경사법을 이용한 퍼지논리 제어기 설계
Design of Fuzzy Logic Controller Using a Descent Method

전 윤 식(기계설계과)
Yoon-Sik Jun (Dept. of Mechanical Design)

Key words: SOC(self-organizing controller, 자기구성제어기), FLC(fuzzy logic controller, 퍼지제어기), Performance table(성능평가표), Descent method(경사법)

ABSTRACT: Control rules that are the most important factor in fuzzy logic controller(FLC) are generally obtained from intuition and experience of the experts, and such rules represented by linguistic rule sets or fuzzy relations. But there is always something difficult to obtain from such control rules, and this makes the design of controller difficult. Therefore, a new self-organizing controller(SOC) algorithm, which has a good tracking performance by self-learning, is designed here. The proposed SOC algorithm uses simple descent method and has smaller calculation time than complex fuzzy relation, hence it is applicable in real time. The results of computer simulation for various processes show that there are a good control performances, though initial fuzzy rules are not given.

1. 서론

퍼지논리제어기(fuzzy logic controller, FLC)에서 가장 중요한 요소인 제어규칙은 일반적으로 제어대상 프로세서에 대한 전문가의 직관과 경험에 의해 습득되고 있으며, 언어적 규칙들의 집합 또는 퍼지관계로 표현된다. 그러나 전문가로부터의 제어규칙의 습득이 항상 가능한 것은 아니며, 제어규칙이 습득되었다 하더라도 완전한 제어규칙의 습득은 매우 어렵다. 이러한 제어규칙 습득의 어려움을 해결하기 위하여 Mamdani와 Procyk에 의하여 자기구성제어기(self-organizing controller, SOC)가 처음으로 제안되었다.[2,4]

SOC의 자기학습 알고리듬은 논리적인 언어적 변수에 근거하여 제안된 것으로서 안정도 해석에 대한 근거를 가지고 있지 않다. 따라서 Mamdani와 Procyk은 매우 다양한 계통에 대하여 SOC를 적용하여 안정성과 유용성을 보여주었다.

본 연구에서는 Mamdani와 Procyk에 의하여 제안된 성능평가 규칙표를 이용하고 Wang[3]이 연속치 계통의 단순 퍼지제어기에 적용한 Descent방법을 사용하여 프로세스의 가장 기본적인 입출력 특성에 대한 정보들만 주어지는 상황에서 실시간 연산이 가능하며 자기학습능력과 제어성능이 우수한 디지털 자기학습 퍼지제어기를 제안하였다.

2. 제어기 설계

2.1. 제어기 구조

그림 1 은 SOC의 구조를 나타내는 그림이며 상층레벨과 하층레벨을 갖는다. 하층레벨은 습득된 제어규칙들을 이용하여 프로세스를 제어하는 일반적인 PI구조의 FLC부분이며 상층레벨은 프로세스의 성능을 평가하여 그에 적절한 제어규칙을 습득하는 자기구성레벨이다.

자기 구성 레벨은 프로세스의 응답특성 e(k), ce(k)을 관찰하여 FLC의 제어성능이 향상되도록 제어규칙을 생성 또는 개정시키는 부분이며, 성능 평가(performance measure), 증분 모델(incremental model), 제어규칙 개정 알고리듬(rule modification algorithm)의 3단계로 구성되어 있다. 각 구성요소에 대하여 기술하면 다음과 같다.[4]

