28) gives

$$\begin{aligned}
\tilde{\mathbf{y}}_o &= \tilde{\mathbf{w}}_e^T \tilde{\boldsymbol{\zeta}} + \hat{\mathbf{w}}_e^T (\boldsymbol{\zeta}_m^T \tilde{\mathbf{m}} + \boldsymbol{\zeta}_v^T \tilde{\mathbf{v}} + \mathbf{H}) + \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \mathbf{E} \\
&= \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \tilde{\mathbf{m}}^T \boldsymbol{\zeta}_m^T \hat{\mathbf{w}}_e + \tilde{\mathbf{v}}^T \boldsymbol{\zeta}_v^T \hat{\mathbf{w}}_e + \mathbf{d}
\end{aligned} \tag{3-32}$$

where  $\hat{\mathbf{w}}_e^T \boldsymbol{\zeta}_m^T \tilde{\mathbf{m}} = \tilde{\mathbf{m}}^T \boldsymbol{\zeta}_m^T \hat{\mathbf{w}}_e$  and  $\hat{\mathbf{w}}_e^T \boldsymbol{\zeta}_v^T \tilde{\mathbf{v}} = \tilde{\mathbf{v}}^T \boldsymbol{\zeta}_v^T \hat{\mathbf{w}}_e$  are used since they are scalars, and the uncertain term  $\mathbf{d} = \hat{\mathbf{w}}_e^T \mathbf{H} + \tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \mathbf{E}$  is assumed to be bounded in the uncertainty bound,

that is  $\|\mathbf{d}\| \leq \Delta$ . The proposed control system is comprised of an FNN identifier and an optimal controller defined in (3-11), in which  $\mathbf{u}_{fnn}$  is used to mimic the ideal controller  $\mathbf{u}_{id}$ , and the compensation tangent controller  $\mathbf{u}_m$  is used to compensate for the difference between the FNN controller and the ideal controller. Considering a single agent in the state-space form, the tracking error vector defined as

$$\mathbf{e}_s = [\mathbf{e}_1 \quad \dot{\mathbf{e}}_1 \quad \mathbf{e}_2 \quad \dot{\mathbf{e}}_2]^T \quad (3-33)$$

which represents the input vector fed into the input node of FNN controller. Substituting (3-11) into (3-8) and using (3-33), the error dynamic equation becomes

$$\dot{\mathbf{e}}_s = \mathbf{K}\mathbf{e}_s + \mathbf{G}_a(\mathbf{u}_{id} - \mathbf{u}_{fnn} - \mathbf{u}_m) = \mathbf{K}\mathbf{e}_s + \mathbf{G}_a(\tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \tilde{\mathbf{m}}^T \zeta_m \hat{\mathbf{w}}_e + \tilde{\mathbf{v}}^T \zeta_v \hat{\mathbf{w}}_e + \mathbf{d} - \mathbf{u}_m) \quad (3-34)$$

where  $\mathbf{K} = \begin{bmatrix} \mathbf{K}_1 & \mathbf{0}_{2 \times 2} \\ \mathbf{0}_{2 \times 2} & \mathbf{K}_2 \end{bmatrix}$  and  $\mathbf{G}_a = \begin{bmatrix} 0 & G_{11} & 0 & G_{21} \\ 0 & G_{12} & 0 & G_{22} \end{bmatrix}^T$ . Since  $\mathbf{K}$  is also a Hurwitz matrix, given a symmetric positive-definite matrix  $\mathbf{Q} \in \mathcal{R}^{2 \times 2}$ , there exists a symmetric positive-definite matrix  $\mathbf{P} \in \mathcal{R}^{2 \times 2}$ , such that the following Lyapunov equation [39, 42]

$$\mathbf{K}^T \mathbf{P} + \mathbf{P} \mathbf{K} = -\mathbf{Q} \quad (3-35)$$

is satisfied.

**Theorem 3-1:** Consider the nonlinear dynamic system represented by (3-1) with the control law in (3-11), where the FNN identifier is designed as (3-27). Then, the weighting vectors  $\hat{\mathbf{w}}_e$ ,  $\hat{\mathbf{m}}$ , and  $\hat{\mathbf{v}}$  will remain bounded, and the performance errors will approach zero. The parameters are updated by the following learning rules:

$$\dot{\hat{\mathbf{w}}}_e = -\dot{\tilde{\mathbf{w}}}_e = \eta_w \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \hat{\boldsymbol{\zeta}} \quad (3-36)$$

$$\dot{\hat{\mathbf{m}}} = -\dot{\tilde{\mathbf{m}}} = \eta_m \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \zeta_m \hat{\mathbf{w}}_e \quad (3-37)$$

$$\dot{\hat{\mathbf{v}}} = -\dot{\tilde{\mathbf{v}}} = \eta_v \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \zeta_v \hat{\mathbf{w}}_e \quad (3-38)$$

$$\mathbf{u}_m = \Delta \tanh(\mathbf{e}_s^T \mathbf{P} \mathbf{G}_a) \quad (3-39)$$

where  $\eta_w$ ,  $\eta_m$ , and  $\eta_v$  are the positive real values. Then the stability of the FNN control system can be guaranteed.

*Proof:*

Let the Lyapunov-like function candidate be

$$V = \frac{1}{2} \mathbf{e}_s^T \mathbf{P} \mathbf{e}_s + \frac{1}{2\eta_w} \tilde{\mathbf{w}}_e^T \tilde{\mathbf{w}}_e + \frac{1}{2\eta_m} \tilde{\mathbf{m}}^T \tilde{\mathbf{m}} + \frac{1}{2\eta_v} \tilde{\mathbf{v}}^T \tilde{\mathbf{v}}. \quad (3-40)$$

Taking the derivative of  $V$  in (3-40) with respect to time and using (3-34) and (3-35), yields

$$\begin{aligned} \dot{V} &= \frac{1}{2} \dot{\mathbf{e}}_s^T \mathbf{P} \mathbf{e}_s + \frac{1}{2} \mathbf{e}_s^T \mathbf{P} \dot{\mathbf{e}}_s + \frac{1}{\eta_w} \tilde{\mathbf{w}}_e^T \dot{\tilde{\mathbf{w}}}_e + \frac{1}{\eta_m} \tilde{\mathbf{m}}^T \dot{\tilde{\mathbf{m}}} + \frac{1}{\eta_v} \tilde{\mathbf{v}}^T \dot{\tilde{\mathbf{v}}} \\ &= \frac{1}{2} \mathbf{e}_s^T (\mathbf{K}^T \mathbf{P} + \mathbf{P} \mathbf{K} + \mathbf{Q}) \mathbf{e}_s + \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a (\tilde{\mathbf{w}}_e^T \hat{\boldsymbol{\zeta}} + \tilde{\mathbf{m}}^T \zeta_m \hat{\mathbf{w}}_e + \tilde{\mathbf{v}}^T \zeta_v \hat{\mathbf{w}}_e) \\ &\quad + \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a (\mathbf{d} - \mathbf{u}_m) + \frac{1}{\eta_w} \tilde{\mathbf{w}}_e^T \dot{\tilde{\mathbf{w}}}_e + \frac{1}{\eta_m} \tilde{\mathbf{m}}^T \dot{\tilde{\mathbf{m}}} + \frac{1}{\eta_v} \tilde{\mathbf{v}}^T \dot{\tilde{\mathbf{v}}} \\ &= -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \tilde{\mathbf{w}}_e^T (\mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \hat{\boldsymbol{\zeta}} + \frac{1}{\eta_w} \dot{\tilde{\mathbf{w}}}_e) \\ &\quad + \tilde{\mathbf{m}}^T (\mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \zeta_m \hat{\mathbf{w}}_e + \frac{1}{\eta_m} \dot{\tilde{\mathbf{m}}}) + \tilde{\mathbf{v}}^T (\mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \zeta_v \hat{\mathbf{w}}_e + \frac{1}{\eta_v} \dot{\tilde{\mathbf{v}}}) + \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a (\mathbf{d} - \mathbf{u}_m) \end{aligned} \quad (3-41)$$

Substituting the learning rules (3-36)–(3-39) into (3-41), (3-41) becomes

$$\begin{aligned} \dot{V} &= -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a (\mathbf{d} - \mathbf{u}_m) \\ &\leq -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \left| \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \right| \|\mathbf{d}\| - \Delta \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \tanh(\mathbf{e}_s^T \mathbf{P} \mathbf{G}_a) \\ &\leq -\frac{1}{2} \mathbf{e}_s^T \mathbf{Q} \mathbf{e}_s + \Delta \left| \mathbf{e}_s^T \mathbf{P} \mathbf{G}_a \right| \\ &\leq -\frac{1}{2} \|\mathbf{e}_s\| \lambda_{\min}(\mathbf{Q}) \|\mathbf{e}_s\| + \Delta \|\mathbf{e}_s\| \|\mathbf{P}\| \|\mathbf{G}_a\| \\ &\leq -\frac{1}{2} (\lambda_{\min}(\mathbf{Q}) - 1) \|\mathbf{e}_s\|^2 + \|\mathbf{P} \mathbf{G}_a\|^2 \|\Delta\|^2 \end{aligned} \quad (3-42)$$

where  $\lambda_{\min}(\mathbf{Q})$  is the minimum eigenvalue of  $\mathbf{Q}$ . Since  $\lambda_{\min}(\mathbf{Q})$  can be chosen as  $\lambda_{\min}(\mathbf{Q}) > 1$ , then (3-42) reveals that

$$\int_0^t \|\mathbf{e}_s\|^2 dt \leq \frac{1}{\lambda_{\min}(\mathbf{Q}) - 1} (|V(0)| + |V(t)|) + \frac{1}{\lambda_{\min}(\mathbf{Q}) - 1} \|\mathbf{P} \mathbf{G}_a\|^2 \int_0^t \|\Delta\|^2 dt \quad (3-43)$$

for all  $t \geq 0$ . Furthermore, if  $\Delta$  is squared integrable, then from (3-43),  $\mathbf{e}_s \in L_2$  has been proven [39]. In addition, the right hand side of (3-43) is bounded, that is,  $\dot{\mathbf{e}}_s \in L_\infty$ . Using

Barbalat's Lemma [39], we can prove that  $\lim_{t \rightarrow \infty} \|\mathbf{e}_s\| = 0$  when  $\int_0^t \|\Delta\|^2 dt < \infty$ . The stability of the overall approximation scheme is guaranteed based on the above results and the Lyapunov stability theorem. Based on (3-36)–(3-38), the adaptive law of weighting factors in an element form can be obtained. Thus, the Lyapunov stability theorem is guaranteed under the optimal approximation model with no modeling error. **Q.E.D.**

### 3.5 The Design of SOM for Task Assignment of MDS

After the single agent-target control system is constructed, the overall MDS (or MAS) consists of  $(N + D)$  numbers of agent-target matches will be discussed in this section. Suppose that  $D \geq N$ , it takes  $P(D, N) = D!/(D - N)!$  computation steps to find the total distances or damaging cost in traditional exhaustive method. In real-time MDS environment, this pre-computation before the targets are launched is time-consuming. Therefore, the principal goal of SOM is to transform an input pattern of arbitrary dimension into a one- or two-dimensional discrete map as well as to perform this transformation adaptively in a topologically ordered fashion [19, 38]. The SOM is suitable for dealing with task assignment because the dimension of the targets can be simplified, and mapped to the corresponding agents. The overall MAS system can be considered a self-organizing system which can adjust its basic structure when its environment changes. The algorithm of the SOM proceeds first by initializing the synaptic weights in the network, such that it can be done by assigning them in random indexed patterns. Considering a multi-agent-multi-target scenario, the positions and angles of  $i^{\text{th}}$  agent and  $d^{\text{th}}$  target can be further defined as

$$\mathbf{a}_i = [x_{a,i} \quad y_{a,i} \quad z_{a,i} \quad \dot{x}_{a,i} \quad \dot{y}_{a,i} \quad \dot{z}_{a,i} \quad \psi_{a,i} \quad \theta_{a,i}]^T \in \mathbf{A},$$

$$\mathbf{t}_d = [x_{t,d} \quad y_{t,d} \quad z_{t,d} \quad \dot{x}_{t,d} \quad \dot{y}_{t,d} \quad \dot{z}_{t,d} \quad \psi_{t,d} \quad \theta_{t,d}]^T \in \mathbf{T},$$

respectively. The proposed control inputs of the  $i^{\text{th}}$  agent can also be defined as

$$\mathbf{u}_i = [a_{yc,i} \quad a_{zc,i}]^T = [u_{1,i} \quad u_{2,i}]^T \in \mathbf{U}$$

where  $a_{yc,i}$  and  $a_{zc,i}$  are the yaw and pitch acceleration commands of the  $i^{\text{th}}$  agent, respectively.

Define the positions of agents  $\bar{\mathbf{A}} = \{\bar{\mathbf{a}}_1, \bar{\mathbf{a}}_2, \dots, \bar{\mathbf{a}}_N\}$  where  $\bar{\mathbf{a}}_i = [x_{a,i} \quad y_{a,i} \quad z_{a,i}]^T$  is the position of the  $i^{\text{th}}$  agent, and denote the random indexed input vectors chosen from the positions of targets as

$$\mathbf{R} = \{\mathbf{r}_1, \mathbf{r}_2, \dots, \mathbf{r}_d, \dots, \mathbf{r}_D\}$$

where  $\mathbf{r}_d = [x_{t,i} \quad y_{t,i} \quad z_{t,i}]^T \in \bar{\mathbf{T}}$  is the position of  $d^{\text{th}}$  target. Once the positions of agents and targets are initialized, the competitive process of SOM can start to find the winner neurons. In traditional SOM, the winning neuron locates the center of a topological neighborhood of excited neurons. In this thesis, the neighborhood of the winner is neglected since the agents move toward their corresponding targets without any cooperative process. In the traditional competitive process, the total Euclidean distances and total damaging cost have to be considered. Therefore, we first assume that all the asset values are neglected, and the competitive mechanism will choose the Euclidean distance between the  $i^{\text{th}}$  agent and the  $d^{\text{th}}$  target defined as

$$D_{i,d} = \|\mathbf{r}_d - \bar{\mathbf{a}}_i\|. \quad (3-44)$$

In traditional SOM, this Euclidean distance is the parameter for the competitive process. However, the other parameters, like total Euclidean distances in [23], should also be considered with the distance expression. The values of assets in MDS is more important than their corresponding Euclidean distances and the motivation of SOM in this chapter is to minimize the total damaging cost, therefore, a new distance expression from (3-44) with the equitable distribution of workload can be defined as

$$\bar{D}_{i,d} = \left( \frac{\delta + \bar{V}_{total}}{\delta + \bar{V}_d} \right) D_{i,d} \quad (3-45)$$

where

$$\bar{V}_{total} = \sum_{d=1}^D \bar{V}_d$$

is the total value of all the assets;  $\delta$  is the adjustment parameter defined by the user to determine the importance of asset value. The smaller the  $\delta$  is, the more important of asset value is. As shown in Fig. 3-5, the new distance expression constructed by the input of  $\delta$ ,  $\bar{\mathbf{a}}_i$ , and  $\mathbf{r}_d$  forms a new  $N$ -by- $D$  distance matrix

$$\bar{\mathbf{D}} = \begin{bmatrix} \bar{D}_{1,1} & \cdots & \bar{D}_{1,D} \\ \vdots & \ddots & \vdots \\ \bar{D}_{N,1} & \cdots & \bar{D}_{N,D} \end{bmatrix}. \quad (3-46)$$

For some given  $\delta$ , agent, and target as input, the output neurons compete to be the winner according to a specified criterion described as

$$[i_w, i_t] = \min \{ \bar{D}_{i,d}, i = 1, 2, \dots, N; d = 1, 2, \dots, D; \{i, r\} \notin \Omega \} \quad (3-47)$$

where  $[i_w, i_t]$  denotes that the match in which the  $i_t^{\text{th}}$  target from the  $i_w^{\text{th}}$  agent is the winner, and  $\Omega$  is the set of neurons in which the winner has been chosen in an iteration. From (3-47), the  $N$  winners can be found to obtain a new  $\mathbf{W} = \{\mathbf{w}_1, \mathbf{w}_2, \dots, \mathbf{w}_N\}$  which is the re-allocated index of agents  $\mathbf{A}$  that corresponds to the random indexed targets  $\mathbf{R}$  to be used in the adaptive process.

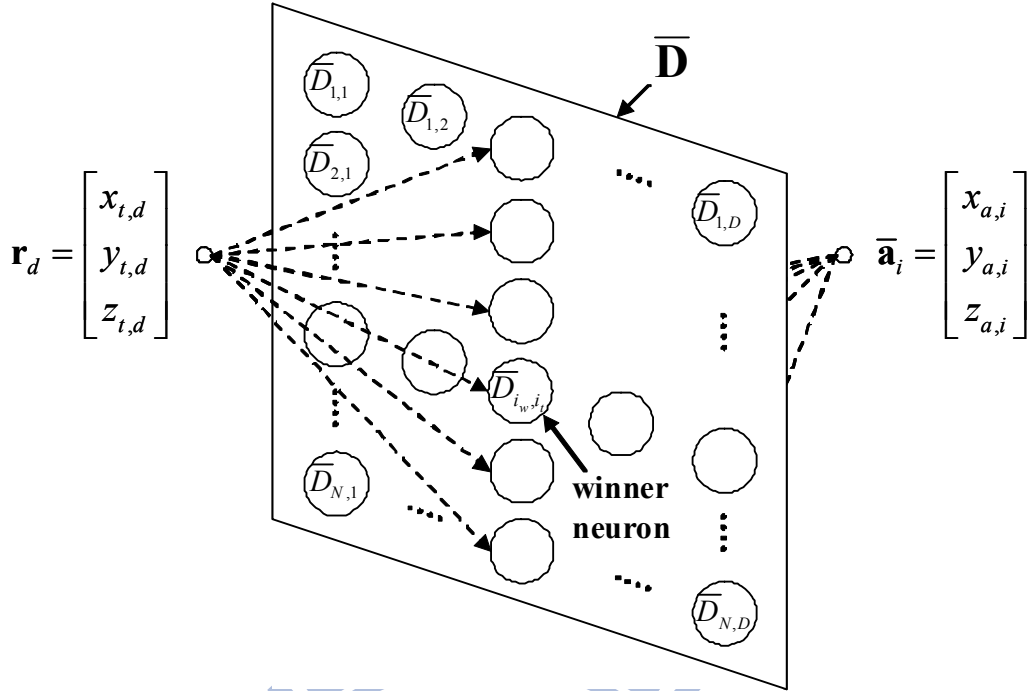


Fig. 3-5. The structure of self-organizing map (SOM).

**Algorithm 3-1:**

- Step 1**  $N$  agents are created in  $\bar{\mathbf{A}}$   
 $D$  targets with random indexed are created in  $\mathbf{R}$   
 $D$  damaging costs are created in  $\bar{\mathbf{V}}$   
calculate  $\bar{V}_{total} = \sum_{d=1}^D \bar{V}_d$   
define the adjustment parameter  $\delta$
- Step 2** for agent  $\bar{\mathbf{a}}_i, i = 1, 2, \dots, N$  in  $\bar{\mathbf{A}}$
- for target  $\mathbf{r}_d, d = 1, 2, \dots, D$  in  $\mathbf{R}$
- calculate Euclidean distance  $D_{i,d} = \|\mathbf{r}_d - \bar{\mathbf{a}}_i\|$
- find the new distance  $\bar{D}_{i,d} = \left( \frac{\delta + \bar{V}_{total}}{\delta + \bar{V}_d} \right) D_{i,d}$
- end
- end
- Step 3** for target  $\mathbf{r}_d, d = 1, 2, \dots, D$  in  $\bar{\mathbf{D}}$  matrix



$i_t$  is  $d$ , find the winner neuron with index  $i_w$   
 $[i_w, i_t]$  is obtained in the  $i_w^{\text{th}}$  row,  $d^{\text{th}}$  column  
 append the index  $i_w$  to the interception list  $\mathbf{L}$

end

**Step 4** agents are dispatched to hit the targets with orders in  $\mathbf{L}$

From the above two processes, the number of computation steps for finding the minimal total damaging cost can be obtained, which is equal to  $N * D$ . In comparison with traditional SOM method, the new adaptive SOM method eliminates the time consuming tuning in neighborhood function and is able to reduce the computational load in the task assignment of MAS. Note that in this thesis, the hit probability of agent is assumed as 100 %. However, the SOM mechanism can also find the minimal total damaging cost even if the leakage of agents is considered. By arranging the winner agents  $\mathbf{W}$ , the interception list  $\mathbf{L}$  can be obtained which is ordered by the index of agent. The list is a useful command or decision for MDS to determine which target should be intercepted by which agent in the future. The last mechanism of SOM is the adaptive process which enables the winner agents  $\mathbf{W}$  to update the positions of the winners.

### **Example 3-1: SOM-based dispatching**

#### **Step 1**

Figure 3-6 shows four steps for SOM example in MDS in which there are the positions of agents  $\bar{\mathbf{A}} = \{\bar{\mathbf{a}}_1, \bar{\mathbf{a}}_2, \bar{\mathbf{a}}_3, \bar{\mathbf{a}}_4\} \subset \mathbf{A}$  ( $N = 4$ ), and the positions of targets with random indexed  $\mathbf{R} = \{\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_4, \mathbf{r}_5, \mathbf{r}_6\}$  ( $D = 6$ ), and three surviving assets  $\mathbf{S} = \{\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3\}$  with their values  $\bar{V}(\mathbf{s}_1) = 1$ ,  $\bar{V}(\mathbf{s}_2) = 2$ , and  $\bar{V}(\mathbf{s}_3) = 3$ . The damaging costs caused from the attacking targets in MDS can be formed as  $\bar{\mathbf{V}} = \{\bar{V}_1, \bar{V}_2, \bar{V}_3, \bar{V}_4, \bar{V}_5, \bar{V}_6\} = \{1, 3, 1, 2, 3, 2\}$  which means  $\mathbf{t}_1$  will attack  $\mathbf{s}_1$ ,  $\mathbf{t}_2$  will attack  $\mathbf{s}_3$ ,  $\mathbf{t}_3$  will attack  $\mathbf{s}_1$ ,  $\mathbf{t}_4$  will attack  $\mathbf{s}_2$ , etc. Before the beginning of attack of targets, the total damaging cost

$$\bar{V}_{total} = \sum_{d=1}^6 \bar{V}_d = 12$$

can be obtained.

