論文96-33B-3-3

전용 하드웨어로 구성한 FLC에 적합한 새로운 자기동조 알고리즘

(A Novel Self-tuning Algorithm Suitable for FLCs Utilizing Dedicated Hardwares)

李 承 夏 **, 卞 增 男 *

(Seung-ha Lee and Zeungnam Bien)

요 약

퍼지제어기(FLC)를 실제로 구현하는 방법으로, 앞으로는 퍼지 추론 전용의 하드웨어가 많이 이용될 전망이다. 기존의 퍼지 하드웨어에는 단순한 추론계산 외에, 자기동조 기능과 같이 퍼지 제어기에 있어 유용한 알고리즘이 구현된 것은 찾기 힘들다. 이것은 전용 하드웨어로 구성된 FLC에 기존의 자기동조 방법을 그대로 하드웨어로 구현하는 것이 어렵거나 바람직하지 않기 때문이다. 따라서, 자기동조 기능을 갖는 퍼지 하드웨어를 구현하기 위해서는 우선, 간단히 회로로 구현가능한 새로운 FLC 자기동조법이 요구된다. 본 논문에서는 이와 같은 목적에 맞는 하나의 FLC 자기동조법을 제안한다. 이 알고리즘은 기본적으로는 제어기 입력의 유효 범위를, 전체 시스템이 보다 더 나은 성능을 보이도록 탐색의 방법으로 찾아 나가는 과정이다. 하드웨어로 구현하기 용이한 성능평가 방법과 더불어 기존의 스케일링 인자만을 조정하는 방법에 옵셋 조정을 더함으로써 퍼지 규칙을 변화시키는 것과 같은 효과를 내는, 제안된 알고리즘의 성능은 모의실험을 통해 검증하였다.

Abstract

More fuzzy hardwares are expected to be utilized in the future to construct fuzzy logic controllers(FLCs). It is hard to find an existing fuzzy hardware which is adopting advanced functions such as self-tuning algorithm in addition to the conventional inferenece calculation. That is mainly because conventional self-tuning algorithms can not be utilized or may not be appropriate for hardwarization. Therefore, an algorithm designed to implement with some hardware circuits is required for fuzzy hardwares to have self-tuning capability. As a first step toward the feature, a novel self-tuning algorithm is proposed in this paper. Based on the search method, the main idea of the proposed algorithm is to determine valid ranges of input variables of an FLC in order to maximize performance indices of the control system. The performance indices are so simple as to be realized by hardware circuit. In addition to the conventional scaling-factor adjustment, the algorithm adjusts offset values as well, which, in effect, modifies fuzzy rules of the FLC. To justify the performance of the proposed algorithm, a simulation study is executed.

* 正會員, 韓國科學技術院 電氣 및 電子工學科

(Dept. of Elec. Eng. Korea Advanced Institute of Science and Technology)

** 正會員, 慶北大學校 電子電氣工學部

(Dept. Electronic & Electrical Eng., Kyungpook Nat'l Univ.)

※ 본 논문은 1994년도 한국과학재단 연구비 지원에 의해 연구되었음

接受日字:1995年4月8日, 수정완료일:1995年12月13日

I. 서 론

퍼지제어기(Fuzzy Logic Controller, FLC)를 이용해서 제어기를 구성하는 경우에 여러가지 장점을 가지고 있으나, 그 핵심이라 할 수 있는 퍼지 규칙과 소속함수를 구하는 체계적인 방법이 없음은 잘 알려진 사실이다.

이를 해결하는 방법으로 여러가지 방법, 즉, 자동 규칙생성[1, 2, 3, 4, 5], 자기동조(Self-tuning) FLC 등[6, 7, 8, 9, 10, 11, 12, 13, 14, 15]이 연구되어왔다. 한편,

퍼지제어기를 실제로 구성하는 방법에는 크게 3가지를 들 수 있다. 즉, 범용 프로세서에 소프트웨어 프로그램으로 구현하는 방법 [16], 메모리 소자를 이용해 룩업 테이블 방식으로 구현하는 방법 [17], 전용 하드웨어를 이용하는 방법[18, 19] 등이 그것이다.

이 중에서 전자의 두 방법이 각각, 추론 속도가 느리다는 점과 제어 정도(resolution) 에 따라 요구되는 메모리의 크기가 기하 급수적으로 커지는 단점 등으로 해서 후자의 전용 하드웨어를 쓰는 방법에 대한 연구가 많이 있었다. 현재에는 칩의 형태로, 혹은 보드의 형태로 된 이러한 퍼지 추론 전용의 하드웨어가 상용으로 구입가능하며 종류도 다양하다[20]. 전용의 하드웨어를 이용한 이러한 퍼지 제어기구성의 경우에는 그 구조가 간단하고 속도가 빠르며 가격면에서도 유리하여 앞으로는 가전제품이나 산업용 프로세스를 위한 퍼지 제어기 구성에서 더욱 많이 이용될 전망이다.

퍼지추론 전용의 하드웨어는 크게 아날로그 형과 디지탈 형으로 또는 코프로세서 형과 stand-alone 형으로 나뉠 수 있는데 기존의 제품들은 프로그램 가능성(programmablility) 및 기존의 디지탈 프로세서와의 인터페이스가 용이한 장점 때문에 디지탈 형(type) 이면서 코프로세서 형이 대부분이다. 이와 같은 코프로세서 형의 하드웨어를 이용한 FLC의 개략적인 구조는 그림 1과 같다. 이 구조는 퍼지 하드웨어가 주 프로세서의 제어 명령을 받아 퍼지 추론을 담당하고 그 과정이 끝나면 주 프로세서가 결과를 읽어가는 방식의 동작 방식을 취한다.

그런데, 빠른 처리를 위해 전용 하드웨어로 구성한 퍼지 제어기의 경우에도 제어기를 설계하는 과정에서 앞서 언급한 자기동조법 등은 여전히 필요하게 된다.