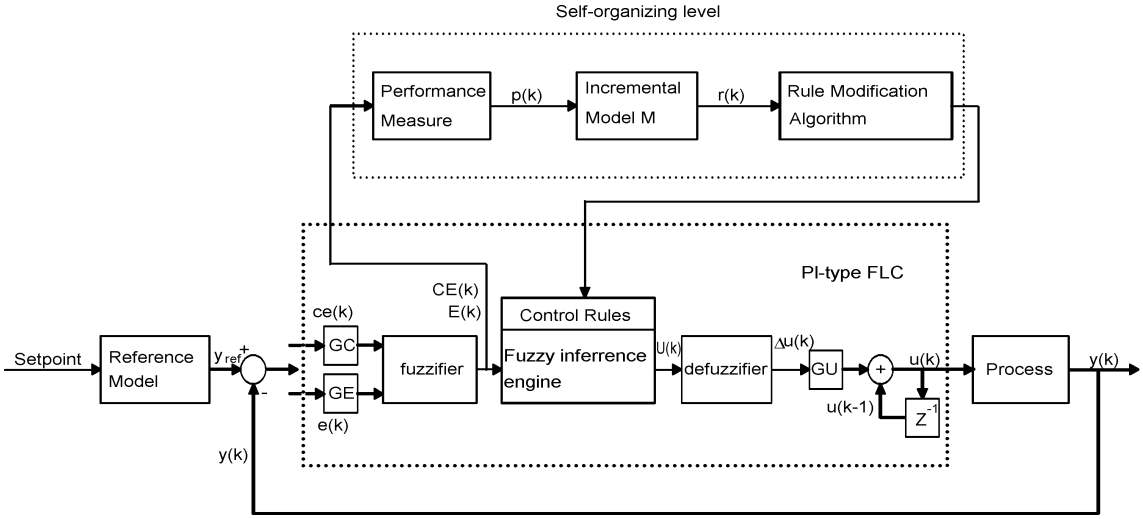


Fig. 1 Block diagram of the self-organizing controller

2.2. 성능 평가 (performance measure)

성능평가는 출력편차 E(k)와 출력편차 변화량 CE(k)를 관측하여 프로세스에 대한 제어기의 성능을 평가하여, 성능 개선을 위하여 요구되는 프로세스 출력 보정량 p(k)를 발생시키는 과정이다. 표 1.은 Mamdani와 Procyk에 의하여 고안된 성능평가표를 나타낸다.[2]

Table 1. Performance table of SOC

ERROR \ CHANGE IN ERROR	-6	-5	-4	-3	-2	-1	0	+1	+2	+3	+4	+5	+6
-6	-6	-6	-6	-6	-6	-6	-6	0	0	0	0	0	0
-5	-6	-6	-6	-6	-6	-6	-6	-3	-2	-2	0	0	0
-4	-6	-6	-6	-6	-6	-6	-4	-5	-4	-2	0	0	0
-3	-6	-5	-5	-4	-4	-4	-4	-3	-2	-2	0	0	0
-2	-6	-5	-4	-3	-2	-2	-2	0	0	0	0	0	0
-1	-5	-4	-3	-2	-1	-1	-1	0	0	0	0	0	0
-0	-4	-3	-2	-1	0	0	0	0	0	0	0	0	0
+0	0	0	0	0	0	0	0	0	0	1	2	3	4
+1	0	0	0	0	0	0	1	1	1	2	3	4	5
+2	0	0	0	0	0	0	2	2	2	3	4	5	6
+3	0	0	0	2	2	3	4	4	4	4	5	5	6
+4	0	0	0	2	4	5	4	6	6	6	6	6	6
+5	0	0	0	2	2	3	6	6	6	6	6	6	6
+6	0	0	0	0	0	0	6	6	6	6	6	6	6

2.3. 증분 모델(incremental model)

증분모델은 성능 평가표에 의하여 구하여진 출력 보정량 p(k)를 현재의 프로세스 출력성능을 개선하고자 이를 발생시켰던 입력에 대한 보정량 r(k)로 바꾸어 주는 부분이다. 이러한 증분모델은 프로세스가 비선형 다변수 계통인 경우 Jacobian 행렬로 표현되며 SISO(single input single output)경우 상수 값을 갖는다. 입력보정량은 증분 모델로부터 다음과 같이 구하여 진다.

$$r(k) = M^{-1} p(k) \tag{1}$$

여기서, M : 증분 모델

2.4 규칙개정 알고리듬(rule modification algorithm)

단순퍼지추론인 경우 퍼지추론식은 다음과 같이 단순화되어 표현될 수 있다.

