

行政院國家科學委員會補助專題研究計畫 ☒ 成果報告
☐ 期中進度報告

泛用型動態虛擬實境操控與運動復健輔助系統研發 —

子計畫 一：動態 VR 運動復健輔助系統之行為轉換與即時模擬研究

計畫類別：☐ 個別型計畫 ☒ 整合型計畫

計畫編號：NSC92-2213-E-009-016

執行期間： 90 年 8 月 1 日 至 93 年 7 月 31 日

計畫主持人：李祖添 講座教授

共同主持人：蘇順豐 教授

計畫參與人員：陳松雄、張原彰、陳政達、王彥評、林君彥、游雅婷、李世翔、劉家和、張富傑、李明橋

成果報告類型(依經費核定清單規定繳交)：☐ 精簡報告 ☒ 完整報告

本成果報告包括以下應繳交之附件：

- ☐ 赴國外出差或研習心得報告一份
- ☐ 赴大陸地區出差或研習心得報告一份
- ☐ 出席國際學術會議心得報告及發表之論文各一份
- ☐ 國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢

☐ 涉及專利或其他智慧財產權，☐ 一年 ☐ 二年後可公開查詢

執行單位：國立交通大學

中 華 民 國 93 年 7 月 日

子計畫 一：動態 VR 運動復健輔助系統之行為轉換與即時模擬研究

計畫編號: NSC90-2213-E-009-104; NSC91-2213-E-009-036;

執行期限: 90/08/01 - 93/07/31

主持人: 李祖添 講座教授; 共同主持人: 蘇順豐 教授

參與人員: 陳松雄、張原彰、陳政達、王彥評、林君彥、游雅婷、李世翔、劉家和、
張富傑

國立交通大學電機與控制系、國立臺灣科技大學電機系

一、計畫概述

在本計畫研發的系統中，操控員透過操控界面下達指令，以操控系統中的虛擬的運動訓練的輔助設備或載具。此操控命令將被輸入所模擬之設備或載具的精確物理模型中，以求得真實情況下系統的反應(包括姿態、速度、加速度、力道等)。這些反應將透過本子計畫的行為轉換與控制模組，而由六軸運動平台、力迴饋模組(Force feedback)及虛擬實境顯示器表現出來，以讓操控員獲得身歷其境的感受。這整個系統中，因為行為轉換模組的設計扮演實際環境裡物體運動狀況以及虛擬環境中操控員身體感受之間橋樑，如何讓操控員有身歷其境的感受，有賴於在動態模擬系統發展過程中，適當的人(操控員)機(動態模擬器)溝通界面及迴饋學習技術，以獲得最符合人類感覺的行為轉換模式。

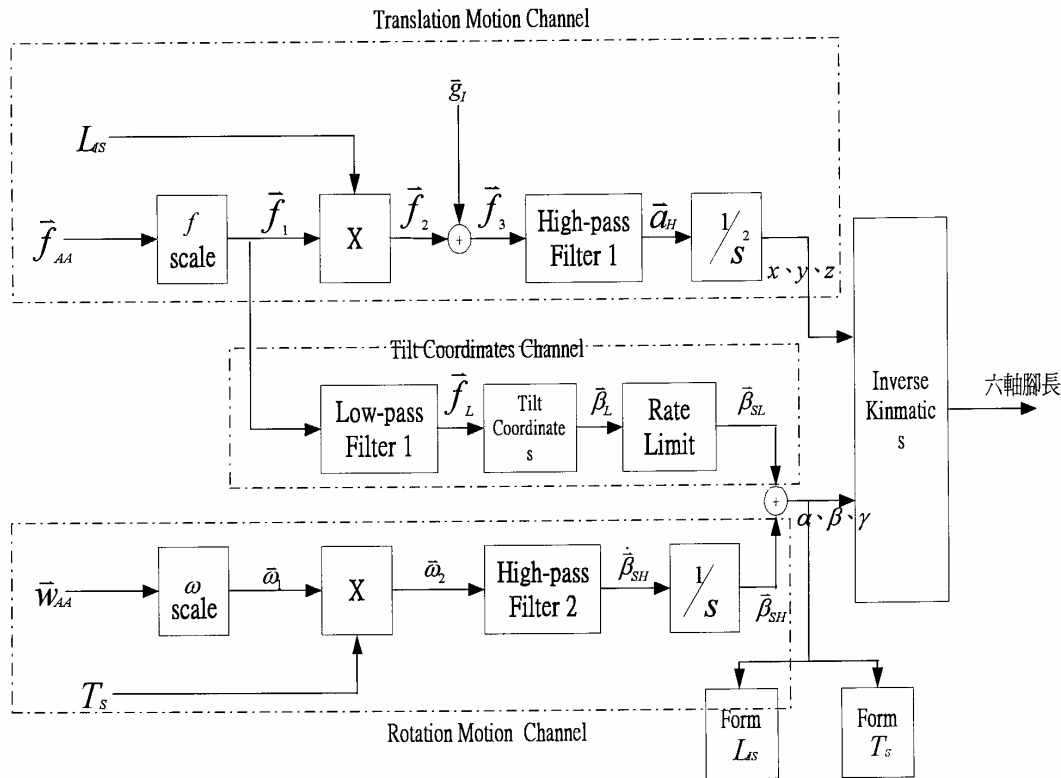
在虛擬實境的運動訓練的輔助動態模擬系統中，當虛擬系統接收到控制命令，而設法產生應有的運動行為描述時，在虛擬實境的顯示以及在模擬運動平台的運動行為，則是要儘可能地模擬並使其有身歷其境的感受，以達到虛擬實境的目的。文獻上之動態模擬有固定平台及動作平台二類，動作平台大部分是在飛機模擬用，動作平台之動作方式若不符合真實情形，比不用還要糟糕。而為了達到此一目標，上述的運動行為描述必須恰當的轉換給虛擬實境運動平台的控制系統。對虛擬實境的顯示而言，由於在顯示部分其為一動畫系統；並在理想上，將盡可能的和真實系統貼近，而且所有之運動行為都是以顯示的方式來呈現，並且由於虛擬實境的顯示的產生是可以由系統再修正，因此在運動行為的呈現上較容易。因此，本子計畫只要將相關之運動行為提供給顯示子系統即可。Stewart Platform 為一具有六自由度運動能力之機械平台，但其運動範圍受到其工作空間的限制，有些動作是無法達成的，因此為了在其有限空間下模擬出騎馬、飛機、輪船等交通工具時之運動感覺。所以一般是以沖淡演算法(Washout Algorithm)架構出對模擬器的力的感受。

本子計畫主要在於建立所欲模擬之場景及運動項目的精確物理模型，以便當接受子計畫三所傳送之使用者操控指令時，能精確獲得被模擬之運動項目在真實環境下的反應，而經由行為轉換與控制送至子計畫二，在虛擬實境中展現出來。因此本子計畫相當於整個動態虛擬實境運動復健輔助系統的輸入單元。在本研究中，由於真實系統並不是在旁邊實際動作，因此如何真實地精細架構真實系統之模式，以產生逼真的系統反應行為於虛擬實境的動態模擬器中，便是個非常重要的課題。同時由於希望能更精確的反應出行為來，傳統的模式建立可能無法達成如此精確的要求，因此本子計畫將研究如何地綜合利用物理運動定律，輸出、入行為，以及利用子計畫五之技術所擷取的人為修正訊息等，來精確地架構出真實系統模式。本子計畫的另一個研究主題為如何將真實系統之運動行為轉換成模擬器的控制命令，以及如何將無限空間中之運動感受，轉換成有限空間中運動的感受(如在加速度前進所感受之反作用等)。因此本子計畫扮演本計畫中真實世界與虛擬世界的橋樑，它將接受即時模擬單元來的行為狀態，再參考子計畫二所提供的六軸動作平台姿態訊息，以及子計畫二、五所感測的使用者運動狀況資訊後，對子計畫二、三、四發出高階的控制命令，以便使用者可以感受運動項目的姿態變化、力迴饋及場景更新。在模擬系統建立過程中，本子計畫也將接受經由子計畫五來的使用者感受反應而調整其行為轉換及控制方式。

二、沖淡演算法級行為轉換演算法

本子計畫主要在於當虛擬系統接收到控制命令，設法產生應有的運動行為描述以及在模擬運動平台的運動行為，以達到虛擬實境的目的。目前文獻上對於如是的轉換是以 Washout 演算法或稱 Washout Filter 的方式來設計[1-15]。Washout Filter 的輸入為由數學運動模型所求得的比例力與角速度。所謂的比例力(Specific Force)是指單位質量所受的非重力力。由向量的觀點來看的話，比例力即是慣性加速度與重力加速度之差，因此比例力可定義為 $\vec{f} = \vec{a} - \vec{g}$ 。此比例力的觀念在航太工程中被廣泛的使用，主要的應用是判斷飛行員在飛行過程中是否會遭遇過大的比例力而飛生不可預期的危險，以及飛行載具的機械結構是否足以承受。

在這裡，我們將完整介紹沖淡演算法的架構以及它在整個系統裡所扮演的角色。目前實現沖淡演算法的方法很多，在此我們是採用廣泛被使用的典型沖淡演算法(University of Toronto Classic Washout Algorithm, UTCWA) [10-12]。圖一為沖淡演算法的架構，我們將以此架構將模擬器上之一參考點之運動感覺轉換為其在上平台對應點之運動。沖淡演算法其實是一種概念，它主要是利用人類對感覺具有門檻值的特性，適當地將某些頻率成分的線性運動或轉動運動加以濾除，並將持續的比例力感覺用緩慢轉動一傾斜角的方式來實現，這個傾斜角所產生的轉動被控制在人類對轉動感覺的門檻值以下，操作員將不會有明顯的感覺，以避免混淆原本的轉動運動的感受。



圖一 沖淡演算法架構

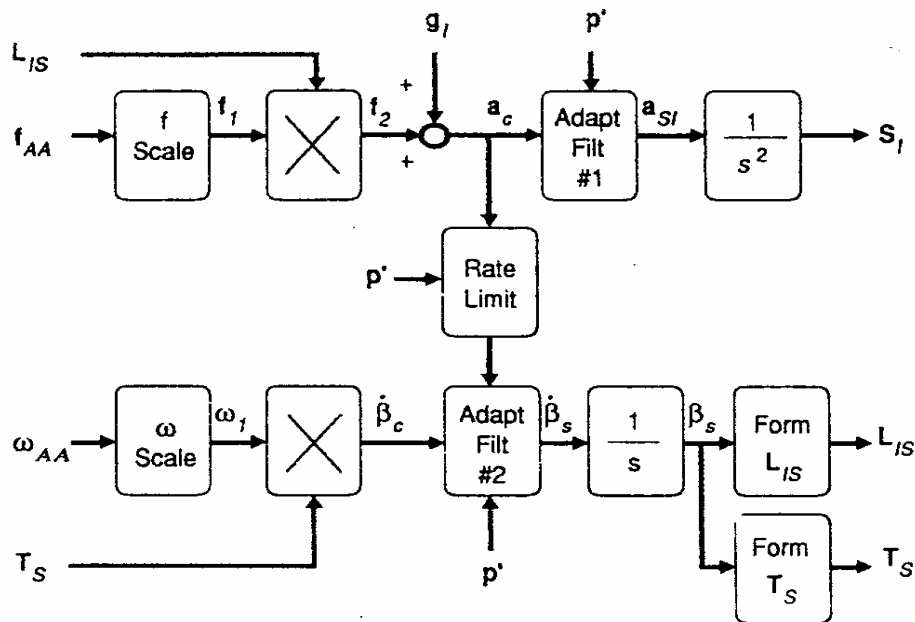
人類對轉動運動的感受主要是由來自半球管，而比例力則是來自內耳石。這裡人體運動感覺模型的輸入為我們前面所介紹的比例力與角速度，並且各自獨立的處理與運作，而得到新的人體真正感覺得到的力與轉動運動。人體對頻率為 0.2 rad/sec~10 rad/sec 的角速度的感覺極為敏銳，當角速度或角加速度較為低頻或為一常數時，感覺較為遲鈍；而對頻率為 0.2 rad/sec~2 rad/sec 的比例力的感覺也極佳，而對持續的加速度感覺則趨於零。沖淡演算法正是利用人類此一特殊的感覺模型所架構出來的。根據上面所述，人體對比例力與角速度之感覺類似一帶通濾波器，但我們在實做上為了將較人體感覺更高頻的部份表現出

來，所以我們以高通濾波器來實現。在平移運動(Translation Motion Channel)的部份，將駕駛員在模擬器的數學模型下，所推導出來的力經過運動學轉換成本身的受力，經過一個比例及極限處理器，或稱為 scale block。這個 scale block 的主要作用就跟人體運動感覺模型中的非線性衰減器一樣，其理由是人類對在門檻值以下之比例力感覺較為遲鈍而將之省略，如此可以避免平台之高頻震顫，同時亦可以調變值的大小，並限定此輸入的大小以免超出平台的動作極限，使其值更適合平台。 $\bar{f}_1$ 和 ω_1 為機身座標的作用力描述，而在 washout filter 中，大部分都會將 $\bar{f}_1$ 和 ω_1 轉為慣性座標上的描述(也就是做座標轉換乘以 L_{1s} 和 T_s)。如是轉換會使在模擬運動時，較容易回歸原點。可是在慣性座標上，運動及轉動動作軸間二者間有 Coupling 的效應。而在早期的實驗研究中[12]，可以看出這些 Coupling 效應是較可忽略的，比起使用機身座標所產生原點的偏移。將 $\bar{f}_1$ 利用轉換矩陣 L_{1s} 轉置為慣性座標系的座標向量後加上重力加速度 $\bar{g}^I$ ，最後將之通過一個高通濾波器。此濾波器主要的功用是將比例力中較低頻的部份加以濾除，只保留高頻中人體所以感受得到的部份，且能使平台在原點附近動作以免超過其工作空間，這也是我們使用沖淡演算法最主要的用意—確保平台在其原點動作而不至超過其工作空間，並能在有限空間內模擬出人體對動作的感受。接下來將濾波器的輸出值 $\bar{a}_H$ 經過兩次積分以得到我們所要的平台三個軸所對應的動作姿態的值，這個值就是模擬操縱者在模擬器中的受力情形所對應的位移運動。而轉動運動(Rotation Motion Channel)的動作跟平移運動是很類似的。這個角度是模擬操縱者在模擬器中所受到的轉動運動所對應的角度，但輸入平台的的角度還必須加上如下所要說明的斜傾座標所產生的傾斜角。

而斜傾座標系(Tilt Coordinate Channel)的作用，主要是用來模擬低頻的受力情形。在模擬器最難實現的模擬狀況之一，就是一段長時間持續加速的過程，因為平台沒有足夠的空間來完成這個動作。但是對人體而言，我們能感受到的也只是一開始加速時身體受到慣性作用，及向後傾與結束加速時身體向前傾的兩種受力情形。而中間這段時間大多沒有感覺的。但是我們一般可以利用視覺上的效果，來使人認為目前是正在動作的，這個動作就是斜傾座標系利用重力加速度在 X - Y 平面上之分量所製造出來的效果。這就是傾斜座標的作用，傾斜一個角度造成視覺上的假像，並利用重力加速度的分量來模擬低頻的比例力。利用重力加速度來產生傾斜角度 $\bar{\beta}_L = [\theta_x \ \theta_y \ \theta_z]^T$ ，其中 $\theta_x = \sin^{-1}(\frac{f_{Ly}}{g})$ 、

$\theta_y = -\sin^{-1}(\frac{f_{Lx}}{g})$ 、 $\theta_z = 0$ ，而 $g \cdot \sin(\theta_x) = f_{Ly}$ 和 $g \cdot \sin(-\theta_y) = f_{Lx}$ 。再將斜傾座標所產生的 $\bar{\beta}_L$

與轉動運動所產生的 $\bar{\beta}_{SH}$ 兩向量合起來前， $\bar{\beta}_L$ 還需經過 Rate Limit Block，這個 Rate Limit 最主要的用意是將 $\bar{\beta}_L$ 限制在人體對轉動速度感受不到的角度內，使人身體無法察覺斜傾座標所產生的轉動動作，避免影響轉動運動所產生的動作，而僅是視覺上的模擬。然而斜傾角的改變速率限制上必須要有限制，以免操作者感受到此一斜傾行為。在斜傾角速率限制上，一般為在 pitch 方向為 13deg/s，在 roll 方向為 2deg/s。



圖二

在文獻中[7-9]也有提出調適式的 Washout filter 如圖二為一調適式之 Washout filter，其概念是根據線上取得的資訊去調整 Filter 中的增益。其主要是將平台的運動和被模擬之運動之間的差異及平台特性的考量來，架構出一性能指標。並經由一最佳化的運算，來找出增益調適的法則，由於指標函數及增益間之關係複雜及即時的需要。一般都是以最陡下降 (steepest decent) 法來產生。如欲追隨之運動已知，則也可利用最佳控制的概念，來直接尋找最佳之線性 filter [10,11]。此一調適演算法主要的功能，即是要減低錯誤動作方式及使平台儘量維持在原點。而在如圖二的架構中，其斜傾處理的計算也有所改變。首先其所利用的信號本身就不一樣。在調適式架中由於是針對慣性座標軸，因此有許多的運動的斜傾反應，也進入角度運動環路中了。因此其在調適的考慮上也變得非常複雜[15]。因此在[15]中，作者也是出以傳統 Washout filter 架構的調適 Filter。因此本計畫將嘗試在這些方面進行探討的研究，就我們所要研究的輔助系統來探討不同的 Washout 演算法在不同狀況下的效能評估。

表一 平移運動之非線性衰減器之參數

平移運動之非線性衰減器之參數			
	X 軸	Y 軸	Z 軸
Threshold Value d_T (m/sec^2)	0.17	0.17	0.25
Slope k_T	0.5	0.5	0.5

表二 轉動運動之非線性衰減器之參數

轉動運動之非線性衰減器之參數			
	Roll	Pitch	Yaw
Threshold Value d_R (deg/sec)	3.0	3.6	3.6
Slope k_R	0.5	0.5	0.5

表三 濾波器之參數

Transfer function of Filters	$T(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$		
	High-Pass Filter 1	High-Pass Filter 2	Low-Pass Filter 1
b_0	0.0378	0.9565	0.0015

b_1	0	-1.9131	0.0030
b_2	-0.0378	0.9565	0.0015
a_1	-1.9187	-1.9112	-1.8890
a_2	0.9244	0.9150	0.8949

在目前交大的 Washout Filter 的參數(針對汽車運動)的非線性衰減器的參數值列於下表一、二、三中。而必需說明的一點是，這些值並非是絕對的，因為每個人對運動的感受程度不同值就會不同，甚至當時的身體狀況、心理狀況與專心程度不同也會影響的這些值，所以表中所列的值僅是參考。我們將以這些參數來當一初始研究的標的，並進而了解在不同狀態下，washout 演算法的調變效果，及若以加強式學習調變參數時，其效能及可變化極限的研究。

而在目前研究中，washout 演算法大部分都是以飛機飛行為主要考慮對象。飛行動作平台主要功能是訓練駕駛員，因此針對 Washout 演算法的設計由於被模擬的飛機是不變的，因此可透過不斷的調整及改變使其效果變好，而且由於飛行模擬必須考慮極大的操作範圍，不過其範圍不會突然變動很大，有時候是以數個不同的 Washout 演算法來設計。在本計劃的研究中，由於是屬於地面上的運動，而且根據不同的需求及不同的狀態。因此如何使 Washout 演算法能在這些系統中仍適用，或必須做恰當的轉換是值得探討與學習的。同時由於地面運動體的運動反應一般較快而多變，這是因為運動體的質量小，因此當施於作用力時，其必然能產生較顯著的反應。因此在 Washout 演算法的設計上也變得較困難。由於運動轉換主要的著眼點是操作者感受的正確性，因此取得操作者的回饋來進行系統修正變得相當重要。因此在本子計劃的研究中，也想將操作者的回饋信息用來修正運動轉換的相關機制。

固定平台的虛擬實境主要是靠影像部分來達成，當運動為直線定速時，如是模擬是相當不錯的，可是當運動路徑是多變的時候，在轉彎加速或上下起伏時，在固定平台上的感受就差很多。若使用動作平台時，由於虛擬系統能設法產生一些相對應的動作反應，使操作者能感受到類似的感受力，此時在輔以影像的虛擬，確實能使虛擬實境達到更逼真的效果。可是當動作平台產生誤動作時，整個系統的感受將是很不好的。因此如何去設計好的動作，便是虛擬實境系統中不可或缺。此一部分的設計一般統稱為 motion cue。然而針對 motion cue 的設計，大致可分為兩類，其一為事先已知運動內容的，例如動態電影院或虛擬歷險等，其所運動的路徑及方式都是事先已經設定好了。因此在實務上大多以經驗法則加上現場測試調整來達成。另一類則是無法事先知道運動的內容，例如相關模擬操控系統，其運動內容是由現場的命令來驅動的。因此無法以如上述的方式來設計 motion cue。在飛機模擬系統中，大部分則是以前面所討論到的 washout filter 來執行。在我們的研究中，以上兩種行為轉換的 motion cue 我們都將加以探討。在前者由於場景的設計及規劃大致其運動範圍是固定的，而許多反應可能是來自於場景而非運動本身。因此我們將以運動動態電影的分析來架構 motion cue 的經驗法則。

Motion Simulation is to rebuild the feeling of passengers in a vehicle from a locally moving simulator. The most popular device utilized in motion simulation is Stewart platform. Stewart platform is a six-degree free device. Because Stewart platform is complicated and hard to design, washout filters are often used to fulfill motion simulation. The idea of washout filters is to ignore the frequency so that a limited workspace of platform can generate infinite motion. Originally, motion simulation is developed to train flight pilot. Nowadays, motion simulation not only is employed in flight simulation, but also can be used in various vehicle simulations. We design a spring device to be added after the washout filter to reduce the chance of reaching the boundary of the workspace.

The position of 6-degree free platform is defined by $(x, y, z, roll, pitch, yaw)$. We divide it into two domains, the shift domain (x, y, z) and the angular domain $(roll, pitch, yaw)$. Now we construct the relationship between each two vectors in each domain respectively. We now focus on the shift domain. Figs. 3~5 are the mutual relationships of each two vectors while other

variables are zero. The outer ellipses are the approximation field of the available field, and the inner ellipses are the safe zone. The fixed vector far away the origin the effective area is smaller. Hence, we can approximate the boundary of shift domain as an ellipsoid. The ellipsoid is described as

$$\frac{x^2}{160^2} + \frac{(y+15)^2}{160^2} + \frac{z^2}{100^2} = 1 \quad (1)$$

The mutual relationships of each vector in the angular domain are shown in Figs. 6~8 while other variables are zero. The boundary of angular domain can also be approximate as an ellipsoid, and it is described as

$$\frac{roll^2}{15.8^2} + \frac{pitch^2}{16^2} + \frac{yaw^2}{15.2^2} = 1 \quad (2)$$

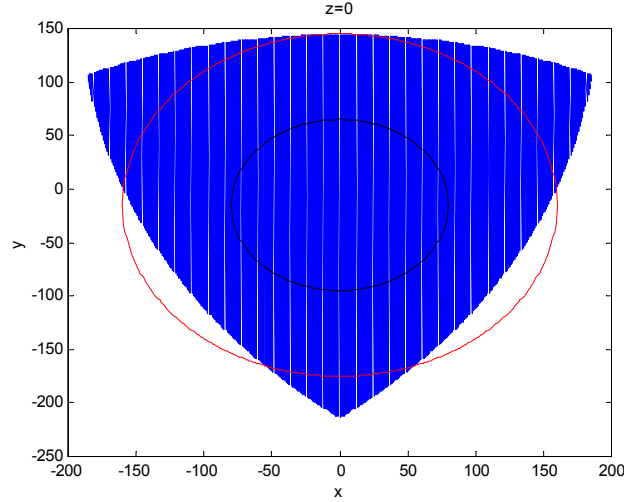


Fig. 3 The mutual relation between x and y with z=0.

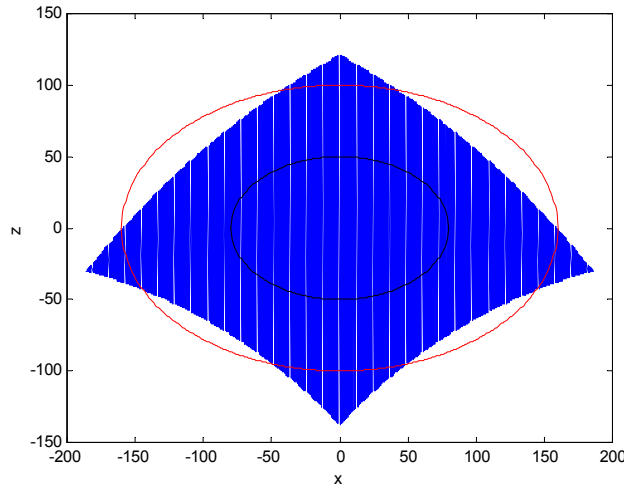


Fig. 4 The mutual relation between x and z with y=0.

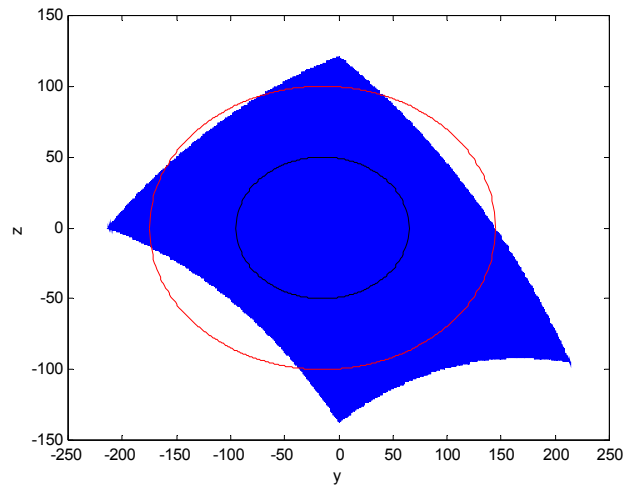


Fig. 5 The mutual relation between y and z with $x=0$.

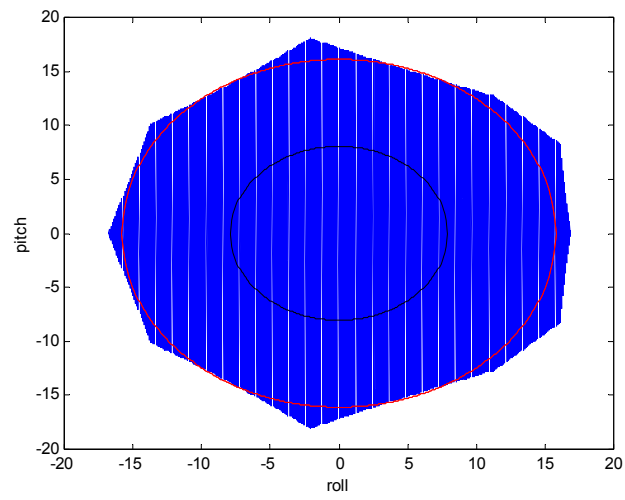


Fig. 6 The mutual relation between roll and pitch with $yaw=0$.

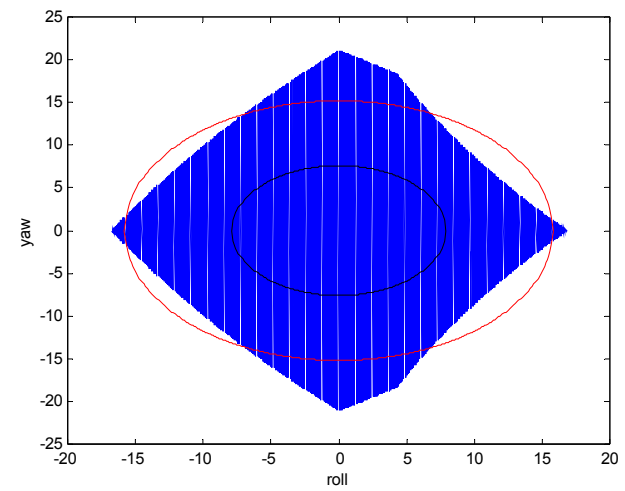


Fig. 7 The mutual relation between roll and yaw with $pitch=0$.

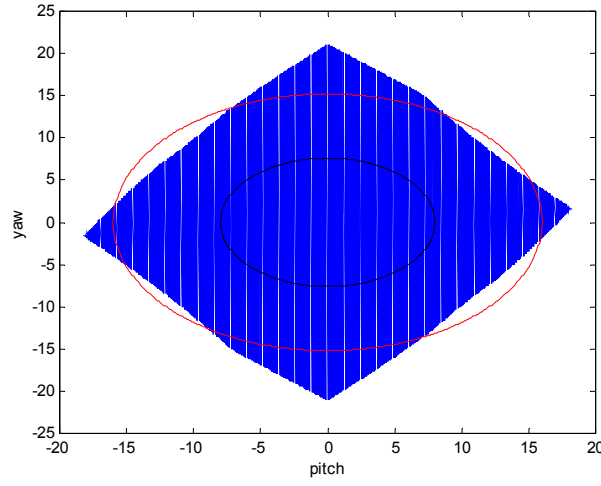


Fig. 8 The mutual relation between pitch and yaw with roll=0.

We have defined the boundary of the working space and the warning zone and dangerous zone are described in Table 4.

Table 4 The equation of the working space surfaces.

	(x, y, z)	(roll, pitch, yaw)
Boundary Surface	$\frac{x^2}{160^2} + \frac{(y+15)^2}{160^2} + \frac{z^2}{100^2} = 1$	$\frac{roll^2}{15.8^2} + \frac{pitch^2}{16^2} + \frac{yaw^2}{15.2^2} = 1$
Warning Surface	$\frac{x^2}{80^2} + \frac{(y+15)^2}{80^2} + \frac{z^2}{50^2} = 1$	$\frac{roll^2}{7.9^2} + \frac{pitch^2}{8^2} + \frac{yaw^2}{7.6^2} = 1$
Dangerous Surface	$\frac{x^2}{120^2} + \frac{(y+15)^2}{120^2} + \frac{z^2}{75^2} = 1$	$\frac{roll^2}{11.85^2} + \frac{pitch^2}{12^2} + \frac{yaw^2}{11.4^2} = 1$

We consider a continuous acceleration motion. In Figs. 9 and 10, the solid line is the original position and the dash line is the position after spring device. We can see the dash line is very similar to the solid line. Thus, the feeling of pilot on the platform is very similar even the same. From Fig. 10, we can find that while the original trajectory is closer to the working space boundary, the new generated trajectory is changed bigger. In the other words, it is harder to move the actuators while the platform is closer to the working space boundary. In Fig. 10, we also can find the three still part of the original trajectory, the new generated trajectory moves slightly toward the original. In Fig. 11, the solid line is the original length of actuators and red dash line is the length of actuators after spring device. We can see the dash line is closer to origin than the solid line obviously.

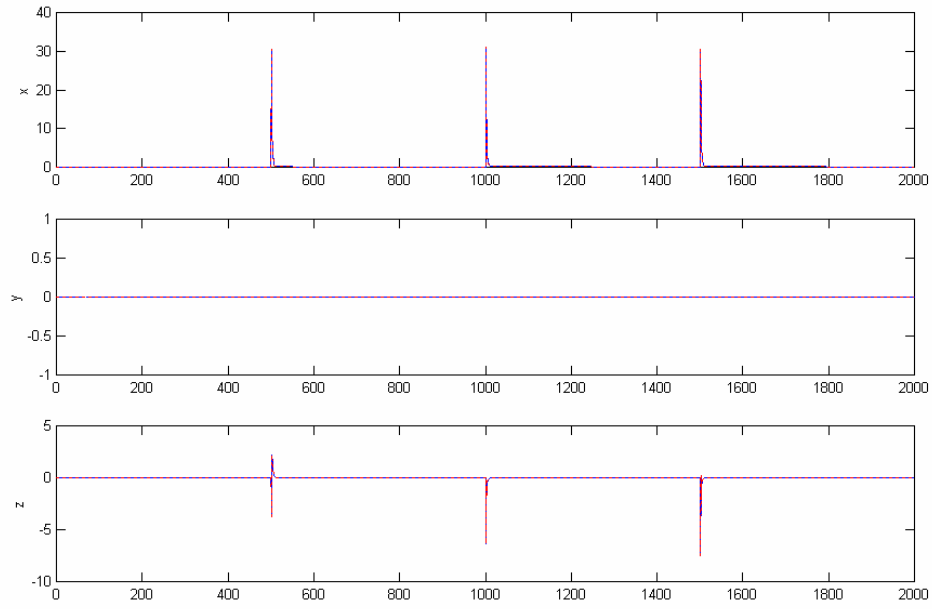


Fig. 9 The displacement outputs after spring device.

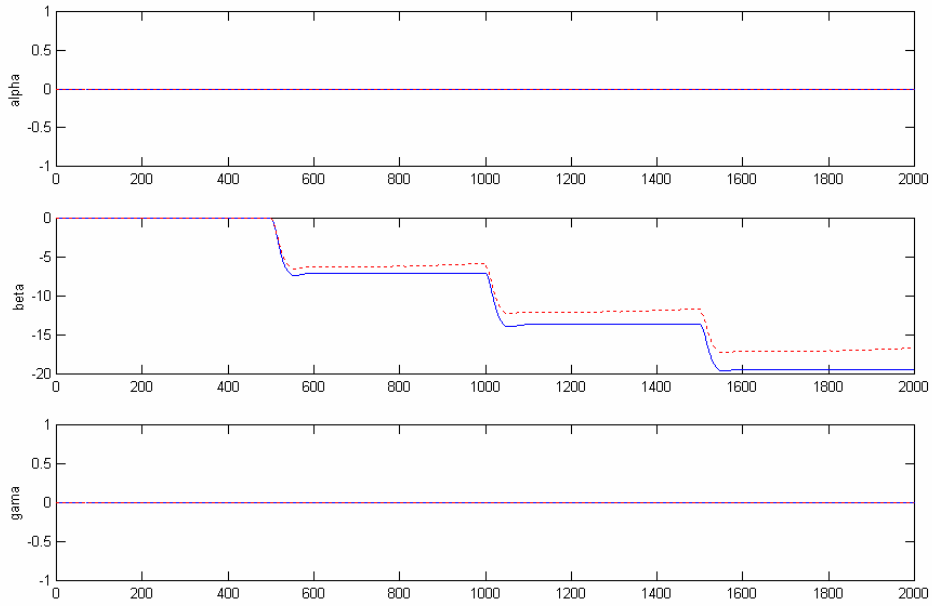


Fig. 10 The angular outputs after spring device.

