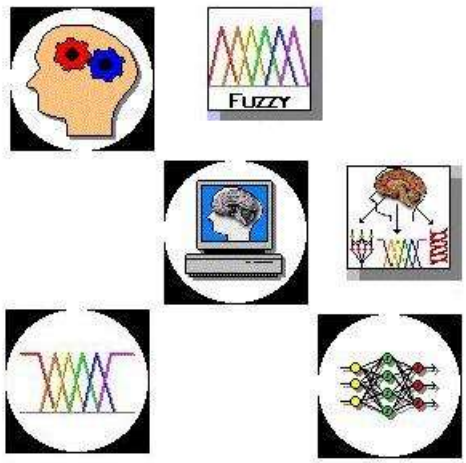
퍼지 제어 툴박스

Fuzzy Control Toolbox



사용자 지침서

목 차

Fuzzy System Creation and Setting	
fuzzy	Invoke fuzzy system editor
newfcs	create a new fuzzy system
loadfcs	load a fuzzy system from disk
savefcs	save the fuzzy system to disk
setfcs	set the fuzzy system properties
addvar	add a new fuzzy variable
rmvar	remove the fuzzy variable
addmf	add a new membership function
rmmf	remove the membership function
editmf	edit the properties of membership function
addrule	add fuzzy rules
rmrule	remove rules

Evaluation Functions	
evalfcs	evaluate fuzzy system output given input
evalfv	evaluate multiple membership functions
evalmf	evaluate the membership function

Membership Functions	
dsigmf	difference of two sigmoid functions
gauss2mf	two-sided Gaussian curve
gaussmf	Gaussian membership function
gbellmf	generalized bell-shaped curve
pimf	PI-shaped curve
psigmf	product of two sigmoid function
smf	S-shaped curve
sigmf	Sigmoid curve
trapmf	trapzoidal membership function
trimf	triangular membership function
zmf	Z-shaped curve

Neuro Fuzzy Functions	
anfis	training routine for Sugeno-type fcs
falcon	training routine for Mamdani-type fcs

Miscellaneous Functions	
showvar	display input/output variables
showmf	display membership functions
showrule	display rules
showfcs	display fuzzy system properties
nvar	return the #s of inputs and outputs
nmf	return the # of membership functions
nrule	return the # of fuzzy rules

Preface

이 매뉴얼은 연구실에서 혹은 현장에서 퍼지 제어에 관련하여 공부하고자 하는 분들이 셈툴 퍼지 제어 툴박스(Fuzzy Control Toolbox)를 보다 잘 활용하도록 하기 위하여 쓰여졌습니다.

퍼지는 개념적으로 그다지 어렵지 않고, 제어기를 쉽게 디자인 할 수 있으며 , 일반적인 현대 제어 기법을 사용한 제어기들과는 달리 제어기 자체에 강인성(Robustness)를 가지고 있습니다. 따라서 퍼지는 처음 제어를 하는 사람들도 비교적 쉽게 접근할 수 있는 분야입니다. 이 책의 주목적이 퍼지 제어 툴박스의 활용에 대한 설명이지만, 퍼지 제어를 처음 접하는 사람들을 위하여 퍼지 제어 이론에 대한 Tutorial을 포함하고 있습니다.

이 책은 크게 세 부분으로 이루어져 있습니다. 첫 번째 부분은 튜토리얼이고, 두 번째 부분은 레퍼런스, 세 번째 부분은 부록입니다.

튜토리얼에서는 툴박스의 간단한 소개와, 퍼지 이론에 대한 역사 및 개념, 여러 가지 제어기 설계기법을 설명하였고, 마지막으로 Inverted Pendulum에 대한 제어기작성의 예를 들어 이해를 높이도록 했습니다.

레퍼런스에서는 퍼지 제어 툴박스에서 제공하는 여러 가지 함수들에 대한 자세한 설명과 GUI 퍼지 시스템 에디터에 대한 설명을 담았습니다.

부록에서는 저희 연구실에서 발표한 강인 퍼지 제어 시스템에 대한 이론을 담은 논문 2편과 퍼지 제어 툴박스를 만들면서 참고한 참고 서적과 참고 논문 리스트, 그리고 각 함수에 대한 인덱스를 담았습니다.

셈툴 프로그램 및 퍼지 툴박스는 인터넷(http://www.cemtool.co.kr)에서 무료로 다운로드 받을 수 있습니다. 사용자들을 위한 게시판도 마련되어 있으므로 퍼지 툴박스에 대한 것뿐만 아니라 제어 전반에 대한 토론의 장으로 활용했으면 바램입니다. 툴박스 관련 의문사항은 게시판을 이용하거나 E-mail(happyboy@postech.ac.kr)을 이용하여 질문해 주십시오.

끝으로, 툴박스 및 매뉴얼 작성에 격려와 도움을 주셨던 여러 분들께 감사의 말씀을 드립니다.

1. 퍼지에 대한 소개

퍼지 이론은 그 소속이 불확실하거나 불분명한 원소들을 뭉뚱그려 하나의 양으로 표현하는 퍼지 집합과 퍼지 논리에 관한 이론을 그 뿌리로 한다. 퍼지 논리는 정확성의 상대적인 중요도를 나타낸다. 상대적인 중요도란 보다 현실적이라는 말이다. 가령 100Kg이라는 수치가 있으면 퍼지 논리는 그것이 몸무게를 나타낸 것이라면 '*무겁다*'라는 말로 뭉뚱그려 표현할 수 있다.

퍼지 이론의 응용은 많은 분야에서 여러 종류의 레벨로 이루어지고 있지만 그 중심에 있는 것은 제어에의 응용이다. 퍼지 제어는 지능제어(Intelligent Control)의 한 갈래로 볼 수 있으며, 플랜트에 대해서 잘 알지 못하는 상황에서 가장 만족할 만한 성능을 보이는 제어 기법중의 하나이다.

퍼지 제어 툴박스라는 이름이 의미하듯이 이 툴박스는 퍼지 제어를 셈툴(CEMTOOL)에서 보다 쉽게 사용할 수 있도록 도움을 주기 위한 도구상자이다. 그리고 이 작은 책자는 퍼지 툴박스를 보다 쉽고 알차게 사용할 수 있도록 만들어진 지침서이다.

1. 퍼지 논리란 무엇인가?

퍼지 제어란 퍼지 논리를 이용한 제어이다. 그리고 퍼지 논리란 입력 공간을 출력 공간으로 전달하는 편리한 방법 중의 하나이다. 이러한 개념은 퍼지 논리의 설명의 시작점이고 가장 강조할 만한 것은 **편리함**이라는 개념이다.

여기서 의미하는 '입력 공간의 출력 공간으로의 전달'이라는 개념은 어떤 의미일까? 몇 가지 예를 들어보자. 만약 당신이 빠른 속도를 원한다면 자동차의 가속페달을 더 세게 밟을 수 있다. 당신이 더 따뜻한 물을 원한다면 적당한 방향으로 밸브를 조정할 수 있다. 사진을 찍을 때 적당한 초점거리를 원한다면 렌즈를 적당히 조정할 수 있다. 이러한 것들은 모두 입력에 대한 적당한 정도의 출력으로의 전달 문제로 볼 수 있다. 이제 입력과 출력 사이에 기러한 일들을 수행하도록 하기 위한 블랙박스의 개념을 생각하자. 이 블랙박스는 입력에 대해서 적당한 출력을 만들어 내는 것이라면 어떤 것이라도 좋다. 이 블랙박스의 내부에는 퍼지 시스템, 선형 시스템, 신경망 회로, 미분 방정식, Lookup table등등 여러 가지 가능한 방법이 있을 수 있다.

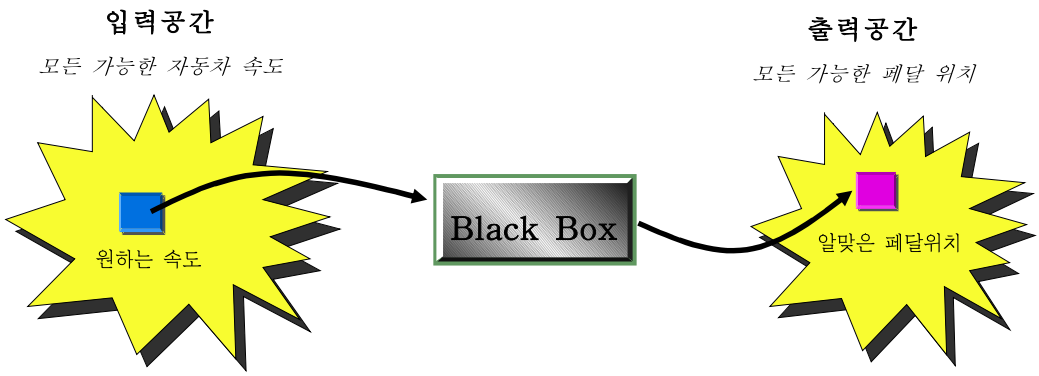


그림 1 원하는 속도에 대한 알맞은 페달 위치

우리는 이러한 일들을 수행하도록 여러 가지 방법들을 블랙박스 내부에 대입할 수 있다. 그리고 결국엔 퍼지 시스템이 가장 좋은 방법임을 알아낼 것이다. 왜 퍼지 시스템일까? 퍼지 논리의 아버지로 간주되는 L. Zadeh교수의 “퍼지 논리를 사용하지 않고도 같은 역할을 하는 제품을 모든 경우에 만들 수 있다. 그러나, 퍼지를 사용하는 것이 가장 빠르고 가장 이익이다.” 라는 말로써 그 대답을 대신한다.

2. 왜 퍼지 논리를 사용하는가?

퍼지 논리의 일반적인 특성을 살펴보면 다음과 같다.

1. 퍼지논리는 개념적으로 이해하기가 쉽다.

- 퍼지 추론 개념의 수학적인 배경은 매우 쉽다. 퍼지 논리를 쉽게 만드는 것은 퍼지 접근 방식의 자연스러움으로 인한 단순함 때문이다.

2. 퍼지 논리는 유연하다.

- 어떤 임의의 주어진 시스템에 대해서 초기 문제 외에도 다른 문제로의 확장성이 좋다.

3. 퍼지 논리는 부정확한 정보에 대하여도 허용 범위가 크다.

- 실제로 모든 일들은 주의 깊은 관찰을 하더라도 정확도를 보장할 수 없다. 퍼지 추론은 이러한 개념을 프로세서 과정에 적용한다.

4. 퍼지 논리는 임의의 복잡한 비선형 시스템을 모델링 할 수 있다.

- 임의의 입출력 정보의 집합을 이용하여 그에 대응되는 퍼지 시스템을 만들 수 있다. 이러한 과정은 특히 Adaptive Fuzzy 알고리즘을 이용하여 쉽게 만들 수 있다.

5. 퍼지 논리는 전문가의 경험의 최상위 레벨로서 만들어진다.

- 신경망과 직접적으로 비교해 보면, 신경망은 전문가의 정보를 이용하여 불투명하고 불명확한 모델을 제시하지만, 퍼지 논리는 대상 시스템을 완전히 이해한 전문가의 완전한 정보를 제공한다.

6. 퍼지 논리는 기존의 제어 기법들과 혼합하여 사용할 수 있다.

- 퍼지 시스템은 기존 제어 기법들을 대체한 방법으로 접근을 필요로 하지는 않는다. 많은 경우에 있어서 퍼지 시스템은 기존 제어 기법들과의 혼합으로 구성을 간략화 한다.

7. 퍼지 논리는 일반적인 일상 언어에 기본을 둔다.

- 퍼지 논리의 기본 바탕은 일상 의사소통의 기본 바탕과 동일하다. 또한 이러한 관찰은 퍼지 논리와 관련된 많은 표현의 기본이 됨을 알 수 있다.

마지막 일곱 번째 특성은 아마도 가장 중요한 특성 중의 하나이고 앞으로 좀 더 많은 토의를 필요로 한다. 일상 언어, 즉 일반인들에 의해 매일 사용되고 있는 언어는 인간 역사의 수 천년을 거치면서 편리함과 효율성을 바탕으로 구성되어졌다. 일상 언어로 쓰여진 문장은 효율적인 의사소통 측면에서도 하나의 성공을 나타낸다. 그럼에도 불구하고 우리는 우리가 일상 생활에서 매일 사용하는 것들과 마찬가지로 이러한 일상 언어의 우수함을 인식하지 못하고 있다. 그러나 퍼지 논리는 이러한 상위 레벨 언어 구조로써 구성되었기 때문에 우리가 퍼지 논리를 사용하기 쉬울 뿐만 아니라 일상 언어가 가지는 장점 또한 모두 포함하고 있다. 즉 다시 말해서, 거의 한 세대 가까이 사용되어진 일상 언어가 퍼지 논리에서 사용하는 도구가 된다.

3. 언제 퍼지 논리를 사용할 수 없는가?

물론 퍼지 논리가 만병 통치약은 아니다. 그러면 언제 퍼지 논리를 사용할 수 없는가? 퍼지 논리의 가장 안전한 표현 방법 중 하나는 퍼지 논리가 입력 공간과 출력 공간 사이의 전달 매개체 역할을 하는 편리한 방법 중 하나라는 것이다. 만약 여러분이 퍼지 논리를 사용하는 데 편리함을 느끼지 못한다면 퍼지 논리 대신 다른 방법으로 시도하기 바란다. 물론 단순한 해가 이미 존재한다면 퍼지 논리를 사용할 필요가 없다. 예를 들어 많은 제어기들은 퍼지 논리를 사용하지 않고도 주어진 역할을 충분히 수행한다. 그러나 만약 여러분들이 퍼지 논리를 익히는 데 얼마간의 시간을 투자한다면, 여러분들은 퍼지 논리가 부정확성과 비선형성을 포함한 대상을 취급하는 데 빠르고 효과적인 강력한 도구가 됨을 알 수 있을 것이다.

4. 퍼지 제어 툴박스는 어떤 일을 하는가?

퍼지 제어 툴박스는 여러분들이 여러 가지 일을 수행하도록 한다. 그러나 그 중에서도 가장 중요한 역할은 퍼지 추론 시스템을 만들고 편집할 수 있도록 하는 기능이다. 여러분은 그래픽화된 도구를 이용하여 손으로 직접 이러한 시스템을 만들거나, Adaptive Fuzzy기법을 이용하여 자동으로 구성할 수 있다.

만약 여러분이 셈툴의 시뮬레이션 도구인 SIMTOOL을 사용한다면, 모듈화된 시뮬레이션 구조에서 쉽게 여러분이 만든 퍼지 제어기를 검증할 수 있으며, *.cem의 확장자를 가지는 매크로 파일 안에서도 여러분이 디자인한 퍼지제어기를 쉽게 삽입하여 사용할 수 있다.

2. 퍼지 이론

언제부터인가 우리 주변에서 쉽게 접하게 되어 버린 퍼지라는 단어. 퍼지 세탁기, 퍼지 냉장고, 퍼지 다리미.... 그리고 퍼지 제어기. 무엇이든지 퍼지라는 단어만 붙이면 '퍼지'하게 되어 버리는 그 퍼지의 실체에 대해서 살펴보기로 하자.

1. 퍼지집합 이론

이제까지 다루어 온 집합(Crisp Set)의 개념은 어느 원소(Element)가 그 집합에 속하는지 속하지 않는지 그 진위가 분명하였다. 그러나 우리가 일상에서 접하는 많은 경우에 있어서 그 집합의 구성 요소를 결정하는 기준이란 명확하지 않다. 가령 목욕탕 물의 온도에 대해서 논한다고 생각해 보자. 우리가 '뜨겁다'라고 말하는 물의 온도의 집합은 어떻게 될까. 계란을 넣어 익을 정도의 펄펄 끓는 물에 대해서는 아마도 모든 사람들이 뜨겁다는 사실을 공감할 것이다. 그런데 물의 온도가 낮아져서 45도가 되었다면? 뜨겁다고 해야할 지 고민이 되기 시작할 것이다. 그렇지만 억지를 부려서 45까지는 뜨거운 물이라고 생각하자. 그렇다면 44.9도에 대해서는? 그보다 조금 더 낮은 44.8도는? 이 정도까지는 뜨겁다는 사실에 그래도 확신이 조금 있겠지만 온도가 더 낮아져서 40도가 되어 버렸을 때, 이 물을 아직도 뜨겁다고 하기엔 마음 한구석이 허전할 것이다. 그렇다고 차다고 할 수도 없고....

이런 경우에 어떤 온도의 물이 뜨거운 지 아닌지에 대한 정도를 표현하는 수가 있다면 어떨까. 45도의 물은 뜨거움의 정도가 1, 40도는 0.5 35도는 0과 같은 식으로 말이다. 확실히 자연스러움을 느낄 것이다. 어떤 온도의 물에 대해서도 그것이 뜨거운 지 아닌지에 대해서 말할 자신이 생기지 않았는가? :)

퍼지 집합이란 위의 경우처럼 어떤 원소가 그 집합에 속한 정도까지 나타낸 집합을 말한다.

'속한 정도까지 나타낸 집합'이란 말이 조금 생소하게 들릴지 모르지만 사실 그다지 생소한 것은 아니다. 일반적으로 우리는 어떤 집합을 표현하기 위해서 세 가지 방법을 쓴다. 첫 번째 방법은 '원소나열법'으로 알려진 방법으로 어떤 집합에 소속된 원소를 일일이 나열하는 방법이다. 가령 '요일' = {월요일, 화요일, 수요일, 목요일, 금요일, 토요일, 일요일}과 같이 나타내는 것이 그렇다. 두 번째 방법은 '조건제시법'으로 그 집합의 성질을 서술하는 방법이다. 가령 유리수의 집합을 다음과 같이 정의할 수 있다. '유리수' = $\{x | x = p/q, \text{단} q, q \neq 0 \text{정수}\}$.

하지만 이와 같은 방법으로 퍼지 집합을 표현하기엔 알맞지 않다. 그래서 '소속함수(Membership Function)표시법'을 사용한다. 소속함수란 어떤 원소가 집합에 속한 정도를 나타내어 주는 함수를 뜻한다. 어떤 집합 A의 소속함수는 ($m_A : X \rightarrow [0,1]$)와 같은 형태를 가지는 어떤 함수라도 좋다. 우리가 배워온 보통의 집합도 물론 소속함수 표시법으로 표현될 수 있다. 이 경우는 아주 특별한 경우로 소속함수가 취하는 값은 0(그 집합에 소속되어 있지 않은 경우)과 1(소속되어 있는 경우)의 값만을 가진다. 다시 말해서($m_A : X \rightarrow \{0,1\}$)의 형태가 된다.

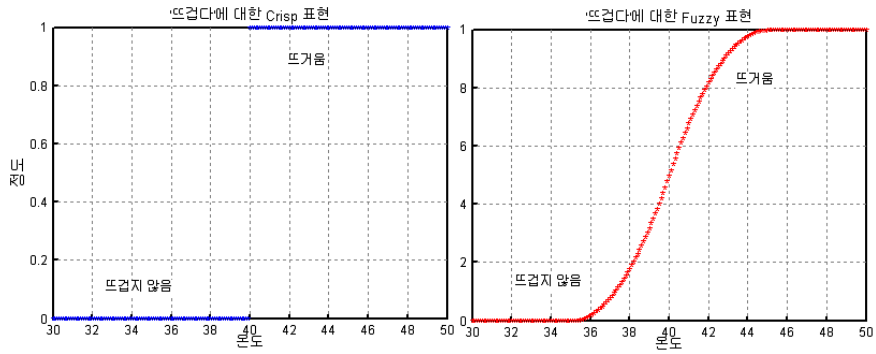
X에 속한 임의의 원소 x 각각에 대해 어떤 특정한 성질을 가지는 정도를 나타내는 소속함수 $m_A(x)$가 정의된다고 하자. 이 경우 순서쌍의 집합 $A = \{(x, m_A(x)) | x \in X\}$를 '소속함수 $m_A(x)$를 갖는 퍼지집합'이라고 한다.

소속함수를 이용하여 목욕탕 물이 뜨거운 정도를 다음과 같이 나타낼 수 있다.

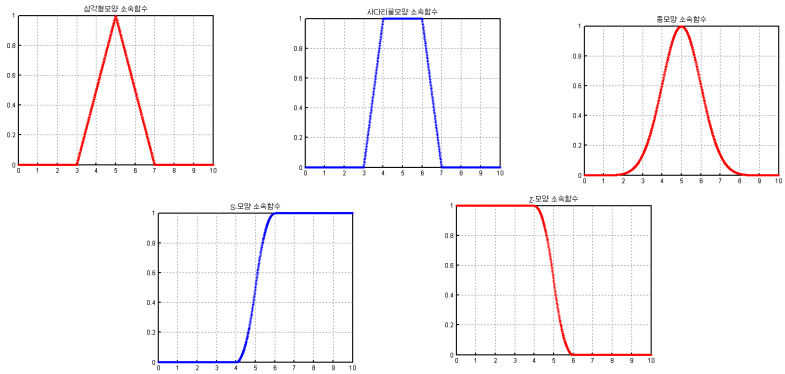
$$A = \text{'뜨거운물'} = \{(x, m_A(x)) | x \in [0,100]\}$$

$$m_A = \left\{ \begin{array}{c} 0, x \leq 35 \\ 2 * \left(\frac{x-35}{10} \right)^2, 35 < x \leq 40 \\ 1 - 2 * \left(\frac{45-x}{10} \right)^2, 40 < x \leq 45 \\ 1, x > 45 \end{array} \right\}$$

퍼지 집합의 특성을 나타내는 소속함수의 필요조건은 위에서 말한 대로 치역이 [0,1]이면 된다. 하지만 집합의 특성 및 계산상의 편리성을 고려해 볼 때 삼각형모양, 사다리꼴 모양, 종형, S-모양, Z-모양 등과 같은 모양의 소속함수들이 주로 사용된다. (그림 2 참고, 자세한 함수 식 및 파라미터는 이 책의 2장 Reference부분을 보기 바란다.)



<그림 2 '뜨겁다'에 대한 소속함수를 이용한 Crisp적인 표현과 Fuzzy적인 표현 >



<그림 3 여러 가지 소속함수들>

2. 퍼지 집합의 연산

기존 집합에서와 같이 퍼지 집합에서도 여러 가지 연산을 정의할 수 있다. 이 중에서 기본적인 합집합, 교집합, 여집합 등을 정의하여 보자.

ⅰ) 퍼지 여집합 (Complement)

퍼지 집합 A에 대한 여집합 $\overline{A}$에 대한 정의는 집합 A의 소속함수로부터 $\overline{A}$의 소속함수로의 매핑 $c:[1,0]\rightarrow[1,0]$로 정의할 수 있다. 이때 매핑 c는 다음의 두 가지 조건을 만족시키는 것이어야 한다.

① c(0) = 1, c(1) = 0 (boundary condition)

② $\forall a, b\in[1,0]$, if $a<b$, then $c(a)\geq c(b)$ (nonincreasing)

가장 간단하면서 가장 많이 쓰이는 여집합 식에는 아래의 식이 있다.

$$m_{\overline{A}}(x)=1-m_A(x),\ \forall x\in X$$

ⅱ) 퍼지 합집합 (S-Norm)

두 퍼지 집합 A와 B의 합집합에 대한 정의도 두 집합의 소속함수를 새로운 집합 $A\bigcup B$의 소속함수로의 매핑 $s:[0,1]\times[0,1]\rightarrow[0,1]$로 정의할 수 있다. 매핑 s는 다음의 네 가지 조건을 만족 시켜야 한다.

① $s(1,1)=1, s(0,a)=s(a,0)=a$ (boundary condition)

② $s(a,b)=s(b,a)$ (commutativity)

③ If ($a\leq a'$, $b\leq b'$), then $s(a,b)\leq s(a',b')$ (nondecreasing)

④ $s(s(a,b),c)=s(a,s(b,c))$ (associativity)

간단하면서 가장 많이 쓰이는 합집합 매핑은 Maximum연산이다.

$$m_{A\cup B}(x)=\max\left(m_A(x), m_B(x)\right)\ \forall x\in X$$

ⅲ) 퍼지 교집합 (T-norm)

두 퍼지 집합 A와 B의 교집합 연산도 두 집합의 소속함수를 새로운 집합 $A\bigcap B$의 소속함수로의 매핑 $t:[0,1]\times[0,1]\rightarrow[0,1]$로 정의할 수

있다. 매핑 t는 다음의 네 가지 조건을 만족 시켜야 한다.

① $t(0,0)=0; t(a,1)=t(1,a)=a$ (boundary condition)

② $t(a,b)=t(b,a)$ (commutativity)

③ If ($a<a'$, $b\leq b'$), then $t(a,b)\leq t(a',b')$ (nondecreasing)

④ $t(t(a,b),c)=t(a,t(b,c))$ (associativity)

퍼지 교집합 연산으로 주로 사용되는 퍼지 매핑은 두 가지가 있는데, 하나는 Minimum연산이고, 하나는 Product연산이다.

Minimum. : $m_{A\cap B}(x)=\min(m_A(x), m_B(x)), \forall x\in X$

Product : $m_{A\cap B}(x)=m_A(x)\cdot m_B(x), \forall x\in X$

3. 퍼지 관계 및 퍼지관계의 합성

일반적으로 관계를 나타내는 집합 R은

$R=\{(x_{1,}\, x_2)|x_1Rx_2,\, x_1\in X\;\; x_2\in Y\}$로 표시한다.

퍼지 관계도 이와 비슷하다. Cartesian Space $X\times Y$에서 관계 R은 $X\times Y$의 퍼지 부분집합으로, 소속함수 $m_R(x,y): X\times Y\rightarrow[0,1]$를 갖는다. 즉, $R=\{((x,y), m_R(x,y))|x\in X, y\in Y\}$를 말한다.

예) $y=f(x)$라는 함수 관계와 유사하게 $y\cong f(x)$ (y 는 $f(x)$와 거의 같다.)라는 관계를 하나의 퍼지 관계 R로 표현하는 경우

$m_R(x,y)=e^{-(y-f(x))^2}$와 같이 표현할 수 있다.

두 퍼지 관계의 합성에 대해서 살펴보자. 두 퍼지관계 $R\subset X\times Y$및 $S\subset Y\times Z$가 주어질 때, 합성 $R\circ S$는

$$R\circ S=\{((x,z), m_{R\circ S}(x,z))|\,(x,y)\in R,(y,z)\in S\}$$

로 정의한다.

전형적인 합성 연산자로는 sup-min composition과 sup-product composition이 있다.

i) sup-min 합성

$$m_{R\circ S}(x,z)=\sup_{y\in Y}\{\min(m_R(x,y), m_S(y,z))\}$$

ii) sup-product 합성

$$m_{R\circ S}(x,z)=\sup_{y\in Y}\{m_R(x,y)\cdot m_S(y,z)\}$$

4. 퍼지 명제, IF-THEN 규칙 및 근사추론

퍼지 명제는 서술부분이 퍼지 개념으로 표현되는 명제이다. 명제 "x는 트다."는 일종의 퍼지 명제로서 여기서 "크다"라는 언어표지(Linguistic Label)는 하나의 퍼지 개념이다.

전건부(Premise)와 후건부(Consequent)가 퍼지 명제로 된 IF-THEN 서술을 퍼지 규칙이라고 부른다. 예컨대,

If x_1 is A_1 & x_2 is A_2 then y is B

가 좋은 예가 될 수 있다. 여기서 x_1, x_2, y는 변수들이고 A_1, A_2, B는 언어 표지 혹은 값이다.

이제 위와 같은 모양의 규칙들을 바탕으로 x_1, x_2에 어떤 값이 주어졌을 때의 y의 값을 추론하는 방법에 대해서 생각해 보자.

퍼지 근사추론을 정리해 보면 다음과 같은 형태를 가진다.

Given: i) Rule : If x is A, then y is B
ii) Input : x is A'
Objective: try to find output : y is B'

Rule 'If x is A, then y is B'는 변수 x, y가 각각 퍼지 입력공간 X와 출력공간 Y의 퍼지 변수라고 가정하면, X와 Y의 Product space 안에서의 관계 $R: A \rightarrow B$로 나타낼 수 있다. R을 함의(Implication)이라고 하는데, 함의함수로는 다음의 두 가지가 많이 쓰인다.

$$m_{A \rightarrow B}(x, y) = \min(m_A(x), m_B(y)) \quad \text{(Mamdani 함의)}$$
$$m_{A \rightarrow B}(x, y) = m_A(x) \cdot m_B(y) \quad \text{(Larson 함의)}$$

이제 주어진 조건들로부터 y는 다음의 식으로 구할 수 있다.

$$B' = A' \circ R \quad (\text{or, } B' = A' \circ (A \rightarrow B))$$

위의 식을 구하는 방법(근사추론 방법)은 $A \rightarrow B$의 함의와 합성연산 $\circ$의 방법에 따라 다음의 두가지가 많이 쓰인다.

ⅰ) Mamdani의 최소최대 연산법 (Sup-Min 방법)

Mamdani는 $R: A \rightarrow B$를 $R = A \times B$로 간주하고 합성 연산 $\circ$로서 Sup-Min 방법을 제안하였다. 즉,

$$B' = A' \circ (A \rightarrow B) = A' \circ (A \times B)$$

$$m_{B'}(y) = \sup_{x \in X} \min\{m_{A'}(x), m_R(x, y)\}$$

$$= \sup_{x \in X} \min\{m_{A'(x)}, m_A(x), m_B(y)\}$$

$$= \min\left\{ \sup_{x \in X} \min\{m_{A'}(x), m_A(x)\}, m_B(y) \right\}$$

$$= \min\{w, m_B(y)\}$$

여기서 $w = \sup_{x \in X} \min\{m_{A'}(x)\}$는 A와 A'의 적합도라고 하며 , A와 A'의 겹치는 정도를 나타내는 지표가 된다.

만약 규칙들이 하나가 아닌 경우에는 어떻게 할까. 여러 개의 'IF-THEN'규칙이 주어진 경우는 그 규칙들을 집합(Aggregation)하여 출력을 구한다. 집합방법으로는 대체로 Maximum 연산을 취한다.

ⅱ) Larson의 Sup-product 연산법

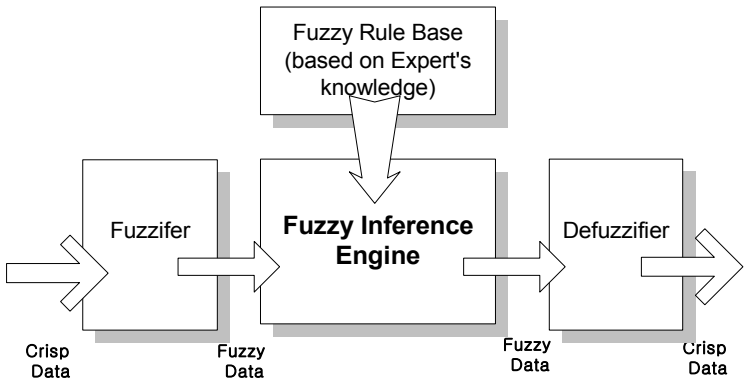
기본적으로 Larson의 방법은 Mamdani의 방법 중 Minimum연산 대신 Product연산을 취한 방법이다. 즉,

$$m_{B'}(y) = \sup_{x \in X} m_{A'}(x) \cdot m_R(x, y)$$

$$= \left\{ \sup_{x \in X} m_{A'}(x) \cdot m_A(x) \right\} \cdot m_B(y)$$

$$= w \cdot m_B(y)$$

5. 퍼지 제어기의 구조



< 그림 4 퍼지 제어기 구조의 그림>

퍼지 제어기의 구조는 위의 그림 4와 같이 표현된다. 크게 퍼지화기(Fuzzifier), FRB(Fuzzy Rule Base)를 포함한 퍼지 추론부, 그리고 비퍼지화기(Defuzzifier)로 구성된다. 이제까지 우리가 다루어 온 부분은 퍼지 제어기의 핵심부라고 할 수 있는 퍼지 추론부이고 이제 부터는 Fuzzifier와 Defuzzifer에 대해서 알아 보자.