$$\Delta u = \frac{\sum_{j=1}^{m}\sum_{i=1}^{n} \min(\mu_{Ei}(e), \mu_{CEi}(ce)) \cdot u_{ij}}{\sum_{j=1}^{m}\sum_{i=1}^{n} \min(\mu_{Ei}(e), \mu_{CEi}(ce))} \tag{2}$$

경사법 방법은 바람직한 출력과 추론한 출력을 최소화시키는 방향으로 어떤 값을 개정시키는 것이다. 따라서 주어진 입출력 데이터 (e(k), ce(k), $\Delta u^*(k)$), $e(k) \in \mathbb{E}$, $ce(k) \in \mathbb{CE}$, $\Delta u^* \in \mathbb{U}$에 대하여 퍼지논리제어기 개정 파라메터인 후건부 변수 u_l 은 다음과 같은 식을 최소화하도록 개정되어 진다.

$$e_u(k) = \frac{1}{2}(\Delta u^*(k) - \Delta u(k-d))^2 \tag{3}$$

여기서, k = 0, 1, 2, 3, ····, d : 프로세스 지연상수(delay)

따라서, 후건부 변수 u_l 은 다음과 같이 표현된다.

$$u_l(k+1) = u_l(k) - \alpha \frac{\partial e_u(k)}{\partial u_l} \tag{4}$$

l = 1, 2, ···· N, N = m×n, α : 학습율(learning rate)

체인 룰(chain rule)에 의하여

$$\begin{aligned} \frac{\partial e_u(k)}{\partial u_l} &= -(\Delta u^* - \Delta u(k-d)) \frac{\partial \Delta u(k-d)}{\partial a} \frac{\partial a}{\partial u_l} \\ &= -r(k) \frac{z_l}{b}\Big|_{k-d} \\ &= -r(k)\, \xi_l(k-d) \end{aligned} \tag{5}$$

$$a = \sum_{l=1}^{N} (u_l z_l(k-d)) \tag{6}$$

$$b = \sum_{l=1}^{N} z_l(k-d) \tag{7}$$

$$z_l(k-d) = \min(\mu_{El}(e(k-d)), \mu_{CEl}(ce(k-d))) \tag{8}$$

$$\xi_l(k-d) = \frac{z_i}{b}|_{k-d} \tag{9}$$

$\Delta u^* - \Delta u(k-d) = r(k)$: 증분모델로부터의 입력 보정량

그러므로 규칙개정 알고리듬은 다음과 같이 주어진다

$$u_l(k+1) = u_l(k) + \alpha r(k)\,\xi_l(k-d) \tag{10}$$

또한 $r(k) = M^{-1}p(k)$ 이므로

$$u_l(k+1) = u_l(k) + \alpha' p(k)\,\xi_l(k-d) \tag{11}$$

여기서 $\alpha' = \alpha M^{-1}$, $p(k)$: 성능 평가표의 출력값

3 시뮬레이션

시뮬레이션 연구를 통하여 본 논문에서 제안한 자기학습 알고리듬의 안정성과 환산계수 GE, GC, GU 및 학습율 α에 따른 제어성능을 분석하였다.

제어성능 분석을 위해 사용된 예제 프로세스는 다음과 같다.[1]

예제 1 : 2차 비 감쇠 선형 시스템

$$\frac{Y(s)}{U(s)} = \frac{100}{s^2+2s+100} \tag{12}$$

예제 2 : 2차 과 감쇠 선형 시스템

$$\frac{Y(s)}{U(s)} = \frac{1}{s^2+2s+1} \tag{13}$$

예제 3 : 1차 비선형 시스템

$$y(k+1) = \frac{y(k)}{1+y^2(k)} + u^3(k) \tag{14}$$

예제 4 : 2차 비선형 시스템

$$y(k+1) = \frac{y(k)y(k-1)[y(k)+2.5]}{1+y^2(k)+y^2(k-1)} + u(k) \tag{15}$$

본 연구에서 사용된 SOC의 입력변수 e(k), ce(k)에 대한 기준퍼지집합, 즉 언어적 라벨의 개수는 각각 NB(negative big), NS(negative small), ZO(zero), PS(positive small), PB(positive big)의 5개로 정의하였으며 입력퍼지변수 E, CE 및 후건부 변수 u_l에 대한 전체공간은 -6 ~ +6으로 하였다. 그림 2.는 이들 퍼지집합의 귀속함수 형태를 나타낸다.