### Step 2

In the SOM mechanism, the positions of agents **A** and random indexed targets **R** will first be used to calculate the Euclidean distance as

$$\{D_{i,d} = \|\mathbf{r}_d - \mathbf{a}_i\|, \quad i=1,2,\dots,4; d=1,2,\dots,6\}.$$

If  $\delta$  is chosen as 0.1, the new distance matrix from (3-45) can be obtained for all the agents as:

$$\bar{\mathbf{D}} = \begin{bmatrix} \frac{0.1+12}{0.1+1} D_{1,1} & \frac{0.1+12}{0.1+3} D_{1,2} & \frac{0.1+12}{0.1+1} D_{1,3} & \frac{0.1+12}{0.1+2} D_{1,4} & \frac{0.1+12}{0.1+3} D_{1,5} & \frac{0.1+12}{0.1+2} D_{1,6} \\ \frac{0.1+12}{0.1+1} D_{2,1} & \frac{0.1+12}{0.1+3} D_{2,2} & \frac{0.1+12}{0.1+1} D_{2,3} & \frac{0.1+12}{0.1+2} D_{2,4} & \frac{0.1+12}{0.1+3} D_{2,5} & \frac{0.1+12}{0.1+2} D_{2,6} \\ \frac{0.1+12}{0.1+1} D_{3,1} & \frac{0.1+12}{0.1+3} D_{3,2} & \frac{0.1+12}{0.1+1} D_{3,3} & \frac{0.1+12}{0.1+2} D_{3,4} & \frac{0.1+12}{0.1+3} D_{3,5} & \frac{0.1+12}{0.1+2} D_{3,6} \\ \frac{0.1+12}{0.1+1} D_{4,1} & \frac{0.1+12}{0.1+3} D_{4,2} & \frac{0.1+12}{0.1+1} D_{4,3} & \frac{0.1+12}{0.1+2} D_{4,4} & \frac{0.1+12}{0.1+3} D_{4,5} & \frac{0.1+12}{0.1+2} D_{4,6} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{121}{11} D_{1,1} & \frac{121}{31} D_{1,2} & \frac{121}{11} D_{1,3} & \frac{121}{21} D_{1,4} & \frac{121}{31} D_{1,5} & \frac{121}{21} D_{1,6} \\ \frac{121}{11} D_{2,1} & \frac{121}{31} D_{2,2} & \frac{121}{11} D_{2,3} & \frac{121}{21} D_{2,4} & \frac{121}{31} D_{2,5} & \frac{121}{21} D_{2,6} \\ \frac{121}{11} D_{3,1} & \frac{121}{31} D_{3,2} & \frac{121}{11} D_{3,3} & \frac{121}{21} D_{3,4} & \frac{121}{31} D_{3,5} & \frac{121}{21} D_{3,6} \\ \frac{121}{11} D_{4,1} & \frac{121}{31} D_{4,2} & \frac{121}{11} D_{4,3} & \frac{121}{21} D_{4,4} & \frac{121}{31} D_{4,5} & \frac{121}{21} D_{4,6} \end{bmatrix}$$

Because we are focusing on the task assignment to minimize the damaging cost, we can assume that the distances between the agents and targets are the same and are normalized to one. This implies that

$$\{D_{i,d} = 1, \quad i=1,2,\dots,4; d=1,2,\dots,6\}.$$

Therefore, the minimum for each agent can be found from the following new distance matrix:

$$\bar{\mathbf{D}} \approx \begin{bmatrix} 11 & 3.9 & 11 & 5.76 & 3.9 & 5.76 \\ 11 & 3.9 & 11 & 5.76 & 3.9 & 5.76 \\ 11 & 3.9 & 11 & 5.76 & 3.9 & 5.76 \\ 11 & 3.9 & 11 & 5.76 & 3.9 & 5.76 \end{bmatrix}.$$

### Step 3

In the first row of new distance matrix  $\bar{\mathbf{D}}$ , the winner neuron is  $\bar{D}_{1,2}$  which means the matching pair is  $\{\mathbf{a}_1, \mathbf{t}_2\}$ . Therefore, after repeating from the first row to the fourth row, the winner-target pairs  $\{\mathbf{a}_1, \mathbf{t}_2\}$ ,  $\{\mathbf{a}_2, \mathbf{t}_5\}$ ,  $\{\mathbf{a}_3, \mathbf{t}_4\}$ , and  $\{\mathbf{a}_4, \mathbf{t}_6\}$  can be obtained by using the

competitive process in (3-47) list.

**Step 4**

Picking the indexes of the targets in the matching pairs, the interception list  $L = \{2,5,4,6\}$  can further be constructed as a MDS command which shows that the  $t_2, t_5, t_4,$  and  $t_6$  targets should be intercepted by the  $a_1, a_2, a_3,$  and  $a_4$  agents, respectively. Defended by the agents, all the assets after this attacking wave have the remaining damaging costs  $\bar{V}' = \{1,0,1,0,0,0\}$  and the final total damaging cost becomes  $\bar{V}'_{total} = \sum_{d=1}^6 \bar{V}'_d = 2$  which is the minimal value. In this example, although the Euclidean distances are almost neglected, the situation for the two or more assets have the same value and there exists relatively short Euclidean distance from some agent to its corresponding target should be taken into consider.

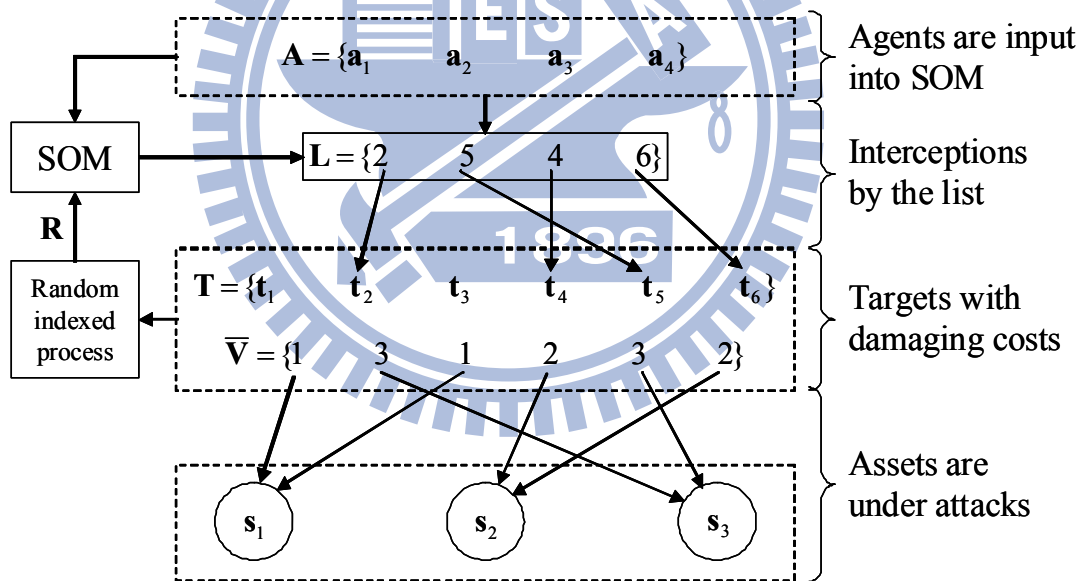


Fig. 3-6. The self-organizing map (SOM) example in MDS.

In traditional exhaustive method to find the minimal total damaging cost, it will take  $6!/(6-4)! (= 360)$  computation steps. However, it only takes at most  $4 * 6 (= 24)$  computation steps by the proposed SOM. Consider  $N$  agents and  $D$  targets in MDS, if we have to find the optimal (or worst) matching pairs under some condition such as the minimal

total paths, the computational load of traditional exhaustive method is  $D!/(D - N)!$ , however, the computational load of the proposed SOM is  $N * D$ . Therefore, as shown in the following Table 3-1, the SOM has relatively smaller computational load than exhaustive method if there are more number of agents and targets.

Table 3-1 Comparisons of computational loads for the task assignment using exhaustive methods and the proposed SOM.

	Optimal (or worst) match by the exhaustive method	SOM
$N = D = 6$	720	36
$N = 6, D = 8$	20160	48
$N = D = 8$	40320	64
$N = 8, D = 64$	$1.7846 \times 10^{14}$	512
$N = D = 64$	$1.2689 \times 10^{89}$	4096

In comparison with incremental adjustment in traditional SOM, the proposed adaptive SOM with the proposed FNN controller can handle the overall TAPE problems in MDS. The closed-loop configuration of SOM with FNN controller for MDS TAPE is shown in Fig. 3-7.

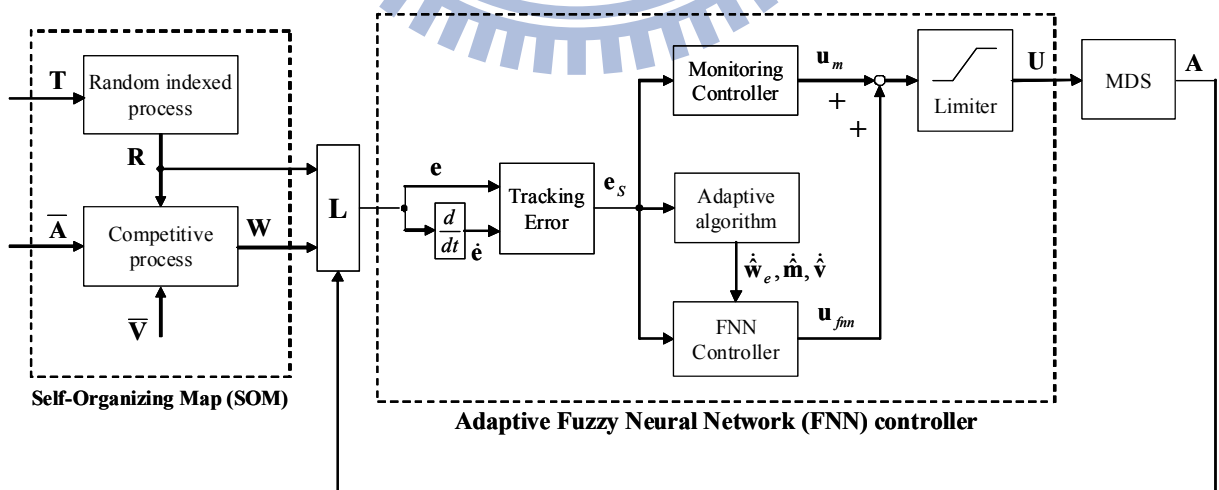


Fig. 3-7. The closed-loop configuration of SOM with FNN controller for MDS TAPE.

### 3.6 Illustrated Examples

In this section, computer simulations are performed to illustrate the effectiveness of the proposed FNN guidance law. In order to assess the performance characteristics in a closed-loop engagement scenario, it is necessary to specify target dynamics. The target motion model is assumed to produce no axial acceleration or roll motion. Then, the simplified dynamics of target motion can be represented in the inertial frame as follows [44]:

$$\begin{cases} \ddot{x}_t = -a_{ty}s\psi_t - a_{tz}s\theta_t c\psi_t \\ \ddot{y}_t = a_{ty}c\psi_t - a_{tz}s\theta_t s\psi_t \\ \ddot{z}_t = a_{ty}c\theta_t - g \\ \dot{\psi}_t = \frac{a_{ty}}{v_t c\theta_t} \\ \dot{\theta}_t = \frac{a_{tz} - gc\theta_t}{v_t} \end{cases} \quad (3-48)$$

where  $a_{ty}$  and  $a_{tz}$  are the  $y$ -axial and  $z$ -axial acceleration of target, respectively. For all the scenarios, assume that the target maneuvers with  $a_{ty} = 0$  g,  $a_{tz} = -1$  g for all the time. To limit the missile's maneuverability, a 30 g ( $g = 9.8$  m/s<sup>2</sup>) maneuvering limiter is considered for simulations. The pitch and yaw autopilot dynamics are chosen as second-order time invariant linear systems and the ground tracker as a simplified differential tracking system with damping ration 0.6 and natural frequency  $6\pi$  rad/s, as shown in Fig. 3-8. The ground tracker provides the estimated values of  $\sigma_t$ ,  $\gamma_t$ ,  $\dot{\sigma}_t$ , and  $\dot{\gamma}_t$ , as well as the measurement data of  $\Delta\sigma$  and  $\Delta\gamma$ . In the following, the estimated value is distinguished from its true value by inserting the symbol  $\hat{\cdot}$  to the corresponding variable. To evaluate the influence of measurement noise, random noises with magnitude between  $\pm 0.3$  degrees are included. The controller parameters in the simulations below are chosen as follows [46]:

$$\mathbf{Q} = \begin{bmatrix} 588 & 73 & 0 & 0 \\ 73 & 16 & 0 & 0 \\ 0 & 0 & 588 & 73 \\ 0 & 0 & 73 & 16 \end{bmatrix}, k_{11} = k_{21} = 18, k_{12} = k_{22} = 51, \eta_w = 5, \text{ and } \eta_m = \eta_v = 0.2.$$

In Case 3-1, the one-to-one agent-target missile guidance laws using CMAC and FNN

controller are discussed to show the capability and efficiency of the proposed FNN controller. Followed by the tow control algorithm comparisons, the proposed SOM is adopted to handle the task assignment for MAS in MDS in Case 2.

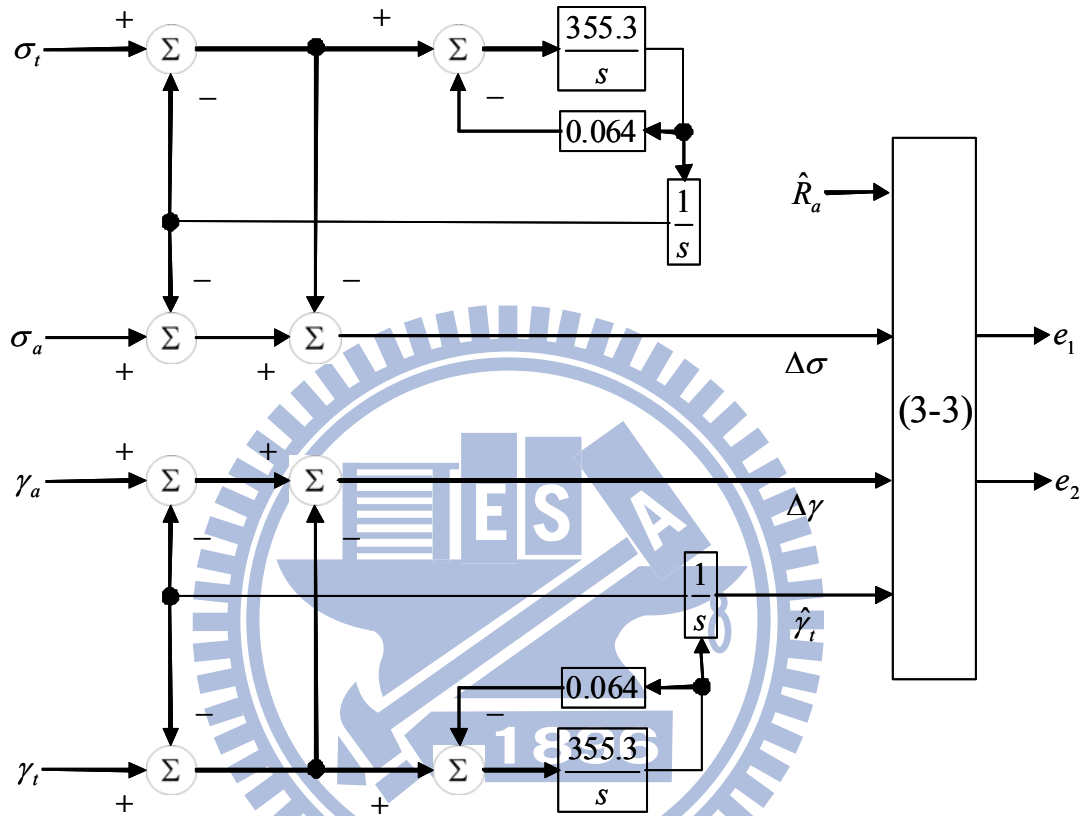


Fig. 3-8. Block diagram representation of estimation algorithm for guidance information [44, 46].

### Case 3-1:

The closed-loop configuration of FNN controller for missile guidance law is shown in Fig. 3-9. For large scale simulation purpose, we have generated 25 initial positions with angles for target in Fig. 3-10, which shows an  $X_I$ - $Y_I$  coordinate system. The subscripted number of target in Fig. 3-10 represents the number of scenario, and the position of target in the  $i^{\text{th}}$  scenario is defined as

$$\begin{aligned} \mathbf{t}_i &: (R_i \cos(30^\circ) \cos(\sigma_{t_i}), R_i \cos(30^\circ) \sin(\sigma_{t_i}), R_i \sin(30^\circ)), \\ \mathbf{t}_{i+12} &: (R_i \cos(60^\circ) \cos(\sigma_{t_i}), R_i \cos(60^\circ) \sin(\sigma_{t_i}), R_i \sin(60^\circ)), \\ \mathbf{t}_{25} &: (0, 0, R_i), \quad i = 1, 2, \dots, 12. \end{aligned} \quad (3-49)$$

In fact, our agent is at (10, 10, 6), which is very close to the ground base in  $X_I$ - $Y_I$ - $Z_I$  coordinate system. There are only three initial positions/angles of target in [4, 6], which were specifically chosen to guarantee the success of convergence. Therefore there are 25 battle scenarios in which  $R_i$  is set as 7000 meters from the ground base, and each initial position and angle of target is chosen from the 25 locations as shown in Fig. 3-10. Once the position of target is chosen, the initial angle of target will also be applied as the heading angle to the ground base.

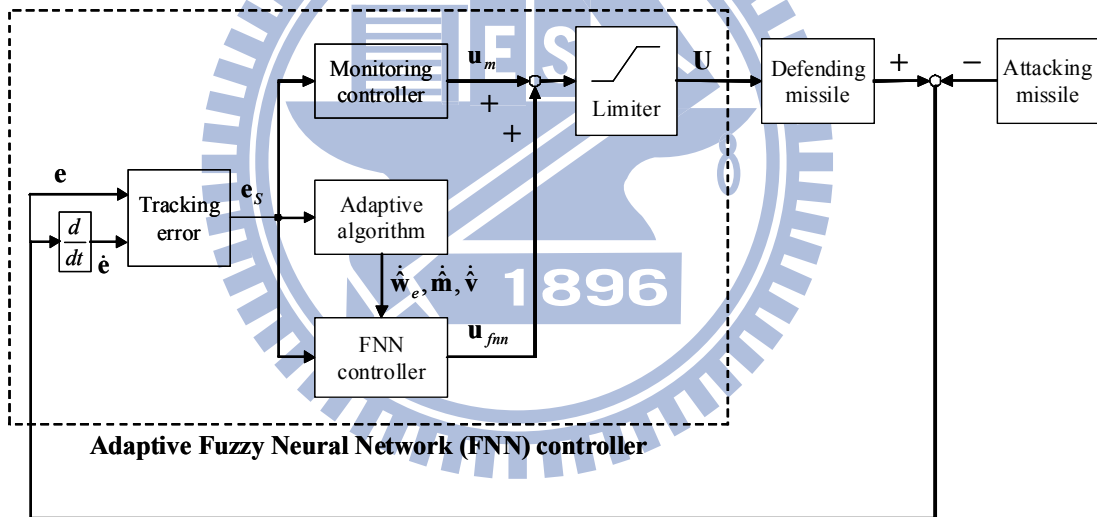


Fig. 3-9. The closed-loop configuration of FNN controller for missile guidance.

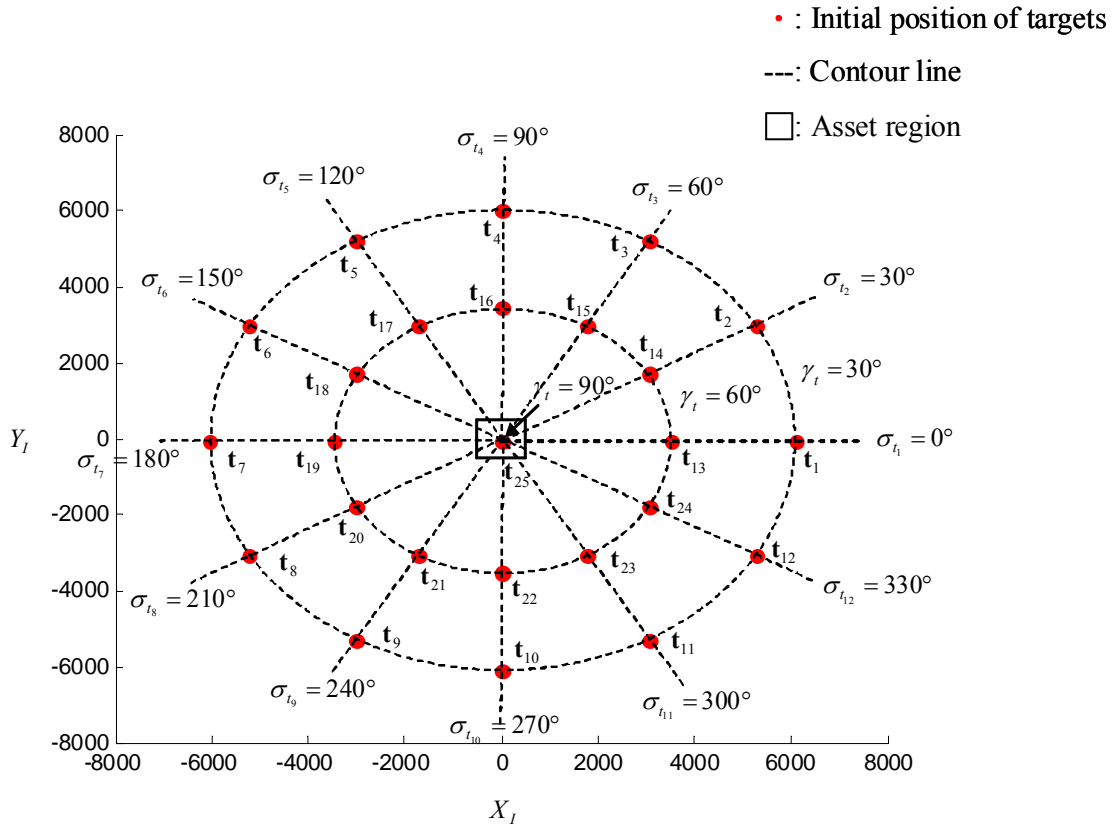


Fig. 3-10. The initial positions and angles of targets in the  $X_I$ - $Y_I$  plane.