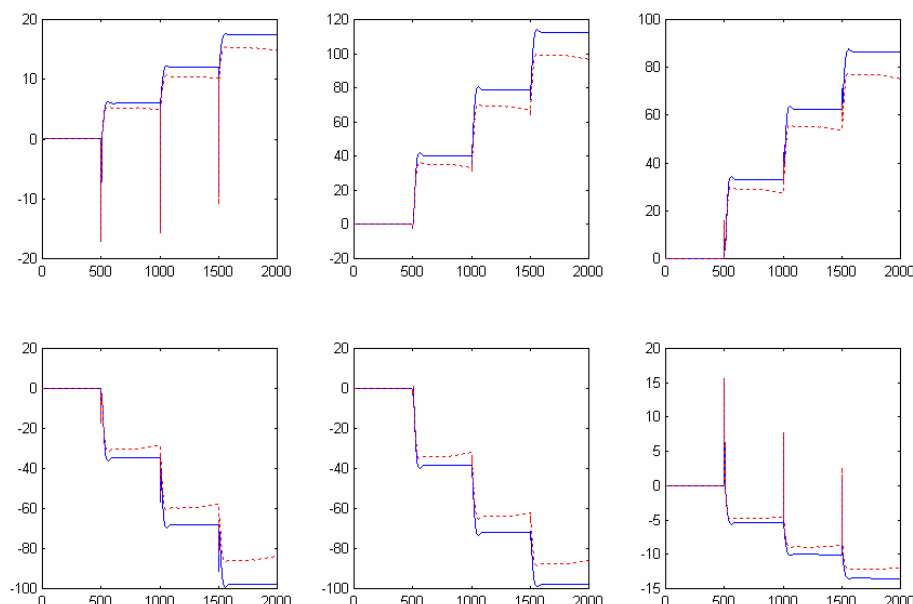


Fig. 11 The length of actuators after spring device.

三、行為轉換演算法

在我們的國科會研究中，我們取得美新科技一組影片及對應的平台操控命令列，目前我們在探討以資料庫的方式來建立的 motion cue 的對應法則。有關 motion cue 方面，我們也分析了一段雲霄飛車影片，企圖來了解一般在 motion cue 方面的設計概念。所得到的心得如下：軸一主要操控上下的運動，而腳長縮短表示感受向上的力(加速度)，例如轉為下坡運動的起始瞬間。而腳長伸長表示感受向下的力(加速度)，可分為兩種狀況，一是運動轉為上波的瞬間，其二是一個下坡路段的盡頭，即轉為平地。軸二主要操控左右的運動，而腳長縮短表示向左的力，例如右彎、右側的碰撞或路面右側高起。反之腳長伸長表示向右的力，例如左彎、左側的碰撞或路面左側高起。軸三主要操控前後的運動，而腳長縮短表示受到向後的力，例如加速。而腳長伸長表示受到向前的力，例如減速或是與前方物體碰撞。軸四主要操控以平行車身為軸做旋轉的動作，而腳長伸長表示向左旋轉，例如路面右側突起或是由左側突起路面轉為平地，也可用來輔助左彎。反之，腳長縮短表示向右旋轉，例如路面左側突起或由右側突起路面轉為平地，也可輔助右彎。軸五主要操控以車身的鉛直線為軸的旋轉運動，而腳長伸長代表逆時針旋轉，例如在左彎時。腳長縮短代表順時針旋轉，例如右彎。軸六主要操控以左右方向為軸的旋轉運動，而腳長縮短代表往下旋轉，例如路面轉為下坡或上坡結束。腳長伸長代表向上旋轉，例如轉為上坡或下坡結束。另外直接以運動的觀點說明腳長的變化；加速為軸三縮短，軸六伸長。減速為軸三伸長，軸六縮短。上坡為軸一伸長，軸六伸長。下坡為軸一縮短，軸六縮短。右彎為軸二縮短，軸五縮短，若是傾斜路面軸四縮短。左彎為軸二伸長，軸五伸長，若是傾斜路面軸四伸長。最後，我們觀察在這段影片中是否有偷偷把腳長拉回原點的動作。我並沒有發現有偷腳長的現象，而是設計者以不超過平台運動範圍內任意設計，因此我們估計並不是每個類似動作發生時乘坐在平台上的人員的感受強度都會一樣。可能發生在影片中這段的加速度比較大但是感受卻比加速度比較小時來的強烈。也就是針對各種模擬主體的運動，我們將建立其相對應的平台運動方式。

There is another way of dealing with the motion of simulators and is referred to as the motion cue. It is often used in the so-called virtual reality emulation theaters. Because this kind of application is an off-line system, engineers can design the platform variation to generate motion cues by their experience. Because of this property, there are two major advantages for motion cue. One is that because motion cue is an off-line mapping, we can plan the whole motion to avoid the

movement of actuators out of the working space. The other one is that special movements can be added to generate more entertaining effects. Because motion cue deeply relies on an experienced engineer to arrange a good motion cue trajectory of Stewart platform positions, we attempt to build fuzzy rules to make the motion cue design more easily. The performance of motion cue deeply depends upon the experience of the designer, and it can usually be designed delicately. The disadvantage of motion cue is that a well-designed motion cue always relies on engineers' experience. Hence, we would like to discuss a method that engineers can plan motion cue easier even that the engineer is with less experience. Then, we focus on the trajectories of actuators. If we can find the relationships between motions and the length of actuators, we can easily design the motion cue without experience. Now we have a series of trajectory of length of actuators. We have found some relations between motions and length of actuators, and the main relations are described in Table 5.

Table 5 The relations between motions and actuators.

Motion \ Actuator	First	Second	Third	Fourth	Fifth	Sixth
Accelerate forward			Shorten			Shorten
Up slope	Extend					Shorten
Turn right		Shorten			Shorten	

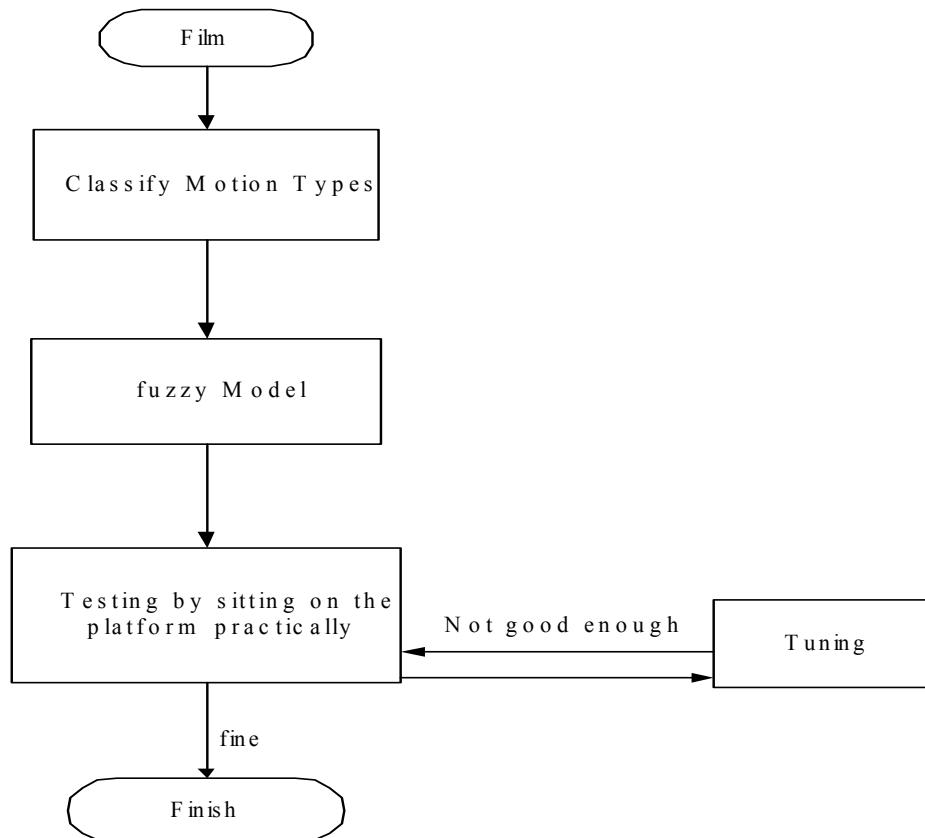


Fig. 12 The motion cue design process with fuzzy model.

Although we have found some relations between motions and length of actuators, it is not easy to build a fuzzy rule set. It is because it is hard to understand the physical meaning that the motion cue provides. Hence, to focus on the trajectories of the length of actuators is not a good way to design motion cue. In order to satisfy the physical meaning, we then focus on the position of the platform. In this study, as shown in Fig. 12, a process is proposed to design motion cue with fuzzy rules of position of the platform. While a film is obtained, we first classify the motion,

and then map different types of motions maps to different fuzzy models. Fuzzy rules are to form trajectories. Then, a pilot is to test whether the generated is realistic. If it is not good enough, the motion cue is tuned until the pilot satisfies the effects. We then have the rule tables as shown in Tables 6~10.

Table 6 The Z variation fuzzy rules of upward slope.

Length of the slope Acceleration	Short	Normal	Long
Slow	L	S	S
Normal	L	M	M
Fast	L	L	L

Table 7 The moving rate of Z fuzzy rules of upward slope.

Length of the slope Acceleration	Short	Normal	Long
Slow	S	S	S
Normal	L	M	S
Fast	L	L	M

Table 8 The Z variation fuzzy rules of downward slope.

Length of the slope Acceleration	Short	Normal	Long
Slow	L	S	S
Normal	L	M	S
Fast	L	L	M

Table 9 The fuzzy rules of making a turn motion.

Radius of Curve	Velocity	Length of Curve	Variation Percentage of Y	Moving Rate of Y
L	L	L	L	S
L	L	M	M	S
L	L	S	S	S
L	M	L	M	M
L	M	M	S	S
L	M	S	S	S
L	S	L	S	S
L	S	M	S	S
L	S	S	S	S
M	L	L	L	L
M	L	M	M	M
M	L	S	M	S
M	M	L	M	M
M	M	M	S	S
M	M	S	S	S

M	S	L	M	M
M	S	M	S	S
M	S	S	S	S
S	L	L	L	L
S	L	M	L	L
S	L	S	L	L
S	M	L	L	L
S	M	M	L	L
S	M	S	L	L
S	S	L	M	M
S	S	M	S	S
S	S	S	S	S

Table 10 The moving rate of Z fuzzy rules of downward slope.

Length of the slope Acceleration	Short	Normal	Long
Slow	S	S	S
Normal	L	M	S
Fast	L	M	S

According to the film offered by Occam Corp., we utilized the fuzzy rules to generate the platform position. Fig 13 is the surge variation of an upward slope motion, in which the solid line is the trajectory offered by Occam Corp. and the dash line is the trajectory generated by the fuzzy rule. Fig 14 is the sway variation of an turn right motion, in which the solid line is the trajectory offered by Occam Corp. and the dash line is the trajectory generated by the fuzzy rule.

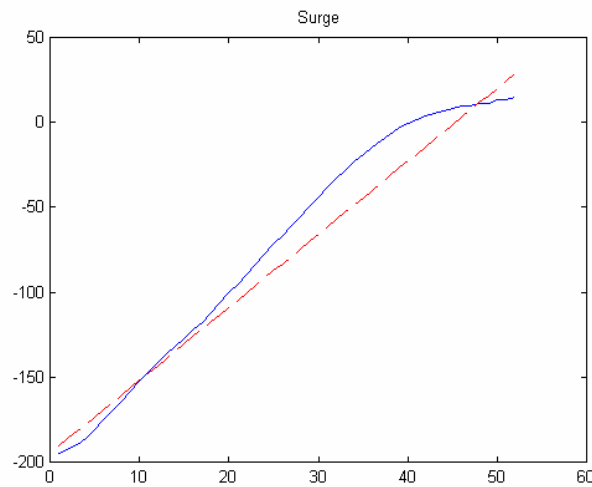


Fig 13 The Surge(Y) variation.

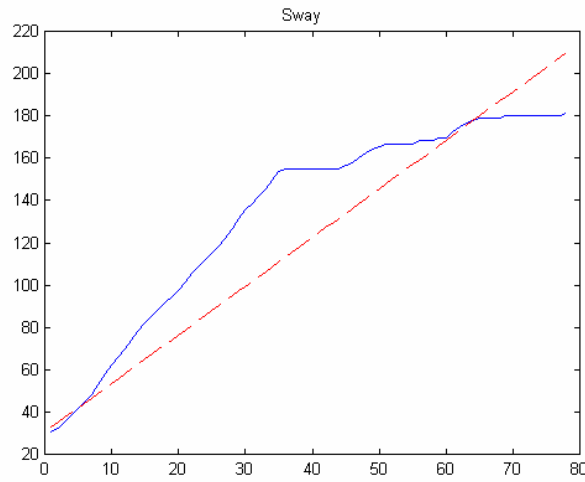


Fig 14 The Sway(Z) variation.

四、函數型類神經網路 (FFANN)

可是這樣的資料庫再使用上可能會很難，因為 generalization 的能力是相當缺乏的。而且此資料庫的型態是以函數為本質而不是點。在我們的研究中我們將研究利用 Function Artificial Neural Network(FFANN) [23-26]來做 Motion cue 資料的建立。而在第一年的研究中，由於其運動體就是操作者，所以其所需要的動作平台的模擬反應較小，而且是可預測的。因此在研究上我們將考慮在我們先前研究中運動轉換資料庫的設計上，運動轉換資料庫主要的概念是針對已知的運動行程(如動態電影院)，直接設計平台動作的方式，由於此部分有些專家的經驗，因此針對如是資料庫，我們可以累積經驗使之成為與被模擬機構及平台機構無關的運動轉換法則，而為了能達到知識粹取的過程，利用類神經方式的建模是值得探討的，除了其能利用 recognition without definition 的特質，其將知識參數化，也可提供未來進一步學習修正的依據。

而在 motion cue 的記錄方面，我們考慮使用函數型類神經網路來記錄 motion cue。在傳統的類神經網路，通常以點對點的方式在時域當中進行系統的建構。而所建構的網路以適應性網路進行網路鍵結值的學習。近來，一種函數型類神經網路架構已發表出來，函數型網路架構使用數學的方式進行函數對函數的輸出輸入對訓練網路之鍵結值。因為函數型網路使用平行處理之數學計算，可以快速估計出約略所需的輸出鍵結值。由於它優異的函數近似能力，使得我們可以將其運用在系統之頻域響應當中。

FFANN 在 1997 年的時候由 Newcomb 及 de Figueiredo 所發表出來的一種類神經網路架構、整個的神經網路架構如圖 15。其突破傳統神經網路以點對點的輸出輸入對的方式訓練神經網路，而改用 function 取代原本以 point 為主要架構。輸入、輸出、鍵節值皆以 function 來表示。而 FFANN 在鍵結值修正學習的機制裡為順向傳遞。由圖可見 $u_1(\cdot) \cdots u_n(\cdot)$ 為 functional input，若 $u(t) = \{u_1(\cdot) \cdots u_n(\cdot)\}$ ，則 n 為它的取樣個數。輸入層與第一層隱藏層所連結的鍵結值為 pattern 的取樣值，進入第一層隱藏層之後，經過 Volterra series 的 model 所產生的值與鍵結函數 $c_1(t) \cdots c_n(t)$ 做乘加，所得出結果 $y_1(t) \cdots y_n(t)$ 。其他的應用，還有參考資料[16]—含未知非線性系統之類神經網路控制，所發展新架構的 FCPBUM，在傳統點對點的類神經網路，為了達到良好的函數近似結果及快速收斂的目的，引用我們所發展的 CPBUM（以契比雪夫多項式為基底的統一模型的類神經網路），在函數的學習上運用 FFANN，在學習架構上使用 CPBUM，而達到更多的優點，包含最佳化的近似，及良好的內差能力之外，根據相似度及靈敏度加入了知識拓展的優點。得以克服目前點對點神經網路在未知線性系統軌跡追蹤學習的缺點。另外在[17]中，作者所提出的 D-FFANN 架構。在文中發表一新式函數型類神經架構，唯一非線性適應性時間序列預知器，以 discrete cosine transform basis function 為基底而產生的 filter bank 來做鍵結值，主要用來 model 一些非線性的動態訊號，例如：人類的與音訊號。

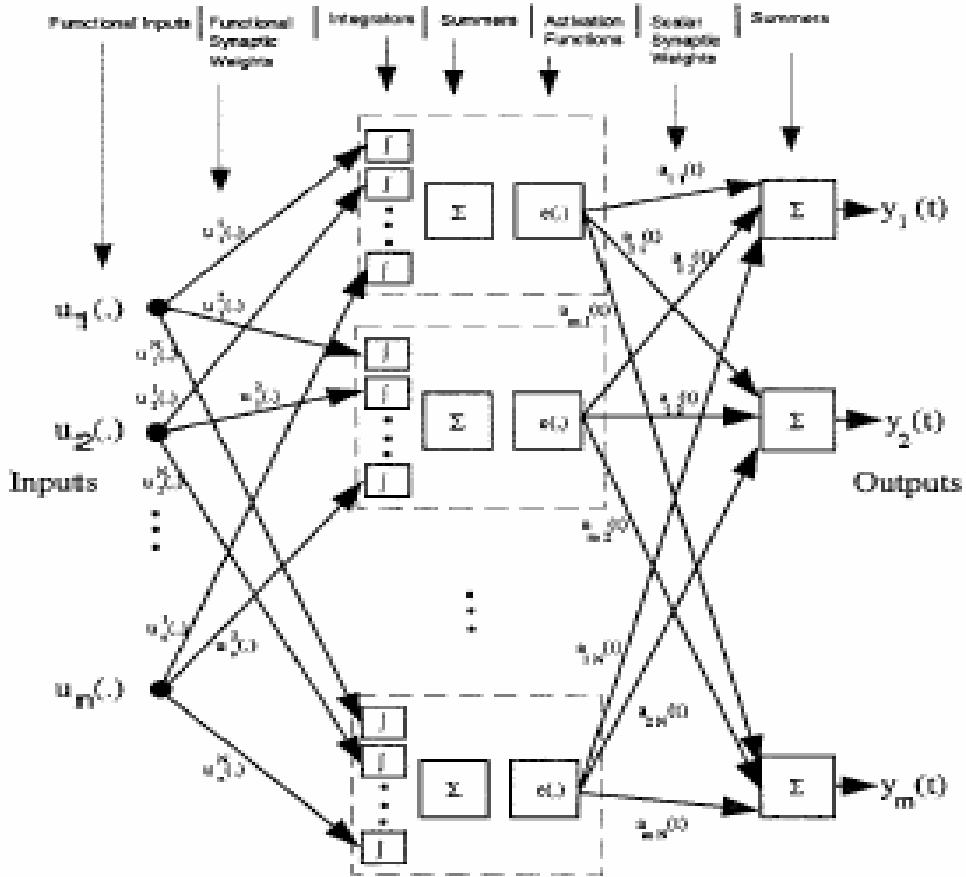


圖 15：MIMO FANN 基本架構

而在我們的未來工作之中，可望結而各種演算法，解決如輸入必須為能量有限的輸入（即為非週期且遞減的輸入曲線）的缺點，例如 $\sin$ 跟 $\cos$ 訊號的學習。藉由前置的離散訊號處理以頻率域替代傳統的時間域學習，並期望在噪音的環境之下由在頻率域作業的優點，而產生抑噪的功能；第二步，並將頻率分解，由 FANN 的近似能力來學習，進而追蹤期望訊號；第三步輸出並轉換為時間域。這個架構可結合訊號處理的優點（高速、抑噪），而達到我們想要的結果。目前 FANN 正處於萌芽時期，其中尚有以上一些缺點必須解決，對於週期性的輸入便無法辨識，還有當期望輸出上下限落差大，FANN 也無法學習，及鍵結值的學習速度也在改進的範圍之中。希望藉由我們提出的架構來改善以上所提之缺點，並提升神經網路的效能。FANN 以函數為輸出及輸入的類神經網路。而為了能進一步粹取出轉換法則，我們也將使用 Function Fuzzy Network(FFN)的方式來設計。FNN 的架構及其學習效果等，都是本研究所要探討的。

FANN is a powerful and fast approach of modeling complex systems. All structure is developed by the mathematical theorem, named the Volterra series. The Volterra approach has been used for modeling nonlinear systems, when prior information is available about the internal characteristics of such systems or when the complexity of the system prevents the postulation of explicit parametric models. The identification problem of a single-input single-output (SISO) nonlinear dynamical system is considered in [26], under the assumption that the input-output relationship can be described by a Volterra series

$$y(t) = V(u)(t) = V_t(u(\cdot))$$

$$= \sum_{k=0}^{\infty} \frac{1}{k!} \int \cdots \int_{t_1 \in I} \cdots \int_{t_k \in I} h_k(t; t_1, \dots, t_k) \cdot u(t_1) \cdots u(t_k) dt_1 \cdots dt_k \quad (3-1)$$

where the input u belongs to real $L^2(I)$. I is an interval of the real line, and, for some nonnegative integer n , the output y is a member of the Sobolev space $H_n^2(I)$ of real-valued functions y on I such that $y^{(i)} = d^i y / dt^i$ is absolutely continuous on I , $i = 0, 1, \dots, n-1$, and

$y^{(n)} \in L^2(I)$. On physical grounds, these conditions state that u and y and sufficiently many, n , of the output derivatives have finite energy by taking all such functions to be square integral over I . It is proven in [26] that when a set of test pairs $\{u_j \in L^2(I), y_j \in H_n^2(I)\} j=1, \dots, N\}$ is available, where $u_j, j=1, \dots, N$, are distinct elements of $L^2(I)$, then the optimal estimate $\hat{V}_i(u)$ of $V_i(u)$ is

$$\hat{V}_i(u) = \sum_{j=1}^N c_j(t) \cdot \exp\left(\frac{1}{r} \langle u_j(\cdot), u(\cdot) \rangle\right) \quad (3-2)$$

where $\langle u(\cdot), v(\cdot) \rangle_{L^2(I)} = \int_{t \in I} u(t)v(t)dt$ and $w_j \in H_n^2(I)$ are obtained by (3-2)

$$\begin{bmatrix} w_1(t) \\ \vdots \\ w_N(t) \end{bmatrix} = G^{-1} \cdot \begin{bmatrix} y_1(t) \\ \vdots \\ y_N(t) \end{bmatrix}, \quad (3-3)$$

$$G = \left[\exp\left(\frac{1}{r} \langle u_i(\cdot), u_j(\cdot) \rangle_{L^2(I)}\right) \right]_{i,j=1, \dots, N} \quad (3-4)$$

The set of test pairs, $\{u_j \in L^2(I), y_j \in H_n^2(I)\} j=1, \dots, N\}$ that describe (12), serve like exemplars of artificial neural networks. This leads to the development of FANN [9][10], which implements as a feed-forward neural network with functional weights $c_j(\cdot)$. In this respect, exemplars of functions while conventional neural networks are trained with exemplars of point values.

The block structure of FANN may be realized as a two hidden layer network [25], where the synaptic weight associated with the connection from $u(k)$ to the j^{th} neuron of the first layer represents the value $u^j(k)$ of the j^{th} exemplar input. The output of each neuron in the first layer being exponential over the sum of weight input products. The second hidden layer consists of a single linear neuron, with synaptic weights $w_j(t)$ and output equal to $y(t)$.

The FANN under consideration, implements an operator that is an optimal L_2 interpolation map of a function $u(\cdot)$ into another function $y(\cdot)$ over a time interval I , according to the formula:

$$y(t) = V(t(\cdot)) = \sum_{j=1}^N w_j(t) \cdot \exp\left(\frac{1}{r} \cdot \sum_{k=1}^{N_s} u^{j^T}(k) u(k) \Delta t\right) \quad (3-5)$$

for input over discrete time of N samples. In general, $u(t)$ and $y(t)$ are vectors of real values at time t ; i.e., $u(\cdot)$ consists of n elements being functions of t , and $y(\cdot)$ consists of m elements being functions of t , resulting in a multiple-input multi-output functional artificial neural network. The patterns $\{u^j(t), t \in I, 1 \leq j \leq N\}$ are exemplar inputs for training the network. In the implementation of a FANN, the exemplar training inputs $u^j(\cdot)$ are represented by N samples in the time interval I , and treated as weight vectors for the first of the two layers of the FANN. During training, $w_j(t)$ is calculated for discrete time instances, or else it is modeled as a continuous vector function. Given an unknown pattern $\{u(t), t \in I\}$, FANN evaluates the similarity of this pattern to each of the exemplars $\{u^j(t), t \in I, 1 \leq j \leq N\}$, and estimates $y(\cdot)$ through weighted average of high selectivity over the corresponding exemplar actions $y^j(\cdot)$, where the degree of selectivity may be adjusted through r . Thus, if $u(\cdot)$ is very similar to $u^k(\cdot)$ and quite dissimilar from all other exemplars $u^i(\cdot)$, then $y(\cdot)$ is approximated by $y^k(\cdot)$.

FANN was further generalized to multiple-input multi-output (MIMO) cases, by replacing $L^2(I)$ with $L_n^2(I)$ and $u(\cdot) = [u_1(\cdot), \dots, u_n(\cdot)]^T$ is a vector of n functions and

$y(\cdot) = [y_1(\cdot), \dots, y_m(\cdot)]^T$ is a vector of m functions. Also, the index j is placed as a superscript in the MIMO case (it was a subscript in the SISO case) in order to avoid confusion with indexing the elements of the vector. Then

$$y(t) = V_t(u(\cdot)) = \sum_{j=1}^N w^j(t) \cdot \exp\left(\frac{1}{r} \cdot \int_{\tau \in I} u^{j^T}(\tau) \cdot u(\tau) d\tau\right), \quad (3-6)$$

where r is a positive real constant, $\{u^j(t) = [u_1^j(t), \dots, u_n^j(t)]^T : t \in I\}$, $1 \leq j \leq N$ are N prototype output patterns, $\{y^j(t) = [y_1^j(t), \dots, y_m^j(t)]^T : t \in I\}$, $1 \leq j \leq N$ with

$$\begin{bmatrix} w^{1^T}(t) \\ \vdots \\ w^{j^T}(t) \\ \vdots \\ w^{N^T}(t) \end{bmatrix} = G^{-1} \bullet \begin{bmatrix} y^{1^T}(t) \\ \vdots \\ y^{j^T}(t) \\ \vdots \\ y^{N^T}(t) \end{bmatrix} \quad (3-7)$$

and $G = [G_{ij}]$ where

$$G_{ij} = \exp\left(\frac{1}{r} \cdot \int_{\tau \in I} u^{i^T}(\tau) \cdot u^j(\tau) d\tau\right). \quad (3-8)$$

The set of test vector I/O function pairs, $\{(u^j, y^j) : j = 1, \dots, N\}$ are the exemplars for training the FANN, i.e., calculating the set of vector functional weight $\{w^j(t) : t \in I, 1 \leq j \leq N\}$.

If an unknown input pattern $\{u(t) : t \in I\}$ is presented, the FANN evaluates the similarity of this pattern to each of the exemplars $\{u^j(t) : t \in I\}$, $1 \leq j \leq N$, and the estimates $y(t)$ through weighted averaging of high selectivity over the corresponding exemplar outputs $y^j(t)$, where the degree of selectivity may be adjusted through γ . From (3-7) and (3-8), notice that $w^j(t)$ is inversely proportional to the exponential of the inner product of $u^j(t)$ with each of the prototype input patterns, and it is proportional to $y^j(t)$. Notice (3-6), $w^j(t)$ is multiplied by the exponential of the inner product of $u^j(t)$ with the input pattern $u(t)$. In this configuration, the exponential terms define normalized weights that emphasize similarity in the averaging. Low selectivity results in smooth averaging over the exemplar outputs, while high selectivity results in a crisper approximation where the output that corresponds to the exemplar input that best matches $u(\cdot)$ is emphasized in the averaging. Thus, if $u(\cdot)$ is very similar to $u^k(t)$ and quite dissimilar from all other exemplars $u^l(t)$, then $y(t)$ is approximated by $y^k(t)$. If the similarity of $u(\cdot)$ to more than one exemplar input is high, then $y(t)$ is approximated as an average over their corresponding exemplar outputs weighted by a measure of the corresponding similarities.

The following is listed the algorithm of the FANN,

Training Phase

- Step 1.** Set the numbers of input layer neurons, second layer neurons, and output layer neurons. The weights between the first layer and second layer are defined in (3-8). From (3-8), the parameter γ is needed.
- Step 2.** Forward the input $x[n] = u[n]$ into the first layer.
- Step 3.** By (3-8), the matrix of second layer neurons, G_{ij} , is calculated.
- Step 4.** By (3-6), compute the output sequence of output layer neurons, Y . From (3-7), the weights between second layer and output layer are calculated. These are the output weights we want.

There is something to be noticed. In the training phase, the algorithm of FANN does not have the learning procedure. That may infect the accuracy of the output, $y(t)$.

Testing Phase

Step 1. Forward the input testing exemplars into the first layer.

Step 2. The weights between the first layer and second layer as step 1 in training phase is used to compute G_{ij} using (3-8).

Step 3. The weights between the first hidden layer and the output layer as step 4 in training phase is used to compute testing output sequence, $y[n]$.

The structure of the FANN in the frequency domain is now proposed. A main difficulty for neural networks working in the frequency domain is that they need complex-valued weights for the networks. Because of complex-valued weights, there are four algorithms are discussed. In order to perform learning for this situation, the structure of FANN must be modified. The modification is to provide mechanism for transferring training data in the time domain into the frequency domain and to improve the performance, as we require. In the first, a FANN-based neural network is constructed. To consider signals in the frequency domain, the fast Fourier transform and inverse fast Fourier transform are necessities. In this stage, some details about sample points and aliases are important to reconstruct signals in the time domain.

It is well known that when the analysis or design of a system is conducted in the frequency domain, the result should be more robust to noisy and/or outliers compared to that in the time domain. For analysis in the frequency domain, sometimes, a discrete low pass filter is required. In our study, the Kaiser window [32] is employed to design such a low pass filter. The Kaiser window is defined as

$$w[n] = \begin{cases} \frac{I_0\left[\beta\left(1 - \left[(n-\alpha)/\alpha\right]^2\right)^{1/2}\right]}{I_0(\beta)}, & 0 \leq n \leq M, \\ 0, & \text{otherwise,} \end{cases}$$

where $\alpha = M/2$, and $I_0(\cdot)$ represents the zero-order modified Bessel function of the first kind. The Kaiser has two parameters: the length $(M+1)$ and a shape parameter β . Let ω_s be the stop-band cutoff frequency and ω_p be the pass-band cutoff frequency, the width of the transition region is defined as $\Delta\omega = \omega_s - \omega_p$. Define $A = -20\log_{10} \delta$. The value of β is given by

$$\beta = \begin{cases} 0.1102(A-8.7), & A > 50, \\ 0.5842(A-21)^{0.4} + 0.07886(A-21), & 21 \leq A \leq 50, \\ 0.0, & A < 21, \end{cases}$$

Then M must satisfy $M = \frac{A-8}{2.285\Delta\omega}$.

The structure of our functional network networks is shown in Fig. 16. In the FANN, the approximated function is obtained through (3-2~3-4). The functions give the FANN a roughly range of global minima. But the network cannot find the optimal output near the desire output. For BP or FP (forward propagation) types of learning algorithms, their learning is usually slow and has several local minima. In [28], the approach of mixing the BP learning algorithm and the original algorithm used for FANN was proposed and the learning of that approach is fast and powerful in finding global minima.

There are four algorithms used in our implementation of the FANN in the frequency domain. First, the original algorithm of the FANN is still used. That is the FANN in the frequency domain without learning. It is the traditional FANN.

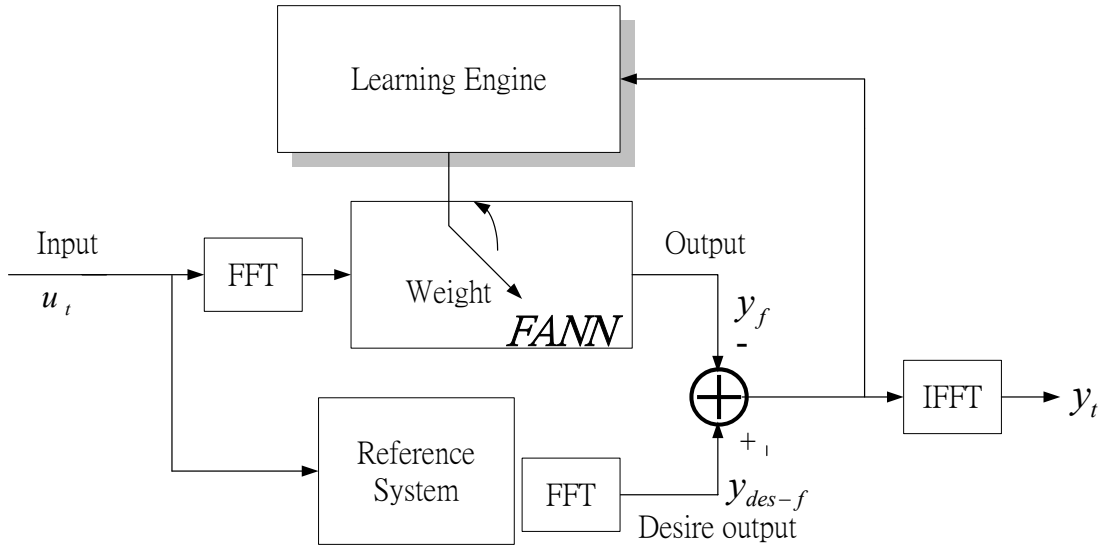


Fig. 16 The modified structure of FANN

Algorithm 1.

- Step 1. Transform all signals into the frequency domain. All signals become complex data.
- Step 2. Set the numbers of neurons in the input layer, in the second layer neurons, and in the output layer neurons, respectively.
- Step 3. Set weights between input layer and hidden layer as exemplars.
- Step 4. Compute the values G_{ij} of neurons in the second layer by using (3-4).
- Step 5. Compute the output values Y_i of neurons in the output layer by using (3-5).
- Step 6. Compute the weights between the second layer and the output layer by using (3-3).
- Step 7. Input the testing signals and compute the output signals by the weights.
- Step 8. Transform the output signals into the time domain.