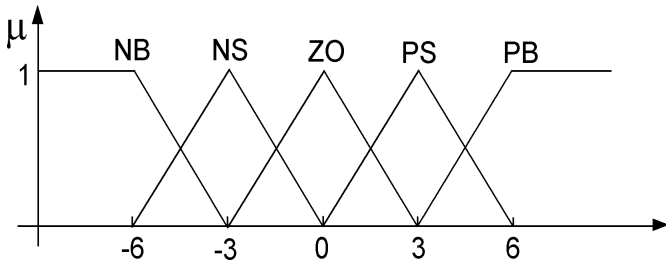


Fig. 2 The form of membership function
in input variables(E,CE)

제어성능 분석을 위한 기준 추종모델로 $y_m = \sin(2\pi t/25)$의 사인파를 추종하도록 제어하였으며, 샘플링 시간은 0.1초로 하였다. 또한 규칙 개정 초기의 후건부 변수 u_l 은 0 으로 하였다.

예제 1 시스템에 있어서 환산계수 조정에 대한 대략적인 식으로서 다음과 같이 제안하였으며 프로세스의 특성에 따라 약간씩 조정되어진다.

$$GE \cong 6/e_{max} \quad (16)$$

$$GC \cong GE \times t_r/h_c \quad (17)$$

$$GU \cong 3/(K \times GC) \quad (18)$$

여기서 e_{max} : $|y_m - y|_{max}$

t_r : 폐루프 상승시간

h_c: 샘플링 시간

K : 시스템 개루프 이득

따라서 예제 1 시스템에 대하여 h_c = 0.1초, K = 1이며, e_{max} = 0.3, t_r = 3초로 하였을 때 GE = 20, GC = 600, GU = 0.005를 얻을 수 있으며 환산계수 및 학습율을 표 2.와 같이 변화시켜 제어특성을 분석하였다.

Table 2. Variations of scaling factors and learning rate for example 1

	Scaling factor			α	M^{-1}
	GE	GC	GU		
Setting Value	20	600	0.005	0.5	0.3
Variation of GE	5	600	0.005		
	50	600	0.005		
Variation of GC	20	50	0.005		
	20	1000	0.005		
Variation of GU	20	600	0.003		
	20	600	0.008		
Variation of α	20	600	0.005	0.1	
				0.9	

4. 결과 및 고찰

4.1 환산계수(scale factor) 및 학습율(learning rate)에 따른 제어 성능 분석

그림 3.은 예제 1에 대한 사인파 추종 응답특성을 나타내고 있다. 그림에서 보는 바와 같이 초기 제어규칙에 대한 아무런 정보가 없었음에도 불구하고 빠른 시간에 제어규칙이 습득되어 주어진 사인파를 잘 추종하고 있음을 보여준다. 표 3.은 예제 1 시스템에 대하여 규칙개정 알고리듬에 의해 학습된 제어규칙을 나타내고 있다.

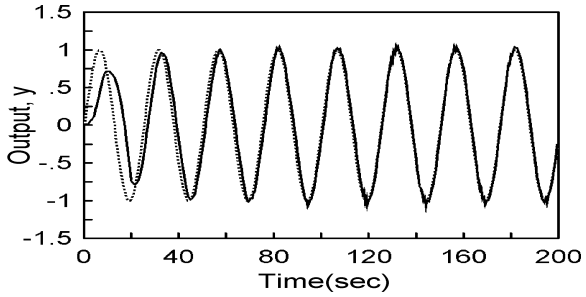


Fig. 3. Outputs of example 1 with setting value
(GE=20, GC=600, GU=0.005, $M^{-1}\times\alpha=0.3\times0.5$,)