The errors and change of errors of FNN controller in 25 battle scenario simulations are shown in Fig. 3-11. From Fig. 3-11, it can be obviously seen that our proposed adaptive FNN controller is capable of performing missile guidance. The control input  $\mathbf{u}$  of FNN controller shown in Fig. 3-12 contains the yaw and pitch acceleration commands which are denoted as  $d_{yc}$  and  $d_{zc}$ , respectively. In comparison with the CMAC used for missile guidance in [4, 6], all the 25 guidance results under the same scenario are listed in Table 3-2 for computational load (CL) and miss distance (MD) of DM.



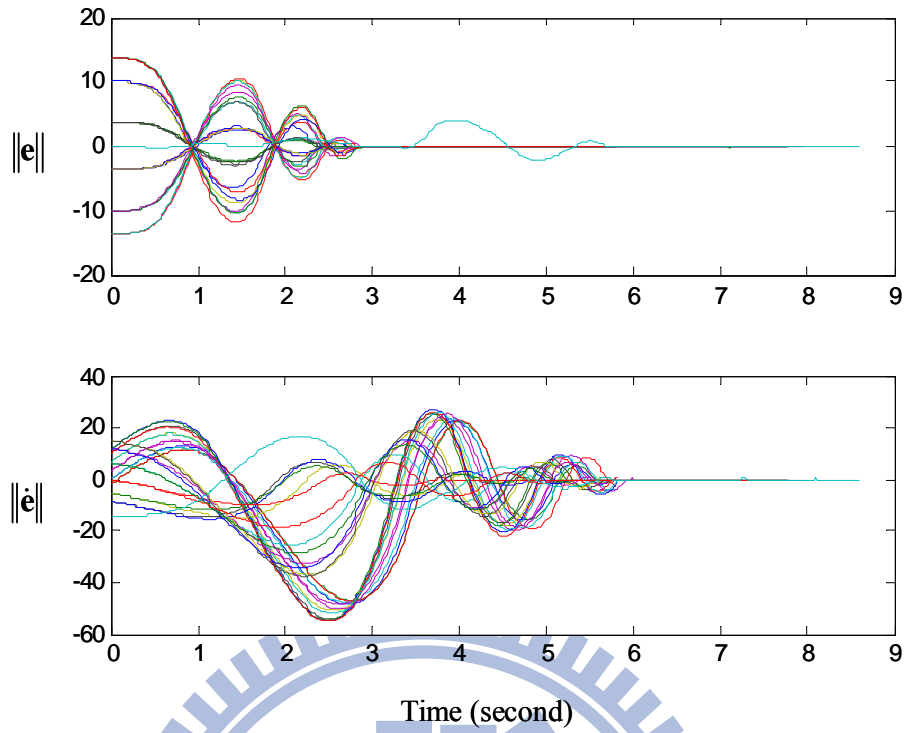


Fig. 3-11. The errors and change of errors of FNN controller in 25 battle scenario simulations.

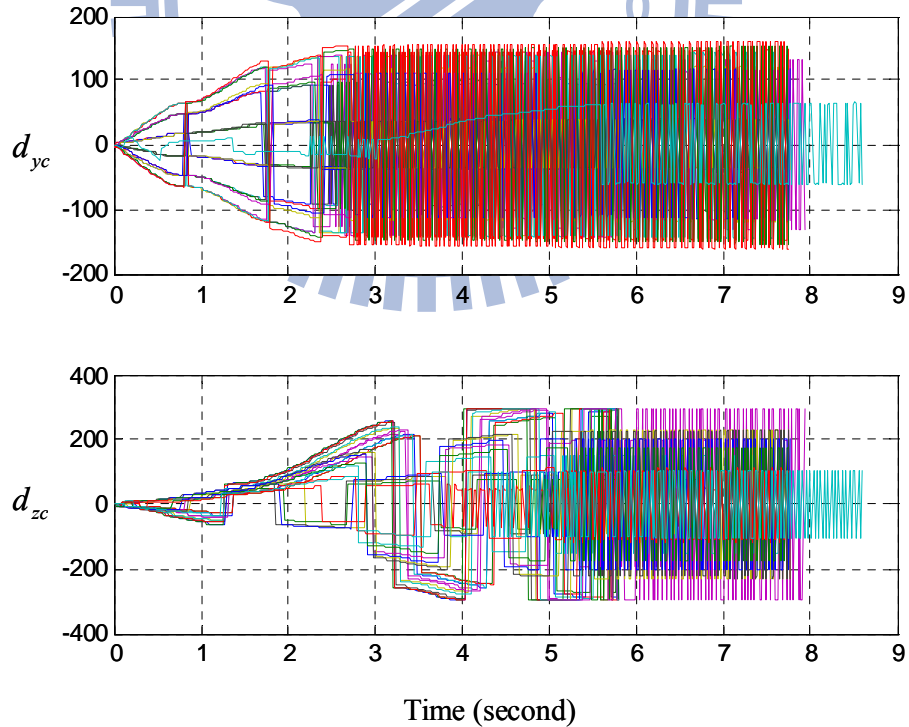


Fig. 3-12. The control inputs of FNN controller in 25 battle scenario simulations.

Table 3-2 The 25 guidance results (MD: Miss Distance; CL: Computational Load).

controller scenario	CMAC		FNN controller	
	MD (m)	CL (sec)	MD (m)	CL (sec)
1	2.7600	6.641	4.7312	0.641
2	2.9284	6.594	1.9743	0.625
3	3.1054	6.578	3.7066	0.640
4	5.8901	6.594	5.9765	0.641
5	0.1236	6.594	10.6379	0.765
6	6.0915	6.594	9.1060	0.766
7	11.4619	6.625	2.6917	0.766
8	7.7567	6.672	2.1204	0.641
9	8.1750	6.609	0.7921	0.641
10	8.6591	6.578	2.0889	0.625
11	3.8047	6.562	8.3483	0.625
12	0.9872	6.578	9.7337	0.750
13	6.6148	8.610	4.0755	0.813
14	0.6873	8.625	4.8888	0.672
15	0.9697	8.625	4.7711	0.672
16	4.6191	8.609	5.0199	0.671
17	2.0204	8.641	5.9066	0.688
18	1.9740	8.625	7.6901	0.688
19	4.5209	8.641	5.8192	0.656
20	4.8498	8.609	2.5172	0.796
21	2.1815	8.625	2.9209	0.672
22	5.8322	8.609	6.2649	0.687
23	4.8678	8.625	7.0819	0.687
24	5.1705	8.609	4.6043	0.672
25	42.5002	9.500	3.6836	0.671

In Table 3-2, the average MD of CMAC is 5.9421 meters which is larger than that of FNN, which is 5.0861 meters. In Table 3-2, it is obvious that the CMAC will fail in 25<sup>th</sup> scenario due to its MD equals to over 40 meters, whereas the agent in using our FNN controller is only 3.6836 meters. It can be seen that the scenarios for CMAC should be chosen carefully to

prevent the divergence of missile guidance. For the comparison of computational load, Table 3-2 also shows the running time under the same Windows XP system. It is amazing to see that the average CL (in seconds) of CMAC is ten times larger than that of FNN. In real-time control system, the larger CL is not preferred, especially when the intelligent missile guidance law is applied to agent. From the above simulation results, the proposed adaptive FNN controller is capable of efficiently maneuvering the agent toward the target in finite time, and the MD of agent can also be reduced to a satisfactory level. Moreover, the proposed FNN controller has much smaller CLs than those of CMAC in all the simulation results.

### **Case 3-2:**

The adjustment parameter is chosen as  $\delta = 0.1$  to emphasize the importance of asset value. For ease of simulation in the following scenarios, the asset value will be randomly chosen as an integer from 1 to 3; the numbers of assets, agents, and targets are also randomly chosen from 1 to 15. The initial positions and angles of targets in [44, 46] are chosen without any reason, which is very sensitive during simulation. If more scenarios are needed, it is time-consuming for the user to determine these initial parameters, even for simulation purpose. In this case, the ground tracker is set at the original point and the initial LOS ranges of targets are also set as 7000 meters. Also, the positions of assets and initial agents are randomly located in the bounded asset region. For ease of simulation, a random number of targets are chosen from the 25 initial positions to represent the attacking targets, which are incoming from different degrees as shown in Fig. 3-10. Once an attacking target is moving toward some asset, the initial angle of this attacking target can be decided by calculating the heading angle to the asset. The hit probability of all the targets for assets is also assumed as 100 % such that the assets will not be destroyed if the agents are forced to intercept the corresponding targets.

### **Scenario 1:**

Consider 9 agents and 9 targets in the same three-dimensional workspace shown in Fig. 3-13. Because the number of agents is the same as the number of targets and the hit probability is

not considered, scenario 1 can be regarded as the extension of single agent-target guidance problem. In this scenario, the asset damaging cost by all the targets is listed as  $\bar{V} = \{1,2,3,1,1,2,3,3,2\}$  without interception. After the task assignment, the 9 agents are forced to their corresponding targets. The interception list  $L = \{3,7,8,9,6,2,1,4,5\}$  can be obtained via SOM with the consideration of the physical distances between the agents and targets. The interception list of  $L = \{1,2,3,4,5,6,7,8,9\}$  will be obtained if the physical distances between the agents and targets are not considered in our SOM. The agents will find their matching targets by  $L$ , and then be forced to the incoming targets via adaptive SOM with FNN controller. From the simulation result, it shows that the total damaging cost is reduced to 0 which is the minimal total damaging cost. The error and change of error of all the agents also approaches to zero as shown in Fig. 3-14.

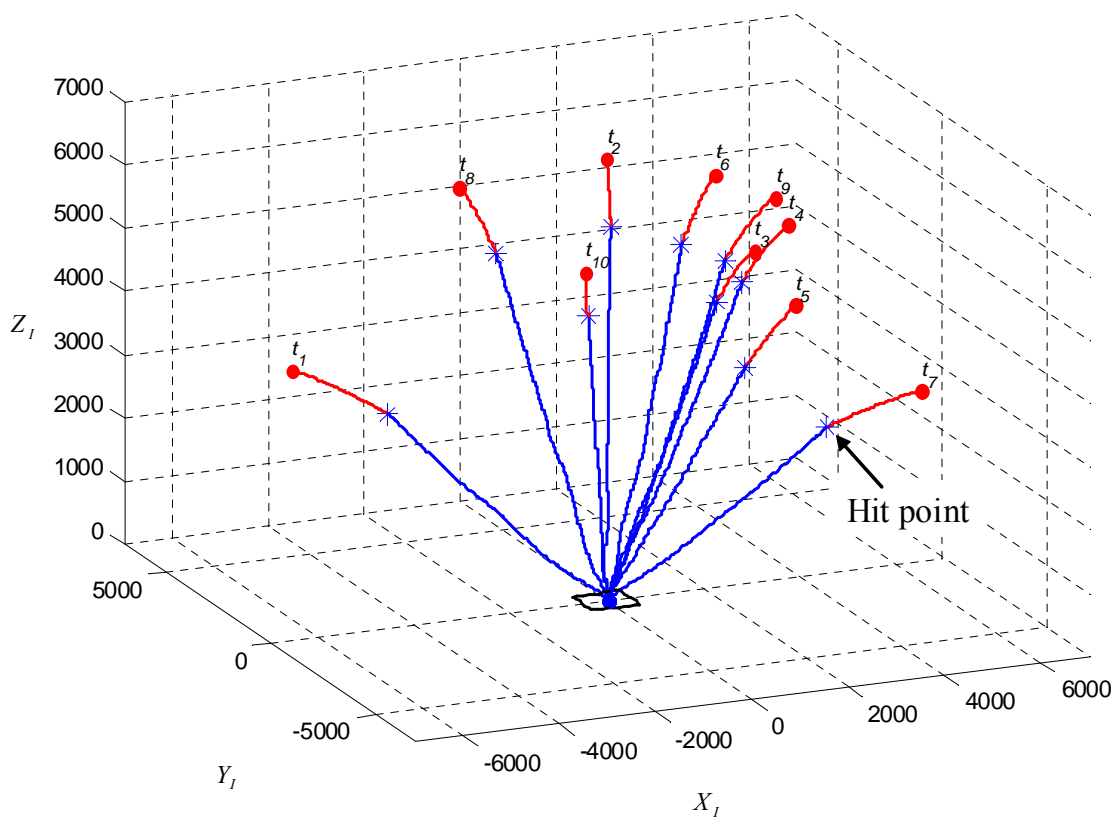


Fig. 3-13. The trajectories of 9 agents with 9 targets.

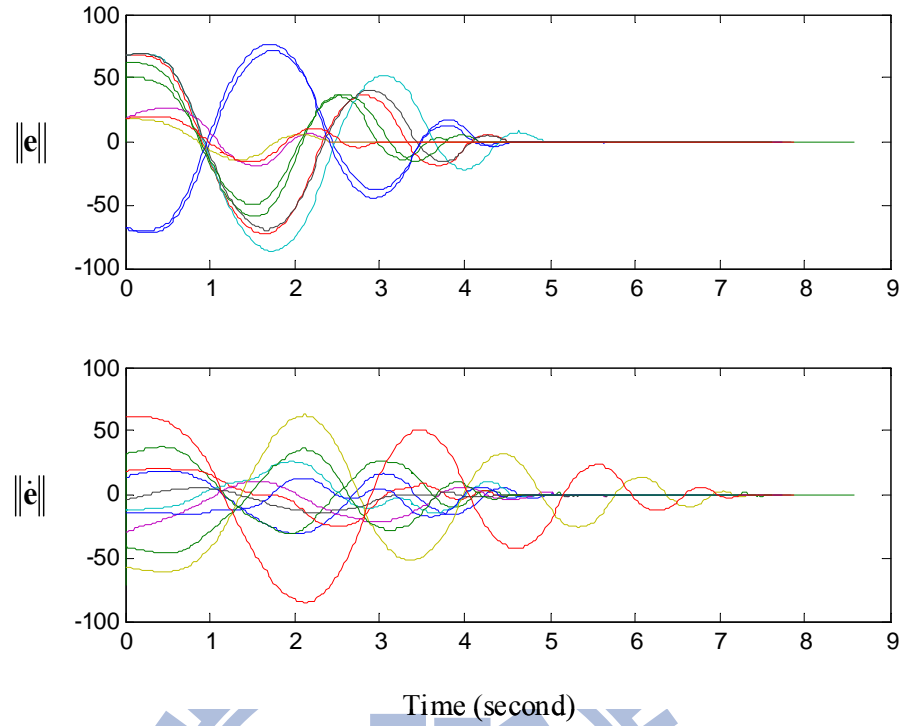


Fig. 3-14. The errors and change of errors of MAS.

**Scenario 2:**

Consider 3 agents and 9 targets in the same three-dimensional workspace shown in Fig. 3-15. In this scenario, the asset damaging cost by all the targets is listed as  $\bar{V} = \{1,2,1,3,2,1,3,3,2\}$  without interception. The interception list  $L = \{7,8,4\}$  can be obtained via SOM. It can be obviously seen that the interception list  $L$  command to the FNN for the 1<sup>st</sup>, 2<sup>nd</sup>, and 3<sup>rd</sup> agent to intercept the 5<sup>th</sup>, 8<sup>th</sup>, and 4<sup>th</sup> target. Therefore, from the simulation result it shows that the total damaging cost is reduced to 9 which is the minimal total damaging cost. The error and change of error of all the agents also approaches to zero as shown in Fig. 3-16. In this scenario, the traditional exhaustive method will take  $9!/(9-3)! (= 504)$  computation steps to find the minimal total damaging cost.

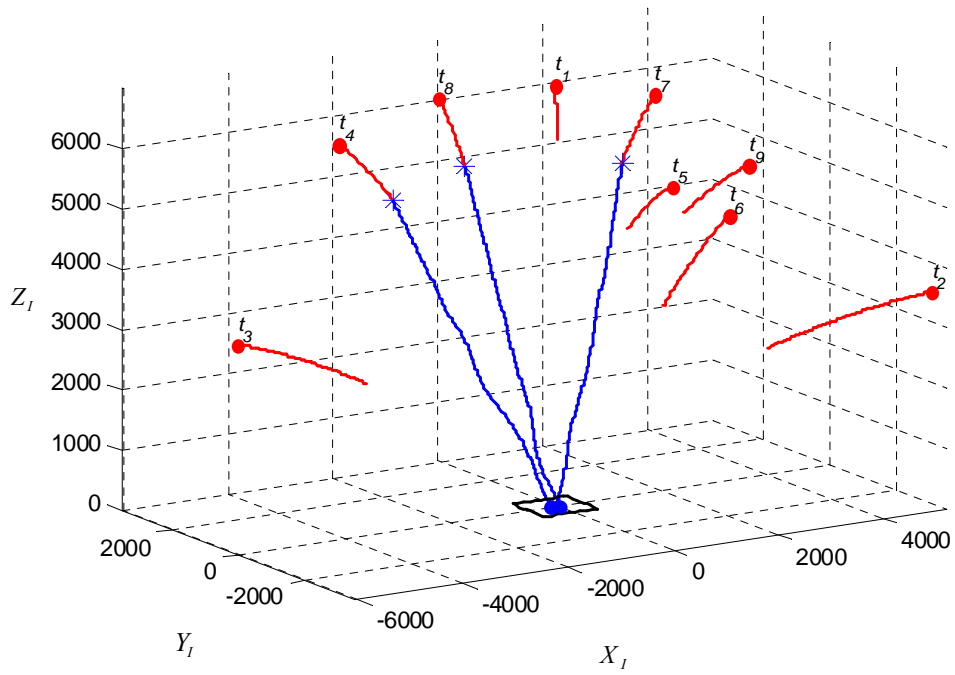


Fig. 3-15. The trajectories of 3 agents with 9 targets.

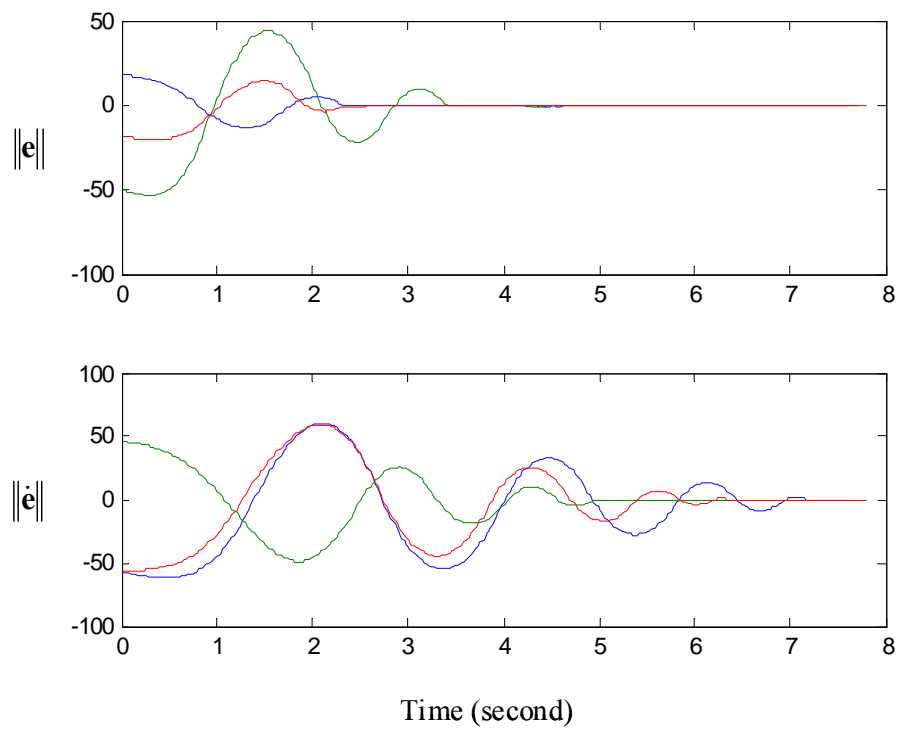


Fig. 3-16. The errors and change of errors of MAS.

**Scenario 3:**

Consider 4 agents and 12 targets in the same three-dimensional workspace shown in Fig. 3-17. In this scenario, the asset damaging cost by all the targets is listed as  $\bar{V} = \{3,2,1,1,1,3,2,2,2,2,3,3,1\}$  without interception. The interception list  $L = \{11,1,6,10\}$  can be obtained via SOM. From the simulation result, it shows that the total damaging cost is reduced to 18 which is the minimal total damaging cost. The error and change of error of all the agents also approaches to zero as shown in Fig. 3-18. In this scenario, the traditional exhaustive method will take more time-consuming  $12!/(12-4)! (= 11880)$  computation steps than that of SOM to find the minimal total damaging cost.