ⅰ) 퍼지화기(Fuzzifier)

Fuzzifier는 Crisp한 수치적 정보를 퍼지 집합으로 변화시키는 연산이다. 가령 방안의 창문을 여는 퍼지 제어기를 생각해 볼 때, 제어규칙이 '온도가 ***높으면***, 창문을 많이 열어라'와 같이 언어적으로 주어진데 반하여 방안의 온도는 32.5도와 같이 수치로 주어진다. 이 수치를 퍼지 입력으로 바꾸는 것이 Fuzzifier이다. 주로 사용되는 Fuzzifier는 Singleton Fuzzifer이다. Singleton Fuzzifier는

$$m_{e_0}(e) = \begin{cases} 1, \text{ if } e = e_0 \\ 0, elsewhere \end{cases}$$

와 같이 표현된다.

Singleton Fuzzifier 이외에 이등변삼각형 Fuzzifier와 같은 여러 가지 종류의 Fuzzifier가 있지만, 계산상의 간소함과, 입력수치가 실제 값이 아닌 노이즈를 포함한 값이라는 점, 그리고 퍼지 제어기가 노이즈에 대해서 매우 강인하다는 점등으로 인해서 다른 방법들은 거의 쓰이지 않는다.

ii) 비퍼지화기(Defuzzifier)

Defuzzifier는 Fuzzifier의 반대기능의 한다. 퍼지 제어기에서 퍼지 추론부의 (퍼지 집합으로 표현되는) 출력으로부터 보통의 수치 데이터 (Crisp number data)를 얻어내는 장치다. Defuzzifier로는 다음과 같은 것들이 쓰인다.

(a) 무게중심법 (Center-of-Gravity Method)

무게중심법은 직관적으로 가장 합리적이고, 타당해 보이는 방법이며, 그래서 가장 많이 쓰는 방법중의 하나이다. 출력 퍼지 집합의 소속함수로부터 출력 데이터를 구할 때 무게중심을 구하는 방법을 쓴다. 수식적으로 표현하면,

$$y^* = \frac{\int_V y \cdot m_{B'}(y)\, dy}{\int_V m_{B'}(y)\, dy}$$

(단, $m_{B'}(y)$는 출력 퍼지 집합의 소속함수)

(b) 평균중심법 (Center-of-Average Method)

무게중심법이 직관적으로 가장 훌륭한 Defuzzifying 방법이라곤 하지만 대체로 적분계산은 많은 시간을 요한다. 그래서 대안으로 제시된 것이 평균중심법이다. 이를 수식으로 표현하면,

$y^* = \frac{\sum_{l=1}^{M}(\bar{y}^l \cdot w_l)}{\sum_{l=1}^{M} w_l}$, (단 $\bar{y}^l$은 l번째 규칙에 의한 출력의 평균치, w_l은 l번째 규칙의 적합도를 나타낸다.)

평균 중심법은 무게 중심법과 거의 차이 없이 잘 동작하므로 가장 많이 쓰이는 비퍼지화 방법이다.

(c) 최대치법 (Maximum Method)

개념적으로, 최대치법은 y^*를, 출력 공간 V에서 $m_{B'}(y)$값이 최대가 되는 점으로 잡는 방법이다. 최대치법 안에는 세 가지의 방법이 있다. 우선,

$$m_{\max}(B') = \left\{ y \in V | m_{B'}(y) = \sup_{y \in V} m_{B'}(y) \right\}$$

를 정의하자. $m_{\max}(B')$은 출력 공간 V에서 $m_{B'}(y)$가 최대를 가지는 점들의 집합이다. 이제 세 가지의 방법을 살펴보자.

우선 'smallest of maxima defuzzifer'는 이름처럼 $m_{\max}(B')$에서 가장 작은 값을 취한다. 수식으로 나타내면,

$$y^* = \inf\{ y \in m_{\max}(B') \}$$

그리고, $m_{\max}(B')$에서 가장 큰 값을 출력 값으로 정하는 비퍼지화기는 'largest of maxima defuzzifier'로 다음의 수식으로 표현된다.

$$y^* = \sup\{ y \in m_{\max}(B') \}$$

비슷하게, $m_{\max}(B')$의 평균값을 취하는 방법인 'mean of maxima defuzzifer'는,

$$y^* = \frac{\int_{m_{\max}(B')} y dy}{\int_{m_{\max}(B')} dy}$$

와 같이 표현된다.

최대치법은 계산량이 적어 유리하지만, 출력값의 연속성을 보장하지 못하는 결점이 있다.

6. Takagi-Sugeno-Kang 퍼지 제어기

이제까지 설명해 온 퍼지 제어기는 Mamdani가 제안한 퍼지 제어기의 형식을 따르고 있다. 그래서 Mamdani type퍼지 제어기라고도 한다. Mamdani type과 함께 많이 쓰이고 있는 퍼지 제어기로는 Takagi-Sugeno-Kang type(줄여서 TSK, 혹은 Sugeno type) 퍼지 제어기가 있다. Sugeno type은 Mamdani type과 거의 유사한데 출력 값이 퍼지 집합으로 표현되는 것이 아니라 입력 변수를 포함한 함수의 형태로 주어진다는 것이 다르다. 보통 TSK type은 계산량 및 효율을 고려해 볼 때 입력값의 1차식의 형태를 많이 쓴다. l번째 규칙을 수식적으로 표현하면,

If x_1 is C_1^l & ..... & x_n is C_n^l,

then $y^l = c_0^l + c_1^l x_1 + \ldots\ldots + c_n^l x_n$

의 형태가 된다. 여기서 x_i는 입력 공간의 퍼지 변수들이고, C_i^l은 퍼지 집합, c_i^l은 계수, $l = 1, 2, 3, \ldots\ldots, M$은 규칙의 인덱스를 나타낸다. 위와 같은 퍼지 규칙으로 정의된 Sugeno type퍼지 시스템의 출력값은

$$y^* = \frac{\sum_{l=1}^{M} y^l w^l}{\sum_{l=1}^{M} w^l}$$ (단, w^l은 l번째 룰의 적합도)

와 같이 주어진다.

Sugeno type퍼지 제어기는 평균중심법의 비퍼지화 기법과 유사한 구석이 있다. 그래서 평균중심법과 마찬가지로 계산이 간편하고 계산량도 적다. 또한 Adaptive기법을 사용할 때 파라미터의 업데이트가 편리하기 때문에 최근에 더욱 많이 쓰이는 방법이다.

3. 퍼지 제어

간단한 개념에 비해서 복잡한 수식들을 보고서 '이런 이론들로 무엇을 할 수 있을까?'라는 의문을 한번쯤 가져보지 않았을까. 이제부터는 지금까지 보아온 이론들이 어떻게 실제 제어에서 어떻게 사용되며, 어떻게 퍼지 제어기를 만들 수 있는 지에 대해서 살펴보고자 한다.

1. 퍼지 제어의 발단

퍼지 제어는 인간의 판단 등, 애매성을 포함한 제어 알고리즘을 if-then 형식으로 표현하고, 퍼지 추론을 이용하여 컴퓨터로 실행시킨 것이다.

세계 최초의 퍼지 제어를 시작한 것은 런던 대학의 Mamdani(1974)이고 그의 연구는 컴퓨터 시뮬레이션이 아니라 연구실에서 만든 스팀엔진을 이용한 실증적인 것이었다.

Mamdani의 퍼지 제어연구의 동기가 되었던 것은 1973년 Zadeh가 발표했던 시스템의 언어적 분석에 대한 논문이었으며, Zadeh는 이 논문 중에서 애매성을 포함한 언어를 이용하여 시스템을 기술하는 방법론을 제창하였고, 특히 퍼지 알고리즘에 대하여 상세히 서술하고 있다. Mamdani의 1974년 논문 타이틀도 사실은 퍼지 제어가 아니고 '제어로의 퍼지 알고리즘의 응용'이라는 것이었다.

퍼지 알고리즘에 대해서 Zadeh는 1968년에 발표한 퍼지 알고리즘이라는 논문 속에서 아이디어를 서술하고 있다. 이것에 의하면 퍼지 알고리즘이란 크리스프(crisp) 알고리즘을 퍼지화 한 것으로 다음과 같은 애매한 지시를 포함한 것이다.

"혹시 x가 약 5라면, y는 약 10으로 하시오"

"x가 크다면, y를 얼마간 증가시키시오, x가 작다면 y를 얼마간 감소시키고, 그렇지 않다면 y는 그대로 하시오."

퍼지 알고리즘은 정보 처리, 제어 , 패턴 인식, 시스템 인식, 인공 지능, 의사 결정 등에 도움이 될 것이라고 서술하고, 그 유용성을 멈추어 있는 2대의 자동차 사이에 차를 주차시키는 문제를 들어서 설명하고 있다. 이 문제를 보통의 제어 문제로 푸는 데는 다음과 같이 생각해야한다.

차의 위치를 w, 차의 방향을 θ로 두고, 차의 상태를 벡터 $x=(w,\theta)$로 나타낸다. 조정량은 $u=(u_1,u_2)$로 u_1, u_2는 각각 전륜의 각도와 차의 속도이다. 그러면 차의 운동 방정식은

$$\dot{x}=f(x,u)$$

로 나타내어 질 것이다. u_1은 크기에 제한이 있다. u_2는 $+\alpha$, 0, $-\alpha$의 세 값만을 취한다고 하자. 두 대의 차 사이의 공간은 조금은 융통성이 있기 때문에 차의 최종 허용 상태는 허용되는 x집합 Γ로 나타낼 수 있다. 초기 상태를 x^0라고 하면, 문제는 x를 x^0로부터 Γ의 한 점으로 움직이게 하는 u를 발견하는 문제가 된다. 제약 조건은 u_1에 대한 것과, 차 두 대의 기하학적 배치에 따른 x의 궤적에 대한 제약이다. 이 문제를 기존의 analytic한 방법으로 푸는 것은 거의 불가능에 가깝다. 핸들이 흔들린다던가, 타이어의 상태, 노면의 상태에 의해 마찰 계수가 변한다던가 등으로 아예 방정식 자체를 기술하기도 힘들다. 따라서 이를 풀기 위한 정확한 알고리즘을 발견할 수 없다.

그러나, 자동차 연습장에서 미경험의 운전자는 선생으로부터 구두로 명령을 받는 것만으로 차의 움직임을 기술하는 미분방정식을 알 필요도 없이 차를 잘 주차시킨다. 운전자가 듣는 명령은 “핸들을 우측으로 꺾으면서 전진하고 좌측으로 돌아와 멈추시오. 다음에 우측으로 꺾으면서 후퇴하고, 좌측으로 되돌아오시오. 실패하면 다시 하시오.”와 같은 식이다. 이것은 퍼지 제어 알고리즘이 실제로 이와 같은 방법으로 자동차 주차 문제를 풀 수 있다는 것을 보여준다. 중요한 것은 퍼지 알고리즘에는 로버스트성(robustness)이 있고, 실행을 위해 알고리즘이 지시하는 정확한 의미를 알 필요가 없다는 것이다. 사실이지 주차 문제를 기술하기 위해

서 문제의 시스템을 $\dot{x} = f(x, u)$와 같은 식으로 기술하는 것은 시간과 노력의 낭비임에 동감할 것이다.

1973년 논문에서 Zadeh가 주장하고 있는 것은 인간이 중심인 시스템(Humanistic System)혹은 그것과 동등한 정도의 복잡한 시스템을 취급하기 위해서는 전통적인 정량적인 기법은 적합하지 않다는 것이다.

전통적인 방법을 대신하여 Zadeh가 제안한 방법은 3가지 특징을 가지고 있는데, 첫째는 수치가 아니라 언어적으로 애매한 변수를 이용하여 시스템을 기술하는 것, 둘째는 조건부 명제에 의해 애매한 변수간의 관계를 기술하는 것, 셋째는 시스템의 기술은 퍼지 알고리즘 방식을 이용하여 행하는 것이다.

2. 퍼지 제어 예 : Mamdani의 스팀엔진 제어

Mamdani가 연구한 퍼지 제어계는 아래 그림이 나타내는 바와 같은 2 변수계로, 스팀엔진에 있어서 보일러의 출구 압력과 엔진 속도를 일정하게 제어하는 것이었다.

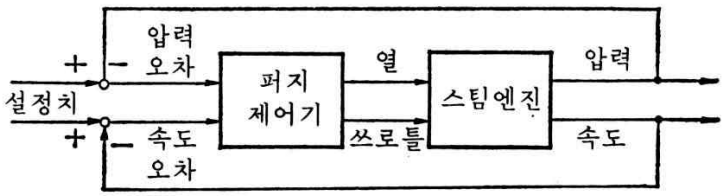


그림 5 스팀 엔진의 퍼지 제어계

입력은 보일러의 공급 열량과 엔진의 쓰로틀이 열려 있는 정도이다. 이제 다음과 같은 변수를 정의한다.

PE = 압력 편차
CPE = 압력 편차의 변화분
SE = 속도 편차
CSE = 속도 편차의 변화분
HC = 공급 열량의 변화분
TC = 쓰로틀이 열려있는 정도의 변화

여기서 편차란 설정치로부터의 차이, 변화분이란 1 샘플링간의 변화분을 의미한다. 이 퍼지 제어계는 미니 컴퓨터 PDP11을 이용하여 실현되었다. 연구에 이용되었던 제어 규칙은 아래 표xx에 나타내었다. Mamdani는 이것들을 LC(Linguistic Control)규칙이라고 불렀다. 퍼지 알고리즘에 있어서의 조건부 명제는 퍼지 제어에서는 Mamdani를 따라 언어적 제어규칙이라든가 혹은 일반적으로 퍼지 제어규칙으로 불려지게 되었다.

그림 6은 표 3.1의 제어 규칙을 사용하여 보일러 출구 압력을 제어한 결과를 PID제어의 결과와 비교한 것이다. Mamdani는 퍼지 제어의 테스트 케이스로서 이용한 스팀 엔진은 비선형성을 가지며, 특성이 시간에

따라 변화하기 때문에 PID제어에서는 때때로 파라미터의 재조정이 필요했지만, 퍼지 제어에서는 그럴 필요가 없었다고 한다.

표 3.1 스팀 엔진 제어의 LC규칙

HEATER ALGOITHM
If PE=NB then ifCPE=not(NB or NM) then HC=PB
If PE-NB or NM then if cPE=NS then HC=PM
If PE=NS then if CPE=PS or NO then HC=PM
If PE=NO then if CPE=PB or PM then HC=PM
THROTTLE ALGORITHM
If SE=PO then if CSE=PB then TC=NS
If SE=PS then if CSE=PB or PM then TC=NS
if SE=PM then if CSE=PB or PM or PS then TC=NS
if SE=PB then if CSE=not(NB or NM) then TC=NB
PB=Positive Big, PM=Positive Medium
PS=Positive Small, PO=Positive Zero, N=Negative

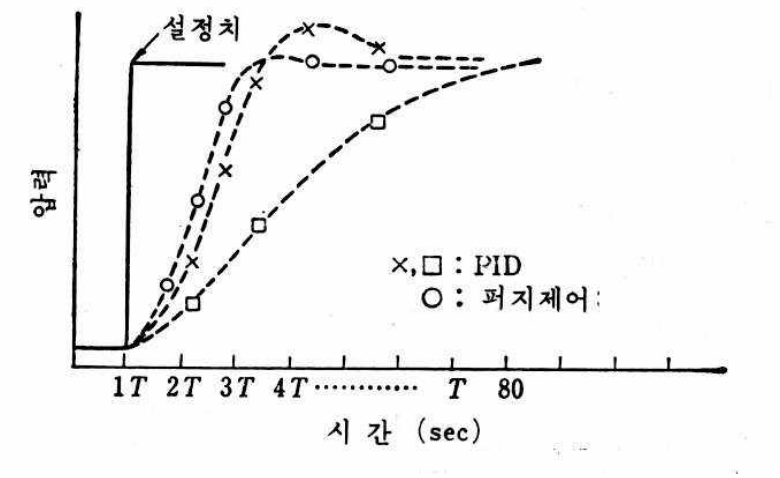


그림 5 스팀 엔진 제어 결과

3. 퍼지 제어기의 설계

그럼 이제부터는 실제로 퍼지 제어기를 어떻게 설계하는 지 알아보자. 퍼지 제어를 실제로 적용 할 때, 최초로 문제가 되는 것은 퍼지 제어 규칙을 어떻게 만드는 가 하는 점이다. 종래의 제어 이론과 같이 설계 방법은 확립되어 있지 않지만, 설계의 지침은 정리되어 있다.

간단한 예로서 그림 7과 같은 응답을 가지는 SISO 플랜트의 퍼지 제어기를 설계해 보자. 제어의 규칙의 형식은 전건부 변수로서 출력의 편차 E와 1 샘플링 시간에서의 E의 변화분 ΔE, 후건부 변수로서는 입력 U의 변화분 ΔU를 생각하기로 한다.

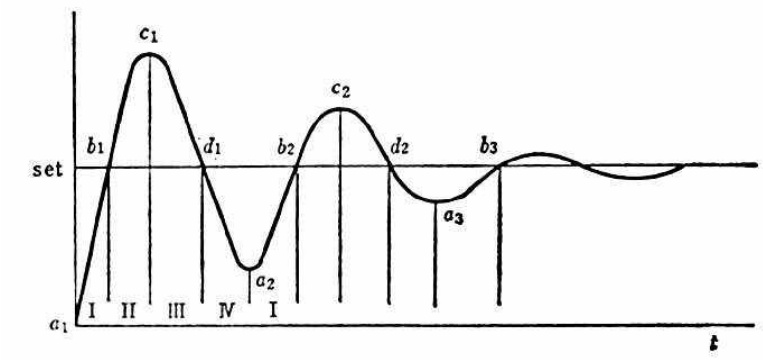


그림 6 플랜트 출력의 시간 응답

시각 n에서의 출력을 Y_n, 편차를 E_n, 입력을 U_n, 설정치를 R이라고 하면,

$$\Delta E = E_n - E_{n-1} = (R - Y_n) - (R - Y_{n-1}) = Y_{n-1} - Y_n$$

$$\Delta U = U_n - U_{n-1}$$

이다. 여기서 ΔE는 출력 Y의 기울기 부호를 역으로 한 것임을 알 수 있다. 출력이 상승하고 있을 때는 $\Delta E < 0$, 하강하고 있을 때는 $\Delta E > 0$이다. 퍼지 제어기의 입출력 관계는

$$(E, \Delta E) \rightarrow \Delta U$$

이다. 이것은 보통 제어 계산 알고리즘에서는 속도형이라 부른다. 이 이름의 유래는 U의 시간 미분, 즉 속도에 상당하는 ΔU를 출력하기 때문이다. 그 구조로서 U 자체를 후건부 변수로 위치형을 채용할 수도 있지만, 그럴 경우 E의 적분치를 전건부 변수로서 사용할 필요가 있어, 계산이 복잡하다. 또 속도형은 후에 서술하는 바와 같이 제어 규칙의 수가 적어도 되는 이점이 있다.

E와 ΔE로부터 ΔU를 추론하는 구조이기 때문에 퍼지 PI제어기라고 불리어지기도 한다. 왜냐하면 E와 ΔU의 관계가 $U(t) = K_I \int E(t)dt$ 라는 적분 함수에 상당하고, ΔE와 ΔU의 관계가 $U(t) = K_p E(t)$라는 비례관계에 상당하기 때문이다.

설계 방식은, 설정되는 플랜트 응답의 여러 가지 상태에 따라서 어떤 제어를 하면 좋겠는가를 기술하는 것이다. 그림으로부터 알 수 있는 것과 같이 이 응답은 4가지의 경우 I, II, III, IV의 반복으로 되어 있다. 2사이클째의 상태는 물론 1사이클째의 상태에 비해 플랜트의 출력의 크기가 감소하고 있다.

우선 각 상태의 특징적인 점을 골라 그곳에서 무엇을 해야 하는 가를 생각해 보자. 퍼지 제어규칙의 전건부는 특징점이 애매하게 기술된다. 예를 들면 1사이클째의 상태 I의 경우 a_1부근에서 편차(설정치 - 출력)가 양으로 되고, 플랜트 출력은 거의 동작이 시작되지 않기 때문에 ΔE는 0에 가깝다. 이것은 $E = NB$ and $\Delta E = ZO$와 같은 모양으로 퍼지 변수의 값을 이용하여 기술한다. 이 부근에서 당연히 입력을 가장 크게하여 $\Delta U = PB$로 증가시켜야 한다. 마찬가지로 하여 상태 II의 점 b_1, III의 점 c_1, IV의 점 D_1등의 부근을 주목하여 제어 규칙을 만들면 다음과 같이 된다.

a_1: if $E = PB$ and $\Delta E = ZO$ then $\Delta U = PB$

b_1: if $E = ZO$ and $\Delta E = NB$ then $\Delta U = NB$

c_1: if $E = NB$ and $\Delta E = ZO$ then $\Delta U = NB$

d_1: if $E = ZO$ and $\Delta E = PB$ then $\Delta U = PB$

2사이클째의 a_2, b_2등의 점 근처에서는 a_1, b_1에 비해 E 혹은 ΔE의 절대치가 상대적으로 작게 되어 있을 뿐이기 때문에 이에 따라서 ΔU의 값도 작게 하면 된다. 예를 들면,

a_2: if $E = PM$ and $\Delta E = ZO$ then $\Delta U = PM$

b_2: if $E = ZO$ and $\Delta E = NM$ then $\Delta U = NM$

으로 하면 된다.

표 3.2 퍼지 규칙 제어표

		ΔE						
		NB	NM	NS	ZO	PS	PM	PB
E	NB				NB			
	NM				NM			
	NS				NS			
	ZO	NB	NM	NS	ZO	PS	PM	PB
	PS				PS			
	PM				PM			
	PB				PB			

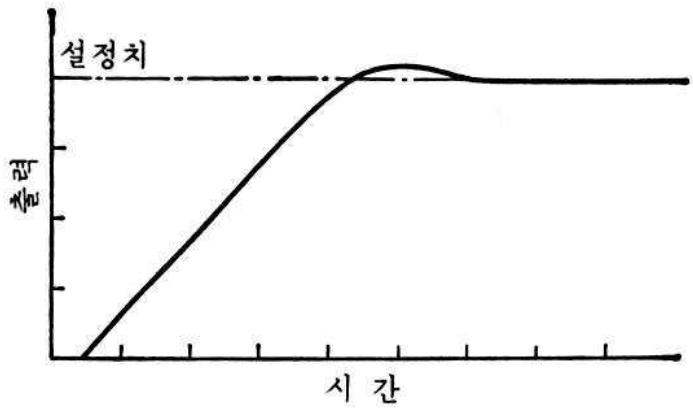


그림 7 표 3.2의 규칙으로 제어한 플랜트의 응답

표 3.2는 이렇게 하여 만든 13개의 규칙을 나타낸 것이다. 세로는 E의 값, 가로는 ΔE, 표 안에는 ΔU의 값을 보여주고 있다. 이 퍼지 제어기로 제어한 결과를 그림...에 나타내었다. 표 3.2의 제어 규칙표는 매우 깨끗이 나열되어 있다. 그림 7의 응답에 따라서 규칙은 a_1, b_1, c_1, d_1의 순으로 적용되어 간다. 이 모습을 $(E, \Delta E)$의 위상면에서 보면 표 3.2의 중앙 밑에 있는 $(E, \Delta E) = (PB, ZO)$인 곳에서부터 시작되어, 시계 방향의 나선형으로 규칙이 적용되어 간다. 표로부터 알 수 있는 바와 같이 최대 규칙의 최대수 $7 \times 7 = 49$에 비해 실재로 있는 것은 13개이고 거의 모든 부분은 공백이다. 왜 이렇게 적은가 하면 하나의 규칙이 커버하는 영역이 넓다고 보기 이전에 우선 표의 4구석과 같이 현실적으로 있을 수 있는 영역(예를 들면, $E = PB$ and $\Delta E = NB$)에서는 규칙이 필요 없다는 것과 속도형 구조를 채용하고 있다는 것이다.

표 3.2의 대집합의 이산적 요소 -6, -5, ..., 0, ..., 6을 참조하여 $E = 5$, $\Delta E = 1$의 입력에 대해서는 다음의 2가지 규칙이 적용된다.

a_1: if $E = PB$ and $\Delta E = ZO$ then $\Delta U = PB$

a_2: if $E = PM$ and $\Delta E = ZO$ then $\Delta U = PM$

이것은 $E=5$에 대한 PB와 PM의 등급이 모두 7이고, 또 $\Delta E=1$에 있어서 ZO의 등급도 모두 7로, 0이 아닌 것으로부터 알 수 있다.

한편 $E=3$, $\Delta E=-3$일때는 어떻게 되겠는가. $E=PM$ or PS, $\Delta E=PM$ or PS정도이기 때문에 사용할 수 있는 규칙은,

if $E=PM$ and $\Delta E=ZO$ then $\Delta U=PM$

if $E=PS$ and $\Delta E=ZO$ then $\Delta U=PS$

if $E=ZO$ and $\Delta E=NM$ then $\Delta U=NM$

if $E=ZO$ and $\Delta E=NS$ then $\Delta U=NS$

의 네 가지이다. 그러나, 최초의 두 가지 규칙은 $\Delta E=-3$에 대하여 $\Delta E=ZO$의 등급이 0, 마지막 두 가지 규칙은 $E=3$에 대하여 $E=ZO$의 등급이 0이기 때문에 적합도가 0으로 되어 이 규칙은 적용되지 않는다. 추론 결과인 ΔU의 값은 당연히 0으로 세트된다. 추론을 실행해도 좋지만 ΔU는 0으로 된다. 그러나 이것으로 인해 결코 부적합성은 생기지 않는다. 그것은 $\Delta U=0$은 현재의 입력 U를 그대로 유지하는 것을 의미하고 있기 때문이다. 만약 위치형으로 하면 적용하는 규칙이 없는 공백의 영역을 만들 수 없다. U의 값이 갑자기 0으로 되어서는 곤론하기 때문이다.

이 예에서 알 수 잇는 바와 같이 퍼지 제어는 눈에 보이는 구체적인 사항으로 무엇을 하면 좋겠는가 라는 알고리즘으로 이루어져 있기 때문에 퍼지 제어기의 능력을 개선하기 쉬운 면도 가지고 있다. 예를 들면 그림 그림 8의 응답이 올라가는 것을 개선한다고 생각해 보자. 이를 위해서는 그림 7에 있어서 a_1의 약간 위쪽 부근에서 더욱 U를 증가하도록 하면 된다. 표 3.2에서는 $E=PB$ and $\Delta E=NS$, 요컨대 플랜트 출력이 조금 올라간 데에 규칙이 없기 때문에 $\Delta U=0$이고 U의 값은 a_1인 곳의 규칙으로 결정되는 것으로 유지되도록 되어 있다. 그래서,

if $E=PB$ and $\Delta E=NS$ then $\Delta U=PM$

이라는 규칙을 추가하여 첫 동작을 가속시켜 보자. 이로서 첫 동작은 좋게 되지만 플랜트의 지연이 있기 때문에 이대로는 Overshoot도 크게 되는 것이 예상된다. 그래서 b_1의 조금 전에서 미리 U의 값을 감속시키기 위해

if $E = PS$ and $\Delta E = NB$ then $\Delta U = NM$

이라는 규칙을 추가한다. 또한 정정을 빨리 하기 위해 $E = PS$ and $\Delta E = NS$인 곳에서 $\Delta U = ZO$라고 한다. 이렇게 해서 표3.2의 퍼지 제어규칙을 개선한 것을 표3.3에 보여 준다. 설정치의 위측에서도 똑같이 생각하여 모두 6개의 규칙이 더해진다. 이 퍼지 제어기로 제어한 결과를 그림 9에 나타내었다. 앞의 결과에 비해 확실히 상승 시간이 빠르게 되고 게다가 Overshoot도 상대적으로 억제되고 있음을 알 수 있다.

이와 같은 개량이 정성적인 판단을 기본으로 간단히 실현될 수 잇는 것은 퍼지 제어기가 영역마다 복수개의 페어 규칙의 집합에 의해 구성되어 있기 때문이다. PID와 같이 하나의 식으로 제어 알고리즘을 기술하면 이와 같이 섬세하게 개량하는 것은 어렵게 된다. 물론 플랜트에 지연이 있을 때는 첫 동작이 상승시간을 빠르게 하는 것과 오버슈트를 작게 하는 것은 서로 모순된 목적이기 때문에 퍼지 제어에 있어서도 한계가 있다. 그러나 PID에 비하면 퍼지 제어 쪽이 훨씬 개선을 위한 전망을 세우기가 쉽다고 말할 수 있다.

참고) 위의 결과를 퍼지 제어 툴박스 디렉토리의 demo1.blk의 심툴 블락으로 작성해두었으므로 확인하기 바란다. 각각의 퍼지 제어기의 파일은 demo1_1.fcs, demo1_2.fcs이다. 두 퍼지 제어기의 퍼지 변수와 멤버쉽 함수는 같고, demo1_2.fcs에는 퍼지 규칙이 몇 개 더 추가되어 있다.

표 3.3 개선된 제어 규칙 표

		ΔE						
		NB	NM	NS	ZO	PS	PM	PB
	NB				NB			
	NM				NM			
	NS				NS			
E	ZO	NB	NM	NS	ZO	PS	PM	PB
	PS				PS			
	PM				PM			
	PB				PB			

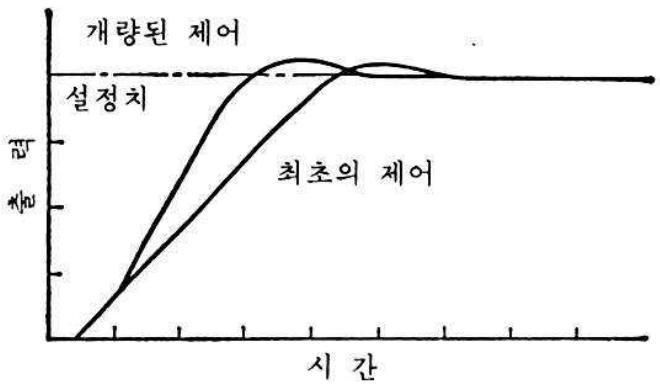


그림 8 개선된 퍼지 제어기 응답

이상과 같이 퍼지 제어기를 설계한다는 것은 제어 규칙의 형식과 추론법을 정하여 구체적으로 제어 규칙을 기술하는 것이지만, 문제는 전건부와 후건부의 결정 2가지로 나누어진다. 전건부에 대해서는, ① 제어를 위한 정보, 즉 퍼지 에어의 전건부 변수 x_1, x_2등의 선택 ② 조건의 설정, 즉 전건부 변수 공간의 퍼지 분할 결정 ③ 퍼지 변수 파라미터의 결정이 필요하게 된다.