만일, 전용의 하드웨어 프로세서를 쓰는 퍼지 제어기에 있어 추론결과의 계산과정에다 퍼지제어기의 자기동조 과정도 하드웨어화 된다면 가장 바람직한 퍼지 하드웨어의 형태가 될 것이다. 그러나 기존의 제품에는 그러한 기능을 가진 선례를 찾기 힘들다. 이것은 하드웨어 퍼지 제어기에 대해 기존의 자기 구성 퍼지제어기나 기타의 자기동조 알고리즘을 적용하기가 쉽지 않기 때문이다.

먼저, 온-라인 동조 알고리즘은 대부분 각 샘플링 순간에 제어기의 관계 행렬이나 규칙, 소속함수의 위치등을 조정(adjust)하게 되는데 이는 하드웨어로 구현 할 때, 그 내부 메모리의 내용을 매 샘플링 순간에 변경하여야 한다. 이 경우, 규칙이나 하드웨어 메모리에 저장된 소속함수의 위치를 변경하는 것은 대개, 많은 양의 데이타를 한꺼번에 옮기는 것을 의미하며 따라서, 다소 긴 시간이 소요되어 빠른 추론을 목적으로 쓴 퍼지 하드웨어 본래의 목적에 위배된다. 예를 들어, [14]에서 입, 출력 이득 동조용의 또 다른 FLC를 통해 이득을 조정하는 방식의 경우, 이 과정에서 시간이 많이 걸리게 된다.

그리고 오프-라인 동조 알고리즘 역시 성능 평가 및 제어기 동조의 방법이 복잡한 과정을 거치게 되어 이를 간단히 하드웨어에 이식 시키는 것이 쉽지 않다. 예를 들어 제어 시스템의 실행 결과로 부터 상승시간, 정상상태 오차 등을 구하는 하드웨어를 만드는 것은 매우 힘든 일이다.

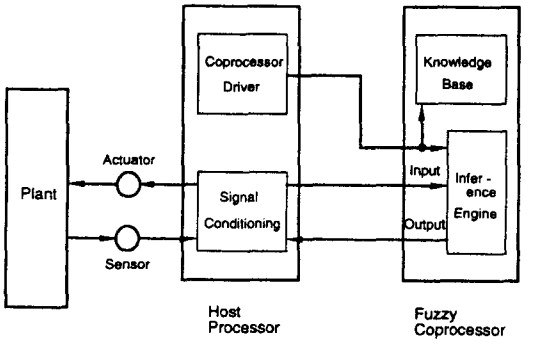


그림 1. 일반적인 코프로세서 형식의 하드웨어 퍼지제어기 개념도

Fig. 1. General structure of FLCs with coprocessor type hardware.

따라서, 자기동조 기능을 갖는 퍼지 하드웨어를 구현하기 위해서는 우선, 회로로 구현가능한 새로운 FLC 자기동조법이 요구된다. 본 논문에서는 퍼지제어기의 자기동조를 하드웨어화하기 위한 첫 단계로 그 구조가 간단하면서도 기존의 동조 알고리즘에 비해 성능면에서도 우수한 새로운 퍼지제어기 자기동조 방법을 제안하고자 한다.

제안된 알고리즘은 두 부분으로 나뉠 수 있는데, 첫째는 성능평가의 부분이고 둘째는 조정 방법에 관한 부분이다.

성능평가를 위해서 본 알고리즘에서는 규칙의 활화정도를 샘플링 시간별로 합산하고 이를 성능 지수로 한다. 여기에 이들 성능 지수중, 원하는 시스템 상태에 해당하는 규칙이 활화된 정도를 통해 시스템의 성능을

평가할 수 있다. 예를 들어 "If e is Zero and $\dot{e}$ is Zero then …"에 해당되는 규칙의 활화정도가 클 수록 보다 나은 성능으로 평가하는 것이다. 이렇게, 규칙의 활화 정도를 통해 시스템 상태의 궤적을 연관 짓고 이를 성능 지수로 선정하면 실제 하드웨어로 성능평가 부분을 구현하는데 매우 간단하게 되는 장점이 있다.

조정 방법으로서, 기존의 자기동조 퍼지 제어기들은 1)제어 규칙이나 2)소속함수, 3)입, 출력 스케일링 인자 혹은 4)이들의 조합을 이용하고 있다. 이 중에서 본 연구에서는 입력 스케일링 인자를 조정하는 방법을 통해 규칙과 소속함수를 조정하는 것과 같은 효과를 얻고자 한다. 이렇게 하면 퍼지 하드웨어의 내부 소속함수 및 규칙 메모리를 수정하지 않으면서 규칙과 소속함수를 바꾸는 것과 같은 효과를 얻을 수 있다.

그림 1에 보인 퍼지 제어기 구조에서 호스트 프로세서의 주요 기능은 센서나 구동기와의 통신을 통한 신호 정합(signal conditioning) 기능과 함께 퍼지 코프로세서와 내부 데이타 베이스 구축, 수행, 결과 얻어가기(retrieving) 등, 드라이버로서의 역할을 수행한다고 할 수 있다. 여기서 보면 퍼지 코프로세서 내부 혹은 그와 연결된 외부의 데이타 베이스를 수정하는 것이 많은 시간이 걸린다는 점과 센서와 구동부와의 인터페이스에는 어떤 형태든 신호 조절 기능이 필수적이라는 두가지 점에 주목하면 앞서 언급한 스케일링 인자 조정법이 적당함을 알 수 있다.

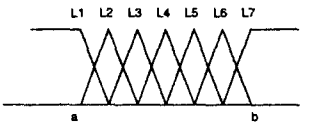
본 논문에서는 또한, 기존의 스케일링 인자의 조정법에 옵셋(offset)까지 조정하는 방법으로, 보다 일반화된 동조 방법을 제안한다. 즉, 기존의 스케일링 인자 조정법은 제안된 방법의 특별한 경우가 된다. 스케일링 인자의 조정시에 단순한 이득 뿐만아니라 옵셋을 조정하게 되면 PD 형의 퍼지제어기에서는 더욱 비선형적인 특성 변화를 줄 수 있고 성능개선의 폭도 크게 된다.