Because complex-valued parameters are used for representing signals in the frequency domain, CNET (Complex-valued weight Neural Network) is proposed to model systems in [28-31]. The error signals are defined as $\varepsilon_i[i] = y_{des-f,i}[n] - y_{f,i}[n]$, $i, n = 1, 2, \dots, N$ [30]. Then, the cost function is usually defined as the sum of squared errors and is

$$E[n] = \sum_{i=1}^N \varepsilon_i[n] \cdot \bar{\varepsilon}_i[n], \quad (3-9)$$

where $\bar{\varepsilon}_i[n]$ is conjugate of $\varepsilon_i[n]$. The BP algorithm minimizes the cost function (3-9) by recursively altering the weights based on the gradient search technique. The partial derivative of E with respect to the real and imaginary part of the weights must be derived separately. Let

$$w_{ij}[n] = wr_{ij}[n] + iwi[n]. \quad (3-10)$$

Consider the adaptation rule of the output layer as

$$wr_{ij}[n+1] = wr_{ij}[n] - \frac{1}{2} \mu \frac{\partial E[n]}{\partial wr_{ij}[n]} \quad (3-11)$$

$$wi_{ij}[n+1] = wi_{ij}[n] - \frac{1}{2} \mu \frac{\partial E[n]}{\partial wi_{ij}[n]}. \quad (3-12)$$

Combining (3-11) and (3-12), we have

$$wr_{ij}[n+1] + iwi_{ij}[n+1] = wr_{ij}[n] + iwi[n] - \frac{1}{2} \mu \left(\frac{\partial E[n]}{\partial wr_{ij}[n]} + i \frac{\partial E[n]}{\partial wi[n]} \right) \quad (3-13)$$

$$\text{or} \quad -w_{ij}[n+1] = w_{ij}[n] - \frac{1}{2} \mu \left(\frac{\partial E[n]}{\partial wr_{ij}[n]} + i \frac{\partial E[n]}{\partial wi[n]} \right) \quad (3-14)$$

$$\text{and} \quad \frac{\partial E}{\partial wr} = \frac{\partial E}{\partial z} \cdot \frac{\partial z}{\partial y_f} \cdot \frac{\partial y_f}{\partial wr} + \frac{\partial E}{\partial \bar{z}} \cdot \frac{\partial \bar{z}}{\partial \bar{y}_f} \cdot \frac{\partial \bar{y}_f}{\partial wr}, \quad (3-15)$$

where $z = f(y_f)$, $y_f = \sum w u_f + \theta$. They are commonly used in neural networks. f is the sigmoidal function and is defined as $f(y) = \frac{1}{1 + e^{-y}}$. Then,

$$\frac{\partial E}{\partial w_r} = -(\bar{y}_{des-f} - \bar{y}_f) f'(y_f) \mu_f - (y_{des-f} - y_f) f'(\bar{y}_f) \bar{\mu}_f. \quad (3-16)$$

Similarly,

$$\frac{\partial E}{\partial w_i} = -i(\bar{y}_{des-f} - \bar{y}_f) f'(y_f) \mu_f + i(y_{des-f} - y_f) f'(\bar{y}_f) \bar{\mu}_f. \quad (3-17)$$

Combining (3-16) and (3-17), we have

$$\frac{\partial E}{\partial w_r} + i \frac{\partial E}{\partial w_i} = -2(y_{des-f} - y_f) f'(\bar{y}_f) \bar{\mu}_f. \quad (3-18)$$

Substituting (3-18) into (3-13)

$$w_{ij}[n+1] = w_{ij}[n] + \mu(y_{des-f} - y_f) f'(\bar{y}_f) \bar{\mu}_f. \quad (3-19)$$

Equation (3-19) is the basic update rule.

Algorithm 2

Steps 1~6 are the same as that in Algorithm 1.

Step 7. Input the training patterns and update weights by (3-19).

Step 8. Input the testing patterns and compute the output signals by the weights.

Step 9. Transform the output signals into time domain.

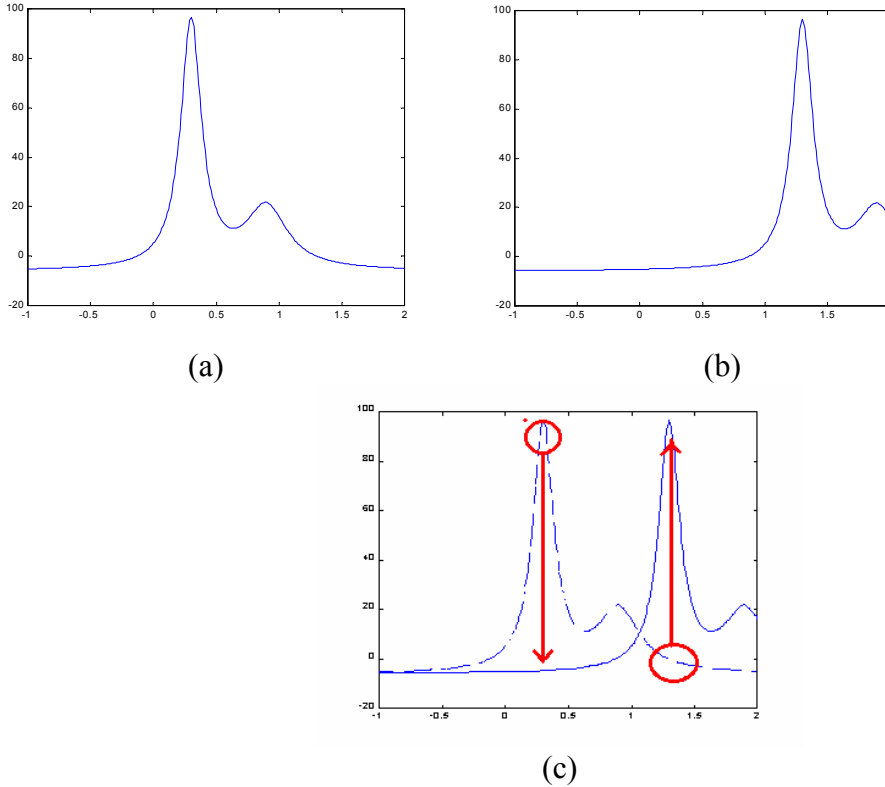


Fig. 17 (a) The output before learning (b) The desire output (c) The algorithm must to depress the signal we do not want and grows the wanted signal up.

There exist problems in the above algorithm. The learning procedure updates the real and imaginary parts of the weights. Algorithm 2 uses only one learning rate to tune the weights and the range of the real part value is different from that of the imaginary part value. That means if the range of the real part value is $[-50, 50]$, the range of imaginary part may be $[-0.5, 0.5]$. By the algorithm in traditional FANN, the input signals and output signals of FANN do not have to normalization to $[-1, 1]$. As a consequence, the updated weights are not optimal. The other problem is the disturbance. The trace after learning is like the desired trace plus a sinusoid with a fixed frequency. The phenomenon is shown in Fig. 17. The output contains a small impulse in the amplitude of the Bode plot. It is obvious that the obtained amplitude is not what we want. After

the signal is reconstructed in the time domain, such an unwanted impulse in the frequency domain becomes a sinusoid in the time domain.

The complex numbers can be expressed as the following form:

$$(a) \text{ (real part)} + (\text{imaginary part}) j \quad (3-20)$$

$$(b) (\text{amplitude}) \times e^{j(\text{phase})} \quad (3-21)$$

In the next discussion, we separate the complex numbers into two neural networks. In (3-20), the FANN (a) is to model the real part of the complex-valued weights and the FANN (b) is to model the imaginary part. Similarly, in (3-21), the FANN (a) is the neural network to define the amplitude of the complex-valued weights and the FANN (b) is to define the phase of the complex-valued weights. Note that the learning rate in FANN (a) is different from that in FANN (b).

Algorithm 3 with ((real part) + (imaginary part) j) method

Step 1~6 are the same as those in Algorithm 1.

Step 7. In the FANN (a), input the training patterns (the real part of all signals) and update weights by (3-19) in the FANN (a).

Step 8. Input the testing patterns (the real part of all signals) and compute the output signals (the real part of all signals) by the weights.

Step 9. Do Step 1~8 for the FANN (b). The output signals (the imaginary part of all signals) are calculated.

Step 10. Combine the real part and the imaginary part into complex numbers.

Step 11. Reconstruct the signal in the time domain.

Algorithm 4 with (amplitude) $\times e^{j(\text{phase})}$ method

Step 1~6 are same as the Algorithm 1.

Step 7. In the FANN (a), input the training patterns (the amplitude of all signals) and update weights by (3-19) in the FANN (a).

Step 8. Input the testing patterns (the amplitude of all signals) and compute the output signals (the amplitude of all signals) by the weights.

Step 9. Do Step 1~8 for the FANN (b). The output signals (the phase of all signals) are calculated.

Step 10. Combine the amplitude and the phase into complex numbers.

Step 11. Reconstruct the signals in the time domain.

In Algorithm 4, updating amplitude and phase is very sensitive in time domain. Algorithm 3 overcomes the disadvantages mentioned for Algorithm 2 and Algorithm 1. The network learns the system quickly and robust. The tracking performance is better than that of the FANN in the time domain. To reserve the idea of the FANN, all the structures and algorithms in our implementation follow the formula in equation (3-8). In other words, all weights between the hidden layer and input layer are not updated.

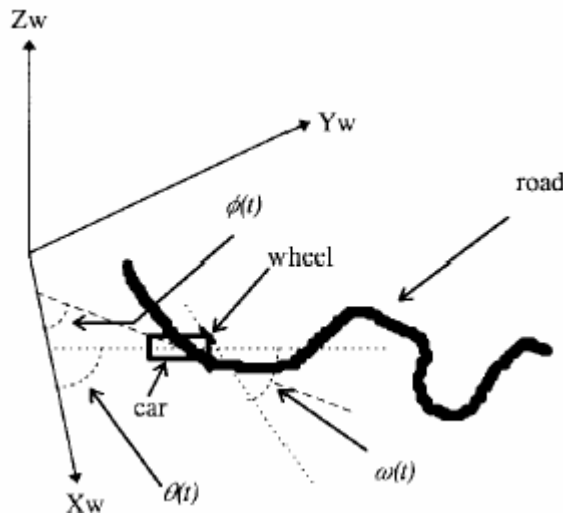


Fig. 18 Example and the parameters we define

The case of a car and a driver-system, which was also used in [8], is used as an example in our implementation. In this example, the output is expressed by its moving direction. The goal of

the driver-system is to keep the car on the road. As shown in Fig. 18, this car is assumed to move on a path resides on the $X_w - Y_w$ plane of the world coordinate system (WCS). The desired path is represented as a function of traveling time t . Let a point on the path then can be expressed as a parametric pair, like $(x_w(t), y_w(t))$ and $\phi(t)$ be the angle of the slope of this path at a point through t and is obtained as

$$\phi(t) = \tan^{-1}(dy_w(t)/dx_w(t)) \quad (3-22)$$

Then, the sequence $\{\phi(t)\}$ represents the shape of the path. Define $\theta(t)$ as the angle between the central axis across the car and the X_w axis of the WCS. Then $\theta(t)$ denotes the direction of the car. The driver's goal, which is to keep the car remaining on the road, then can be expressed through the relation $\theta(t) = \phi(t)$. The action of the driver-system at time t is the amount of angle $w(t)$ by which the system must turn the wheels with respect to the central axis across the car to keep the relation $\theta(t) = \phi(t)$ satisfied for each t .

The road is modeled as a composition of segments, where each segment may be a path as a straight line or a concave path forming a turn. Assuming a minimum resolution time Δt by which the driver-system scans the road ahead at a speed U_s , the road segments are sampled at intervals $\Delta S = U_s \cdot \Delta t$. Each road segment then is modeled as a discrete-time sequence of angles of slopes $\{\phi(i): t = i \times \Delta t, i \in Z^+ \cup \{0\}\}$. The driver-system has a perspective view of the road ahead, instead of its actual floor plan. Thus, it views a transformed image of the road ahead. Consequently it perceives a discrete-time sequence of angles of slopes $\{\sigma(i): t = i \times \Delta t, i \in Z^+ \cup \{0\}\}$ on the transformed image. This sequence is used by the driver-system in order to plan its actions; i.e., to generate a sequence $\{\omega_p(i): t = i \times \Delta t, i \in Z^+ \cup \{0\}\}$ before the car enters the viewed road segment. When the car enters the road segment, $t = i \times \Delta t$ will be $\tan(\theta(i))$, where $\theta(i) = \theta(i-1) + \omega(i-1)$, or equivalently, $\theta(i) = \theta(0) + \sum_{k=0}^{i-1} \omega(k)$, and $\omega(i)$ is determined by $\omega(i) = \omega_p(i) + \Delta\omega_p(i)$. Here ω_p is the input u_p and $\Delta\omega_p$ is the correction through feedback.

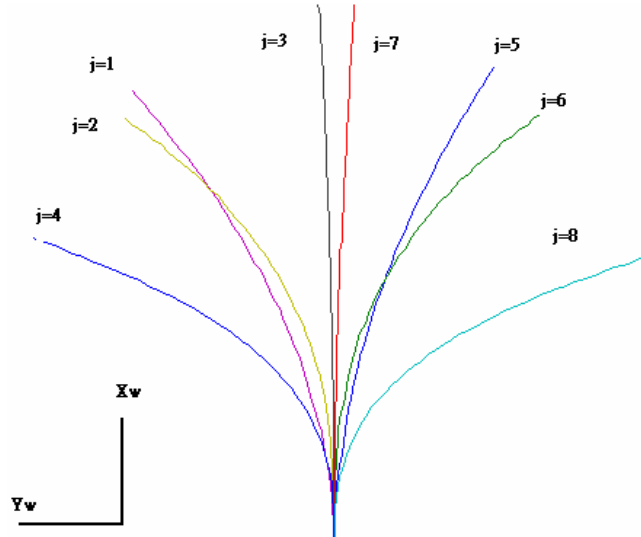


Fig. 19. Floor-plan for eight case of road segments.

Fig. 19 shows the floor-plan for $N=8$ cases of prototype road segments that are considered for training the FANN. In this example, each of those paths were generated by a third degree polynomial,

$$y_w = a_3 \cdot x_w^3 + a_2 \cdot x_w^2 + a_1 \cdot x_w + a_0 \quad (3-23)$$

where the variables x_w and y_w refer to the coordinates of each point along the path. This is the least degree of polynomial that satisfies continuity in position and slope at both ends of each segment and allows their smooth concatenation in modeling general shapes of paths. Since it is desired that $\{\theta(t)\} = \{\phi(t)\}$, we need to have the sequence $\{\phi(t)\}$ for each of the prototype

patterns to train the FANN. Since the prototype path segments are generated by a third degree polynomial, the angle of the tangent at each point along the path is given as

$$\phi(t) = \tan^{-1} \left(\frac{dy_w}{dx_w} \right) = \tan^{-1} (3 \cdot a_3 \cdot x_w^2(t) + 2 \cdot a_2 \cdot x_w(t) + a_1). \quad (3-24)$$

The origin of each path is at the point $(x_w(0), y_w(0)) = (0, 0)$ and then $a_0 = 0$. The slope of the tangent at the origin of each path is zero coinciding with the direction of the car as it enters the path segment, and also $a_1 = 0$. This assumption is natural in this example, because the driver-system may consider a single road segment head of it each time when the car is located at the origin of that segment. Besides, at that time, the moving direction of the car should coincide with the slope at the origin of the segment for a smooth ride. Then, it is also natural to perceive that segment as if it were described with reference to a world coordinate system whose origin coincides with the driver's position and the X_w axis coincides with the moving direction of the car. Then, a straight segment becomes trivial since $a_3 = a_2 = a_1 = a_0 = 0$, and thus $\phi(t) = 0$ at each point along the segment. For this reason, we did not include a straight segment among the eight prototypes in this example, although we could have.

The X_w axis is calculated as

$$x_w(i) = x_w(i-1) + |\vec{U}_s| \cdot \Delta t \cdot \cos(\phi(i)), \quad (3-25)$$

$$\phi(i) \approx \tan^{-1} (3 \cdot a_3 \cdot x_w^2(i-1) + 2 \cdot a_2 \cdot x_w(i-1) + a_1), \quad (3-26)$$

and thus,

$$x_w(i) = x_w(i-1) + |\vec{U}_s| \cdot \Delta t \cdot \cos(\tan^{-1} (3 \cdot a_3 \cdot x_w^2(i-1) + 2 \cdot a_2 \cdot x_w(i-1) + a_1)), \quad (3-27)$$

where $\omega(i)$ is calculated as

$$\omega(i) = \Delta\phi(i) = \phi(i+1) - \phi(i) \quad (3-28)$$

Next, the restoration map of the perceived image is calculated. Let $[C_x, C_y, C_z]$ be the world coordinates of the driver's camera, $N = [N_x, N_y, N_z]$ be the viewing direction of the camera, $\vec{V}$ be a view-up vector that specifies the orientation of the camera, and d be the distance of lens from its screen. Choosing $\vec{L} = \vec{V} \times \vec{N}$ results in a right-hand viewing coordinate system $(\vec{L}, \vec{V}, \vec{N})$ centered at the lens of the camera. Then, a viewing transformation is used in order to map world coordinates $[x_w, y_w, z_w]$ to viewing coordinates $[x_v, y_v, z_v]$, and subsequently a perspective transformation maps the viewing coordinates to screen coordinates $[x_s, y_s, d]$. The angle of slope $\sigma(t)$ that is perceived at each point on the transformed image on the screen of the camera is a mapping of $\phi(t)$, and is calculated as

$$\sigma(t) = \tan^{-1} (dy_s(t)/dx_s(t)). \quad (3-29)$$

Differentiation by parts,

$$dy_s/dx_s = dy_s/dx_w \cdot dx_w/dx_s \quad (3-30)$$

followed by long calculations, (3-29) becomes

$$\sigma(t) = \tan^{-1} \left(\frac{\begin{aligned} &2a_3 \cdot (V_Y \cdot N_X - V_X \cdot N_Y) \cdot x_w^3(t) \\ &+ \left(3a_3 \cdot \left(\begin{aligned} &C_X \cdot (V_X \cdot N_Y - V_Y \cdot N_X) \\ &+ C_Z \cdot (V_Z \cdot N_Y - V_Y \cdot N_Z) \end{aligned} \right) + a_2 \cdot (V_Y \cdot N_X - V_X \cdot N_Y) \right) \cdot x_w^2(t) \\ &+ 2a_2 \cdot (C_X \cdot (V_X \cdot N_Y - V_Y \cdot N_X) + C_Z \cdot (V_Z \cdot N_Y - V_Y \cdot N_Z)) \cdot x_w(t) \\ &+ a_1 \cdot C_X \cdot (V_X \cdot N_Y - V_Y \cdot N_X) + a_1 \cdot C_Z \cdot (V_Z \cdot N_Y - V_Y \cdot N_Z) \\ &+ C_Y \cdot (V_Y \cdot N_X - V_X \cdot N_Y) + C_Z \cdot (V_Z \cdot N_X - V_X \cdot N_Z) \end{aligned} \right)}{\begin{aligned} &2a_3 \cdot (L_Y \cdot N_X - L_X \cdot N_Y) \cdot x_w^3(t) \\ &+ \left(3a_3 \cdot \left(\begin{aligned} &C_X \cdot (L_X \cdot N_Y - L_Y \cdot N_X) \\ &+ C_Z \cdot (L_Z \cdot N_Y - L_Y \cdot N_Z) \end{aligned} \right) + a_2 \cdot (L_Y \cdot N_X - L_X \cdot N_Y) \right) \cdot x_w^2(t) \\ &+ 2a_2 \cdot (C_X \cdot (L_X \cdot N_Y - L_Y \cdot N_X) + C_Z \cdot (L_Z \cdot N_Y - L_Y \cdot N_Z)) \cdot x_w(t) \\ &+ a_1 \cdot C_X \cdot (L_X \cdot N_Y - L_Y \cdot N_X) + a_1 \cdot C_Z \cdot (L_Z \cdot N_Y - L_Y \cdot N_Z) \\ &+ C_Y \cdot (L_Y \cdot N_X - L_X \cdot N_Y) + C_Z \cdot (L_Z \cdot N_X - L_X \cdot N_Z) \end{aligned} \right)} \right), \quad (3-31), \end{aligned}$$

then

$$\phi(k) = \sum_{j=1}^N c^j(k) \cdot \exp \left(\frac{1}{r} \cdot \sum_{l=0}^{N_s-1} \sigma^j(l) \cdot \sigma(l) \cdot \Delta t \right), \quad (3-32)$$

where $\sigma(l)$ is the input and $\phi(k)$ is the output. In our simulation, the driver's camera is located at 1.2m above the ground and $C_Z = 1.2m$. The car is positioned at the origin of each segment. Then $C_X = 0$ and $C_Y = 0$. The viewing direction of the camera coincides with the slope at the origin of each segment. Then $N_X = \left| \frac{dy_w(0)}{dx_w(0)} \right| \cdot \cos(\phi(0))$ and $N_Y = \left| \frac{dy_w(0)}{dx_w(0)} \right| \cdot \sin(\phi(0))$. An

elevation angle of 79° is assumed and then $N_Z = -\left| \frac{dy_w(0)}{dx_w(0)} \right| \cdot \frac{1}{\tan(79^\circ)}$. The direction of the

camera is $\vec{V}' = [0, 0, 1]$. Then, $\vec{V}$ can be calculated as $\vec{V} = \vec{V}' - (\vec{V}' \cdot \vec{N}) \cdot \vec{N}$. Also, $d=2.5$ is assumed. This example is used to train the original FANN and the modified FANN in our simulations.

There are eight patterns and 50 points every pattern. By the theorem of FANN, number of neurons in the input layer is 50, and number of neurons in the hidden layer is 8, and the output neurons is 50 in Algorithm 0. In Algorithm 1, 2, and 3, there are 64 points every pattern. Thus number of neurons in the input and output layer is 64, and number of neurons in the hidden layer is also 8.

Five FANNs are used in this experiment, the original FANN, which is referred to as Algorithm 0, Algorithm 1, 2, 3, and Algorithm 4. Algorithm 1, 2, 3, and Algorithm 4 work in the frequency domain. The learned results of Algorithm 0, 1, 2, 3, and 4 are shown in Fig. 20. In all algorithms, the epoch is 1,000 to train the neural networks. Every vector contains 50 points. To avoid alias, all vectors are attached with 14 zero elements to become 64-points vectors. The stop-band cutoff frequency ω_s is 0.6π , and the pass-band cutoff frequency ω_p is 0.4π . A shape parameter β is 3.3943.

Next, we try to add noise signals and outliers to test the robust capability. The frequency of noise is 1.6π and the amplitude of noise is the value generated randomly from 0.2 to -0.2 . The amplitude of outliers is 0.5 rad. The noise signals and outliers are generated randomly. We take the advantage of ten training times to calculate the error. Table 11 lists the error of testing phase computed in the time domain, and Table 12 lists the error of testing phase computed in the frequency domain. We define the error functions, respectively, in the time domain and in the frequency domain as

$$E_t = \frac{\| \text{desire_output} - \text{experiment_output} \|^2}{\text{desire_output}} \quad (3-33)$$

$$E_f = \frac{real(desire_output - experiment_output)^2 + image(desire_output - experiment_output)^2}{desire_output}$$

From the above simulation results, Algorithm 0 (FANN in the time domain) is not the best. Algorithm 1 is similar to Algorithm 0. Algorithm 1 is more precise than Algorithm 0. Considering the I/O pairs in different domain, the I/O pairs in the time domain are time-variant and changed every time index. The I/O pairs in the frequency domain have the frequency index like ones in time domain. The modeled system has similar frequency responses for different input vectors. Thus the frequency range of the system is located at the same area.

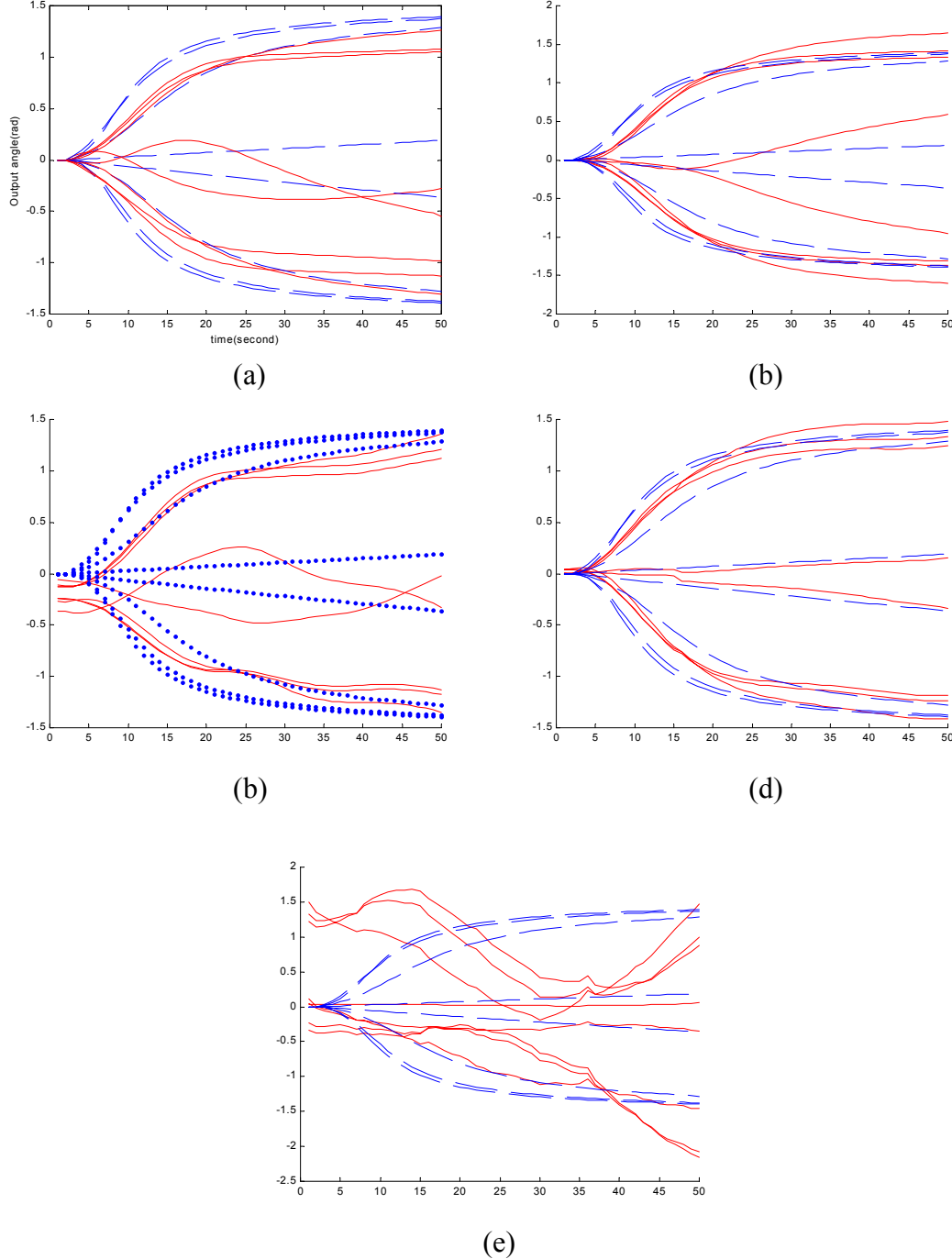


Fig. 20 The vectors of output angle (a) Algorithm 0 (b) Algorithm 1 (c) Algorithm 2 (d) Algorithm 3 (e) Algorithm 4

Traditional neural networks have some disadvantages like slow learning speed and difficult in tracking spectrums. The FANN can speed up the learning phase using Volterra series. But the performance is not good enough for spectrum analysis. In this research, we attempted to find a better neural network by combining the advantages of FANN and of frequency analysis techniques.

Table 11 Error computed in the time domain

ERROR (%)	Algorithm 0	Algorithm 1	Algorithm 1 with LPF	Algorithm 2	Algorithm 3	Algorithm 4
Testing	1.14	0.95	0.69	0.79	0.55	7.62
+Noise	1.230	1.088	0.748	0.908	0.612	8.05
+Outlier	1.108	0.950	0.666	0.789	0.55	7.61

Table 12 Error value computed in the frequency domain

ERROR (%)	Algorithm 0	Algorithm 1	Algorithm 1 with LPF	Algorithm 2	Algorithm 3	Algorithm 4
Testing	18.72	15.73	22.54	24.61	11.31	91.84
+Noise	20.366	17.924	24.124	26.596	12.112	94.234
+Outlier	18.752	15.620	22.27	24.605	11.33	92.062

In this research, we introduced a new algorithm working in modified mixed-mode functional artificial neural networks in the frequency domain. The main structure of our neural network is based on the ideas in FANN. The structure is modified since the original algorithm only works for real numbers. The complex-valued neural network is separated into two parts, FANN (a) and FANN (b). FANN (a) calculates the real part of the complex-valued neural network and FANN (b) calculates the imaginary part. After the functional approximation, the learning engine updates the weights of FANN (a) and FANN (b). Our approach has managed to keep a correct proportion between the real part and the imaginary part. The new algorithm can model functions more precisely than the others do. Because of frequency domain spectrum is required, some Fourier transformation elements are necessary in our structure. The problems of sampling and aliasing are important and must be dealt with in the algorithm. Otherwise, those problems introduce large distortion in the reconstructed signals. This structure can also be employed to analyze the frequency response of control plants such that functional tracking control may become possible. Moreover, due to high computational power existing in the hardware of DSP, this approach is more powerful to control the system.

五、訓練者修正的加強式學習

在本子計畫的研究中，我們也將從事有關訓練者修正的加強式學習，主要的兩個方向是在建模方面，另一則是 motion cue 的部分，主要的觀念乃是在虛擬實境上，主要是要人的感受。因此當操作者對於系統有認為不好的地方，我們希望能將這一類的感受用於修正系統，對建模部分而言，主要是對反應的不真實性，加以修正，若是能建立這樣的系統，則針對建模部分就可以只要建一個簡單大致的模式，而後透過加強式學習來使得模式反應更逼真，如是未來對不同的系統開發可以不需要每一次都要找一個所要開發系統的建模專家，而只要有一對該系統反應狀況較熟悉的人就可以了。同樣的，在 motion cue 設計的部分，也是如此，只要一感受靈敏的人，針對系統反應的評估，即可設計出較精準而感受逼真的動態模擬系統。

在 motion cue 的部分，我們也考慮到修正的問題。畢竟虛擬實境最重要的還是人的感受。而一般 washout 演算法或 motion cue 設計是去經由設計者調整部分參數來改善。而在調適式的 washout 演算法中則是以指標函數的設計來調整 Filer 的增益值，而我們考慮的是以類似在加強式學習建模的方式，來看是否能利用加強式學習[18-20]的方式來達到調適的目的。而利用加強式學習一個重要的步驟是將系統參數化，然後利用參數改變所產生效果的評估來作為加強式學習參數改變的根據。這一部份我們的研究也是分兩方面來探討。針對 motion cue 資料庫的部分，若是以 FNN 架構資料庫的方式能有不錯的結果的話，系統參數化部分就沒有問題了，此部分的研究不僅可以對我們計畫所要探討的問題加以解決也可應用於動態電影院效果上。而針對 washout 演算法部分，我們當然可以直接套用原調適式的演算法的架構，也就是將原來指標函數的部分，改用加強式學習中的評估網路來達到所需要的效果。若效果的改善若有限時，我們目前也考慮將整個 washout 演算法，利用無模式建模的方式來加以建模。此部分將會和 motion cue 資料庫的部分極為類似。所以在研

究上其研究的價值是相當值得投入的。因為如果建模部分如果成功的話，由於我們將使用 TSK 模糊法則方式來建模。屆時，兩種系統的合併及效能比較的因素的分析等研究將可進行。同時，由於其是無需模式方式的建模，其在多變數的 coupling 效應也較能處理。

在模擬建模方面我們也將著手加強是學習的方式來做操作者回饋系統修正的研究。其主要的觀念在於我們的模式為 TSK 模糊網路，因此其已經參數化了。所以將我們對在運動轉換所提到的語言化加強式模式修正的概念應用於模糊模式的建模式。此系統的開發可以使得未來在改變模擬樑時，不需大費周章來精準建模，而只需建立一大致模式然後經由修正的方式來達到較精準的模式。同時在前面所提到的外差防制的部分在使用模糊網路建模時更是需要考慮的一部份。

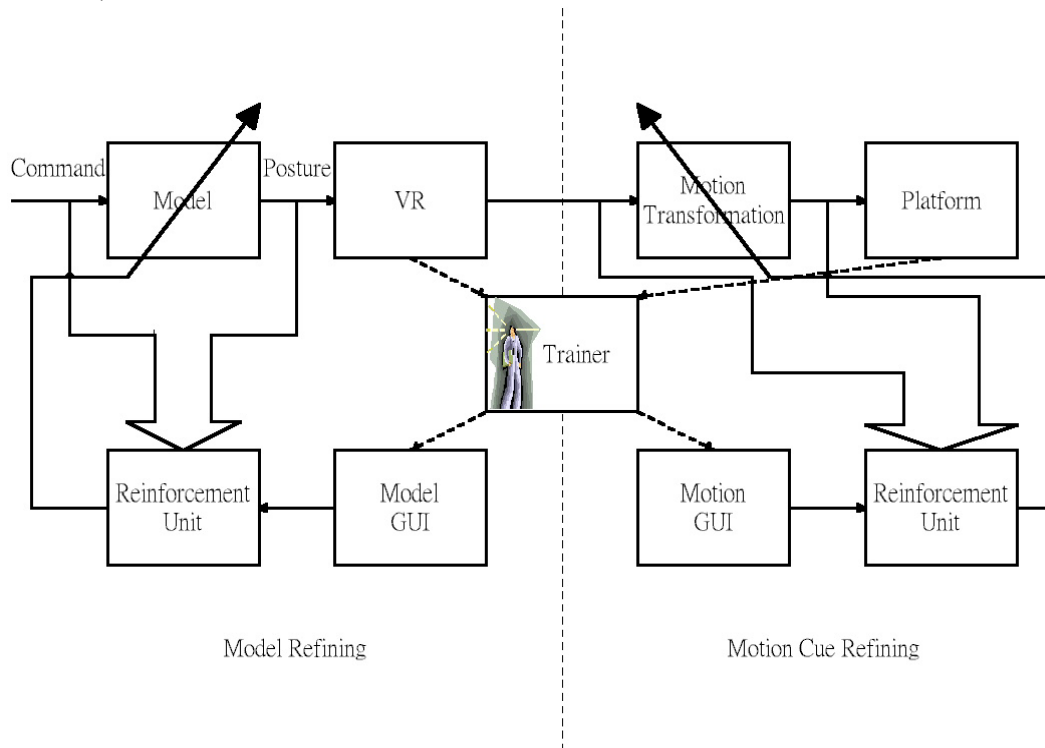


圖 21 加強式學習系統架構

在加強式的研究中，初步的架構如圖 21 上，訓練者在運動平台上，針對平台的反應來修正而修正分為部分，其一為 Model 部分，而另一為 motion cue 部分。在做法上，我們要研究訓練者的評估在加強式學習中的效能。而系統對訓練者其評估的不確定性，也是研究所需要析的。這是因為對同樣的狀況，在不同的時候，由於人類感受評估的不精確性，其可能會有不同的評估值。而同時，針對訓練者評估息的定義，也不必然和事先設定的概念一致。因此如何的調適評估信息的歸屬函數，及其在加強式學習中的學習法則，都需要深入的探討與研究。而目前我們加強式學習的架構是以[21]中的系統為基礎，並考慮用 SONFIN[22]的方式來使得每一網路的法則數不需要那麼多，可是由於 SONFIN 的學習在使用 BP 下較慢，而使用 LS 方法其所欲之輸出不知道，這些在研究中我們都將加以探討。最後，由於如上二種加強式學習。在訓練者的感受上似乎有不可分的現象。目前的概念是先分別探討與學習，以便個別系統能正確做加強式學習，若在分析中能正確分割兩系統效應。則可據以做加強式信號回饋的依據，若不能分割，則可考慮將二系統直接合併。由於加強式學習本來就是可用於較複雜，而不容易數學化的系統而當系統參數化後，其利用 trial-and-error 的特質，也是可以架構出加強式學習系統的。

由於運動轉換主要的著眼點是操作者感受的正確性，因此取得操作者的回饋來進行系統修正變得相當重要，因此在本子計劃的研究中也想將操作者的回饋信息用來修正運動轉換的相關機制。而這樣的系統面臨幾個問題，第一個問題為回饋信息認定的任意性及不精準性，因此如何有效而強健的轉換操作者回饋的信息是需要設計與探討的。第二個則為將運動轉換機制的參數化，以使信息類的加強式學習有操作的標的。而我們在先前的研究中也針對在類神經網路及模糊系統都有探討且有一些成果。因此我們將運動轉換機制(也就是

Washout 演算法及運動轉換資料庫)利用類神經網路或模糊網路加以參數化或部分參數化，進而利用加強式學習來做修正。

再者，當以類神經網路或模糊網路來建立 Washout 演算法及運動轉換資料庫時，我們也必須克服這些網路在使用時，其外插效果的不可預測性。一般系統其外插的可接受，仍是由於其系統是由物理定律或經驗法則所建立的這和類神經網路或模糊網路是不同的，因此我們的研究中也將對此一問題提出解決之道；也就在將轉換機制參數化的同時也建立操作範圍的確立方式，而後將物理定律或經驗法則加於操作範圍外。而在我們的先前研究中，我們也提出混合式建模的概念，此方法則是將知識納入原網路的學習架構中，其效果相當顯著，而在這裡我們所要設計的則完全不同，由於如是物理定理或經驗法則將有可能減低原訓練後網路的精準度，因此如何有效而精準的使用這些知識則是我們要研究的課題。

同時，Washout 演算法及運動轉資料庫為兩種不同的運動轉換處理方式，因此在未來的研究中，我們也將探討這兩者特質及其是否可合併的探討研究，具體的說其一的研究方式是將資料庫中的轉換做去 Washout 的計算，如此資料庫或所謂的轉換法則就可和 Washout 合併動作，另一種作法則是將資料庫法則化及通用化，並將之納入 Washout 演算法中。不過這些方法都有待設計與探討。

在行為的轉換方面，目前文獻上所提到的是利用 washout filter 的方式來處理的。如是的方式是可以將在無限空間中的運動設法在有限空間中呈現，可是在動態模擬器中許多的運動效果並非只是簡單的物理運動定律所能描述的。較多而必要的是若干的特效運動感受，而這些感受則是要利用所謂 motion cue 來設計。而傳統的動態模擬器，由於其設計上也不是利用真實的運動物体的模擬建構來產生虛擬之運動感受，而是去設計不同的 motion cue 來搭配。因此本計劃的研究中即去了解 motion cue 及其效果。而研究主要是提出以訓練的方式來達到對 motion cue 設計的探討。目前主要的構想是利用使用者的回饋信息來修正 motion cue 的內容。而修正的標的則可分為參數修正及段落重分。而由於回饋信息是由使用者所給的，目前探討的學習主要是針對信息的認定，目前簡單作法是利用 reinforcement learning 修正。而信息的內容則是使用者對動態模擬器的動作行為加以評分，而由於必須在 try-and-error 的情形下學習，模擬器動作是由若干 motion cue 所組成而後重覆的操作，因此使用者的評分行為和 motion cue 的修正能產生有義意的比對。而目前我們只考慮到參數修正部分，也就是針對每一段 motion cue，我們以 TSK Fuzzy 模式表達，因此學習的行為就化約為 membership function 及後件部參數的修正，在這一方面傳統的倒傳學習法被用來修正參數(由於沒有確實想要的輸出值，最小平方差(LS)修正法無法使用)。而在內部加強式信號的產生部分，目前我們是以傳統的 Temporal Difference (TD)預估的方式來進行。由於最近的研究大都使用遺傳演算法，我們也會考慮使用，在未來的研究中我們將更進一步的探討分析其結果，同時也將納入類似 washout filter 的概念以及段落分割之學習方面。

六、加強式學習研究與探討

Most of the supervised and unsupervised learning algorithms for neural networks require usable training data sets for setting the link weights and link connectivity of the neurons for various applications. For some real-world applications, usable data for supervised learning are usually difficult, expensive, and sometimes impossible, to obtain. For this reason, there has been a growing interest in the study of reinforcement learning algorithms. In this report, we report our investigation on the reinforcement learning based on the structure of TSK fuzzy model.