후건부 변수는, 일반적으로 플랜트에 부가되는 입력인 경우가 많기 때문에 이는 스스로 결정되는 셈이며, 문제는 퍼지 변수의 파라미터뿐이다. 단, TSK형식의 퍼지 제어기를 만들 때는 후건부 선형식의 계수를 결정하지 않으면 안 된다. 그러나 본질적인 부분은 어디까지나 전건부에 달려 있다. 많은 경우 퍼지 제어가 잘되는 가 어떤가는 적적한 정보의 선택과 조건에 따른다고 해도 좋다. 퍼지 제어는 크게 나누어 ① 전문가의 경험과 지식에 의거한 방법, ② 오퍼레이터의 조작 모델을 만드는 방법, ③ 플랜트의 퍼지 모델에 의거한 방법 등이 있다.

- 전문가의 경험과 지식 (Expert's Knowledge)

전문가의 경험과 지식을 활용한다는 것은 숙련된 오퍼레이터의 경험이나 해당 분야 제어 전문가의 지식 등을 정성적인 언어로 표현하고 이를 퍼지 제어 규칙의 형태로 논리화하는 것이다.

퍼지 제어기의 설계 문제 중에 전건부 변수는 이미 조업되고 있는 플랜트라면 저절로 알 수 있기 때문에 문제는 없다. 가령 자동차 운전의 문제라면 위치와 방향을 전건부 변수로 하는 것은 당연하고, 속도가 변하는 경우에는 속도도 필요하게 된다. 단, 퍼지 제어의 경우 종래의 제어에서 생각되지 않았던 정보를 사용할 수 있다. 이것은 퍼지 제어에서는 조작량(입력)을 결정하기 위한 논리 판단에 유익한 것은 무엇이라도 전건부에 넣으면 좋기 때문이다. 이것은 물리적인 양이 아니어도 좋고, 온라인으로 계측할 수 없는 것이라도 좋다. 예를 들면, 맑음인가 비인가와 같은 날씨의 예측 정보라도 좋다.

- 오퍼레이터의 조작 모델

복잡한 플랜트의 조작을 숙련된 오퍼레이터는 교묘하게 행하고 있지만, 전문가의 노하우를 논리화하는 것은 쉽지 않은 일이다. 특히, 병의 진단과 같은 경우 전문가와 제어 전문가 사이에는 질적으로 차이가 보여지는 것이 있다. 첫째로 제어 전문가로서 플랜트 오퍼레이터가 언어 레벨의 사고 에 따라 제어 구조를 정리하고, 언어 표현에 의해 경험을 축적해도, 그것에 의해 잘 제어된다고 할 수만 없기 때문이다. 예를 들면, 자동차 운전에 있어서의 세심한 조작, 커브 각도에서의 핸들링, 브레이크의 타이밍 등을 언어도 나타내는 것은 불가능에 가깝다. 이것은 눈의 감각과 수족의 움직임 등, 말하자면 기능으로서의 경험을 축적하고 있기 때문이다. 언어에 의한 뇌의 지(知)와 몸에 의한 기(技)와의 질적인 차이이다. 물건을 직접 움직이게 할 때 중요한 것은 '기' 쪽이다.

두 번째로 현장에 오퍼레이터의 협력이 잘 얻어지지 않는 경우가 있다. 오퍼레이터가 자신의 비밀을 전부 밝혀 낸다고는 할 수 없다. (안다해도 세세한 것까지 언어적으로 표현하기는 매우 힘들 것이다.) 결과로서 단편적인 언어에 의한 플랜트 조작의 기술밖에 얻어질 수 없다.

지(知)만으로 제어기 구성이 가능하다고 생각할 지 모르겠지만, 제어에 있어서는 기(技)의 획득도 필요하다. '지'만에 의해 퍼지 제어 규칙을 만들었을 때는 시뮬레이션에 따른 보상과 개선이 필요하며, 스스로 '기'를 획득하려는 노력도 필요하다. 오퍼레이터 조작의 언어 표현이 얻어질 수 없을 때의 설계 방법으로서 오퍼레이터의 조작 모델링 방법이 있다. 이것은 오퍼레이터가 사용하는 정보와 조작 출력 사이의 입출력 관계 모델을 만드는 것이다. 모델의 형식으로 if-then 형식의 퍼지 제어 규칙과 같은 것을 고르면, 그대로 퍼지 제어를 위한 규칙으로 사용할 수 있다. 모델의 설정에는 오퍼레이터 자작에 있어서의 입출력 데이터를 이용한다. 첫 번째 설계법(Expert's Knowledge를 활용하는 방법)에서도 오퍼레이터의 조작 언어 모델을 만들고 있다고 말 할 수 없는 것은 아니다. 여기에서는 언어가 아니라 직접 소속 함수 레벨에서의 모델을 만드는 것

으로 생각하는 것이 차이점이다.

- 플랜트의 퍼지 모델

앞에서 서술한 두 가지 방법은 전문가의 지식을 활용하여 마든 퍼지 제어기였다. 하지만 이것으로는 전문가를 뛰어 넘을 수는 없다. 전문가가 없는 플랜트를 대상으로 하는 경우 혹은 오퍼레이터보다 좋은 제어를 목표로 하기 위한 퍼지 제어기 설계법으로서 플랜트의 퍼지 모델에 의거한 방법이 있다.

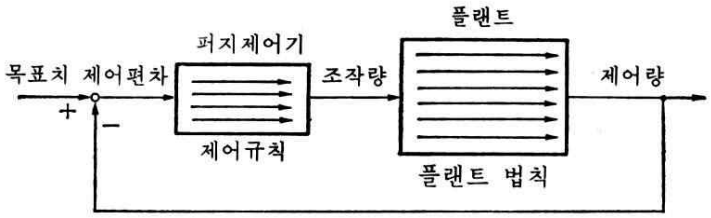


그림 9 퍼지 제어계

여기서 퍼지 모델이란 퍼지 제어 규칙과 같은 if-then 형식을 이용하여 플랜트의 동특성을 기술한 것이다. 플랜트의 퍼지 모델에 있어서 하나의 if-then 표현은 플랜트의 거동 혹은 플랜트의 법칙이라고 불리어진다. 법칙이라고 불리어지는 것은 퍼지 제어에서 사용하는 규칙이라던가 언어가, 무언가 인공적인 것을 의미하기 때문이다. 요컨대, 법치이란 인공적인 규칙에 대한 자연의 법칙이라는 의미다. 따라서 퍼지 모델이란, 복수개의 플랜트 법칙이 모인 것으로 모델로의 입력에 대해 출력은 퍼지 추론에 의해 구해진다. 그림 10은 퍼지 제어 개념을 나타낸 것이다. 그림에서 보여주는 개념은 고전 제어론, 그리고 현대 제어론의 개념과 현저하게 다르다. 그것은 제어계의 기술 언어를 중심으로 보아서 그런 셈이다. 고전 제어론에서는 전달함수로, 현대 제어론에서는 상태변수의 미분 방정식으로 시스템을 기술한다. 그리고 퍼지 제어론에서는 if-then 형식으로 기술한다. 이렇게만 본다면 단지 기술 언어의 차이에 지나지

않지만; 전자의 2가지는 단일 식으로 제어기와 플랜트의 특성을 기술하는 것에 비하여, 퍼지 제어론에서는 복수개의 식으로 기술하기 때문이다. 즉, 고전 제어론도 현대 제어론도 유일한 법칙으로 전체적으로 시스템을 기술하려는 하는 생각에 준하기 때문이다. 한편 퍼지 제어론은 비선형 시스템을 표현하는 경우에 시스템을 전체적으로 기술하는 것은 거의 불가능하다는 관점이다.

플랜트의 퍼지 모델을 만들려고 하는 동기는 물론 지금까지의 제어 이론이 해 온 것과 같이 플랜트 모델로부터 제어 알고리즘을 잘 이끌어 내려는 것이지만, 퍼지 제어기로 설계할 때에 왜 if-then형식의 퍼지 모델인가라고 하면 그것은 제어기와 제어 대상의 기술의 정합성을 구하기 때문이다. 가령 고전 제어론에서 PI제어기의 경우 플랜트의 특징을 나타내는 여러 파라미터들과 PI제어기의 파라미터 K_P, K_I 등이 잘 관계 지어진다. 이것은 현대 제어론에서도 똑같아서 플랜트를 나타내는 행렬 A, B , C, D등을 이용하여 피드백 계수 K를 설계한다.

퍼지 제어기가 추론 방법에 따라서 Mamdani 형, TSK형 등으로 나누어지듯이 퍼지 모델의 기술 형식도 그와 같이 나누어 질 수 있다. 일반적으로 플랜트 모델을 기술하는데는 TSK형을 많이 쓴다. Mamdani 형으로도 기술 할 수는 있지만, 플랜트의 Behavior를 기술하기 위해서 필요한 규칙의 수가 너무 커져버린다.

퍼지 모델링 방법과 퍼지 모델로부터 제어기를 설계하는 방법은 말로 서술하기엔 복잡하므로 참고서적[1,2]을 참고하기 바란다. 여기서는 개략적인 방침만을 생각해보자.

일반적으로 모델이 플랜트 모델이 주어지고 제어기를 설계하는데는 2가지 생각하는 방식이 있다. 하나는 바람직한 플랜트 behavior를 생각하여 제어한 결과가 그렇게 되도록 제어 알고리즘을 발견하는 것이다. 예를 들어, 현재 제어론의 모델 규범 제어(Model Reference Control) 방식의 제어기 설계는 이 방식에 따라 설계된다. 퍼지 제어에서도 플랜트 법칙과 제어 규칙을 직렬로 결합한 결과가 바람직한 응답을 주는 것으로부터,

플랜트 법칙 + 제어 규칙 = 바람직란 플랜트 응답

이라는 도식으로 제어 규칙을 찾아 낼 수 있다.

또 하나의 방법은 제어할 때의 플랜트 응답 특징을 스칼라 양으로 평가하여 그 값을 기준으로 알고리즘을 찾아내는 것이다. 예를 들면, Overshoot가 10%미만 되게 제어기를 작성하는 것 등이 있을 수 있다. 또한 현대 제어에서 Cost Function이 대표적인 예라고 할 수 있다. 도식적으로 나타내면,

플랜트 법칙, 플랜트 응답의 평가 → 제어 규칙

으로 된다. 처음과 같은 사고 방식을 직접법, 평가함수를 사용하는 방법을 간접법이라고 말하기도 한다.

이들 방식은 고전제어, 현대제어, 퍼지 제어에 공통적으로 유효한 것이지만, 이것을 실현할 때에도 퍼지 제어의 특색이 나타난다. 즉 플랜트 모델이 복수개의 플랜트 법칙에 의해 기술되므로 제어 규칙도 분산적으로 유도된다는 것이다.

4. 퍼지 적응제어

앞 절에서 설명한 방식으로 퍼지 제어기를 만들 수도 있지만, 여기에 학습 알고리즘을 도입할 수도 있다. 전문가의 지식을 바탕으로 만들어진 제어기나 혹은 플랜트의 퍼지 모델을 바탕으로 만든 제어기가, 물론 잘 만들어진 경우라면 최적에 가깝겠지만, 반드시 최적일 수는 없다. 그렇지만 학습 알고리즘을 이용하면 Locally Optimum한 제어 파라미터들을 얻을 수 있다.

특히 시스템 파라미터가 변하는 시스템의 경우는 적응 제어가 더욱 절실해 진다. 퍼지 제어기는 강인성을 지니기 때문에 어느 정도의 시스템 파라미터 변화에는 민감하지 않지만, 그 폭을 벗어나 버리면 역시 제대로 동작하지 못 하게된다. 이런 경우에는 반드시 적응제어가 필요하다.

적응 제어란 플랜트의 파라미터 변동에 적응하여 제어 알고리즘을 변화시켜 제어하는 방식이다. 구체적인 형식으로는 모델 규범형 적응 제어가 많이 연구되고 있다. 이것은 바람직한 응답을 보여주는 규범 모델(Reference Model)을 하나 준비하여 모델의 출력에 플랜트의 출력이 일치하도록 플랜트를 제어하는 방식이다.

셈툴 퍼지 제어 툴박스에서는 적응제어를 위하여, Mamdani형의 퍼지 시스템의 경우 falcon이라는 함수를, TSK형의 퍼지 시스템의 경우는 anfis라는 함수를 제공한다.

이들에 대한 설명 및 Neuro(Adaptive)-Fuzzy에 대해서는 뉴로 퍼지 툴박스 사용자 가이드를 참고하기 바란다. 이 장에서는 더 이상의 설명을 마치도록 한다.

4. 예제 : Inverted Pendulum 제어

당신은 앞 절까지의 내용으로 퍼지제어에 관한 어지간한 지식을 갖추게 되었다. 하지만 머리 속의 지식은 활용하지 않는 한 아무런 소용이 없게 된다. 이제 직접 Inverted Pendulum제어를 해보자.

1. 문제 설정 및 분석

우리가 대상으로 제어 대상으로 삼을 플랜트는 아래와 같다.

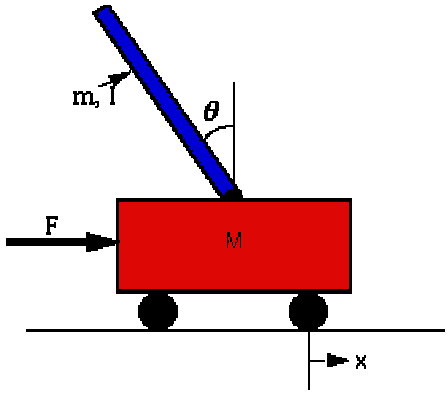


그림 10 Inverted Pendulum on a Cart

Inverted Pendulum을 모델링한 식은 많지만 (직접 해 봐도 좋다.), 우리는 아래와 같은 식을 가지고 플랜트를 모델링 하려고 한다.

1) $(M+m)\ddot{x}+b\dot{x}+ml\ddot{\theta}\cos\theta-ml\dot{\theta}^2\sin\theta=F$

2) $(I+ml^2)\ddot{\theta}-ml\dot{p}\theta\sin(\theta)-mgl\sin\theta=-ml\ddot{x}\cos\theta$

이 모델은 Inverted Pendulum을 선형화 하지 않고 비선형인 항들을 모두 포함하고 있는 식이며, 우리는 막대의 각도 θ뿐만 아니라 위치 x까지 제어하려고 한다. 우리의 목적은 카터의 위치와 막대의 각도를 (0,0)에 수렴시키는 것이다.

우선 가정으로,

M = mass of the cart = 5.0(kg)

m = mass of the pendulum = 1.0(kg)

b = friction of the cart = 0.0(N/m/sec)

I = inertia of the pendulum = 0.0(kg*m^2)

l = length to the pendulum's center of mass = 1.5(m)

F = impulse force applied to cart

와 같이 두자.

플랜트에 가해지는 힘 F를 입력 u로 하고, 출력 y는 카터의 위치 x와 막대의 각도 θ로 이뤄진 벡터 $y=[x\ \theta]'$로 두자. 위의 방정식을 셈툴에서 계산 할 수 있도록 State Space form과 비슷한 형식으로 바꾸어 나타내면,

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} M+m & ml\cos\theta \\ ml\cos\theta & I+ml^2 \end{bmatrix}^{-1} \begin{bmatrix} u-b\dot{x}+mg\dot{x}^2\sin\theta \\ mlp\dot{\theta}\sin(\theta)+mgl\sin\theta \end{bmatrix}$$

$$y=\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix}$$

이제 플랜트는 구해졌으므로, 현재의 상태 변수 $X=[x\ \dot{x}\ \theta\ \dot{\theta}]^T$가 주어졌을 때의 힘 u를 구하는 퍼지 제어기를 구해 보자.

2. 제어기 구성

Inverted Pendulum은 그리 쉽게 제어되는 플랜트는 아니다. 특히 각도뿐만이 아니라 그 위치까지 제어하려고 하면 상당히 까다로운 일이 된다. PID제어기로 Pendulum을 제어해 본 사람은 얼마나 많은 trial-and-error를 거쳐야 하는 지 알 수 있을 것이다. 하지만 퍼지로 하면 쉽다(!).

퍼지 제어기를 구성하기는 해야하는데, 우리가 Inverted Pendulum을 바로 세우는데 익숙한 전문가도 아니므로(만약 그런 희귀한 일에 종사하고 있다면, 당신의 지식을 퍼지 제어기로 구현하는 것도 좋을 것이다.) 우리가 아는 물리학적 지식을 이용하여 룰을 만드는 수밖에 없다. 그리고, 각도 및 위치까지 제어하려고 하면 각도, 각도의 변화율, 위치, 위치의 변화율까지 모두 네 개의 변수에 대하여 값을 고려해야하므로 예제로 삼기엔 상당히 복잡한 시스템이 된다. 여기서는 위치에 대한 제어는 일단은 관심 두지 않고, 각도의 제어만을 살펴보도록 하겠다. (독자들은 위의 모델에 대하여 위치 제어까지 가능한 제어기를 만들어 보기를 권장한다.)

제어 규칙을 어떻게 만들 것인지에 대해서 생각해 보자. 머리 속에 Inverted Pendulum이 있는 수레를 상상해 보자. 그 수레 위의 진자가 한쪽으로 기울고 있는 모습을 상상하라. 진자가 왼쪽(negative)으로 기울고 있다고 가정하면 수레를 왼쪽으로 잡아 당겨야 (negative) 진자는 그 힘에 대한 반작용으로 오른쪽으로 향하게 될 것이라는 것을 짐작할 수 있다. 진자가 이루는 각이 negative이고 각속도가 positive라면 힘을 주지 않아도 0에 가까워 질 것이다. 진자가 오른편으로 기울 때는 왼편으로 기울 때와는 반대 방향으로 힘을 가하면 될 것이라는 것도 쉽게 짐작할 것이다.

이제 퍼지 제어기를 디자인하자. 우선 제어기 구성을 아래와 같이 잡는다.

시스템 입력 : 각도(θ), 각도의 변화율($\dot{\theta}$)

시스템 출력 : 수레에 가해지는 힘(f)

시스템 종류 : Mamdani type

퍼지 시스템 에디터를 실행시켜서 직접 퍼지 제어기를 수정한다. 퍼지 시스템 에디터의 사용법은 제 2장 Reference의 두 번째 절을 참고하기 바란다.

시스템의 입력 변수를 theta, dtheta라고 하자. 각각의 변수들에 대하여 우리가 관심을 가지는 범위를 theta ∈[-0.5, 0.5], dtheta ∈[-1, 1] 정도가 될 것이다. 두 변수에 동일하게 각각 3개씩의 소속함수 negative, zero, positive를 둔다. 소속함수의 종류로는 삼각형 모양의 소속함수가 가장 무난할 것이다.

시스템의 출력을 force라고 하자. force의 값의 범위는 normalize하여 [-1, 1] 이라고 잡는다. 소속함수는 7개를 잡자. 각각의 이름은 nb(negative big), nm(negative medium), ns(negative small), zero, ps(positive small), pm(positive medium), pb(positive big) 이라고 두자. 그리고 아래와 같은 9개의 규칙을 추가하자.

1. IF (theta is negative) & (dtheta is negative) THEN (force is nb)
2. IF (theta is negative) & (dtheta is zero) THEN (force is nm)
3. IF (theta is negative) & (dtheta is positive) THEN (force is zero)
4. IF (theta is zero) & (dtheta is negative) THEN (force is ns)
5. IF (theta is zero) & (dtheta is zero) THEN (force is zero)
6. IF (theta is zero) & (dtheta is positive) THEN (force is ps)
7. IF (theta is positive) & (dtheta is negative) THEN (force is zero)
8. IF (theta is positive) & (dtheta is zero) THEN (force is pm)
9. IF (theta is positive) & (dtheta is positive) THEN (force is pb)

위의 작업은 셈툴의 명령어 라인에서도 할 수 있으며 다음과 같다.

```
//--- Creating fuzzy "inverted pendulum controller " ---

newfcs("a","inverted pendulum controller", "mamdani");
```

```
addvar("a","input","theta",[-1  1],  0.5);
addvar("a","input","dtheta",[-1  1]);
addvar("a","output","force",[-1  1]);
// add MFs to "theta"
addmf("a",1,1,"negative","trimf",[-2  -1  0]);
addmf("a",1,1,"zero","trimf",[-1  0  1]);
addmf("a",1,1,"positive","trimf",[0  1  2]);
// add MFs to "dtheta"
addmf("a",1,2,"negative","trimf",[-2  -1  0]);
addmf("a",1,2,"zero","trimf",[-1  0  1]);
addmf("a",1,2,"positive","trimf",[0  1  2]);
// add MFs to "force"
addmf("a",0,1,"nb","trimf",[-2  -1  -0.66]);
addmf("a",0,1,"nm","trimf",[-1  -0.66  -0.33]);
addmf("a",0,1,"ns","trimf",[-0.66  -0.33  0]);
addmf("a",0,1,"zero","trimf",[-0.33  0  0.33]);
addmf("a",0,1,"ps","trimf",[0  0.33  0.66]);
addmf("a",0,1,"pm","trimf",[0.33  0.66  1]);
addmf("a",0,1,"pb","trimf",[0.66  1  2]);
// addrules
rl = [ 1  1  1  1  1; 1  2  2  1  1; 1  3  4  1  1; 2  1  3  1  1; 2  2  4  1  1;
       2  3  5  1  1; 3  1  4  1  1; 3  2  6  1  1; 3  3  7  1  1];
addrule("a",rl);
savefcs("a","ivpcon.fcs");
//--------------------------------------------------
```

이상의 작업을 거친 퍼지 제어기가 퍼지 제어 툴박스 디렉토리에 있는 “invcon.fcs”이다. Inverted Pendulum의 플랜트 미분 방정식을 포함한 함수는 은 "ivpen.cem"이다. 그리고, demo_ip.cem으로 우리가 만든 제어기를 실험해 볼 수 있다.

3. Simulation

다음은 ivpen.cem, demo_ip.cem의 코드와 제어 결과이다. 간단한 물리적 상식을 이용하여 만든 제어기가 무척 잘 동작하는 것을 볼 수 있을 것이다.

1. ivpen.cem

```
------------------------------------------------------------
function;
y<>x
/*
 Inverted Pendulum 플랜트 모델
 y = ipendul(x)
 x(1:4): current state  x = [theta theta_dot p p_dot]
 x(5): input u , scalar value
 y: updated state

 Plant dynamic equiation
 (M+m)p_2dot + ml*cos(theta)theta_2dot - ml*sin(theta)*theta_dot^2
= u
 mp_2dot*cos(theta) + ml*theta_2dot - mlp_dot*theta_dot*sin(theta)
- mg*sin(theta) = 0
          where M = 5kg, m = 1.0 kg, l = 1.5m, g=9.8m/s^2
*/
u = x(5);
//Constants
M = 5;
m = 1.0;
l = 1.5;
g = 9.8;
//equations
x1 = x(1); x2 = x(2); x3 = x(3); x4 = x(4);
//*
A = [ M+m          m*l*cos(x1); m*cos(x1)       m*l];
B = [u+m*l*sin(x1)*x2*x2; m*l*x4*x2*sin(x1)+m*g*sin(x1)];

dx1 = x2;
dx3 = x4;

temp = inv(A)*B;
dx2 = temp(2); dx4 = temp(1);
```

```
//output
y = [dx1; dx2; dx3; dx4];
return
```

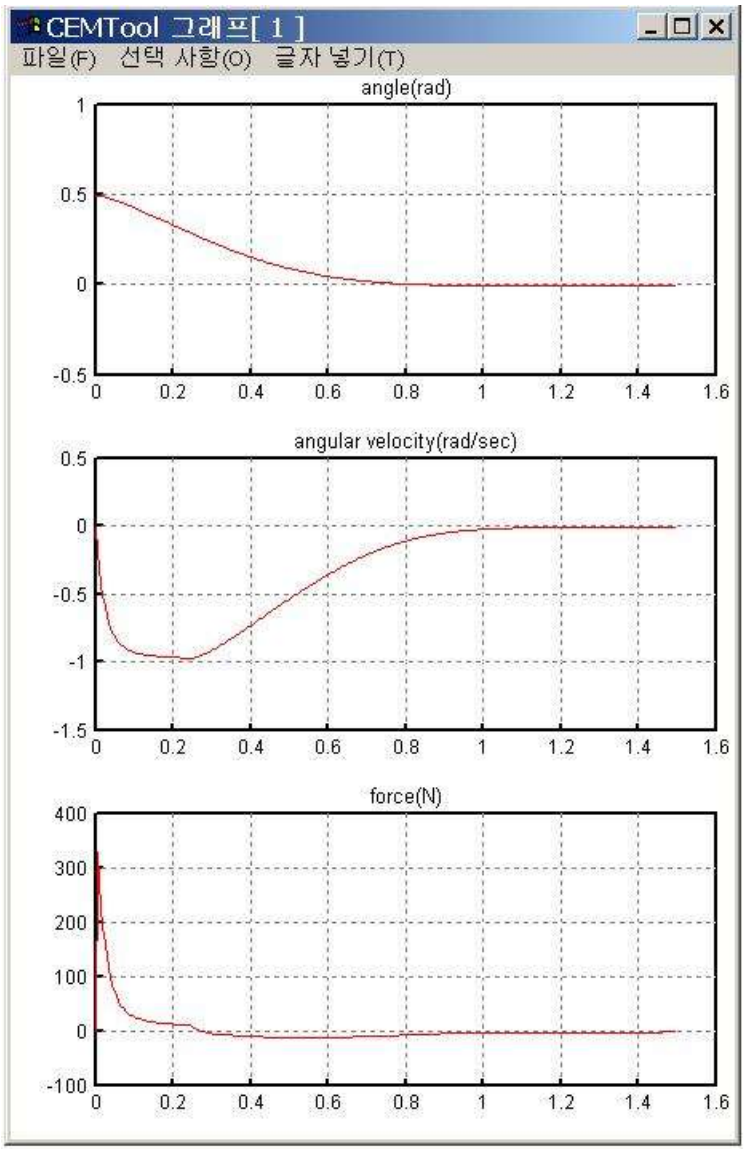
2. demo_ip.cem

```
-------------------------------------------------------------
/*
     Inverted Pendulum Controller Demo program
*/

// variable declaration
stepsize = 0.005; // time step
u = 0; //force
x = [0.5 0 0 0]'; // initial state

iteration = 0;
MaxIteration = 300; // maximum iteration #
x1 = x(1);
x2 = x(3);
x3 = u;
t = [0];
dx = [0 0 0 0]';
loadfcs("a","ivpcon.fcs");
for( iteration = 1; iteration <= MaxIteration; iteration++){
    u = 50*evalfcs("a", [x(1) x(2)]); // cares theta & theta dot
          dx = stepsize * ivpen([x;u]);
          x = x + dx;
          t = [t, iteration*stepsize];
          x1 = [x1, x(1)];
          x2 = [x2, x(2)];
          x3 = [x3, u];
}
subplot(3,1,1);
plot(t,x1);
title("angle(rad)");
subplot(3,1,2);
plot(t,x2);
title("angular velocity(rad/sec)");
subplot(3,1,3);
plot(t,x3);
title("force(N)");
//-------------end ----------------------//
```

3. 결과 그래프


CEMTool 그래프[1]
파일(F) 선택 사항(O) 글자 넣기(T)
angle(rad)
1
0.5
0
-0.5
0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6
angular velocity(rad/sec)
0.5
0
-0.5
-1
-1.5
0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6
force(N)
400
300
200
100
0
-100
0 0.2 0.4 0.6 0.8 1 1.2 1.4 1.6

5. 퍼지 에디터 (Fuzzy Editor)

앞 절의 함수들을 이용하여 퍼지 시스템을 구성 할 수도 있지만, 셈툴의 퍼지 툴박스에서는 보다 직관적이고 편리한 시스템 구성 툴을 제공한다. 그것의 이름을 '퍼지 에디터'라고 붙였다.

이제부터 퍼지 제어기를 셈툴의 퍼지제어 툴박스의 퍼지 에디터를 이용하여 디자인하는 방법을 설명하고자 한다.

셈툴의 퍼지제어 툴박스 GUI환경은 대단히 직관적이다. Matlab의 퍼지 로직 툴박스의 GUI환경에 익숙해진 사람이라도 적응하는데 그다지 시간을 필요로 하지는 않을 것이다. Matlab과 비교한다면 Rule viewer와 Surface Viewer의 기능이 빠져 있는데, 이것은 차후에 추가될 것이다.

셈툴 퍼지제어 툴박스는 '메인 윈도우(혹은 FCS Editor)', '퍼지규칙 편집 창', '소속함수 편집 창'로 구성되어 있다. (그림 12 참조) FCS Editor는 퍼지 제어기의 이름,, 입출력의 개수, 그리고 여러 가지 특징(And Method, Or Method, Implication Method 등등)을 설정한다. '소속함수 편집 창' 는 각 퍼지 변수에 소속된 소속함수들을 더하거나 빼고, 수정할 수 있다. '퍼지 규칙 편집 창'에서는 퍼지 룰들을 고칠 수있다.

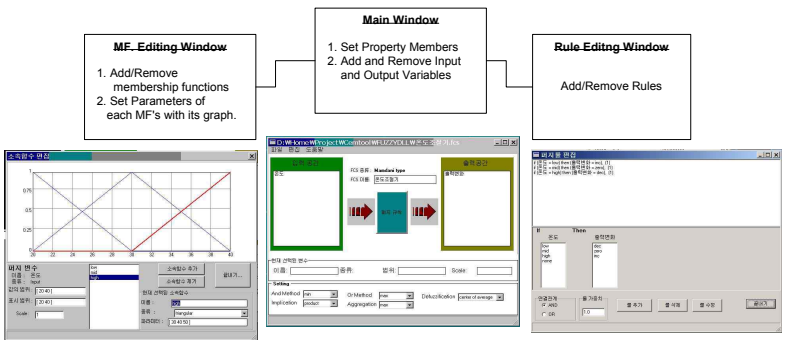


그림 12 퍼지 툴박스의 GUI환경 구성

시작하기

이제부터 간단한 예제를 이용하여 퍼지 제어 툴박스의 GUI 환경을 어떻게 사용하는 지에 대해서 설명하도록 하겠다.

방안 온도 조절 문제 : 방안의 온도 x가 20도에서 40도 사이로 변할 때 에어컨의 출력 y는 어떤 값을 가져야 할까? (편의상 y값이 양수일 때는 에어컨으로 동작하고, 음수일 때는 히터로 동작한다고 생각하자.)

위의 문제 해결을 위해서 에어컨 출력 조정을 위한 규칙을 만들어 보자. 아주 쉽게 생각할 수 있는 법칙으로

1. 온도가 높으면 출력을 높인다.
2. 온도가 적당하면 출력에 변화가 없다.
3. 온도가 낮으면 출력을 낮춘다.

를 생각할 수 있다. 이를 IF-THEN룰로 표현하면 다음과 같다.

1. If x is *high*, then Δy is *positive*.
2. If x is *mid*, then Δy is *zero*.
3. If x is *low*, then Δy is *negative*.

위의 퍼지 제어기는 직관적으로 명확하고, 어지간한 온도의 변화에도 잘 동작할 것으로 보인다. 이제 위의 퍼지 제어기를 퍼지제어 툴박스의 GUI 환경을 이용하여 만들어 보자.

1. The FCS Editor

우리는 지금 새로운 퍼지 제어시스템을 만들려고 한다. 이를 시작하기 위해서 셈툴의 커맨드 라인에서 다음과 같이 타이핑하라.

fuzzy

그러면 아래의 창이 떠는 것을 볼 수 있을 것이다. (물론 설명구름까지 뜨는 것은 아니다.)

