클러스터링과 GA를 이용한 퍼지 제어기 설계 자동화

윤용석, 공성곤
숭실대학교 전기공학과

Automatic Design of Fuzzy Controller Using Clustering and Genetic Algorithm

Yong-Seock Yoon, Seong-Gon Kong

Abstract - 본 논문에서는 전문가의 지식이 없는 상황에서 자동적으로 최적의 퍼지 제어기를 설계하는 방법에 대해 연구한다. 먼저 퍼지 제어기의 규칙 설정을 위해 기존의 PID 제어기의 입출력 데이터를 클러스터링한다. 군집된 데이터들로부터 클러스터의 수를 파악하고 이를 바탕으로 퍼지 제어를 위한 규칙의 수를 결정한다. 둘째로 퍼지 제어기의 여러 파라미터들은 유전자 알고리즘을 적용하여 최적화한다. GA를 이용한 최적화 과정에서는 성능평가 기준으로 기준입력에 대한 시스템 응답간의 오차와 오버슈트의 크기를 사용하여 응답이 빠르고 안정적인 제어기를 설계하도록 진화방향을 설정한다. 이렇게 만들어진 퍼지 제어기의 성능을 기존의 PID 제어기와 비교·평가한다.

1. 서 론

시스템이 복잡하여 특성을 판별하기 어렵거나 기존의 방법들로는 제어가 힘든 시스템들에 대해서 인간의 논리와 유사하고 대략적인 개념을 사용하는 퍼지 로직 제어기(FLC)의 적용은 적합하고도 편리한 방법이라 할 수 있다. 일반적으로 퍼지 로직 컨트롤러를 설계하는데는 전문가의 지식, 경험, 직관에 의존하므로 시스템에 대한 충분한 사전지식이 필요하다. 그러나 이와 같은 배경지식이 부재한 상황에서는 제어기 설계가 어렵게 된다. 따라서 이러한 문제를 위한 여러 시도가 있어왔다[1]. 그리고 전문가 지식을 기반으로 퍼지시스템을 설계한다해도 최적의 퍼지 제어기의 설계를 위해서는 파라미터 튜닝의 과정이 필요하다. 따라서 전문가의 배경 지식이 부재한 상황과 각종 파라미터 튜닝을 위한 새로운 설계 방법이 요구된다.

클러스터링 알고리즘은 데이터들의 군집된 곳을 찾는 것으로 일종의 반복되는 패턴을 찾기 위한 것이라 할 수 있다. 제어 시스템의 입출력 상태에는 자주 나타나는 패턴이 있는데, 시스템의 여러 상태에 대한 데이터를 추출하고 이를 바탕으로 반복되는 패턴을 분류해 낼 수 있다. 클러스터링에 쓰이는 데이터는 기존의 제어기를 구성한 뒤 시뮬레이션을 통해 얻을 수 있다. 샘플링 시간마다 필요한 시스템의 상태를 저장하고, 이를 가지고 클러스터링한다. 그 결과로 퍼지 제어를 위한 규칙을 설정할 수 있다.

퍼지 제어기의 설계를 위해서는 클러스터링에 의해 구할 수 있는 규칙에 관한 파라미터 외에도 멤버쉽 함수의 폭과 같은 파라미터들의 설정이 필요하다. 일반적으로 Gradient Decent 방법이 많이 쓰이나 이는 초기 파라미터 설정에 따른 지역 최소점에 수렴할 우려가 있다. 본 논문에서는 그 탐색능력을 인정받고 최근에 널리 쓰이고 있는 GA를 이용한다[5][6]. GA에서 적합도 함수는 기준 입력에 대한 시스템 응답간의 오차와 오버슈트의 크기를 사용하며, GA의 염색체가 오차가 적고 오버슈트도 제한된 값보다 작아지는 방향으로 진화할 수 있도록 한다. 결과적으로 GA는 최적의 퍼지 제어기를 찾는 방향으로 진화한다. 마지막으로 GA의 결과를 기존의 PID 제어기와 비교하고 그 성능을 확인한다.