For reinforcement learning problem, training data are very rough and coarse. They are just “evaluative” information as compared with the “instructive” feedback in the supervised learning. Training a network with this kind of evaluative feedback is called *reinforcement learning*. In addition to the roughness and non-instructive nature of the reinforcement signal, a more challenging problem to the reinforcement learning is that a reinforcement signal may only be available at a time long after a sequence of action has occurred. To solve the long time-delay problem, prediction capabilities are necessary in a reinforcement learning system. Reinforcement learning with prediction capabilities is much more useful than the supervised learning schemes in dynamic control problem and artificial intelligence, since usually the success or failure signal

might only be known after a long sequence of control action. From the biological and cognitive points of view, reinforcement learning is much closer to the modern animal learning theory [65] than the supervised learning. This is also true to the learning of many high-level intelligent skills such as how to drive a car.

In this report, we report our study of employing TSK fuzzy models as the networks in reinforcement learning systems. In our implementation, a TSK type of Reinforcement Neural-Network-Based Fuzzy Logic Control System (RNN-FLC) adopted from [66] is employed. In the structure, to achieve the goal of solving reinforcement learning problems in fuzzy systems, there are two NN-FLCs. One NN-FLC, the action network, is used for the fuzzy logic controller, which generates a proper action or decision according to the current input vector. The other NN-FLC, the evaluation network, is used as the fuzzy predictor, which performs the multi-step prediction. The fuzzy predictor provides the action network with more informative and beforehand internal reinforcement signal for learning. These two NN-FLCs share the first two layers. In other words, they use the same distributed representation of input patterns. This representation is the overlapping type and is dynamically adjustable through the learning process.

Associated with the proposed TSK type RNN-FLCS is the reinforcement parameter learning algorithm which uses the temporal difference technique on the evaluation network to decide the output error for multi-step prediction. With the knowledge of output error, a supervised parameter learning algorithm can be applied to train the evaluation network to obtain the proper membership functions and fuzzy logic rules. For the action network, the reinforcement parameter learning algorithm allows its output nodes to perform stochastic exploration. With the internal reinforcement signals from the fuzzy predictor, the output nodes of the action network can perform effective stochastic searches with a high probability of choosing a good action as well as discovering its output error.

It should be noted that in [66], the consequence parts are fuzzy singletons. From [66] or our simulation, it is possible to have fair learning if the initial values of those singletons are assigned reasonable. However, if those singletons are assigned arbitrary, the learning is very bad. It is expected because the values of those singletons are spread in the range of $[-10, 10]$ and then if the learning constant is small, some rule of which the consequence values are large cannot have enough learning effects to achieve their values. On the other hand, when the learning constant is large, some rule of which the consequence values are small cannot have subtle learning effects. As a result, in our study, it seems impossible to have successful learning. In our study, we attempt to use TSK fuzzy models, of which the consequence parts are linear functions. Thus, when a linear function is used, it may become difficult to reasonably assign initial values. In order to solve the initialization problem in our TSK models, we employ a genetic algorithm to search for more suitable initial parameters. Because GA can search for better solutions for any problem through selection, crossover, and mutation, the GA based prior knowledge is suited for our learning process.

In this study, we perform the two kinds of learning process of previous chapter in different initial status of controlled plant. Besides performing, we will demonstrate the importance of prior knowledge. The initial weights in layer four of five-layered neural fuzzy inference network are developed through human-knowledge. Because the bell-shaped is used to describe the membership function, therefore, we can represent the membership function through center or width. We can collocate to the cart position and pole angle which is finding the rough and usable initial weight. Because the TSK type five-layered neural fuzzy inference network is more complex, we can not easily obtain rough and suitable initial weight. Then we will use GA based prior knowledge to solve complex condition. In [66], NN-FLC has the function of structure learning. We are finding some issue of structure learning in [66].

The proposed RNN-FLCS with multistep fuzzy predictor has been written in MATLAB language for the cart-pole balancing problem or the so-call inverted pendulum balancing problem. This problem is often used as an example of inherently unstable and dynamic systems to demonstrate both modern and classic control techniques [67], as well as the learning control techniques of neural networks using supervised learning methods [68] or reinforcement learning methods [69] [70].

As shown in Figure 22, the cart-pole balancing problem is the problem of learning how to balance an upright pole. The bottom of the pole is hinged to a cart that travels along a finite-length track to its right or left. Both the cart and the pole can move only in the vertical plane; that is, each has only one degree of freedom. There are four state variables in this system.

θ : the angle of the pole from an upright position (in degree)

$\dot{\theta}$: the angular velocity of the pole (in degree/second)

x : the horizontal position of the cart's center (in meters)

$\dot{x}$: the velocity of the cart (in m/second)

The only action is f , which is the amount of force (N) applied to the cart to move it toward its left or right. The system fails and receives a penalty signal of -1 when the pole falls past a certain angle ($\pm 12^\circ$ degrees is used here) or the cart runs into the bounds of its track (the distance is $2.4m$ from the center to both bounds of the track). The goal of this control problem is to train the RNN-FLCS such that it can determine a sequence of forces with proper magnitudes to apply to the cart to balance the pole for as long as possible without failure.

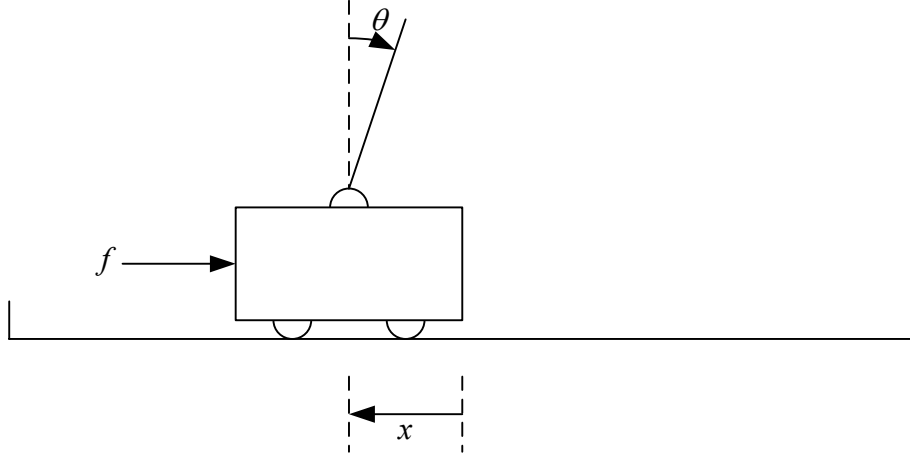


Figure 22 The Cart-Pole balancing problem

The model and corresponding parameters of the cart-pole balancing system for our computer simulation are adopted from [66]. This model and its parameters are also used by Barto, Sutton, and Anderson [70]. The equations of motion that we used are shown below

$$\begin{aligned}\theta(t+1) &= \theta(t) + \Delta \dot{\theta}(t) \\ \dot{\theta}(t+1) &= \dot{\theta}(t) + \Delta \frac{mg \sin \theta(t) - \cos \theta(t) \left\{ f(t) - m_p l \dot{\theta}^2(t) \sin \theta(t) - \mu_c \operatorname{sgn} \left[\dot{x}(t) \right] \right\} - \frac{\mu_p m \dot{\theta}(t)}{m_p l}}{\frac{4ml}{3} - m_p l \cos^2 \theta(t)} \\ x(t+1) &= x(t) + \Delta \dot{x}(t) \\ \dot{x}(t+1) &= \dot{x}(t) + \Delta \frac{f(t) + m_p l \left[\dot{\theta}^2(t) \sin \theta(t) - \ddot{\theta}(t) \cos \theta(t) \right] - \mu_c \operatorname{sgn} \left(\dot{x}(t) \right)}{m}\end{aligned}$$

where

$g = -9.8 \text{ m/s}^2$: acceleration due to the gravity

$m = 1.1 \text{ kg}$: combined mass of the pole and the cart

$m_p = 0.1 \text{ kg}$: mass of the pole

$l = 0.5 \text{ m}$: half-pole length

$\mu_c = 0.0005$: coefficient of friction of the cart on the track

$\mu_p = 0.000002$: coefficient of friction of the pole on the cart

$\Delta = 0.02$: sampling interval

The constraints on the variables are $-12^\circ \leq \theta \leq 12^\circ$, $-2.4m \leq x \leq 2.4m$, and $-10N \leq f \leq 10N$. In

designing the controller, the equations of motion of the cart-pole balancing system are assumed to be unknown to the controller. A more challenging part of this problem is that the only available feedback is a failure signal that notifies the controller when a failure occurs; that is, either $|\theta| > 12^\circ$ or $|x| > 2.4m$. This is a typical reinforcement learning problem, and the feedback failure signal serves as the reinforcement signal. Since a reinforcement signal may only be available after long sequence of time steps in this failure avoidance task, this cart-pole balancing problem belong to the reinforcement learning problems. Thus, a multi-step fuzzy predictor is required for the RNN-FLCS. Moreover, since the goal is to avoid failure for as long as possible, there is no exact success in finite time. Also, we hope that the RNN-FLCS can balance the pole for as long as possible for infinite trials, not just for one particular trial, where a “trial” is defined as the time steps from an initial state to a failure. The reinforcement signal is defined as

$$r(t) = \begin{cases} -1 & \text{if } |\theta(t)| > 12^\circ \text{ or } |x(t)| > 2.4 m \\ 0 & \text{otherwise} \end{cases} \quad (4.1)$$

and the goal is to maximize the sum $\sum_{k=0}^{\infty} \gamma^k (t+k)$, where γ is the discount rate.

In our computer simulation, the learning system was tested for ten runs by trying to use the same learning parameter values. A *run* means consisted of at most 100 trials, unless the duration of each run exceeded 500'000 time steps. Each run consisted of a sequence of trials; each trial began with the same initial condition $x(0) = \dot{x}(0) = \theta(0) = \dot{\theta}(0) = 0$ or with a randomized initial condition, and ended with a failure signal indicating that either $|x(t)| > 2.4 m$ or $|\theta(t)| > 12^\circ$. The randomized initial condition means that, the initial configuration was independently and randomly chosen such that $-1.8 m \leq x(0) \leq 1.8 m$ and $-9^\circ \leq \theta(0) \leq 9^\circ$. The input fuzzy partitions were set as $|T(x)| = 3$, $|T(\dot{x})| = 3$, $|T(\theta)| = 7$, $|T(\dot{\theta})| = 3$ for all runs. For each run, the input (output) membership functions were initialized so that they covered the whole input (output) space evenly, and the output fuzzy partition was initialized as $|T(f)| = 189$. In [15], the membership functions were chosen to be bell-shaped function (3.5) with initial centers and widths shown in Table 13. Figure 23 showing the membership function of input term nodes according to Table 13.

Table 13 Membership Function of Input Term Nodes

Linguistic Variable	Term	Parameter	
		Center	Width
x	1	-2.4	2.6
	2	0.0	1.5
	3	2.4	2.6
$\dot{x}$	1	-20.0	20.0
	2	0.0	10.0
	3	20.0	20.0
θ	1	-12.0	6.0
	2	-6.0	6.0
	3	-3.0	3.0
	4	0.0	3.0
	5	3.0	3.0
	6	6.0	6.0
	7	12.0	6.0
$\dot{\theta}$	1	-100.0	100.0
	2	0.0	50.0
	3	100.0	100.0

The results of three “trials” are shown in Figure 24. A run was successful and terminated after 500’000 time steps before 100 trials took place; otherwise, it was called “a failure” and terminated at the end of its 100th trial. The three trials shown in Figure 24 are three failure trials. The status of controlled plant and actual force in the cart-pole balancing problem are also shown in this figure. The example in Figure 24 is a trial in the early stage of a “run”, so it failed in only about 382 time steps. From the figure showing the cart position, we can see that the system failed at the 382th time steps because the cart bumped in to the right wall (the cart position and canter of track are greater than 2.4 meter apart). At this stage, the FLC had learned to some extent in the area around the zero state of the input space since the FLC did try to decrease the pole’s deviation angle before the failure occurred (see the figure showing the actual force). However, it still needs extensive learning in the other cart position of the input space.

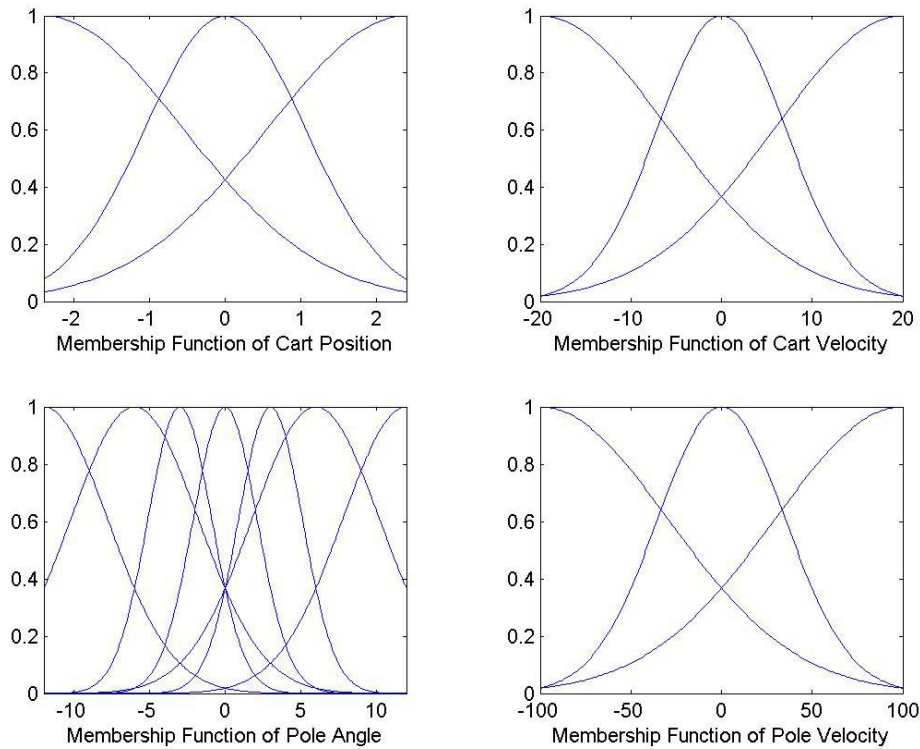
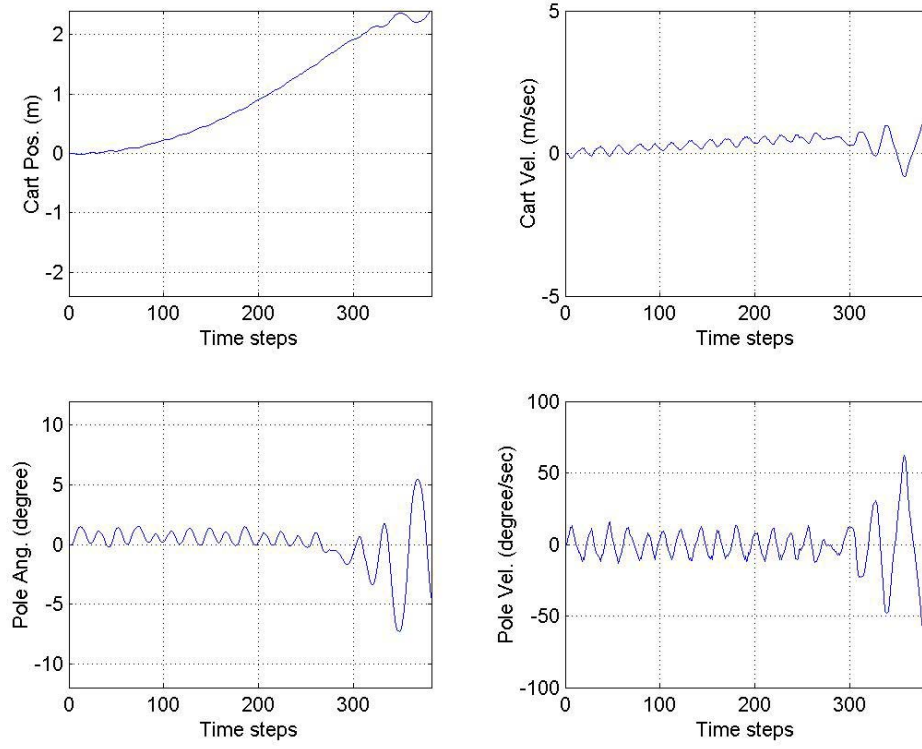


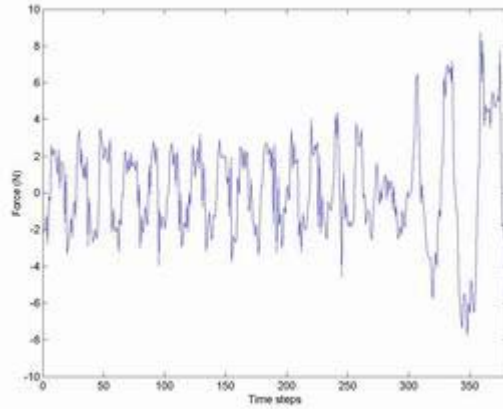
Figure 23 Membership Function of Input Term Nodes

The prior knowledge is an important role playing in the TSK type RNN-FLCS that will decide the learning performance and learning speed when initial weight is great or poor. In TSK type inference, the initialization problem is developed. We use five parameters and four variables to describe our output term node. How do I define our initial weight for TKS type?

In RNN-FLC, we can collocate to the cart position and pole angle to find the rough and usable initial weight (mean and variance) in RNN-FLCS. For example, if the cart position and canter of track are 2 meter apart and the pole angle away from the upright position -10 degree. Then, the output of action network must push to left which maintain the pole and cart within the boundary. Therefore, assuming the center of expected action is designed to -8.5 Newton in above case (Figure 25). The designed membership function is shown in Figure 26.



(a)



(b)

Figure 24 Three trials in the learning process of the NN-FLC on the cart-pole balancing problem
(a) status of controlled plant (b) actual force

In TSK type RNN-FLCS, the initialization of TSK type is more complex than other one; hence, we can not easily finding suitable initial weight. We adopt GA to solve the problem of TSK fuzzy inference initialization. The consequence of TSK is $y_i = a_i \frac{x}{x_{\text{boundary}}} + b_i \frac{\dot{x}}{\dot{x}_{\text{boundary}}} + c_i \frac{\theta}{\theta_{\text{boundary}}} + d_i \frac{\dot{\theta}}{\dot{\theta}_{\text{boundary}}} + e_i$, where a_i , b_i , c_i , d_i , and e_i are parameters of corresponding input variable respectively. We can not decide the weight of corresponding input variable from different status of controlled plant. Hence, it is very difficult job to obtain the initial value of parameter.

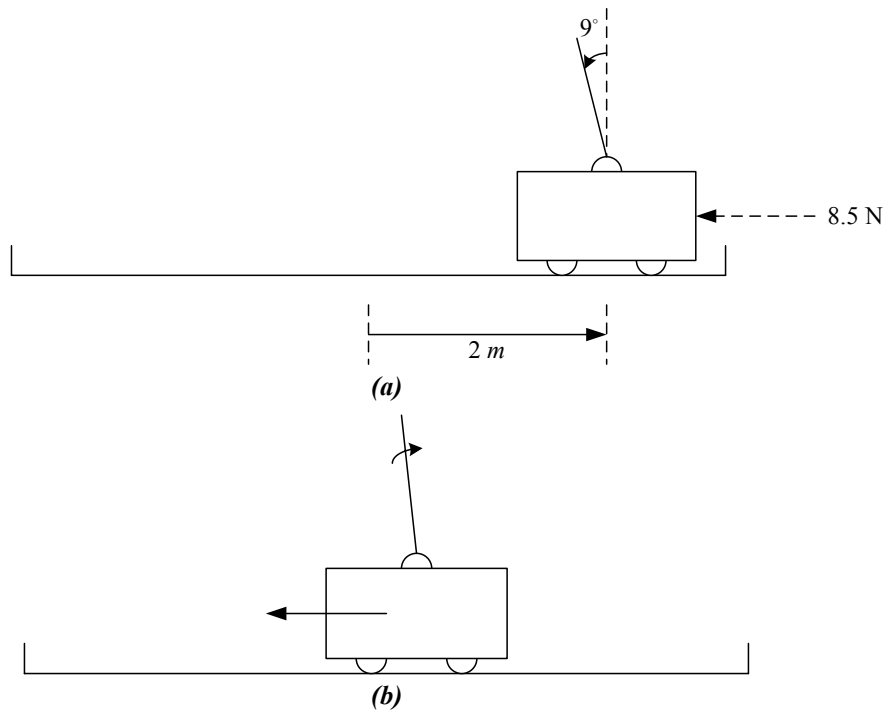


Figure 25 Design prior knowledge (a) controlled plant status (b) after control

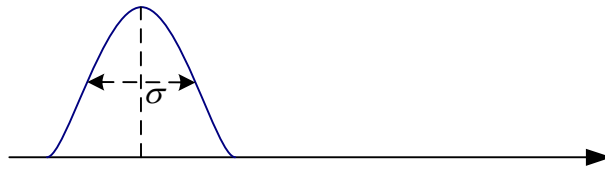


Figure 26 Designed membership function through controlled plant

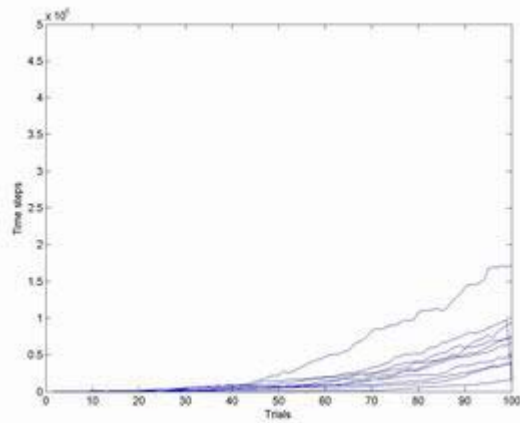


Figure 27 The performance of TSK type RNN-FLCS with random prior knowledge

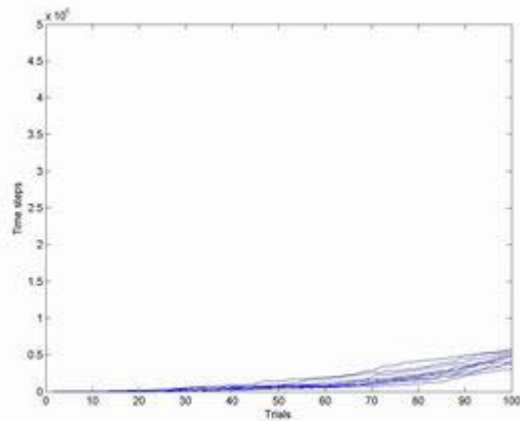


Figure 28 The performance of TSK type RNN-FLCS with zero status prior knowledge

In order to solve the problem of initialization, we first adopt random initialization and zero initialization in consequence of TSK. The random initialization means that the parameters of TSK is developed through random; the zero initialization means that is all of the parameters of TSK are zero. We are performing learning process based on the two kinds of initialization. Figure 27 and Figure 28 are respectively shown the results of random initialization and zero initialization. We are finding some issue in the initialization. In the random initialization case, because we can not control the initial values of parameter, hence, the performance of learning will be low if the initial parameters are extreme poor. Based on the above controlled plant status (Figure 25), we provide an example of extreme case to explain the problem of random initialization. If the status of controlled plant are $x = 2$, $\dot{x} = 2$, $\theta = -9$, and $\dot{\theta} = -50$ respectively. In extreme case, if the parameter of initialization are $a_i = 2$, $b_i = 2$, $c_i = -2$, $d_i = -2$, and $e_i = 2$, the value of TSK inference is $y_i = 7.17$ (Figure 29). The consequence will speed up the cart position and pole angle bump the boundary. If all of the TSK rules are extreme case, the learning performance will be bad.

In the other case, the learning performance of zero initialization case is poor. This case can not increase learning performance in condition of limited runs and trials. We are trying to raise the performance through increase learning rate, but the result is not our desired outcome. Because the linear functions are associative in different control status, the zero initialization case dose not success in the finite condition.

In order to solve the problem of initialization efficiently, we adopt the genetic algorithm to search more suitable and efficient initial parameter. Because GA can search the better solution for any problem through selection, crossover, and mutation, the GA based prior knowledge is suit for our learning process. The operation not only solves the problem of initialization but also raise our learning performance conspicuously. In Figure 30, we clearly understand that the performance of GA based prior knowledge in our algorithm is actually solving our problem and raising the performance. In our algorithm, the GA process only provides the initial value of parameter to increase performance; this process has not the function of learning.

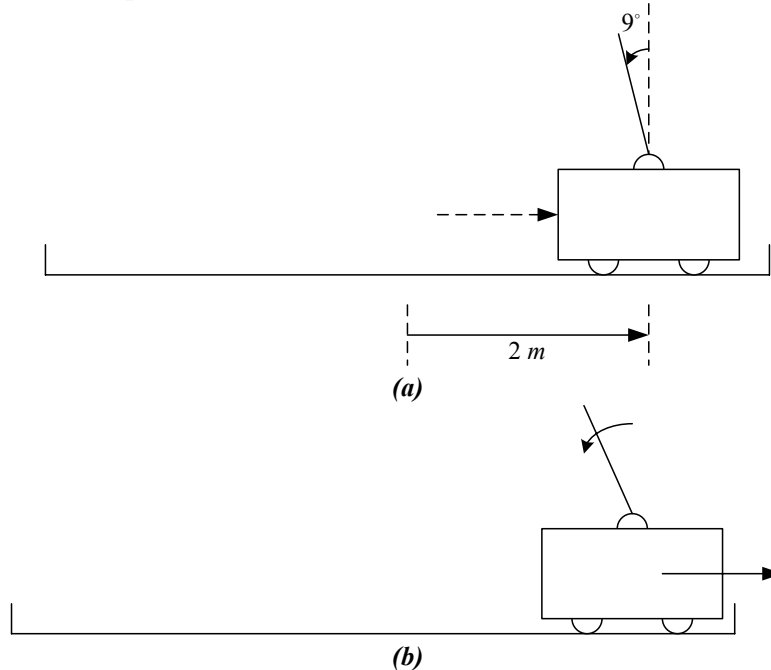


Figure 29 The extreme case of fail control (a) controlled plant status (b) after control

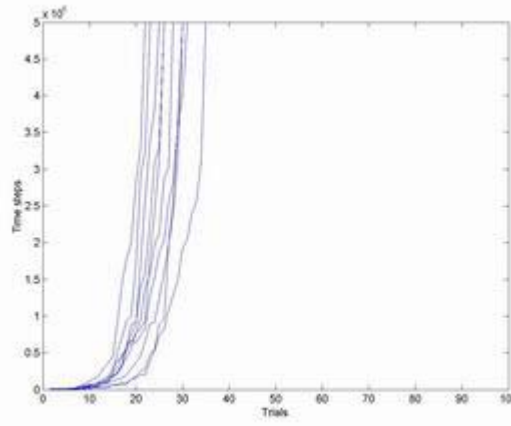


Figure 30 The performance of TSK type RNN-FLCS with GA based prior knowledge

We develop different condition to demonstrate the effects of different initial status of controlled plant. The seven cases of initial condition are shown in Table 14.

Table 14 Seven cases of initial controlled status

Case	Initial Condition		Unit
1	Cart Position (x)	0	m
	Pole Angle (θ)	0	$degree$
2	Cart Position (x)	1.8	m
	Pole Angle (θ)	0	$degree$
3	Cart Position (x)	-1.8	m
	Pole Angle (θ)	0	$degree$
4	Cart Position (x)	0	m
	Pole Angle (θ)	9	$degree$
5	Cart Position (x)	0	m
	Pole Angle (θ)	-9	$degree$
6	Cart Position (x)	1.8	m
	Pole Angle (θ)	9	$degree$
7	Cart Position (x)	-1.8	m
	Pole Angle (θ)	-9	$degree$

The simulated results are shown in follow table. In best initial condition, case 1, we tried at least 31 trials to achieve success. The average value of trials is 38.2. In worst case, case 7, we must try at least 52 trials to achieve success. The value is worse than minimum trials of case 1. Because the initial controlled plant status are close to boundary of input variables, the controlled plant easily bump boundary. Hence, in worse case, the learning process must more trials to achieve success (i.e., extra fails). The case 1 is better than other cases; the average value is smaller than others. The problem is derived which the initial controlled plant status decide the speed of learning. Table 15 is shown the performance of different initial condition.

Table 15 The performance of different initial controlled plant status (RNN-FLCS)

Control Trials			
Initial Condition	Best	Worst	Mean
Cart_Pos.=0 m Pole_Ang.=0 degree	31	46	38.2
Cart_Pos.=1.8 m Pole_Ang.=0 degree	40	57	48.0
Cart_Pos.=-1.8 m Pole_Ang.=0 degree	40	58	49.1
Cart_Pos.=0 m Pole_Ang.=9 degree	47	66	55.5
Cart_Pos.=0 m Pole_Ang.=9 degree	51	62	55.9
Cart_Pos.=1.8 m Pole_Ang.=9 degree	56	69	62.5
Cart_Pos.=-1.8 m Pole_Ang.=-9 degree	52	65	60.2

Figure 31 shows the deviations of cart position and pole angle, cart velocity, angular velocity of pole, and actual force.

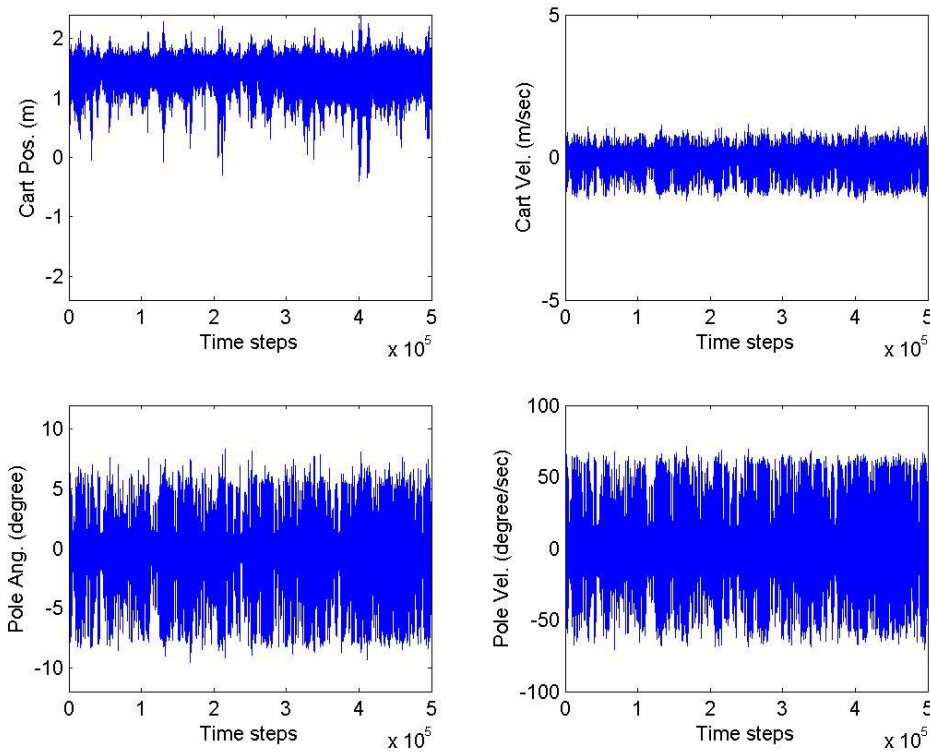


Figure 31

Consequence of learned RNN-FLCS

Based on the proposed learning process, we will perform the learning process for different initial controlled plant status. The different cases of initial controlled plant status are the same as those in Table 14.