Ⅱ. 기본 가정 및 정의

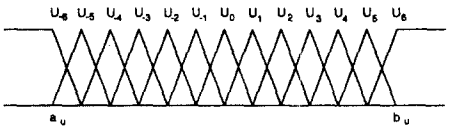
이 문제를 다루는 데 있어 다음과 같은 가정을 두고자 한다.

1. FLC는 일반적으로 많이 다루는 PD형, 즉, 입력으로 에러와 에러의 시간에 대한 미분이 있는 2입력 1출력 형으로 가정한다. 실제로, 많은 제어기가 이러한 PD 형으로 되어있다. 여기서 FLC 입력은 e 와 $\dot{e}$, 출력은 u로 표기하기로 한다.
2. 대상 플랜트는 제어입력의 극성에 대해 대칭의 성질을 갖는다고 가정한다.
3. 본 연구에서 다루고자 하는 문제는 레귤레이션 문제이다.
4. 최초의 퍼지 규칙은 전문가 혹은 앞 절에서 언급한 자동 규칙 생성 방법에 의해 주어졌다고, 그리고 오차가 줄어드는 성능을 보이는 것으로 가정한다.
5. 입, 출력 변수의 퍼지 분할과 전체 집합은 그림 2에 보인 것과 같고 퍼지추론은 MAX-MIN 방법, 비퍼지화 방법은 무게중심법을 가정한다.

이러한 가정의 대부분은 여타의 자기동조 알고리즘에서 가정하고 있는 것이다. 또한, 제어기 출력 스케일링 인자는 정해진 것으로 가정한다. 본 연구에서 관심을 갖는 것은 입력 스케일링 인자이다. 출력 스케일링 인자의 경우에는 단순히 폐루프 이득에만 직접적으로 영향을 주기 때문에 이에 대한 동조 알고리즘은 추후 연구과제로 남긴다.



(a)



(b)

그림 2. (a)입력 퍼지 분할, (b)출력 퍼지 분할 및 언어 레이블

Fig. 2. Fuzzy partitions and their linguistic labels: (a)input and (b)output

여기서 a 와 b 를 입력 변수의 유효한 전체집합의 하한 그리고 상한이라 하면, a_e 와 $a_{\dot{e}}$는 각각 입력변수 e와 $\dot{e}$의 하한 값으로, b_e 와 $b_{\dot{e}}$는 각각 e와 $\dot{e}$의 상한 값으로 표시 한다. 이때, $a \cdot b \leq 0$라고 가정한다.

분할의 갯수, n은 언어 레이블의 갯수에 1을 뺀 수로 하며 이 값은 짝수라 가정한다. 본 논문에서는

$n=6$ 으로 하였다. 그리고 폭, w와 옵셋, s는 다음과 같이 정의된다.

$$w = b - a \tag{1}$$

$$s = \frac{1}{2}w + a \tag{2}$$

$$- -\frac{1}{2}w + b \tag{3}$$

본 논문에서는 일반적으로 많이 쓰이는 표 1과 같은 규칙 테이블을 가정한다. 이러한 형태의 규칙은 Mac-Vicar-Whelan 행렬[21] 형으로 잘 알려진 것으로 출력 언어값이 좌상에서 우하에 이르는 대각선 방향으로 같은 U_0이고 이를 중심으로 대칭적인 특성을 같는 규칙 집합이다. 또한 우상(↗) 혹은 좌하(↙)로 갈수록 출력이 증가(U_6) 혹은 감소하는(U_{-6}) 경향을 띤 것이다.

표 1. MacVicar-whelan형의 규칙 테이블
Table 1. The rule table which is of Mac-Vicar-whelan type.

		e						
		L1	L2	L3	L4	L5	L6	L7
$\dot{e}$	L7	U_6	U_1	U_2	U_3	U_4	U_5	U_6
	L6	U_1	U_0	U_1	U_2	U_3	U_4	U_5
	L5	U_{-2}	U_1	U_0	U_1	U_2	U_3	U_4
	L4	U_{-3}	U_{-2}	U_1	U_0	U_1	U_2	U_3
	L3	U_4	U_{-3}	U_2	U_1	U_0	U_1	U_2
	L2	U_5	U_4	U_3	U_2	U_1	U_0	U_1
	L1	U_6	U_5	U_4	U_3	U_2	U_1	U_0

이러한 규칙에서 $e=\dot{e}=0$ 일 때 $u=0$ 의 출력이 나오기 위한 충분조건은 다음과 같다. 이는 입력 변수의 전체집합에 옵셋이 있는 경우 두 입력변수의 옵셋(s)과 전체집합의 폭(w)의 비율이 크기는 같고 부호가 반대가 되어야 함을 뜻한다.

$$\frac{s_e}{w_e} = -\frac{s_{\dot{e}}}{w_{\dot{e}}} \tag{4}$$

이 조건은 입력의 퍼지 분할이 그림 2와 같으면, 규칙 테이블에서 인접해 있는 최대 4개의 규칙이 활화된다는 사실로부터 식 (4)가 만족되고 입력이 $e=\dot{e}=0$ 일 때 결론부가 U_0, U_1, U_{-1} 인 최대 4개의 규칙 만이 활화되며, 이 중에서 결론부가 U_1, U_{-1}인 규칙의 활화정도가 같음을 보이고 출력 퍼지 소속함수가 U_0를 중심으로 대칭의 형태가 됨을 보임으로써 쉽게 증명할 수 있다.