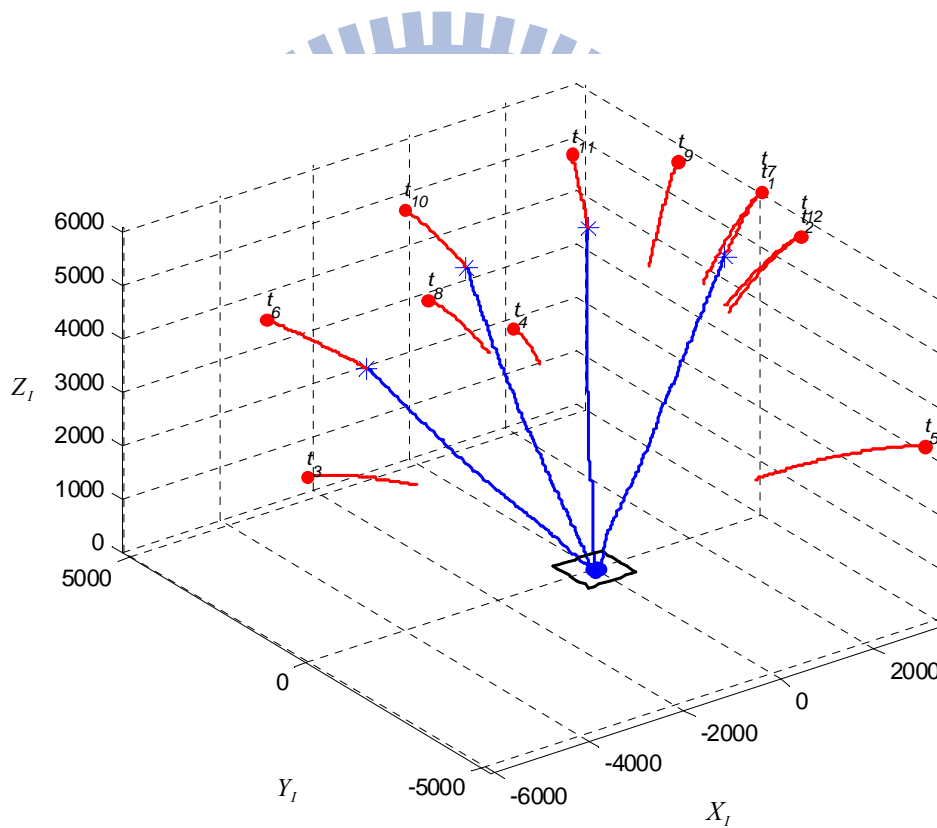


Fig. 3-17. The trajectories of 4 agents with 12 targets.

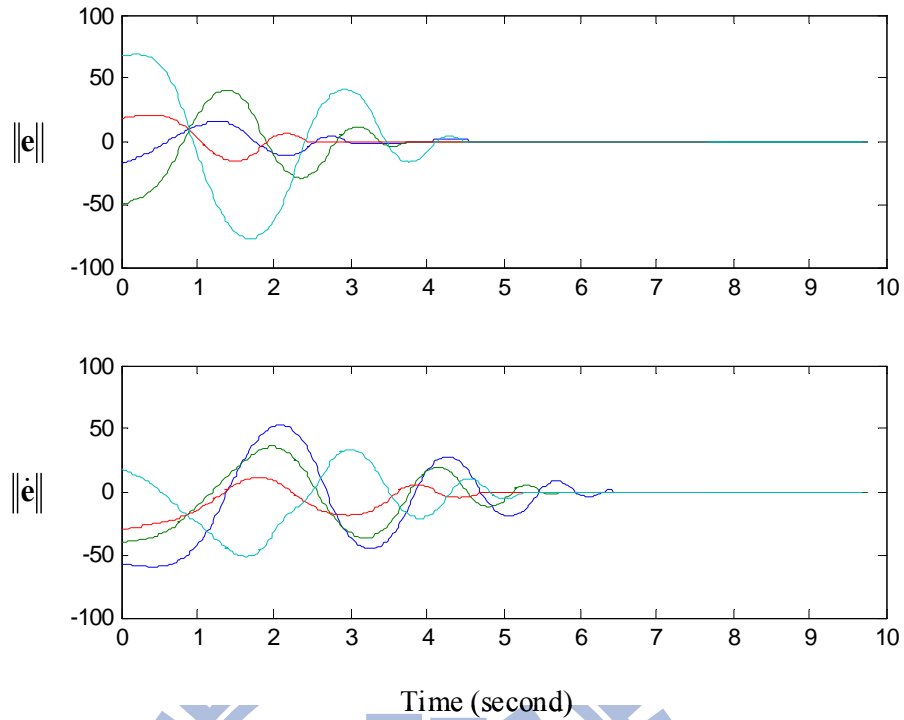


Fig. 3-18. The errors and change of errors of MAS.

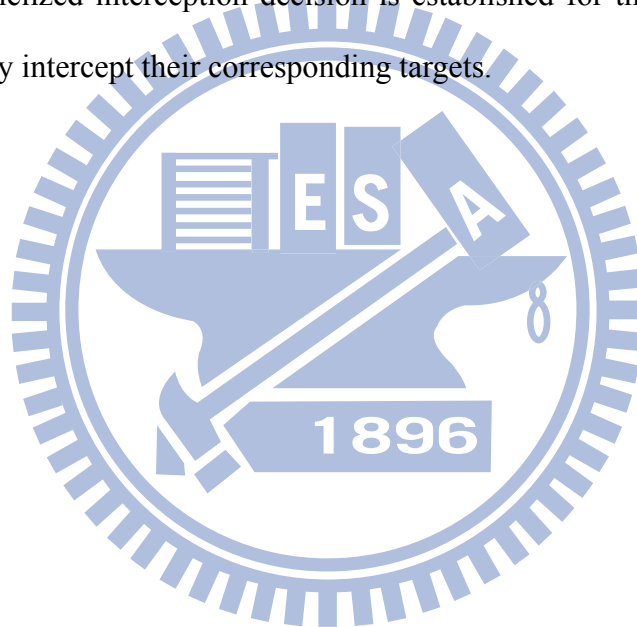
From the above simulation results, the adaptive SOM with FNN control method is capable of handling task assignment of the agents and then efficiently maneuvering the agents toward the corresponding targets in finite time. The control errors of all the agents can also be reduced to a satisfactory level. Moreover, the total damaging cost can be minimized and the number of computation steps takes no more than  $N * D$  in our adaptive SOM with FNN method.

### 3.7 Conclusions

In this chapter, a SOM with FNN controller is adopted in the MDS to find the minimal total damaging cost from the asset values and perform path evolution (or planning). In comparison with the traditional exhaustive method and simple incremental path planning in traditional SOM to determine the interception list, the proposed adaptive SOM with FNN structure can efficiently handle task assignment under the high nonlinearities and uncertainties of the agents in this chapter. The proposed main controller combined with CLOS guidance



law is the FNN controller, in which fuzzy rules are combined into the neural network, and a new monitoring controller is also designed to work with FNN controller. This forces the agents to go to their corresponding targets within the constraints of nonlinear dynamics and uncertainties of the agents. It is obvious that the weighting factors are updated via the Lyapunov stability constraints, a process which is very different from the simple update method used by the traditional SOM. From the simulation results, excellent TAPE for all agents has been obtained via the intelligent adaptive SOM with FNN controller. Summarizing the above approaches proposed in this chapter, not only the MDS model is delineated, but also the effective parallelized interception decision is established for the agents from a bounded region to efficiently intercept their corresponding targets.



# Chapter 4

## Dynamical System Identification using High-Order Hopfield-based Neural Network (HOHNN)

### 4.1 Background and Motivation

The high-order Hopfield-based neural network (HOHNN) with functional link net (FLN) has been developed in this chapter for the purpose of uncertain dynamical system identification. In comparison with the traditional Hopfield neural network (HNN) and the high-order neural network (HONN), the compact structure of FLN with a systematic order mathematical representation combined into the proposed HOHNN has additional inputs for each neuron for faster convergence rate and less computational load. The weighting factors in HOHNN are tuned via the Lyapunov stability theorem to guarantee the convergence performance of real-time system identification. The robust learning analysis of HOHNN to improve the convergence in the performance is also discussed.

### 4.2 High-Order Hopfield-based Neural Network (HOHNN) Models

Three different types of artificial neural networks can be classified based on their feedback link connection architecture. This chapter focuses on the recurrent neural networks. Figure 4-1 shows an HNN structure that forms a multiple-loop feedback system where the number of feedback loops is equal to the number of neurons [38]. The output of each neuron fed back as an input to each neuron in the network. Considering the architecture of HNN, as illustrated in Fig. 4-1, the synaptic weighting vector  $\mathbf{w}^i(t) = [w_1^i(t) w_2^i(t) \dots w_n^i(t)]$  represents conductance, the input vector  $\mathbf{x}^i(t) = [x_1^i(t) x_2^i(t) \dots x_n^i(t)]$  represents voltages, and  $n$  represents the number of inputs. In Fig. 1, the input vector  $\mathbf{x}^i(t)$  is fed back from the output vector  $\mathbf{y}(t) = [y_1(t) y_2(t) \dots y_n(t)] = [v_1(t) v_2(t) \dots v_n(t)]$ , and a current source  $I_i$  represents the externally

applied bias. The nonlinear function  $\varphi(\cdot)$  is a sigmoid function that limits the permissible amplitude range of the sum of the inputs as defined by the following hyperbolic tangent function:

$$\varphi(v_i) = \tanh\left(\frac{a_i v_i}{2}\right) \quad (4-1)$$

which has a slope of  $a_i/2$  at the origin, as shown by

$$\frac{a_i}{2} = \left. \frac{d\varphi(v_i)}{dv_i} \right|_{v_i=0} \quad (4-2)$$

where  $a_i$  and  $v_i$  are referred to as the gain and the output voltage of  $i^{\text{th}}$  neuron, respectively.

Based on the Kirchhoff's current law, the following dynamic node equation can be obtained:

$$C_i \frac{dv_i(t)}{dt} + \frac{v_i(t)}{R_i} = \sum_{i=1}^n \mathbf{w}^i(t)^T \mathbf{x}^i(t) + I_i, \quad i = 1, \dots, n. \quad (4-3)$$

Because the input is the feedback of the combination of the output, Equation (4-3) becomes

$$C_i \frac{dv_i(t)}{dt} + \frac{v_i(t)}{R_i} = \sum_{i=1}^n \mathbf{w}^i(t)^T \varphi(v_i(t)) + I_i, \quad i = 1, \dots, n. \quad (4-4)$$

The stability analysis of the above HNN was proven in [38], in which an energy function was defined, and the derivative of the energy function was negative to yield an asymptotically stable system.

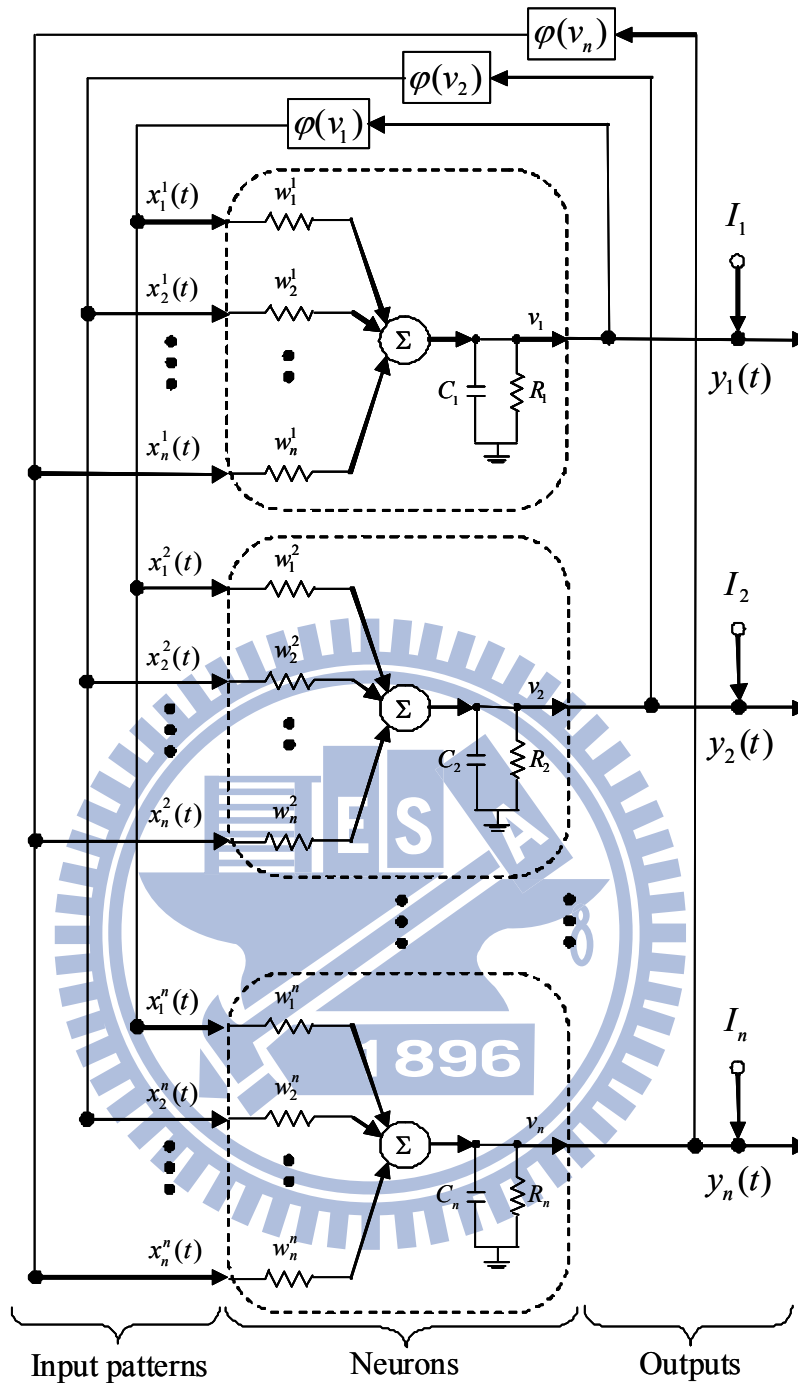


Fig. 4-1. The architecture of Hopfield neural network.

#### 4.2.1 Description of HOHNN

The FLN recently studied in [71–73] was a single-layer network whose need for hidden layer was removed. The FLN in [58] was formed by the  $i$ -dimensional input vector  $\mathbf{x}$  and enhanced to the  $j$ -dimensional input vector  $\mathbf{x}_p$  with elements  $\{x_{p1}, x_{p2}, \dots, x_{pi}, \dots, x_{pj}\}$  by a

systematic order or non-ordered mathematical representation. The expansions of the first-order Hopfield model with non-ordered mathematical representation in [74] formed high-order neural networks (HONNs). However, the systematic order mathematical representation is adopted into the proposed HOHNN in this thesis.

#### 4.2.2 Major Works

The systematic-ordered enhanced patterns are considered with high-order terms beyond third-order terms to form the fully extended HOHNN (FHOHNN) structure. However, such expansive transforms increase the number of components greatly as the dimensions of input vector increases as shown in Table 4-1.

Table 4-1 Increase number of input pattern components with enhancement.

HNN	HOHNN	FHOHNN
Number of initial patterns $\{x_i\}$	Number of components $\{x_i, x_i x_j\}, j > i$	Number of components $\{x_i, x_i x_j, x_i x_j x_k\}, k > j > i$
2	3	3
3	6	7
4	10	14
10	35	155

Hence, it was suggested in [58] that high-order terms beyond the second-order terms are not required in the enhanced patterns of input vector of HOHNN by omitting the terms with two or more equal indices. Therefore, a compact FLN is defined with rigorous formulae as shown in Fig. 4-2 and the equations below. The input pattern vector  $\mathbf{Z}$  in Fig. 4-2 is defined precisely as follows:

$$\mathbf{Z} = [z_1 \quad z_2 \quad \cdots \quad z_N \quad z_{N+1} \quad \cdots \quad z_{N(N+1)/2}] \quad (4-5)$$

and

$$\begin{aligned}
& \left\{ z_{N+\ell_1} = z_1 z_{\ell_1+1} \mid \ell_1 = 1, 2, \dots, N-1 \right\} \\
& \left\{ z_{(2N-1)+\ell_2} = z_2 z_{\ell_2+2} \mid \ell_2 = 1, 2, \dots, N-2 \right\} \\
& \left\{ z_{(3N-3)+\ell_3} = z_3 z_{\ell_3+3} \mid \ell_3 = 1, 2, \dots, N-3 \right\} \\
& \quad \vdots \\
& \left\{ z_{[kN-k(k-1)/2]+\ell_k} = z_k z_{\ell_k+k} \mid \ell_k = 1, 2, \dots, N-k \right\} \\
& \quad \vdots \\
& \left\{ z_{[(N-1)N-(N-1)(N-2)/2]+\ell_{N-1}} = z_{N-1} z_{\ell_{N-1}+N-1} \mid \ell_{N-1} = 1, 2, \dots, N-(N-1) \right\}
\end{aligned} \tag{4-6}$$

The above (4-5) states that the  $\mathbf{Z}$  vector has  $N(N+1)/2$  terms, in which  $N$  is the number of original input variables. In addition, Equation (4-6) describes all the extra second-order terms for  $\{z_{N+1} z_{N+2} \dots z_{N(N+1)/2}\}$ . The input vector of HNN shown in Fig. 4-1 can act as a compact FLN, as defined in Fig. 4-2.

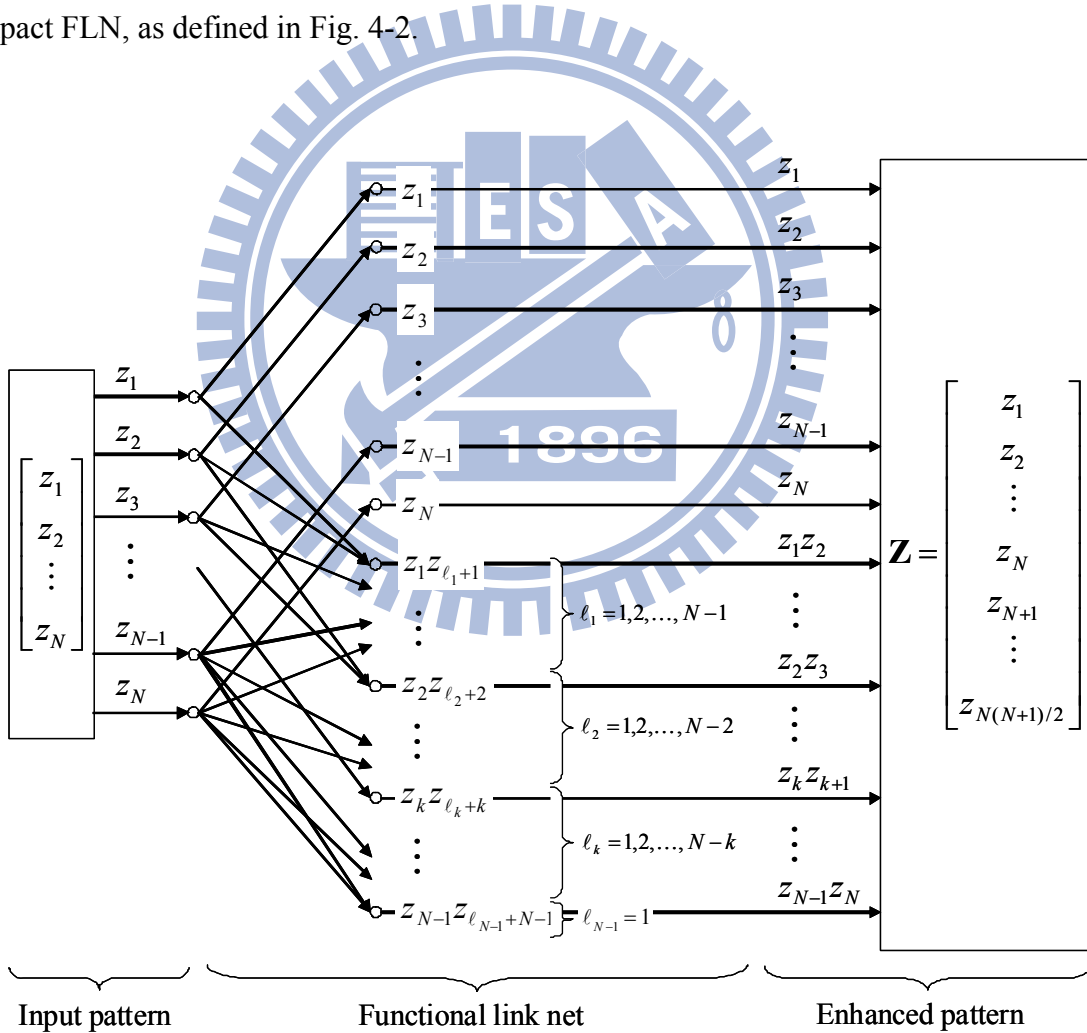


Fig. 4-2. A compact structure of functional link net.

In this thesis, the HNN with a compact FLN is to form HOHNN, which can be demonstrated

to be an approximation by the Stone-Weierstrass theorem for continuous time dynamical systems. The detailed proof is given in the next section.