The difference between RNN-FLCS and GA based TSK type RNN-FLCS is in layer four. In RNN-FLC, we adopt bell-shaped membership function to describe output term nodes. Therefore, an output term node is described by mean (m) and variance (σ). In GA based TSK type RNN-FLCS, the output term nodes are represented by TSK type inference system that are including four variables (x , $\dot{x}$, θ , and $\dot{\theta}$) and five parameters (a , b , c , d , and e), i.e., $y = ax + b\dot{x} + c\theta + d\dot{\theta} + e$. Comparing the inference system of layer four mentioned, the TSK type inference has more elasticity according to our requirement. Hence, the efficiency of TSK type is

better than other one.

Table 16 The performance of different initial controlled plant status (TSK type RNN-FLCS)

Control Trials			
Initial Condition	Best	Worst	Mean
Cart_Pos.=0 m Pole_Ang.=0 degree	22	35	27.6
Cart_Pos.=1.8 m Pole_Ang.=0 degree	33	44	39.3
Cart_Pos.=-1.8 m Pole_Ang.=0 degree	31	46	38.5
Cart_Pos.=0 m Pole_Ang.=9 degree	38	52	44.4
Cart_Pos.=0 m Pole_Ang.=9 degree	38	50	43.3
Cart_Pos.=1.8 m Pole_Ang.=9 degree	49	58	53.6
Cart_Pos.=-1.8 m Pole_Ang.=9 degree	56	62	58.7

In previous sections, we were obtained the performance of RNN-FLC and GA based TSK type RNN-FLCS through different initial cases. The consequence is shown in Table 17. In learning process of TSK type, because we can adapt five parameters for different input variables through backpropagation, the learning efficiency is promoted. We clearly understand the learning efficiency in this table. Figure 32 shows the deviations of cart position and pole angle, cart velocity, and angular velocity of pole.

Table 17 The comparison with the performance of RNN-FLCS and GA based TSK type RNN-FLCS

Average Trials		
Initial Condition	RNN-FLCS	GA based TSK type RNN-FLCS
Cart_Pos.=0 m Pole_Ang.=0 degree	38.2	27.6
Cart_Pos.=1.8 m Pole_Ang.=0 degree	48.0	39.3
Cart_Pos.=-1.8 m Pole_Ang.=0 degree	49.1	38.5
Cart_Pos.=0 m Pole_Ang.=9 degree	55.5	44.4
Cart_Pos.=0 m Pole_Ang.=9 degree	55.9	43.3
Cart_Pos.=1.8 m Pole_Ang.=9 degree	62.5	53.6
Cart_Pos.=-1.8 m Pole_Ang.=9 degree	60.2	58.7

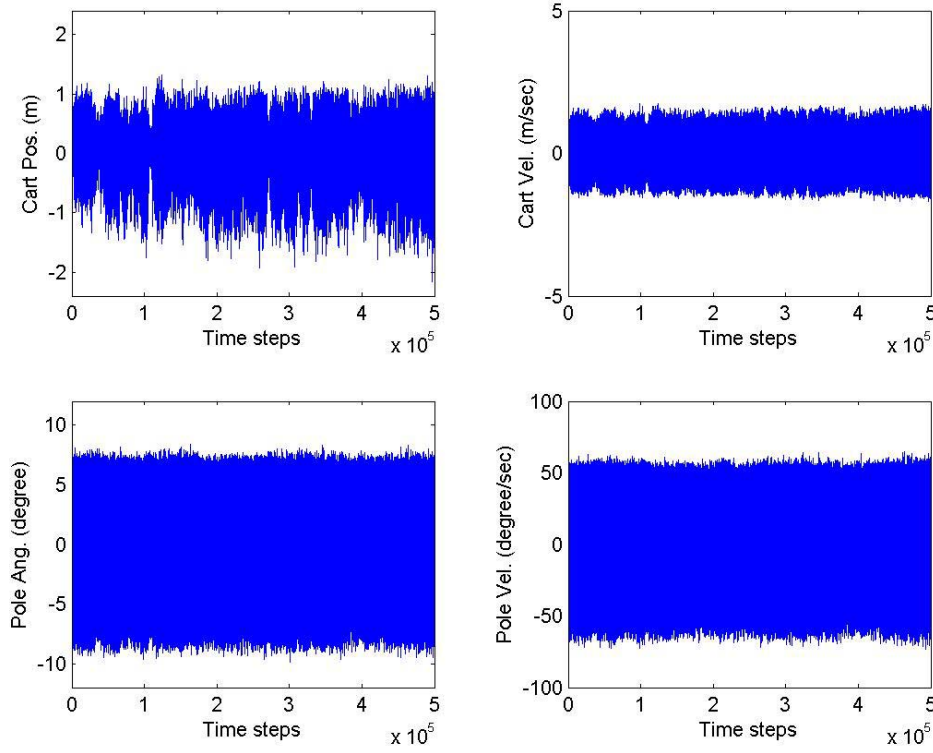


Figure 32 Consequence of learned GA based TSK type RNN-FLCS

In [66], the learning process of RNN-FLCS is including structure learning. When entering this process, it means that the i th term node in layer four is improperly assigned as the sequent of some fuzzy logic rules which have just been *strongly*. The more proper consequent for these fuzzy rules should be the i -closest node. Before structure learning, the system must decide whether the current structure should be change according to the expected updated amount of the mean and variance parameters. The expected mean and variance are, respectively, computed

$$\begin{aligned} m_{i-\text{new}} &= m_i(t) + \Delta m_i(t) \\ \sigma_{i-\text{new}} &= \sigma_i(t) + \Delta \sigma_i(t) \end{aligned} \quad (4.2)$$

From the current membership functions of output linguistic variables, [66] want to find the one which is the most similar to the expected membership function by measuring their fuzzy similarity. The fuzzy similarity measure determines the similarity between two fuzzy sets.

```

IF degree( $i, t$ ) <  $\alpha(t)$ 
  THEN (1) create a new node  $M(m_{i-\text{new}}, \sigma_{i-\text{new}})$  in layer four
        (2) denote this new node as the  $i$ -closest node
        (3) do the structure learning process
ELSE
  IF  $M(m_{i-\text{closest}}, \sigma_{i-\text{closest}}) \neq M(m_i, \sigma_i)$ 
    THEN do the structure learning process
  ELSE do the following parameter adjustment in layer five:
         $m_i(t+1) = m_{i-\text{new}}$ 
         $\sigma_i(t+1) = \sigma_{i-\text{new}}$ 

```

degree(i, t) is similarity between two fuzzy sets, $\alpha(t)$ is monotonically increasing scalar similarity criterion such that the lower similarity is allowed in the initial stages of the learning. According to this judgment, degree(i, t) is first compared to the similarity criterion. If there is not enough similarity, then a new term node (new membership function) with the expected

parameters is built because, in this case, all the current membership functions are too different from the expected one. A new node with the expected membership function is necessary, and the output connections of some just firing rule nodes should be changed to point to this new term node through the structure learning process. If no new term node is necessary, the learning algorithm will then check if the i th term node is the i -closest node. If this is false, some of the just fired fuzzy logic rules should have the i -closest (term) node instead of the original i th term node as their consequent. In this case, the structure learning process should be performed to change the current structure properly. If the i th term node is the i -closest node, then no structural change is necessary; only the parameter learning should be performed by the standard backpropagation algorithm. The learned RNN-FLCS with structure learning is shown in Figure 33. Based on the structure of Figure 33 and parameters of Table 18, the status of controlled plant is shown in Figure 34.

The problem of structure learning is developed. Because the structure learning process reduce some connection between layer two and layer three, the consequence of t -norm operation obtain different value that will change the consequence of COA. The reduced regulations are: (1) they have the same consequence, (2) some preconditions are common to all the rule nodes in this set, and (3) the union of other precondition of these rule nodes composes the whole term set of some input linguistic variables. An example is illustrated in Figure 35. Figure 36 shows the effects after reducing connection.

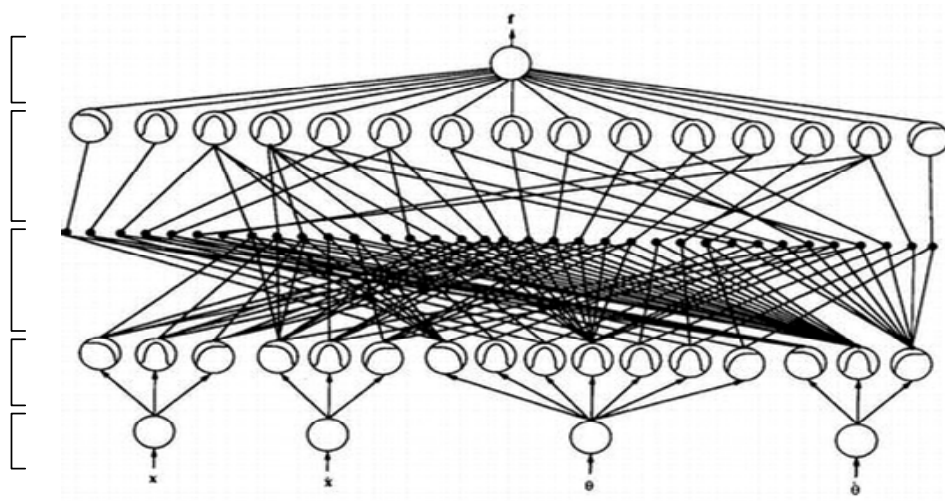
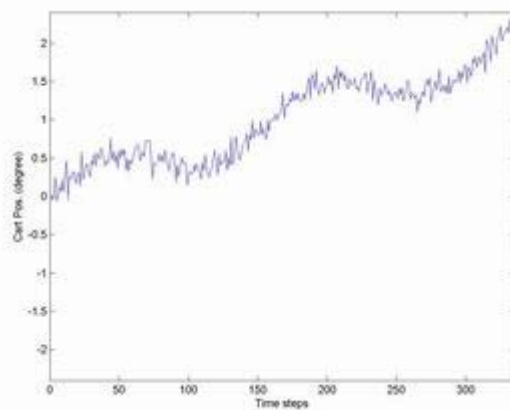
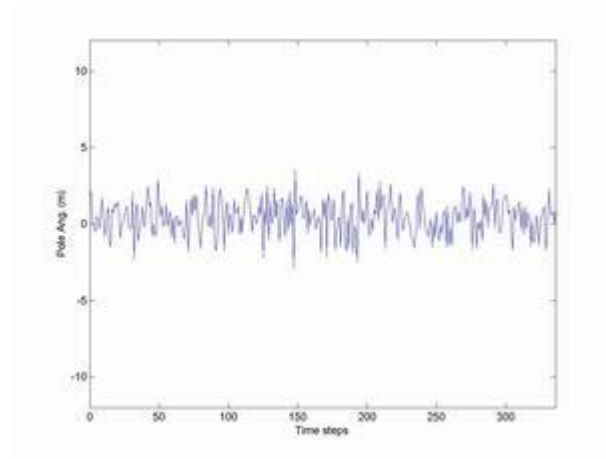


Figure 33 Learned RNN-FLCS with structure learning



(a)



(b)

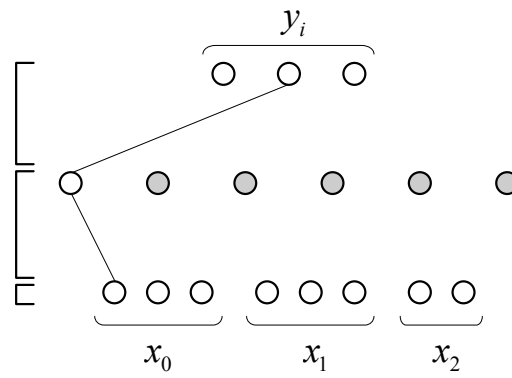
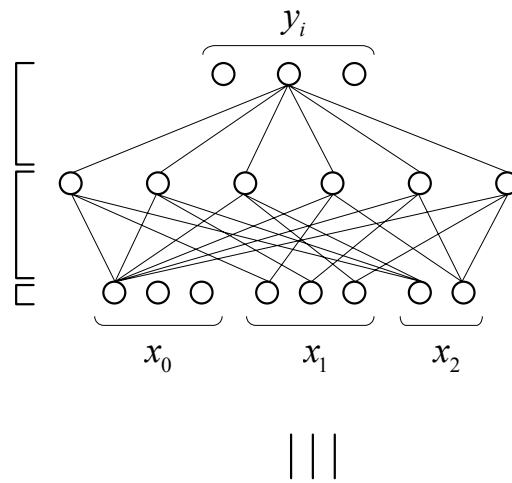
Figure 34 Status after structure learning (a) cart position (b) pole angle

Table 18 Membership Function of Input Term Nodes of After Structure Learning

Linguistic Variable	Term	Parameter	
		Center	Width
x	1	-1.98	3.11
	2	0.13	1.28
	3	2.17	4.13
$\dot{x}$	1	-15.42	17.21
	2	-0.21	7.39
	3	17.25	16.01
θ	1	-10.22	7.39
	2	-5.01	4.81
	3	-1.82	2.12
	4	0.21	1.92
	5	2.19	1.89
	6	5.31	3.87
	7	8.82	8.01
$\dot{\theta}$	1	-84.29	84.62
	2	-0.32	39.42
	3	73.18	110.39
f	1	-9.48	4.21
	2	-7.89	2.89
	3	-6.42	2.72
	4	-4.03	3.02
	5	-2.10	1.36
	6	-1.32	0.98
	7	-0.98	0.72
	8	-0.19	0.57
	9	1.12	0.33
	10	1.86	0.68
	11	2.77	1.03
	12	4.59	2.92
	13	7.21	3.57
	14	8.88	3.24
	15	10.26	4.56

In order to solve the problem, we adopt the parameter learning to re-learn the parameter of f . After parameter re-learning, we will improve the problem. The sequence of after re-learning is

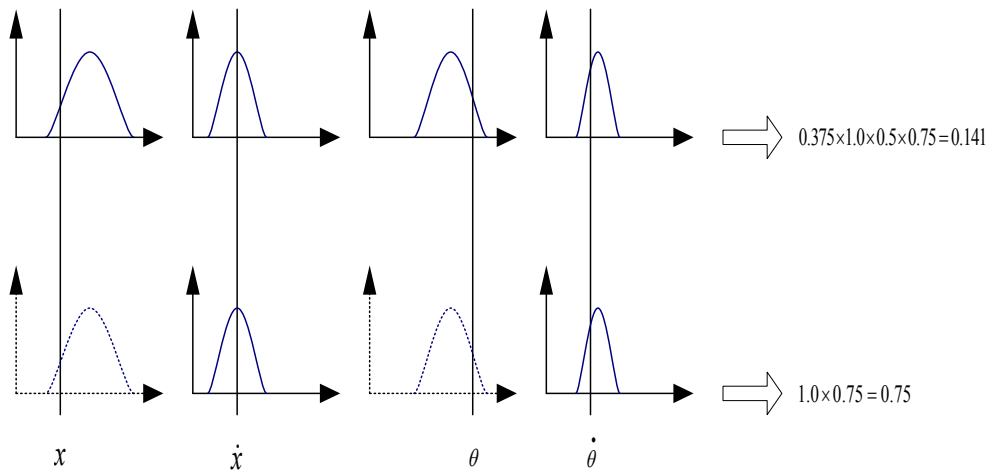
shown in Figure 37.



Layer 4

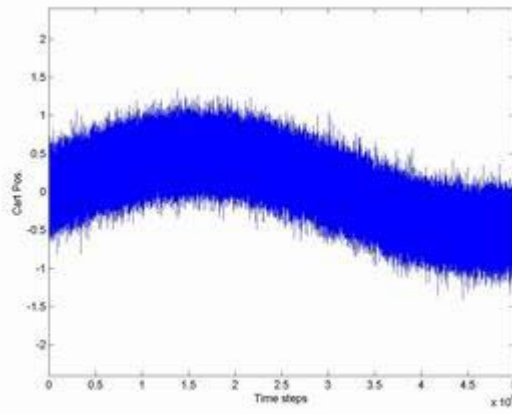
Figure 35 Example of combination of rule nodes

Layer 3

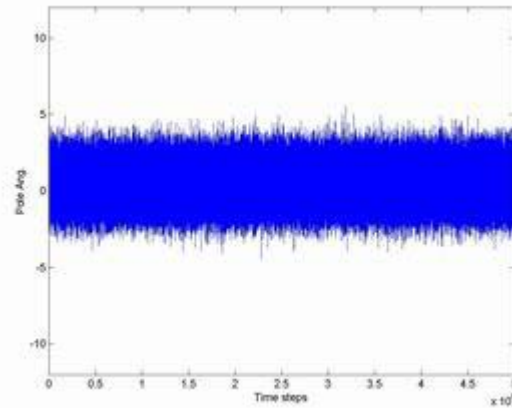


Layer 2

Figure 36 Rule strength of after structure learning



(a)



(b)

Figure 37 Status of controlled plant of parameter re-learning (a) cart position (b) pole angle

七、模糊控制器穩定設計與探討

在我們的前三年的研究中，我們提出了階層式控制的概念，在這些研究中我們探討了利用模糊預估器來做預估控制的階層式控制系統，而在此一新提計畫中，我們打算也將階層式控制用於在運動轉換的動作平台控制上使用，由於一般動作平台的控制，一般是以位置為主，而在我們交大目前已可做速度控制或加速度控制等的平台控制，可是在行為要求上，一般這三種控制都有可能在運動轉換中被要求實現，而且可能同時。因此如何達成這三種要求的控制，由於如是狀態為要求多於可控變數，尋求最佳化的控制便是一可行的方案。所以在本子畫中，我們將延續我們先前的研究，利用階層式控制在運動轉換上的探討，而由於此階層式處理的方式較適用於有運動轉換資料庫的情形，

In practical systems, there always exist nonlinearities and uncertainties. The design of controllers for such systems is difficult and complicated. During last decades, various approaches have been proposed to overcome those difficulties in the control design for nonlinear systems with uncertainty. Fuzzy control is one of them. It can provide an effective solution to the control of plants that are complex, uncertain, and ill defined. It is because a complex system is usually decomposed into several simple subsystems in a fuzzy control approach and a simple control law is then employed to emulate the human control strategy in each subsystem.

Though the fuzzy control approach is useful, its main drawback is the lack of systematic analysis and design methodologies for fuzzy logic controllers. Recently, based on the Takagi-Sugeno (T-S) fuzzy model [33,60], researchers have proposed many methods of designing controllers for nonlinear systems with uncertainties. The main feature of a T-S fuzzy model is to characterize the local dynamics of each fuzzy rule by a linear system model. The appeal of the T-S model is that the stability and performance characteristics of the system can be verified by using a Lyapunov function approach [34]-[38]. The basic idea of these methods is to design feedback controllers for individual local models respectively and then to construct a global controller from these local controllers so that the global stability of the overall fuzzy system is

guaranteed. State feedback controllers were addressed in most research works. However, in practical applications, state variables are not always measurable. Therefore, in the literatures [62]-[64], fuzzy observers are proposed to estimate inaccessible states of the controlled fuzzy systems. Although dynamic output feedback controllers can overcome the problem that states are not accessible, the design of state estimators will increase the difficulty of stabilizing the overall fuzzy system. If possible, we may like to consider the controller synthesis problem via static output feedback. This is a more challenging problem. For nonlinear systems with uncertainties or/and with time delays, no results have been reported for the controller synthesis problem via static output feedback based on the T-S fuzzy model. In this report, we shall propose an LMI-based fuzzy static output feedback control design method for nonlinear interval time-delay systems.

In traditional, the stability of a fuzzy system can be assured by finding a common positive definite matrix in a Lyapunov function. That matrix and the corresponding controller can be simultaneously found by solving a set of linear matrix inequalities (LMIs). However, LMI-based fuzzy static output-feedback gains are not obtained by traditional approaches for nonlinear systems with uncertainties or with time delays in the literatures. In this report, a new approach will be proposed to stabilize nonlinear interval time-delay systems via static output feedback. This approach does not need to find a common positive definite matrix to guarantee the system stability. The verification process is via computing the matrix measure for each separate subsystem. In this report, we shall derive the stability conditions based on the matrix measures of the system matrices of the subsystems of a fuzzy model.

Time delays occur in many dynamical systems such as biological systems, chemical systems, electrical networks, hydraulic, and rolling mill systems. Time delays are often a source of instability and poor performance. The study of stability and stabilization for a class of nonlinear time-delay systems based on the T-S fuzzy model has attracted considerable attentions [46]-[48]. As an example, Cao et al. [46] dealt with the issue of stability analysis and control design for both continuous and discrete time nonlinear systems with time delays in T-S fuzzy model-based control. Generally, to consider the stability and stabilization issue for nonlinear time-delay systems with uncertainties is desired. However, to our best knowledge, the research found in the literature is only for nonlinear systems either with uncertainties [39]-[45] or with time delays [46]-[48]. Kiriakids [39] treated the issue of stability robustness against modeling uncertainty in T-S fuzzy model-based control. Parametric uncertainties regarding those approaches are also dealt with in [40]-[45]. There also exist approaches that can handle time-delay systems with uncertainties, but only for linear systems [49]-[54]. There is no methodology that can be used in the stability analysis and control design for nonlinear systems with the existence of both uncertainties and time delays due to the intricate nature of the problem. In this report, we propose a way of conducting stability analysis and control design for nonlinear interval time-delay systems under the representation of the T-S fuzzy model. It is shown that the state-feedback controller design problem for nonlinear interval time delay systems is solvable if a matrix measure assignment problem is solvable. Afterward, the matrix measure assignment problem is shown to be equivalent to a LMI feasibility problem. Thus, the solutions can be found through LMI tools.

Takagi and Sugeno [33] proposed an elegant modeling method, which is often referred to as the T-S fuzzy model, to represent or approximate a nonlinear dynamical system. In each fuzzy rule, a linear model is given to describe the nonlinear system locally. The overall system is described by fuzzily “blending” those local models. A typical IF-THEN rule of T-S fuzzy models is represented as

$$\begin{aligned} \text{Rule } i: & \text{ IF } z_1(t) \text{ is } M_1^i \text{ and } \dots z_p(t) \text{ is } M_p^i \\ & \text{ THEN } \begin{cases} \dot{\mathbf{x}}(t) = \mathbf{A}_i \mathbf{x}(t) + \mathbf{B}_i \mathbf{u}(t) \\ \mathbf{y}(t) = \mathbf{C}_i \mathbf{x}(t) \end{cases}, \text{ for } i = 1, 2, \dots, r \end{aligned} \quad (5-1)$$

where $z_1(t) \dots z_p(t)$ are the premise variables, and M_j^i is the corresponding fuzzy set for $j = 1, \dots, p$, $\mathbf{x}(t) = [x_1(t), x_2(t), \dots, x_n(t)]^T$ is the state vector, $\mathbf{u}(t) \in \mathcal{R}^m$ is the input vector,

$\mathbf{A}_i \in \mathfrak{R}^{n \times n}$, $\mathbf{B}_i \in \mathfrak{R}^{n \times m}$, $\mathbf{C}_i \in \mathfrak{R}^{q \times n}$, $\mathbf{y}(t) \in \mathfrak{R}^q$ is the controlled output and r is the number of IF-THEN rules. In our implementation, we simply set $p=q$ and $z_1(t) = y_1(t), \dots, z_q(t) = y_q(t)$.

By using the center of gravity method for defuzzification, the final output of the T-S fuzzy model is inferred as:

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \frac{\sum_{i=1}^r w_i(z(t))(\mathbf{A}_i \mathbf{x}(t) + \mathbf{B}_i \mathbf{u}(t))}{\sum_{i=1}^r w_i(z(t))} \\ &= \sum_{i=1}^r h_i(z(t))(\mathbf{A}_i \mathbf{x}(t) + \mathbf{B}_i \mathbf{u}(t)) \\ \mathbf{y}(t) &= \frac{\sum_{i=1}^r w_i(z(t))(\mathbf{C}_i \mathbf{x}(t))}{\sum_{i=1}^r w_i(z(t))} \\ &= \sum_{i=1}^r h_i(z(t))(\mathbf{C}_i \mathbf{x}(t))\end{aligned}\tag{5-2}$$

where $\mathbf{z}(t) = [z_1(t), z_2(t), \dots, z_p(t)]^T$, $w_i(z(t)) = \prod_{j=1}^p M_j^i(z(t))$, $h_i(z(t)) = \frac{w_i(z(t))}{\sum_{i=1}^r w_i(z(t))}$, and

$M_j^i(z_j(t))$ is the fuzzy membership grade of $z_j(t)$ in M_j^i . It is assumed in this report that $w_i(z(t)) \geq 0$, $i = 1, 2, \dots, r$ and $\sum_{i=1}^r w_i(z(t)) > 0$ for all t . Therefore, $h_i(z(t)) \geq 0$, $i = 1, 2, \dots, r$ and $\sum_{i=1}^r h_i(z(t)) = 1$, for all t .

We now introduce several properties about matrix measure as follows. The matrix measure of a constant matrix $\mathbf{M}$ is defined as

$$\mu_v(\mathbf{M}) \equiv \lim_{\theta \rightarrow 0^+} \frac{(\|\mathbf{I} + \theta \mathbf{M}\|_v - 1)}{\theta}\tag{5-3}$$

where $\|\cdot\|_v$ is a suitable matrix norm (see [55]).

Lemma 1 [55]: The matrix measure has the following properties.

$$(a). \mu_v(\cdot) \text{ is convex; i. e., } \mu_v\left(\sum_{j=1}^k \alpha_j \mathbf{M}_j\right) \leq \sum_{j=1}^k \alpha_j \mu_v(\mathbf{M}_j) \text{ for all } \alpha_j \geq 0.\tag{5-4}$$

$$(b). \text{ For any norm and any constant matrix } \mathbf{M} \\ -\|\mathbf{M}\|_v \leq -\mu_v(-\mathbf{M}) \leq \text{Re } \lambda(\mathbf{M}) \leq \mu_v(\mathbf{M}) \leq \|\mathbf{M}\|_v.\tag{5-5}$$

(c). Suppose m_{ij} is the ij -th element of $\mathbf{M}$, then

$$\mu_1(\mathbf{M}) = \max_j \left[\text{Re}(m_{jj}) + \sum_{i \neq j} |m_{ij}| \right],\tag{5-6}$$

$$\mu_2(\mathbf{M}) = \max_i \left[\lambda_i(\mathbf{M} + \mathbf{M}^*) / 2 \right],\tag{5-7}$$

$$\mu_\infty(\mathbf{M}) = \max_i \left[\text{Re}(m_{ii}) + \sum_{i \neq j} |m_{ij}| \right].\tag{5-8}$$

Consider a nonlinear interval time-delay system described by the following T-S fuzzy model:

Rule i : IF $y_1(t)$ is M_1^i and ... $y_q(t)$ is M_q^i

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \hat{\mathbf{A}}_i \mathbf{x}(t) + \hat{\mathbf{D}}_i \mathbf{x}(t - \tau) + \mathbf{B}_i \mathbf{u}(t) \\ \text{THEN } \mathbf{y}(t) &= \mathbf{C}_i \mathbf{x}(t) \end{aligned} \quad , \text{ for } i = 1, 2, \dots, r \quad (5-9)$$

$$\mathbf{x}(t) = \boldsymbol{\psi}(t), \quad t \in [-\tau, 0]$$

where $\boldsymbol{\psi}(t)$ is a continuous vector valued initial function. Then, the overall system is

$$\begin{aligned} \dot{\mathbf{x}}(t) &= \sum_{i=1}^r h_i(\mathbf{y}(t)) (\hat{\mathbf{A}}_i \mathbf{x}(t) + \hat{\mathbf{D}}_i \mathbf{x}(t - \tau) + \mathbf{B}_i \mathbf{u}(t)) \\ \mathbf{y}(t) &= \sum_{i=1}^r h_i(\mathbf{y}(t)) (\mathbf{C}_i \mathbf{x}(t)) \end{aligned} \quad (5-10)$$

with $h_i(\mathbf{y}(t)) \geq 0$, $i = 1, 2, \dots, r$ and $\sum_{i=1}^r h_i(\mathbf{y}(t)) = 1$ for all t , where τ is the time-delay of the

system. Assume that the matrices $\hat{\mathbf{A}}_i$ and $\hat{\mathbf{D}}_i$ are

$$\begin{aligned} \hat{\mathbf{A}}_i &= [a_{jk}^i], \quad \underline{a}_{jk}^i \leq a_{jk}^i \leq \bar{a}_{jk}^i \\ \hat{\mathbf{D}}_i &= [d_{jk}^i], \quad \underline{d}_{jk}^i \leq d_{jk}^i \leq \bar{d}_{jk}^i \end{aligned} \quad i = 1, 2, \dots, r \quad (5-11)$$

where a_{jk}^i is the jk -th element of the matrix $\hat{\mathbf{A}}_i$, $\underline{a}_{jk}^i$ and $\bar{a}_{jk}^i$ denote a_{jk}^i 's low bound and upper bound, respectively, d_{jk}^i is the jk -th element of the matrix $\hat{\mathbf{D}}_i$, and $\underline{d}_{jk}^i$ and $\bar{d}_{jk}^i$ denote d_{jk}^i 's low bound and upper bound, respectively. We further assume that those bounds $\underline{a}_{jk}^i$, $\bar{a}_{jk}^i$, $\underline{d}_{jk}^i$, and $\bar{d}_{jk}^i$ are known real values. Denote

$$\underline{\mathbf{A}}_i = [\underline{a}_{jk}^i], \quad \bar{\mathbf{A}}_i = [\bar{a}_{jk}^i], \quad i = 1, 2, \dots, r \quad (5-12)$$

$$\underline{\mathbf{D}}_i = [\underline{d}_{jk}^i], \quad \bar{\mathbf{D}}_i = [\bar{d}_{jk}^i], \quad i = 1, 2, \dots, r \quad (5-13)$$

and define

$$\mathbf{A}_i = \frac{1}{2}(\underline{\mathbf{A}}_i + \bar{\mathbf{A}}_i), \quad \mathbf{D}_i = \frac{1}{2}(\underline{\mathbf{D}}_i + \bar{\mathbf{D}}_i), \quad i = 1, 2, \dots, r \quad (5-14)$$

$$\mathbf{M}_i = \bar{\mathbf{A}}_i - \mathbf{A}_i, \quad \mathbf{N}_i = \bar{\mathbf{D}}_i - \mathbf{D}_i, \quad i = 1, 2, \dots, r \quad (5-15)$$

Since the consequent parts of T-S fuzzy models are described by linear state equations, the linear control theory can be used to design the consequent parts of a fuzzy controller. Fuzzy controllers for stabilizing the fuzzy system (5-10) can be designed via parallel-distributed control (PDC) [34][36][56]. In PDC, fuzzy controllers share the same premise parts with (5-10); that is, the controller for Rule i is

$$\begin{aligned} \text{IF } y_1(t) \text{ is } M_1^i \text{ and } \dots y_q(t) \text{ is } M_q^i \\ \text{THEN } \mathbf{u}(t) &= \mathbf{F}_i \mathbf{y}(t), \quad i = 1, 2, \dots, r. \end{aligned} \quad (5-16)$$

Then, the overall output of this fuzzy controller is

$$\mathbf{u}(t) = \sum_{i=1}^r h_i(\mathbf{y}(t)) \mathbf{F}_i \mathbf{y}(t), \quad (5-17)$$

where $h_i(\mathbf{y}(t))$ is the same as that of the i th rule of the fuzzy system (5-10). By substituting (5-17) into (5-10), we get

$$\dot{\mathbf{x}}(t) = \sum_{i=1}^r \sum_{j=1}^r \sum_{k=1}^r h_i(\mathbf{y}(t)) h_j(\mathbf{y}(t)) h_k(\mathbf{y}(t)) ((\hat{\mathbf{A}}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k) \mathbf{x}(t) + \hat{\mathbf{D}}_i \mathbf{x}(t - \tau)) \quad (5-18)$$

Let $\hat{\mathbf{A}}_i = \mathbf{A}_i + \Delta \mathbf{A}_i$ and $\hat{\mathbf{D}}_i = \mathbf{D}_i + \Delta \mathbf{D}_i$. It is known that $\|\Delta \mathbf{A}_i\| < \|\mathbf{M}_i\|$ and $\|\Delta \mathbf{D}_i\| < \|\mathbf{N}_i\|$ (see [61]), for $i = 1, 2, \dots, r$.

We first derive a new sufficient condition to ensure the stability of the following T-S fuzzy model (without uncertainty)

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \sum_{i=1}^r h_i(\mathbf{y}(t))(\mathbf{A}_i \mathbf{x}(t) + \mathbf{D}_i \mathbf{x}(t - \tau)) \\ \mathbf{y}(t) &= \sum_{i=1}^r h_i(\mathbf{y}(t))(\mathbf{C}_i \mathbf{x}(t))\end{aligned}\quad (5-19)$$

Theorem 1: Suppose $\varepsilon > 0$. If

$$\mu_2(\mathbf{A}_i) < -\frac{1}{2\varepsilon} - \frac{1}{2}\varepsilon \|\mathbf{D}_i\|^2, \quad i = 1, 2, \dots, r, \quad (5-20)$$

then the equilibrium of the unforced system (26) is asymptotically stable for all τ .