한편, $\omega_i(k)$ 를 이산 시간 k에서의 i번째 규칙의 입력별 적합도의 최대값이라 할때, i번째 규칙 R_i의 활화 정도(firing strength)의 합, $\beta_i\lambda_i$를 다음과 같이 정의 한다.

$$\beta_i = \sum_{k=0}^{K} \eta_{i(k)} \tag{5}$$

$$where\ \eta_{i(k)} = \begin{cases} \omega_{i(k)} & \text{if } \omega_{i(k)} \geq \rho \\ 0 & otherwise \end{cases} \tag{6}$$

$$\lambda_i = \sum_{k=0}^{K} \nu_{i(k)} \tag{7}$$

$$where\ \nu_{i(k)} = \begin{cases} 1 & \text{if } \omega_{i(k)} \geq \rho \\ 0 & otherwise \end{cases} \tag{8}$$

즉, β_i는 K 시간 까지의 제어동작에서, 입력 값들이 규칙 R_i을 얼마나 크게 활화 시켰는가(ρ 이상으로 활화한 것 중에서) 하는 지수이며 λ_i는 단순히 몇 번 활화되었는가를 나타내는 지수이다. 여기서 ρ는 [0,1] 인 수이다.

한편, 두개의 입력 변수의 퍼지 분할의 갯수가 각각 n 이면 가능한 규칙의 수는 $r=(n+1)^2$이다. r개의 규칙을 갖는 규칙 테이블에서 다음과 같은 지수를 정의 한다.

Z_i는 규칙 테이블에서 1 열에 속한 규칙에 해당되는 λ_i들의 합으로 정의된다. 같은 방법으로 Z_r는 규칙 테이블의 $n+1$ 열의 λ_i들의 합으로 정의된다. 역시 같은 방법으로 Z_u는 1 행의 규칙, Z_d는 $n+1$ 행의 λ_i의 합으로 정의 된다.

또한, w 와 s 가 $e, \dot{e}$에 대해 주어졌을 때, R_z는 $e=\dot{e}=0$ 일때 $\omega_z=1$로 활화된 z-번째 규칙으로 정의 한다. 이때

$$\frac{s}{w}n \in \{-n, -n+1, \cdots, 0, \cdots, n-1, n\} \tag{9}$$

이 되어야 하며 동시에, R_z가 유일함을 s와 w의 정의와 입력 퍼지 분할의 모양으로 부터 쉽게 보일 수 있다. 식 (9)는 $\omega_z=1$로 활화된 해당 입력변수의 퍼지 레이블을 의미한다. 만일 R_z가 존재하고, 그 규칙이 규칙 테이블의 i_z-번째 행, j_z-번째 열의 규칙이라 하면 이때 i_z와 j_z는 식 (9)로 부터 다음과 같이 계산된다.

$$i_z = \frac{n}{2} + \frac{s_{e'}}{w_{e'}} n+1 \quad (10)$$

$$= n(\frac{1}{2} + \frac{s_{e'}}{w_{e'}})+1 \quad (11)$$

$$j_z = \frac{n}{2} - \frac{s_e}{w_e} n+1 \quad (12)$$

$$= n(\frac{1}{2} - \frac{s_e}{w_e})+1 \quad (13)$$

만일, 조건식 (4)가 만족되면, j_z는 i_z와 같은 값이다.

Ⅲ. 새로운 자기동조 알고리즘

제안된 알고리즘은 탐색 알고리즘으로, 가능한 $w_e, s_e, w_{e'}, s_{e'}$의 조합중 성능지수를 최대로 하는 쌍을 찾는 과정이다.

우선 새로운 파라메타의 조합을 찾는 방법을 아래에 설명한다. 이때 파라메타를 찾는 과정에서 MacVicar-Whelan 규칙 행렬의 조건, 식 (4)가 한 번 만족되면 계속해서 만족되도록 설계되었다.

w와 s를 어떤 입력변수의 유효한 전체집합의 폭과 옵셋이라 하고 초기에 $s_e = s_{e'} = 0$라 가정 할때 갱신된 (updated) 폭과 옵셋 w'와 s'는 각 방법에 따라 다음과 같이 주어진다.

S1: 입력 전체집합의 상, 하한을 $s=0$ 조건하에서 줄인다.

$$w' = \frac{n-2}{n} w \quad (14)$$

$$s' = s \quad (15)$$

E1: 입력 전체집합의 상, 하한을 $s=0$ 조건하에서 늘인다.

$$w' = \frac{n}{n-2} \quad (16)$$

$$s' = s \quad (17)$$

S2: S1을 e 와 $\dot{e}$에 같이 적용한다.

E2: E1을 e 와 $\dot{e}$에 같이 적용한다.

S3: w를 줄이면서 하한(a)을 w/n 만큼 늘인다.

$$w' = \frac{w-2s-\frac{2w}{n}}{1-2\frac{w}{s}} \quad (18)$$

$$s' = \frac{s}{w} w' \quad (19)$$

E3: w를 늘이면서 하한(a)을 w/n 만큼 줄인다.

$$w' = \frac{w-2s}{1-2\frac{s}{w}-\frac{2}{n}} \quad (20)$$

$$s' = \frac{s}{w} w' \quad (21)$$

S4: w를 줄이면서 상한(b)을 w'/n 만큼 줄인다.

$$w' = \frac{w+2s-\frac{2w}{n}}{1+2\frac{s}{w}} \quad (22)$$

$$s' = \frac{s}{w} w' \quad (23)$$

E4: w를 늘이면서 상한(b)을 w'/n 만큼 늘인다.

$$w' = \frac{w+2s}{1+2\frac{s}{w}-2n} \quad (24)$$

$$s' = \frac{s}{w} w' \quad (25)$$

S5: j_z 와 i_z 각각을 1 감소시킨다. 이때 w는 줄이며 선행 조건은 $i_z \geq 3$ 이다.

$$s' = \frac{(w+2s)(\frac{s}{w}+\frac{1}{n})}{1+2(\frac{s}{w}+\frac{1}{n})} \quad (26)$$

$$w' = w + 2(s-s') \quad (27)$$

E5: j_z 와 i_z 각각을 1 증가시킨다. 이때 w는 늘이며 선행 조건은 $i_z \leq n$ 이다.

$$s' = \frac{(w+2s)(\frac{s}{w}-\frac{1}{n})}{1+2(\frac{s}{w}-\frac{1}{n})} \quad (28)$$

$$w' = w + 2(s-s') \quad (29)$$

S6: j_z 와 i_z 각각을 1 증가시킨다. 이때 w는 줄이며 선행 조건은 $i_z \leq n$ 이다.