2. 본 론

2.1 문제 제기

제어기를 설계하는 방법이 2차 시스템에 대해 고찰되었다. 문제는 아래의 전달함수를 갖는 플랜트를 위한 제어기의 구조와 파라미터를 동시에 설계하는 것이다[2].

$$P(s)=\frac{K}{(1+\tau s)^2} \tag{1}$$

식(1)은 플랜트의 전달함수이다. 제어기의 조건은 4% 이내의 오버슈트, 2초보다 적은 정착시간을 갖도록 설계하는 것이다. 플랜트를 위한 기존의 제어기[2]는 입력을 전처리 하는 저역 통과 프리필터와 PID 제어기로 이루어져 있고, 문제에 대해 만족할만한 성능을 보여주고 있다.

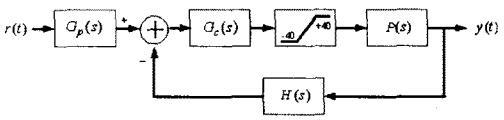


그림 1. PID 제어 시스템

그림 1은 기존의 제어기가 포함된 전체 시스템을 나타낸 것이다.

$$G_p(s)=\frac{42.67}{42.67+11.38s+s^2} \tag{2}$$

$$G_c(s)=12(11.38+\frac{42.67}{s}+s) \tag{3}$$

식 (2)는 그림 1의 $G_p(s)$를 나타내며, 식 (3)은 그림 1에서 $G_c(s)$를 나타낸다.

$$H(s)=\frac{1024.08}{s^3+26s^2274.12s+1024.08} \tag{4}$$

식 (4)는 내부이득 K=2, 시정수 τ=1일 때 제어기를 포함한 플랜트의 전체 전달 함수이다.

$$H[k]=\frac{0.0001599z^2+0.0005994z+0.0001404}{z^3-2.746z^2+2.518z-0.7711} \tag{5}$$

식 (5)는 식 (4)에 대해 샘플링 타임을 0.01초로 놓고, zero-order hold로 z-변환했을 때의 전체 시스템의 전달함수이다.

2.2 Subtractive 클러스터링

본 논문에서는 여러 클러스터링 방법 중 Subtractive

Clustering 방법을 사용한다[3]. 클러스터링을 위해 기존 시스템의 입출력 데이터를 사용한다[1]. 기존 제어 시스템의 입출력 데이터를 구하기 위해 먼저 기존의 시스템을 구성하고 시스템의 기준 입력을 1초간 스텝입력, 그 다음 1초간 0으로 하고, 위 시스템을 시뮬레이션 한다. 시뮬레이션 동안 각 샘플링 시점에서 기준신호와 실제 플랜트와의 오차, 오차의 변화량 그리고 제어기의 출력을 구하고 이 3차원의 데이터를 클러스터링한다. 클러스터링 과정은 다음과 같다. 첫 번째로 아래와 같은 식을 사용하여 각 데이터마다 포텐셜 함수 값을 구한다.

$$P_1(i) = \sum_{j=1}^{K} \exp(-\alpha \|X_i - X_j\|^2) \quad (\text{단, } i=1,2,\cdots,K) \qquad (6)$$

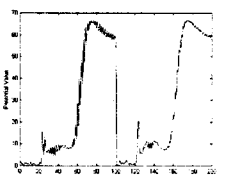
$P_1(i)$는 i번째 데이터의 포텐셜 값이며 α는 양의 상수, K는 전체 데이터 셋의 수이다. 두 번째로, 위에서 구한 포텐셜 값 중 가장 높은 값을 P_1^*로 표시하고 이때의 데이터를 첫 번째 클러스터 중심 X_1^*로 선정한다. 세 번째로, 첫 번째 클러스터 중심의 영향을 제거한다. 이것은 첫 번째 클러스터 중심 근처에는 많은 데이터가 존재하기 때문에 두 번째 클러스터 중심이 첫 번째 클러스터 중심의 근처에서 발생될 가능성이 높기 때문이다. 따라서 다음 식과 같이 첫 번째 클러스터 중심의 영향을 제거하여 새로운 포텐셜 값을 구한다.