Table 3. Control rules obtained by SOC for the example 1

E \ CE		Change in error(CE)				
		NB	NS	ZO	PS	PB
Error (E)	NB	-1.7259	-0.0921	-3.3254	-3.9946	-6.0000
	NS	-5.3699	-1.6449	-4.5492	-5.6972	-5.9216
	ZO	6.0000	0.5855	0.9663	0.5312	-0.7198
	PS	6.0000	4.9712	5.0584	3.4428	6.0000
	PB	6.0000	5.2505	3.2277	0.4817	1.4859

그림 4.(a)(b)에서 보는바와 같이 GE값이 각각 5, 50일 때의 응답특성을 나타내고 있으며, 그림에서 보는 바와 같이 GE값이 작을 때 자기 학습이 늦어지는 경향이 있으며 학습이 된 후에도 사인파와 약간의 오차가 나타남을 보여준다. 이는 오차에 대한 퍼지논리의 환산 값이 작아서 작은 오차에 대한 규칙개정을 하지 않기 때문이다. GE값이 클 경우 자기 학습은 빨라지나 학습이 된 후에 사인파의 극대점에서 출력이 진동하는 경향을 보인다. 이는 오차에 대한 환산 값이 커서 작은 오차에 대하여서도 과도한 규칙개정이 이루어지고 있음을 나타낸다.

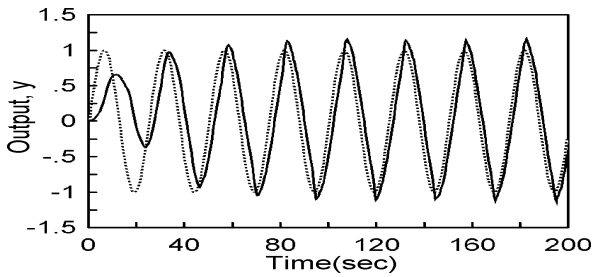
그림 5.(a)(b)는 GC값이 각각 50, 1000일 때의 응답특성을 나타내고 있으며, 그림에서 보는 바와 같이 GC값이 작을수록 학습이 빨라지는 경향이 있으나 사인파의 극대점에서는 오차가 나타난다. GC값이 클 경우 학습이 늦어지는 경향이 있으나 사인파의 극대점에서 오차는

나타나지 않는다. 그러나 전반적으로 그림 3.과 비교해볼 때 SOC는 GC값의 선정에 민감하지 않음을 보여준다.

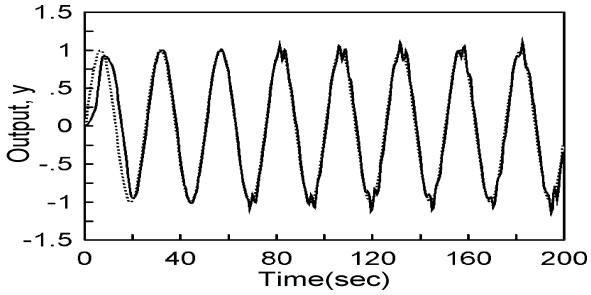
그림 6.(a)(b)는 GU값이 각각 0.003, 0.008일 때의 응답특성을 나타내고 있으며, 그림에서 보는 바와 같이 GU값이 작을 경우 제어입력에 대한 환산 값이 작아서 사인파를 추종하는 충분한 제어입력값 발생되지 않아 계속해서 오차가 발생되고 있음을 보여준다. 반대로 GU값이 클 경우 과도한 제어입력이 발생되어 프로세스 출력이 불안정해짐을 보여주고 있다.

그림 7.(a)(b)는 α값이 각각 0.1, 0.9일 때의 응답특성을 나타내고 있다. 그림에서 보는 바와 같이 α값이 작을 경우 제어규칙의 학습율이 낮아 추종이 늦어짐을 알 수 있으며, 반대로 α값이 클 경우 제어규칙의 학습율이 높아져서 빠른 추종성능을 보여주고 있으나 작은 오차에 대한 미세한 규칙개정이 이루어지지 않아 사인파의 극대점에서 약간의 오차가 나타남을 보여준다

.