$$s' = \frac{(w-2s)(\frac{s}{w}-\frac{1}{n})}{1-2(\frac{s}{w}-\frac{1}{n})} \quad (30)$$

$$w' = w - 2(s-s') \quad (31)$$

E6: j_z 와 i_z 각각을 1 감소시킨다. 이때 w는 늘이며 선행 조건은 $i_z \geq 3$ 이다.

$$s' = \frac{(w-2s)(\frac{s}{w}+\frac{1}{n})}{1-2(\frac{s}{w}+\frac{1}{n})} \quad (32)$$

$$w' = w - 2(s-s') \quad (33)$$

위 조정 법에서 E는 확장을 S는 축소를 의미한다. 따라서 예를 들어 E1을 적용하고 S1을 적용하면 원래와 같은 결과가 된다. S1, E1, S3, E3, S4, E4를 적용했을때의 입력 전체집합의 축소, 확장을 그림으로 나

타내면 그림 3과 같다.

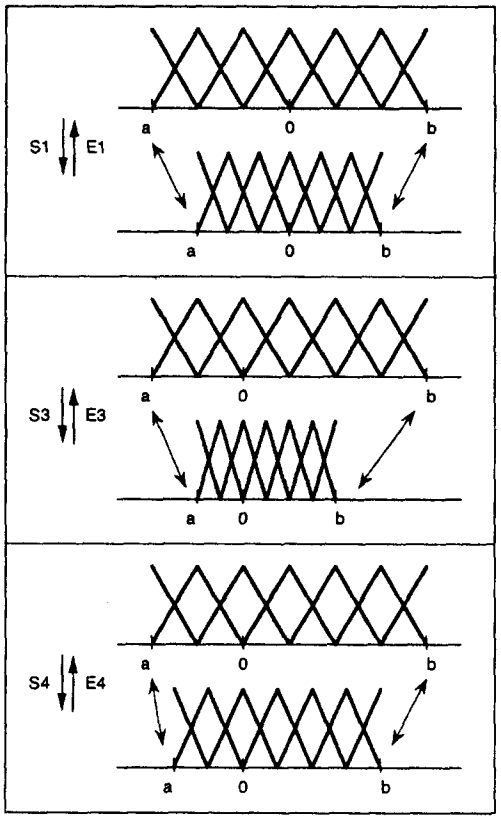


그림 3. 조정법의 도식적 표시(S1/E1, S3/E3, S4/E4)

Fig. 3. Graphical explanation of the adjustment methods(S1/E1, S3/E3, S4/E4)

앞에서와 같은 조정 방법을 이용하는 전체 동조 알고리즘을 설명하면 다음과 같다.

Step 1: 최초의 w 와 s 값으로 시스템을 실행한다. 여기서

$$I_1 = \beta_z \tag{34}$$

$$I_2 = \lambda_z \tag{35}$$

라 한다. 이때 β_z 와 λ_z는 R_z에 대한 β 와 λ 값이 된다.

Step 2: 적용 가능한 조정 방법을 찾는다. 이때 표 2에서 보인 대로 해당 조건에 맞는 가장 높은 우선순위에 해당되는 것을 선택한다. 만일 선택할 수 있는 방법이 없을 때 알고리즘은 정지한다. 그렇지 않으면 다음 단계로 넘어간다.

표 2. 각 갱신과정에 있어서의 우선순위와 선행조건

Table 2. Priorities and conditions for each update.

Input Variable	Method	Priority	Condition
e	S1	1	$s_e = 0, Z_l = Z_r = 0$
	S2	3	$s_e = S\dot{e} = 0, Z_l = Z_r = Z_u = Z_d = 0$
	S3	5	$s_e \neq 0, Z_r = 0$
	S4	7	$s_e \neq 0, Z_l = 0$
	S5	9	$j_z \leq 3$
	S6	11	$j_z \geq n$
$\dot{e}$	S1	2	$s_{\dot{e}} = 0, Z_u = Z_d = 0$
	S2	4	$s_e = S\dot{e} = 0, Z_l = Z_r = Z_u = Z_d = 0$
	S3	6	$s_{\dot{e}} \neq 0, Z_d = 0$
	S4	8	$s_{\dot{e}} \neq 0, Z_u = 0$
	S5	10	$j_z \leq 3$
	S6	12	$j_z \geq n$

Step 3: 제어 시스템을 2단계에서 선택한 방법을 적용하여 수행한다. 그리고 다음과 같이 표시되는 성능지수를 구한다.

$$I_1 = \beta_z \tag{36}$$

$$I_2 = \lambda_z \tag{37}$$

만약 $I_1 \geq I_1$ 혹은 $I_2 \geq I_2$이면, 해당 조정법을 확정하고 w와 s 값을 갱신한다.

또한, I_1는 I_1로, I_2는 I_2로 갱신하고 2 단계로 간다.

이러한 과정의 제안된 알고리즘을 흐름도로 표시하면 그림 4와 같다.

Ⅳ. 모의실험 및 고찰

제안된 자기 동조 알고리즘은 잘 알려진 도립진자 시스템에 적용하여 그 성능을 검증하였다.

시스템의 수식은 [22]에 설명된 바와 같다. 수레의 무게는 10kg으로, 막대의 무게는 1 Kg, 초기 막대의 각은 5°, 제어 시스템의 샘플링 시간은 10msec로 하였다. 알고리즘에서는 $\rho = 0.5$로 선택하였다. 또한 초기에 $w_e = 20, w_{\dot{e}} = 150, w_u = 140, s_e = s_{\dot{e}} = s_u = 0$ 로 설정되었다고 가정하였다. 이는 error의 범위는 [-10,10] 도, error의 변화율은 [-75,75] 도/sec로 한 것에 해당된다.