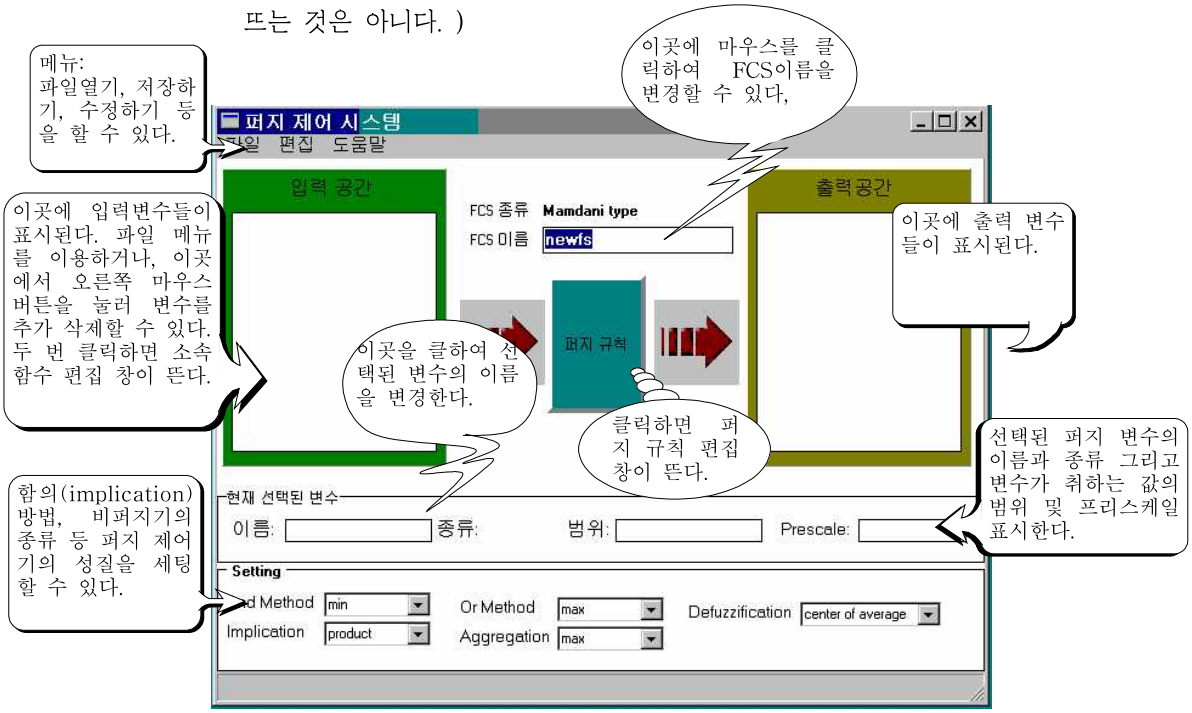


그림 13 FCS editor

그다지 예쁜 모양의 창은 아니지만, C-code 혹은 셈툴 코드로 퍼지 제어기를 구성하는 경우와 비교한다면 당신의 노력을 1/10로 줄여 줄 것이다.

(참고로 퍼지 툴박스 GUI는 한글 WindowsNT에서 C++ Builder로 컴파일 된 것이다. 16-bit OS혹은 영문 Windows에서는 제대로 동작하지 않을 것이다.)

우리가 만들려고 하는 퍼지 제어기의 종류는 Mamdani type퍼지 제어기다. 만약 Takagi-Sugeno-Kange type의 제어기를 만들려고 한다면 파일 메뉴의 '새 퍼지제어기 -> Sugeno type'을 선택하라.

우선 퍼지 제어기의 이름부터 정하자. 'FCS 이름'이라는 글씨 옆에 에디터 창이 보인다. 그곳을 클릭하고 이름으로 '온도 조절기'라고 입력한다. 그리고 일단 저장부터 하자. 저장은 창의 상단 '파일' 메뉴를 선택하면 '저장하기'라는 서브 메뉴를 클릭하면 된다. 그러면 원하는 이름으로 저장할 수 있다. 단축키로 'Ctrl-S'를 눌러도 된다.

이제 입력 변수를 추가하자. 입력 변수를 추가하려면 메뉴바의 '편집-> 입력변수 추가'를 선택한다. 다른 편리한 방법으로, '입력공간'이라고 적혀진 녹색 사각형 안의 하얀색 사각형에 커서를 가져가서 마우스의 오른쪽 버튼을 누르면 팝업 메뉴가 나온다. 그 메뉴 중에서 '입력 변수 추가'를 선택하여도 된다. 이렇게 하여 추가된 변수의 이름을 정해야 한다. 이름을 '온도'라고 정하자. 다중 입력 시스템을 구성하려고 한다면 입력의 개수만큼 위와 같은 방식으로 변수를 추가한다.

출력변수도 비슷한 방식으로 추가할 수 있다. 지금 만들려는 '온도 조절기'는 출력 변수가 하나만 필요하다. 그 이름을 '출력변화'라고 명명하자.

그리고 퍼지 제어기의 여러 가지 설정을 하자. (꼭 지금 설정을 할 필요는 없다. 순서를 요하는 것이라면, 퍼지 규칙을 퍼지 변수들의 소속함수 설정을 끝낸 다음 해야 한다는 정도이다.)

And Method라고 되어 있는 부분은 앞장을 공부한 사람은 알겠지만 퍼지 집합 연산중에서 And 연산(T-norm)의 방법을 뜻하고, Or Method는 Or연산, Defuzzification은 비퍼지화 방법이다. Mamdani형과 TSK형의 방법이 상이한데, 그 종류에 대해서는 앞 절의 setfcs함수 도움말을 참고하기 바란다. 그리고, Implication은 함의 방법, Aggregation은 집합 방법을 정의한다. 무언가 하나가 빠졌다고 생각되는가? Fuzzification인 빠져있다. 셈툴 퍼지 툴박스는 기본적으로 Singleton Fuzzifier를 쓴다고 가정한다.

이제 And 및 Implication 방법을 'product'로, Defuzzification은 'center of average', Or 및 Aggregation은 'max'로 설정하자.

눈치가 빠른 사람은 알겠지만, 우리는 지금 평균중심법과 더불어 Larson의 근사추론 방식을 이용한 퍼지 제어기를 만들려고 하는 중이다. 이 시점에서 'Ctrl-S'를 눌러 저장을 자. 화면의 모양이 아래와 같이 되었다면 제대로 하고 있는 중이다.

이제부터는 각 퍼지 변수에 소속함수를 추가하도록 하자. 소속함수 추가 및 편집은 입력공간(혹은 출력공간)의 퍼지 변수를 선택하고 메뉴바의 '편집->소속함수 편집'항을 선택하거나, 아니면 퍼지 변수를 선택한 다음 마우스의 오른쪽 버튼을 누르면 '소속함수 편집'이라는 항목을 선택하면 된다.

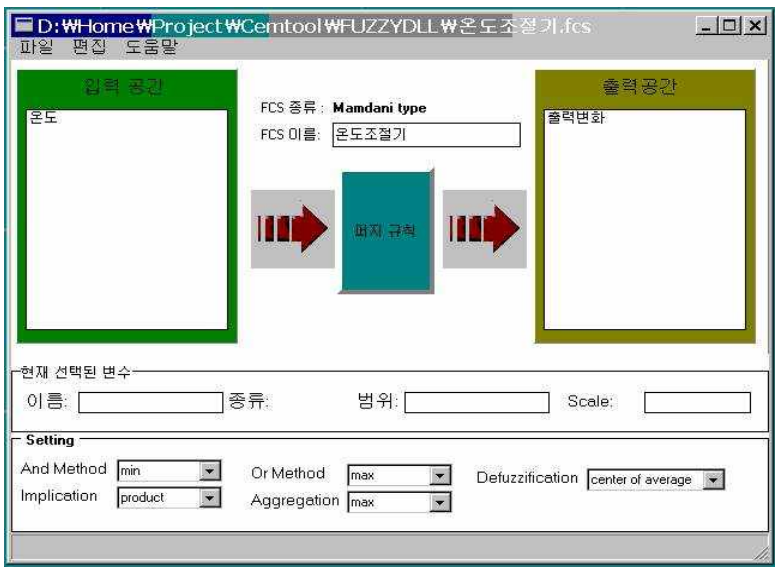


그림 14 온도 조절기 예제

2. 소속함수 편집 창

이제 화면에는 다음의 형태를 한 새로운 창이 만들어졌을 것이다.

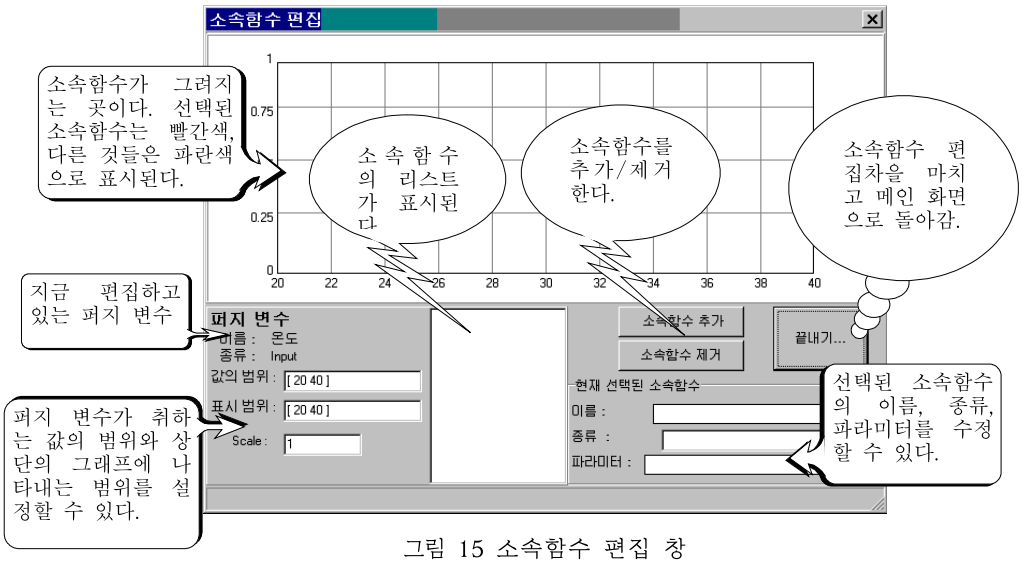


그림 15 소속함수 편집 창

우선 창에 대한 설명부터 하자. 위쪽 격자는 소속함수가 그려질 부분이다. 선택되지 않은 소속함수들은 파란색으로 그려지고, 선택된 소속함수는 빨간색으로 그려진다. (만약 지금 편집하는 퍼지 변수가 Takagi-Sugeno-Kang type의 출력 퍼지 변수일 경우에는 소속함수가 그려지는 대신 수식이 표시된다.) 아래 왼쪽에는 '퍼지 변수'에 대한 설명이 나와있다 - "이름: 온도", "종류: Input". 그 아래에 '값의 범위'와 '표시 범위'가 있는데, 값의 범위는 퍼지 변수가 취하는 값의 범위를 뜻하고, 표시범위는 위의 격자에 그려지는 구간 범위를 말한다. 그리고 Scale이라는 항이 있는데, 그것은 입력 퍼지 변수의 경우 퍼지 변수의 입력으로 들어오는 스칼라 값에 곱해지는 pre-scale factor를, 출력 퍼지 변수의 경우 defuzzification후의 값에 곱해지는 de-scale factor를 의미한다. 이 scale factor를 활용하면 쉽게 normalization을 이룰 수 있다.

그리고 그 옆에 지금은 아무 것도 보이지 않는 사각형이 있는데, 여기는 소속함수들의 리스트가 표시된다. 특정 소속함수를 선택하려고 하면 리스트에서 그 소속함수를 클릭하면 된다. 그 오른쪽에는 '소속함수 추가', '소속함수 제거' 버튼이 있고, 그리고 '끝내기' 버튼도 있다. 그 아래에는 현재 선택된 소속함수에 대한 설명창이 나오는데, '이름'은 소속함수의 이름(엄밀히 말하자면 퍼지 집합의 이름)이고, 종류는 소속함수의 종류로서 콤보박스를 클릭하여 여러 가지 소속함수들 중의 하나를 선택할 수 있다. 그 아래 부분에 파라미터는 소속함수의 특성을 결정하는 파라미터이다. 만약 Sugeno type 출력변수라면 그 파라미터는 식의 계수 $[c_0^l \; c_1^l \; \ldots\ldots c_n^l]$를 입력하면 된다. 소속함수의 종류 및 파라미터에 대해서는 이 튜토리얼의 마지막장의 Reference를 참고하기 바란다.

자, 이제 우리는 '온도 조절기'를 만드는 작업을 계속하자. 우선 퍼지 변수의 범위부터 결정하자. 온도가 20도~40도 사이에서 변한다고 했으므로 '값의 범위' 및 '표시 범위'를 "[20 40]"으로 입력한다.

이제부터는 소속함수를 추가해 보자. '소속함수 추가' 버튼을 누르면 빨간색 삼각형이 그래프에 그려질 것이다. 그것은 default로 만들어지는 소속함수가 삼각형 모양의 소속함수이기 때문이다. 일단, 소속함수의 이름을 정하자. 온도에 대한 퍼지 집합의 소속함수니까 알아보기 쉽도록 "low"라고 정하자. 소속함수의 종류는 그냥 "triangular"로 하고, 파라미터는 "[10 20 30]"으로 한다. 여기서 파라미터 10, 20, 30은 각각 삼각형 모양 소속함수의 꺾이는 점을 나타낸 것이다. 표시범위가 20~40 사이이므로 20보다 작은 범위는 그래프에 나타나지는 않을 것이다. 이제 새로운 소속함수를 추가하고 이름을 "mid", 파라미터는 "[20 30 40]"으로 한다. 다시 하나의 소속함수를 추가하고 이번에는 "high"라는 이름을 붙이고 파라미터는 "[30 40 50]"이라고 하자.

이제 화면은 그림 16과 같은 모양이 되었을 것이다. '끝내기' 버튼을 눌러 소속화면 편집 창을 빠져나간다. 그리고 출력 퍼지 변수 "출력변화"에 대해서도 똑같은 작업을 하도록 하자. 이번에는 출력 변수의 범위 및 표

시 범위를 “[-1 1]”로 하자. 그리고 역시 세 개의 소속함수를 추가하되, 그 이름을 “negative", "zero", "positive"로 한다. 그리고 종류 및 파라미터를 정해 주자. 삼각형 모양의 소속함수로 정해주어도 좋지만, 이번에는 조금 다른 모양의 소속함수를 설정해 주자. ”negative"는 “Z-shaped", "[-1 0]"의 파라미터로 하고, ”positive"는 “S-shaped", "[0 1]"로, ”zero"는 "Gaussian", "[0.375 0]"로 한다. Z-shaped 및 S-shaped의 파라미터는 소속함수의 꺾어지는 점을 나타내고, Gaussian소속함수의 파라미터는 “[표준편차 평균]”을 나타낸다. 왜 이런 모양의 소속함수를 정해 주느냐고 의문이 드는 사람이 있을 지 모르겠다. 그것은 특별한 까닭이 있어서가 아니고, 이 책이 튜토리얼이기 때문에 일부러 다양한 소속함수를 선보인 것 뿐이다. :)

이제 소속함수의 편집도 끝났으니, 퍼지 규칙을 정해야 할 때이다. 퍼지 규칙 편집은, 메인 윈도우로 돌아가서 ‘FCS 이름’아래에 있는 ‘퍼지 규칙’이라고 적혀져 있는 박스를 마우스로 누르면 된다. 그러면 다음 페이지의 창이 뜰 것이다.

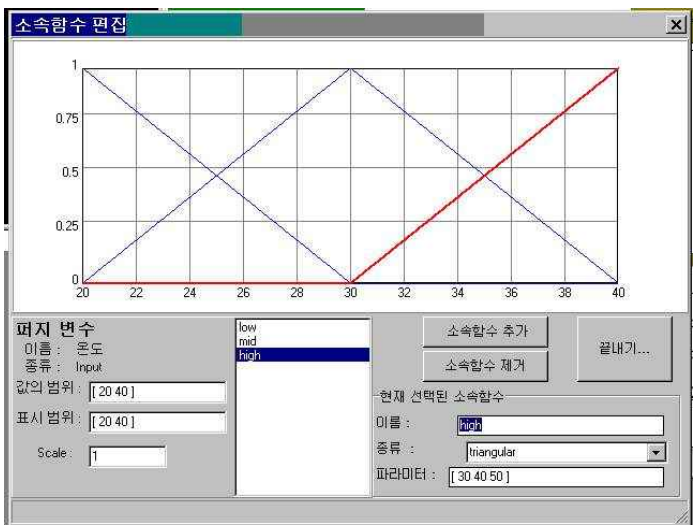


그림 16 “온도”의 소속함수 편집 창

3. 퍼지 규칙 편집 창

이번에는 아래 모양과 같은 창이 떠 있는 것이 보일 것이다. 이 '간단무료'하게 생긴 창은 소속함수 편집 때까지와는 달리 조금 어리둥절할지도 모르겠다. 당황하지 말고 일단 창에 대한 설명부터 보라.

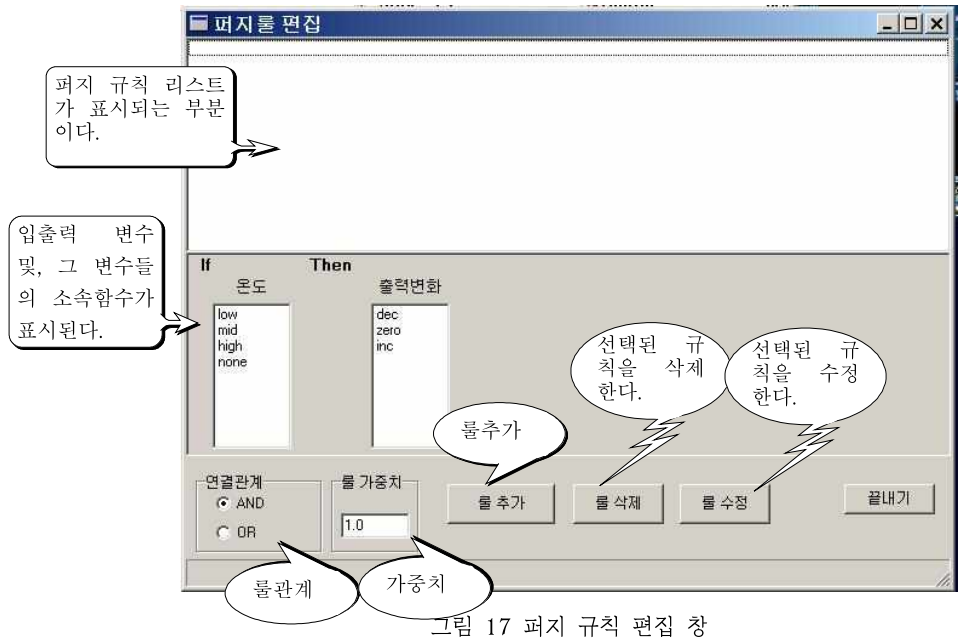


그림 17 퍼지 규칙 편집 창

우선 새로운 규칙을 추가하는 것을 보임으로써 여러 기능들을 설명해 보겠다. If라는 글자 오른쪽 아래에 보면 '온도'가 보이고 그 아래에 온도에 소속된 여러 가지 소속함수들이 보인다. IF 파트는 전건부를 나타내므로, 전건부 변수의 값을 '온도' 아래에 있는 값 중의 하나를 클릭 함으로써 선택할 수 있다. (소속함수들의 값 중 'none'은 그 값이 선택되지 않았음을 나타낸다.) 후건부도 전건부와 비슷하게 정한다. "Then" 오른쪽 아래편에 '출력변화'라는 변수 이름이 보이고, 그 아래 출력 변화의 여러 값들이 나열되어 있다. 역시 하나를 선택할 수 있다.

이제 '룰 추가'라는 버튼을 누르면 위의 리스트 박스에 새로운 룰이 표시되는 것을 볼 수 있을 것이다.

'룰 추가' 버튼 옆에 보면 '연결 관계' 와 '가중치'라는 박스가 보인다. 연결관계는 전건부의 각 조건들끼리의 연결 관계를 나타내며 'and'와 'or'의 값을 가질 수 있다. 현재는 전건부 변수가 '온도' 하나 뿐이므로 어느 값을 선택하든지 소용이 없지만, 전건부 변수가 둘 이상일 때 이 차이점을 알 수 있게 될 것이다. '가중치'는 그 규칙의 가중치를 뜻한다. 기본적으로 가중치를 1로 두지만, 경우에 따라서는 그보다 작거나 큰 값을 정할 수 있다.

짐작하겠지만 '룰 삭제' 버튼은 위쪽 룰 리스트에서 선택된 규칙을 삭제할 때 쓰인다. 그리고 '룰 수정' 버튼은 룰 리스트에서 선택된 규칙을 새로운 값들로 수정할 때 쓰인다.

규칙 추가 및 수정 때 중복되는 규칙이 있을 수도 있는데, 현재는 디자이너의 자율성을 인정한다는 차원에서 중복을 허용하고 있다. 만약 같은 규칙이 두 개가 있다면 그것은 그 규칙의 가중치가 2가 되는 것과 동일한 기능을 하게된다.

이제 여러분이 직접 규칙을 추가하도록 하자. 추가할 규칙은 다음과 같다.

if (온도 = low) then (출력변화 = inc)
if (온도 = mid) then (출력변화 = zero)
if (온도 = high) then (출력변화 = dec)

입력을 제대로 했다면, 규칙 편집 창은 그림 18과 같이 되었을 것이다. 규칙 편집 창을 닫고, 'Ctrl-S'를 눌러 우리가 구성한 제어기를 저장하자. 우리가 이렇게 만든 제어기의 테스트 결과는 simtool을 실행시켜 온도데모.blk을 시뮬레이션 시켜 보면 된다.

그림 18 퍼지 규칙 편집창 2

6. Robust Fuzzy Control Theory

6.1 Takagi-Sugeno 퍼지 모델에 근거한 성능 보장 퍼지 제어기 설계 연구

1. 서론

다양한 실제 환경에서의 실제 시스템은 보통 비선형이며, 이는 고전적인 선형 제어 이론으로는 제어하기가 거의 힘들다. 이러한 사실은 Lyapunov 이론에 근거한 direct 또는 adaptive 비선형 제어이론의 개발을 가져왔다 [1, 2, 3]. 그렇지만, 이러한 제어 이론은 시스템의 구조 및 조건에 너무 많이 의존하기 때문에 다양한 종류의 비선형 시스템에 대한 보편적인 접근을 거의 불가능하게 한다.

따라서 이러한 어려움을 극복하기 위한 세 가지 시도가 있었다. 첫 번째는 제어 선형화기법(feedback linearization method)을 사용하여 비선형 시스템을 선형 시스템으로 바꾸는 것인데 이 방법으로는 여전히 그러한 transform 들을 찾기가 너무 어렵다 [4, 5, 6, 7, 8]. 두 번째 방법은 동작점 근처에서 비선형 시스템을 선형적인 모델로 표현하여 비선형 시스템과 선형적으로 표현된 시스템 사이의 model mismatch에 대해서 robust한 선형시스템 이론을 적용하는 것이다 [9, 10]. 이 방법은 현재 세계에서 가장 많이 연구되는 방법이며, 두 단계를 필요로 한다. 선형적으로 모델을 나타내는 것과 robust한 선형 제어 이론을 적용하는 것이다. 그렇지만 이 방법 역시 원래의 비선형 시스템과 근접한 선형 모델을 제시하기가 어렵다는 단점이 있다. 따라서 원하는 performance를 가지는 global 시스템을 제어하는 데에 실패하기도 한다. 마지막 방법은 다양한 지능 제어 방법들에 근거한 것인데, 본 연구는 그 방법들 중에서 퍼지 제어 방법에 초점을 둘 것이다.

최근 퍼지 제어에 대한 많은 연구가 발표되었는데, 이는 이 방법을 사용하여 비선형 시스템을 제어하기가 개념적으로 쉽기 때문이며, 또한 다양한 방면에서 이 제어 방법을 이용한 실제 문제에의 많은 성공적인 적용 사례들이 있었다. 퍼지 제어에서 떠오르는 가장 흥미로우면서도 가장

중요한 이슈들 중의 하나는 제어기를 어떻게 하면 조직적으로 설계할 수 있는가 하는 문제이다. 그러기 위해서는, 보통 비선형 시스템을 Takagi-Sugeno 퍼지 모델링 방법을 사용하여 표현하는데, 여기서의 중요한 특징은 멤버쉽 함수들에 대한 grades 들에 대한 하위 선형 시스템들의 가중치된 합으로 선형 시스템을 근사화시킴으로서 선형 시스템 이론들을 차용한 제어기를 구현할 수 있다는 점이다. 이러한 시스템들에 대한 stability analysis에 근거하여 [11, 12, 13] state-feedback 제어 방법이 제안되었는데 [14, 15, 16], 이때의 state-feedback stabilizer에 대한 조건들을 선형 matrix 부등식을 이용한 방법과 폐루프(교차)- 하위 시스템들의 pole들을 특정한 영역에 대하여 할당하는 방법을 사용하여 증명하였다. 이러한 방법이 다른 방법들에 비해서 훨씬 간편해진 것은 사실이지만, 원하는 시스템 성능에 대한 적당한 pole들의 위치를 예측하기는 여전히 어렵다. 본 연구에서는 pole들의 위치를 예측하는 방법이 아닌 cost의 개념을 도입한 더욱 체계적인 방법을 도입할 것이다.

Cost로써는 Takagi-Sugeno 퍼지 모델에 근거한 모든 가능한 membership grade 들에 대하여 입력값의 상한값과 LQ performance라고 불리우는 출력 에너지를 고려할 것이다. 기본적인 두 가지 종류의 제어기를 제안할 것인데, 첫 번째는 guaranteed LQ fuzzy controller이며 두 번째는 guaranteed LQ static controller이다. 전자는 후자의 경우보다 훨씬 신축성이 크며, 후자의 경우는 단순하게 feedback 루프에서 상수값의 gain을 구상하면 되기 때문에 매우 간단한 장점이 있다.

본 연구를 위하여 다음의 비선형 시스템을 고려한다.

$$\dot{x}(t) = f(x(t), u(t)), \tag{1}$$

여기서 입력값은 다음과 같은 비선형 state-feedback의 형태로 주어져 있다.

$$u(t) = g(x(t)). \tag{2}$$

Takagi-Sugeno 퍼지 시스템 [11, 12]은 그 각각이 하위 선형 시스템과의 입출력 관계들을 나타내는 퍼지 IF-THEN rule 들에 의해서 표현된다. i의 범위가 $1 \le i \le r$ 일 때 퍼지 시스템에서의 i_{th} rule 은 다음과 같다. :

The i_{th} Model Rule: IF $x_j(t) \in M_{ij}$ for $1 \le j \le n$,

THEN $\dot{x}_i(t) = A_i x(t) + B_i u(t)$,

이때 $A_i \in R^{n \times n}$ 이고, M_{ij} 는 퍼지 set 이며, $\dot{x}_i(t)$는 IF-THEN rule의 i_{th} 출력이며, $x(t)$ 와 $u(t)$ 는 각각 다음과 같다.

$$x(t) \equiv [\, x_1^T(t) \ \cdots \ x_n^T(t) \,]^T,$$

$$u(t) \equiv [\, u_1^T(t) \ \cdots \ u_m^T(t) \,]^T.$$

결과로 얻어지는 퍼지 시스템의 출력은 다음과 같이 추론될 수 있다.

$$\begin{aligned} \dot{x}(t) = & \sum_{i=1}^{r} \hat{w}_i(t)\, \dot{x}_i(t) = \\ & \left\{ \sum_{i=1}^{r} \hat{w}_i(t) A_i \right\} x(t) + \left\{ \sum_{i=1}^{r} \hat{w}_i(t) B_i \right\} u(t), \end{aligned} \tag{3}$$

여기서 M_{ij}에서의 $x_j(t)$의 membership의 grade에 대해서 $\hat{w}_i(t)$ 와 $w_i(t)$는 ,

$$\hat{w}_i(t) \equiv \frac{w_i(t)}{\sum_{l=1}^{r} w_l(t)}, \quad w_i(t) \equiv \prod_{j=1}^{n} M_{ij}(x_j(t)).$$

이다. 또한 항상 $\sum_{i=1}^{r} \hat{w}_i(t) = 1$ 이 성립한다. Fuzzy set 들이 정규(normal)화 되었을때 임의의 $x(t)$에 대해서 $\hat{w}_i(t)$는 ,

$$0 \leq \hat{w}_i(t) \leq 1.$$

이다. 이 경우, 시스템의 안정성에 대한 다음의 보조정리가 성립한다.

보조정리 1.1 (3)에서의 개루프 시스템 (즉, $u(t) \equiv 0$) 이 asymptotically stable in the large이기 위한 필요충분 조건은 공통의 positive definite matrix P 가 존재하여 아래 (4)식의 조건을 만족시키는 것이다.

$$A_i^T P + P A_i < 0 \text{ for all } 1 \leq i \leq r. \tag{4}$$

만약 제어기를 고려하고자 한다면, 즉 (3)에서의 폐루프 시스템을 생각할 때, 안정성 해석은 더욱 더 복잡해진다. 퍼지 제어는 퍼지 시스템 (3)과 함께 다음과 같이 동일한 퍼지 sets를 갖는다.

The i_{th} Control Rule: IF $x_j(t) \in M_{ij}$ for $1 \leq j \leq n$, THEN $u_i(t) = K_i x(t)$,

또 그 결과는 다음과 같다.

$$u(t) = \sum_{l=1}^{r} \hat{w}_l(t)\, u^l(t) = \left\{ \sum_{l=1}^{r} \hat{w}_l(t) K_l \right\} x(t). \tag{5}$$

따라서 폐루프 시스템은 다음과 같이 쓸 수 있다.

$$\dot{x}(t) = \sum_{i=1}^{r} \hat{w}_i(t)\, \dot{x}_i(t) = \left\{ \sum_{i=1}^{r} \sum_{j=1}^{r} \hat{w}_i(t)\ \hat{w}_j(t)\{A_i + B_i K_j\} \right\} x(t), \tag{6}$$

또한 $\sum_{i=1}^{r} \sum_{j=1}^{r} \hat{w}_i(t)\ \hat{w}_j(t) = 1$ 가 성립하고, 임의의 $x(t)$에 대해서,

$$0 \le \hat{w}_i(t)\ \hat{w}_j(t) \le 1,$$

이므로, 정리 1.1 에서의 결과를 적용하여 (3)과 (5)의 폐루프 시스템은 만약 아래의 조건식 (7)을 만족시키는 공통의 positive definite matrix P가 존재할 경우 asymptotically stable in the large 라고 말할 수 있다.

$$(A_i + B_i K_j)^T P + P(A_i + B_i K_j) < 0 \quad \text{for all } 1 \le i, j \le r. \tag{7}$$

참고 사항: 여기서는 단순하게 서로 연결된 항들에 대해서만 확인하고자 한다. 다시 말해서, 만약 모든 $t \ge 0$에 대해서 $\hat{w}_i(t)\ \hat{w}_j(t) \equiv 0$ 이라면, ij_{th} 조합에 대해서는 고려할 필요가 없다.

그러나, 사실 Tanaka와 Sugeno [12]는 다음과 같이 더욱 flexible 한 충분 조건들을 제공하였다.