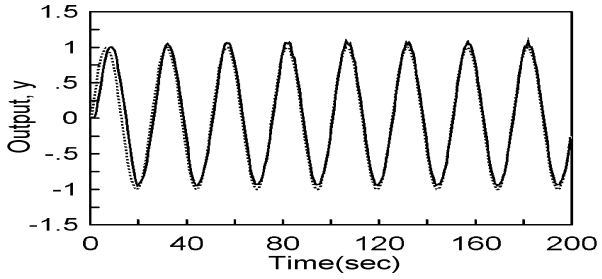


(a) GE=5

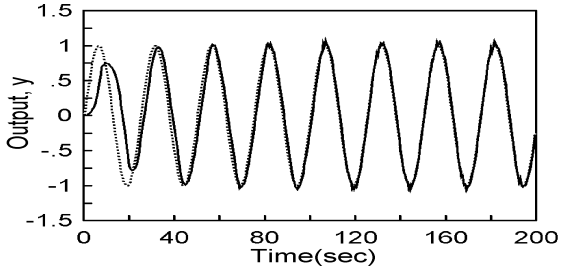


(b) GE=50

Fig. 4. Outputs of the example 1 for the change in GE

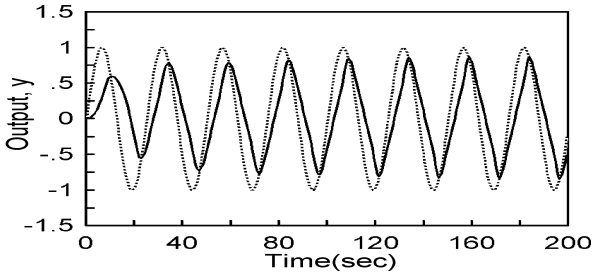


(a) GC=50

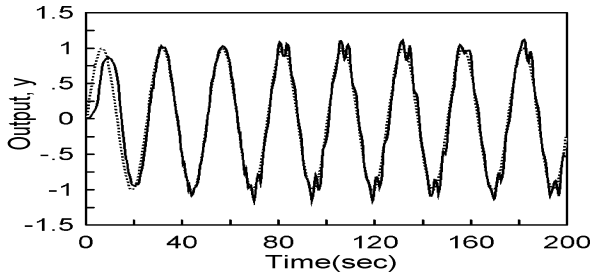


(a) GC=1000

Fig. 5. Outputs of the example 1 for the change in GC

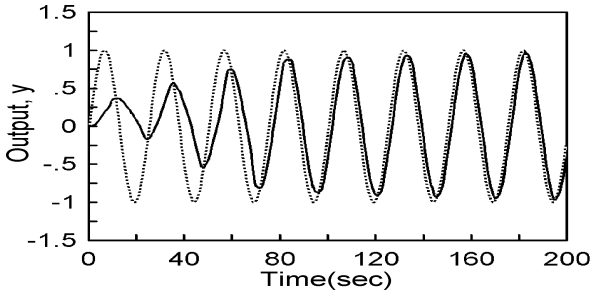


(a) GU=0.003

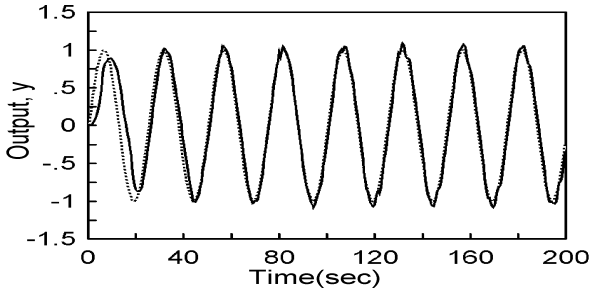


(a) GU=0.008

Fig. 6. Outputs of the example 1 for the change in GU



(a) $\alpha = 0.1$

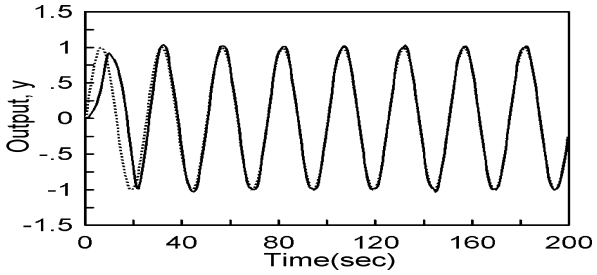


(a) $\alpha = 0.9$

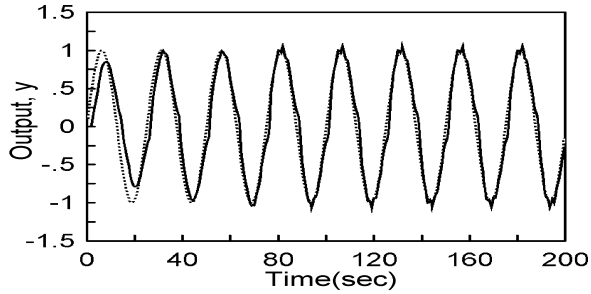
Fig. 7. Outputs of the example 1 for the change in α

4.2 여러 가지 프로세스에 대한 제어성능 분석

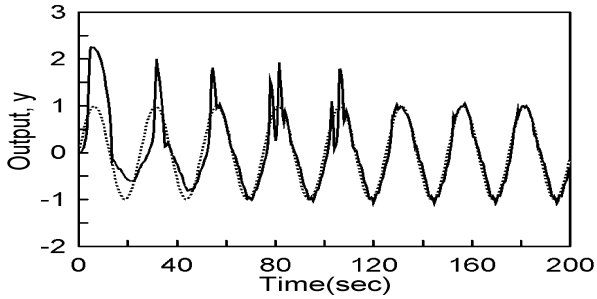
그림 8.(a)(b)(c)는 각각 예제 2, 3, 4 프로세스에 대한 자기학습 퍼지제어기 추종성능을 나타낸 그림이며, 우수한 추종성능을 나타내고 있다. 특히 예제 4의 비선형 프로세스인 경우 다른 프로세스에 비하여 학습이 느리며 초기에 출력이 매우 불안정함을 보여주고 있으나 어느 정도 학습이 진행된 후에는 안정된 추종성능을 나타낸다.



(a) example 2 (GE=50, GC=600, GU=0.005, $M^{-1} \times \alpha = 0.3 \times 0.5$)