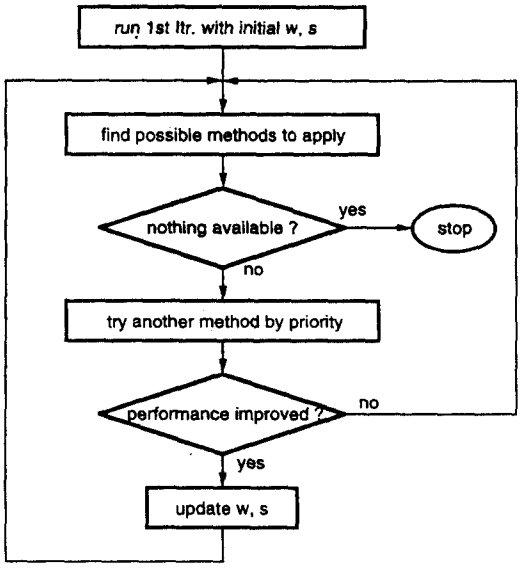


그림 4. 제안된 알고리즘의 흐름도
Fig. 4. Flowchart of the proposed algorithm.

표 3. 첫번째 실행에서의 성능지수 I_1 (I_2)
Table 3. Performance indices I_1 (I_2) at the first iteration step.

$\dot{e}$ \ e	L1	L2	L3	L4	L5	L6	L7
L7	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)
L6	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)
L5	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)
L4	0.00 (0)	0.00 (0)	0.00 (0)	212.10 (241)	37.25 (57)	2.01 (4)	0.00 (0)
L3	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)
L2	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)
L1	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)

모의실험을 통해 최초의 성능 및 최종 성능을 표시하는 성능지수, β_i, $i=1,49$가 표 3과 4에 각각 나타나 있다. 최초에 $j_z=i_z=4$ 이므로, 최초 수행시에 성능지수는 $I_1=212.10$ 이고 $I_2=241$ 였다. 그러나 제안된 알고리즘을 통해 자기 동조를 거치고 난 후에는 $I_1=239.59$ 와 $I_2=246$로 성능이 향상되었으며 실험결과는 그림 5에 나타난 바와 같다. 여기서 보면 제안된 동조 알고리즘이 실행을 반복함에 따라 보다 나은 성능을 보이는 방향으로 제어기 성능을 개선시키며 특히 상승시간(rise-time)을 줄이는 데 효과가 있음을 알 수 있다.

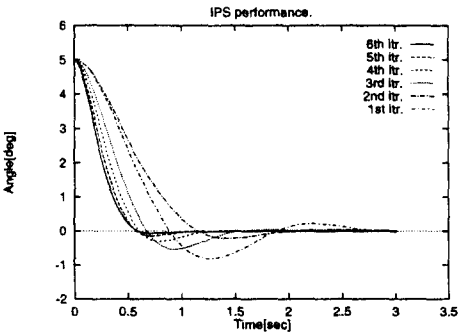


그림 5. 제안된 알고리즘을 적용한 시스템의 성능 향상
Fig. 5. Performance of the proposed algorithm.

이 실험에서의 동조과정은 표 5에 요약되어 있다. 실험과정에서 탐색되어진 조정 방법은 S1($\dot{e}$), S2, S2, S5, S5순으로 채택되고 7번째 실행에서 더 이상의 성능 개선이 없어 알고리즘이 정지하였다. 표에서 NA(Not Available)는 해당 방법에 대한 선행조건이 만족되지 않아 쓰일 수 없었음을 표시한다.

표 4. 최종 실행에서의 성능지수 I_1 (I_2)
Table 4. Performance indices I_1 (I_2) at the final iteration step.

$\dot{e}$ \ e	L1	L2	L3	L4	L5	L6	L7
L7	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)
L6	0.00 (0)	238.59 (246)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	1.54 (2)
L5	0.00 (0)	6.83 (11)	2.95 (5)	0.00 (0)	0.00 (0)	0.00 (0)	1.63 (2)
L4	0.00 (0)	0.00 (0)	4.83 (7)	1.77 (3)	0.00 (0)	0.00 (0)	2.95 (4)
L3	0.00 (0)	0.00 (0)	0.00 (0)	3.08 (5)	5.03 (7)	4.08 (6)	2.66 (4)
L2	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)
L1	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)	0.00 (0)

표 5. 제안된 알고리즘을 통한 성능지수의 변화 및 조정법
Table 5. Update procedure of performance indices.

Itr	S1(e)	S1($\dot{e}$)	S2	S3(e)	S3($\dot{e}$)	S4(e)	S4($\dot{e}$)	S5	S6
1	212.10 (241)								
2	162.28 (199)	216.05 (233)							
3	185.63 (210)	194.32 (219)	219.49 (238)						
4	196.08 (211)	219.42 (233)	227.93 (240)						
5	NA	216.48 (229)	NA	NA	NA	NA	NA	235.72 (244)	
6	NA	NA	NA	218.92 (228)	223.70 (234)	NA NA	176.13 (200)	238.59 (246)	
7	NA	NA	NA	NA*	226.10 (236)	NA NA	NA*	NA	NA

한편, 제안된 자기동조법에서는 전체 시스템의 성능을 두가지로 평가하고 있다. 즉, 하나는 규칙 R_z의 활화 정도의 합인데 이는 규칙의 전건부가 ``If e is Zero and $\dot{e}$ is Zero then …''인 규칙과 유사한 입력이 얼마나 많이 들어왔는가를 나타내는 지수이다. 이 값이 크면 정해진 기간동안 보다 많은 데이타에서 오차가 0으로 레귤레이션 된 것을 의미한다. 다른 한가지는 입력의 전체집합 영역이 제대로 선택되었는가 하는 것을 조사하는 것이다. 후자는 제안된 알고리즘에서 Z_1, Z_2, Z_m, Z_n 값을 조사하는 것에 해당되는 것이다.