정리 1.2 (3)과 (5)에서의 폐루프 시스템은 만약 다음의 조건식 (8)을 만족시키는 공통의 positive definite matrix P가 존재한다면 asymptotically stable in the large를 만족한다.

$$G_{ij}^T P + P G_{ij} < 0 \quad \text{for all } 1 \le i \le j \le r, \tag{8}$$

여기서 G_{ij}은 다음과 같이 정의된다.

$$G_{il} \equiv \frac{(A_i + B_i K_l) + (A_l + B_l K_i)}{2}. \quad (9)$$

참고 사항: 조건식 (7)은 조건식 (8)을 항상 뜻한다. 이 결과는 많은 문제들에서 성공적으로 쓰여져 왔다 [12, 13, 14, 15, 16]. 다음의 section에서는 이 결과를 확장하여 guaranteed cost fuzzy controller를 소개할 것이며, 이는 제어기를 체계적으로 설계할 수 있게 끔 한다.

2. 성능 보장 퍼지 제어기 설계

2.1 개념 및 제어기

아래의 식 (10)과 같이 정의되는 상태 및 입력의 식과 연관된 LQ cost의 상한값을 최소화시키는 guaranteed cost fuzzy controllers를 소개할 것이다.

$$\min_{K_l} \quad \max_{\widehat{w}_i(t) \in W} \int_0^\infty \{x^T(t) C^T Q C x(t) + u^T(t) R u(t)\} dt, \quad (10)$$

여기서, $Q > 0, R > 0$ 이며 W 는 membership 함수들의 모든 admissible grade들의 set으로 정의된다. 이러한 기준에 대한 의미를 설명하기 위해서, 먼저 모든 K_l gain들을 한 개의 이득값 K로 고정시켜야 한다. 이 경우, LQ cost는 membership 함수들의 grades $w_i(t)$의 함수가 될 것이다. 만약 $w_i(t)$의 정확한 궤적을 알 수 있다면, 정확한 LQ cost의 값을 계산할 수 있을 것이다. 그러나 그것은 실제로 매우 복잡하며 심지어는 불가능하기도 하다. 따라서, 모든 admissible grades $w_i(t)$의 조건 아래에서의 LQ cost의 상한값이 한 가지 타당한 후보가 되며, 그 까닭으로 이는 guaranteed cost라고 불리운다. 다음의 theorem은 membership 함수들의 모든 admissible grades $w_i(t)$에 대해서 guaranteed LQ cost를 최소화시키는 최적 제어기를 어떠한

방법으로 찾을 수 있는지를 요약한다.

정리 2.1 (Guaranted Cost Fuzzy Controller의 설계) (3)과 (5)에서의 퍼루프 시스템은 만약 모든 $0 \le i \le l \le r$ 인 i, l, r 에 대해서 $\widehat{w}_i(t)\ \widehat{w}_l(t) \neq 0$ 일 때 $t \ge 0$에 대해서 다음의 식 (11)을 만족하는 weighed gain matrices F_l과 공통의 positive definite matrix Z가 존재하면 “asymptotically stable in the large"이다.

$$0 > \begin{bmatrix} ZA_i^T + ZA_l^T + F_l^T B_i^T + F_i^T B_l^T + A_i Z + A_l Z + B_i F_l + B_l F_i & F_l^T + F_i^T & 2ZC^T \\ F_l + F_i & -2R^{-1} & 0 \\ 2CZ & 0 & -2Q^{-1} \end{bmatrix} \quad (11)$$

이때 각각의 K_l 은 다음과 같이 정의되며,

$$K_l \equiv F_l Z^{-1} \quad (12)$$

cost (10)은 $x^T(0) Z^{-1} x(0)$에 의해서 bound된다. 조건식 (11)은 변수 Z와 F_l에 대해서 선형이므로, 최적의 guaranteed cost는 다음과 같은 linear programming에 의해서 얻어질 수 있다 : (11)이 성립하고 다음의 식 (13)을 만족시키는 γ를 최적화한다.

$$0 \le \begin{bmatrix} \gamma & x^T(0) \\ x(0) & Z \end{bmatrix}. \quad (13)$$

증명: 다음의 어떤 positive definite matrix P에 관한 Lyapunov 방정식을 고려해 보자. 이 방정식은 cost(10)의 상위제한(upper bound)이다.

$$\int_t^\infty \{ x^T(t) C^T Q C x(t) + u^T(t) R u(t) \} dt < x^T(t) P x(t)$$

만약 시스템이 안정하다면 양변은 시간이 무한대로 감에 따라 0으로

된다. 그러므로 양변을 초기시간 t에대해서 미분하면 모든 $x(t)$에대하여 다음 식을 만족해야한다.

$$0 > \dot{x}^T(t)Px(t) + x^T(t)P\dot{x}(t) + x^T(t)C^TQCx(t) + u^T(t)Ru(t),$$

이 식은 다음 식을 낳는다.

$$0 > \left\{\sum_{i=1}^{r}\sum_{l=1}^{r}\hat{w}_i(t)\ \hat{w}_l(t)\{A_i + B_iK_l\}\right\}^T P + P\left\{\sum_{i=1}^{r}\sum_{l=1}^{r}\hat{w}_i(t)\ \hat{w}_l(t)\{A_i + B_iK_l\}\right\} +$$

$$C^TQC + \left\{\sum_{i=1}^{r}\hat{w}_l(t)K_l\right\}^T R\left\{\sum_{i=1}^{r}\hat{w}_l(t)K_l\right\},$$

또한, $\sum_{i=1}^{r}\hat{w}_l(t) = 1$ 이라는 성질을 이용하면 위의 조건은 아래와 같이 정의 할 수 있고,

$$0 > \sum_{i=1}^{r}\sum_{l=1}^{r}\hat{w}_i(t)\ \hat{w}_l(t)\begin{bmatrix} (A_i + B_iK_l)^TP + P(A_i + B_iK_l) + C^TQC & C_l^T \\ K_l & -R^{-1}\end{bmatrix}.$$

색인(index) i와 l에대해서 행렬 요소들을 대칭되게 만들어 주기 위해서 다음과 같은 기교를 이용하여 위 식을 아래와 같이 재구성 할 수 있다.

$$0 > \sum_{i=1}^{r}\sum_{l=1}^{r}\hat{w}_i(t)\ \hat{w}_l(t)\begin{bmatrix} G_{il}^TP + PG_{il} + C^TQC & K_l^T \\ K_l & -R^{-1}\end{bmatrix}$$

이제 다음을 정의한다: $Z \equiv P^{-1}$, $F_i \equiv K_iP^{-1}$. 그러면, 앞 조건은 다음과 같이 또 다시 재구성된다.

$$0 > \sum_{i=1}^{r}\sum_{l=1}^{r}\hat{w}_i(t)\ \hat{w}_l(t)\begin{bmatrix} (1,1) & F_l^T + F_i^T & 2ZC^T \\ F_l + F_i & -2R^{-1} & 0 \\ 2CZ & 0 & -2Q^{-1}\end{bmatrix},$$

단

$$(1,1) \equiv ZA_i^T + ZA_l^T + F_l^TB_i^T + F_i^TB_l^T + A_iZ + A_lZ + B_iF_l + B_lF_i.$$

따라서 (11)이 성립하면 cost 함수 (10)은

$x^T(0)P(0) = x^T(0)Z^{-1}x(0)$, 에 의해 제한값을 가진다. 다시 말해, 다음 식이 만족된다.

$$\int_0^\infty \{x^T(t)C^TQCx(t) + u^T(t)Ru(t)\}dt < x^T(0)Z^{-1}x(0).$$

어떠한 (11)과 (13)을 낳는 구현가능한 γ, Z, F_l 은 어떠한 소속 함수에 대해서도 다음을 만족하기 때문에, $x^T(0)Z^{-1}x(0)$를 최소화하는데 (13)을 사용한다.

$$\int_0^\infty \{x^T(t)C^TQCx(t) + u^T(t)Ru(t)\}dt < x^T(0)Z^{-1}x(0) \le \gamma. \quad \blacksquare$$

참고 사항: Matrix 부등식 (11)과 (13)은 $1 \le l \le r$인 모든 변수 Z, γ, F_l에 대해서 선형이다. 따라서 convex optimization 방법을 써서 한정된 계산량을 필요로 하는 global solution을 찾을 수 있으며, 추가된 선형 제한 조건식들을 쉽게 다룰 수 있다.

어떠한 feasible matrices F_l과 (11)의 결과를 갖는 positive definite matrix Z 라도 임의의 membership functions 에 대해서 다음의 부등식을 만족시킨다는 점을 주목한다.

$$\int_0^\infty \{x^T(t)C^TQCx(t) + u^T(t)Ru(t)\}dt < x^T(0)Z^{-1}x(0).$$

즉, 그러한 Z와 F_l를 사용함으로 해서 (12)로부터 선택된 어떠한 gain 이라도 전체 시스템을 안정화시킬 것이다. (11)과 (13)을 만족시키는 최소의 γ 값을 찾으면 최적의 제어기를 얻을 수 있다. 여기서 새로운 방법의 취약점은 최적의 이득값은 상태의 초기값들에 의존한다는 것이다. 이 말은 이득값이 초기값들에 따라서 변화된다는 것을 뜻한다. 이러한

단점을 없애기 위해서, 다음의 subsection에서 또 다른 제어기를 제안할 것이다.

2.2 초기치와 무관한 제어기 구성

State-feedback을 이용할 수 있으므로 state 들의 초기값은 측정될 수 있으며, 이는 정리 2.1을 통해서 얻어진 최적의 제어기를 이용할 수 있음을 뜻한다. 그러나 초기값들에 대한 어떠한 정보도 없이 gain 값을 미리 결정해야만 하는 경우, 즉 계산에 필요한 시간이 전혀 없을 경우가 있을 수도 있을 것이다. 이러한 상황을 다루려면, 다음의 criterion이 필요하다.

$$\min_{K_l} \max_{x(0)\neq 0} \max_{\hat{w}_i(t)\in W} \int_0^\infty \{x^T(t)C^TQCx(t) + u^T(t)Ru(t)\}dt \,/\, x^T(0)x(0), \tag{14}$$

여기서 $Q>0$ 이고 $R>0$이다. 이러한 criterion의 의미는 모든 admissible membership grades $w_i(t)$에 대해서 뿐만 아니라 모든 admissible initial states $x(0)$에 대해서도 robust한 fuzzy controller를 설계해야 한다는 것이다. 다음의 corollary 는 그러한 제어기를 어떠한 방법으로 설계할 수 있는지를 제시해 준다.

유사정리 2.1 (Independent of Initial States Controller 설계) Criterion (14)를 만족시키는 최적 제어기는 다음과 같이 구할 수 있다: 조건식 (11)과 다음의 (15)를 만족하는 γ를 최소화한다.

$$0\le\begin{bmatrix} \gamma I & I \\ I & Z \end{bmatrix}. \tag{15}$$

이 경우, 최적의 gain은 $K_l = F_l Z^{-1}$에 의해서 계산된다.

증명 : 조건식 (11)을 만족시키는 임의의 feasible solution Z, F_l에 대해서, 그리고 임의의 가능한 membership functions에 대해서, 이 부등식은 다음을 만족시킨다.

$$\max_{\hat{\omega}_i(t) \in W} \int_0^\infty \{x^T(t) C^T Q C x(t) + u^T(t) R u(t)\} dt < x^T(0) Z^{-1} x(0).$$

이 식으로부터 모든 0이 아닌 $x(0)$ 값에 대해서 다음의 관계식이 나온다.

$$\max_{\hat{\omega}_i(t) \in W} \int_0^\infty \{x^T(t) C^T Q C x(t) + u^T(t) R u(t)\} dt / x^T(0) x(0) < \sigma_{\max}(Z^{-1})$$

여기서 $\sigma_{\max}(Z^{-1})$는 Z^{-1}의 maximal singular value를 뜻한다. 따라서, 조건식 (15)를 만족시키는 γ를 최소화함으로 해서 $\sigma_{\max}(Z^{-1})$를 최소화시킬 수 있다. ∎

이 제어 법칙으로부터 초기 상태값들에 무관한 gain을 얻는다.

3. 성능 보장 고정 이득 제어기 설계

3.1 개념 및 제어기

퍼지 제어기들의 모양을 간단히 하는 방법중의 하나는 K_i의 구조에 어떠한 제한을 두는 것이다. 예를 들면, 규칙들을 비슷한 이득모양(gain shape)을 가지는 계층들로 묶음 짓는 것이 가능하다. 극단적인 예로서 우리는 K_i대신 단순한 하나의 이득 K를 사용하길 원할 수도 있는데, 이 경우는 비선형적 퍼지 제어기보다는 정적 제어기(a static controller)를 사용하려는 경우이다. 다시 말해서, 우리는 제어기 형태를 정적 상태-궤환(state-feedback)을

$$u(t) = Kx(t)$$

와 같은 모양으로 제한한다. 이렇게 되면 더 이상 퍼지(비선형 혹은 시변(time-varing)) 제어기가 아니고 선형 시불변(linear

time-invariant)제어기가 되는 것이다. 이것은 모든 $0 \le l, i \le r$에 대해서 $K_l = K_i$인 특별한 경우의 퍼지 제어로 생각할 수 있다. 그러나, 제어기를 설계하는 기본적인 아이디어들은 여전히 비선형 시스템(1)을 퍼지모델(3)의 형태로 추론해 내는 데 있다. 다음 정리는 정리 2.1의 약간 변형된 모양이다.

정리3.1 (Guaranted Cost Static Controller 설계)

(3)과(5)에서의 닫힌 고리 시스템(Closed-loop system)은 만약 $0 \le i \le r$인 모든 i에 대해서

$$0 > \begin{bmatrix} ZA_i^T + F^T B_i^T + A_i Z + B_i F & F & ZC^T \\ F & R^{-1} & 0 \\ CZ & 0 & Q^{-1} \end{bmatrix} \tag{21}$$

단 K는 다음과같이 정의되고

$$K = FZ^{-1} \tag{22}$$

cost 함수(10)은 $x^T(0) Z^{-1} x(0)$에 제한된 공통의 positive definite matrix Z와 공통의 행렬 F가 존재한다면 전 영역에서 점근적으로 안정(asymtotically stable in the large)하다.

조건 (21)은 변수 Z와 F에 대해서 선형적이기 때문에, 최적의 보장된 cost는 선형 프로그램으로 구할 수 있다. 이때 프로그램은 조건 (21)과 다음을 만족하는 γ를 최소화 하는 프로그램이다.

$$. 0 \le \begin{bmatrix} \gamma & x^T(0) \\ x(0) & Z \end{bmatrix}$$

증명: 생략

식(21)의 조건은 단일 이득(Single gain)을 여러개의 시스템을 안정화 시키는데 이용하려고 할 때 나타난다는 것을 언급한다. 조건 (11)의

$r(r+1)/2$와 비교해서, 여기서는 단지 r개의 조건밖에 없다. 그러므로, 이론적으로, 이 방법은 덜 유연하다. 그러나, 이 방법의 주된 장점은 제어기가 아주 단순한 모양, 다시 말해서 제어 동작에서 퍼지 룰을 사용하지 않는다는 점이다.

3.2 초기치와 무관한 제어기 설계

앞에서 언급했듯이, 초기상태를 측정하고 나서 계산할 시간이 전혀 없는 경우가 있다. 이것은 초기 상태에 대한 어떠한 정보 없이 이득(gain)을 미리 결정해야 한다는 것을 의미한다. 이와 같은 상황을 처리하기 위해, 기준(criterion) (10)대신에 (14)가 고려되었다. 단지 차이점은 모든 K_l가 이제 모두 동일하다는 점이다. 그러면 유사정리 (corollary) 2.1은 다음과 같이 바뀐다.

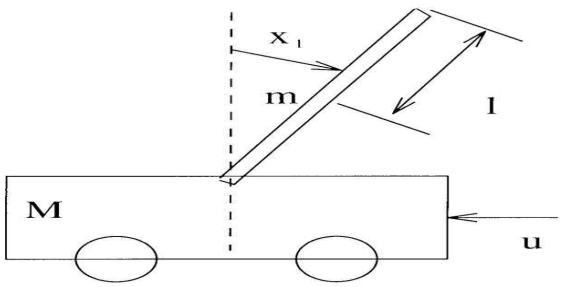
유사정리 3.1 (Independent of Initial States Controller 설계) 모든 K_l이 단일 이득 K로 제한되었을 때, 기준 (14)를 만족시키는 최적 제어기는 ; (21)과 (15)에 대해서 γ를 최소화 하는 것으로 얻어질 수 있다. 이 경우 최적 이득은 $K = FZ^{-1}$로 계산된다.

4. 실험 및 고찰 (차위의 역진자.(Inverted Pendulum on a Car))

다음으로 Guaranteed cost 퍼지 제어기의 성능을 알아보기 위해서, Wang et al [16]에 의해서 쓰인 방법인 cart 위에 설치된 inverted pendulum의 균형을 맞추는 문제를 생각해 본다 (그림 4.1.1을 참고한다.) Pendulum의 운동 방정식은 [17] 아래와 같다.

$$\dot{x}_1 = x_2, \tag{16}$$

$$\dot{x}_2 = \frac{g\sin(x_1) - aml x_2^2 \sin(2x_1)/2 - a\cos(x_1)u}{4l/3 - aml\cos^2(x_1)}, \qquad a \equiv \frac{1}{m+M}, \tag{17}$$



(그림 4.1.1 : Inverted Pendulum on a Car 모델)

이때 x_1과 x_2는 각각 라디안으로 표시된 각도와 수직면으로부터의 pendulum의 각속도를 말하며, $g=9.8m/s^2$는 중력가속도 상수, m은 pendulum의 질량, M은 cart의 질량, $2l$은 pendulum의 길이, u는 뉴턴의 단위로 표시된 cart에 가해진 힘을 말한다. Wang et al [16]에 의해서 사용된 방법인 pole-placement 기법에 의한 결과와 비교해 보기 위해서, 거기서 사용했던 것과 같은 값들을 다음과 같이 선택한다. :

$$m=2.0kg,\quad M=8.0kg,\quad l=0.5m.$$

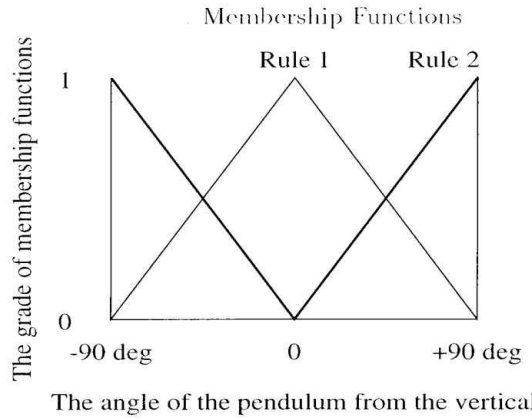
또한 마찬가지로 그들의 결과와 성능을 비교하기 위해서 Wang et al [16]과 똑같은 fuzzy rule을 이용할 것이다. 초기 상태는 다음과 같다고 가정한다.

$$x_1(0)\in(-\pi/2,\pi/2),\quad x_2(0)=0.$$

Takagi-Sugeno 퍼지 모델은 다음의 두 rule 들에 의해서 구성된다. :

Rule 1: IF x_1 is about 0, THEN $\dot{x}^1=A_1x+B_1u.$

Rule 2: IF x_1 is about $\pm\pi/2$, THEN $\dot{x}^2=A_2x+B_2u.$



(그림 4.1.2 : Membership 함수의 정의)

이때, matrix A_i와 B_i는 다음과 같이 정의된다.

$$A_1 \equiv \begin{bmatrix} 0 & 1 \\ \frac{g}{4l/3 - aml} & 0 \end{bmatrix}, \quad B_1 \equiv \begin{bmatrix} 0 \\ -\frac{a}{4l/3 - aml} \end{bmatrix},$$

$$A_2 \equiv \begin{bmatrix} 0 & 1 \\ \frac{2g}{\pi(4l/3 - aml\cos^2(88^\circ))} & 0 \end{bmatrix},$$

$$B_2 \equiv \begin{bmatrix} 0 \\ -\frac{a\cos(88^\circ)}{4l/3 - aml\cos^2(88^\circ)} \end{bmatrix}.$$

Rule 1과 2에 대한 membership functions는 그림 2와 같다. 이때의 설계 rule은 guaranteed cost control 방법에 근거할 것이다. 여기서, $C = [1 \ 0]$, $Q = 1$이며, $R = 0$이다. 이는 정리 2.1에서의 필요조건인 $R > 0$에 위배되는 것처럼 보인다. 그러나 단순하게 선형 matrix 부등식 (11)을 아래의 형태로 변화시킨다.

$$0 > \begin{bmatrix} ZA_i^T + ZA_l^T + F_l^T B_i^T + F_i^T B_l^T + A_i Z + A_l Z + B_i F_l + B_l F_i & 2ZC^T \\ 2CZ & -2Q^{-1} \end{bmatrix}.$$

Matlab을 이용하고 Boyd와 Wu의 sdp solver [18]을 이용해서, 다음과 같은 각각의 서로 다른 초기 상태에 따른 정리 2.1에서 제안된 최적의 gains K_1과 K_2를 구한다. :

$$[K_1|K_2] = \begin{bmatrix} 294.58 & 32.10 & 6726.71 & 1101.11 \\ 294.33 & 32.08 & 6719.04 & 1100.50 \\ 293.92 & 32.05 & 6706.21 & 1099.48 \\ 293.10 & 31.99 & 6680.92 & 1097.47 \\ 292.58 & 31.95 & 6664.70 & 1096.18 \\ 291.98 & 31.91 & 6646.03 & 1094.69 \\ 291.72 & 31.89 & 6637.87 & 1094.04 \end{bmatrix} \quad \text{when}$$

$$x_{1(0)} = \begin{bmatrix} 15^\circ & (0.2618) \\ 30^\circ & (0.5236) \\ 45^\circ & (0.7854) \\ 65^\circ & (1.1345) \\ 75^\circ & (1.3090) \\ 85^\circ & (1.4835) \\ 89^\circ & (1.5533) \end{bmatrix}$$

(18)

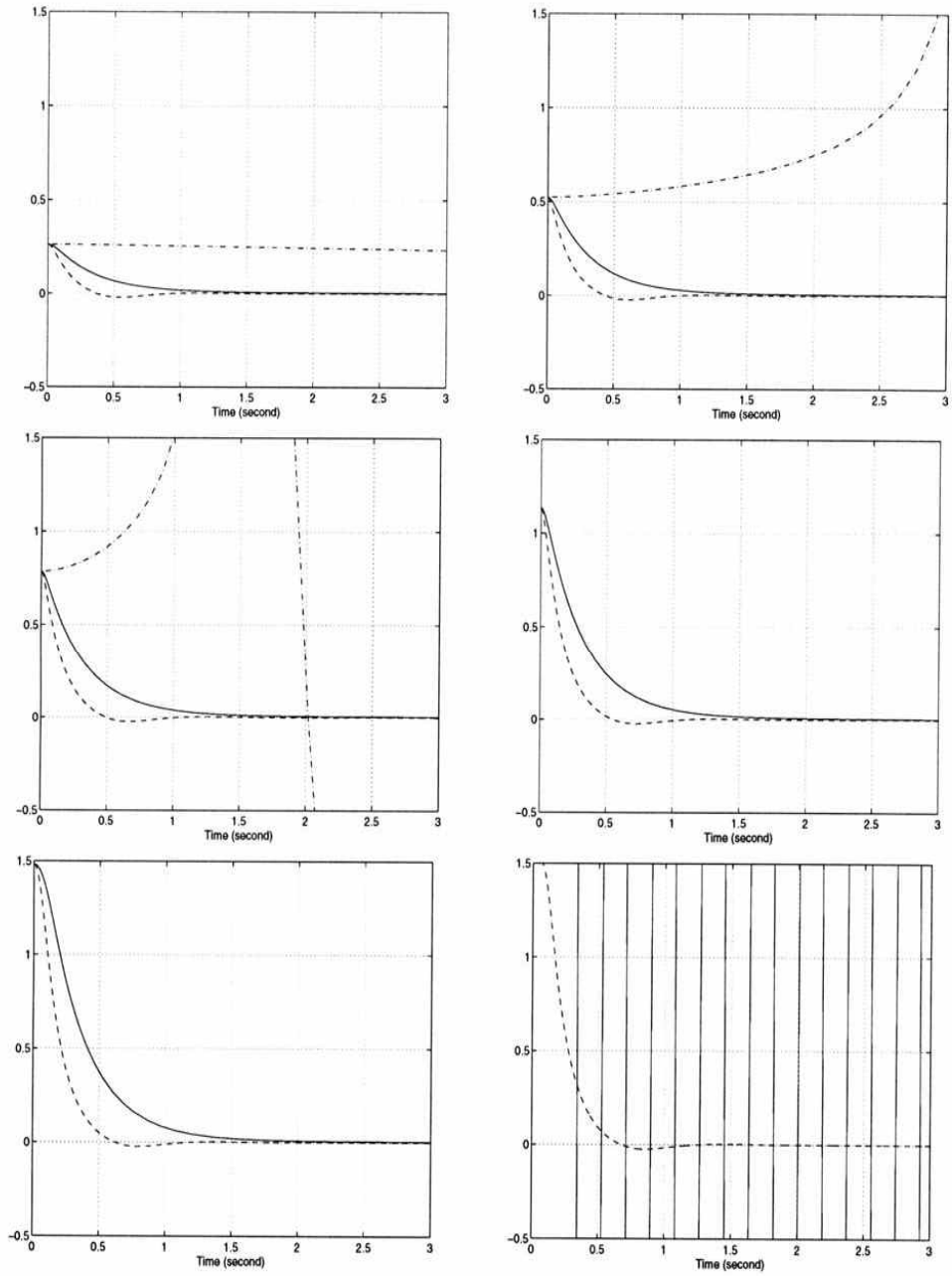
이러한 gains 값에 근거한 시뮬레이션 결과를 Wang et al [16]에 의해서 보여진 두 개의 다른 시뮬레이션 결과들과 비교할 것이다. 여기서 다음의 두 개의 경우들을 고려한다.

- Case 1: Static control ($u = \overline{K}_1 x$).
- Case 2: Fuzzy control by Wang et al [16] ($u^1 = \overline{K}_1 x$ and $u^2 = \overline{K}_2 x$)

여기서 각각의 폐루프 sub-linear 시스템의 pole 들을 -2에 위치시키는 gains는 다음과 같다.

$$[\overline{K}_1 | \overline{K}_2] = [120.67 \quad 22.67 | 2551.60 \quad 764.00] .$$

(19)



이러한 gains 값들에 근거하여 matlab simulink tool을 이용한 3초 동안의 실험을 보인다.

(그림 4.1.3 : 실선: 극 배치 Fuzzy 제어기, 점선: 성능 보장 Fuzzy 제

어기, 실선+점선: 극 배치 고정 제어기 x1(0)=15, 30, 45, 65, 85 and 89 (deg))

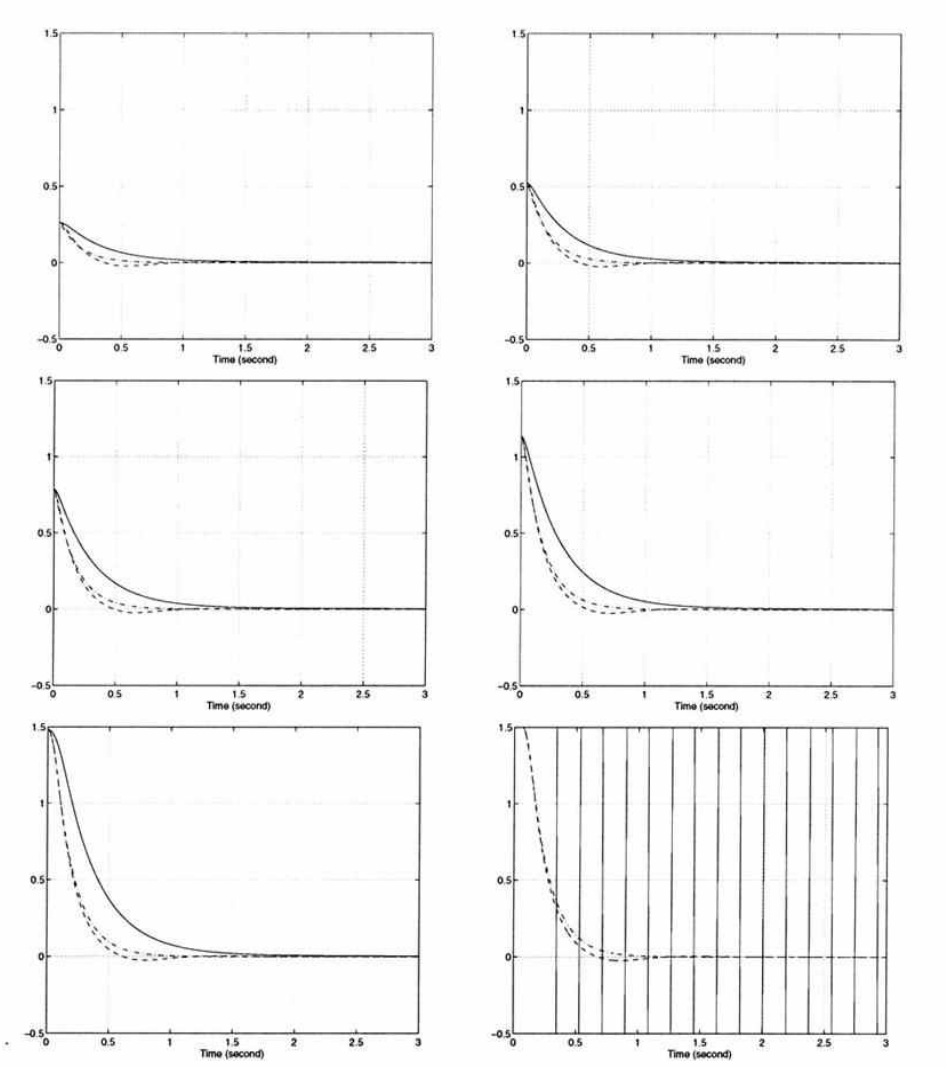
그림 4.1.3에 의해서 제시된 $x_1(t)$의 각도를 나타내는 시뮬레이션 결과는, 이 state-feedback controller는 $x_1(0)$가 $\pm 15^\circ$ 의 밖에 위치할 때에는 stabilize시킬 수 없음을 보여준다. 그림에서의 굵은 실선과 빗금친 선들은 각각 Wang et al [16]의 결과와 여기서 얻은 결과를 의미한다. Wang et al [16]에 의해서 사용된 방법은 부-시스템의 poles를 -2에 위치시키며, $x_1(0)$가 $\pm 88^\circ$ 의 사이에 위치한 거의 대부분의 경우에 대해서 stabilize시킨다. 이러한 pole-assignment 방법은 stability를 제외한 시스템 성능과는 아무런 직접적인 관계를 가지고 있지 않다. 그러나 여기서 얻은 결과는 $x_1(t)$의 전체 에너지에 초점을 맞추었기 때문에 $x_1(t)$는 Wang et al [16]에 의한 결과들과 비교하여 볼 때에 매우 빨리 0 으로 수렴함을 알 수 있다. 게다가 $x_1(0)$이 $\pm 89^\circ$ 이내의 범위에 있는 모든 경우에 대해서 stabilize시킴을 볼 수 있다. 한 가지 재미있는 사실은 guaranteed cost control 방법에 의해서 생성된 궤적에서 settling time을 감소시키는 overshoot를 확인할 수 있다는 것이다.

두 번째 모의 실험 결과는 유사정리 2.1에서 제안된 제어기에 대해 적용하여 얻은 것이다. 유사정리 2.1에서의 조건식들을 풀면 다음의 최적화된 gain을 얻을 수 있다.

$$[K_1|K_2] = [296.76 \;\; 32.13|6744.45 \;\; 1101.89] . \tag{20}$$

시뮬레이션 결과는 그림 4.1.3의 빗금친 선들과 거의 동일한 결과를 보인다.