$$P_2(i) = P_1(i) - P_1^* \exp(-\beta \|X_i - X_1^*\|^2) \quad (\text{단, } i=1,2,\cdots,K) \qquad (7)$$

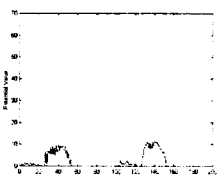
$P_2(i)$는 첫 번째 클러스터의 영향을 제거한 i번째 데이터의 포텐셜 값이고 β는 양의 상수이다. 네 번째 과정으로, 위에서 구한 $P_2(i)$중 가장 큰 값을 P_2^*라 표시하고 이 때의 데이터를 두 번째 클러스터 중심 X_2^*로 선정한다. k번째 클러스터 중심 X_k^*가 선정되었을 때, k번째 클러스터 중심의 영향을 제거한 포텐셜 값은 다음 식과 같고 이중 가장 큰 값을 $(k+1)$번째 클러스터 중심 값으로 선정한다.

$$P_{k+1}(i) = P_k(i) - P_k^* \exp(-\beta \|X_i - X_k^*\|^2) \quad (\text{단, } i=1,2,\cdots,K) \qquad (8)$$

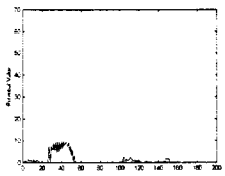
이 과정을 특정한 조건이 만족 될 때까지 계속한다. 본 논문에서는 $(k+1)$번째 함수의 최대 값이 첫 번째 클러스터 중심의 포텐셜 값의 5%보다 작아질 때까지로 정의하고 클러스터링에 직접적인 영향을 미치는 상수들인 α, β 는 각각 4, 1로 설정한다. 그 결과 3개의 클러스터 중심을 구하고, 각 단계에서 포텐셜 함수의 모양은 아래와 같다.



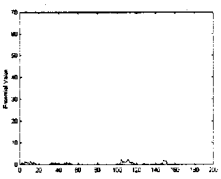
(a) $P_1(i)$



(b) $P_2(i)$



(c) $P_3(i)$



(d) $P_4(i)$

그림 2. 클러스터링 단계에 따른 포텐셜 함수

2.3 퍼지 제어기 설계

퍼지 제어기를 설계하고 시뮬레이션 하는데 있어서 제어기의 출력은 실제와 유사하게 [-40,40] [volt]의 범위로 제약을 둔다.

퍼지 제어기의 입력으로는 클러스터링 데이터에서와 마찬가지로 오차와 오차의 변화량을 사용한다. 그리고 클러스터링에 의한 퍼지제어기의 규칙은 다음과 같다.

Rule 1 :
IF $e(k)$ is u_{11} AND $\Delta e(k)$ is u_{12} THEN y is y_1^*
Rule 2 :
IF $e(k)$ is u_{21} AND $\Delta e(k)$ is u_{22} THEN y is y_2^*
Rule 3 :
IF $e(k)$ is u_{31} AND $\Delta e(k)$ is u_{32} THEN y is y_3^*

사용한 퍼지 시스템의 입력 멤버쉽 함수는 가우시안 함수로 다음 식으로 표현된다.

$$u_{i1}(e(k)) = \exp\left[-\frac{1}{2}\left(\frac{e(k)-u_{i1}^*}{\sigma_{i1}}\right)^2\right] \qquad (9)$$

$$u_{i2}(\Delta e(k)) = \exp\left[-\frac{1}{2}\left(\frac{\Delta e(k)-u_{i2}^*}{\sigma_{i2}}\right)^2\right] \qquad (10)$$

식 (9),(10)은 각각 오차와 오차의 변화량에 대한 멤버쉽 함수이고, $i=1,2,3$으로 규칙을 나타낸다. 그리고 u_{i1}^*, σ_{i1}은 가우시안 멤버쉽 함수 u_{i1}의 중심과 폭을 나타낸다.