이들 값 중 하나가 0이면 해당 영역의 경계를 줄여 전체 입력 범위를 실제의 범위에 맞출 수 있는 것이다. 이와 같이 입력의 유효 범위를 정하는 일은 퍼지 제어기에서 매우 중요한 성능의 차이를 가져오게 된다. 왜냐하면 정한 범위 밖의 입력은 경계값과 같은 입력 값으로 간주되어 경계 밖에서의 제어기 출력은 모두 같은 값이 되기 때문이다.

실제로, error 입력의 범위(w_e)를 줄이는 것은 제어기 이득을 늘이는 것과 유사하다. 또 error의 변화율 입력의 범위($w_{\dot{e}}$)를 줄이는 것은 반대로 제어기 이득을 줄이는 것과 유사하다. 따라서 제어기의 성능은 이러한 퍼지 입력의 전체집합 범위를 선택함에 따라 좌우된다. 이와 같은 특성변화의 한 예가 그림 6에 잘 나타나 있다.

그림 6에서 보면 입력변수 e에 S1 조정방법을 적용하면 $e>0$인 입력 영역에 대해서는 이득이 커지고 $e<0$인 입력에 대해서는 이득이 작아지는 경향을 보인다. 이는 기존의 스케일링 인자 조정의 방법에서 얻어지는 전형적인 제어기 특성 변화이다.

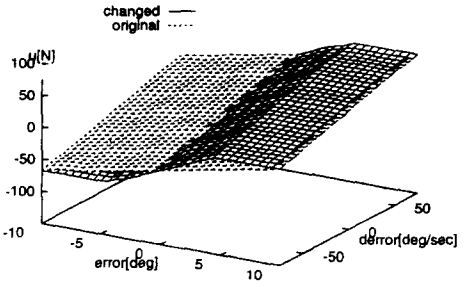


그림 6. S1을 e에 적용하였을 때의 제어기의 입, 출력 관계 변화
Fig. 6. Change of the I/O relation. (S1 is applied to e)

그러나 제안된 알고리즘에서의 입, 출력 관계의 변화는 보통의 스케일링 인자 조정의 경우와 다르다. 예를 들어, S5를 적용했을 때, 입, 출력 관계의 변화는 그림 7과 같다. 여기서 보면 단순한 스케일링 인자 조정의 경우와 달리 입력 영역별로 제어기 특성의 변화가 다르게 나타난다. 즉, S5 적용후의 특성은 초기 오차에 대해서는 이득이 커지고 정상 상태에 가까우면 이득이 적어져 상승시간과 오버슈트(overshoot)를 줄일 수 있다.

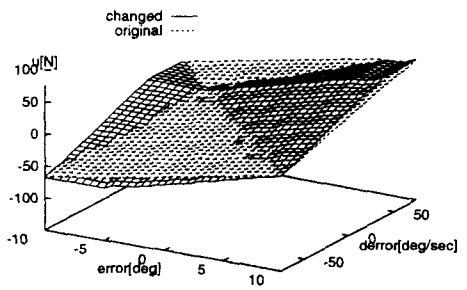


그림 7. S5를 e에 적용하였을 때의 제어기의 입, 출력 관계 변화
Fig. 7. Change of the I/O relation. (S5 is applied to e)

이와 같은 특성은 S5 방법이 옵셋, s를 변화시키므로 마치 규칙을 바꾸는 것과 같은 효과를 내기 때문이다. 이와 같이 기존의 스케일링 인자 조정 방법에 옵셋 조정을 추가한 제안된 방법은 규칙을 바꾸는 것과 같

은 효과를 보이므로 보다 더 나은 성능을 예상할 수 있다. 이를 확인하기 위해 같은 규칙과 입력 전체집합을 갖는 퍼지제어기에 S1 방법을 적용한 경우와 S5 방법을 적용한 경우, 두 가지에 대해 도립진자를 대상으로 적용한 모의실험 결과가 그림 8에 보인 바와 같다. 이와 같이, 옵셋이 없는 기존의 스케일링 인자만(SF only)의 방법과 비교할 때, 제안된 방법이 제어기의 비선형성을 더하여 보다 성능이 우수함을 알 수 있다.

한편, 앞서 설명한 제안된 알고리즘은 초기조건이 양인 경우만을 고려하였으나 같은 방법으로 음의 경우에도 적용할 수 있다. 즉, 시스템이 대칭이라는 조건으로부터 신호 조절 블록의 부호만 바꾸면 초기조건이 음의 부호를 갖는 경우에도 꼭같이 적용할 수 있다.

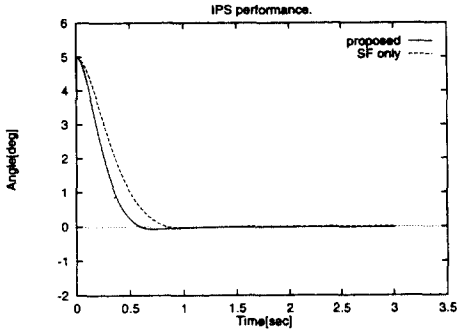


그림 8. 제안된 알고리즘과 기존의 스케일링 인자 조정법의 비교

Fig. 8. Comparison between the proposed algorithm with the conventional one.

V. 결 론

본 논문에서는 향후, 전용 하드웨어로 구성하기 용이한 퍼지 자기동조 알고리즘을 제안하였다.

제안된 알고리즘의 제어기 조정부분은 일반적인 퍼지 추론 하드웨어의 구성요소에서 신호 조절블록을 통해 실제로 구현될 수 있는 구조이다. 이때, 잘 알려진 기존의 스케일링 인자 조정법에 옵셋까지 추가로 조정함으로써 기존의 스케일링 인자 조정법을 포함하는 보다 일반화 된 스케일링 인자조정법으로 확장 하였으며 앞서 살펴본 바와 같이 비선형성을 높여 간단하면서도 보다 더 나은 성능을 기대 할 수 있다.

또한 하드웨어 내부의 소속함수 및 규칙 메모리의 데이타 내용을 수정함에 따른 시간적인 오버헤드가 없고 외부의 인자를 조정하여 소속함수의 전체집합 뿐만 아니라 특히, 내부의 규칙을 바꾸는 것과 같은 효과를 얻고 있다.