두 번째 모의 실험은 보장된 LQ(guaranteed LQ) 라 불리는 새로운 개념이 제어기를 설계 K_1하는데 어떻게 작용하는 지 보여줄 것이다.



(18)에서의 이득들(gains)로부터, 그 이득들 and K_2가 비슷한 모양, 즉, 이득들이 비슷한 방향을 따라 위치한다는 사실을 계산해낼 수 있다. 따라서 두 개의 이득 K_1과 K_2대신 단일 이득 K를 사용하기를 원할

(그림 4.1.4 : 실선: 극 배치 Fuzzy 제어기, 점선: 성능 보장 Fuzzy 제어기, 실선+점선: 성능 보장 고정 제어기 x1(0)=15, 30, 45, 65, 85 and 89 (deg)일때)

수도 있다. matlab과 Boyd and Wu'sdp solver를 이용하여 정리 3.1에서의 최적 이득 K를 다른 초기 상태들에의해서 계산하는 데, 다음과 같다.

$$K=\begin{bmatrix} 6473.36 & 1099.34 \\ 6465.88 & 1098.72 \\ 6453.36 & 1097.69 \\ 6428.69 & 1095.65 \\ 6412.85 & 1094.34 \\ 6394.29 & 1092.84 \\ 6386.67 & 1091.73 \end{bmatrix} \quad \text{when}$$

$$x_1(0)=\begin{Bmatrix} 15°(0.2618) \\ 30°(0.5236) \\ 45°(0.7854) \\ 65°(1.1345) \\ 75°(1.6090) \\ 85°(1.4835) \\ 89°(1.5533) \end{Bmatrix}$$

그림 4.1.4는 그림 4.1.3과 위의 값들을 Wang et al[16]에 의해 구해진 정적되먹임이득(static feedback gain) K_1과 바꾼 경우와 동일한 결과를 보인다. 또한 유사정리 3.1에 의해 이득은 다음과 같이 계산된다.

$$K=[\ 6485.11 \quad 1100.19] \tag{25}$$

이 제어기는 (24)에서의 제어기와 동일한 그림을 그린다.

이러한 모의실험(simulation) 결과로부터, 보장된 cost LQ 정적 제어기가 보장된 cost LQ 퍼지 제어기와 비슷한 역할을 한다고 말할 수 있다. 또다른 재미있는 특징은 이러한 정적 이득이 K_1보다는 K_2 ((18)식에서)에 가까운 것인데, 이는 0°근처에서보다 ±90°근처에서 비선형 시스템 (16)과 (17)의 선형화된 모델이 안정화 측면에서 그 시스템의 양상을 기술하는데 중요한 역할을 함을 의미한다.

5. 결론

본 연구에서는 Tatagi-Sugeno 퍼지 모델링에 기반을 둔 고정 이득 되먹임(state-feedback) 퍼지 제어기를 체계적으로 설계하기 위한 보장된 LQ 라 불리는 성능 지표를 소개하였다. 제어에서 가장 유용한 cost는 입력(inputs)과 상태(states)에 관련된 에너지이다. 퍼지 모델에서 미래의 전체 에너지 양을 계산하기란 아주 고비용을 필요로 하거나 아예 불가능하다. 왜냐하면 회원함수(membership functions)의 등급들이 과거와 현재 환경의 비선형 함수이기 때문이다.

그래서, 기본적인 개념은, LQ cost 자체보다는 회원함수의 모든 허용가능한 등급들에 대해서 LQ cost의 상위한도(upper bound)를 최소화하도록 제어기를 만드는 것이다. 모든 변수들을 사용한 선형 행렬 부등식을 만듦으로써 어떤 의미에서 유한한 수의 계산으로 최적 이득을 찾을 수 있고, 선형 프로그램을 거쳐 추가된 제한(constraints)들을 또한 선형적으로 다룰 수 있다.

결과로 나오는 최적의 보장된 LQ 퍼지 제어기는 초기 상태 값의 함수인데, 이것은 초기치 측정 후 계산 시간이 주어지지 않은 경우 실현 불가능하다. 이 경우, 초기 상태 값에 관련 없는 미리 결정된 이득을 사용해야 한다. 이러한 상황을 처리하기 위해, 약간 변형된 모양의 보장된 LQ 퍼지 제어기도 제시되었다.

이러한 퍼지 제어기들은 회원함수들의 등급에 관련해 시변 혹은 비선형이기 때문에, 실제 현장에서 그것들을 구현하는데 어려움이 있다. 그래서 본 연구에서는 퍼지 대신 정적 제어(static control)를 제안했다; 그렇지만 그것을 설계하는 법칙은 역시 퍼지 제어기에 기반을 두고 있다. 결과로 나온 정적제어기들은 보장된 LQ 퍼지 제어기들보다 덜 유연하나(후자(後者)들 중에서 전자(前者)는 특변한 경우로 해석되기 때문), 제어 동작은 회원함수들의 등급과 아무런 관련이 없다. 구현은, 단지 피드백 루프에서 정적이득을 추가하기만 하면 되므로 아주 간단하다.

본 연구는, 잘 알려진 극배치 기반(pole-assignment-based)의 퍼지 및 정적 제어기들에 비교하여 새로운 제어기의 작동을 보여주기 위해서 '차 위의 역진자'를 이용했다. 모의실험(simulations)결과는 해로운 제어기가 다른 것들보다 얼마나 잘 동작하는 지를 보여주었다. 그 까닭은 간단하다: 극배치(pole-assignment) 방법은 시간 영역에서 상태의 변화(behavior)를 적절하게 제어하지 못해지고 energy-cost에 기반한 방법도 그렇기 때문이다. 거꾸로 선 진자의 중심을 맞추는 것에 대해 고려해 본 대로, 보장된 LQ 정적 제어기는 사용될 수 없다. 재미있는 관찰 결과는 균형이 맞추어지지 않은 상태에서 선형화 모델보다 균형 맞추어진 상태의 선형화 모델이 덜 중요한 역할을 한다는 점이다.

앞에서 소개한 것과 같이 모든 설계 방법들은 선형 행렬부등식으로 이루어져 있는데, 이는 선형 프로그램을 통하여 계산 가능하게 해 준다. 따라서, 항상 유한한 횟수의 계산으로 최적의 해답을 찾을 수 있고, 추가된 제한요소들을 선형적으로 쉽게 취급할 수 있다.

6.2 퍼지 모델에 대한 파라메타 종속형 제어기 설계 연구

1. 서론

산업 현장에 사용되는 실제 시스템은 대부분이 비선형 시스템이며 이러한 비선형 시스템은 전통적인 방법의 제어기법으로는 실제로 제어하기가 매우 어렵다. 그러한 이유로 전통적인 제어기법이 아닌 다른 수많은 시도가 있었다. 우선 리아푸노프(Lyapunov) 방정식을 바탕으로 직접적이거나 적응기법을 첨가한 비선형 시스템 제어에 대한 연구가 개발되었다. 그러나 이러한 제어 기법은 대상 시스템의 구조와 조건에 대해서 너무 많은 의존성을 가지기 때문에 일반적이고 광범위한 기법으로의 접근이 불가능하다는 것이 증명되었다.

그러나 이와 같은 비선형 시스템에 대하여 불확실 선형 파라메타 변형 시스템의 설계 문제에 주로 사용되는 이득 보간법(Gain-Scheduling)이라는 기법을 이용하여 산업 현장의 많은 비선형 문제들을 성공적으로 해결해온 사례가 있다. 본 연구도 이를 기본으로 하여 진행한다.

이득 보간법은 최근에 와서야 그 연구가 활발해진 분야로서 이론적인 관점과 실제적인 관점에서 그 특성을 파악할 수 있다. 우선 이론적인 관점에서 그 발전 추이를 살펴보면 다음과 같다. 초기 이득 보간법은 폐루프 비선형 시스템의 안정성과 안정성의 성능보장 영역을 위한 적절한 리아푸노프 방정식을 찾는데서 시작했다. 이러한 방법은 무척 고전적인 접근방법으로서 국소적으로 설계된 제어기를 연결하기 위하여 반복하여 보간방법을 사용한 경우이다. 그러나 이러한 접근방법은 국소적인 시스템을 통합하였을 때 그 전체 시스템의 안정성 보장을 논할 수 없는 이론적인 약점을 가지고 있었다. 이러한 초기 약점을 보완하기 위해서 두 번째로 제안된 방법이 선형 부분 변환법 (Linear Franctional Transformation)으로서 - 혹은 2차 이득 보간법 - 이 방법은 고정된 단일 리아푸노프 방정식을 사용하여 전체 시스템의 안정성 해석을 시도하는데 역시 이 방법에도 보간된 변수에 리아푸노프 방정식이 전적으로 종속된다는 결점이 있다. 그리고 최근 이러한 기법에 대한 중요한 개선이 이루어 졌는데 이는 강인한 분석방법의 하나로서 파라메타에 종속된

리아푸노프 방정식 또는 파라메타 종속형 제어기 개념을 확장한 것이다. 본 연구에서 제안한 안정석 분석에 관한 문제의 동기가 위와 같은 개념을 기본으로 한다. 즉 비선형 시스템에 대하여 일반적이고 새로운 형태의 파라메타 종속형 시스템으로 표현을 하고 이러한 시스템에 대하여 파라메타 종속형 제어기 설계 기법을 적용한다. 이러한 적용 방법은 본 연구의 매우 중요한 부분이다.

이득 보간법의 또다른 관점인 실제적인 측면에서 분석을 해보면, 비선형 시스템에 대한 이득 보간법이란 여러 동작점에 대하여 국소적인 선형 제어기를 설계하고 이를 전체적으로 보간하는 방법으로서 이러한 방법은 일반적으로 각 동작점에 대하여 선형 파라메타 변형 궤환 추적기로서 이해가 된다. 그러나 이러한 방법은 단순한 반면 각 동작점에 대하여 국소적으로 선형 간략화 된 모델을 기준으로 하기 때문에 원래의 본 시스템을 완전히 표현하기에는 불리하고 좀더 강건한 성질이 필요하게 된다. 위와 같은 이유로 이론적인 접근 방법의 좋은 해석 방법인 파라메타 종속형 리아푸노프 방정식과 강건 제어 기법을 혼합하여 시스템의 실제적인 성능을 개선한 방법이 널리 사용하게 되었다. 이러한 분석은 본 연구의 또 다른 연구동기가 되는 것으로서 이득 보간된 요소의 약점을 보완하기 위해서 본 연구에서는 보장된 LQ-Cost 제어 기법을 소개 혼합 하도록 한다.

본 연구의 기본방향은 비선형 시스템에 대하여 새로운 표현 형태인 파라메타 종속형 제어기를 제안 설계하는 것이다. 진행 순서는 다음과 같다. 우선 2장에서는 시간이나 상태변수에 종속되는 가중치 함수를 이용한 선형 세부시스템들의 Convex 조합으로 이루어진 새로운 비선형 시스템의 표현법을 유도 소개하고 실제 시스템에서의 존재성을 다룬다. 다음으로 3장에서는 선형 행렬 부등식(LMI)을 이용하여 모든 허용 가능한 가중치 함수의 범위를 위한 안정성 해석 방법을 제안한다. 4장에서는 3장에서 제안된 이론에 보장된 LQ-Cost 개념을 추가하여 실제적인 성능보장 제어기 설계 기법을 소개한다. 그리고 5장에서는 예제로서 증명한다.

2. 파라메타 종속형 비선형 시스템

하나의 비선형 시스템은 하나의 파라메타 종속형 선형 시스템으로 나타낼수 있는 것이 일반적이다. 즉 여러 동작점을 선정하고 각 동작점에 대하여 다음과 같이 국소 선형 파라메타 불확실 세부 시스템을 이끌어 낸다.

$$\dot{x}(t) = \{A_0 + \sum_{i=1}^{r} w_i A_i\} x(t) + \{B_0 + \sum_{i=1}^{r} w_i B_i\} u(t), \qquad (1)$$

여기서 $x \in R^n$은 상태 벡터, $u \in R^m$은 제어입력 벡터, A_0, B_0, A_i, B_i는 적당한 차원의 실상수 행렬이다. 그리고 ω_i는 $x(t)$를 R로 투영하는 매개체이며 다음의 제한 조건을 가진다.

$$\sum_{i=1}^{r} w_i = \overline{w}(t) \le \overline{w}_{\max}, \qquad \beta_i \ge w_i \ge \alpha_i \ge 0. \qquad (2)$$

그리고 본 연구의 목적은 위 시스템에 대하여 아래와 같은 파라메타 종속형 제어기를 설계하는 것이다.

$$u(t) = \{K + \sum_{i=1}^{r} w_i K_i\} x(t), \qquad (3)$$

여기서 K, K_i는 적당한 차원의 실상수 행렬이다.

그런 다음 새로운 가중치를 가진 세부시스템들과 그에 대응하는 제어기를 결합하면 전체 폐루프 시스템의 모양은 다음과 같아진다.

$$\dot{x}(t) = [A_0 + B_0 K + \sum_{i=1}^{r} w_i \{A_i + B_i K\} + \sum_{j=1}^{r} w_j B_0 K_j + \sum_{i=1}^{r} \sum_{j=1}^{r} w_i w_j B_i K_j] x(t). \qquad (4)$$

즉 위의 표현은 파라메타 종속형 매개체 ω_i를 이용한 비선형 시스템의 새로운 표현법이다.

이러한 표현법에서 중요한 역할을 하는 가중치 함수인 파라메타 종속형 매개체 ω_i에 대하여 설명을 하면 다음과 같은 특징이 있다. 일반적인

파라메타 종속형 매개체 ω_i 는 다음과 같이 시간 t, 상태변수 x(t), 그리고 입력변수 u(t)의 함수로 표현된다.

$$w_i = w_i(t, x(t), u(t)). \tag{5}$$

그리고 하나의 비선형 시스템으로부터 일련의 적절한 파라메타 종속형 매개체의 집합을 선택할 수 있다면, 우리는 이것을 이용하여 시스템으로부터 비선형성을 제거할 수 있게된다. 즉 일련의 적절한 파라메타 종속형 매개체 집합과 이에 대응되는 선형 세부 시스템을 이용하여 하나의 비선형 시스템을 완벽하게 표현할 수 있다.

본 연구에서는 이러한 표현, 그 중에서도 이러한 일련의 파라메타 종속형 매개체 w_i의 선택이 매우 중요한 문제이고 또한 이러한 매개체의 선택은 쉽지 않다. 그러나 많은 경우 일반적으로 파라메타 종속형 매개체 w_i는 시간, 상태변수, 입력변수 들 중에서 입력변수 하나, 또는 상태변수 한, 둘 정도의 함수로 나타나는 현상을 보이므로 원래 시스템에 대한 간단한 분석 등으로 찾을 수 있는 보완점이 있다.

특별한 경우에 해당하는 예제로서, 하나의 일련의 상태변수에 종속된 매개체 집합을 선택하여 하나의 비선형 시스템을 새로운 표현 방법으로 완벽히 구현해 보고자 한다. 특히 이러한 표현법은 넓은 의미로 Takagi-Sugeno 퍼지 모델을 특별한 하나의 경우로 포함하는 방법임을 보인다.

본 연구의 일반적인 형태인 파라메타 종속형 표현법은 Takagi-Sugeno 퍼지 모델의일반형과 같은 모양을 보이는데, 일반적인 Takagi-Sugeno 퍼지 모델은 퍼지 “IF-THEN" 규칙의 집합에 의해 구성되는데 IF의 전건부는 퍼지 집합이고 THEN의 후건부는 LTI 시스템의 형태를 나타낸다. 즉 Takagi-Sugeno 퍼지 모델은 다음과 같은 규칙들로 이루어진다.

· i^{th} *Plant Rule* : *IF* $x_1(t)$ *is* $\widehat{M_{i1}}$ and · · · , $x_j(t)$ *is* $\widehat{M_{ij}}$, *THEN* $\dot{x} = A_i x + B_i u$.

여기서 x는 상태 벡터이고, r은 규칙의 수이며, $\widehat{M}_{ij}$는 입력 퍼지 집합이다.

퍼지 시스템의 수식화를 위해서 Singleton 퍼지 입력기, Max-Product 퍼지 추론기, 그리고 무게평균 퍼지 출력기를 사용하면 다음과 같이 퍼지 시스템을 수식화할 수 있다.

$$\dot{x} = \frac{\sum_{i=1}^{r} \eta_i(x)(A_i x + B_i u)}{\sum_{i=1}^{r} \eta_i(x)}, \tag{6}$$

여기서 η_i는 i번째 규칙의 j번째 퍼지 집합의 등급 함수 μ_{ij}에 의해 정의되며, 다음과 같다

$$\eta_{i(x)} = \prod_{j=1}^{l} \mu_{ij}(x_j), \tag{7}$$

또한 다음과 같이 정의를 하면

$$w_{i(x)} = \frac{\eta_i(x)}{\sum_{i=1}^{r} \eta_i(x)}. \tag{8}$$

퍼지 시스템 (6)을 다음과 같이 나타낼 수 있다.

$$\dot{x} = \sum_{i=1}^{r} w_i(x)(A_i x + B_i u), \tag{9}$$

여기서 $\omega_i \geq 0$, $\sum_{i=1}^{r} \omega_i = 1$의 조건을 만족한다.

그리고 이러한 형태는 우리가 제안한 파라메타 종속형 비선형 시스템 표현법 형태 중에서도 파라메타 매개체가 상태변수에 종속되는 형태를 가지는 경우이며 특히 매개체가 $\omega_i(x), \overline{\omega}(x) = 1$의 특별한 형식을 가진다. 이러한 이유로, 본 연구에서 제안한 비선형 시스템에 대한 파라메타

종속형 표현법은 Takagi-Sugeno 퍼지 모델도 특별한 하나의 경우로 포함하는 강력하고 광범위한 표현법임을 증명할 수 있다. 즉. 본 연구의 가장 중요한 자랑거리가 된다.

제어기 설계를 위하여 각 퍼지 모델의 퍼지 규칙에 대응되도록 퍼지 제어 규칙을 설계하면 다음과 같고

$\cdot i^{th}$ *Controller Rule* : *IF* $x_1(t)$ *is* $\hat{M}_{i1}$ and $\cdots$, $x_j(t)$ *is* $\hat{M}_{ij}$, *THEN* $\dot{x} = A_i x + B_i u$.

이것을 수식화 하면 다음과 같다.

$$u = \{K + \sum_{i=1}^{r} w_i(x) K_i\} x, \tag{10}$$

그리고 퍼지 모델(9)와 퍼지 제어기(10)을 혼합하여 전체 폐루프 시스템을 구현하면 다음과 같다

$$\dot{x} = \sum_{i=1}^{r} \sum_{j=1}^{r} w_i(x) w_j(x) \{(A_i + B_i K) + B_i K_j\} x. \tag{11}$$

이러한 표현법은 본 연구에서 제안한 표현법과 일치한다. 더욱이 파라메타 매개체에 특수성을 가지고 있는 특별한 경우로서 나타난다. 본 장에서 유도한 시스템 표현을 이용하여 다음 장에서는 안정성 해석을 시도한다.

3. 안정성 해석을 위한 정리

2장에서 유도된 파라메타 종속형 매개체 ω_i, ω_j에 의해 표현되는 폐루프 시스템은 일반적으로 다음과 같고

$$\dot{x}(t) = [A_0 + B_0 K + \sum_{i=1}^{r} w_i \{A_i + B_i K\} + \sum_{j=1}^{r} w_j B_0 K_j + \sum_{i=1}^{r} \sum_{j=1}^{r} w_i w_j B_i K_j] x(t), \tag{12}$$

여기서 안정성 보장을 위해서 리아푸노프 2차 안정성해석을 고려한다. 우선 n-차원 벡터를 1-차원 스칼라 값으로 투영하는 리아푸노프 함수 V(x)를 다음과 같이 정의한다.

$$V(x) = x^T Y x, \tag{13}$$

$$[A_0 + B_0 K + \sum_{i=1}^{r} w_i\{A_i + B_i K\} + \sum_{j=1}^{r} w_j B_0 K_j + \sum_{i=1}^{r}\sum_{j=1}^{r} w_i w_j B_i K_j]^T Y$$
$$+ Y[A_0 + B_0 K + \sum_{i=1}^{r} w_i\{A_i + B_{iK}\} + \sum_{j=1}^{r} w_j B_0 K_j + \sum_{i=1}^{r}\sum_{j=1}^{r} w_i w_j B_i K_j] < 0, \tag{14}$$

그러면 다음과 같은 조건을 구할 수 있다.

또한 다음과 같은 정의식을 이용하면

$$P = Y^{-1}, \Pi_j = K_j P$$

$$P(A_0 + B_0 K)^T + (A_0 + B_0 K)P + \sum_{i=1}^{r} w_i\{P(A_i + B_i K)^T + (A_i + B_i K)P\} +$$
$$\sum_{j=1}^{r} w_j\{\Pi_j^T B_0^T + B_0 \Pi_j\} + \sum_{i=1}^{r}\sum_{j=1}^{r} w_i w_j\{\Pi_j^T B_i^T + B_i \Pi_j\} < 0, \tag{15}$$

위의 조건을 다음과 같이 변형할 수 있다.

그리고 위 시스템의 완전한 안정성 해석을 위해서는 위의 조건 외에도 식(2)에서 나타나는 조건도 고려해야 모든 허용가능한 파라메타 매개체 즉, 가중치 함수를 유지할 수 있다. S-Procedure와 같은 방법을 사용하여 제한조건 소거 방법을 적용하면 다음 정리와 같은 새로운 안정성 해석 이론을 제안할 수 있다.

< 정리 3.1 >

전체 파라메타 종속형 폐루프 시스템 식(12)는 다음과 같은 조건을 만족하는 행렬 P, Λ, Λ_i, Π, Π_j가 존재 할 경우 globallly asymptotically stable in the large 이다.

$$\begin{bmatrix} X & \Gamma_1 & \Gamma_2 & \cdots & \Gamma_r \\ \Gamma_1^T & D_1 & \Delta_{12} & \cdots & \Delta_{1r} \\ \Gamma_2^T & \Delta_{12}^T & D_2 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \Delta_{(r-1)r} \\ \Gamma_r^T & \Delta_{1r}^T & \cdots & \Delta_{(r-1)r}^T & D_r \end{bmatrix} < 0, \tag{16}$$

$$\Lambda + \Lambda^T \geq 0, \tag{17}$$

$$\Lambda_i + \Lambda_i^T \geq 0, \tag{18}$$

$$P > 0, \tag{19}$$

여기서 위식의 변수들을 다음과 같이 정의한다.

$$X = PA_0^T + A_0P + \Pi^T B_0^T + B_0\Pi + \overline{\omega_{\max}}(\Lambda + \Lambda^T) - \sum_{i=1}^{r} \alpha_i\beta_i(\Lambda_i + \Lambda_i^T),$$
$$\Gamma_i = PA_i^T + \Pi^T B_i^T - \Lambda^T + (\alpha_i + \beta_i)\Lambda_i^T + \Pi_i^T B_0^T,$$
$$D_i = -(\Lambda_i + \Lambda_i^T) + \Pi_i^T B_i^T + B_i\Pi_i,$$
$$\Delta_i = \Pi_i^T B_j^T + \Pi_j^T B_i^T.$$

이 경우 제어기 이득 K, K_j는 Π, Π_j, P를 이용하여 찾을 수 있는데 그 관계는 다음과 같다

$$K = \Pi P^{-1}, K_j = \Pi_j P^{-1}$$

< 증명 >

증명을 간략화 하기 위해서 r=3으로 가정한다.

다음과 같은 positive-definite 행렬 P가 포함된 리아푸노프 방정식을 고려한다.

$$PA_0^T + \Pi^T B_0^T + A_0P + B_0\Pi + \sum_{i=1}^{3} w_i\{PA_i^T + \Pi^T B_i^T + A_iP + B_i\Pi\} +$$
$$\sum_{j=1}^{3} w_j\{\Pi_j^T B_0^T + B_0\Pi_j\} + \sum_{i=1}^{3}\sum_{j=1}^{3} w_i w_j\{\Pi_j^T B_i^T + B_i\Pi_j\} < 0,$$

(a.1)

또한 매개체에 대한 두가지 제한조건은 다음과 같다.

$$(\overline{w}_{\max} - \sum_{i=1}^{3} w_i)(\Lambda + \Lambda^T) \geq 0, \qquad \text{(a.2)}$$

$$-\{w_i^2 - (\alpha_i+\beta_i)w_i + \alpha_i\beta_i\}(\Lambda_i + \Lambda_i^T) \geq 0, \qquad \text{(a.3)}$$

여기서 Λ는 조건 $(\Lambda+\Lambda^T)\geq 0$을 만족하는 임의의 R^{n*n}행렬이며, Λ_i모든 i=1,2,3에 대하여 조건 $(\Lambda_i+\Lambda_i^T)\geq 0$를 만족한다.

위의 모든 조건을 종합하면 다음과 같은 전체 조건식을 구할 수 있다.

$$\begin{aligned}&\Big[PA_0^T+\Pi^T B_0^T+A_0P+B_0\Pi+\sum_{i=1}^{3} w_i\{PA_i^T+\Pi^T B_i^T+A_{iP}+B_i\Pi\}+\\&\qquad\sum_{j=1}^{3} w_j\{\Pi_j^T B_0^T+B_0\Pi_j\}+\sum_{i=1}^{3}\sum_{j=1}^{3} w_{iw_j}\{\Pi_j^T B_i^T+B_i\Pi_j\}\Big]+\\&\Big[(\overline{w}_{\max}-\sum_{i=1}^{3} w_i)(\Lambda+\Lambda^T)-\sum_{i=1}^{3}(w_i^2-(\alpha_i+\beta_i)w_i+\alpha_i\beta_i)(\Lambda_i+\Lambda_i^T)\Big]<0.\end{aligned}$$

(a.4)

또한 위의 조건식은 다음과 같은 두 조건의 합으로 나타낼 수도 있다.

$$\Phi^T\begin{bmatrix} PA_0^T+\Pi^T B_0^T+A_0P+B_0\Pi & PA_1^T+\Pi^T B_1^T+\Pi_1^T B_0^T & PA_2^T+\Pi^T B_2^T+\Pi_2^T B_0^T & PA_3^T+\Pi^T B_3^T+\Pi_3^T B_0^T \\ A_1P+B_1\Pi+B_0\Pi_1 & \Pi_1^T B_1^T+B_1\Pi_1 & \Pi_2^T B_1^T+\Pi_1^T B_2^T & \Pi_3^T B_1^T+\Pi_1^T B_3^T \\ A_2P+B_2\Pi+B_0\Pi_2 & B_1\Pi_2+B_2\Pi_1 & \Pi_2^T B_2^T+B_2\Pi_2 & \Pi_3^T B_2^T+\Pi_2^T B_3^T \\ A_3P+B_3\Pi+B_0\Pi_3 & B_1\Pi_3+B_3\Pi_1 & B_2\Pi_3+B_3\Pi_2 & \Pi_3^T B_3^T+B_3\Pi_3 \end{bmatrix}\Phi,$$

(a.5)

$$\Phi^T\begin{bmatrix} \overline{w}_{\max}(\Lambda+\Lambda^T)-\sum\alpha_i\beta_i(\Lambda_i+\Lambda_i^T) & (*) & (*) & (*) \\ -\Lambda+(\alpha_1+\beta_1)\Lambda_1 & -(\Lambda_1+\Lambda_1^T) & 0 & 0 \\ -\Lambda+(\alpha_2+\beta_2)\Lambda_2 & 0 & -(\Lambda_2+\Lambda_2^T) & 0 \\ -\Lambda+(\alpha_3+\beta_3)\Lambda_3 & 0 & 0 & -(\Lambda_3+\Lambda_3^T) \end{bmatrix}\Phi.$$

(a.6)

여기서 $\Phi\equiv[I\ w_1 I\ w_2 I\ w_3 I]^T$이다.

그러므로 만약에 (16)~(18)까지의 조건식이 만족한다면, 안정성 조건

(a.1)은 모든 허용 가능한 매개체 ω_i에 대하여 만족한다. 증명 끝.

■

위 정리에 대해서 다음과 같은 부연 설명을 할 수 있다. 우선 본 연구에서 제안한 정리는 시간 영역의 보간법으로 일반적인 비선형 시스템에 대한 새로운 파라메타 종속형 안정성 분석을 다룬 것이다. 또한 위 행렬 조건식의 모든 변수들에 대해서 선형이기 때문에 위 결과는 선형 행렬 부등식 형태를 만족하고 이러한 형태의 시스템 안정성 문제는 선형 프로그램을 이용하여 쉽게 그 해를 구할 수 있다. 그러나 이러한 제안은 이론적인 범주이고 실제적인 문제는 항상 어떤 설계 목적을 만족해야 하는데 이러한 논의는 다음 장에서 보인다.

4. 성능 보장 제어기(Guaranteed Cost Controller) 설계

4. 1 기본 개념과 제어기 설계

이 장에서는 아래 식과 같이 정의되는 Cost 함수를 이용하여 파라메타 종속형 시스템의 상태와 입력과 연관된 LQ-Cost의 상한값을 최소화하는 성능 보장 제어기를 소개한다.(퍼지 시스템 포함)

$$\min_{K_j} \quad \max_{w_i \in W} \int_0^\infty x^T(t) C^T Q C x(t) + u^T(t) R u(t)\, dt, \tag{20}$$

여기서 $Q > 0, R > 0$이고 W는 모든 허용가능한 가중치 함수의 값들의 집합이다.

이 경우, LQ-Cost는 가중치 함수들의 값, 즉 매개체 ω_i의 함수가 될 것이다. 만약 ω_i의 정확한 궤적을 알 수 있다면, 정확한 LQ-Cost의 값을 계산할 수 있을 것이다. 그러나 그것은 실제로 매우 복잡하며 심지어는 불가능하다. 따라서, 모든 허용 가능한 매개체 ω_i의 조건 아래에서의 LQ-Cost의 상한값이 한 가지 타당한 후보가 되며, 그 까닭으로 이는 성능 보장 제어기라고 불리운다. 다음의 정리는 가중치 함수들의 모든 허용 가능한 매개체 ω_i에 대해서 성능 보장 LQ-Cost를 최소화시키는

최적 제어기를 어떠한 방법으로 찾을 수 있는지를 요약한다.

< 정리 4. 1 > (성능 보장 제어기 설계)

파라메타 종속형 퍼루프 시스템은 만약 아래의 조건을 만족하는 행렬 P, Λ, Λ_i, Π, Π_j 가 존재한다면, globally asymptotically stable in the large이다.

$$\begin{bmatrix} X & \Gamma_1 & \Gamma_2 & \cdots & \Gamma_r & PC^T & \Pi^T \\ \Gamma_1^T & D_1 & \Delta_{12} & \cdots & \Delta_{1r} & 0 & \Pi_1^T \\ \Gamma_2^T & \Delta_{12}^T & D_2 & \ddots & \vdots & 0 & \Pi_2^T \\ \vdots & \vdots & \ddots & \ddots & \Delta_{(r-1)r} & \vdots & \vdots \\ \Gamma_r^T & \Delta_{1r}^T & \cdots & \Delta_{(r-1)r}^T & D_r & 0 & \Pi_r^T \\ CP & 0 & 0 & \cdots & 0 & -Q^{-1} & 0 \\ \Pi & \Pi_1 & \Pi_2 & \cdots & \Pi_r & 0 & -R^{-1} \end{bmatrix} < 0, \tag{21}$$

$$\Lambda + \Lambda^T \geq 0, \tag{22}$$

$$\Lambda_i + \Lambda_i^T \geq 0, \tag{23}$$

$$P > 0, \tag{24}$$

여기서

$$\begin{aligned} X &= PA_0^T + A_0P + \Pi^T B_0^T + B_0\Pi + \overline{w}_{\max}(\Lambda + \Lambda^T) - \sum_{i=1}^{r} \alpha_i\beta_i(\Lambda_i + \Lambda_i^T), \\ \Gamma_i &= PA_i^T + \Pi^T B_i^T - \Lambda^T + (\alpha_i + \beta_i)\Lambda_i^T + \Pi_i^T B_0^T, \\ D_i &= -(\Lambda_i + \Lambda_i^T) + \Pi_i^T B_i^T + B_i\Pi_i, \\ \Delta_{ij} &= \Pi_i^T B_j^T + \Pi_j^T B_i^T. \end{aligned}$$ 이다.