### 4.3 The Function Approximation using HOHNN

Consider a continuous-time nonlinear dynamical system of the following form:

$$\begin{aligned}\dot{\mathbf{x}} &= F(\mathbf{x}, \mathbf{u}) \\ \mathbf{y} &= \mathbf{x}\end{aligned}\quad (4-7)$$

where  $\mathbf{x} = [x_1 \ x_2 \ \dots \ x_n]^T$  is the system state vector assumed to be available for measurement,  $\mathbf{u}$  and  $\mathbf{y}$  are the input and output vectors of the system, and  $F(\mathbf{x}, \mathbf{u})$  is the nonlinear function which describes either affine or non-affine system dynamics. In addition, a bounded-input, bounded-output (BIBO) condition is also imposed for (4-7) (i.e., if the admissible control input is bounded then the state trajectories are uniformly bounded for any finite initial condition). The aim of this chapter is to discuss the capability of function approximation of HOHNN for unknown nonlinear systems. In order to approximate the unknown nonlinear system, HOHNN with single-layer, fully connected, recurrent nets and functional link model is proposed. Assuming that  $n$  neurons are needed to identify an  $n^{\text{th}}$  order unknown nonlinear dynamical system and from (4-4), the mathematical model of the proposed HOHNN for system identification with zero bias can be expressed as follows:

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{W}\mathbf{Z} \quad (4-8)$$

where  $\hat{\mathbf{x}} = [\hat{x}_1 \ \hat{x}_2 \ \dots \ \hat{x}_n]^T = [v_1 \ v_2 \ \dots \ v_n]^T$  is the  $n$ -dimensional state vector;  $\mathbf{A} = \text{diag}[-1/R_1C_1 \ -1/R_2C_2 \ \dots \ -1/R_nC_n]$  and  $\mathbf{B} = \text{diag}[1/C_1 \ 1/C_2 \ \dots \ 1/C_n]$  are both  $n \times n$  diagonal matrices. For this compact FLN, the dimension of input pattern has been expanded to  $N' = (n + M) \times (n + M + 1) / 2$ . Thus, the  $N'$ -dimension input vector  $\mathbf{Z}$  can be defined as

$$\mathbf{Z} = [\Phi \ \mathbf{U} \ \mathbf{Z}_h]^T. \quad (4-9)$$

Furthermore, the associated weight can also be defined as an  $n \times N'$  matrix of synaptic weighting factor matrix

$$\mathbf{W} = [\mathbf{w}_\varphi \quad \mathbf{w}_u \quad \mathbf{w}_h] \quad (4-10)$$

where  $\mathbf{w}_\varphi$ ,  $\mathbf{w}_u$ , and  $\mathbf{w}_h$  are the weighting factors of feedback input, control voltage input, and high-order term, respectively. By assumption,  $(\mathbf{x}(t), \mathbf{u}(t)) \in S$ ,  $\forall t \in [0, T]$ . Let

$$S_\varepsilon = \left\{ (\hat{\mathbf{x}}, \mathbf{u}) \in \mathfrak{R}^{n+M} : |(\hat{\mathbf{x}}, \mathbf{u}) - (\hat{\mathbf{x}}_m, \mathbf{u}_m)| \leq \varepsilon, (\hat{\mathbf{x}}_m, \mathbf{u}_m) \in S \right\} \quad (4-11)$$

where  $S$  is an  $(n + M)$ -dimensional compact subset to be properly chosen. Clearly,  $S_\varepsilon$  is also a compact subset of  $\mathfrak{R}^{n+M}$  and  $S \subset S_\varepsilon$ . In other words,  $S_\varepsilon$  is larger than  $S$  by an arbitrary small value  $\varepsilon$ . Because of the universal approximation theorem [38, 75, 76] and the bound of the input, the modeling error can be arbitrarily small. Substituting (4-9) and (4-10) into (4-8),

$$\dot{\hat{\mathbf{x}}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{w}_\varphi \Phi + \mathbf{B}\mathbf{w}_u \mathbf{U} + \mathbf{B}\mathbf{w}_h \mathbf{Z}_h \quad (4-12)$$

where

$$\Phi = [q\varphi(\hat{x}_1) \quad q\varphi(\hat{x}_2) \quad \cdots \quad q\varphi(\hat{x}_n)]^T, \quad (4-13)$$

$$\mathbf{U} = [u_1 \quad u_2 \quad \cdots \quad u_M]^T, \quad (4-14)$$

and

$$\mathbf{Z}_h = [q\varphi(\hat{x}_1) \cdot q\varphi(\hat{x}_2) \quad \cdots \quad q\varphi(\hat{x}_1) \cdot q\varphi(\hat{x}_n) \quad \cdots \quad \cdots \quad u_{M-1} \cdot u_M]^T. \quad (4-15)$$

In Fig. 4-3,  $\Phi$  is the  $n$ -dimension vector of the network feedback in a standard HNN with  $\varphi(\cdot)$  being a nonlinear sigmoid function, and  $q$  is a positive feedback constant;  $\mathbf{U}$  is the  $M$ -dimension vector of the control force, and  $\mathbf{Z}_h$  in (4-15) is the high-order nonlinear vector to the system which comes from (4-6) in the compact FLN shown in Fig. 4-2. This combination is the HOHNN for a single neuron, which is shown in Fig. 4-3. The input pattern of HOHNN produced from the FLN is the enhanced pattern in which the neural feedback, control voltage input, and the combination of these two inputs are contained. The function approximation problem consists of determining whether to allow sufficient high-order connections with weighting factor matrix  $\mathbf{W}$ , such that the HOHNN model approximates the input-output



behaviour of an arbitrary unknown nonlinear dynamical system. To come up with a well-posed problem, assume that  $\mathbf{W}$  has an optimal  $\mathbf{W}^* = [\mathbf{w}_\varphi^* \ \mathbf{w}_u^* \ \mathbf{w}_h^*]$  that can approximate the nonlinear dynamical system to any degree of accuracy. Therefore, the state equation in (4-7) can be approximated by an HOHNN using the following optimal form:

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{w}_\varphi^* \Phi + \mathbf{B}\mathbf{w}_u^* \mathbf{U} + \mathbf{B}\mathbf{w}_h^* \mathbf{Z}_h. \quad (4-16)$$

Moreover, the optimal matrices can be further defined as [38, 75, 76]

$$(\mathbf{w}_\varphi^*, \mathbf{w}_u^*, \mathbf{w}_h^*) = \arg \min_{\substack{\mathbf{w}_\varphi \in \Omega_{\mathbf{w}_\varphi}, \\ \mathbf{w}_u \in \Omega_{\mathbf{w}_u}, \\ \mathbf{w}_h \in \Omega_{\mathbf{w}_h}}} \left[ \sup_{\substack{\mathbf{x} \in \Omega_{\mathbf{x}}, \\ \hat{\mathbf{x}} \in \Omega_{\hat{\mathbf{x}}}, \\ u \in \Omega_u}} |F(\mathbf{x}, \mathbf{u}) - (\mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{w}_\varphi \Phi + \mathbf{B}\mathbf{w}_u \mathbf{U} + \mathbf{B}\mathbf{w}_h \mathbf{Z}_h)| \right] \quad (4-17)$$

where

$$\Omega_{\mathbf{w}_\varphi} = \{\mathbf{w}_\varphi : tr(\mathbf{w}_\varphi^T \mathbf{w}_\varphi) \leq D_{\mathbf{w}_\varphi}\} \quad (4-18)$$

$$\Omega_{\mathbf{w}_u} = \{\mathbf{w}_u : tr(\mathbf{w}_u^T \mathbf{w}_u) \leq D_{\mathbf{w}_u}\} \quad (4-19)$$

$$\Omega_{\mathbf{w}_h} = \{\mathbf{w}_h : tr(\mathbf{w}_h^T \mathbf{w}_h) \leq D_{\mathbf{w}_h}\} \quad (4-20)$$

are compact constraint sets for  $\mathbf{w}_\varphi$ ,  $\mathbf{w}_u$ , and  $\mathbf{w}_h$ , respectively, specified by designers. Here,  $D_{\mathbf{w}_\varphi}$ ,  $D_{\mathbf{w}_u}$ , and  $D_{\mathbf{w}_h}$  can be seen as bounded in a ball of radius, also specified by designers to avoid the arbitrarily large weight values. Secondly, suppose that  $F$  is continuous and satisfies a local Lipschitz condition such that (4-7) has a unique solution and  $(\mathbf{x}(t), \mathbf{u}(t)) \in S$  for all  $t$  in some time interval  $I_T = \{t: 0 \leq t \leq T\}$ . The interval  $I_T$  represents the time period over which the approximation is to be performed.

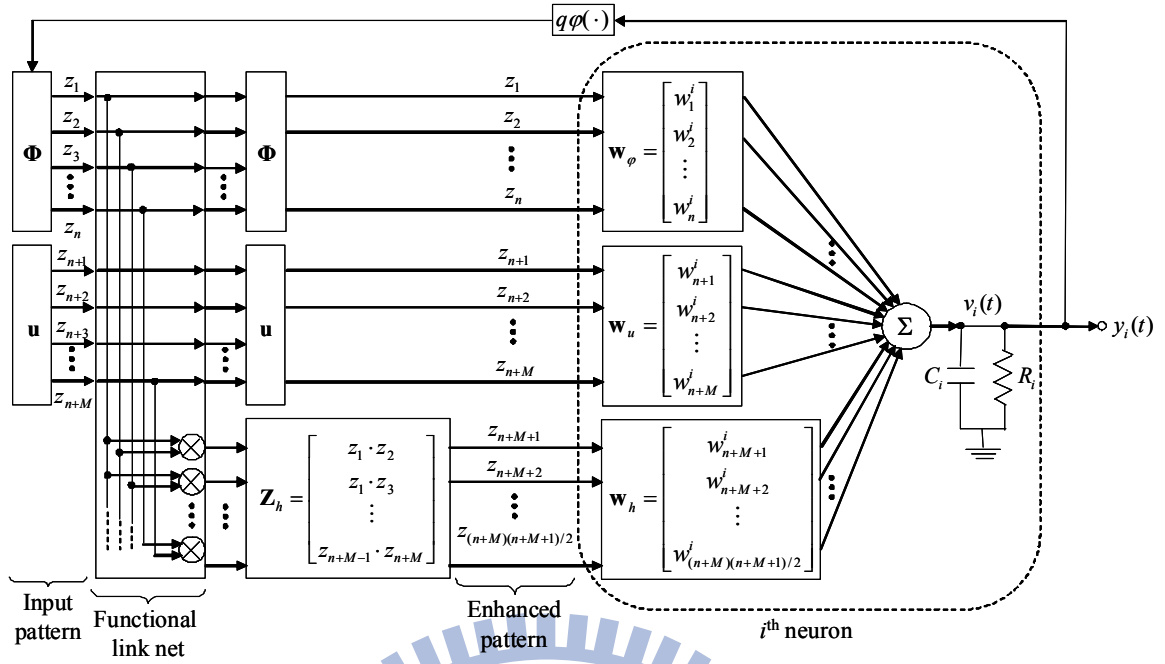


Fig. 4-3. The HOHNN structure for a single neuron.

Assume that  $F(\mathbf{x}, \mathbf{u})$  is a continuous function that satisfies the local Lipschitz condition, such that the states can be confined in the compact set  $S$ . Based on the above assumptions, the following result can be obtained.

**Lemma 4-1 [75]:** Suppose that the system in (4-8) is initial at  $\hat{\mathbf{x}}(0) = \mathbf{x}(0)$ . For the small degree of approximation  $\varepsilon > 0$  and any finite time interval  $T > 0$ , there exists an optimal weighting matrix  $\mathbf{W}^*$ . Thus, the state  $\hat{\mathbf{x}}(t)$  of the HOHNN model (4-8) with  $N'$  high-order connections and weight values  $\mathbf{W} = \mathbf{W}^*$  satisfies

$$\sup_{0 \leq t \leq T} |\mathbf{x}(t) - \hat{\mathbf{x}}(t)| \leq \varepsilon \quad (4-21)$$

where  $|\cdot|$  denotes the Euclidean vector norm. Using the Bellman-Gronwall Lemma [76], the error function  $e(t)$  can be limited in the following bounded value:

$$|e(t)| \leq \frac{\varepsilon}{2}. \quad (4-22)$$

Supposing that  $[\hat{\mathbf{x}}(t), \mathbf{u}(t)]$  does not belong to the set  $S_\varepsilon$  for all  $t \in [0, T]$ , then, by the continuity of  $\hat{\mathbf{x}}(t)$ , there exists a  $T^*$  where  $0 < T^* < T$ . Thus,  $[\hat{\mathbf{x}}(T^*), \mathbf{u}(T^*)] \in \partial S_\varepsilon$  where  $\partial S_\varepsilon$

denotes the boundary of  $S_\varepsilon$ . If the same analysis for  $t \in [0, T^*]$  is carried out,  $|\mathbf{x}(t) - \hat{\mathbf{x}}(t)| \leq \varepsilon/2$  is obtained in this interval. Therefore, the proposed HOHNN model is capable of approximating the behavior of dynamical systems to any degree of accuracy if a sufficiently large number of high-order connections between neurons is allowed.

#### 4.4 The Lyapunov Tuning of HOHNN for Identification

From above discussion, clearly, for a given nonlinear system, the HOHNN model with a sufficiently large number of high-order connections can be obtained to approximate any dynamical system to any degree of accuracy. Here, the adaptive laws for weighting factors training have to be appropriately designed to guarantee the approximation performance. The approximation error between states of the HOHNN identifier and the real system is defined as

$$\mathbf{e} = \mathbf{x} - \hat{\mathbf{x}}. \quad (4-23)$$

Thus, the derivative of  $\mathbf{e}$  with respect to time can be obtained by (4-12) and (4-16)

$$\dot{\mathbf{e}} = \mathbf{A}\mathbf{e} + \mathbf{B}\tilde{\mathbf{w}}_\varphi \Phi + \mathbf{B}\tilde{\mathbf{w}}_u \mathbf{U} + \mathbf{B}\tilde{\mathbf{w}}_h \mathbf{Z}_h \quad (4-24)$$

where  $\tilde{\mathbf{w}}_\varphi = \mathbf{w}_\varphi^* - \mathbf{w}_\varphi$ ,  $\tilde{\mathbf{w}}_u = \mathbf{w}_u^* - \mathbf{w}_u$  and  $\tilde{\mathbf{w}}_h = \mathbf{w}_h^* - \mathbf{w}_h$ . Because  $\mathbf{A}$  is a stable matrix (since  $|s\mathbf{I} - \mathbf{A}|$  is stable), there exists a unique positive definite symmetric  $n \times n$  matrix  $\mathbf{P}$  that satisfies the Lyapunov equation  $\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} = -\mathbf{Q}$ , where  $\mathbf{Q}$  is an arbitrary  $n \times n$  positive definite matrix. The weight adaptive laws that guarantee nonlinear dynamical system for approximation error minimization and approximation process convergence have been considered in Theorem 4-1 based on the absence of the modeling error. The theorem states the main result concerning the convergence of the proposed approximation scheme.

**Theorem 4-1:** A nonlinear dynamical system considered in (4-7) is assumed to be modeled exactly by (4-16) and the approximation system is designed as (4-12). If the adaptive law of weighting factors in  $i^{\text{th}}$  neuron are chosen as

$$\dot{w}_{\varphi,j}^i = -\eta_{\varphi} q \varphi(v_j) b_{ii} p_{ii} e_i, \quad \text{for } j = 1, 2, \dots, n, \quad (4-25)$$

$$\dot{w}_{u,(n+k)}^i = -\eta_u u_k b_{ii} p_{ii} e_i, \quad \text{for } k = 1, 2, \dots, M, \quad (4-26)$$

and

$$\dot{w}_{h,(n+M+l)}^i = -\eta_h z_l b_{ii} p_{ii} e_i, \quad \text{for } l = 1, 2, \dots, (n+M)(n+M-1)/2 \quad (4-27)$$

where  $\eta_{\varphi}$ ,  $\eta_u$  and  $\eta_h$  are positive learning rates; and  $b_{ii}$  and  $p_{ii}$  are the diagonal elements of  $\mathbf{B}$  and  $\mathbf{P}$ , respectively. The stability of the overall identification scheme is therefore guaranteed.

*Proof:*

Consider the Lyapunov candidate function as

$$V = \frac{1}{2} \mathbf{e}^T \mathbf{P} \mathbf{e} + \frac{1}{2\eta_{\varphi}} \text{tr}(\tilde{\mathbf{w}}_{\varphi}^T \tilde{\mathbf{w}}_{\varphi}) + \frac{1}{2\eta_u} \text{tr}(\tilde{\mathbf{w}}_u^T \tilde{\mathbf{w}}_u) + \frac{1}{2\eta_h} \text{tr}(\tilde{\mathbf{w}}_h^T \tilde{\mathbf{w}}_h) \quad (4-28)$$

where  $\mathbf{P} > 0$  is chosen to satisfy the Lyapunov equation  $\mathbf{P}\mathbf{A} + \mathbf{A}^T\mathbf{P} = -\mathbf{Q}$ . Taking the derivative of  $V$  with respect to time and using (4-24) yields

$$\begin{aligned} \dot{V} &= \frac{1}{2} (\dot{\mathbf{e}}^T \mathbf{P} \mathbf{e} + \mathbf{e}^T \mathbf{P} \dot{\mathbf{e}}) + \frac{1}{\eta_{\varphi}} \text{tr}(\dot{\tilde{\mathbf{w}}}_{\varphi}^T \tilde{\mathbf{w}}_{\varphi}) + \frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) + \frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h) \\ &= -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} + \frac{1}{2} (\Phi^T \tilde{\mathbf{w}}_{\varphi}^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e} \\ &\quad + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_{\varphi} \Phi + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_u \mathbf{U} + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_h \mathbf{Z}_h) + \frac{1}{\eta_{\varphi}} \text{tr}(\dot{\tilde{\mathbf{w}}}_{\varphi}^T \tilde{\mathbf{w}}_{\varphi}) + \frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) + \frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h). \end{aligned}$$

Because  $\Phi^T \tilde{\mathbf{w}}_{\varphi}^T \mathbf{B}^T \mathbf{P} \mathbf{e}$ ,  $\mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e}$ ,  $\mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e}$ ,  $\mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_{\varphi} \Phi$ ,  $\mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_u \mathbf{U}$ , and  $\mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_h \mathbf{Z}_h$  are all scalars, the relationships can hold as

$$\Phi^T \tilde{\mathbf{w}}_{\varphi}^T \mathbf{B}^T \mathbf{P} \mathbf{e} = (\Phi^T \tilde{\mathbf{w}}_{\varphi}^T \mathbf{B}^T \mathbf{P} \mathbf{e})^T = \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_{\varphi} \Phi,$$

$$\mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e} = (\mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e})^T = \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_u \mathbf{U},$$

and

$$\mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e} = (\mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e})^T = \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_h \mathbf{Z}_h.$$

Hence, the derivative of  $V$  with respect to time can be reorganized as

$$\begin{aligned} \dot{V} = & -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} + \Phi^T \tilde{\mathbf{w}}_\varphi^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e} \\ & + \frac{1}{\eta_\varphi} \text{tr}(\dot{\tilde{\mathbf{w}}}_\varphi^T \tilde{\mathbf{w}}_\varphi) + \frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) + \frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h). \end{aligned} \quad (4-29)$$

Here, select the following equations

$$\frac{1}{\eta_\varphi} \text{tr}(\dot{\tilde{\mathbf{w}}}_\varphi^T \tilde{\mathbf{w}}_\varphi) = -\Phi^T \tilde{\mathbf{w}}_\varphi^T \mathbf{B}^T \mathbf{P} \mathbf{e}, \quad (4-30)$$

$$\frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) = -\mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e}, \quad (4-31)$$

and

$$\frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h) = -\mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e}. \quad (4-32)$$

Substituting (4-30)–(4-32) into (4-29) gives

$$\dot{V} = -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} \leq 0. \quad (4-33)$$

Therefore, the stability of overall approximation scheme is guaranteed based on the above results and the Lyapunov stability theorem. Based on (4-30)–(4-32), the adaptive law of weighting factors in an element form can be obtained. The Lyapunov stability theorem is guaranteed under the optimal approximation model with no modeling error. **Q.E.D.**

#### 4.5 Robust Learning Analysis

The above assumptions are violated in many cases because of the existing modeling error in the HOHNN model. The adaptive laws of weighting factors cause the modeling error to achieve infinity if the standard adaptive laws are used for updating the weighting factors. Therefore, the modified weight adjustment laws are discussed to avoid the parameter drift problem. In formulating the problem, the identification model in (4-16) with modeling error can be corrected as

$$\dot{\mathbf{x}} = \mathbf{A} \mathbf{x} + \mathbf{B} \mathbf{w}_\varphi^* \Phi + \mathbf{B} \mathbf{w}_u^* \mathbf{U} + \mathbf{B} \mathbf{w}_h^* \mathbf{Z}_h + \mathbf{m} \quad (4-34)$$

where  $\mathbf{m}$  is a modeling error changing with respect to time, and the optimal weight vectors are

defined by (4-17)–(4-20). If  $\mathbf{m}$  is not equal to zero but is small, the stability proof may not be guaranteed, that is,  $\dot{V} > 0$ . Assume that there exists a finite bounded constant  $\delta$  so that

$$\int_0^t \|\mathbf{m}(\tau)\|^2 d\tau < \delta, \quad 0 \leq t < \infty. \quad (4-35)$$

Thus, from the BIBO stable, the time-varying  $\mathbf{m}$  is assumed to be bounded in a finite constant  $\delta$  in (4-35). Using (4-12) and (4-34), the state error satisfies

$$\dot{\mathbf{e}} = \mathbf{A}\mathbf{e} + \mathbf{B}\tilde{\mathbf{w}}_\varphi \Phi + \mathbf{B}\tilde{\mathbf{w}}_u \mathbf{U} + \mathbf{B}\tilde{\mathbf{w}}_h \mathbf{Z}_h + \delta \quad (4-36)$$

where  $\tilde{\mathbf{w}}_\varphi$ ,  $\tilde{\mathbf{w}}_u$  and  $\tilde{\mathbf{w}}_h$  as defined earlier. Because the change of weighting factors can not be guaranteed to be bounded in the ball of radius, the learning laws given by (4-25)–(4-27) are modified as follows:

$$\dot{w}_{\varphi,j}^i = \begin{cases} -\eta_\varphi q\varphi(v_j) b_{ii} p_{ii} e_i, & \text{if } |w_{\varphi,j}^i| \leq D_{w_\varphi} \text{ and } q\varphi(v_j) b_{ii} p_{ii} w_{\varphi,j}^i e_i \geq 0 \\ \mathbf{Pr}\{-\eta_\varphi q\varphi(v_j) b_{ii} p_{ii} e_i\}, & \text{if } |w_{\varphi,j}^i| > D_{w_\varphi} \text{ and } q\varphi(v_j) b_{ii} p_{ii} w_{\varphi,j}^i e_i < 0 \end{cases}, \quad (4-37)$$

$$\dot{w}_{u,(n+k)}^i = \begin{cases} -\eta_u u_k b_{ii} p_{ii} e_i, & \text{if } |w_{u,(n+k)}^i| \leq D_{w_u} \text{ and } u_k b_{ii} p_{ii} w_{u,(n+k)}^i e_i \geq 0 \\ \mathbf{Pr}\{-\eta_u u_k b_{ii} p_{ii} e_i\}, & \text{if } |w_{u,(n+k)}^i| > D_{w_u} \text{ and } u_k b_{ii} p_{ii} w_{u,(n+k)}^i e_i < 0 \end{cases}, \quad (4-38)$$

and

$$\dot{w}_{h,(n+M+l)}^i = \begin{cases} -\eta_h z_l b_{ii} p_{ii} e_i, & \text{if } |w_{h,(n+M+l)}^i| \leq D_{w_h} \text{ and } \eta_h z_l b_{ii} p_{ii} w_{h,(n+M+l)}^i e_i \geq 0 \\ \mathbf{Pr}\{-\eta_h z_l b_{ii} p_{ii} e_i\}, & \text{if } |w_{h,(n+M+l)}^i| > D_{w_h} \text{ and } \eta_h z_l b_{ii} p_{ii} w_{h,(n+M+l)}^i e_i < 0 \end{cases} \quad (4-39)$$

where the projection operator  $\mathbf{Pr}\{\cdot\}$  is defined as [40]

$$\mathbf{Pr}\{-\eta_\varphi q\varphi(v_j) b_{ii} p_{ii} e_i\} = -\eta_\varphi q\varphi(v_j) b_{ii} p_{ii} e_i + \eta_\varphi \frac{q\varphi(v_j) b_{ii} p_{ii} w_{\varphi,j}^i e_i}{|w_{\varphi,j}^i|^2} w_{\varphi,j}^i,$$

$$\mathbf{Pr}\{-\eta_u u_k b_{ii} p_{ii} e_i\} = -\eta_u u_k b_{ii} p_{ii} e_i + \eta_u \frac{u_k b_{ii} p_{ii} w_{u,(n+k)}^i e_i}{|w_{u,(n+k)}^i|^2} w_{u,(n+k)}^i,$$

and

$$\mathbf{Pr}\{-\eta_h z_l b_{ii} p_{ii} e_i\} = -\eta_h z_l b_{ii} p_{ii} e_i + \eta_h \frac{z_l b_{ii} p_{ii} w_{h,(n+M+l)}^i e_i}{|w_{h,(n+M+l)}^i|^2} w_{h,(n+M+l)}^i.$$

The closed-loop configuration of HOHNN for function approximation is shown in Fig. 4-4.