그리고 성능을 평가하는 방법도 하드웨어에서 간단히 구현이 될 수 있는 알고리즘이다. 즉, 각 규칙의 활화 정도를 합산하여 이를 토대로 제어 시스템의 상태 괘적의 양상을 파악하는 방법으로 성능을 평가하고 있다. 이렇게 하면, 퍼지 추론 하드웨어의 내부에는 대부분, 각 입력에 대한 적합도를 계산하게 되어 있으므로 이 적합도 값을 계수기(counter)나 덧셈기(adder)에 연결시켜 본 알고리즘에서 성능평가 지수로 이용하고 있는 I_1 혹은 I_2를 구하는 회로를 구현하는 것이 매우 용이하다. 따라서, 기존의 하드웨어 퍼지 프로세서에 자기동조 기능을 추가하기 위한 하나의 방법으로, 제안된 자기동조 알고리즘을 쓸 수 있다.

참 고 문 헌

[1] L.-X. Wang and J. Mendel, "Generating fuzzy rules by learning from examples," *IEEE Trans. on Syst., Man, Cybern.*, vol. 22, pp. 1414-1427, Nov/Dec 1992.

[2] T.Sudkamp and R. Hammell II, "Interpolation, completion and learning fuzzy rules," *IEEE Trans. on Syst., Man, Cybern.*, vol. 24, pp. 332-342, Feb. 1994.

[3] L.-X. Wang and J. Mendel, "Fuzzy basis fuctions, universal approximation, and orthogonal least-square learning," *IEEE Trans. on Neural Networks*, vol. 3, pp. 807-814, Sept. 1992.

[4] T. Sudkamp, "Similarity, interpolation, and fuzzy rule construction," *Fuzzy Sets and Systems*, vol. 58, pp. 73-86, 1993.

[5] B. Kosko, *Neural Networks and Fuzzy Systems: A dynamical systems approach to machine intelligence*. Englewood Cliffs, NJ: Prentice Hall, 1992.

[6] B. Graham and R. Newell, *"Fuzzy adaptive control of a first-order process." Fuzzy Sets and Systems*, vol. 31,

pp. 47-65, 1989.

[7] T. Procky and E. Mamdani, "A linguistic self-organizing process controller," *Automatica*, vol. 15, pp. 15-30, 1979.

[8] S. Shao, "Fuzzy self-organizing contorller and its application for dynamic processes," *Fuzzy Sets and Systems*, vol. 26, pp. 151-164, 1988.

[9] W. Bare, R. Mulholland, and S. Sofer, "Design of a self-tuning rule based controller for a gasoline refineary catalytic reformer," *IEEE Trans. on Automatic Control*, vol. 35, pp. 156-164, Feb. 1990.

[10] M. Braae and D.Rutherford, "Selection of parameters for a fuzzy logic controller," *Fuzzy Sets and Systems*, vol. 2, pp. 185-199, 1979.

[11] J. Buckley and H. Ying, "Expert fuzzy controller," *Fuzzy Sets and Systems*, vol. 44, pp. 373-390, 1991.

[12] D.Burkhardt and P.Bonissone, "Automated fuzzy knowledge base generation and tuning," in Proc. *IEEE Int. Conf. on Fuzzy Systems 1992*, pp. 179-187, 1992.

[13] M.Maeda and S.Murakami, "A self-tuning fuzzy controller," *Fuzzy Sets and Systems*, vol. 51, pp. 29-40, 1992.

[14] C. Chou and H. Lu, "A heuristic self-tuning fuzzy controller," *Fuzzy Sets and Systems*, vol. 61, pp. 249-264, 1994.

[15] W.Daugherity, B.Rathakrishnan, and J. Yen, "Performance evaluation of a self-tuning fuzzy controller," in Proc. *IEEE Int. Conf. on Fuzzy Systems 1992*, pp. 389-397, 1992.

[16] Y. Saito et al., "A high speed software fuzzy inference controller," in *Proc. 3rd IFSA Congress*, pp. 12-15, 1989.

[17] H. Arikawa et al., "Virtual paging fuzzy chip & fuzzy workstation as its design environment," in *Proc. 3rd IFSA Congress*, pp. 647-650, 1989.

[18] T. Yamakawa, "A fuzzy inference engine in nonlinear analog mode and its application to a fuzzy logic control," *IEEE Trans. Neural Networks*, vol. 4, pp. 496-522, May 1993.

[19] H. Watanabe, W.D.Dettloff, and K. E. Yount, "A VLSI fuzzy logic controller with reconfigurable, cascadable architecture," *IEEE J. of Solid-State Circuits*, vol. 25, pp. 376-382, Apr. 1990.

[20] R. Johnson, "Recent U.S. trends in fuzzy inference chips," *J. of Japan Society for Fuzzy Theory ans Systems*, vol. 6, pp. 446-450, June 1994.

[21] P. Macvicar-Whelan, "Fuzzy sets for man-machine interaction," *Int. J. Man-Machine Studies*, vol. 8, pp. 687-697, 1976.

[22] A. Barto, R. Sutton, and C. Anderson, "Neuronlike adaptive elements that can solve difficult learning control problems," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-13, pp. 834-846, 1983.

저 자 소 개

李 承 夏(正會員)

1965년 8월 17일생. 1988년 2월 경북대학교 공과대학 전자공학과 졸업(공학사). 1990년 2월 한국과학기술원 전기 및 전자공학과 졸업(공학석사). 1995년 2월 한국과학기술원 전기 및 전자공학과 졸업(공학박사). 1995년 3월 ~ 1996년 2월 한국과학기술원 전기 및 전자공학과 위촉연구원.(Post-Doc.) 1996년 3월 ~ 현재 경북대학교 전자전기공학부 국책계약교수. 주관심 분야는 퍼지 시스템, 지능제어, 로보틱스 등임

卞 增 男(正會員) 第 30卷 B編 第 10號 參照
한국과학기술원 전기 및 전자공학과 교수