이 때, 제어기 이득 K, K_j는 행렬 Π, Π_j, P에서 구해지는데 다음과 같이 정의된다.

$$K = \Pi P^{-1}, \quad K_j = \Pi_j P^{-1} \tag{25}$$

그리고 Cost 함수 (20)은 $x^T(0)P^{-1}x(0)$에 의해서 bound된다. 조건식 (21)은 변수 P, Λ, Λ_j, Π, Π_j 에 대해서 선형이므로, 최적의 성능보장 Cost는 다음과 같은 선형 프로그램에 의해서 얻어질 수 있다. : (21)식과 다음의 식을 만족하는 γ를 최적화 한다.

$$\begin{bmatrix} \gamma & x^T(0) \\ x(0) & P \end{bmatrix} \geq 0, \tag{26}$$

< 증명 >

r=3으로 가정한다.

다음과 같이 전체 폐루프 시스템을 고려한다.

$$\dot{x}(t) = [A_0 + B_0K + \sum_{i=1}^{3} w_i\{A_i + B_iK\} + \sum_{j=1}^{3} w_jB_0K_j + \sum_{i=1}^{3}\sum_{j=1}^{3} w_iw_jB_iK_j]x(t), \tag{b.1}$$

그리고 cost 함수는 상태와 입력과 관계가 있는데 다음과 같이 정의한다.

$$\min_{K_j} \max_{w_i \in W} \int_0^\infty \{x^TC^T(t)QCx(t) + u^T(t)Ru(t)\}dt, \tag{b.2}$$

그리고 이 경우 하나의 positive-definite 행렬 Y를 고려하여 다음의 리아푸노프 방정식을 고려하는데, 이 때 그 상한 값은 다음과 같다.

$$\int_t^\infty \{x^T(\tau)C^TQCx(\tau) + u^T(\tau)Ru(\tau)\}d\tau < x^T(t)Yx(t), \tag{b.3}$$

만약 위의 전체 폐루프 시스템이 안정하다면, 위 식의 양쪽 모두 시간이 지남에 따라 0으로 수렴하고 또한 계속하여 부등식을 유지한다. 이 경우 위 식을 초기 시간에 대하여 미분하면 모든 상태 x(t)에서 다음 식을 만족한다.

$$\dot{x}^T(t)Yx(t) + x^T(t)Y\dot{x}(t) + x^T(t)C^TQCx(t) + u^T(t)Ru(t) < 0, \tag{b.4}$$

또한 위식은 다음과 같이 유도된다.

$$\begin{aligned} &x^T(t)\Big[A_0 + B_0K + \sum_{i=1}^{3} w_i\{A_i + B_iK\} + \sum_{j=1}^{3} w_jB_0K_j + \sum_{i=1}^{3}\sum_{j=1}^{3} w_iw_jB_iK_j\Big]^T Yx(t) \\ &+ x^T(t)Y\Big[A_0 + B_0K + \sum_{i=1}^{3} w_i\{A_i + B_iK\} + \sum_{j=1}^{3} w_jB_0K_j + \sum_{i=1}^{3}\sum_{j=1}^{3} w_iw_jB_iK_j\Big]x(t) \\ &+ x^T(t)Qx(t) + x^T(t)\Big[K + \sum_{j=1}^{r} w_jK_j\Big]^T R\Big[K + \sum_{j=1}^{r} w_jK_j\Big]x(t) < 0, \end{aligned} \tag{b.5}$$

위식에서 안정성 조건만 고려하면 다음의 조건식이 된다.

$$\begin{aligned} &\Big[A_0 + B_0K + \sum_{i=1}^{3} w_i\{A_i + B_iK\} + \sum_{j=1}^{3} w_jB_0K_j + \sum_{i=1}^{3}\sum_{j=1}^{3} w_iw_jB_iK_j\Big]^T Y \\ &+ Y\Big[A_0 + B_0K + \sum_{i=1}^{3} w_i\{A_i + B_iK\} + \sum_{j=1}^{3} w_jB_0K_j + \sum_{i=1}^{3}\sum_{j=1}^{3} w_iw_jB_iK_j\Big] \\ &+ Q + \Big[K + \sum_{j=1}^{r} w_jK_j\Big]^T R\Big[K + \sum_{j=1}^{r} w_jK_j\Big] < 0, \end{aligned} \tag{b.6}$$

여기서 다음과 같이 정의하면

$$P \equiv Y^{-1}, \Pi \equiv KP, \Pi_j \equiv K_jP \tag{b.7}$$

$$\begin{aligned} &PA_0^T + \Pi^TB_0^T + A_0P + B_0\Pi + \sum_{i=1}^{3} w_i\{PA_i^T + \Pi^TB_i^T + A_iP + B_i\Pi\} \\ &+ \sum_{j=1}^{3} w_j\{\Pi_j^TB_0^T + B_0\Pi_j\} + \sum_{i=1}^{3}\sum_{j=1}^{3} w_iw_j\{\Pi_j^TB_i^T + B_i\Pi_j\} \\ &+ PQP + \Big[\Pi + \sum_{i=1}^{3} w_j\Pi_j^T\Big]R\Big[\Pi + \sum_{i=1}^{3} w_j\Pi_j\Big] < 0, \end{aligned} \tag{b.8}$$

그러면 위 안정성 조건은 다음과 같이 변형된다.

여기에 정리 3. 1과 같은 아래 식의 매개체 제한 조건을 고려한다.

$$
\begin{aligned}
&(\ \overline{w}_{\max} \ - \ \sum_{i=1}^{3} w_{i)(}\Lambda \ + \ \Lambda^{T}) \ \geq \ 0, \\
-(\,w_i^2 \ - \ &(\alpha_i + \beta_i) w_i \ + \ \alpha_i \beta_i)(\Lambda_i \ + \ \Lambda_i^{T}) \ \geq \ 0,
\end{aligned}
\tag{b.9}
$$

여기서 Λ는 조건 $(\Lambda + \Lambda^{T}) \geq 0$을 만족하는 임의의 R^{n*n}행렬이며, Λ_i모든 i=1,2,3에 대하여 조건 $(\Lambda_i + \Lambda_i^{T}) \geq 0$를 만족한다.

위의 모든 조건을 종합하면 다음과 같다.

$$
\begin{aligned}
&PA_0^{T} + \Pi^{T} B_0^{T} + A_0 P + B_0 \Pi + \sum_{i=1}^{3} w_i \{PA_i^{T} + \Pi^{T} B_i^{T} + A_i P + B_i \Pi\} \\
&\quad + \sum_{j=1}^{3} w_j \{\Pi_j^{T} B_0^{T} + B_0 \Pi_j\} + \sum_{i=1}^{3} \sum_{j=1}^{3} w_i w_j \{\Pi_j^{T} B_i^{T} + B_i \Pi_j\} \\
&\quad + PQP + \{\sum_{i=1}^{3} w_j \Pi_j^{T}\} R \{\sum_{i=1}^{3} w_j \Pi_j\} \\
+ \Big[(\ \overline{w}_{\max} \ - \ &\sum_{i=1}^{3} w_i)(\Lambda \ + \ \Lambda^{T}) - \sum_{i=1}^{3} (w_i^2 \ - \ (\alpha_i + \beta_i) w_i \ + \ \alpha_i \beta_i)(\Lambda_i \ + \ \Lambda_i^{T}) \Big] < 0.
\end{aligned}
\tag{b.10}
$$

그 다음으로 정리 3. 1의 과정과 Shur complement를 이용하면 조건 (21)과 (24)를 구할 수 있다.

만약 조건식(21)이 만족되면, cost 함수 (20)은 $x^{T}(0) Y x(0) = x^{T}(0) P^{-1} x(0)$ 에 의해 bound된다. 즉

$$
\int_0^{\infty} x^{T}(t) C^{T} Q C x(t) + u^{T}(t) R u(t)\, dt \ < \ x^{T}(0) Y x(0)
\tag{b.11}
$$

또한 feasible 해 γ, P, Π_j가 조건 (21), (26)을 만족하므로 임의의 매개체에 대해서도 다음과 같은 조건이 성립한다.

$$
\int_0^{\infty} x^{T}(t) C^{T} Q C x(t) + u^{T}(t) R u(t)\, dt \ < \ x^{T}(0) Y x(0) \ \leq \ \gamma,
\tag{b.12}
$$

그러므로 식 (26)을 $x^T(0)P^{-1}x(0)$를 최소화하는데 사용할 수 있다. 증명 끝 ■

정리 4.1에 대한 참고 설명은 다음과 같다. 행렬 부등식 (21)과 (23)은 $1 \le j \le r$인 모든 변수 $\gamma, P, \Lambda, \Lambda_j, \Pi, \Pi_j$에 대해서 선형이다. 따라서 Convex Optimization 방법을 써서 한정된 계산량을 필요로 하는 Global 해를 찾을 수 있으며, 추가된 선형 제한 조건식들을 쉽게 다룰 수 있다.

어떠한 feasible 행렬 $\Lambda, \Lambda_j, \Pi, \Pi_j$과 (21)의 결과를 갖는 positive definite matrix P에 대해서 임의의 가중 함수가 다음의 부등식을 만족시킨다는 점을 주목한다.

$$\int_0^\infty \{x^T(t)C^T QCx(t) + u^T(t)Ru(t)\}dt < x^T(0)P^{-1}x(0).$$

즉, 그러한 P, Π, Π_j를 사용함으로 해서 (25)로부터 선택된 어떠한 이득이라도 전체 시스템을 안정화시킬 것이다. (21)과 (26)을 만족시키는 최소의 γ 값을 찾으면 최적의 제어기를 얻을 수 있다. 여기서 새로운 방법의 취약점은 최적의 이득값은 상태의 초기값들에 의존한다는 것이다. 이 말은 이득값이 초기값들에 따라서 변화된다는 것을 뜻한다. 이러한 단점을 없애기 위해서, 다음의 세부 절에서 또 다른 제어기를 제안할 것이다.

4. 2 초기치와 무관한 제어기 구성

State-feedback을 이용할 수 있으므로 상태들의 초기값은 측정될 수 있으며, 이는 정리 4. 1을 통해서 얻어진 최적의 제어기를 이용할 수 있음을 뜻한다. 그러나 초기값들에 대한 어떠한 정보도 없이 이득 값을 미리 결정해야만 하는 경우, 즉 계산에 필요한 시간이 전혀 없을 경우가 있을 수도 있을 것이다. 이러한 상황을 다루려면, 다음의 Criterion이 필요하다.

$$\min_{K_j} \max_{x(0)\neq 0} \max_{\omega_i \in W} \int_0^\infty \{x^T(t) C^T Q C x(t) + u^T(t) R u(t)\} dt \,/\, x^T(0) x(0), \tag{27}$$

여기서 $Q > 0$ 이고 $R > 0$이다. 이러한 criterion의 의미는 모든 허용 가능한 매개체 $w_i(t)$에 대해서 뿐만 아니라 모든 허용 가능한 상태 초기치 $x(0)$에 대해서도 강인한 퍼지 제어기를 설계해야 한다는 것이다. 다음의 유사정리는 그러한 제어기를 어떠한 방법으로 설계할 수 있는지를 제시해 준다.

<유사정리 4.1> (Independent of Initial States Controller 설계)

Criterion (27)을 만족시키는 최적 제어기는 다음과 같이 구할 수 있다: 조건식 (21)과 다음의 (28)를 만족하는 γ를 최소화한다.

$$0 \leq \begin{bmatrix} \gamma I & I \\ I & Z \end{bmatrix}. \tag{28}$$

이 경우, 최적의 이득은 $K = \Pi P^{-1}$ 와 $K_j = \Pi_j P^{-1}$에 의해서 계산된다.

<증명>

조건식 (21)을 만족시키는 임의의 feasible solution P, Π_j에 대해서, 그리고 임의의 가능한 매개체에 대해서, 이 부등식은 다음을 만족시킨다.

$$\max_{\omega_i \in W} \int_0^\infty \{x^T(t) C^T Q C x(t) + u^T(t) R u(t)\} dt < x^T(0) P^{-1} x(0). \tag{29}$$

이 식으로부터 모든 0이 아닌 $x(0)$ 값에 대해서 다음의 관계식이 나온다.

$$\max_{\omega_i \in W} \int_0^\infty \{x^T(t) C^T Q C x(t) + u^T(t) R u(t)\} dt \,/\, x^T(0) x(0) < \sigma_{\max}(P^{-1}) \tag{30}$$

여기서 $\sigma_{\max}(P^{-1})$는 P^{-1}의 maximal singular value를 뜻한다. 따라서, 조건식 (28)을 만족시키는 γ를 최소화함으로 해서 $\sigma_{\max}(P^{-1})$를 최소화시킬 수 있다. ∎

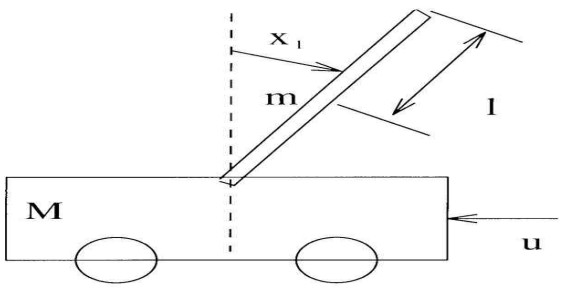
이 제어 법칙으로부터 초기 상태 값들에 무관한 이득을 얻는다.

5. 예제 : Inverted Pendulum

다음으로 본 연구에서 제안한 방법을 검증하기 위해서 Wang et al에 의해서 쓰인 방법인 cart 위에 설치된 Inverted pendulum의 균형을 맞추는 문제를 생각해 본다.(그림 5.1 참고) Pendulum의 운동 방정식은 아래와 같다.

$$\dot{x}_1 = x_2, \tag{31}$$

$$\dot{x}_2 = \frac{g\sin(x_1) - aml x_2^2 \sin(2x_1)/2 - a\cos(x_1)u}{4l/3 - aml\cos^2(x_1)}, \quad a \equiv \frac{1}{m+M},$$

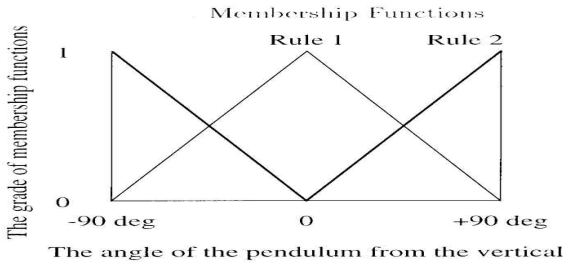


(그림 5.1 : Inverted Pendulum on a Car 모델)

이때 x_1과 x_2는 각각 라디안으로 표시된 각도와 수직면으로부터의 pendulum의 각속도를 말하며, $g=9.8m/s^2$는 중력가속도 상수, m은 pendulum의 질량, M은 cart의 질량, $2l$은 pendulum의 길이, u는 뉴턴의 단위로 표시된 cart에 가해진 힘을 말한다. 그리고 본 연구 결과를 Wang et al 에 의해서 사용된 방법인 pole-placement 기법에 의한 결과, Jadbaabid et al 등이 주장한 guaranteed-cost 퍼지 제어기에 의한 결과 방법과 비교해 보기 위해서 거기서 사용했던 것과 같은 값들을 다음과 같이 선택한다. :

$$m=2.0kg,\quad M=8.0kg,\quad l=0.5m.$$

일반적인 이득 보간법의 절차를 사용하여 본 예제의 해를 위한 진행 방향을 소개하면, 우선적으로 비선형 시스템인 Inverted pendulum 시스템을 각 동작점에서 매개체 ω_i를 가진 파라메타 종속형 선형 시스템으로 구현한다. 그리고 구현된 시스템에 파라메타 종속형 제어기를 조건에 맞도록 설계하여 최종적으로 안정된 전체 시스템을 구한다. 본 문제에서 특이 할 만한 것은 각 동작점이 앞선 연구 Wang et al의 동작점과 일치하기 때문에 파라메타 매개체 ω_i, ω_j가 앞선 연구의 퍼지 멤버쉽 함수 grade와 일치한다는 것이다. (그림 5. 2 참고)



(그림 5.2 : Membership 함수의 정의)

또한 위와 같은 이유로 Inverted pendulum시스템의 파라메타 종속형 표현법은 앞선 연구인 Wang et al의 Takagi-Sugeno 퍼지 모델과 동일한 모양을 가지는데, 이러한 모양은 본연구와 앞선 연구간의 비교에 많은 도움을 준다.

초기 상태는 다음과 같다고 가정한다.

$$x_1(0) \in (-\pi/2, \pi/2), \quad x_2(0) = 0.$$

본 연구의 파라메타 종속형 시스템은 파라메타 매개체 ω_i, ω_j에 의해 다음과 같이 구할 수 있다.

$$\dot{x} = w_1\{A_1 x + B_1 u\} + w_2\{A_2 x + B_2 u\}, \qquad (32)$$

여기서 $x_{oper1} = (0,0)$, $x_{oper2} = (\pm\pi/2, 0)$이다.

이때, 행렬 A_i와 B_i는 다음과 같이 정리된다.

$$A_1 \equiv \begin{bmatrix} 0 & 1 \\ \dfrac{g}{4l/3 - aml} & 0 \end{bmatrix}, \quad B_1 \equiv \begin{bmatrix} 0 \\ -\dfrac{a}{4l/3 - aml} \end{bmatrix},$$

$$A_2 \equiv \begin{bmatrix} 0 & 1 \\ \dfrac{9g}{4\pi(4l/3 - aml\cos^2(80^\circ))} & 0 \end{bmatrix},$$

$$B_2 \equiv \begin{bmatrix} 0 \\ -\dfrac{a\cos(80^\circ)}{4l/3 - aml\cos^2(80^\circ)} \end{bmatrix}.$$

이것은 Jadbabaid et al의 퍼지 모델에서 사용한 행렬과 같다. 그리고 이러한 형태는 본 연구에서 제안한 시스템 표현법의 특별한 경우가 된다. 또한 상태에 종속하는 매개체는 normalization에 의해 항상 그 합이 1이 되고 이러한 경우 이 시스템의 안정성 해석과 제어기 이득 설계 문제는 매우 쉬워진다.

실제적인 제어기 설계의 문제에 있어서는 guaranteed-cost 제어기 설계에 그 바탕을 두고 있으며,

이때 $C=[1,0]$, $\alpha_1=\alpha_2=0$, $\beta_1=\beta_2=1$ 이다. (그림 5.2 참고) 그리고 적절하게 $Q=3$, $R=1$을 선택한다. Matlab의 LMI 툴박스와 정리 4. 1을 이용해서 다음과 같은 각각의 서로 다른 초기 상태에 대해서 아래의 제어기 이득 K, K_1, K_2를 구한다

$$K=\begin{bmatrix} 0.00120 & 0.0011 \\ 0.04630 & 0.0163 \\ 0.48380 & 0.1515 \\ 0.03180 & 0.0099 \\ 0.51910 & 0.1625 \\ 0.22720 & 0.0705 \end{bmatrix},\quad K_1=\begin{bmatrix} 5346.10 & 1631.3 \\ 5347.50 & 1632.3 \\ 5347.20 & 1631.0 \\ 5343.70 & 1630.5 \\ 5350.60 & 1631.7 \\ 5349.20 & 1633.0 \end{bmatrix},\quad K_2=\begin{bmatrix} 809.990 & 247.16 \\ 810.210 & 247.30 \\ 809.710 & 246.96 \\ 809.080 & 247.06 \\ 810.160 & 247.05 \\ 810.330 & 247.37 \end{bmatrix}, \tag{33}$$

여기서 초기 상태는 다음과 같다.

$$x_1(0)=\begin{pmatrix} 15^\circ(0.2618) \\ 30^\circ(0.5236) \\ 45^\circ(0.7854) \\ 65^\circ(1.1345) \\ 75^\circ(1.3090) \\ 80^\circ(1.3962) \end{pmatrix}. \tag{34}$$

본 연구의 결과를 앞선 두 다른 논문의 결과와 비교하기 위해서 다음에 그 제어기를 소개한다.

.Case 1 : Wang et al의 Pole-placement에 의한 Parallel

distributed 보상기 설계

$$u = [w_1(x)\ \overline{K}_1 + w_2(x)\ \overline{K}_2]x. \quad (35)$$

.Case 2 : Jadbabaied et al의 퍼지 모델에 대한 Guaranteed LQ-cost 설계

$$u = [w_1(x)\ \hat{K}_1 + w_2(x)\ \hat{K}_2]x. \quad (36)$$

Case 1의 경우 우리가 원하는 pole은 -2이고 이에 대한 제어기 이득은 다음과 같다.

$$\overline{K}_1 = [120.67\ \ 22.67], \quad \overline{K}_2 = [2551.6\ \ 764.0]. \quad (37)$$

Case 2의 경우 제어기 이득은 다음과 같다.

$$\hat{K}_1 = [225.6\ \ 55.8], \quad \hat{K}_2 = [272.1\ \ 75.9]. \quad (38)$$

여기서 $C = [1, 0], Q = 3, R = 2$이다.

이 모든 경우 시뮬레이션은 3초동안으로 단일화하여 그 성능을 비교한다.

그림 5. 3에 있는 시뮬레이션은 상태변수 $x_1(t)$의 궤적이다. 실선-점선은 Wang의 결과를 나타내는데, 이 경우 $x_1(0)$가 $\pm 88^\circ$ 사이의 모든 경우에 대해서 안정성을 보장한다. 그러나 이 경우는 시스템의 안정성외의 그 조건도 충족시켜주지 않는다. 다음 결과로서 점선은 Jadbabaed의 결과를 보여주는데, 이 경우 주목할 만한 것은 제어기 이득이 다른 경우의 결과보다 매우 작다는 것이다. 그러나 이것 역시 안정성과 안정성을 보장하는 영역이 작다는 큰 약점이 있다. 이에 반하여 실선으로 표현된 본 연구의 결과는 본 연구에서 충분히 보장한 안정성과 상태와 입력 모두의 에너지를 고려한 결과를 보여준다.

아울러 초기치에 무관한 제어기를 설계할 수도 있는데 이 경우 유사정리 4. 1에 의하면 쉽게 구해진다. 그 이득은 다음과 같다.(시뮬레이션 결과는 생략)

$$K=[0.1719\ \ 0.0536],\ K_1=[5348.6\ \ 1632.0],\ K_2=[810.270\ \ 247.24].$$

(39)

마지막으로 그림 5.4에 있는 시뮬레이션 결과는 상태와 입력의 에너지 고려 문제에 있어서 입력변수의 에너지를 자유변수로 두고 시뮬레이션 한 결과이며 이 경우 오히려 안정성만 고려한 Wang의 결과보다 더 우수한 성능을 보임을 증명한다.

다음은 이 경우 제어기 이득과 최적의 입력변수 가중치 이다.(Q=3일 때)

$$K=[11282\ \ 715],\ K_1=[17370\ \ 1100],\ K_2=[7377.7\ \ 467.8],\ R=1.0014\times10^{-9}.$$

(40)

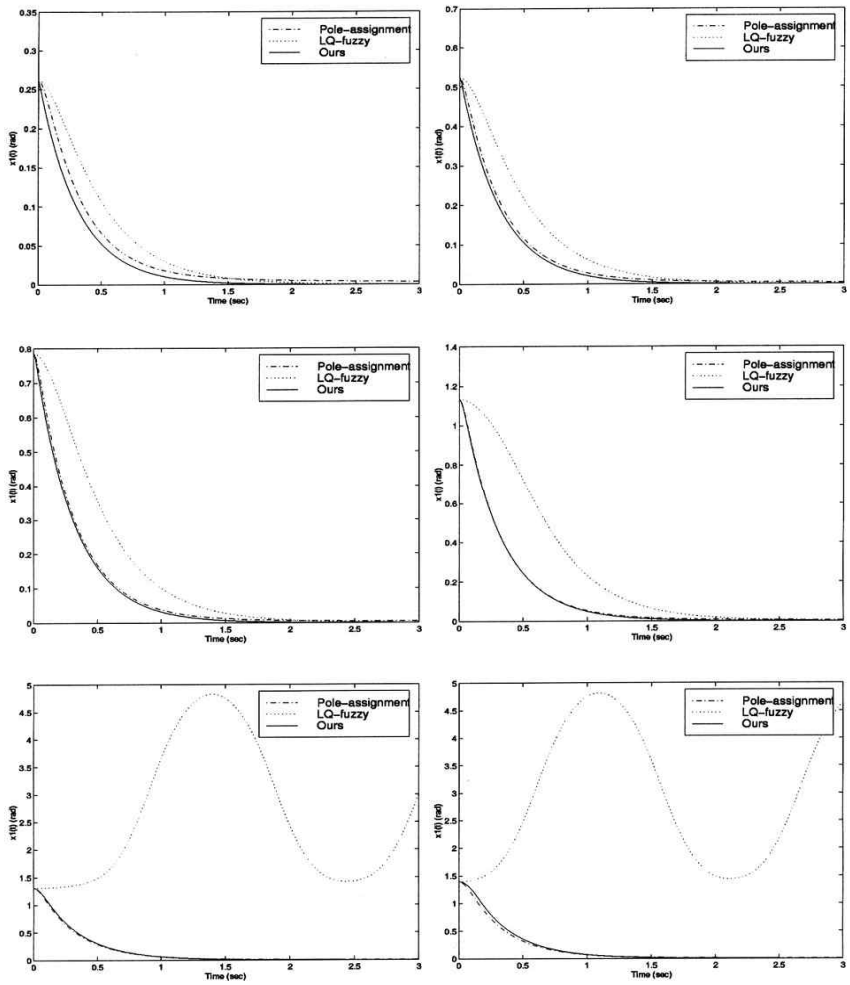
6. 결 론

본 연구에서는 기존의 존재하는 결과보다 좀더 유연한 파라메타 종속형 비선형 시스템에 대한 안정성 해석을 제안했다. 또한 이와 같은 안정성 해석을 바탕으로 본 연구에서는 새로운 guaranteed LQ-cost 제어기 설계 기법을 제안했다.

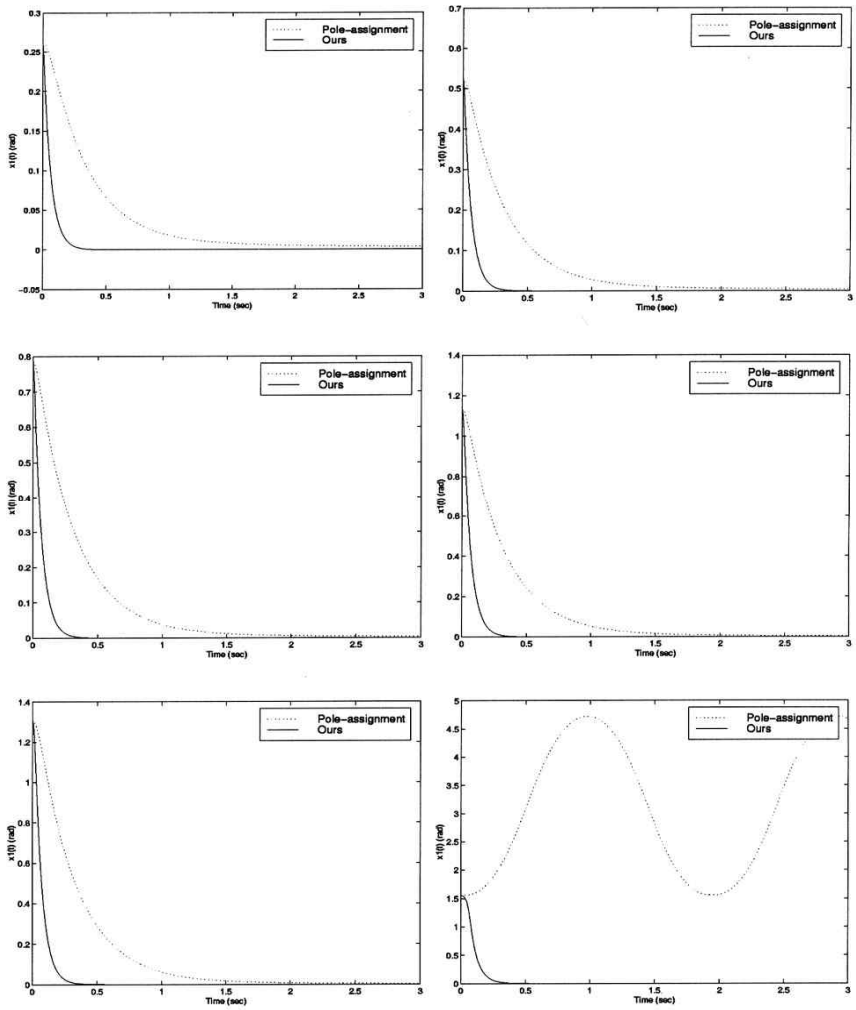
실제 시스템은 거의 모두가 비선형 시스템이다. 그리고 이러한 시스템을 위해서는 이득 보간법(Gain-Scheduling)이 좋은 해가 된다. 본 연구의 기본도 이러한 이득 보간법에 바탕을 둔다. 가중치 함수와 국소 선형성을 이용한 보간법의 개념으로 일반적인 비선형 시스템으로부터 새로운 파라메타 종속형 시스템을 추출한다. 그리고 그러한 구조의 시스템에 대하여 새로운 안정성해석을 하였고, 이러한 조건의 해를 구하기 위해서 LMI를 이용 feasible한 해를 구했다. 더불어 에너지 문제를 고려하기 위해서 guaranteed LQ cost 개념을 도입하여 실제적인 제어기 구성

문제를 취급하였다.

본 연구의 결과는 비선형 문제에서 널리 사용되고 있는 Inverted pendulum 예제에 적용을 하고 다른 결과들과 비교하여 그 성능을 보였다.