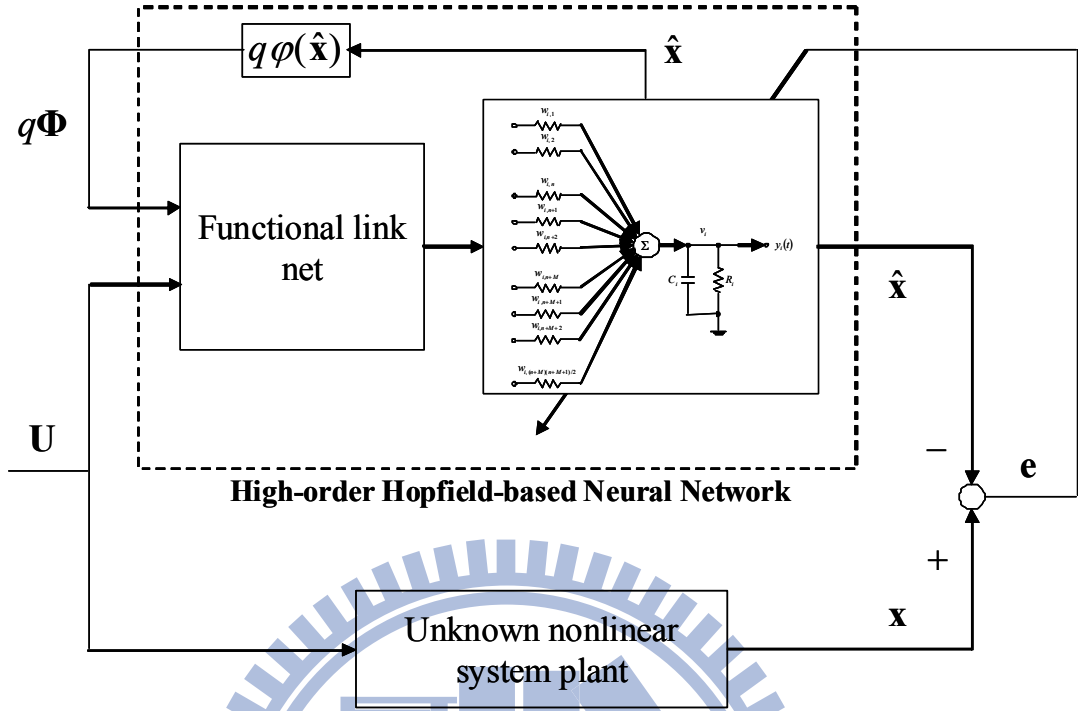


Fig. 4-4. The closed-loop configuration of HOHNN for function approximation.

Considering the same Lyapunov candidate function in (4-28) and using (4-36). Taking the derivative of  $V$  with respect to time, the following can be obtained:

$$\begin{aligned}
\dot{V} &= \frac{1}{2} [\mathbf{e}^T \mathbf{A}^T \mathbf{P} \mathbf{e} + \Phi^T \tilde{\mathbf{w}}_\varphi^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \delta^T \mathbf{P} \mathbf{e} + \mathbf{e}^T \mathbf{P} \mathbf{A} \mathbf{e} \\
&\quad + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_\varphi \Phi + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_u \mathbf{U} + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_h \mathbf{Z}_h + \mathbf{e}^T \mathbf{P} \delta] \\
&\quad + \frac{1}{\eta_\varphi} \text{tr}(\dot{\tilde{\mathbf{w}}}_\varphi^T \tilde{\mathbf{w}}_\varphi) + \frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) + \frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h) \\
&= -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} + \frac{1}{2} (\Phi^T \tilde{\mathbf{w}}_\varphi^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_h \mathbf{Z}_h + \delta^T \mathbf{P} \mathbf{e} + \mathbf{e}^T \mathbf{P} \delta) \\
&\quad + \mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_\varphi \Phi + \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_u \mathbf{U} + \frac{1}{\eta_\varphi} \text{tr}(\dot{\tilde{\mathbf{w}}}_\varphi^T \tilde{\mathbf{w}}_\varphi) + \frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) + \frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h)
\end{aligned}$$

Because  $\Phi^T \tilde{\mathbf{w}}_\varphi^T \mathbf{B}^T \mathbf{P} \mathbf{e}$ ,  $\mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e}$ ,  $\mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e}$ ,  $\mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_\varphi \Phi$ ,  $\mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_u \mathbf{U}$ ,  $\mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_h \mathbf{Z}_h$ ,  $\delta^T \mathbf{P} \mathbf{e}$  and  $\mathbf{e}^T \mathbf{P} \delta$  are all scalars, the relationships can hold as

$$\Phi^T \tilde{\mathbf{w}}_\varphi^T \mathbf{B}^T \mathbf{P} \mathbf{e} = (\Phi^T \tilde{\mathbf{w}}_\varphi^T \mathbf{B}^T \mathbf{P} \mathbf{e})^T = \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_\varphi \Phi,$$

$$\mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e} = (\mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e})^T = \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_u \mathbf{U},$$

$$\mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e} = (\mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e})^T = \mathbf{e}^T \mathbf{P} \mathbf{B} \tilde{\mathbf{w}}_h \mathbf{Z}_h,$$

and

$$\delta^T \mathbf{P} \mathbf{e} = (\delta^T \mathbf{P} \mathbf{e})^T = \mathbf{e}^T \mathbf{P} \delta.$$

Hence, the derivative of  $V$  with respect to time can be arranged as

$$\begin{aligned} \dot{V} = & -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} + \Phi^T \tilde{\mathbf{w}}_\phi^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e} + \mathbf{e}^T \mathbf{P} \delta \\ & + \frac{1}{\eta_\phi} \text{tr}(\dot{\tilde{\mathbf{w}}}_\phi^T \tilde{\mathbf{w}}_\phi) + \frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) + \frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h). \end{aligned} \quad (4-40)$$

Some useful variables are defined as

$$J_{\mathbf{w}_\phi} = \frac{1}{\eta_\phi} \text{tr}(\dot{\tilde{\mathbf{w}}}_\phi^T \tilde{\mathbf{w}}_\phi) + \Phi^T \tilde{\mathbf{w}}_\phi^T \mathbf{B}^T \mathbf{P} \mathbf{e},$$

$$J_{\mathbf{w}_u} = \frac{1}{\eta_u} \text{tr}(\dot{\tilde{\mathbf{w}}}_u^T \tilde{\mathbf{w}}_u) + \mathbf{U}^T \tilde{\mathbf{w}}_u^T \mathbf{B}^T \mathbf{P} \mathbf{e},$$

and

$$J_{\mathbf{w}_h} = \frac{1}{\eta_h} \text{tr}(\dot{\tilde{\mathbf{w}}}_h^T \tilde{\mathbf{w}}_h) + \mathbf{Z}_h^T \tilde{\mathbf{w}}_h^T \mathbf{B}^T \mathbf{P} \mathbf{e}.$$

Thus, the equation shown in (4-40) can be rewritten as

$$\dot{V} = -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} + J_{\mathbf{w}_\phi} + J_{\mathbf{w}_u} + J_{\mathbf{w}_h} + \mathbf{e}^T \mathbf{P} \delta. \quad (4-41)$$

By using (4-37) and assuming  $J_{\mathbf{w}_\phi} = 0$ , the conditions  $|w_{\phi,j}^i| \leq D_{\mathbf{w}_\phi}$  and

$q\varphi(v_j) b_{ii} p_{ii} w_{\phi,j}^i e_i \geq 0$  can be obtained. For the conditions  $|w_{\phi,j}^i| = D_{\mathbf{w}_\phi}$  and

$q\varphi(v_j) b_{ii} p_{ii} w_{\phi,j}^i e_i < 0$ ,

$$J_{\mathbf{w}_\phi} = \sum_{i,j} q\varphi(v_j) \frac{\tilde{w}_{\phi,j}^i w_{\phi,j}^i}{|w_{\phi,j}^i|^2} b_{ii} p_{ii} w_{\phi,j}^i e_i \quad (4-42)$$

can be obtained. Because  $\mathbf{w}_\phi^*$  belongs to the compact constraint set  $\Omega_{\mathbf{w}_\phi}$ , the inequality

$|w_{\phi,j}^{i*}| \leq D_{\mathbf{w}_\phi} = |w_{\phi,j}^i|$  can be obtained. The inequality equation then becomes



$\tilde{w}_{\varphi,j}^i w_{\varphi,j}^i = \frac{1}{2} (|w_{\varphi,j}^{i*}|^2 - |w_{\varphi,j}^i|^2 - |\tilde{w}_{\varphi,j}^i|^2) \leq 0$ . Thus, Equation (4-42) can be rewritten as

$$J_{w_\varphi} = \sum_{i,j} \frac{q\varphi(v_j)}{2} \cdot \frac{|w_{\varphi,j}^{i*}|^2 - |w_{\varphi,j}^i|^2 - |\tilde{w}_{\varphi,j}^i|^2}{|w_{\varphi,j}^i|^2} \cdot b_{ii} p_{ii} w_{\varphi,j}^i e_i \leq 0. \quad (4-43)$$

Similarly, by using (4-38) and (4-39), the following inequality equations can be obtained:

$$J_{w_u} = \sum_{i,j,k} \frac{u_k}{2} \cdot \frac{|w_{u,(n+k)}^{i*}|^2 - |w_{u,(n+k)}^i|^2 - |\tilde{w}_{u,(n+k)}^i|^2}{|w_{u,(n+k)}^i|^2} \cdot b_{ii} p_{ii} w_{u,(n+k)}^i e_i \leq 0 \quad (4-44)$$

and

$$J_{w_h} = \sum_{i,j,k,l} \frac{z_l}{2} \cdot \frac{|w_{h,(n+M+l)}^{i*}|^2 - |w_{h,(n+M+l)}^i|^2 - |\tilde{w}_{h,(n+M+l)}^i|^2}{|w_{h,(n+M+l)}^i|^2} \cdot b_{ii} p_{ii} w_{h,(n+M+l)}^i e_i \leq 0. \quad (4-45)$$

Hence, for any possible condition that occurs in (4-37)–(4-39), the conditions  $J_{w_\varphi} \leq 0$ ,

$J_{w_u} \leq 0$ , and  $J_{w_h} \leq 0$  can be satisfied. Then, Equation (4-41) can be simplified to

$$\begin{aligned} \dot{V} &= -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} + J_{w_\varphi} + J_{w_u} + J_{w_h} + \mathbf{e}^T \mathbf{P} \delta \\ &\leq -\frac{1}{2} \mathbf{e}^T \mathbf{Q} \mathbf{e} + \mathbf{e}^T \mathbf{P} \delta \\ &\leq -\frac{1}{2} [\lambda_{\min}(\mathbf{Q}) - 1] \|\mathbf{e}\|^2 + \frac{1}{2} \|\mathbf{P} \delta\|^2 \end{aligned} \quad (4-46)$$

where  $\lambda_{\min}(\mathbf{Q})$  is the minimum eigenvalue of  $\mathbf{Q}$ . Integrating both sides of (4-46) from  $t = 0$  to  $t = T$  ( $0 < T < \infty$ ), and choosing  $\lambda_{\min}(\mathbf{Q}) > 1$  (because  $\mathbf{Q}$  is determined by the designer), we have

$$\frac{1}{2} [\lambda_{\min}(\mathbf{Q}) - 1] \int_0^T \|\mathbf{e}\|^2 dt \leq -[V(t) - V(0)] + \frac{1}{2} \|\mathbf{P}\|^2 \int_0^T \|\delta\|^2 dt. \quad (4-47)$$

In particular,  $V(t) \geq 0$ , Equation (4-47) can be arranged as

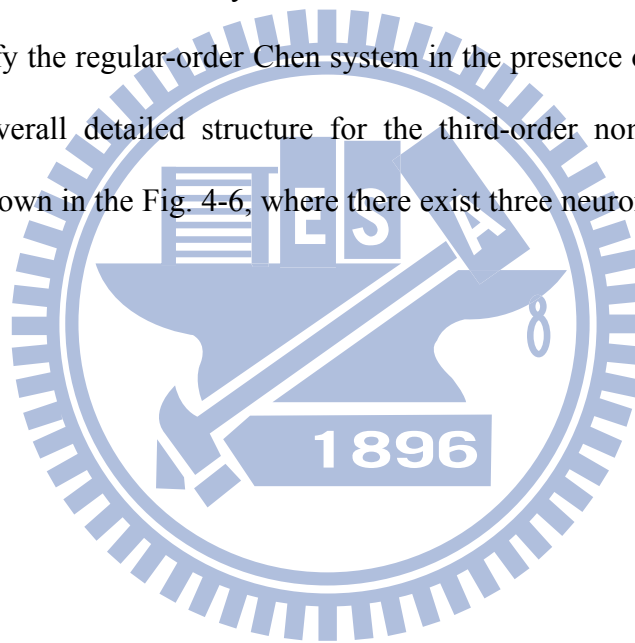
$$\int_0^T \|\mathbf{e}\|^2 dt \leq \frac{2}{\lambda_{\min}(\mathbf{Q}) - 1} V(0) + \frac{1}{\lambda_{\min}(\mathbf{Q}) - 1} \|\mathbf{P}\|^2 \int_0^T \|\delta\|^2 dt \quad (4-48)$$

where  $\frac{2}{\lambda_{\min}(\mathbf{Q}) - 1} V(0)$  and  $\frac{1}{\lambda_{\min}(\mathbf{Q}) - 1} \|\mathbf{P}\|^2$  are constants and  $\delta$  defined by (4-35) is a

maximum bounded for  $\mathbf{m}$ . If  $\delta$  is squared integrable, that is,  $\int_0^\infty |\delta(t)|^2 dt < \infty$ , then  $\lim_{T \rightarrow \infty} |e(t)| = 0$ . Therefore, from (4-48), the approximation performance with HOHNN identifier is proven to converge to a certain small boundary.

#### 4.6 Illustrated Examples

In this section, the simulation results of nonlinear system identification are presented. The uncontrolled regular-order Chen system [77, 78] simulated in Fig. 4-5 is the sort of stochastic fractional-order chaotic systems within 50 seconds. The application of proposed HOHNN to identify the regular-order Chen system in the presence of external disturbances is illustrated. The overall detailed structure for the third-order nonlinear dynamical system identification is shown in the Fig. 4-6, where there exist three neurons and one control force  $u$  for the HOHNN.



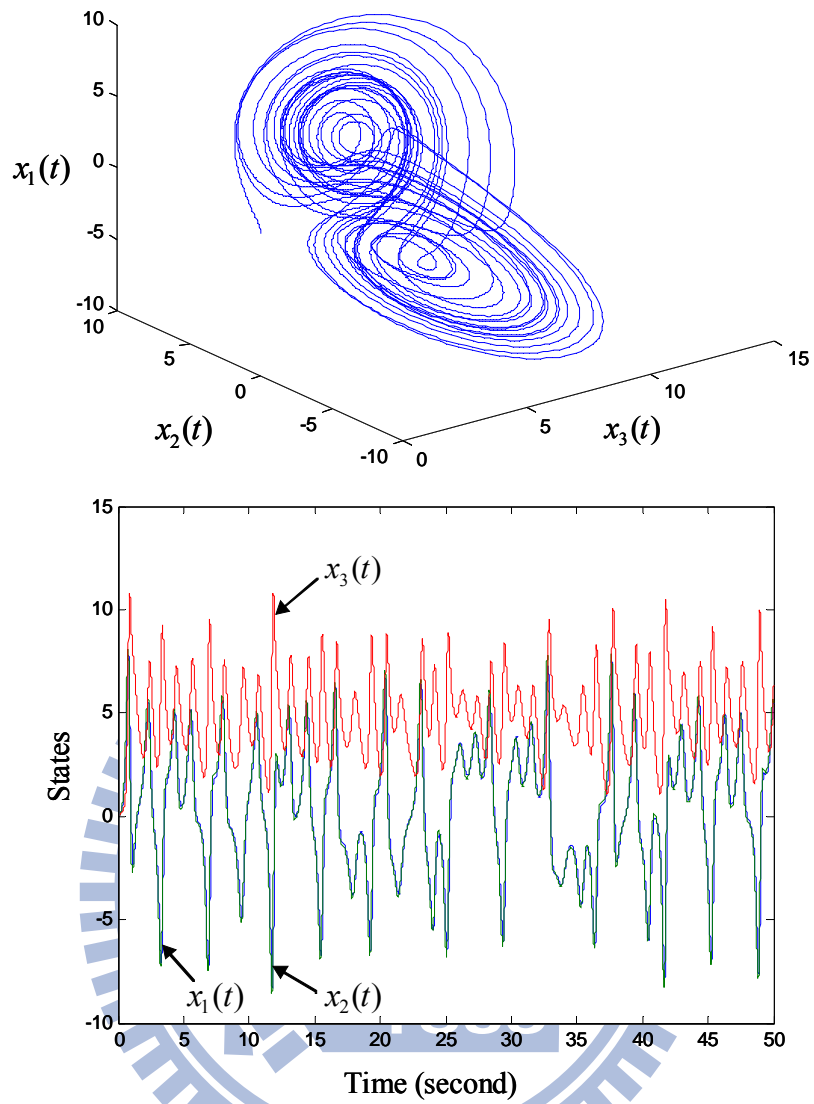


Fig. 4-5. The uncontrolled regular-order Chen system plotted in the (a)  $x_3$ - $x_2$ - $x_1$  space, and (b) time series within 50 seconds.

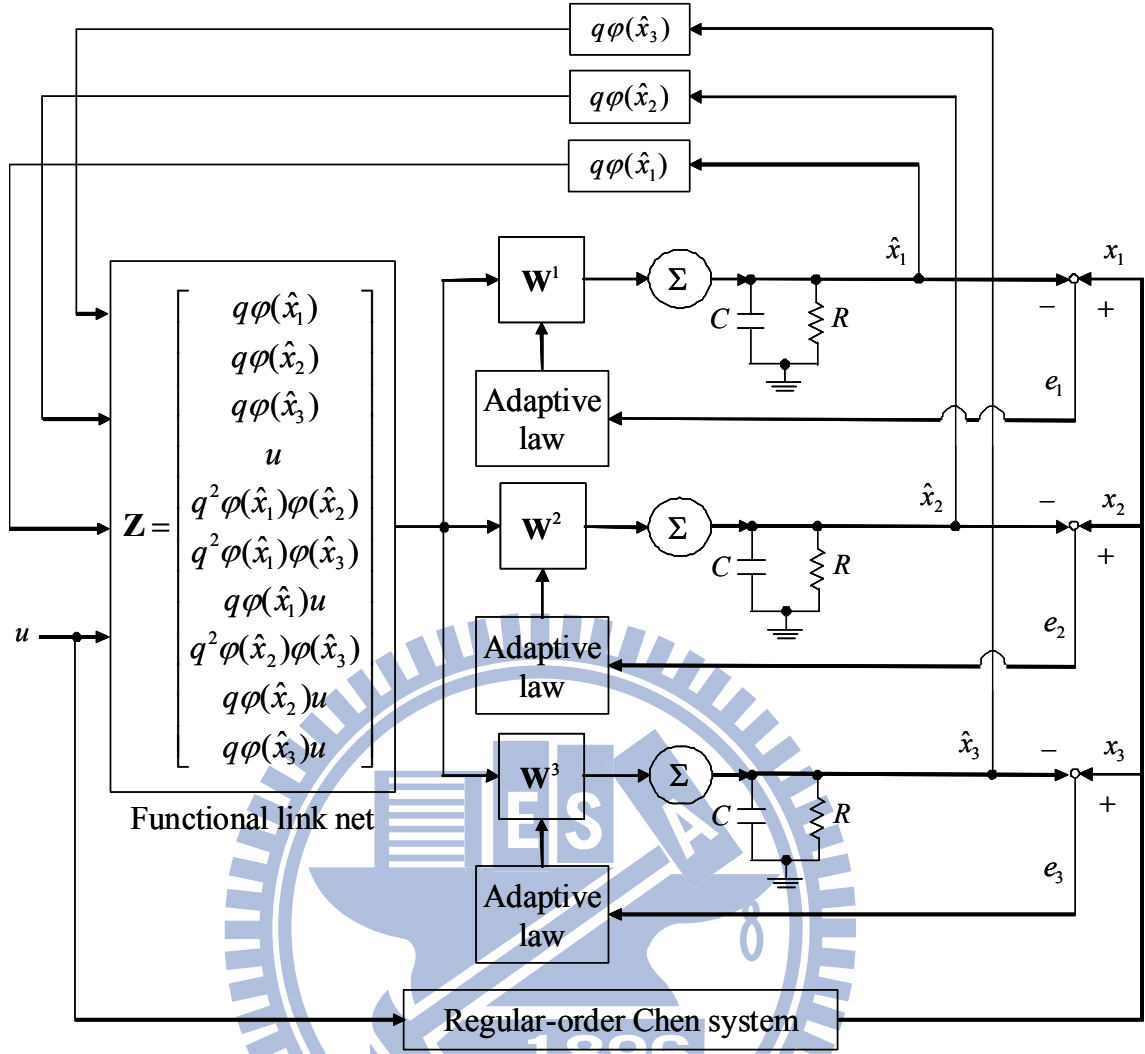


Fig. 4-6. The overall identification diagram of regular-order Chen system.