$$\tau_i = u_{i1}(e(k)) \cdot u_{i2}(\Delta e(k)) \qquad (11)$$

식(11)은 본 논문에서 사용한 적합도(DOF)의 식으로, Singleton 방법을 적용하고, min 연산 대신 product 연산 방법을 사용한다. 그리고 MAMDANI 타입의 퍼지 로직과 Simplified Reasoning 방법에 의한 후건부 계산 방법을 사용한다[4]. 따라서 퍼지 시스템의 최종 출력 y는 다음과 같이 계산된다.

$$y = v_1 y_1^* + v_2 y_2^* + v_3 y_3^* \qquad (12)$$

식 (12)에서 y_i^*($i=1,2,3$)는 i번째 규칙의 후건부 값이고, $v_i = \tau_i / \sum_{j=1}^{2} \tau_j$으로 정규화된 적합도(normalized DOF)라 한다.

2.4 유전자 알고리즘의 적용

제안한 퍼지제어기에서 최적화 되어야할 파라미터들은 입력 멤버쉽 함수(gaussian)의 중심과 폭, 그리고 후건부 출력의 중심값이다. 규칙이 3개, 입력 변수가 2개, 각 입력에 대해 가우시안 함수의 구성을 위한 변수가 2개씩 있으므로 12개의 입력 멤버쉽 함수에 관계된 파라

미터와 후건부 중심값 3개를 최적화해야 하므로 총 15개의 파라미터를 튜닝 해야 한다.

설계에서의 편리성과 일반성을 위해 퍼지 제어기의 입·출력 구간을 모두 $[-1,1]$의 범위로 정규화하여 설계한다. 그러므로 제어기의 모든 입·출력들은 정규화 과정을 거치게 된다.

유전자의 각 염색체는 2진수로 코딩하는 방법을 선택한다. 최적화할 파라미터의 분해능을 0.01보다 작게 하기 위해($2/2^8 = 0.0078$) 각 파라미터마다 8비트를 할당하고, 염색체의 크기는 120비트가 된다. 그러므로 유전자 알고리즘이 찾는 최적의 염색체는 2^{120}개의 가능한 조합들 중의 하나가 되는 것이다. 제안된 염색체의 구조는 그림 3과 같다.

u_{11}^*	σ_{11}	···	u_{21}^*	σ_{21}	···	y_1^*	y_2^*	y_3^*

그림 3. 제안된 염색체의 구조

본 연구에서 사용한 유전자 알고리즘의 파라미터들은 표 1과 같다.

표 1. 유전자 알고리즘 적용시 파라미터

Population Size(N)	200
Crossover Probability	0.8
Mutation Probability	0.05
Number of Generations	1000

더 나은 파라미터들의 조합을 선택하기 위해, GA의 각 개체를 다음의 적합도 함수에 의해 평가한다.

$$J = \frac{1}{1+\sum_{k=1}^{K} |e(k)|} \qquad (13)$$

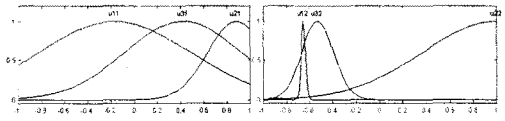
식 (12)는 본 논문에서 사용된 적합도 함수이다. 그리고 K는 시뮬레이션 동안의 샘플링 수이다. $Ref(k)$를 기준입력, $y(k)$를 플랜트의 실제 출력이라 할 때 오차 $e(k) = Ref(k) - y(k)$이다. 즉, 적합도는 각 개체의 유전자의 정보로 퍼지제어기를 구성하여 시뮬레이션을 했을 때의 오차의 총합의 역수이다.

오버슈트도 제어기의 성능평가 기준으로 사용하여 오버슈트가 기준입력보다 2% 이상 커지는 것을 제한하기 위해 오버슈트가 제한된 기준을 넘는 염색체는 적합도 값이 높더라도 다음 세대로의 진화를 제한한다. 따라서 GA는 위의 적합도 함수를 최대화하면서 오버슈트가 2% 이내가 되도록 하는 방향, 즉 응답도 빠르고 오버슈트도 작은 안정된 제어기를 찾는 방향으로 진화하게 된다.