Proof: Consider a Lyapunov function as

$$V(\mathbf{x}(t)) = \mathbf{x}^T(t) \mathbf{x}(t) + \frac{1}{\varepsilon} \int_{t-\tau}^t \mathbf{x}^T(s) \mathbf{x}(s) ds. \quad (5-21)$$

The time derivative of $V(\mathbf{x}(t))$ is

$$\begin{aligned}\dot{V}(\mathbf{x}(t)) &= 2\mathbf{x}^T(t) \dot{\mathbf{x}}(t) + \frac{1}{\varepsilon} \mathbf{x}^T(t) \mathbf{x}(t) - \frac{1}{\varepsilon} \mathbf{x}^T(t - \tau) \mathbf{x}(t - \tau) \\ &= \mathbf{x}^T(t) \left(\sum_{i=1}^r h_i(\mathbf{y}(t)) (\mathbf{A}_i \mathbf{x}(t) + \mathbf{D}_i \mathbf{x}(t - \tau)) \right) \\ &\quad + \left(\sum_{i=1}^r h_i(\mathbf{y}(t)) (\mathbf{A}_i \mathbf{x}(t) + \mathbf{D}_i \mathbf{x}(t - \tau)) \right)^T \mathbf{x}(t) \\ &\quad + \frac{1}{\varepsilon} \mathbf{x}^T(t) \mathbf{x}(t) - \frac{1}{\varepsilon} \mathbf{x}^T(t - \tau) \mathbf{x}(t - \tau) \\ &= \sum_{i=1}^r h_i(\mathbf{y}(t)) \left(\mathbf{x}^T(t) (\mathbf{A}_i^T + \mathbf{A}_i + \frac{1}{\varepsilon} \mathbf{I}) \mathbf{x}(t) \right) \\ &\quad + \sum_{i=1}^r h_i(\mathbf{y}(t)) (\mathbf{x}^T(t) \mathbf{D}_i \mathbf{x}(t - \tau)) \\ &\quad + \sum_{i=1}^r h_i(\mathbf{y}(t)) (\mathbf{x}^T(t - \tau) \mathbf{D}_i^T \mathbf{x}(t)) \\ &\quad - \frac{1}{\varepsilon} \sum_{i=1}^r h_i(\mathbf{y}(t)) (\mathbf{x}^T(t - \tau) \cdot \mathbf{x}(t - \tau))\end{aligned}\quad (5-22)$$

Set $\bar{\mathbf{x}}(t) = [\mathbf{x}^T(t) \quad \mathbf{x}^T(t - \tau)]^T$. From (5-22), we obtain

$$\dot{V}(\mathbf{x}(t)) = \sum_{i=1}^r h_i(\mathbf{y}(t)) \bar{\mathbf{x}}^T(t) \begin{bmatrix} \mathbf{A}_i^T + \mathbf{A}_i + \frac{1}{\varepsilon} \mathbf{I} & \mathbf{D}_i \\ \mathbf{D}_i^T & -\frac{1}{\varepsilon} \mathbf{I} \end{bmatrix} \bar{\mathbf{x}}(t). \quad (5-23)$$

By Schur complement, if $\mathbf{A}_i^T + \mathbf{A}_i + \frac{1}{\varepsilon} \mathbf{I} + \varepsilon \mathbf{D}_i \mathbf{D}_i^T < 0$, $\begin{bmatrix} \mathbf{A}_i^T + \mathbf{A}_i + \frac{1}{\varepsilon} \mathbf{I} & \mathbf{D}_i \\ \mathbf{D}_i^T & -\frac{1}{\varepsilon} \mathbf{I} \end{bmatrix}$ is negative definite.

From (5-20), we obtain $\mathbf{A}_i^T + \mathbf{A}_i + \frac{1}{\varepsilon} \mathbf{I} + \varepsilon \mathbf{D}_i \mathbf{D}_i^T < 0$. Then, $\dot{V}(\mathbf{x}(t)) < 0$ for all $\mathbf{x}(t) \neq 0$. This completes the proof. $\square$

Theorem 1 provides a simple method to verify the stability of the time-delay fuzzy system (19). In what follows, we shall consider the stability conditions of time-delay fuzzy systems with uncertainty.

Theorem 2: Consider the interval time-delay fuzzy system

$$\dot{\mathbf{x}}(t) = \sum_{i=1}^r h_i(\mathbf{y}(t))(\hat{\mathbf{A}}_i \mathbf{x}(t) + \hat{\mathbf{D}}_i \mathbf{x}(t-\tau)) \quad (5-24)$$

$$\mathbf{y}(t) = \sum_{i=1}^r h_i(\mathbf{y}(t))(\mathbf{C}_i \mathbf{x}(t))$$

where $\hat{\mathbf{A}}_i$ and $\hat{\mathbf{D}}_i$ are defined in (5-11). If

$$\mu_2(\mathbf{A}_i) < -\frac{1}{2\varepsilon} - \|\mathbf{M}_i\| - \frac{1}{2}\varepsilon\|\mathbf{D}_i\|^2 - \varepsilon\|\mathbf{N}_i\| \cdot \|\mathbf{D}_i\| - \frac{1}{2}\varepsilon\|\mathbf{N}_i\|^2, i=1,2,\dots,r, \quad (5-25)$$

then (5-24) is robustly asymptotically stable for all τ .

Proof: Since $2\mu_2(\hat{\mathbf{A}}_i) + \frac{1}{\varepsilon} + \|\hat{\mathbf{D}}_i\|^2 = 2\mu_2(\mathbf{A}_i + \Delta\mathbf{A}_i) + \frac{1}{\varepsilon} + \varepsilon\|\mathbf{D}_i + \Delta\mathbf{D}_i\|^2$

$$= 2\mu_2(\mathbf{A}_i) + 2\mu_2(\Delta\mathbf{A}_i) + \varepsilon\|\mathbf{D}_i\|^2 + 2\varepsilon\|\mathbf{D}_i\| \cdot \|\Delta\mathbf{D}_i\| + \varepsilon\|\Delta\mathbf{D}_i\|^2$$

$$< 2\mu_2(\mathbf{A}_i) + 2\|\mathbf{M}_i\| + \frac{1}{\varepsilon} + \varepsilon\|\mathbf{D}_i\|^2 + 2\varepsilon\|\mathbf{N}_i\| \cdot \|\mathbf{D}_i\| + \varepsilon\|\mathbf{N}_i\|^2 < 0 \quad (5-26)$$

From Theorem 1, we can conclude that (5-24) is asymptotically stable for all τ . $\square$

With the above theorems, we have the following corollary.

Corollary 1: Suppose that static output feedback gains $\mathbf{F}_j$, $j=1,2,\dots,r$, satisfy the following conditions

$$\mu_2(\mathbf{A}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k) < -\frac{1}{2\varepsilon} - \|\mathbf{M}_i\| - \frac{1}{2}\varepsilon\|\mathbf{D}_i\|^2 - \varepsilon\|\mathbf{N}_i\| \cdot \|\mathbf{D}_i\| - \frac{1}{2}\varepsilon\|\mathbf{N}_i\|^2, \quad (5-27)$$

for $i, j, k=1,2,\dots,r$, then the equilibrium of the closed-loop nonlinear time-delay system with norm-bounded uncertainties represented as

$$\dot{\mathbf{x}}(t) = \sum_{i=1}^r \sum_{j=1}^r \sum_{k=1}^r h_i(\mathbf{y}(t)) h_j(\mathbf{y}(t)) h_k(\mathbf{y}(t)) ((\hat{\mathbf{A}}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k) \mathbf{x}(t) + \hat{\mathbf{D}}_i \mathbf{x}(t-\tau)) \quad (5-28)$$

$$\mathbf{y}(t) = \sum_{i=1}^r h_i(\mathbf{y}(t))(\mathbf{C}_i \mathbf{x}(t))$$

is robustly asymptotically stable for all τ .

Corollary 1 reveals that if each constant control gain in each local model satisfies (5-27), then the overall nonlinear interval time-delay system is robustly asymptotically stable.

For simplicity of notation, define

$$\gamma_i = -\frac{1}{2\varepsilon} - \|\mathbf{M}_i\| - \frac{1}{2}\varepsilon\|\mathbf{D}_i\|^2 - \varepsilon\|\mathbf{N}_i\| \cdot \|\mathbf{D}_i\| - \frac{1}{2}\varepsilon\|\mathbf{N}_i\|^2, \quad i=1,2,\dots,r.$$

Thus, (5-27) becomes

$$\mu_2(\mathbf{A}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k) < \gamma_i, \quad i, j, k=1,2,\dots,r. \quad (5-29)$$

Now, we turn our attention to reduce the matrix measure assignment problem (5-29) to an LMI feasibility problem. For a matrix $\mathbf{U}$, define $\mathbf{U}_\perp$ as a matrix whose columns form bases of the null bases of $\mathbf{U}$. Then, we can have the following theorem.

Theorem 3:

1). The static output feedback gains $\mathbf{F}_j$, $j=1,2,\dots,r$, satisfy the following conditions

$$\mu_2(\mathbf{A}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k) < \gamma_i, \quad i, j, k=1,2,\dots,r. \quad (5-30)$$

if and only if $\mathbf{F}_j$ satisfy the following LMIs

$$(\mathbf{A}_i + \mathbf{A}_i^T - 2\gamma_i \mathbf{I}) + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k + \mathbf{C}_k^T \mathbf{F}_j^T \mathbf{B}_i^T < \mathbf{0}, \quad i, j, k=1,2,\dots,r. \quad (5-31)$$

2). There exists an $\mathbf{F}_j$ that satisfies (5-31) if and only if

$$(\mathbf{B}_i)_\perp (\mathbf{A}_i + \mathbf{A}_i^T - 2\gamma_i \mathbf{I}) (\mathbf{B}_i)_\perp^T < \mathbf{0}, \quad i=1,2,\dots,r, \quad (5-32)$$

and

$$(\mathbf{C}_i^T)_\perp (\mathbf{A}_i + \mathbf{A}_i^T - 2\gamma_i \mathbf{I}) (\mathbf{C}_i)_\perp < \mathbf{0}, \quad i=1,2,\dots,r. \quad (5-33)$$

Proof: We first prove part 1). From (5-7), it can be shown that $\mu_2(\mathbf{A}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k) < \gamma_i$, $i, j, k = 1, 2, \dots, r$, are equivalent to

$$(\mathbf{A}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k) + (\mathbf{A}_i + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k)^* - 2\gamma_i \mathbf{I} < \mathbf{0}, \quad i, j, k = 1, 2, \dots, r,$$

which are also equivalent to

$$(\mathbf{A}_i + \mathbf{A}_i^T - 2\gamma_i \mathbf{I}) + \mathbf{B}_i \mathbf{F}_j \mathbf{C}_k + \mathbf{C}_k^T \mathbf{F}_j^T \mathbf{B}_i^T < \mathbf{0}, \quad i, j, k = 1, 2, \dots, r.$$

This completes the proof of part 1).

For part 2), recall the result in [58] that given a symmetric matrix $\Psi \in \Re^{n \times n}$ and two matrices $\mathbf{U}$ and $\mathbf{V}$ both with a column dimension n , there exists a matrix Θ with compatible dimension such that $\Psi + \mathbf{U}^T \Theta^T \mathbf{V} + \mathbf{V}^T \Theta \mathbf{U} < \mathbf{0}$ if and only if $\mathbf{U}_\perp^T \Psi \mathbf{U}_\perp < \mathbf{0}$ and $\mathbf{V}_\perp^T \Psi \mathbf{V}_\perp < \mathbf{0}$. By letting $\Psi_i = (\mathbf{A}_i + \mathbf{A}_i^T - 2\gamma_i \mathbf{I})$, $\mathbf{V}_i = \mathbf{B}_i^T$, $\mathbf{U}_i = \mathbf{C}_i$, and $\Theta = \mathbf{F}$, part 2) is obvious.

□

Theorem 3 tells us that if (5-32) and (5-33) hold, then there exist static output feedback gains $\mathbf{F}_j$, $j = 1, 2, \dots, r$, that satisfy LMIs (5-31). In fact, such static output feedback gains $\mathbf{F}_j$ also satisfy (5-30). Note that static output feedback gains $\mathbf{F}_j$ satisfying LMIs (5-31) can easily be obtained by using Matlab's LMI Control Toolbox. The obtained static output feedback gains $\mathbf{F}_j$ then can also solve the considered problem.

In this section, an example is used to verify the performance of the proposed controller. Assume that an unknown nonlinear interval time-delay system can be modeled as the following simple T-S fuzzy model system:

Rule 1: IF $y_2(t)$ is M_2^1 THEN

$$\dot{\mathbf{x}}(t) = \hat{\mathbf{A}}_1 \mathbf{x}(t) + \hat{\mathbf{D}}_1 \mathbf{x}(t - \tau) + \mathbf{B}_1 \mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}_1 \mathbf{x}(t)$$

Rule 2: IF $y_2(t)$ is M_2^2 THEN

$$\dot{\mathbf{x}}(t) = \hat{\mathbf{A}}_2 \mathbf{x}(t) + \hat{\mathbf{D}}_2 \mathbf{x}(t - \tau) + \mathbf{B}_1 \mathbf{u}(t)$$

$$\mathbf{y}(t) = \mathbf{C}_2 \mathbf{x}(t)$$

where $\mathbf{x}(t) = [x_1(t) \ x_2(t) \ x_3(t)]^T$, $\psi(t) = 0$, $t \in [-\tau, 0]$, $\tau = 1$, and

$$\underline{\mathbf{A}}_1 = \begin{bmatrix} 3.8 & -5.6 & 3.5 \\ -4.3 & 19.1 & -19.2 \\ 29.4 & 3.6 & -36.5 \end{bmatrix}, \quad \overline{\mathbf{A}}_1 = \begin{bmatrix} 4.6 & -4.2 & 4.5 \\ -3.7 & 22.9 & -16.8 \\ 34.6 & 4.4 & -30.5 \end{bmatrix},$$

$$\underline{\mathbf{D}}_1 = \begin{bmatrix} -0.4 & 0.7 & -0.8 \\ 0.3 & -2.2 & -1.1 \\ -1.3 & -0.5 & -2.3 \end{bmatrix}, \quad \overline{\mathbf{D}}_1 = \begin{bmatrix} -0.2 & 1 & -0.3 \\ 0.8 & -1.8 & 0.1 \\ -0.7 & 0.5 & -1.2 \end{bmatrix},$$

$$\underline{\mathbf{A}}_2 = \begin{bmatrix} 5.2 & -6.7 & 7.1 \\ -9.3 & 11.9 & -15.9 \\ 21.9 & 8.6 & -24.6 \end{bmatrix}, \quad \overline{\mathbf{A}}_2 = \begin{bmatrix} 5.6 & -6.2 & 9.3 \\ -8.2 & 15.1 & -13.7 \\ 25.3 & 9.9 & -27.3 \end{bmatrix},$$

$$\underline{\mathbf{D}}_2 = \begin{bmatrix} -0.7 & 0.2 & -1.1 \\ 0.9 & -2.8 & -3 \\ -2.3 & -0.9 & -2.2 \end{bmatrix}, \quad \overline{\mathbf{D}}_2 = \begin{bmatrix} -0.3 & 0.8 & -0.7 \\ 1.3 & -1.5 & -1 \\ -1.7 & 0.2 & -1.6 \end{bmatrix},$$

$$\mathbf{B}_1 = \begin{bmatrix} 1 & -7 \\ 5 & -3 \\ 3 & -4 \end{bmatrix}, \quad \mathbf{B}_2 = \begin{bmatrix} 1 & -6 \\ 3 & -1 \\ 2 & -3 \end{bmatrix},$$

$$\mathbf{C}_1 = \begin{bmatrix} -7 & 8 & 1 \\ 2 & -5 & -3 \end{bmatrix}, \quad \mathbf{C}_2 = \begin{bmatrix} -6 & 9 & 2 \\ 1 & -4 & -2 \end{bmatrix}.$$

Figure 38 shows the membership functions of M_2^1 and M_2^2 for $y_2(t)$. The following simulated systems are chosen in random from the above low-bounded and upper-bounded data. Figure 39 shows the output responses of the unforced system for the initial condition $\mathbf{x}(0) = [10 \ 15 \ 5]^T$ for $t \in [0, 1]$. Now we want to design a fuzzy controller via PDC to stabilize the system. By using Theorem 3 and setting $\varepsilon = 1$, we obtain the static output feedback gains of the fuzzy controller

$$\mathbf{F}_1 = \begin{bmatrix} -6.4121 & -8.2238 \\ -3.8267 & -7.4992 \end{bmatrix},$$

$$\mathbf{F}_2 = \begin{bmatrix} -5.5867 & -5.5340 \\ -4.0118 & -7.2285 \end{bmatrix}.$$

It is easy to check the stability of the overall fuzzy system by (5-27). Figure 40 shows the output responses of y_1 in the closed-loop system for various initial conditions $\mathbf{x}(0)$'s. Figure 41 shows the output responses of y_2 for various initial conditions $\mathbf{x}(0)$'s. Figure 42 shows the responses of x_1 for various initial conditions $x_1(0)$'s. Figure 43 shows the responses of x_2 for various initial conditions $x_2(0)$'s. Figure 44 shows the responses of x_3 for various initial conditions $x_3(0)$'s. From the these simulation results, it is evident that the designed T-S fuzzy model based static output feedback controller not only can stabilize the nonlinear system, but has strong robustness against norm-bounded uncertainties and time delays.

In this research, we have proposed a new robust fuzzy static output feedback controller design methodology for nonlinear interval time-delay systems with norm-bounded uncertainties and time delays. Finding an admissible solution to the matrix measure assignment problem can solve the problem of stabilizing controller design via static output-feedback for nonlinear interval time-delay systems represented by a T-S fuzzy model. The designed controller can indeed tolerate norm-bounded parametric uncertainties and against the effect of time delays. In the report, we have shown that the matrix measure assignment problem is equivalent to an LMI feasibility problem. A sufficient condition for the existence of the fuzzy static output-feedback gain is obtained. Our approach does not need to find a common positive definite matrix for all rules and the verification of stability is very easy. Simulation results have verified and confirmed the effectiveness of the new approach in controlling a nonlinear interval time-delay system.

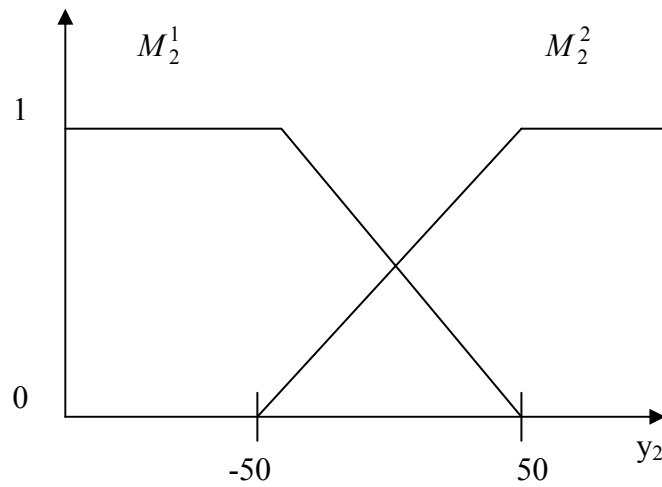


Fig. 38. Membership function of the example.

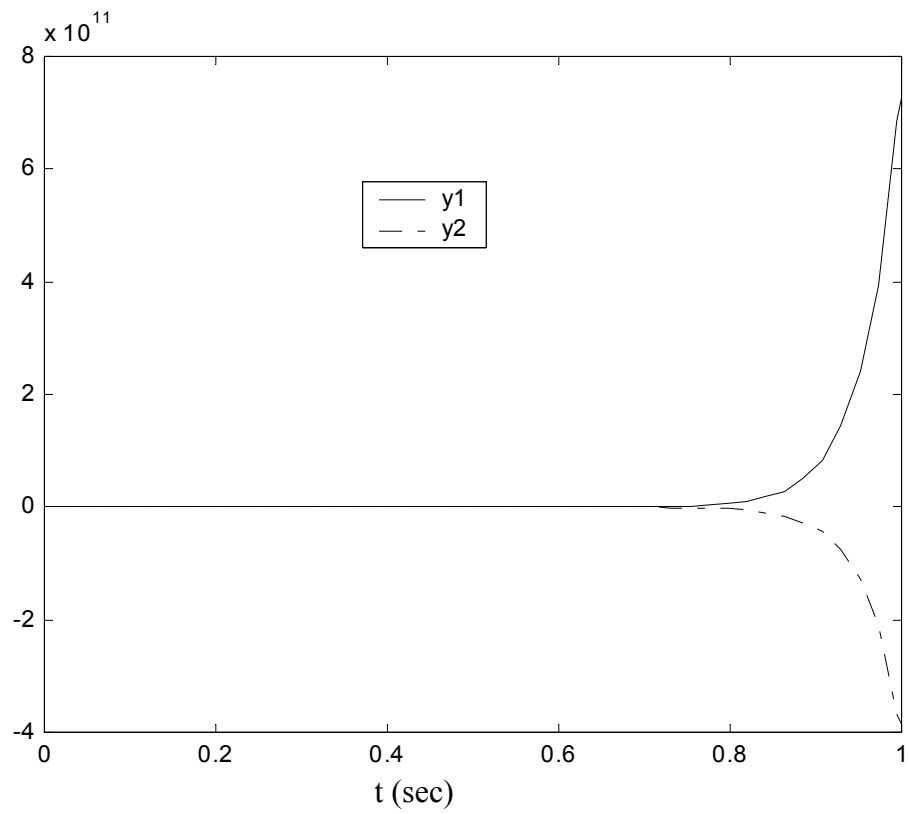


Fig. 39. Output response of the unforced system for $x(0)=[10 \ 15 \ 5]^T$.

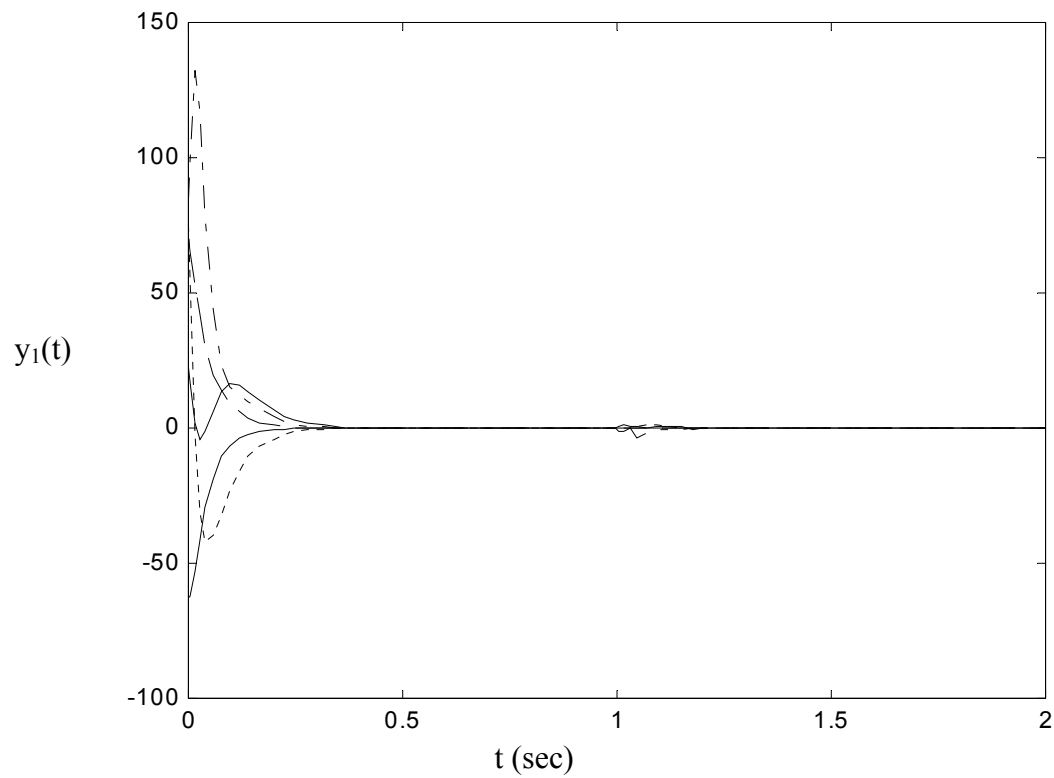


Fig. 40. Output $y_1(t)$ response of the closed-loop system for various $x(0)$ and $\tau = 1$.

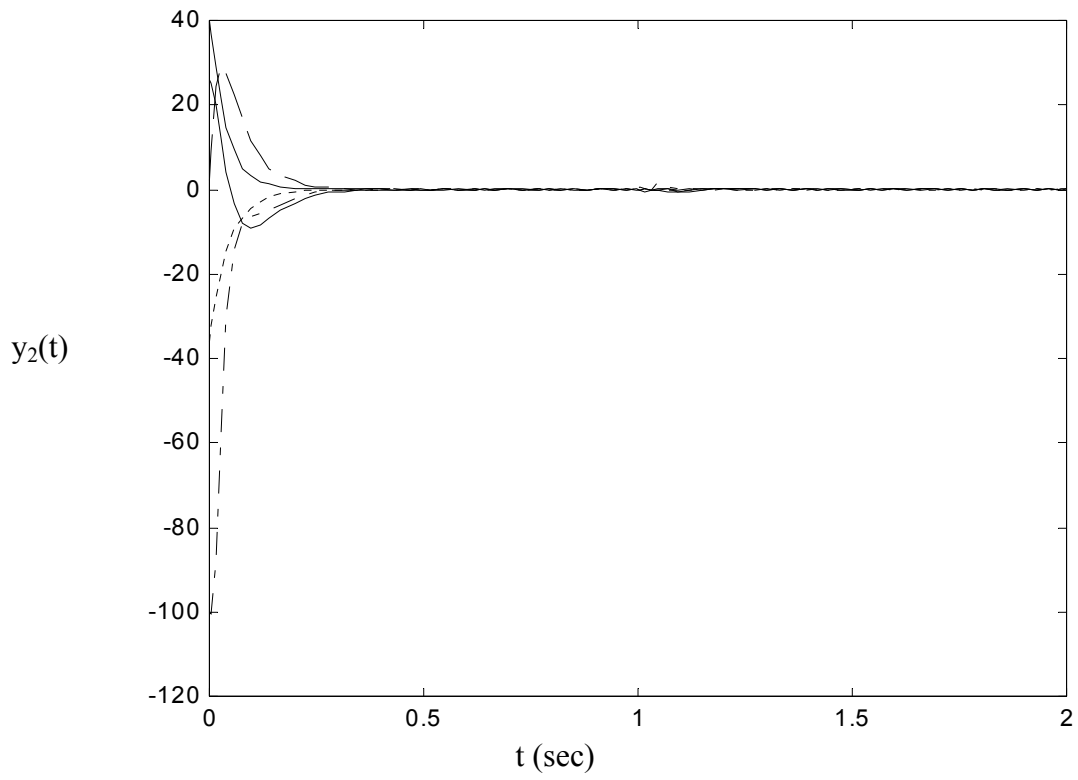


Fig. 41. Output $y_2(t)$ response of the closed-loop system for various $x(0)$ and $\tau = 1$.

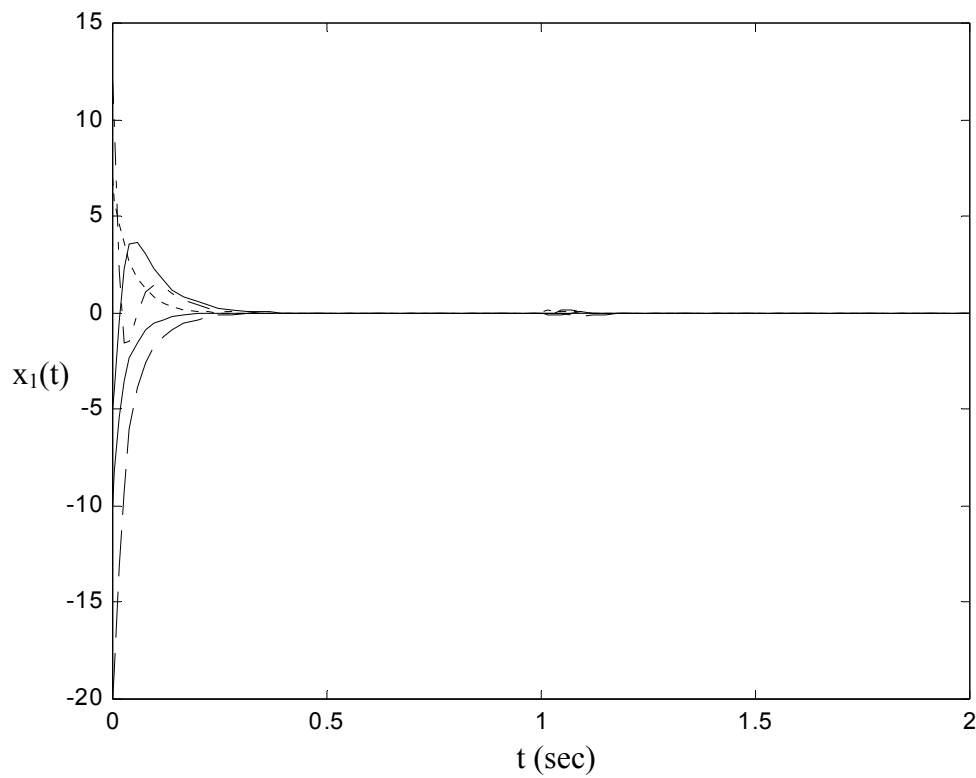


Fig. 42. State $x_1(t)$ response of the closed-loop system for various $x_1(0)$ and $\tau = 1$.

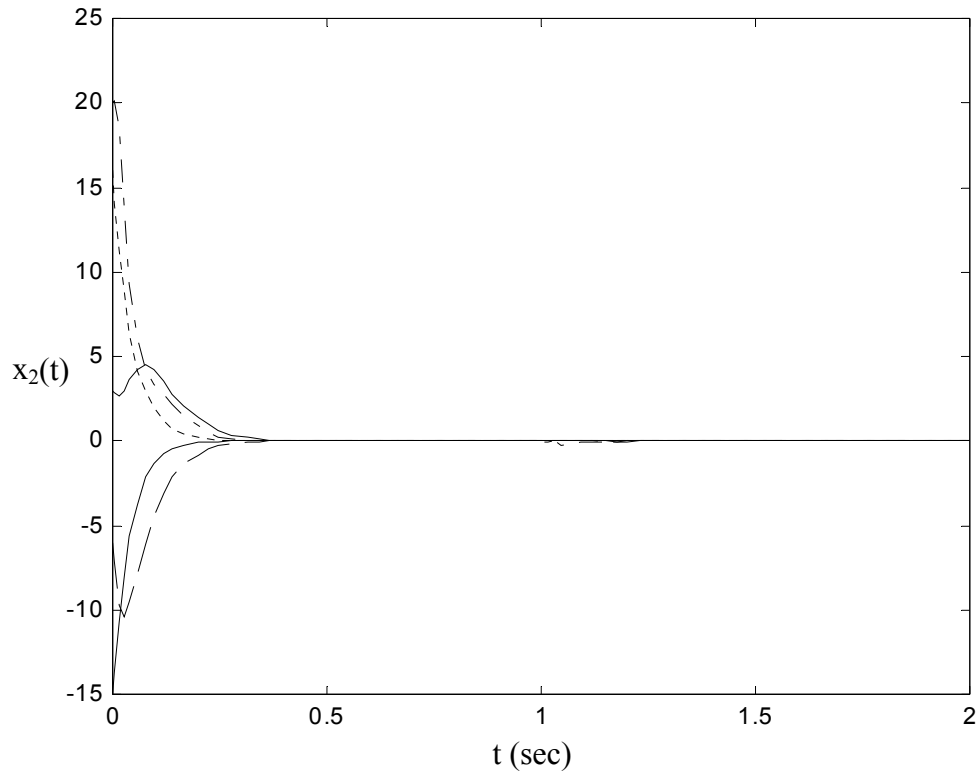


Fig. 43. State $x_2(t)$ response of the closed-loop system for various $x_2(0)$ and $\tau = 1$.

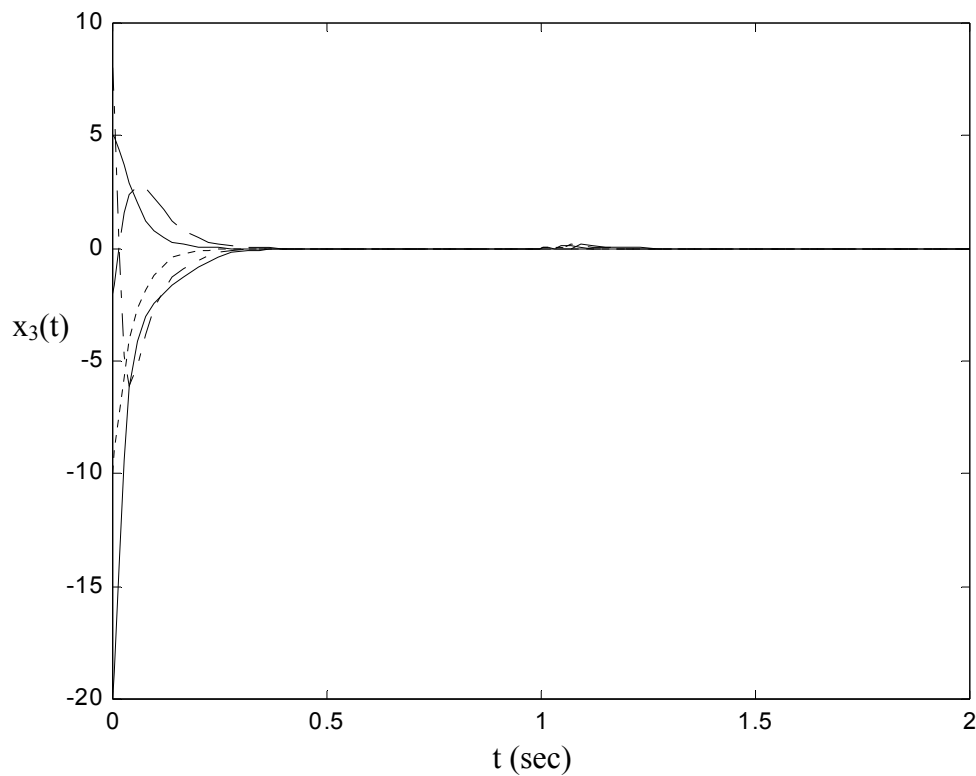


Fig. 44. State $x_3(t)$ response of the closed-loop system for various $x_3(0)$ and $\tau = 1$.

The study of stable analysis and control design for fuzzy systems has attracted many researchers recently. In the control design study, there are two types of control design problems found in the literature. The first one is to design a fuzzy controller for a nonlinear or unknown system and the other one is to find a controller for a fuzzy system. In the former case, a so-called adaptive fuzzy control is usually used. Generally, the basic objective of adaptive control is to maintain consistent control performance in the presence of large uncertainties or unknown

variations in plant parameters or structure [71]-[74]. In those reports, some adaptive laws are proposed to guarantee the stability of the system in the sense of Lyapunov. By combining other learning approaches or by considering different control properties, various variations of adaptive fuzzy control approaches have been proposed in the literature [75]-[83]. In this report, we reported our study on the issue of finding a suitable controller for a fuzzy system. In this study, the structure and parameters of the fuzzy controller are fixed and we shall find a way of designing fuzzy controllers to satisfy requirements.