Assume the feedback constant  $q=1$ , the inputs from the network feedback vector  $\Phi = \{\varphi(\hat{x}_1) \ \varphi(\hat{x}_2) \ \varphi(\hat{x}_3)\}$  and the control force vector  $U = \{u\}$  are combined with the extra compact high-order term  $Z_h = \{\varphi(\hat{x}_1)\varphi(\hat{x}_2) \ \varphi(\hat{x}_1)\varphi(\hat{x}_3) \ \varphi(\hat{x}_1)u \ \varphi(\hat{x}_2)\varphi(\hat{x}_3) \ \varphi(\hat{x}_2)u \ \varphi(\hat{x}_3)u\}$  to form the full input vector  $Z = [\Phi \ U \ Z_h]^T$  for the Hopfield-based neural network. The dynamic motion equation of the regular-order Chen system in this thesis is designed as follows:

$$\dot{x}_1 = a(x_2 - x_1) + u + d \quad (4-49)$$

$$\dot{x}_2 = (c - a)x_1 - x_1x_3 + cx_2 + u + d \quad (4-50)$$

$$\dot{x}_3 = x_1 x_2 - b x_3 + u + d \quad (4-51)$$

where  $a = 21$ ,  $b = 2$ ,  $c = 13$ ;  $x_1$ ,  $x_2$ , and  $x_3$  are the state variables;  $u = 3\sin(1.5t)$  is chosen as the control force. In order to examine the robustness of the proposed scheme, an external disturbance  $d = 1.5\sin(3t) + 2\cos(2t)$  is added to the system after 10 seconds. In the simulation, the regular-order Chen system is shown in Fig. 4-6 with a sampling time of 0.005 second and within total simulation time of 15 seconds. In the three HOHNN neurons, the parameters are selected as  $R = 100\Omega$ ,  $C = 0.01F$ ; the initial voltages are set as  $v_1(0) = v_2(0) = v_3(0) = 1V$ , and the weighting factors are set as  $\mathbf{w}_\varphi(0) = 0.08$ ,  $\mathbf{w}_u(0) = 0.025$ , and  $\mathbf{w}_h(0) = 0.01$ , respectively. For comparison purpose, the recurrent HONN (RHONN), traditional HNN, and FHOHNN models have been adopted to show the approximation performance on the same regular-order Chen system.

Figure 4-7(a) shows the system approximation comparisons between these Hopfield-based neural networks. Figures 4-7(b) and (c) show the detailed performances for  $t = 0 \sim 0.5$  seconds and  $t = 9.9 \sim 10.4$  seconds, respectively. The error comparisons of system approximation are shown in Fig. 4-8(a), and the detailed approximation errors are shown in Figs. 4-8(b) and (c) for  $t = 0 \sim 0.5$  seconds and  $t = 9.9 \sim 10.4$  seconds, respectively. Although the detailed performance comparisons between the networks for  $t = 9.9 \sim 10.4$  seconds are not obvious, the detailed error comparisons can clarify that HOHNN can perform the best system identification above the other Hopfield-based neural networks. The mean square error comparisons for all the Hopfield-based neural networks and their detailed drawing for  $t = 0 \sim 0.5$  seconds and  $t = 9.9 \sim 10.4$  seconds are shown in Figs. 4-9(a), (b), and (c), respectively. It can be obviously seen from Figs. 4-7 to 4-9 show that the HOHNN structure behaves the best approximation performance among the other structures.

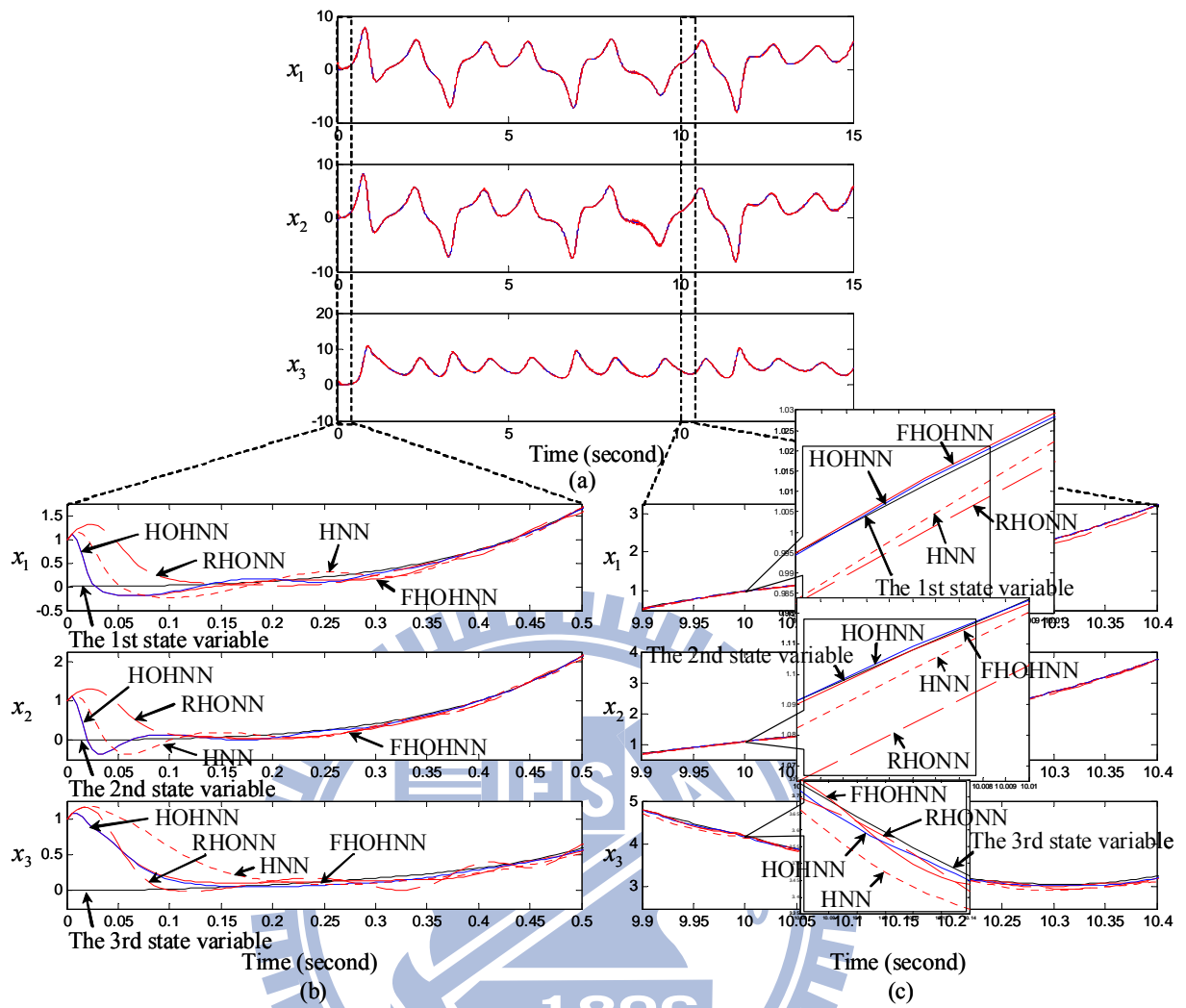


Fig. 4-7. The system approximation comparisons between RHONN, HNN, FHOHNN, and HOHNN (a) for  $t = 0 \sim 15$  seconds, and the detailed drawing for (b)  $t = 0 \sim 0.5$  seconds and (c)  $t = 9.9 \sim 10.4$  seconds.

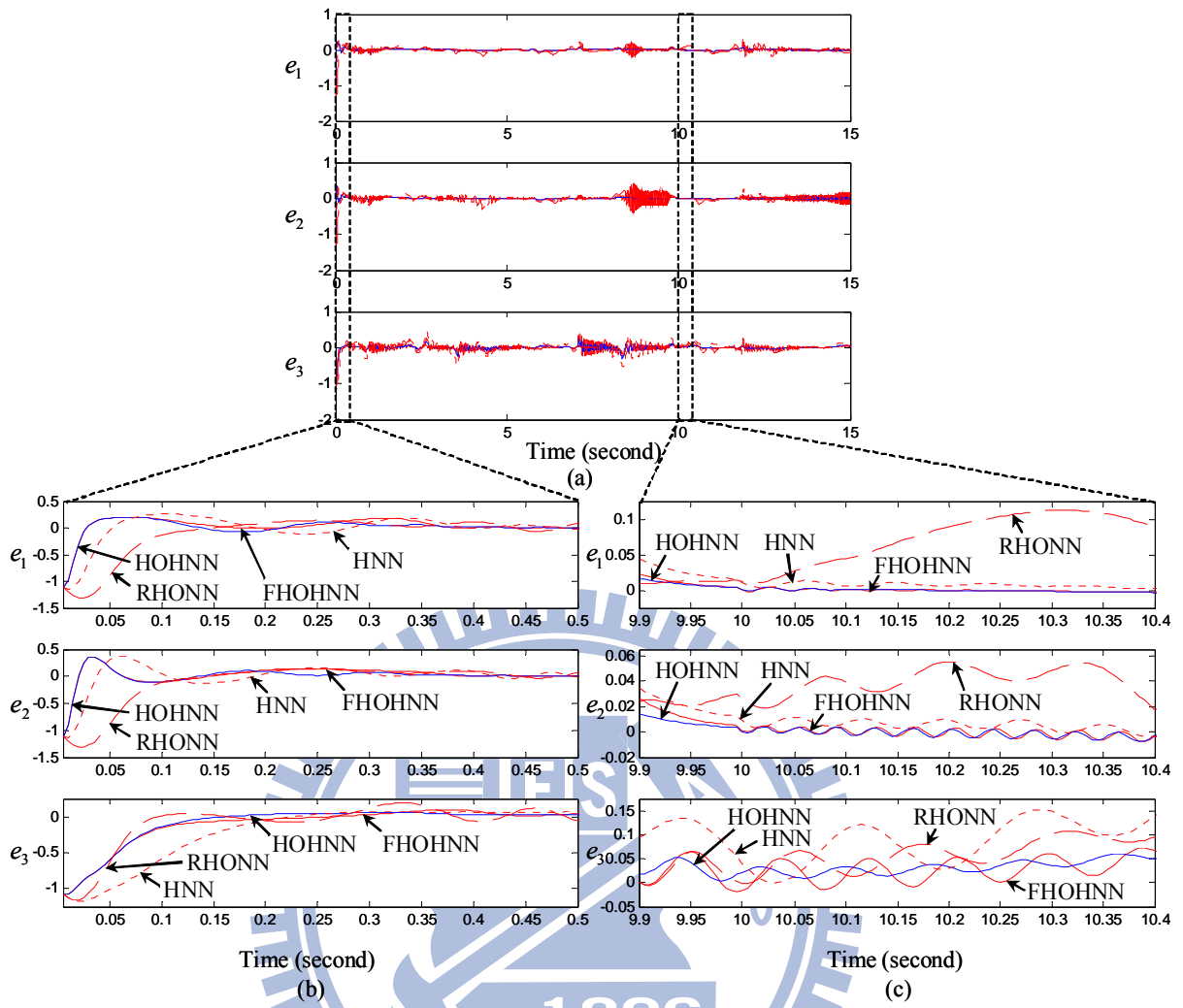


Fig. 4-8. The error comparisons of system approximation between RHONN, HNN, FHOHNN, and HOHNN (a) for  $t = 0 \sim 15$  seconds, and the detailed drawing for (b)  $t = 0 \sim 0.5$  seconds and (c)  $t = 9.9 \sim 10.4$  seconds.

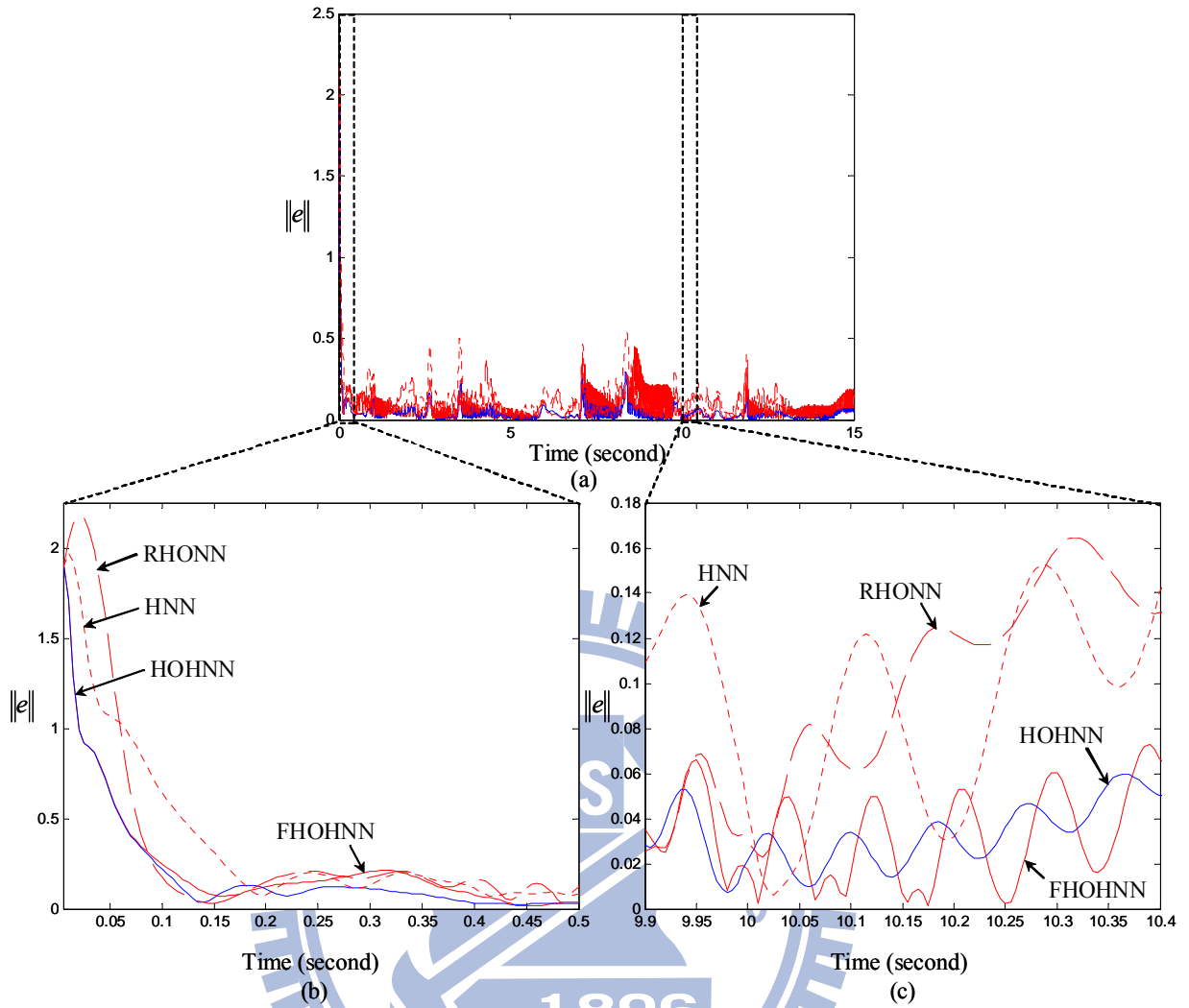


Fig. 4-9. The mean square error comparisons of system approximation between RHONN, HNN, FHOHNN, and HOHNN (a) for  $t = 0 \sim 15$  seconds, and the detailed drawing for (b)  $t = 0 \sim 0.5$  seconds and (c)  $t = 9.9 \sim 10.4$  seconds.

The following Table 4-2 shows a computation index that indicates the RHONN and FHOHNN structures perform longer execution time than the HNN and HOHNN structures. In the proposed HOHNN structure, there are four components considered in the original input pattern to produce ten enhanced input components; however, the same number of components is largely extended to sixteen and fourteen components in the RHONN and FHOHNN structures, respectively. This number of components does not only increase the execution time, but also decreases the tracking performance caused by the large number of redundant



weighting factors. Furthermore, the traditional HNN has the advantage of average execution time; however, it has the disadvantage of slowest approximation performance. Thus, the system identification of the proposed HOHNN is the best among those of the other Hopfield-type neural networks in simulation results and computation analysis within a proper given time interval, even under a certain external disturbance.

Table 4-2 Execution time for all the Hopfield-based neural networks.

Execution time (sec) \ Network architecture	Maximum	Minimum	Average
FHOHNN	0.2040	0.1870	0.1906
HOHNN	0.1870	0.1720	0.1765
HNN	0.1880	0.1560	0.1688
RHONN	0.2350	0.2190	0.2283

#### 4.7 Conclusions

In this thesis, HOHNN has been proposed for the unknown nonlinear dynamical system identification. In comparison with the non-ordered mathematical representation in FLN, a compact structure of FLN with systematic order mathematical representation has been combined into the proposed HOHNN. The approximation capability of HOHNN is first discussed to show that the proposed FLN is capable of approximating the behavior of dynamical systems to any degree of accuracy if a sufficiently large number of high-order connections between neurons is allowed. The adaptive laws via Lyapunov tuning theorem to the weighting factor matrix can reduce the approximation error to a small yet satisfactory level. In case modeling errors are present, the robust learning analysis is then proposed to guarantee the stability of the overall scheme. The simulation results for the RHONN, FHOHNN, HNN, and HOHNN are finally conducted to show the effectiveness of HOHNN in

uncertain dynamical system identification. The system identification performance for HOHNN is better than that of RHONN, FHOHNN, and HNN. Even when the dynamical system is perturbed by unwanted disturbance, the improved performance of the proposed HOHNN is evident in the computer simulation established with the benchmark examples in this chapter.



# Chapter 5

## Conclusions and Future Works

### 5.1 Conclusions

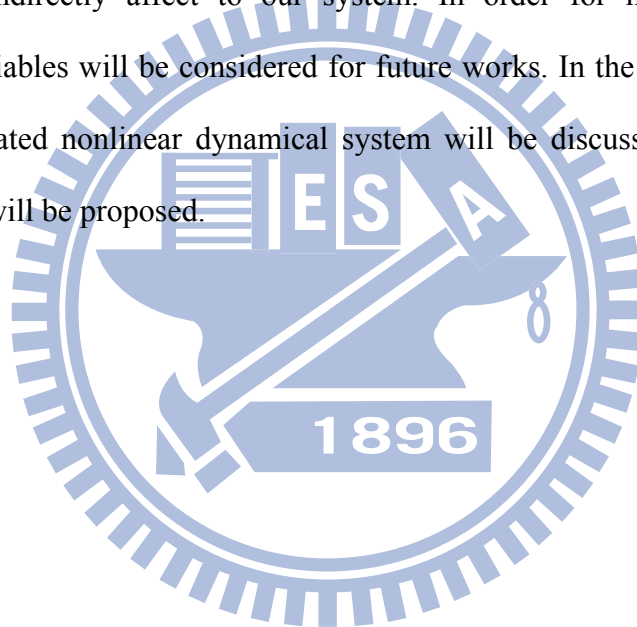
For decades, there have been many researches focused on one-to-one tracking control. The fuzzy neural network (FNN) controller proposed in Chapter 2 and 3 is easier to be implemented than the other complicated neural network controllers for guidance purpose. In the FNN controller, the Gaussian function is adopted as the membership function and the fuzzy operations are utilized as the inference mechanism. The online learning methodology is developed in the Lyapunov sense, meanwhile the stability of the closed-loop system can also be guaranteed. The multi-agent system (MAS) extends the one-to-one tracking control to an environment for the consensus and communication of a group of agents and targets. From the help of self-organizing map (SOM) in MAS, the task assignment between agents and targets can efficiently be handled within a satisfactory computational load. In Chapter 3, the missile defense system (MDS) is established for two case of simulations. The one-to-one agent-target missile guidance using FNN control is proposed in the first case; the multi-agent-multi-target battle scenarios are proposed in the second case. From the simulation results, not only the proposed FNN controller has better performance than the cerebellar model articulation controller (CMAC), but the SOM-based FNN controller for MDS can also completely establish a real-time battle environment.

In addition, the high-order Hopfield-based neural network (HOHNN) is proposed for the nonlinear dynamical system identification. The proposed functional link net (FLN) with a systematic mathematical representation for the input patterns is capable of approximating the behavior of nonlinear dynamical systems. The Lyapunov tuning theorem and the robust learning analysis for weighting factors can not only reduce the approximation error to a small

satisfactory level, but the stability of the overall closed-loop system can also be guaranteed. From the simulation results and computational analysis, the system identification performance for HOHNN is better than that of the other Hopfield-based neural networks even under disturbances.

## 5.2 *Suggestions for Future Works*

In the first part of the thesis, the agents are located on the fixed positions. Because in the real missile battle scenario, there are unknown number of objects and parameters which can directly or indirectly affect to our system. In order for missile battle reality, the corresponding variables will be considered for future works. In the second part of the thesis, the more complicated nonlinear dynamical system will be discussed and a more complete stability theorem will be proposed.