2.5. 시뮬레이션

GA에 의해 찾은 최적 퍼지 제어기의 멤버쉽 함수와 출력의 파라미터들은 아래와 같다.

$u_{11}^*=-0.168627$, $\sigma_{11}=0.647059$, $u_{21}^*=0.882353$, $\sigma_{21}=0.247059$, $u_{31}^*=0.419608$, $\sigma_{31}=0.498039$, $u_{12}^*=-0.654902$, $\sigma_{12}=0.0196078$, $u_{22}^*=1$, $\sigma_{22}=0.623529$, $u_{32}^*=-0.529412$, $\sigma_{32}=0.137255$이다. 그리고 각 규칙에서 후건부의 중심은 $y_1^*=0.388235$, $y_2^*=0.929412$, $y_3^*=-0.984314$이다. 이때 멤버쉽 함수의 모양은 다음의 그림과 같다.



(a) $e(k)$ (b) $\Delta e(k)$

그림 4. GA로 최적화한 멤버쉽 함수

그림 4는 GA가 찾은 입력변수 $e(k)$, $\Delta e(k)$에 대한 멤버쉽함수의 모양을 나타낸 것이다. GA가 찾은 멤버쉽 함수들의 모양이 일반적으로 생각할 수 있는 모양과는 거리가 있지만, 그 성능 면에서는 기존의 제어기를 능가하고 있다. PID 제어기로 제어한 시스템에서의 시뮬레이션 동안의 오차의 평균 $\frac{1}{K}\sum_{k=1}^{K}|e(k)| = 0.2881$이고, 제안된 퍼지제어기의 오차의 평균은 0.1398로 PID 제어기보다 더 나은 성능을 보이고 있는 것을 알 수 있다. 아래의 그림 5는 PID 제어기의 출력과 퍼지제어기의 출력을 비교한 것이다.

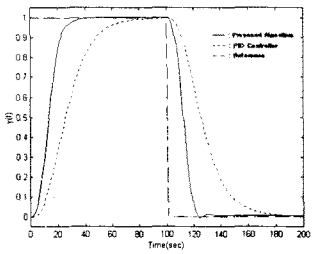


그림 5. 제어기 출력의 비교

3. 결 론

본 연구에서는 전문가의 지식이 전무한 상태에서 퍼지 제어기의 자동적인 설계 방법에 대해 고찰해 보았다. 먼저 퍼지 제어기의 규칙의 수를 결정하는데는 기존 제어기의 입출력 데이터를 클러스터링하는 방법을 사용하였다. 퍼지 제어기에는 Simplified Reasoning을 적용하고 제어기의 여러 파라미터들은 GA를 사용하여 최적화하였다. 그리고 본 논문에서 제안된 퍼지 제어기가 기존의 제어기 보다 우수함을 확인할 수 있었다. 앞으로의 연구방향으로는 유전자 알고리즘의 적용에 있어서 강인성에 대한 성능 지표를 사용하여 응답도 빠르고 강인한 성능의 퍼지 제어기를 설계하는 것과 제어기의 주파수 특성 고찰을 통해 더욱 개선된 제어기의 설계방법을 연구해보는 것이다.

[참 고 문 헌]

[1] Ching-Chang Wong and Chia-Chong CHEN, "A Clustering-based Method for Fuzzy Modeling", IEICE Trans. on Information & Systems, V.E82-D N.6, 1058-1065, 1999

[2] Dorf, Richard C. and Bishop, Robert H. Modern Control Systems, Addison Willey, 1998

[3] S. Chiu., "Fuzzy Model Identification Based on Cluster Estimation", Journal of Intelligent & Fuzzy Systems, Vol. 2, No3, Sept. 1994

[4] Ronald R. Yager, "Essentials of Fuzzy Modeling and Control", John Wiley & Sons, Inc., 1994

[5] D.E. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning", Addison Wesley, 1989

[6] K.S. Tang and K.F. Man, S. Kwong and Q. He, "Genetic Algorithms and their Applications," IEEE Signal Processing Magazine, pp22-37, Nov. 1996.