For the analysis of stability of fuzzy system, based on the Lyapunov stability criterion, Tanaka and Sugeno [84] have proposed a useful theory for conservatively assuring the stability of a TSK or T-S fuzzy model [85]. The theory states that if a common positive definite matrix required in Riccati equations can be found for all fuzzy rules, the fuzzy model is stable in the sense of Lyapunov. Thathachar *et al.* [86] also proposed a necessary condition and a sufficient condition for the stability of fuzzy systems. They showed that under a formal sufficient condition, a common Lyapunov matrix exists for all subsystems. It is easy to see that to analytically find such a matrix is a linear matrix inequality (LMI) problem [87][111]. Based on the LMI concept, Kim *et al.* [88] proposed a numerical stability analysis method for singleton-type linguistic fuzzy control systems. Most of recent fuzzy control design approaches are to employ the parallel-distributed compensation (PDC) concept [89], in which controllers are designed individually for fuzzy rules and then stability condition is checked through LMI to validate the design. Some relaxed LMI stability conditions are also proposed in [90,91]. Parametric uncertainties regarding those approaches are dealt with in [92-94]. However, those methods are stability checking approaches instead of design approaches. When the designed controller cannot satisfy the stability criterion, another controller is tried. Such a process is somehow a trial-and-error design procedure. Lam *et al.* [95] proposed a designing method that can be used to help solving the relaxed LMI problem and in the approach, the feedback gains of the subsystem are determined by using a genetic algorithm, which can also be viewed as a trial-and-error procedure. Another problem rises when the number of rules is large because it may be difficult to find solutions for LMI in that case. Moreover, since the validation of LMI is only for stability, the control performance assured in individual rules cannot be anticipated in the overall system performance under those design approaches.

Several researchers [91,96,97] actually proposed ways of designing controllers by directly solving LMIs. Gao *et al.* [96] and Tanaka *et al.* [90] used LMIs combined with stable conditions of fuzzy systems to design a stable fuzzy system. When the control performance is considered, those approaches need to include performance constraints into LMIs for all fuzzy rules [97]. Korba *et al.* [98] proposed an extended fuzzy scheduler (EFS) controller that can guarantee the stability and the tracking performance requirements. The idea is to introduce an additional tracking-error state variable into the state variables of the controlled system, and then an extended-state feedback-gain matrix is design by means of LMIs. Cao *et al.* [99] also developed a two-part controller; one is a state-feedback matrix like that in [87,90] and the other is to satisfy global stable conditions. Note that in these approaches, the homogeneous fuzzy systems are considered. Here, “homogeneous” means that the consequent parts are linear equations without constant terms. Such fuzzy systems can only represent a certain class of nonlinear systems. Kim *et al.* [100] proposed a discrete iterative LMI approach to analyze stability for discrete affine fuzzy systems, where “affine” means that the consequent parts are linear equations with constant terms. However, the above approaches may become infeasible, when the number of rules is large. It is because the analysis techniques are to find a common positive definite matrix for all subsystems (fuzzy rules) to fulfill the Lyapunov stability analysis. It is difficult to find such a matrix when the rule number is large. Sometimes, there even does not exist such a matrix.

There also exist other approaches that do not make use of the common Lyapunov matrix validation scheme. Kang *et al.* [101] proposed a controller that is inferred by using their consistency conditions [102], but the desired performance has not yet been addressed. Kiriakidis *et al.* [103] viewed a TSK fuzzy model as a linear system subject to a class of nonlinear uncertainties and claimed that the computational cost of checking such a stability criterion is less than that of finding a common Lyapunov matrix. However, this approach leads to a convex

programming problem and the design procedure is still in a trial-and-error manner. Feng [104] proposed a sufficient condition, which leads to a search algorithm for solving a Riccati equation, and then a state feedback can be designed accordingly for such a system. Zak [105] also proposed a design algorithm to obtain a controller, which is separated into two parts to stabilize the nominal system and variations residing in all fuzzy rules, respectively. This approach is to treat variations of all rules as a whole in a robust control sense. For those approaches, when subsystems are distributed in a wide range, the upper bound of such variations may become large, and then the stabilization algorithm may fail. In addition, the control performance of the closed-loop system cannot be anticipated. In this report, we proposed a novel robust control design approach that can design controllers directly for affine continuous or discrete time fuzzy systems and the closed-loop performance can be theoretically anticipated.

The considered problem is that a system is described by a set of fuzzy rules and we want to develop a way of designing controllers for such a system. The considered fuzzy models are first introduced in [84] and the i th rule of a discrete system is:

$$\begin{aligned} \text{Rule } i : & \text{ IF } x_1(k) \text{ is } X_1^i \text{ and } \cdots \text{ and } x_n(k) \text{ is } X_n^i \\ & \text{ THEN } x_n(k+1) = a_0^i + a_1^i x_1(k) + a_2^i x_2(k) + \cdots + a_n^i x_n(k) \\ & \quad + b_1^i u(k) + \cdots + b_m^i u(k-m+1) \end{aligned} \quad (6-1)$$

where $x_j(k) = x_{j-1}(k+1)$ for $j=2, \dots, n$, $u(k)$, ..., and $u(k-m+1)$ are the input and m past inputs, X_j^i are fuzzy sets, and $a_0^i, a_1^i, \dots, a_n^i$ and $b_1^i, \dots, b_m^i$ are parameters in describing the input and output relationships in the i th fuzzy rule. Note that the premise part is only determined by $x_1(k), x_2(k), \dots$, and $x_n(k)$ and is independent to $u(k), \dots, u(k-m+1)$. This relationship matches the properties of common systems. Besides, the consequence part contains a constant term, a_0^i . Such a fuzzy model is referred to as a class of affine TSK fuzzy model [100].

The consequence part can be written in a state-space form as $x(k+1) = A^i x(k) + B^i u(k) + d^i$,

$$\text{where } x(k) = [x_1(k), \dots, x_n(k)]^T, \quad A^i = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ a_1^i & a_2^i & a_3^i & \cdots & a_n^i \end{bmatrix}, \quad B^i = [0, 0, \dots, b_1^i]^T, \quad \text{and}$$

$d^i = [0, \dots, a_0^i + b_2^i u(k-1) + \cdots + b_m^i u(k-m+1)]^T$. Let $\mu_j^i(x_j(k))$ be the membership degree of the $x_j(k)$ in the fuzzy set X_j^i . Then the overall system can be inferred as:

$$x(k+1) = \sum_{i=1}^r w_i A^i x(k) + \sum_{i=1}^r w_i B^i u(k) + \sum_{i=1}^r w_i d^i, \quad (6-2)$$

$$\text{where } r \text{ is the rule number, } w_i = \frac{\prod_{j=1}^n \mu_j^i(x_j(k))}{\sum_{i=1}^r \left(\prod_{j=1}^n \mu_j^i(x_j(k)) \right)} \quad \text{with } 0 \leq w_i \leq 1, \text{ and } \sum_{i=1}^r w_i = 1.$$

For a continuous TSK fuzzy model system, the i th rule is of the form:

$$\text{Rule } i : \text{ IF } x_1(t) \text{ is } X_1^i \text{ and } \cdots \text{ and } x_n(t) \text{ is } X_n^i \quad (6-3)$$

$$\text{ THEN } \dot{x}_n(t) = a_0^i + a_1^i x_1(t) + a_2^i x_2(t) + \cdots + a_n^i x_n(t) + b_1^i u(t)$$

where $x_j(t) = \dot{x}_{j-1}(t)$ for $j=2, \dots, n$ with $\dot{x}_{j-1}(t)$ is the time derivative of $x_{j-1}(t)$. Then, the consequence part can also be written as $\dot{x}(t) = A^i x(t) + B^i u(t) + d^i$, where $x(t) = [x_1(t), \dots, x_n(t)]^T$, A^i and B^i are the same as those in the discrete case, and $d^i = [0, 0, \dots, a_0^i]^T$. Similarly, the related items for a continuous affine TSK fuzzy model can be

defined.

In the literature, based on the Lyapunov theorem, several design approaches have been proposed to design stable controllers for homogenous fuzzy models; i.e., the constant vector is absent in (6-2). One of the most commonly used approaches is the Parallel Distributed Compensation (PDC) control [89,106], in which fuzzy controllers share the same premise parts with the considered fuzzy systems. The sufficient conditions for the stability of the closed-loop system are summarized in [90]. It is required that the stability of the overall system be checked by solving LMIs [87]. If the common Lyapunov matrix cannot be found, another design is tried. It should be noted that the number of inequalities to be solved is $r(r+1)/2$. It can be found that when the rule number is large, it may be very difficult to solve those inequalities.

As mentioned in above, there is another kind of approaches [103,105] in which a controller is defined for the nominal system of fuzzy rules first, and then the deviations from this nominal system existing in all rules are dealt with in a robust control fashion. Our approach shared the same idea as this kind of approach. The nominal system can be defined as $A_0 \in \text{conv}(A^1, \dots, A^r)$

and $B_0 \in \text{conv}(B^1, \dots, B^r)$, where $\text{conv}(A^1, \dots, A^r) = \left\{ \sum_{i=1}^r w_i A^i : w_i \geq 0, \sum_{i=1}^r w_i = 1 \right\}$ is a convex hull defined by the set of vertex matrices $A^1, \dots, A^r$ and a similar convex hull is defined

for $\text{conv}(B^1, \dots, B^r)$. Let $A_0 = \begin{bmatrix} 0 & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 1 \\ a_1^0 & a_2^0 & \dots & a_n^0 \end{bmatrix}$ and $B_0 = [0, \dots, b_0]^T$. The formation of A_0

and B_0 can be arbitrary as long as they are stabilizable. Usually, a simple arithmetic average is taken; that is, $a_j^0 = (\text{Max}_i \{a_j^i\} + \text{Min}_i \{a_j^i\})/2$ and $b_0 = (\text{Max}_i \{b_1^i\} + \text{Min}_i \{b_1^i\})/2$. The differences between (A^i, B^i) and (A_0, B_0) are:

$$A_i = A^i - A_0 = \begin{bmatrix} 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 \\ a_{1i} & a_{2i} & \dots & a_{ni} \end{bmatrix}, \quad B_i = B^i - B_0 = [0, \dots, b_i]^T \quad (6-4)$$

where $a_{ji} = a_j^i - a_j^0$ and $b_i = b_1^i - b_0$. Then, the system (2) can be rewritten as

$$\begin{aligned} x(k+1) &= \sum_{i=1}^r w_i A^i x(k) + \sum_{i=1}^r w_i B^i u(k) + \sum_{i=1}^r w_i d^i \\ &= (A_0 + \Delta A)x(k) + (B_0 + \Delta B)u(k) + \sum_{i=1}^r w_i d^i \end{aligned} \quad (6-5)$$

where $\Delta A = \sum_{i=1}^r w_i A_i$ and $\Delta B = \sum_{i=1}^r w_i B_i$. Similarly, with the same definition as above, a continuous affine TSK fuzzy model can be written as

$$\dot{x}(t) = (A_0 + \Delta A)x(t) + (B_0 + \Delta B)u(t) + \sum_{i=1}^r w_i d^i. \quad (6-6)$$

Note that ΔA and ΔB are regarded as uncertainties in this kind of approaches.

The design idea is to design a controller for each local model under the idea of Parallel Distributed Compensation (PDC) control [89,106], in which fuzzy controllers share the same premise parts with the considered fuzzy systems. Then, the i th control rule can be written as

Rule i : IF x_1 is X_1^i and $\dots$ and x_n is X_n^i for $i = 1, \dots, r$.
THEN $u = F_i(x)$

Thus, the overall fuzzy controller is $u = \sum_{i=1}^r w_i F_i(x)$. In our study, a novel design approach, which consists of two parts of controllers to stabilize the nominal system and variations residing in all rules, respectively, is proposed. The propose feedback control law can be written as

$$u = Gx + G_c(x), \quad (6-7)$$

where G is to control the nominal system with the desired control performance and $G_c(x)$ is to compensate the variations in all rules individually. In the following, we shall introduce how the above control law is obtained. This type of control laws is called the hybrid compensation control (HCC) in our study. The control law proposed in [103] is referred to as the nominal compensation control (NCC) and the PDC controller [89,106] is called the distributed compensation control (DCC). [105] also proposed a similar controller, in which the second part is to treat those variations as a whole instead of individually in a robust control sense. If those subsystems are distributed in a wide range, the required norm bound conditions may not be satisfied. In that case, the design algorithm will fail. This can be seen in our example shown later. For the proposed HCC, it is not required to find a common Lyapunov matrix for all subsystems. This advantage greatly increases the applicability of our approach, especially when the rule number of the system is large.

As mentioned earlier, ΔA and ΔB (in Eq. (6-5) for the discrete case or in Eq. (6-6) for the continuous case) are regarded as uncertainties in our approach. For those terms, the following assumptions are made:

A1) The nominal part of the system (6-5) or (6-6) is stabilizable.

A2) Uncertainties may vary with time within a prescribed bounded set.

A3) There exists a $1 \times n$ vector $\tilde{a}$, a scalar $\tilde{b}$ such that $\Delta A = B_0 \tilde{a}$ and $\Delta B = \tilde{b} B_0$.

Those assumptions are also used in [107] and [108] in dealing with uncertainties in the system matrix and input channels. In our case, since those uncertainties are in fact the summation of the deviations of all rules, it is easy to see that those assumptions can easily be satisfied. This can be verified in our control design for various systems. The control law designing methods for discrete and continuous TSK fuzzy models are derived in the following two subsections.

A. Discrete-time Case

Consider a discrete TSK fuzzy model described by (6-5). Let the control input be (7), where $G = [g_1 \ g_2 \ \dots \ g_n]$ is to stabilize the nominal model in (6-5) and $G_c(x)$ is to compensate the variations of all rules. Substituting (6-7) into (6-5), we have

$$x(k+1) = (A_0 + B_0 G)x(k) + B_0 (\tilde{a}x(k) + \tilde{b}G_c(x) + \tilde{d}) = \bar{A}_0 x(k) + B_0 h(x) \quad (6-8)$$

where $\tilde{a} = \sum_{i=1}^r k_i \tilde{a}_i$ with $\tilde{a}_i = \left[\frac{a_{1i} + b_i g_1}{b_0}, \frac{a_{2i} + b_i g_2}{b_0}, \dots, \frac{a_{ni} + b_i g_n}{b_0} \right]$, $\tilde{b} = \sum_{i=1}^r k_i \tilde{b}_i$ with

$\tilde{b}_i = 1 + \frac{b_i}{b_0}$, $\tilde{d} = \sum_{i=1}^r k_i \tilde{d}_i$ with $\tilde{d}_i = \frac{d^i}{b_0}$, and $h(x) = \tilde{a}x(t) + \tilde{b}G_c(x) + \tilde{d}$. In order to let the

system be stable, there must exist symmetry positive definite matrices P and Q such that $\bar{A}_0^T P \bar{A}_0 - P = -Q$. Define $V(x(k)) = x^T(k) P x(k)$ and we have

$$\Delta V(x(k)) = -x^T(k) Q x(k) + h(x)[\alpha h(x) + 2\beta(x)]$$

where $\alpha = B_0^T P B_0$, $\beta(x) = B_0^T P \bar{A}_0 x(k)$. It is easy to verify that when $h(x)[\alpha h(x) + 2\beta] \leq 0$, the system is stable. Since $h(x) = \tilde{a}x(t) + \tilde{b}G_c(x) + \tilde{d}$, if the compensation part is set as

$G_c(x) = -\frac{\tilde{a}x(k) + \tilde{d}}{\tilde{b}}$, then $h(x) = 0$. Hence we have

$$G_c(x) = -\frac{\tilde{a}x(k) + \tilde{d}}{\tilde{b}} = -\frac{\sum_i w_i \tilde{a}_i}{\sum_i w_i \tilde{b}_i} x(k) - \frac{\sum_i w_i \tilde{d}_i}{\sum_i w_i \tilde{b}_i}. \quad (6-9)$$

We then have the following theorem.

Theorem 1: Suppose that the chosen nominal system matrix (A_0, B_0) for the discrete fuzzy model system (6-1) is stabilizable. Then, the closed-loop system driven by the control law (6-7) is asymptotically stable in the large, where G is the stable state-feedback gain for the nominal model and $G_c(x)$ is given by (6-9).

Proof: With the compensation part of the control law in (6-9), the equation $h(x)=0$ holds. Thus the closed-loop system (6-8) becomes

$$x(k+1) = (A_0 + B_0 G)x(k). \quad (6-10)$$

Since (A_0, B_0) is stabilizable, it proves the Theorem. Q.E.D.

From Theorem 1, we can also conclude the following lemmas.

Lemma 1: The behavior of the discrete fuzzy model (6-1) controlled by the control law (6-7) is the same as the closed-loop nominal model (6-10).

The closed-loop system is a linear system (nominal model) whose eigenvalues can be arbitrarily assigned by the state-feedback gain G . Hence we can employ various stability design approaches, such as pole-placement, linear optimal control or other design techniques in designing controllers for the nominal model.

Lemma 2: Consider the fuzzy model system (6-1) without the constant term. Suppose that the eigenvalues assigned for each subsystem are the same as that for the nominal model. Then the state feedback gain of the i th subsystem is

$$F_i = G - \frac{\tilde{a}_i}{\tilde{b}_i}, \quad (6-11)$$

where $i = 1, \dots, r$, and F_i and G are the state feedback gains of the i th subsystem and of the nominal model, respectively.

It is easy to verify this lemma by simply substituting the feedback gain into rules.

In Eq. (6-9), if $\tilde{b}_i$'s are the same sign in all rules that implies $\tilde{b} \neq 0$. In most of existing approaches, $\tilde{b} \neq 0$ is always assumed. When $\tilde{b}_i$ changes sign among rules for some $i \in \{1, \dots, r\}$, $\tilde{b}$ may be near 0 or equal to 0. If such a situation occurs, it means that in a transition instant, the input can hardly change the system states. In other words, the system becomes uncontrollable. We think $\tilde{b}=0$ is only a transition state and usually, the system can still work well without this assumption. In our implementation, a threshold ε is used to avoid large input values for such situations. As in Example 1 shown later, when $\tilde{b}_i$ changes signs among rules, the control input transits from large negative values to large positive values. By using a threshold, the input is limited to a reasonable range and the system still work well. We further analyze such a case in the following. If $\tilde{b}=0$, it means that the inferred system is $x(k+1) = Ax(k)$. It is an unforced system, and no matter what input is, the system behavior is uncontrollable. If the current uncontrollable state is an equilibrium state of $x(k+1) = Ax(k)$, then there is no way of moving out this uncontrollable state. This is rarely the case and it is the problem of the system and is not caused by our control law. If not, the system will transit and then it is easy to find that $\tilde{b} \neq 0$ because the system state is changed.

B. Continuous-time Case

Similarly, we can derive the control law for continuous-time systems. The control law (6-7) is also used for a continuous TSK fuzzy model system (6-6). Since $\bar{A}_0$ is Hurwitz stable, there exist symmetry positive definite matrices P and Q such that $\bar{A}_0^T P + P^T \bar{A}_0 = -Q$. Let $V(x) = x^T(t)Px(t)$, and we have

$$\dot{V}(x) = -x^T(t)Qx(t) + 2h(x)B_0^T Px(t).$$

Obviously, if the inequality $2h(x)B_0^T Px(t) \leq 0$ holds, the system is asymptotically stable. We let $2h(x)B_0^T Px(t) = 0$; i.e., $h(x) = \tilde{a}x(t) + \tilde{b}G_c(x) + \tilde{d} = 0$. Therefore,

$$G_C(x) = -\frac{\tilde{a}x(t) + \tilde{d}}{\tilde{b}} = -\frac{\sum_{i=1}^r k_i \tilde{a}_i}{\sum_{i=1}^r k_i \tilde{b}_i} x(t) - \frac{\sum_{i=1}^r k_i \tilde{d}_i}{\sum_{i=1}^r k_i \tilde{b}_i} . \quad (6-12)$$

Clearly, equation (6-12) is the same as that for the discrete case; i.e., equation (6-9). In other words, *Theorem 1* can also work for continuous fuzzy models. Consequently, *Lemmas 1* and *2* also work for continuous fuzzy models.

Finally, the proposed HCC design method is summarized as follows:

Step 1: Select a stabilizable central rule as the nominal fuzzy model and calculate the variations for all subsystems by equation (6-3).

Step 2: Find a state feedback matrix G such that the nominal fuzzy model in (6-8) is assigned the desired eigenvalues. Such a G can be designed by various techniques to provide specified system behaviors.

Step 3: Get the compensation part $G_C(x)$ by using (6-8) and (6-9).

In this section, various examples are considered to illustrate good design performance of the proposed HCC design scheme.

Example 1: Consider an unstable continuous model, in which there are two rules in the model and the system matrices are

$$A_1 = \begin{bmatrix} 0 & 1 \\ -0.588 & 2.718 \end{bmatrix}, \quad B_1 = \begin{bmatrix} 0 \\ 0.603 \end{bmatrix} \\ A_2 = \begin{bmatrix} 0 & 1 \\ -0.361 & 2.256 \end{bmatrix}, \quad B_2 = \begin{bmatrix} 0 \\ -1.12 \end{bmatrix}.$$

Define the nominal system matrices as $A_0 = \frac{1}{2}(A_1 + A_2) = \begin{bmatrix} 0 & 1 \\ -0.4745 & 2.487 \end{bmatrix}$ and

$B_0 = \frac{1}{2}(B_1 + B_2) = \begin{bmatrix} 0 \\ -0.2611 \end{bmatrix}$. Let the desired eigenvalues are $[-0.5 + j, -0.5 - j]$ and then

$G = [2.9703 \ 13.3558]$. Define $Q = I$, we have $\nu = 23.4183 > 1$ [103] for NCC. Such a design cannot guarantee the stability of the system. Using our HCC approach, the compensation parts are $\tilde{a}_1 = [-9.3957 \ -45.087]$, $\tilde{b}_1 = -2.3096$ and $\tilde{a}_2 = [9.3369 \ 44.8226]$, $\tilde{b}_2 = 4.8298$. Suppose that the same eigenvalues are assigned for all subsystem (DCC). From *Lemma 2*, we obtain $F_1 = [-1.0978 \ -6.1658]$ and $F_2 = [-0.7937 \ -2.9071]$. The solution for LMIs cannot be obtained for this case. In other words, the stability condition is not satisfied, and the controllers for all rules must be redesigned in the PDC approach. Fig. 45 shows the response of the three closed-loop systems for the initial condition $x = [1 \ 0]$. Simulations confirm the above assertions. Now, consider the approach in [105], since the norm bound of uncertainties is bigger than 1, that approach fails in this case. It should be noted that in this example b_i changes signs among rules. We can find that the control input transits from large negative values to large positive values as shown in Fig. 46. By using a threshold, the input is limited to a reasonable range.

Example 2: Consider a nonlinear discrete system in [109]

$$y(k+1) = \frac{y(k)y(k-1)y(k-2)u(k-1)(y(k-2)-1) + u(k)}{1 + y(k-2)^2 + y(k-1)^2}$$

The TSK fuzzy model of such a system is derived in [101] and shown as follows:

R^1 : IF $x_2(k)$ is A1 and $x_1(k)$ is B1

THEN $x_3(k+1) = 0.08 + 0.974x_3(k) + 0.333x_2(k) - 0.279x_1(k) \\ + 0.494u(k) - 0.468u(k-1),$

R^2 : IF $x_2(k)$ is A1 and $x_1(k)$ is B2

$$\text{THEN } x_3(k+1) = 0.021 - 0.03x_3(k) + 0.058x_2(k) - 0.127x_1(k) \\ + 0.419u(k) + 0.029u(k-1),$$

R^3 : IF $x_2(k)$ is A2 and $x_1(k)$ is C1

$$\text{THEN } x_3(k+1) = 0.004 - 0.173x_3(k) + 0.211x_2(k) + 0.009x_1(k) \\ + 0.619u(k) + 0.111u(k-1),$$

R^4 : IF $x_2(k)$ is A2 and $x_1(k)$ is C2

$$\text{THEN } x_3(k+1) = 0.002 - 0.012x_3(k) + 0.005x_2(k) + 0.007x_1(k) \\ + 0.977u(k) + 0.007u(k-1),$$

R^5 : IF $x_2(k)$ is A2 and $x_1(k)$ is C3

$$\text{THEN } x_3(k+1) = 0.003 - 0.099x_3(k) + 0.011x_2(k) - 0.02x_1(k) \\ + 0.587u(k) + 0.058u(k-1),$$

R^6 : IF $x_2(k)$ is A3 and $x_1(k)$ is D1

$$\text{THEN } x_3(k+1) = -0.003 - 0.062x_3(k) + 0.01x_2(k) - 0.192x_1(k) \\ + 0.38u(k) + 0.079u(k-1),$$

R^7 : IF $x_2(k)$ is A3 and $x_1(k)$ is D2

$$\text{THEN } x_3(k+1) = -0.112 + 0.131x_3(k) - 0.001x_2(k) + 0.144x_1(k) \\ + 0.382u(k) - 0.138u(k-1),$$

where $x_3(k) = y(k)$, $x_2(k) = x_3(k-1)$, and $x_1(k) = x_2(k-1)$. Define the state vector $x = [x_1 \ x_2 \ x_3]$ and the membership functions are the same as those used in [101]. It can be found that the rule number is 7, LMI kinds of approaches may have difficulties in solving it. Now, the optimum control design is used. The considered cost function is of the form

$$J = \sum_{k=0}^{\infty} [x^T(k)Qx(k) + u^T(k)Ru(k)]. \text{ Two situations are tested here:}$$

$$(I) \ J_1 : Q = \begin{bmatrix} 10 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } R=1 \text{ and } (II) \ J_2 : Q = \begin{bmatrix} 10 & -10 & 0 \\ -10 & 10 & 0 \\ 0 & 0 & 0 \end{bmatrix} \text{ and } R=1.$$

Design case (I): $A_0 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -0.0675 & 0.166 & 0.4005 \end{bmatrix}$ and $B_0 = \begin{bmatrix} 0 \\ 0 \\ 0.6792 \end{bmatrix}$. It is easy to verify

that (A_0, B_0) is stabilizable. By minimizing J_1 , the required eigenvalues are $[-0.1123 + 0.2661j \ -0.1123 - 0.2661j \ 0.2392]$ and then the state-feedback gain is $G = [0.1288 \ -0.2881 \ -0.5682]$. The compensation parts are

$$\begin{aligned} \tilde{a}_1 &= [-0.3465 \ 0.3244 \ 0.9993], & \tilde{b}_1 &= 0.7273, \\ \tilde{a}_2 &= [0.237 \ -0.0486 \ -0.4162], & \tilde{b}_2 &= 0.6169, \\ \tilde{a}_3 &= [0.1012 \ 0.0918 \ -0.7941], & \tilde{b}_3 &= 0.9114, \\ \tilde{a}_4 &= [0.1662 \ -0.3634 \ -0.8565], & \tilde{b}_4 &= 1.4385, \\ \tilde{a}_5 &= [0.0525 \ -0.1891 \ -0.6583], & \tilde{b}_5 &= 0.8643, \\ \tilde{a}_6 &= [-0.24 \ -0.1028 \ -0.4307], & \tilde{b}_6 &= 0.5595, \\ \tilde{a}_7 &= [0.2551 \ -0.1198 \ -0.1482], & \tilde{b}_7 &= 0.5624. \end{aligned}$$

Design case (II): By minimizing the cost function J_2 , the desired eigenvalues become

$[-0.0059 + 0.1458j \quad -0.0059 - 0.1458j \quad 0.766]$. Thus, the state feedback gain is $G = [0.123 \quad -0.263 \quad 0.521]$ and the compensation parts are $\tilde{a}_1 = [-0.35 \quad 0.32 \quad 0.70]$, $\tilde{a}_2 = [0.24 \quad -0.06 \quad -0.83]$, $\tilde{a}_3 = [0.11 \quad 0.09 \quad -0.90]$, $\tilde{a}_4 = [0.16 \quad -0.35 \quad -0.38]$, $\tilde{a}_5 = [0.05 \quad -0.19 \quad -0.81]$, $\tilde{a}_6 = [-0.24 \quad -0.11 \quad -0.91]$, $\tilde{a}_7 = [0.26 \quad -0.13 \quad -0.62]$, and $\tilde{b}_1, \dots, \tilde{b}_7$ are the same as those in case (I).

The simulation results are shown in Fig. 47 for the initiate state $(1, -1, 1)$. It can be found that the overall system can indeed have good control performance as expected.

Example3 : Consider the problem of balancing an inverted pendulum on a cart. The dynamic equations are [110]

$$\begin{aligned}\dot{x}_1 &= x_2 \\ \dot{x}_2 &= \frac{g \sin(x_1) - a m l x_2^2 \sin(2x_1) / 2 - a \cos(x_1) u}{4l/3 - a m l \cos^2(x_1)}\end{aligned}$$

where x_1 and x_2 are the angle of the pendulum from the vertical and the angle velocity, m is the mass of the pendulum, M is the mass of the cart, $a = 1/(m + M)$, $g = 9.8m/s^2$ is the gravity constant, $2l$ is the length of the pendulum, and u is the force applied to the cart. Similar to [87], $m = 2.0kg$ and $M = 8.0kg$. The fuzzy model can be approximated as [87]:

$$R^1 : \text{ IF } x_1 \text{ is about } 0 \text{ THEN } \dot{x} = A_1 x + B_1 u$$

$$R^2 : \text{ IF } x_1 \text{ is about } \pm \pi/2 (|x_1| < \pi/2) \text{ THEN } \dot{x} = A_2 x + B_2 u$$

where $A_1 = \begin{bmatrix} 0 & 1 \\ \frac{g}{4l/3 - a m l} & 0 \end{bmatrix}$, $B_1 = \begin{bmatrix} 0 \\ a \end{bmatrix}$, $A_2 = \begin{bmatrix} 0 & 1 \\ \frac{2g}{\pi(4l/3 - a m l \beta^2)} & 0 \end{bmatrix}$, $B_2 = \begin{bmatrix} 0 \\ -\frac{a \beta}{4l/3 - a m l \beta^2} \end{bmatrix}$, and $\beta = \cos(88^\circ)$. The design specification is considered as no overshoot with the rising time $2s$.

We select the nominal fuzzy model as $A_0 = \begin{bmatrix} 0 & 1 \\ 13.3271 & 0 \end{bmatrix}$ and $B_0 = \begin{bmatrix} 0 \\ -0.0918 \end{bmatrix}$. It is easy to verify that (A_0, B_0) is stabilizable. For the given specifications, the required two pole locations are $(-2, -2)$. Then, we have $G = [188.8267 \quad 43.5911]$. From (6-8), the compensation parts are:

$$\begin{aligned}\tilde{a}_1 &= [131.0811 \quad 40.2407], & \tilde{b}_1 &= 1.9231, \\ \tilde{a}_2 &= [-134.8203 \quad -41.1039], & \tilde{b}_2 &= 0.0571\end{aligned}$$

The controller is used for the initial conditions $(x_1, x_2) = (80^\circ, 0^\circ/s)$. For the same desired pole locations, the PDC approach in [87] and the approach in [105] are also considered. The simulation results are shown in Fig. 48 and corresponding rise-time are listed in Table 19. It can be found that our approach is the nearest one. Moreover, when the rise-time is 0.75 sec, the required two poles are about $(-5, -5)$. By using our HCC approach, the simulation result and the rise-time are also shown in Fig. 48 and in Table 19, respectively. It is clearly that HCC could achieve such a specification.

Since our approach is to directly compensate the uncertainties, robustness again parameter changes is an important issue. Nevertheless, our approach is derived in a Lyapunov control fashion. The obtained controller is always stable under a certain range of uncertainty. Thus, if a parameter slightly changes, we believe the proposed controller is robust. Fig 49 shows the simulation results of changing m (the mass of the pendulum), M (the mass of the cart) and l (the length of the pendulum), respectively, within $\pm 10\%$, and without re-designing the controllers (two poles are located at $(-2, -2)$). It is evident that the responses of using our controller are only

slightly changed even though the direct compensation is implemented. For the other two approaches, the changes are much less in our simulation. It is because those two approaches do not directly compensate those uncertainties and as a consequence, the effects of parameter changes are small.

In this report, we have proposed a method of designing controllers for affine TSK fuzzy models. In this approach, the controller is separated into two parts. The first one is to move the eigenvalues of the central rule to the desired locations. The second one is to individually compensate the variations residing in all rules. The design procedure is simple and effective. The stability analysis becomes the Lyapunov theorem for linear systems. Such an approach not only can result in the use of various control design techniques but also can guarantee desired performance of the closed-loop system. Various examples are conducted in the report. Some of them even cannot be solved by other existing approaches. All results illustrated good control performances as desired.

Table 19. The rise-times of Zak's, PDC and FCC for Example 3

	Zak's Poles at $(-2 \ -2)$	PDC Poles at $(-2 \ -2)$	FCC Poles at $(-2 \ -2)$	FCC Poles at $(-5 \ -5)$
Rise-time	3 sec	0.75 sec	1.31 sec	0.74 sec

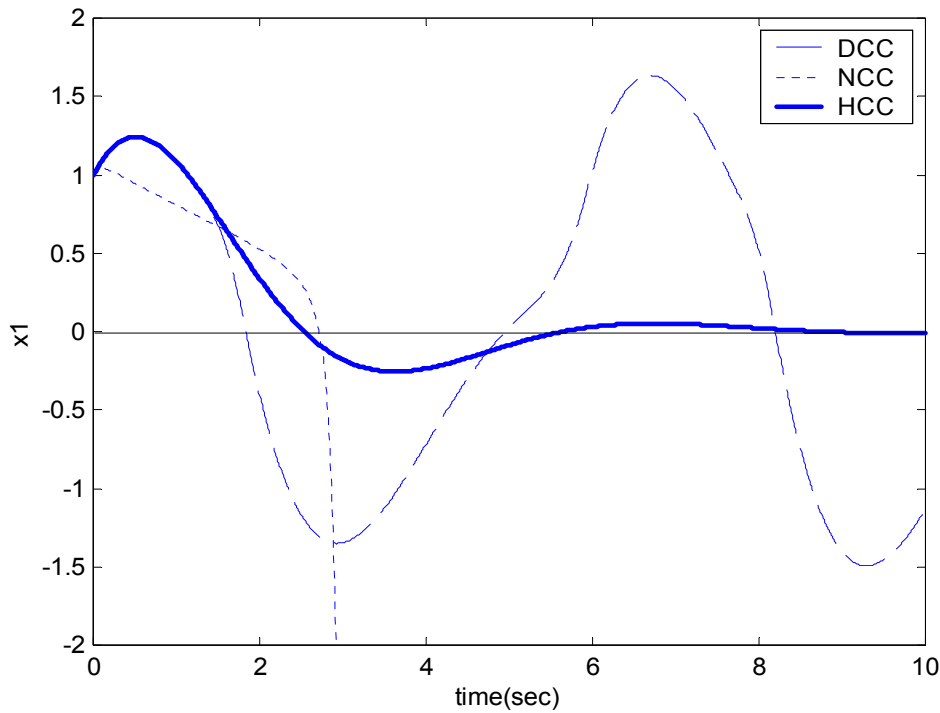


Fig. 45 The responses of using various controllers for the system in Example 1.