## References

- [1] S.-K. Elham and K. Khorasani, "Optimal consensus algorithms for cooperative team of agents subject to partial information," *Automatica*, vol. 44, no. 11, pp. 2766–2777, 2008.
- [2] L. Cheng, Z.-G. Hou, and M. Tan, "Decentralized robust adaptive control for the multiagent system consensus problem using neural networks," *IEEE Trans. on Systems, Man, and Cybernetics–Part B: Cybernetics*, vol. 39, no. 3, pp. 636–647, 2009.
- [3] Y.-G. Sun and L. Wang, "Consensus of multi-agent systems in directed networks with nonuniform time-varying delays," *IEEE Trans. on Automatic Control*, vol. 54, no. 7, pp. 1607–1613, 2009.
- [4] W. Ren, "Distributed cooperative attitude synchronization and tracking for multiple rigid bodies," *IEEE Trans. on Control Systems Technology*, vol. 19, no. 2, pp. 383–392, 2010.
- [5] L. Cheng, Z.-G. Hou, M. Tan, Y. Lin, and W. Zhang, "Neural-network-based adaptive leader-following control for multiagent systems with uncertainties," *IEEE Trans. on Neural Networks*, vol. 21, no. 8, pp. 1351–1358, 2010.
- [6] S.-K. Elham and K. Khorasani, "Team consensus for a network of unmanned vehicles in presence of actuator faults," *IEEE Trans. on Control Systems Technology*, vol. 18, no. 5, pp. 1155–1161, 2010.
- [7] X. Liu, W. Lu, and T. Chen, "Consensus of multi-agent systems with unbounded time-varying delays," *IEEE Trans. on Automatic Control*, vol. 55, no. 10, pp. 2396–2401, 2010.
- [8] E. Nuno, R. Ortega, L. Basanez, and D. Hill, "Synchronization of networks of nonidentical Euler-Lagrange systems with uncertain parameters and communication delays," *IEEE Trans. on Automatic Control*, vol. 56, no. 4, pp. 935–941, 2011.
- [9] N. Papadoglou and E. Stipidis, "Investigation for a global AVL system," *IEEE Trans. on*

*Intelligent Transportation Systems*, vol. 2, no. 3, pp. 121–126, Sep. 2001.

- [10] V. de Nitto Persone and V. Grassi, “Performance analysis of caching and prefetching strategies for palmtop-based navigational tools,” *IEEE Trans. on Intelligent Transportation Systems*, vol. 4, no. 1, pp. 23–34, Mar. 2003.
- [11] S. Kim, M.-E. Lewis, I. Chelsea, and C. White, “Optimal vehicle routing with real-time traffic information,” *IEEE Trans. on Intelligent Transportation Systems*, vol. 6, no. 2, pp. 178–188, Jun. 2005.
- [12] K. Dorer and M. Calisti, “An adaptive solution to dynamic transport optimization,” *Proc. 4<sup>th</sup> Int. Joint Conf. Autonomous Agents and Multi-Agent Systems (AAMAS’05): Industry Track*, Utrecht, The Netherlands, pp. 45–51, Jul. 2005.
- [13] Z. Liao, “Taxi dispatching via global positioning systems,” *IEEE Trans. on Engineering Management*, vol. 48, no. 3, pp. 342–347, Aug. 2001.
- [14] K.-T. Seow, N.-H. Dang, and D.-H. Lee, “A collaborative multiagent taxi-dispatch system,” *IEEE Trans. on Automation Science and Engineering*, vol. 7, no. 3, pp. 607–616, July 2010.
- [15] P.-C. Chu and J.-E. Beasley, “A genetic algorithm for the generalized assignment problem,” *Computers and Operations Research*, vol. 24, no. 1, pp. 17–23, 1997.
- [16] R. Akkiraju, P. Keskinocak, S. Murthy, and F. Wu, “An agent-based approach for scheduling multiple machines,” *Applied Intelligence*, vol. 14, pp. 135–144, 2001.
- [17] A.-J. Higgins, “A dynamic tabu search for large-scale generalized assignment problems,” *Computers and Operations Research*, vol. 28, pp. 1039–1048, 2001.
- [18] K.-S. Kwok, B.-J. Driessen, and C.-A. Phillips, “Analyzing the multiple-target-multiple-agent scenario using optimal assignment algorithms,” *Journal of Intelligent and Robotic Systems*, vol. 35, pp. 111–122, 2002.
- [19] T. Kohonen, *Self-Organizing Maps*, 2nd ed. New York: Springer-Verlag, 1997.
- [20] K.-J. Kim and S.-B. Cho, “Fuzzy integration of structure adaptive SOMs for web

- content mining,” *Fuzzy Sets and Systems*, vol. 148, pp. 43–60, 2004.
- [21] M.-S. Yang, and W.-L. Hung, D.-H. Chen, “Self-organizing map for symbolic data,” *Fuzzy Sets and Systems*, vol. 203, pp. 49–73, 2012.
- [22] R. Rathi, M. Choudhary, and B. Chandra, “An Application of Face Recognition System using Image Processing,” *Int. Joint Comp. Tech. Appl.*, vol. 3, no. 1, pp. 45–49, 2012.
- [23] A. Zhu and S.-X. Yang, “A neural network approach to dynamic task assignment of multirobots,” *IEEE Trans. on Neural Networks*, vol. 17, no. 5, pp. 1278–1287, September 2006.
- [24] H. Kusumoto and Y. Takefuji, “ $O(\log_2 M)$  Self-Organizing Map Algorithm Without Learning of Neighborhood Vectors,” *IEEE Trans. on Neural Networks*, vol. 17, no. 6, pp. 1656–1661, November 2006.
- [25] D. Wang and J. Huang, “Neural network-based adaptive dynamic surface control for a class of uncertain nonlinear systems in strict-feedback form,” *IEEE Trans. on Neural Networks*, vol. 16, pp. 195–202, 2005.
- [26] K.-P. Tee, S.-S. Ge, and F. E. H. Tay, “Adaptive Neural Network Control for Helicopters in Vertical Flight,” *IEEE Trans. on Control Systems Technology*, vol. 16, no. 4, pp. 753–762, July 2008.
- [27] Z. Sun and A.-K. Sen, “Neural Network Control of Resistive Wall Modes in Tokamaks,” *IEEE Trans. on Plasma Science*, vol. 38, no. 11, pp. 3226–3233, November, 2010.
- [28] H.-D. Patino, R. Carelli, and B.-R. Kuchen, “Neural networks for advanced control of robot manipulators,” *IEEE Trans. on Neural Networks*, vol. 13, no. 2, pp. 343–354, Mar. 2002.
- [29] S.-X. Yang and M. Q. H. Meng, “Real-time collision-free motion planning of a mobile robot using a neural dynamics-based approach,” *IEEE Trans. on Neural Networks*, vol. 14, no. 6, pp. 1541–1552, Nov. 2003.
- [30] R. Reeve and J. Hallam, “An analysis of neural models for walking control,” *IEEE*

*Trans. on Neural Networks*, vol. 16, no. 3, pp. 733–742, May 2005.

- [31] C.-T. Lin and C. S. G. Lee, *Neural Fuzzy Systems: A Neural-Fuzzy Synergism to Intelligent Systems*, Englewood Cliffs, NJ: Prentice-Hall, 1996.
- [32] X. Deng and X. Wang, “Incremental learning of dynamic fuzzy neural networks for accurate system modeling,” *Fuzzy Sets and Systems*, vol. 160, pp. 972–987, 2009.
- [33] D. Lin and X. Wang, “Observer-based decentralized fuzzy neural sliding mode control for interconnected unknown chaotic systems via network structure adaptation,” *Fuzzy Sets and Systems*, vol. 161, pp. 2066–2080, 2010.
- [34] C.-F. Juang, Y.-Y. Lin, and C.-C. Tu, “A recurrent self-evolving fuzzy neural network with local feedbacks and its application to dynamic system processing,” *Fuzzy Sets and Systems*, vol. 161, pp. 2552–2568, 2010.
- [35] Y. Yu, C.-L. Hui, T.-M. Choi, and R. Au, “Intelligent Fabric Hand Prediction System With Fuzzy Neural Network,” *IEEE Trans. on Systems, Man, and Cybernetics-part C: Applications and Reviews*, vol. 40, no. 6, pp. 619–629, November 2010.
- [36] C.-F. Juang, T.-C. Chen, and W.-Y. Cheng, “Speedup of Implementing Fuzzy Neural Networks With High-Dimensional Inputs Through Parallel Processing on Graphic Processing Units,” *IEEE Trans. on Fuzzy Systems*, vol. 19, no. 4, pp. 717–728, August 2011.
- [37] V. Gazi, “Swarm aggregations using artificial potentials and sliding-mode control,” *IEEE Trans. on Robotics*, vol. 21, no. 6, pp. 1208–1214, 2005.
- [38] S. Haykin, *Neural Networks*, Upper Saddle River, NJ: Prentice-Hall, 1999.
- [39] J. J. E. Slotine, and W. Li, *Applied Nonlinear Control*, Englewood Cliffs, NJ: Prentice-Hall, 1991.
- [40] L.-X. Wang, *Adaptive Fuzzy Systems and Control: Design and Stability Analysis*, Englewood Cliffs, NJ: Prentice-Hall, 1994.
- [41] D. W. C. Ho, P.-A. Zhang, and J. Xu, “Fuzzy Wavelet Networks for Function Learning,”



- IEEE Trans. on Fuzzy Systems*, vol. 9, no. 1, pp. 200–211, 2001.
- [42] K.-J. Astrom and B. Wittenmark, *Adaptive Control*, Reading, MA: Addison-Wesley, 1995.
- [43] D. P. Bertsekas, Mark L. Homer, David A. Logan, Stephen D. Patek, and Nils R. Sandell, “Missile Defense and Interceptor Allocation by Neuro-Dynamic Programming,” *IEEE Trans. on Systems, Man, and Cybernetics–Part A: Systems and Humans*, vol. 30, no. 1, pp. 42–51, January 2000.
- [44] I. J. Ha and S. Chong, “Design of a CLOS guidance law via feedback linearization,” *IEEE Trans. on Aerospace and Electronic Systems*, vol. 28, no. 1, pp. 51–63, Jan. 1992.
- [45] C. Y. Kuo, D. Soetanto, and Y. C. Chiou, “Geometric Analysis of Flight Control Command for Tactical Missile Guidance,” *IEEE Trans. on Control Systems Technology*, vol. 9, no. 2, pp. 234–243, 2001.
- [46] C.-M. Lin and Y.-F. Peng, “Missile Guidance Law Design Using Adaptive Cerebellar Model Articulation Controller,” *IEEE Trans. on Neural Networks*, vol. 16, no. 3, pp. 636–644, May 2005.
- [47] Y. Oshman and D. Arad, “Differential-Game-Based Guidance Law using Target Orientation Observations,” *IEEE Trans. on Aerospace and Electronic Systems*, vol. 42, no. 1, pp. 316–326, 2006.
- [48] M. Innocenti, L. Pollini, and D. Turra, “A Fuzzy Approach to the Guidance of Unmanned Air Vehicles Tracking Moving Targets,” *IEEE Trans. on Control Systems Technology*, vol. 16, no. 6, pp. 1125–1137, 2008.
- [49] H. Yan and H.-B. Ji, “Guidance Laws Based on Input-to-State Stability and High-Gain Observers,” *IEEE Trans. on Aerospace and Electronic Systems*, vol. 48, no. 3, pp. 2518–2529, 2012.
- [50] J. J. Hopfield, “Neural networks and physical systems with emergent collective computational abilities,” *Proc. National Academy of Sciences*, vol. 79, pp. 2554–2558,

1982.

- [51] Z.-G. Hou, M. M. Gupta, P. N. Nikiforuk, M. Tan, and L. Cheng, "A recurrent neural network for hierarchical control of interconnected dynamic systems," *IEEE Trans. on Neural Networks*, vol. 18, no. 2, pp. 466–481, 2007.
- [52] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Computation*, vol. 1, pp. 270–280, 1989.
- [53] T. W. S. Chow, X.-D. Li, and Y. Fang, "A real-time learning control approach for nonlinear continuous-time system using recurrent neural networks," *IEEE Trans. on Industrial Electronics*, vol. 47, no. 2, pp. 478–486, 2000.
- [54] Y. Zhang and S. S. Ge, "Design and analysis of a general recurrent neural network model for time-varying matrix inversion," *IEEE Trans. on Neural Networks*, vol. 16, no. 6, pp. 1477–1490, 2005.
- [55] D.-L. Lee, "Pattern sequence recognition using a time-varying Hopfield network," *IEEE Trans. on Neural Networks*, vol. 13, no. 2, pp. 330–342, 2002.
- [56] Z. Ma and A. Jutan, "Control of a pressure tank system using a decoupling control algorithm with a neural network adaptive scheme," *IEE Trans. on Control Theory*, vol. 150, no. 4, pp. 389–400, 2003.
- [57] Y. Li, Z. Tang, G.-P. Xia, R.-L. Wang, "A positively self-feedbacked Hopfield neural network architecture for crossbar switching," *IEEE Trans. on Circuits and Systems-part I: Regular papers*, vol. 52, no. 1, pp. 200–206, 2005.
- [58] Y. H. Pao, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, 1989.
- [59] Y. H. Pao, *Functional-Link Net Computing: Theory, System Architecture, and Functionalities*, Computer, 1992.
- [60] G. H. Park, Y. H. Pao, "Unconstrained word-based approach for offline script recognition using density-based random-vector functional-link net," *Neurocomputing*,

vol. 31, no. 1–4, pp. 45–65, 2000.

- [61] B.-S. Lin, B.-S. Lin, F.-C. Chong, and F. Lai, “A functional link network with higher order statistics for signal enhancement,” *IEEE Trans. on Signal Processing*, vol. 54, no. 12, pp. 4821–4826, 2006.
- [62] H. Zhao and J. Zhang, “Pipelined Chebyshev functional link artificial recurrent neural network for nonlinear adaptive filter,” *IEEE Trans. on Systems Man Cybernetics, Part B, Cybernetics*, vol. 40, no. 1, pp. 162–172, 2010.
- [63] I. Hassanzadeh, S. Khanmohammadi, J. Jiang, and G. Alizadeh, “Implementation of a functional link net-ANFIS controller for a robot manipulator,” *Proc. International Workshop on Robot Motion and Control*, pp. 399–404, 2002.
- [64] C. L. Giles and T. Maxwell, “Learning, invariance, and generalization in higher-order neural networks,” *Applied Optics*, no. 26, pp. 4972–4978, 1987.
- [65] M. Klassen, Y. H. Pao, and V. Chen, “Characteristics of the functional-link net: A higher order delta rule net,” *Proc. IEEE Annual International Conf. on Neural Networks*, San Diego, CA, USA, pp. 507–513, July 1988.
- [66] A. Sierra, J. A. Macias, and F. Corbacho, “Evolution of functional link networks,” *IEEE Trans. on Evolutionary Computation*, vol. 5, no. 1, pp. 54–65, 2001.
- [67] J. C. Patra, R. N. Pal, B. N. Chatterji, and G. Panda, “Identification of nonlinear dynamic systems using functional link artificial neural networks,” *IEEE Trans. on Systems Man Cybernetics, Part B, Cybernetics*, vol. 29, no. 2, pp. 254–262, 1999.
- [68] G. A. Barreto and A. F.R. Araújo, “Identification and control of dynamical systems using the self-organizing map,” *IEEE Trans. on Neural Networks*, vol. 15, no. 5, pp. 1244–1259, 2004.
- [69] Z. Xiang and X. Deyun, “Fault diagnosis based on the fuzzy-recurrent neural network,” *Asian Journal of Control*, vol. 3, no. 2, pp. 89–95, 2001.
- [70] C.-H. Wang and K.-N. Hung, “High-order Hopfield-based neural network for nonlinear

system identification,” *IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio, TX, USA, pp. 3346–3351, 2009.

- [71] J. C. Patra and A. C. Kot, “Nonlinear dynamic system identification using Chebyshev functional link artificial neural networks,” *IEEE Trans. on Systems Man Cybernetics, Part B, Cybernetics*, vol. 32, no. 4, pp. 505–511, 2002.
- [72] Y.-C. Hu and F.-M. Tseng, “Functional-link net with fuzzy integral for bankruptcy prediction,” *Neurocomputing*, vol. 70, pp. 2959–2968, 2007.
- [73] C.-H. Chen, C.-J. Lin, and C.-T. Lin, “A functional-link-based neurofuzzy network for nonlinear system control,” *IEEE Trans. on Fuzzy Systems*, vol. 16, no. 5, pp. 1362–1378, 2008.
- [74] G. A. Rovithakis, “Tracking control of multi-input affine nonlinear dynamical systems with unknown nonlinearities using dynamical neural networks,” *IEEE Trans. on Systems Man Cybernetics, Part B, Cybernetics*, vol. 29, no. 2, pp. 179–189, 1999.
- [75] E. B. Kosmatopoulos, M. M. Polycarpou, M. A. Christodoulou, and P. A. Ioannou, “High-order neural network structures for identification for dynamical systems,” *IEEE Trans. on Neural Networks*, vol. 6, no. 2, pp. 422–431, 1995.
- [76] J. K. Hale, *Ordinary Differential Equations*, New York: Wiley, 1969.
- [77] C. Li and G. Chen, “Chaos and hyperchaos in fractional-order Rössler equations,” *Phys. A*, vol. 341, pp. 55–61, 2004.
- [78] J. G. Lu and G. Chen, “A note on the fractional-order Chen system,” *Chaos, Solitons, Fractals*, vol. 27, pp. 685–688, 2006.

# Vita

**Name:** Kun-Neng Hung

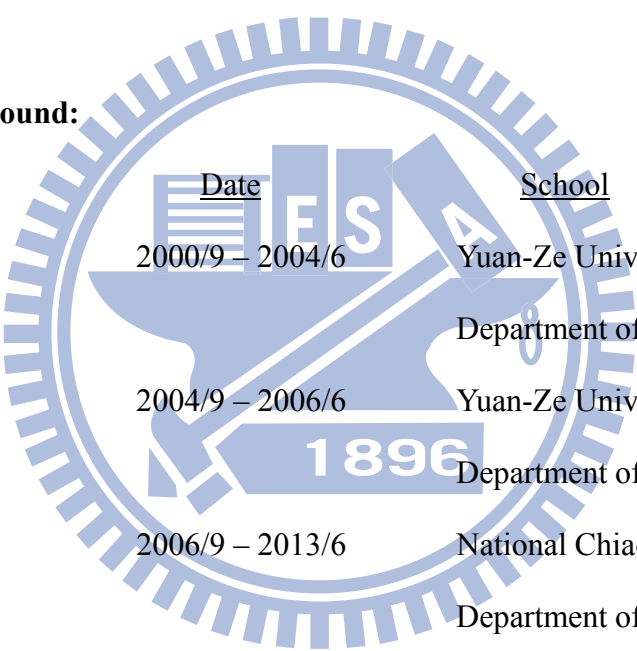
## Personal Details:

Place of Birth: Tainan, Taiwan, R.O.C.

Day of Birth: January 19, 1982

Gender: Male

## Education Background:



<u>Degree</u>	<u>Date</u>	<u>School</u>
B.S. E.E.	2000/9 – 2004/6	Yuan-Ze University, Department of Electrical Engineering
M.S. E.E.	2004/9 – 2006/6	Yuan-Ze University, Department of Electrical Engineering
Ph.D. E.E.	2006/9 – 2013/6	National Chiao Tung University, Department of Electrical Engineering

## Advisor(s):

M.S. –

Professor Chih-Min Lin, Yuan-Ze University

Ph.D. –

Professor Chi-Hsu Wang, National Chiao Tung University

## Publication List

### **Accepted Journal Papers:**

- [1] Chi-Hsu Wang, and Kun-Neng Hung, “Intelligent Adaptive Law for Missile Guidance Using Fuzzy Neural Networks,” *International Journal of Fuzzy Systems*, 2013.
- [2] Chi-Hsu Wang, and Kun-Neng Hung, “Dynamical System Identification using High-Order Hopfield-based Neural Network (HOHNN),” *Asian Journal of Control*, vol. 14, no. 6, pp. 1-14, 2012.

### **Submitted Journal Papers:**

- [1] Chi-Hsu Wang, and Kun-Neng Hung, “Dynamic Task Assignment with Path Control for Multi-Agent System using Intelligent Adaptive SOM-Based Fuzzy Neural Network,” submitted to *Fuzzy Sets and Systems*, Nov. 2012.
- [2] Chi-Hsu Wang, and Kun-Neng Hung, “Toward a New Task Assignment and Path Evolution (TAPE) for Missile Defense System (MDS) using Intelligent Adaptive SOM with Fuzzy Neural Networks,” submitted to *IEEE Trans. on Systems, Man, and Cybernetics—Part B: Cybernetics*, Oct. 2012.

### **International Conference Papers:**

- [1] Chi-Hsu Wang, and Kun-Neng Hung, “Adaptive SOM-Based Fuzzy Neural Network Controller Design for Multi-Agent System Dispatching and Path Planning,” *IEEE World Congress on Computational Intelligence*, Brisbane, Australia, pp. 1-7, June 2012.
- [2] Chi-Hsu Wang, and Kun-Neng Hung, “Adaptive High-Order Hopfield-based Neural Network Tracking Controller for Uncertain Nonlinear Dynamical System,” *IEEE International Conference on Networking, Sensing, and Control*, Chicago, IL, USA, pp. 382-387, April 2010.
- [3] Chi-Hsu Wang, and Kun-Neng Hung, “High-Order Hopfield-based Neural Network for Nonlinear System Identification,” *IEEE International Conference on Systems, Man, and Cybernetics*, San Antonio, Texas, USA, pp. 3346-3351, Oct. 2009.

### **國內研討會論文：**

- [1] Chi-Hsu Wang, and Kun-Neng Hung, “Direct Adaptive Control Design using High-Order Hopfield-based Neural Network for Affine Nonlinear System,” *National Conference on Fuzzy Theory and Its Applications*, Kaohsiung, Taiwan, pp. 717-722, Dec. 2009.