(b) example 3 (GE=20, GC=600, GU=0.005, $M^{-1}\times\alpha=0.3\times0.5$)



(c) example 4 (GE=10, GC=600, GU=0.005, $M^{-1}\times\alpha=0.3\times0.5$)

Fig. 8. Outputs of the various processes with SOC

5. 결론

SOC구조를 갖는 PI형의 퍼지제어기에 있어서 기존의 퍼지관계에 의한 자기학습 알고리듬을 사용하는 대신에 Descent방법을 이용하여 학습능력이 우수하고 연산이 적어 실시간 제어가 가능한 새로운 자기학습 퍼지알고리듬을 제안하였다. 컴퓨터 시뮬레이션 결과, 환산계수에 변동에 따른 제어기의 안정성을 보여주었으며, 여러 가지 선형 및 비선형 시스템에 대하여 빠른 자기학습 능력과 우수한 추종성능을 보였다.

참고문헌

(1) Narendra, K. S. and Parthasarathy, K., 1990, *"Identification and Control of Dynamical Systems Using Neural Networks," IEEE Trans. on Neural Networks*, Vol. 1, No. 1, pp. 4~27.

(2) Procyk, T. J. and Mandani, E. H., 1979, *"A Linguistic Self-Organizing Process Controller," Automatica* Vol. 15, pp. 15~30.

(3) Wang, L. X., 1994, *Adaptive Fuzzy Systems and Control : Design and Stability Analysis*, Prentice-Hall Int. Inc.

(4) 배상욱, 1994, "자기학습 알고리듬에 근거한 퍼지모델 식별 및 퍼지 논리제어", 고려대학교 전기공학과 박사학위 논문.