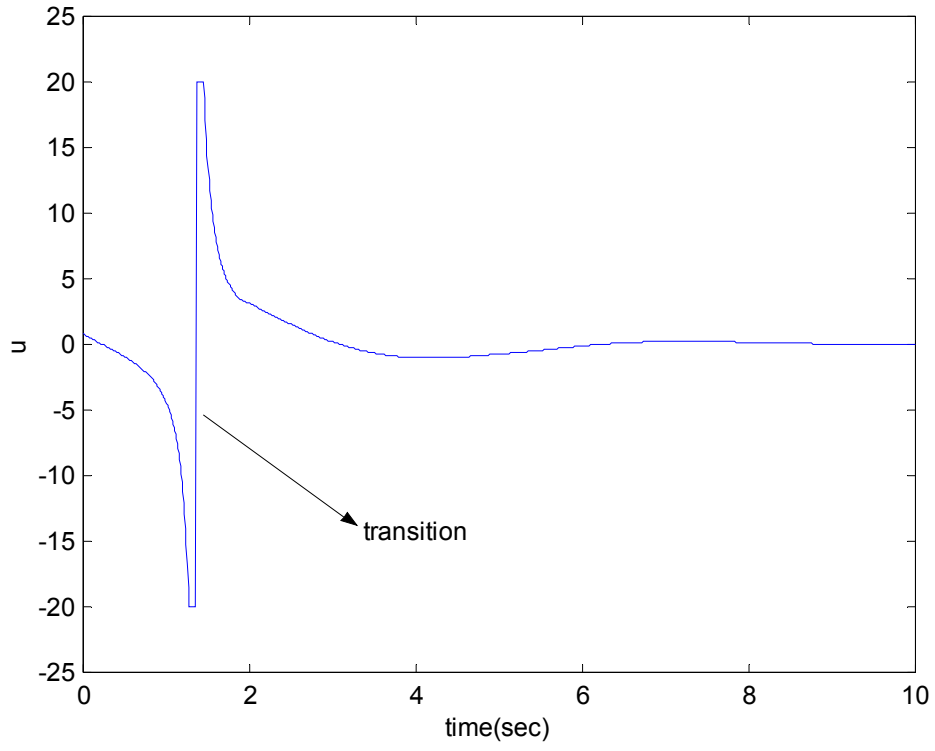


Fig. 46 The control input by using the proposed HCC for the system in Example 1.

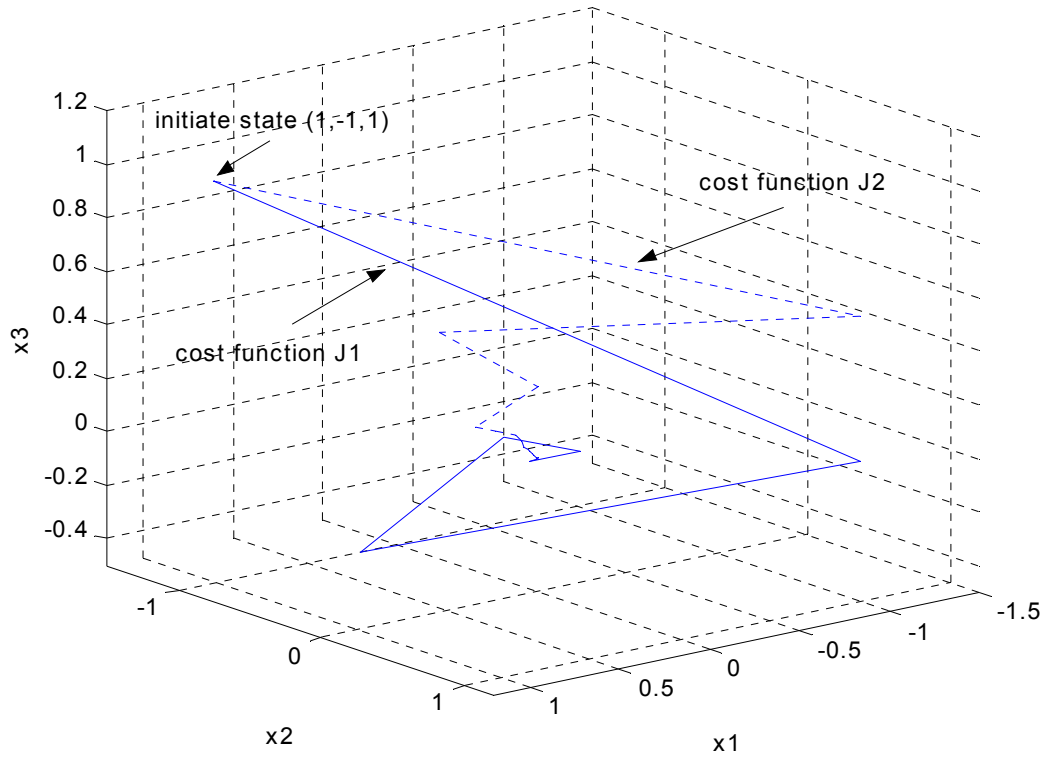


Fig. 47 The state trajectories of the nonlinear discrete system in Example 2.

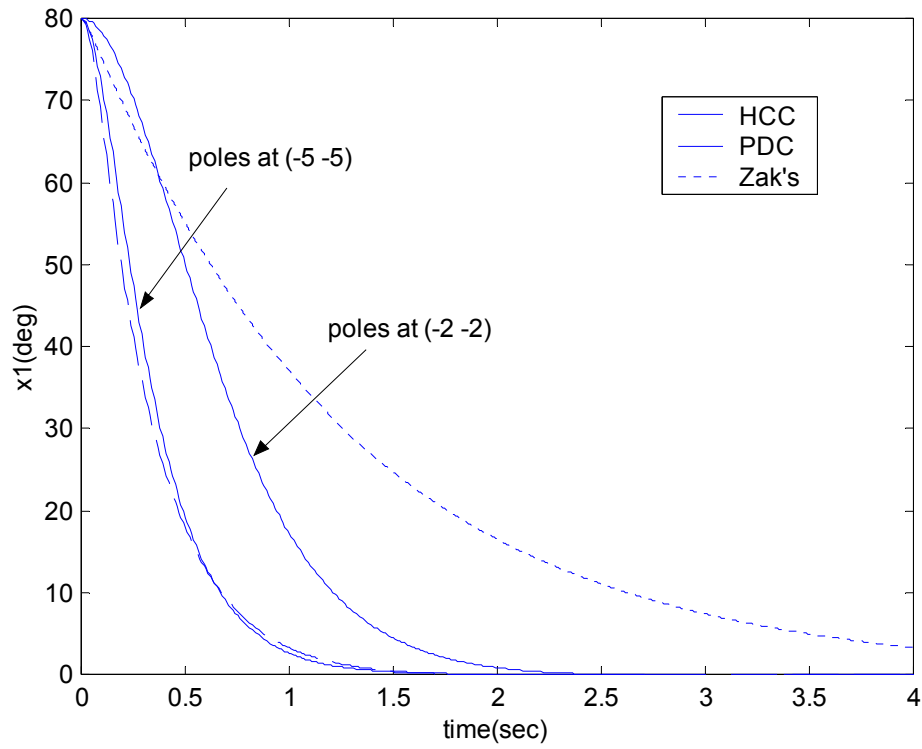


Fig. 48 The angle responses of using various controllers for the system in Example 3.

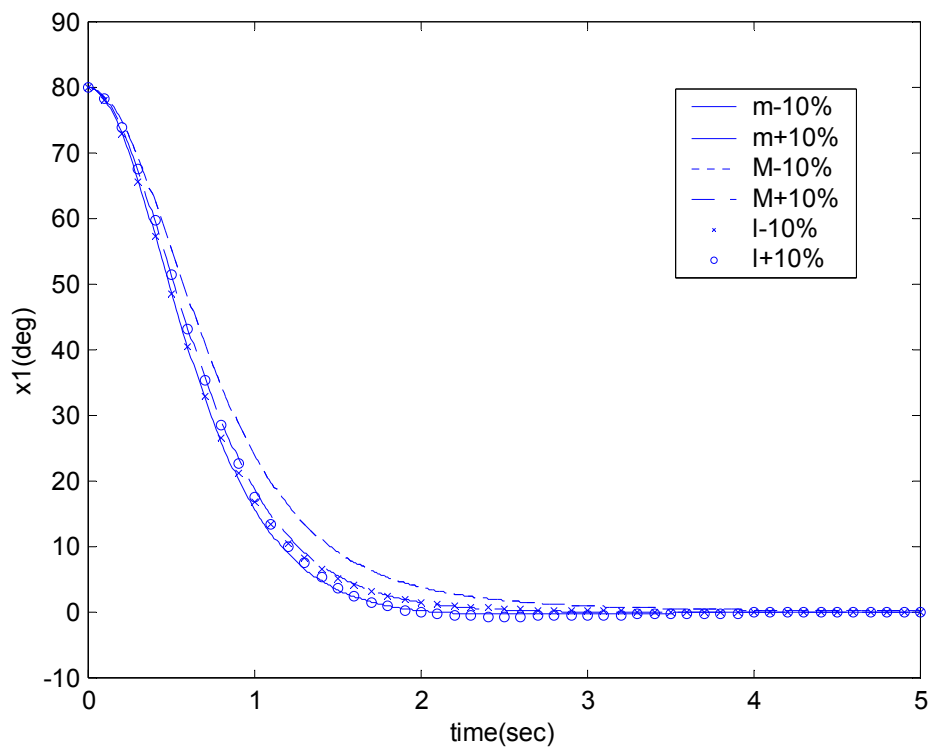


Fig. 49 The angle responses of using the same HCC for the system with various parameter variations in Example 3.

八、即時系統之建構

本計劃所探討的是一實際操作的系統，因此即時性的操作是必要的，為了能有較佳的即時操作，我們提出利用外部 clock 來達到即時的概念，在傳統的 Windows 中由於其

插斷其設計不佳使得其不能有即時的操作，而在較新的 NT 中其已能有好的處理能力，可是由於 HAL 及驅動程式的難撰寫，由其本身系統中去讀取很難做到。而在我們初步的實驗中，利用外部 clock 輔以 Event 方式的插斷處理，在其他計算程式及硬體操作干擾下，其即時精確度仍然可達 $360\mu\text{s}$ ，如下表所示。如是精確度是符合我們需要的。因此我們也納入此一方式來達到在 Windows 下的即時操控的目的。

Table 20

Mode Process	Polling	Event-driven	Error (Event-driven)
No Process	20.08ms	20.08ms	$80\mu\text{s}$
1 Process	25.68ms	20.16ms	$160\mu\text{s}$
2 Processes	25.71ms	20.24ms	$240\mu\text{s}$
12 Processes	37.7ms	20.24ms	$240\mu\text{s}$
12 Processes+存取硬碟	45.6ms	20.36ms	$360\mu\text{s}$
12 Processes+存取硬碟 + IE 上網	47.8ms	20.36ms	$360\mu\text{s}$
12 Processes+存取硬碟 + IE 上網+讀取光碟	無法 量測	20.36ms	$360\mu\text{s}$
12 Processes+存取硬碟 +play game	無法 量測	20.36ms	$360\mu\text{s}$
12 Processes+存取硬碟 +開關 IE	無法 量測	20.36ms	$360\mu\text{s}$

REFERENCES

- [1] C. A. Avizzano, F. Barbagkli, and M. Bergabasco, "Washout filter design for a motorcycle simulator," *Proc. of 2000 IEEE SMC Conf.*, vol. 2, pp. 995-1000, 2000.
- [2] M. Idan and M. A. Nahon, "Offline comparison of classical and robust flight simulator motion control," *Journal of Guidance, Control, and Dynamics*, vol. 22, no. 5, pp. 702-709, 1999.
- [3] W. Wu and M. Cardullo, "Is there an optimum motion cueing algorithm? Pilot and aircraft simulators," *AIAA, Modeling and Simulation Technology Conf.*, 1997.
- [4] S. F. Schmidt and B. Conrad, "Motion drive signals for piloted flight simulators," *NASA CR-1601*, 1970.
- [5] M. Baarspul, "The generation of motion cues on a six-degrees-of-freedom motion system," *Rept. LR-248*, Dept. of Aerospace Eng., Delft Univ. of Technology, The Netherlands, 1977.
- [6] R. V. Parrish, *et al.*, "Coordinated adaptive washout for motion simulators," *Journal of Aircraft*, vol. 12, no. 1, pp. 44-50, 1977.
- [7] R. V. Parrish and D. J. Jr., Martin, "Comparison of a linear and nonlinear washout for motion simulators utilizing objective and subjective data from CTOL transport landing approaches," *NASA TN D-8157*, 1976.
- [8] D. Ariel and R. Sivan, "False cue reduction in moving flight simulators," *IEEE Trans. On Systems, Man, and Cybernetics*, vol. 14, no. 4, pp. 665-671, 1984.

- [9] R. Sivan, J. Ish-Shalom, and J.-K. Huang, "An optimal control approach to the design of moving flight simulators," *IEEE Trans. On Systems, Man, and Cybernetics*, vol. 12, no. 6, pp. 818-827, 1982.
- [10] L. D. Reid and M. A. Nahon, "Flight simulator motion-base drive algorithms: Part 1: Developing and testing the equations," *UTIAS Rept. 296*, Univ. of Toronto, Canada, 1982.
- [11] L. D. Reid and M. A. Nahon, "Flight simulator motion-base drive algorithms: Part 2: Selecting the system parameters," *UTIAS Rept. 307*, Univ. of Toronto, Canada, 1985.
- [12] L. D. Reid and M. A. Nahon, "Flight simulator motion-base drive algorithms: Part 3: Pilot evaluation," *UTIAS Rept. 319*, Univ. of Toronto, Canada, 1986.
- [13] L. D. Reid and M. A. Nahon, "The response of airline pilots to variations in flight simulator motion algorithms," *Journal of Aircraft*, vol. 25, no. 7, pp. 639-646, 1988.
- [14] M. A. Nahon and L. D. Reid, "Simulator motion drive algorithms—A designer's perspective," *Journal of Guidance, Control, and Dynamics*, vol. 13, no. 2, pp. 356-362, 1990.
- [15] M. A. Nahon and L. D. Reid, "Adaptive simulator motion software with supervisory control," *Journal of Guidance, Control, and Dynamics*, vol. 15, no. 2, pp. 376-383, 1992.
- [16] 陳德睦，含未知非線性系統之類神經網路控制，國立台灣科技大學電機工程研究所，2000.
- [17] T. Eltoft and R. J. P. deFigueiredo, "A DCT-based D-FANN nonlinear adaptive time series prediction," *IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing*, vol. 47, no. 10, 2000.
- [18] Shun-Feng Su, Sheng-Hsiung Hsieh, and Chen-Chia Chuang, "On the Study of Embedding Fuzzy Concept in Reinforcement Learning," scheduled to appear in *Journal of the Chinese Institute of Engineers*, vol. 24, no. 4, 2001.
- [19] C. T. Lin and M. C. Kan, "Adaptive fuzzy command acquisition with reinforcement learning," *IEEE Trans. on Fuzzy Systems*, vol. 6, no. 1, pp.102-121, 1998.
- [20] C. T. Lin and Y. C. Lu "A neural fuzzy system with linguistic teaching signals," *IEEE Trans. on Fuzzy Systems*, vol. 3, no. 2, pp. 169-189, 1995.
- [21] Shun-Feng Su and Sheng-Hsiung Hsieh, "Embedding fuzzy mechanisms and knowledge in box-type reinforcement learning controllers" to appear in *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2002.
- [22] C. F. Juang, and C. T. Lin, "An on-line self-constructing neural fuzzy inference network and its applications," *IEEE Trans. on Fuzzy Systems*, vol. 6, no. 1, pp.12-32, 1998.
- [23] D. A. Panagiotopoulos, R. W. Newcomb, and S. K. Singh, "Planning with a functional neural-network architecture," *IEEE Trans. on Neural Networks*, vol. 10, no. 1, pp. 115-127, January 1999.
- [24] D. A. Panagiotopoulos, S. K. Singh, T. R. Darden, and R. W. Newcomb, "Hardware oriented semistate descriptions of functional artificial neural networks," *IEEE Trans. on Neural Networks*, vol. 2, pp. 1197-1198, 1994.
- [25] D. A. Panagiotopoulos, S. K. Singh, and R. W. Newcomb, "VLSI implementation of a functional neural network," *1997 IEEE International Symposium on Circuits and Systems*, Hong Kong, pp. 701-704, June 1997.
- [26] L. V. Zyla and R. J. P. de Figueiredo, "Nonlinear system identification based on a Fock space framework," *SIAM J. Contr. Optimization*, vol. 21, no. 6, pp. 931-939, Nov. 1983.
- [27] C. Citterio, A. Pelagotti, V. Piuri, and L. Rocca, "Function approximation-a fast-convergence neural approach based on spectral analysis," *IEEE Trans. on Neural Network*, vol. 10, no. 4, July 1999.
- [28] G. Georgiou and C. Koutsougeras, "Complex domain backpropagation," *IEEE Trans. Circuits Syst. II*, vol. 39, pp. 330-334, 1992.
- [29] M. S. Kim and C. C. Guest, "Modification of back-propagation for complex valued signal processing in frequency domain," in *Proc. Int. Joint Conf. Neural Networks*, San Diego, pp. 27-31, 1990.

- [30] H. Leung and S. Haykin, "The complex backpropagation algorithm," *IEEE Trans. Signal Processing*, vol. 39, pp. 2101-2104, 1991.
- [31] B. Igel'nik, M. Tabib-Azar, and S. R. LeClair, "A net with complex weights," *IEEE Trans. Neural Networks*, vol. 12, pp. 236-249, Mar. 2001.
- [32] A. V. Oppenheim, R. W. Schaffer, and J. R. Buck, *Discrete-time Signal Process.*, Prentice Hall, 1996.
- [33] T. Takagi, and M. Sugeno, "Fuzzy identification of system and its applications to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol. 15, no. 1, pp. 116-132, 1985.
- [34] H. O. Wang, K. Tanaka, and M. F. Griffin, "An approach to fuzzy control of nonlinear system: stability and design issues," *IEEE Trans. on Fuzzy Systems*, vol. 4, no. 1, pp. 14-23, 1996.
- [35] D. Niemann, J. Li, H. O. Wang, and K. Tanaka, "Parallel distributed compensation for Takagi-Sugeno models: new stability conditions and dynamic feedback designs," *Proceeding of 14th World Congress of IFAC*, pp. 207-212, 1999, Beijing.
- [36] H. O. Wang, K. Tanaka, and M. F. Griffin, "Parallel distributed compensation of nonlinear systems by Takagi-Sugeno fuzzy model," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, pp. 531-538, 1995.
- [37] K. Tanaka, T. Ikeda, and H.O. Wang, "Fuzzy regulators and fuzzy observers: relaxed stability conditions and LMI-based design," *IEEE Trans. on Fuzzy Systems*, vol. 6, no. 2, pp. 250-265, 1998.
- [38] J. Li, H. O. Wang, D. Niemann, and K. Tanaka, "Dynamic parallel distributed compensation for Takagi-Sugeno fuzzy systems: An LMI approach," *Information Sciences*, vol. 123, pp. 201-221, 2000.
- [39] K. Kiriakidis, "Fuzzy model-based control of complex plants," *IEEE Trans. on Fuzzy Systems*, vol. 6, pp. 517-529, Nov. 1998.
- [40] M. C. M. Teixeira, and S. H. Zak, "Stabilizing controller design for uncertain nonlinear systems using fuzzy models," *IEEE Trans. Fuzzy System*, vol. 7, pp.133-142, 1999.
- [41] K. Tanaka, T. Ikeda, and H.O. Wang, "Robust stabilization of a class of uncertain nonlinear systems via fuzzy control: Quadratic stabilizability, H^∞ control, and linear matrix inequalities," *IEEE Trans. Fuzzy System*, vol. 4, no. 1, pp.1-33, 1996.
- [42] H. K. Lam, F. H. F. Leung, and P. K. S. Tam, "Stable and robust fuzzy control for uncertain nonlinear systems," *IEEE Trans. on Systems, Man, and Cybernetics- Part A: Systems and Humans*, vol. 30, no. 6, pp. 825-840, Nov. 2000.
- [43] W. Chang, Y. H. Joo, J. B. Park, and G. Chen, "Robust fuzzy-model-based controller for uncertain systems," in *Proc. IEEE Int. Conf. Fuzzy Syst.*, pp. 486-491, 1999.
- [44] H. J. Lee, J. B. Park, and G. Chen, "Robust fuzzy control of nonlinear systems with parametric uncertainties," *IEEE Trans. Fuzzy Systems*, vol. 9, no. 2, pp. 369-379, 2001.
- [45] G. Feng, and J. Ma, "Quadratic stabilization of uncertain discrete-time fuzzy dynamic systems," *IEEE Trans. on Circuits and Systems-I: Fundamental Theory and Applications*, vol. 48, no. 11, pp.1337-1344, Nov. 2001.
- [46] Y.-Y. Cao, and P. M. Frank, "Analysis and synthesis of nonlinear time-delay systems via fuzzy control approach," *IEEE Trans. Fuzzy Syst.*, vol. 8, no. 2, pp. 200-211, Apr. 2000.
- [47] S. K. Nguang, and P. Shi, "Stabilization of a Class of nonlinear time-delay systems using fuzzy models," in *Proc. of the 39th IEEE Conference on Decision Control*, pp. 4415-4419, Dec. 2000
- [48] Y. Gu, H. O. Wang, K. Tanaka, and L. G. Bushnell, "Fuzzy control of nonlinear time-delay systems: stability and design issues," in *Proc. of the American Control Conference*, pp. 4771-4776, 2001.
- [49] S. H. Esfahani, and I. R. Petersen, "An LMI approach to the output-feedback guaranteed cost control for uncertain time-delay systems," in *Proc. of the 37th IEEE Conference on Decision Control*, pp. 1358-1363, 1998.
- [50] S. I. Niculescu, Carlos E. de Souza, L. Dugard, and J. M. Dion, "Robust exponential stability of uncertain systems with time-varying delays," *IEEE Trans. on Automatic Control*, vol. 43, pp. 743-748, 1998.

- [51] Y.-Y. Cro, Y.-X. Sun, and C. Cheng, "Delay-dependent robust stabilization of uncertain systems with multiple state delays," *IEEE Trans. on Automatic Control*, vol. 43, pp. 1608-1612, 1998.
- [52] X. Li, and Carlos E. de Souza, "Criteria for robust stability and stabilization of uncertain linear systems with state delay," *Automatica*, vol. 33, pp. 1657-1662, 1997.
- [53] T.-J. Su, C.-Y. Lu, and J. S.-H. Tsai, "LMI approach to delay-dependent robust stability for uncertain time-delay system," *IEE Pro. Control Theory Appl.*, vol. 148, pp. 209-212, 2001.
- [54] E. Tissir, and A. Hammed, "Stability tests of interval delay systems," *Systems & Control Letters*, vol. 23, pp. 263-270, 1994.
- [55] Y. Fang, K. A. Loparo, and X. Feng, "Sufficient condition for the stability of interval matrices," *Int. J. Control*, vol. 58, no. 4, pp 969-977, 1993.
- [56] K. Tanaka, and M. Sano, "A robust stabilization problem of fuzzy control systems and its application to backing up control of a Truck-Trailer," *IEEE Trans. Fuzzy System*, vol. 2, no. 2, pp. 119-134, May 1994.
- [57] P. Gahinet, A. Nemirovskii, A. J. laub, and M. Chilali, *LMI Control Toolbox*, The Math Works, Inc., 1994.
- [58] P. Gahinet, and P. Apkarian, "A linear matrix inequality approach to H^∞ control," *Int. J. Robust Nonlinear Contr.*, vol. 4, pp. 412-448, 1994.
- [59] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*, SIAM, Philadelphia, 1994.
- [60] K. Tanaka and M. Sugeno, "Stability analysis and design of fuzzy control system," *Fuzzy Sets Syst.*, vol. 45, pp. 135-156, 1992.
- [61] P. Lancaster, *Theory of Matrices*. Academic Press, New York and London, 1969.
- [62] Z. X. Han, G. Feng, B. L. Walcott, and J. Ma, "Dynamic output feedback controller design for fuzzy systems," *IEEE Trans. Syst., Man, Cybern. Part B*, vol. 30, no. 1, pp. 204-210, 2000.
- [63] Kiriakos Kiriakidis, "Dynamic output feedback control of gas furnaces via fuzzy modeling," *Proceedings of the 1997 IEEE International Conference on Control Applications*, pp. 79-84, 1997.
- [64] K. Tanata, and T. Hori, "Dynamic output feedback design for nonlinear systems," *IEEE International Fuzzy Systems Conference Proceedings*, pp. III-56-61, 1999.
- [65] I. P. Pavol, *Conditioned Reflexes*, London: Oxford Univ. Press, 1927.
- [66] C. T. Lin and C.S.G. Lee, "Reinforcement Structure/Parameter Learning for Neural-Network-Based Fuzzy Logic Control Systems," *IEEE Trans. On Fuzzy Systems*, vol. 2, no. 1, 1994.
- [67] K. C. Cheok and N. K. Loh, "A ball-balancing demonstration of Optimal and Disturbance-Accommodating Control," *IEEE Contr. Syst. Mag.*, pp. 54-57, 1987.
- [68] B. Widrow, "The original adaptive Neural Net Broom-Balancer," *Proc. Int. Symp. Circ. and Syst.*, pp. 351-357, 1987.
- [69] D. Michie and R. A. Chambers, "BOXES: An experiment in Adaptive Control," in *Mach. Intell.*, Dale and Michie, Eds. Edinburgh, Scotland: Oliver and Boyd, vol. 2: 137-152, 1968.
- [70] C. W. Anderson, "Strategy Learning with Multilayer Connectionist Representations," *Proc. Fourth Int. Workshop on Mach. Learn.*, Irvine, CA, pp. 103-114, 1987.
- [71] M. Alata, C. Y. Su, and K. Demirli, "Adaptive control of a class nonlinear systems with a first-order parameterized Sugeno fuzzy approximator," *IEEE Tran. Syst., Man, Cybern. C*, vol. 31, no.3, pp. 410-419, 2001.
- [72] C. Shi and A. D. Cheok, "Performance comparison of fused soft control/hard observer type controller with hard control/hard observer type controller for switched reluctance motors," *IEEE Tran. Syst., Man, Cybern. C*, vol. 32, no.2, pp. 99-112, 2002.
- [73] Z. Huaguang and L. Cai, "Nonlinear adaptive control using the fourier integral and its application to CSTR systems," *IEEE Tran. Syst., Man, Cybern. B*, vol. 32, no.3, pp. 367-372, 2002.
- [74] Y. Tang and D. Vélez-Díaz, "Robust fuzzy control of mechanical systems," *IEEE Tran. Fuzzy Syst.*, vol. 11, no.3, pp. 411-418, 2003.

- [75] Y. G. Leu, T. T. Lee, and W. Y. Wang, "On-line tuning of fuzzy-neural network for adaptive control of nonlinear dynamical systems," *IEEE Tran. Syst., Man, Cybern. B*, vol. 27, no.6, pp. 1034-1043, 1997.
- [76] Y. G. Leu, T. T. Lee, and W. Y. Wang, "Observer-based adaptive fuzzy-neural control for unknown nonlinear dynamical systems," *IEEE Tran. Syst., Man, Cybern. B*, vol. 29, no.5, pp. 583-591, 1999.
- [77] W. Y. Wang, M. L. Chan, T. T. Lee, and C. H. Liu, "Adaptive fuzzy control for strick-feedback canonical nonlinear systems with H_∞ tracking performance," *IEEE Tran. Syst., Man, Cybern. B*, vol. 30, no.6, pp. 878-885, 2000.
- [78] W. Y. Wang, M. L. Chan, C. C. J. Hsu, and T. T. Lee, " H_∞ tracking-based sliding mode control for uncertain nonlinear systems via an adaptive fuzzy-neural approach," *IEEE Tran. Syst., Man, Cybern. B*, vol. 32, no.4, pp. 483-492, 2002.
- [79] Song-Shyong Chen, Shun-Feng Su, and Tsu-Tian Lee, "Direct Adaptive Model Reference Fuzzy Controllers," *2002 Chinese Automatic Control Conference*, 2002.
- [80] C. H. Wang, T. C. Lin, T. T. Lee, and H. L. Liu, "Adaptive hybrid intelligent control for uncertain nonlinear dynamical systems," *IEEE Tran. Syst., Man, Cybern. B*, vol. 32, no.5, pp. 583-597, 2002.
- [81] Song-Shyong Chen, Shun-Feng Su, and Yuan-Chang Chang, "The Stable Tracking Adaptive Fuzzy Control of Nonlinear Dynamic Systems Using the Takagi-Sugeno Fuzzy Logic," *Proceedings of the 7th Conf. On Artificial Intelligence and Applications*, pp. 65-70, 2002.
- [82] M. J. Er and Y. Gao, "Robust adaptive control of robot manipulators using generalized fuzzy neural networks," *IEEE Tran. Indt. Electron.*, vol. 50, no.3, pp. 620-628, 2003.
- [83] C. W. Tao, M. L. Chan, and T. T. Lee, "Adaptive fuzzy sliding mode controller for linear systems with mismatched time-varying uncertainties," *IEEE Tran. Syst., Man, Cybern. B*, vol. 33, no.2, pp. 283-294, 2003.
- [84] K. Tanaka and M. Sugeno, "Stability analysis and design of fuzzy control system," *Fuzzy Sets Syst.*, vol. 45, pp. 135-156, 1992.
- [85] T. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control," *IEEE Trans. Syst., Man, Cybern.*, vol.15, no.1, pp.116-132, 1985.
- [86] M. A. L. Thathachar and P. Viswanath, "On the stability of fuzzy systems," *IEEE Tran. Fuzzy Syst.*, vol.5, no. 1, pp. 145-151, 1997.
- [87] H. O. Wang, K. Tanaka, and M. F. Griffin, "An approach to fuzzy control of nonlinear system: Stability and design issues," *IEEE Trans. Fuzzy Syst.*, vol. 4, pp. 14-23, 1996.
- [88] E. Kim, H. J. Kang, and M. Park, "Numerical stability analysis of fuzzy control systems via quadratic programming and linear matrix inequalities," *IEEE Tran. Syst., Man, Cybern. A*, vol. 29, pp. 333-346, 1999.
- [89] K. Tanaka and M. Sano, "A robust stabilization problem of fuzzy control systems and its application to backing up control of a truck trailer," *IEEE Trans. Fuzzy Syst.*, vol. 2, pp. 119-134, 1994.
- [90] K. Tanaka, T. Ikeda, and H. O. Wang, "Fuzzy regulators and fuzzy observers: relaxed stability conditions and LMI-based designs," *IEEE Tran. Fuzzy Syst.*, vol.6, no. 2, pp. 250-265, 1998.
- [91] M. Chadli, D. Maquin, and J. Ragot, "Relaxed stability conditions for Takagi-Sugeno fuzzy systems," *IEEE Int. Conf. Syst., Man, Cybern.*, vol. 5, pp. 3514-3519, 2000.
- [92] M. C. M. Teixeira and S. H. Zak, "Stabilizing controller design for uncertain nonlinear systems using fuzzy models," *IEEE Trans. Fuzzy Syst.*, vol. 7, pp. 133-142, 1999.
- [93] K. Tanaka, T. Ikeda, and H. O. Wang, "Robust stabilization of a class of uncertain nonlinear systems via fuzzy control: Quadratic stabilizability, H^∞ control, and linear matrix inequalities," *IEEE Tran. Fuzzy Syst.*, vol.4, no. 1, pp. 1-13, 1996.
- [94] H. K. Lam, F. H. F. Leung, and P. K. S. Tam, "Stability and robustness analysis and gain design for fuzzy control systems subject to parameter uncertainties," *The Ninth IEEE Inter. Conf. on Fuzzy Syst.*, vol.2, pp. 682-687, 2000.
- [95] H. K. Lam, F. H. Leung, and P. K. S. Tam, "Design and stability analysis of fuzzy model-based nonlinear controller for nonlinear systems using genetic algorithm," *IEEE Tran.*

- Syst., Man, Cybern. B*, vol. 33, no.2, pp. 250-257, 2003.
- [96] W. Gao and J. Xie, "Design of stable fuzzy system based on LMI," in *Proc. the 3rd World Cong. Intelligent Control and Automation*, vol.3, pp. 1552–1555, 2000.
 - [97] J. Joh, R. Langari, E. T. Jeung, and W. J. Chung, "A new design method for continuous Takagi-Sugeno fuzzy controller with pole placement constraints: An LMI approach," *IEEE Int. Conf. Syst., Man, Cybern.*, vol. 3, pp. 2969–2974, 1997.
 - [98] P. Korba, R. Babuška, H. B. Verbruggen, and P. M. Frank, "Fuzzy gain scheduling: controller and observer design based on Lyapunov method and convex optimization," *IEEE Tran. Fuzzy Syst.*, vol. 11, no.3, pp. 285-298, 2003.
 - [99] S. G. Cao, N. W. Rees, and G. Feng, "Analysis and design of fuzzy control systems using dynamic fuzzy-state space models," *IEEE Trans. Fuzzy Syst.*, vol. 7, no. 2, pp. 192-200, 1999.
 - [100] E. Kim and D. Kim, "Stability analysis and synthesis for an affine fuzzy system via LMI and ILMI: discrete case," *IEEE Tran. Syst., Man, Cybern. B*, vol. 31, pp. 132-140, 2001.
 - [101] G. Kang, and W. Lee, "Design of TSK fuzzy controller based on TSK fuzzy model using pole placement," in *Proc. IEEE Int. Conf. Fuzzy Systems*, vol. 1, pp. 246–251, 1998.
 - [102] G. Kang, *Fuzzy Modeling of Systems and Design of Fuzzy Controller*, Ph.D. Thesis, Tokyo Institute of Technology, 1987.
 - [103] K. Kiriakidis, A. Grivas, and A. Tzes, "Quadratic stability analysis of the Takagi-Sugeno fuzzy model," *Fuzzy Sets Syst.*, vol. 98, pp. 1-14, 1998.
 - [104] G. Feng, "Approaches to quadratic stabilization of uncertain fuzzy dynamic systems," *IEEE Trans. Circuit and Syst. I: Fundamental and applications*, vol. 48, no. 6, pp.760-769, 2001.
 - [105] S. H. Zak, "Stabilizing fuzzy system models using linear controller," *IEEE Trans. Fuzzy Syst.*, vol. 7, pp. 236-240, 1999.
 - [106] H. O. Wang, K. Tanaka, and M. Griffin, "Parallel distributed compensation of nonlinear systems by Takagi and Sugeno's fuzzy model," in *Proc. FUZZ-IEEE'95*, pp. 531-538, 1995.
 - [107] M. S. Mahmoud and A. A. Bahnasawi, "Asymptotic stability for a class of linear discrete systems with bounded uncertainties," *IEEE Trans. Automat. Contr.*, vol.33, no.6, pp.572-575, 1988.
 - [108] W. Yang and M. Tomizua, "Discrete time robust control via state feedback for single input systems," *IEEE Trans. Automat. Contr.*, vol.35, pp. 590-598, 1990.
 - [109] K. S. Narendra, and K. Parthasarathy, "Identification and control dynamical systems using neural networks," *IEEE Tran. On Neural networks*, vol. 1, pp. 4-27, 1990.
 - [110] R. H. Cannon, *Dynamics of Physical Systems*. New York: McGraw-Hill, 1967.
 - [111] K. Tanaka and H. O. Wang, *Fuzzy Control Systems Design and Analysis : A LMI Approach*. John Wiley & Sons , 2001.