(그림 5.3 : 시뮬레이션 결과1 , 실선-점선: Pole-assignment fuzzy, 점선: Guaranteed LQ cost fuzzy, 실선: Guaranteed LQ cost control with new parameter stabilizing criteria at $x_1(0) = 15^\circ, 30^\circ, 45^\circ, 65^\circ, 75^\circ, 80^\circ$)



(그림 5.4 : 시뮬레이션 결과2 , 점선: Pole-assignment fuzzy, 실선: Guaranteed LQ cost control with new parameter stabilizing criteria at $x_1(0) = 15^\circ, 30^\circ, 45^\circ, 65^\circ, 75^\circ, 89^\circ$)

References

Li-Xin Wang : A Course in Fuzzy Systems and Control, 1997, Prentice Hall

R. Kruse, J. Gebhardt, F. Klawonn : Foundation of Fuzzy System, 1994, John Wiley & Sons

Lefteri H. Tsoukalas, Robert E. Uhring : Fuzzy and Neural approaches in Engineering, 1997, John Wiley & Sons

Bart Kosko: Fuzzy Engineering, 1997, Prentice Hall

Mamdani, E. H. [1974], "Applications of fuzzy algorithms for simple dynamic plant", Proc. IEE, 121, 12, pp. 1585-1588

Zadeh, L.A. [1962], "From circuit theory to system theory", Proc. Institution of Radio Engineers, 50, pp. 856-865

Zadeh, L.A. [1968], "Fuzzy algorithms", Information and Conntrol, 12, no. 2, pp 94- 102

Zadeh, L.A. [1973], "Outline of a new approach to the analysis of complex systems and decision process", IEEE Trans. on Systems, Man, and Cybern, 3, 1, pp 28-44

부록 A. 함수 리스트 (List of Functions)

이번 절에서는 퍼지 제어 툴박스에 포함된 모든 함수들을 자세히 설명하고 있다. 먼저 기능별로 묶어 정리해 두었고, 이후 알파벳순으로 상세한 설명을 했다. 셈툴 실행 중에도 도움말 기능을 이용하여 함수에 대한 설명을 얻을 수 있다.

Fuzzy System Creation and Setting	
fuzzy	Invoke fuzzy system editor
newfcs	create a new fuzzy system
loadfcs	load a fuzzy system from disk
savefcs	save the fuzzy system to disk
setfcs	set the fuzzy system properties
addvar	add a new fuzzy variable
rmvar	remove the fuzzy variable
addmf	add a new membership function
rmmf	remove the membership function
editmf	edit the properties of membership function
addrule	add fuzzy rules
rmrule	remove rules

Evaluation Functions	
evalfcs	evaluate fuzzy system output given input
evalfv	evaluate multiple membership functions
evalmf	evaluate the membership function

Membership Functions	
dsigmf	difference of two sigmoid functions
gauss2mf	two-sided Gaussian curve
gaussmf	Gaussian membership function
gbellmf	generalized bell-shaped curve
pimf	PI-shaped curve
psigmf	product of two sigmoid function
smf	S-shaped curve
sigmf	Sigmoid curve
trapmf	trapzoidal membership function
trimf	triangular membership function
zmf	Z-shaped curve

Neuro Fuzzy Functions	
anfis	training routine for Sugeno-type fcs
falcon	training routine for Mamdani-type fcs

Miscellaneous Functions	
showvar	display input/output variables
showmf	display membership functions
showrule	display rules
showfcs	display fuzzy system properties
nvar	return the #s of inputs and outputs
nmf	return the # of membership functions
nrule	return the # of fuzzy rules

addmf

▷ 목적

퍼지 변수에 소속함수를 추가한다.

▷ 문법

addmf(fs, "type1", "var", "mf", "type2", [params])
addmf(fs, type1, varindex, "mf", "type2", [params])

▷ 설명

퍼지 시스템의 퍼지 변수에 소속함수를 추가하는 함수.
fs는 퍼지 시스템. "type1"은 퍼지 변수의 종류로 "input", "output"의 값을 가진다. “var"는 퍼지 변수의 이름. ”mf"는 추가될 소속함수의 이름이며, 기존에 그 퍼지 변수에 존재하는 다른 소속함수들과 이름이 같으면 안된다. "type2"는 소속함수의 종류로서 dsigmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf 등이 있다. [params]는 각각의 소속함수의 특징을 결정짓는 파라미터들이다.
type1을 문자열 "input"/"output" 대신 1/0으로 쓸수도 있다. 그리고 퍼지 변수 이름대신 퍼지 변수의 인덱스를 써도 된다. 변수의 인덱스는 showvar 함수를 이용해 알 수 있다.

▷ 예제

```
newfcs("a","tipper","mamdani");
addvar("a","input","service",[0 10]);
addmf("a","input","service","poor","gaussmf",[1.5 0]);
addmf("a","input","service","good","gaussmf",[1.5 5]);
addmf("a","input","service","excellent","gaussmf",[1.5 10]);
showmf("a", "input", "service");
```

▷ 관련 함수

addvar, editmf, rmmf, dsigmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, and zmf

addrule

▷ 목적

퍼지 시스템에 룰을 추가한다.

▷ 문법

addrule(fs, rulelist)

▷ 설명

퍼지 시스템 fs에 퍼지 룰을 추가한다. rulelist는 입력의 개수가 n, 출력의 개수가 m인 시스템에 l개의 룰을 추가할 경우 그 사이즈가 $l \times (n+m+2)$ 인 행렬이 된다. 처음 n개의 column은 전건부(if part)를 나타내며 입력 퍼지 변수의 소속함수의 인덱스이다. 다음 m개의 column은 후건부(then part)를 나타내며 출력 퍼지 변수의 인덱스이다. 그 다음 column은 룰들의 weight를 나타내며 그 값은 0보다 크고 1보다 작거나 같은 양수 값이어야 한다. 맨 마지막 column은 전건부의 각 퍼지 변수들의 관계를 나타내며 그 관계들은 서로 'AND' 관계이거나 'OR'관계일 수 있다. 그 값이 1이면 'AND', 2이면 'OR'이다. 자세한 사항은 예제를 참고하기 바란다.

▷ 예제

```
loadfcs("a","tempcon.fcs")
addrule("a",[1 3 1 1; 2 2 1 1; 3 1 1 1]);
showrule("a","symbolic");
```

▷ 관련 함수

rmrule, showrule, nrule

addvar

▷ 목적

퍼지 시스템에 변수를 추가한다.

▷ 문법

addvar(fs, "type", "var", [Range] {,scale-factor})
addvar(fs, type, "var",[Range] {,scale-factor})

▷ 설명

퍼지 시스템에 입력 혹은 출력 변수를 추가한다. type은 문자열일 경우에는 "input"/"output", 숫자로는 1/0의 값을 가진다. "var"는 변수 이름으로 기존에 존재하는 변수의 이름과 다른 값을 가지면 된다.
Range는 변수가 의미를 가지는 구간으로, [구간시작, 구간끝]의 사이즈가 2인 벡터이다. scalefactor는 입력 변수의 pre-scaler, 출력변수의 de-scaler로 입/출력을 normalize할 때 사용하면 편리하다. 생략해도 무방한 값이며 생략시에는 1이 디폴트로 할당된다.

▷ 예제

```
newfcs("a","tipper","mamdani");
addvar("a","input","service",[0 10]);
addvar("a","output","tip",[10 100]);
showvar("a");
```

▷ 관련 함수

addmf, rmvar

anfis

※ 현재 개발 중인 함수입니다.

▷ 목적

TSK Type 퍼지 시스템 파라미터를 훈련시킨다.

▷ 문법

▷ 설명

▷ 예제

▷ 관련 함수

editmf

▷ 목적

소속함수의 파라미터들을 수정한다.

▷ 문법

editmf(fs, "type1", "var", "mf", "type2", [new params])
editmf(fs, type1, varindex, mfindex, "type2", [new params])

▷ 설명

소속함수의 파라미터를 수정한다. 기능적으로는 rmmf + addmf와 동일하나, 내부적으로 다른 함수를 사용하며 좀 더 빠르다. 인수로서 문자열이나 인덱스 모두 사용 가능하지만 둘을 섞어 쓰면 안 된다. 인덱스를 알기 위해서는 showvar 함수와 showmf함수를 이용하기 바란다. 각각의 인자들에 대한 설명은 addmf를 참고할 것.

▷ 예제

```
newfcs("a","tipper","mamdani");
addvar("a","input","service",[0 10]);
addmf("a","input","service","poor","gaussmf",[1.5 0]);
addmf("a","input","service","good","gaussmf",[1.5 5]);
addmf("a","input","service","excellent","gaussmf",[1.5 10]);
msgprint("original mf");
showmf("a", "input", "service");
addmf("a","input","service","good","trimf",[0 5 10]);
msgprint("modified mf");
showmf("a", "input","service");
```

▷ 관련 함수

rmmf, addmf

evalfcs

▷ 목적

주어진 입력에 대하여 퍼지 시스템의 출력을 구한다.

▷ 문법

y = evalfcs(fs, x)

▷ 설명

주어진 입력 x에 대한 출력을 반환한다. x, y는 vector이며 사이즈는 각각 입력변수의 개수, 출력변수의 개수와 동일하다. 시스템의 입/출력 변수의 개수를 알기 위해서는 nvar 함수를 사용하면 된다.

▷ 예제

```
loadfcs("a","온도조절기.fcs")
for(i = 20; i <= 40; i++) evalfcs("a", i )
```

▷ 관련 함수

nvar, evalfv, evalmf

evalfv

▷ 목적

여러 개의 소속함수 값을 한번에 계산한다.

▷ 문법

y = evalfv(fs, "type", "var", x)

y = evalfv(fs, type, varindex, x)

▷ 설명

주어진 스칼라 입력 x에대한 퍼지 변수- "type"과 “var"로 규정되는 -에 소속된 모든 소속함수의 적합도를 계산하여 반환한다. y는 그 퍼지 변수에 소속된 소속함수의 개수와 동일한 사이즈의 벡터 값이된다. 소속함수의 개수를 알아보려면 showmf 혹은 nmf 함수를 사용하면 된다.

▷ 예제

loadfcs("a","온도조절기.fcs")

evalfv("a", “input", "온도”, 26)

▷ 관련 함수

evalfcs, evalmf, nmf

evalmf

▷ 목적

소속함수의 값을 계산한다.

▷ 문법

y = evalmf(fs, "type1", "var", "mf", x)

y = evalmf(fs, type1, varindex, mfindex, x)

▷ 설명

퍼지 시스템에 속한 소속함수의 적합도를 계산한다. 입력 및 출력은 모두 스칼라 값이다. 각각의 인자들에대한 설명은 addmf함수의 설명을 참고할 것.

퍼지 시스템에 소속된 소속함수가 아닌 함수들에대한 값을 구하려면 evalmf대신 직접 소속함수값을 계산하는 것이 좋다. 셈툴에서는 많이 쓰는 소속함수로서 dsigmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf 및 zmf를 제공한다.

▷ 예제

```
loadfcs("a","온도조절기.fcs");
onplot(1,1);
onminmax(1,20,40,0,1);
startplot
for(i = 20; i <= 40; i++)
element(1, i, evalmf( "a", "input", "온도","low", i))
endplot
```

▷ 관련 함수

evalfcs, evalfv, dsigmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf

dsigmf

▷ 목적

두 개의 sigmoidal 함수의 차이값을 구한다.

▷ 문법

```
y = dsigmf(x, params)
```

▷ 설명

두 S자형 MF(membership function)의 차이의 절대값을 반환한다. PARAMs는 네개의 요소로 이루어진 벡터이며 이 MF의 모양과 위치를 결정한다. SIGMF를 이용하여 DSIGMF를 나타내면, 다음과 같은 식이 된다.

```
DSIGMF(X, PARAMS)
  = ABS(SIGMF(X, PARAMS(1:2)-SIGMF(X, PARAMS(3:4))))
```

▷ 예제

```
x = (0:10:0.2)';
params1 = [5 2];
y1 = sigmf(x, params1);
params2 = [2 7];
y2 = sigmf(x, params2);
y3 = dsigmf(x, [params1 params2]);
subplot(2,1,1); plot(x,y1,"r",x,y2,"b");
subplot(2,1,2); plot(x,y3,"B");
```

▷ 관련 함수

evalmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf

falcon

※ 현재 개발 중인 함수입니다.

▷ 목적

Mamdani Type 퍼지 시스템 파라미터를 훈련시킨다.

▷ 문법

▷ 설명

▷ 예제

▷ 관련 함수

fuzzy

▷ 목적

퍼지 시스템 에디터를 실행시킨다.

▷ 문법

fuzzy

fuzzy(fs)

▷ 설명

퍼지 시스템 GUI 에디터를 실행한다. 인자 없이 실행시켜도 되고, 인자로서 퍼지 시스템 변수 이름을 사용해도 된다. 퍼지 시스템 변수 fs를 인자로 사용하면 그 변수를 수정하게 된다.

퍼지 에디터는 퍼지 시스템을 디자인하기 위한 비쥬얼툴이다. 퍼지 시스템에 입/출력 변수를 추가/삭제하거나, 소속함수를 추가/삭제하고, 또한 퍼지 룰을 추가/삭제 할 수 있다. 자세한 설명은 2장 Reference의 두 번째 섹션을 참고하기 바란다.

▷ 예제

fuzzy

▷ 관련 함수

newfcs, loadfcs

gauss2mf

▷ 목적

2-sided Gaussian curve 모양의 소속함수 값을 계산한다.

▷ 문법

y = gauss2mf(x, params)

▷ 설명

x에서 구해진 Gaussian-constructed MF(Membership function)의 값을 반환한다. PARAMS = [S1 C1 S2 C2]는 네개의 요소로 된 벡터이며, S1,C1은 MF의 왼쪽 어깨부분을 정의하고, S2, C2는 오른쪽 어깨부분을 정의한다.

▷ 예제

```
x = (0:10:0.1)';
y1 = gauss2mf(x, [2 4 1 8]);
y2 = gauss2mf(x, [2 5 1 7]);
y3 = gauss2mf(x, [2 6 1 6]);
y4 = gauss2mf(x, [2 7 1 5]);
y5 = gauss2mf(x, [2 8 1 4]);
plot(x,y1,"r", x,y2,"g", x,y3,"b", x,y4,"B", x,y5,"y");
```

▷ 관련 함수

dsigmf, evalmf, gaussmf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf

gaussmf

▷ 목적

Gaussian curve모양의 소속함수 값을 계산한다.

▷ 문법

y = gaussmf(x, params)

▷ 설명

x에서 계산된 가우시안 멤버쉽펑션값을 반환한다. 파라미터는 2-요소로 된 벡터이며 멤버쉽펑션의 모양과 위치를 결정한다. 그리고, 아래와 같은 수식으로부터 계산된다.

GAUSSMF(X, [SIGMA, C])

= EXP(-(X - C).^2/(2*SIGMA^2))

▷ 예제

```
x = (0:10:0.1)';
y1 = gaussmf(x, [0.5 5]);
y2 = gaussmf(x, [1 5]);
y3 = gaussmf(x, [2 5]);
y4 = gaussmf(x, [3 5]);
subplot(2,1,1);
plot(x,y1,"r", x,y2,"g", x,y3,"b", x,y4,"B");
y1 = gaussmf(x, [1 2]);
y2 = gaussmf(x, [1 4]);
y3 = gaussmf(x, [1 6]);
y4 = gaussmf(x, [1 8]);
subplot(2,1,2);
plot(x,y1,"r", x,y2,"g", x,y3,"b", x,y4,"B");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf

gbellmf

▷ 목적

일반화된 종 모양의 소속함수 값을 계산한다.

▷ 문법

```
y = gbell( x, params )
```

▷ 설명

일반화된 종모양의 멤버쉽 함수의 x에서 계산된 값을 반환한다. 파라미터는 3-요소의 벡터로서 모양과 위치를 결정한다. 수식은 다음과 같다.

```
GBELLMF(X, [A, B, C])
        = 1./((1+ABS((X-C)/A))^(2*b));
```

참고로, 이 멤버쉽함수는 Cauchy probability dist. function의 확장이다.

▷ 예제

```
x = (0:10:0.1)';
y1 = gbellmf(x, [1 2 5]);
y2 = gbellmf(x, [2 4 5]);
y3 = gbellmf(x, [3 6 5]);
y4 = gbellmf(x, [4 8 5]);
subplot(2,1,1);
plot(x,y1,"r",x,y2,"g",x,y3,"b",x,y4,"B");
y1 = gbellmf(x, [2 1 5]);
y2 = gbellmf(x, [2 2 5]);
y3 = gbellmf(x, [2 4 5]);
y4 = gbellmf(x, [2 8 5]);
subplot(2,1,2);
plot(x,y1,"r",x,y2,"g",x,y3,"b",x,y4,"B");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gbellmf, pimf, psigmf, smf, sigmf, trapmf, trimf, zmf

loadfcs

▷ 목적

파일로부터 퍼지 시스템 변수를 읽어 들인다.

▷ 문법

loadfcs(fs, "filename")

▷ 설명

disk파일로부터 퍼지 시스템 변수를 읽어 들인다. “filename"은 확장자까지 포함한 문자열이다. 퍼지 시스템 파일의 디폴트 확장자는 *.fcs 이다.

만약 workspace상에 똑같은 이름의 퍼지 시스템 변수가 존재하면 파일로부터 읽어들인 퍼지 시스템 정보로 대치된다.

▷ 예제

loadfcs("a", “온도조절기.fcs")

▷ 관련 함수

savefcs

newfcs

▷ 목적

새로운 퍼지 시스템을 만든다.

▷ 문법

newfcs(fs, "description", "type")

▷ 설명

새로운 퍼지 시스템 변수를 Workspace상에 만든다. "description"은 퍼지 시스템에대한 설명 혹은 묘사를 뜻하는 문자열이다. "type"은 “mamdani"와 ”sugeno" 두 가지 값을 가지며, 각각 Mamdani Type과 Takagi-Sugeno-Kang Type의 퍼지 시스템을 의미한다.

▷ 예제

newfcs("a","Motor Controller", "sugeno")

▷ 관련 함수

setfcs

nmf

▷ 목적

퍼지 변수의 소속함수의 개수를 반환한다.

▷ 문법

y = nvar(fs, "type", "var")

y = nvar(fs, type, varindex)

▷ 설명

퍼지 변수에 포함된 소속함수의 개수를 반환한다. 반환값은 스칼라 값이다.

▷ 예제

loadfcs("a","온도조절기.fcs");

num_of_mf = nmf("a","input","온도“)

▷ 관련 함수

nvar, nrule, showmf, showvar

nrule

▷ 목적

퍼지 시스템의 퍼지 룰의 개수를 반환한다.

▷ 문법

y = nurle(fs)

▷ 설명

퍼지 시스템 내에 정의된 if-then룰의 개수를 반환한다. showrule, rmrule등에 사용되는 룰 인덱스의 범위를 체크할 때 사용된다.

▷ 예제

```
loadfcs("a","온도조절기.fcs");
num_of_rule = nrule("a")
```

▷ 관련 함수

nmf, nvar, showrule, rmrule

nvar

▷ 목적

퍼지 시스템의 입/출력 변수의 개수를 반환한다.

▷ 문법

y = nvar(fs)

▷ 설명

퍼지 시스템 fs의 입력 및 출력의 개수를 반환한다. 반환값 y는 사이즈가 2인 벡터이며, 첫째 요소는 입력의 개수, 두 번째 요소는 출력의 개수를 나타낸다. 각 변수에 대한 자세한 정보는 showvar함수를 사용하여 알 수 있다.

▷ 예제

```
loadfcs("a","온도조절기.fcs");
iosize = nmf("a")
```

▷ 관련 함수

nmf, nrule, showvar

pimf

▷ 목적

Π 곡선 모양의 소속함수값을 계산한다.

▷ 문법

pimf(x, params)

▷ 설명

x에서 계산된 Pi모양의 멤버쉽함수값을 반환한다.
파라미터 Params=[A B C D]는 4-요소의 벡터이고 이 멤버쉽 함수의 beaking points를 결정짓는다. pimf는 smf와 zmf의 곱으로부터 계산된다.

```
PIMF(X, PARAMS)
      = SMF(X, PARAMS(1:2)).*ZMF(X, PARAMS(3:4))
```

▷ 예제

```
x = (0:10:0.1);
y1 = pimf(x, [1 4 9 10]);
y2 = pimf(x, [2 5 8 9]);
y3 = pimf(x, [3 6 7 8]);
y4 = pimf(x, [4 7 6 7]);
y5 = pimf(x, [5 8 5 6]);
plot(x,y1,"r",x,y2,"b",x,y3,"g",x,y4,"y",x,y5,"B");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gaussmf, gbellmf, psigmf, smf, sigmf, trapmf, trimf, zmf

psigmf

▷ 목적

Product of two sigmid curve 모양의 소속함수 값을 계산한다.

▷ 문법

psigmf(x, params)

▷ 설명

x에서 계산된 두개의 S자 모양 함수의 곱을 반환한다.
파라미터 Params는 4-요소 벡터로 멤버쉽 함수의 모양과 위치를 결정한다. 계산식은 아래와 같다.

```
PSIGMF(X, PARAMS)
      = SIGMF(X, PARAMS(1:2)).*SIGMF(X, PARAMS(3:4))
```

▷ 예제

```
x = (0:10:0.2)';
y1 = sigmf(x, [2 3]);
y2 = sigmf(x, [-5 8]);
y3 = psigmf(x, [2 3 -5 8]);
subplot(2,1,1);
plot(x,y1,"r",x,y2,"b");
subplot(2,1,2);
plot(x,y3,"B");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gaussmf, gbellmf, pimf, smf, sigmf, trapmf, trimf, zmf

rmmf

▷ 목적

소속함수를 삭제한다.

▷ 문법

rmmf(fs, "type1", "var", "mf")

rmmf(fs, type1, varindex, mfindex)

▷ 설명

퍼지 시스템의 선택된 퍼지 변수로부터 소속함수를 제거한다. 인수에 대한 설명은 addmf함수 설명을 참고하기 바란다.

▷ 예제

```
newfcs("a","tipper","mamdani");
addvar("a","input","service",[0 10]);
addmf("a","input","service","poor","gaussmf",[1.5 0]);
addmf("a","input","service","good","gaussmf",[1.5 5]);
addmf("a","input","service","excellent","gaussmf",[1.5 10]);
showmf("a", "input", "service");
rmmf("a","input","service","good");
showmf("a","input","service");
```

▷ 관련 함수

addmf, editmf

rmrule

▷ 목적

퍼지 룰을 삭제한다.

▷ 문법

rmrule(fs, ruleindex)

▷ 설명

퍼지 시스템의 룰을 삭제한다. ruleindex는 삭제할 룰의 인덱스이며, 룰 정보는 showrule 함수를 이용하여 알 수 있다.

▷ 예제

```
loadfcs("a","온도조절기.fcs");
showrule("a");
rmrule("a",2);
showrule("a");
```

▷ 관련 함수

showrule, addrule, nrule

savefcs

▷ 목적

퍼지 시스템 정보를 파일에 저장한다.

▷ 문법

savefcs(fs, "filename")

▷ 설명

퍼지 시스템 정보를 파일 이름이 "filename"인 disk file로 저장한다. 이 때 "filename"은 확장자를 포함한 문자열이어야 한다.

▷ 예제

savefcs("a","myfuzzy.fcs")

▷ 관련 함수

loadfcs, newfcs, fuzzy

setfcs

▷ 목적

퍼지 시스템의 여러 가지 특징을 세팅한다.

▷ 문법

setfcs(fs, "arg1", "value1", "arg2", "value2", ...)

▷ 설명

퍼지 시스템의 여러 가지 특징을 세팅한다. “arg#“은 "name", ”and", "or", "imp", "agg", "def" 가 될 수 있으며 그 값들은 아래 표와 같이 될 수 있다.

arg#	meaning	value#
"name"	decription	description of the fuzzy system
"and"	and method	"pro", "min"
"or"	or method	"sum", "max"
"imp"	implication	"pro", "min"
"agg"	aggregation	"sum", "max"
"def"	defuzzification	"cog", "coa", "som", "lom", "mom" (Madani)
		"wtave","wtsum" (TSK type)

"pro"와 "min"은 각각 t-norm operation에서 product와 minimum operation을 의미하며, “sum", "max"는 s-norm operation에서 summation과 maximum operation을 의미한다.

defuzzificaton에서는 각각 "cog" - ceter of gravity, "coa" - center of average, "som" - smallest of maximum, "lom" - largest of maximum, "mom" - median of maximum을 의미하며 이상의 다섯가지 비퍼지화 방법은 Mamdani type 퍼지 시스템에서 사용된다. TSK타입의 경우에는 "wtave" - weighted average, "wtsum" - weighted sum 의 두 가지 비퍼지화 방법을 제공한다.

이상에 대한 자세한 수식은 제 1장 Tutorial의 두 번째 섹션을 참고하기 바란다.

▷ 예제

```
loadfcs("a","온도조절기.fcs");
showfcs("a");
setfcs("a","and","pro","agg","sum","def","cog");
sowfcs("a");
```

▷ 관련 함수

newfcs, fuzzy

showfcs

▷ 목적

퍼지 시스템의 정보를 보여준다.

※ 현재 버전의 셈툴에서 퍼지 시스템을 구조체 변수로 표현하지 못하여 임시로 만든 함수이다.

▷ 문법

showfcs(fs)

▷ 설명

퍼지 시스템 fs의 이름 및 성질, 입출력 개수, 룰 개수 등의 정보를 간략하게 보여준다.

▷ 예제

```
loadfcs("a","온도조절기.fcs");
showfcs("a");
```

▷ 관련 함수

setfcs, showvar, showrule

showmf

▷ 목적

퍼지 변수의 소속함수들을 보여준다.

▷ 문법

showmf(fs, "type", "var")

showmf(fs, type, varindex)

▷ 설명

퍼지 시스템 fs의 변수 “var”에 속한 소속함수들을 보여준다. “type"은 ”input"/"output"의 값을 가지며, 변수의 이름대신 인덱스를 사용할 때는 type은 1/0의 값을 가지게 된다. (각각 입/출력 변수의 의미)
표시되는 값들은 차례로, 이름 - 소속함수 종류 - 파라미터이다.

▷ 예제

```
newfcs("a","tipper","mamdani");
addvar("a","input","service",[0 10]);
addmf("a","input","service","poor","gaussmf",[1.5 0]);
addmf("a","input","service","good","gaussmf",[1.5 5]);
addmf("a","input","service","excellent","gaussmf",[1.5 10]);
showmf("a", "input", "service");
```

▷ 관련 함수

addmf, rmmf, showvar, showrule

showrule

▷ 목적

퍼지 시스템의 룰을 보여준다.

▷ 문법

showrule(fs, {ruleindex, {“format"}})

▷ 설명

퍼지 시스템 fs에 포함된 룰들을 보여준다. ruleindex는 보고자하는 룰의 인덱스이며 스칼라 혹은 벡터값이다. 생략하면 모든 룰을 보여준다. “format"은 룰을 "symbolic"(언어적 표현) 및 ”indexed“(숫자 인덱스)의 두 가지 값을 가질 수 있으며 기본 값은 "symbolic"이며 생략 가능하다. 맨 마지막 괄호 안에 표시되는 것은 룰 weight이다.

▷ 예제

```
loadfcs("a","tempcon.fcs")
addrule("a",[1 3 1 1; 2 2 1 1; 3 1 1 1]);
showrule("a");
showrule("a","indexed");
```

▷ 관련 함수

addrule, rmrule, showmf, showvar

showvar

▷ 목적

입/출력 퍼지 변수들을 보여준다.

▷ 문법

showvar(fs)

▷ 설명

퍼지 시스템 fs에 소속된 입력 및 출력 퍼지 변수들의 인덱스, 이름 및 Range와 Scale factor들을 보여준다.

▷ 예제

```
newfcs("a","tipper","mamdani");
addvar("a","input","service",[0 10]);
addvar("a","output","tip",[10 100]);
showvar("a");
```

▷ 관련 함수

addvar, rmvar, nvar, showrule, showmf

sigmf

▷ 목적

Sigmoid curve모양의 소속함수 값을 계산한다.

▷ 문법

sigmf(x, params)

▷ 설명

x에서 계산된 S자 모양의 멤버쉽 함수값을 반환한다.

파라미터 Params는 2-요소 벡터이다. 수식은 아래와 같다.

```
SIGMF(X, [A, C])
      = 1./(1 + EXP(-A*(X-C)))
```

▷ 예제

```
x = (0:10:0.2)';
y1 = sigmf(x, [-1 5]);
y2 = sigmf(x, [-3 5]);
y3 = sigmf(x, [4 5]);
y4 = sigmf(x, [8 5]);
subplot(2,1,1);
plot(x,y1,"r",x,y2,"b",x,y3,"g",x,y4,"B");

y1 = sigmf(x, [5 2]);
y2 = sigmf(x, [5 4]);
y3 = sigmf(x, [5 6]);
y4 = sigmf(x, [5 8]);
subplot(2,1,2);
plot(x,y1,"r",x,y2,"b",x,y3,"g",x,y4,"B");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, smf, trapmf, trimf, zmf

smf

▷ 목적

S자 모양의 소속함수 값을 계산한다.

▷ 문법

smf(x, params)

▷ 설명

x에서 계산된 S자 모양 멤버쉽함수값을 반환한다. 파라미터 params=[X0 X1]은 두개의 요소로 이루어진 벡터이며, 이 함수의 breaking points를 결정한다.
X0 < X1일때, 0에서 1로의 부드러운 곡선모양이고, X0 >= X1이면 (X0+X1)/2에서 0에서 1로 변하는 스텝함수가 된다.

▷ 예제

```
x = 0:10:0.1;
plot(x, smf(x,[2 8]),"m",x,smf(x,[3,7]),"b");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, sigmf, trapmf, trimf, zmf

trapmf

▷ 목적

사다리꼴 모양의 소속함수 값을 계산한다.

▷ 문법

trapmf(x, params)

▷ 설명

x에서 계산된 사다리꼴 모양의 멤버쉽함수값을 반환한다.
파라미터 Params는 4-요소 벡터로 멤버쉽 함수의 break points를 결정한다. Params=[a b c d]는 a<=b c<=d이어야 한다. 아니면 1보다 작은 삼각형모양의 곡선이 된다. (예제 참고)

▷ 예제

```
x = (0:10:0.1);
y1 = trapmf(x, [2 3 7 9]);
y2 = trapmf(x, [3 4 6 8]);
y3 = trapmf(x, [4 5 5 7]);
y4 = trapmf(x, [5 6 4 6]);
plot(x,y1,"r",x,y2,"b",x,y3,"g",x,y4,"B");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, sigmf, smf, trimf, zmf

trimf

▷ 목적

삼각형 모양의 소속함수값을 계산한다.

▷ 문법

trimf(x, params)

▷ 설명

x에서 계산된 삼각형모양의 멤버쉽 함수값을 반환한다.
파라미터 Params=[a b c]는 3-요소 벡터로 멤버쉽함수의 break points를 결정한다. 보통 a <= b <= c이다.

▷ 예제

```
x = [0:10:0.2];
y1 = trimf(x, [3 4 5]);
y2 = trimf(x, [2 4 7]);
y3 = trimf(x, [1 4 9]);
subplot(2,1,1);
plot(x, y1,"r",x, y2,"g",x, y3,"b");
y1 = trimf(x, [2 3 5]);
y2 = trimf(x, [3 4 7]);
y3 = trimf(x, [4 5 9]);
subplot(2,1,2);
plot(x, y1,"r",x, y2,"g",x, y3,"b");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, sigmf, smf, trapmf, zmf

zmf

▷ 목적

Z자 모양의 소속함수값을 계산한다.

▷ 문법

zmf(x, params)

▷ 설명

x에서 계산된 Z-모양의 멤버쉽 함수값을 반환한다.
파라미터 Params=[X0,X1]는 멤버쉽 함수의 break points를 결정한다.

▷ 예제

```
x = [0:10:0.1];
y1 = zmf(x,[2 8]);
y2 = zmf(x,[4 6]);
y3 = zmf(x,[6 4]);
plot(x, y1,"m",x,y2,"b",x,y3,"g");
```

▷ 관련 함수

dsigmf, evalmf, gauss2mf, gaussmf, gbellmf, pimf, psigmf, sigmf, smf, trapmf, trimf

Index

- functions

- examples