한국퍼지 및 지능 시스템학회 논문지 1998, Vol. 8, No. 4. (1998) 36~42

이동로봇의 제어를 위한 진화형 퍼지 시스템의 실험적 분석

An Empirical Analysis of Evolutionary Fuzzy System for Mobile Robot Control

이승익 · 조성배

Seung-Ik Lee and Sung-Bae Cho

연세대학교 컴퓨터과학과

요 약

동적으로 변화하는 환경속에서 적절히 적응하며 행동하는 이동로봇의 제어기를 구성하기는 어렵다. 이를 위하여 진화적 방식을 적용하여 제어기를 구성하고자 하는 연구가 활발히 진행되고 있는데 아직까지는 진화결과 구성된 제어기의 정확한 동작 메커니즘에 대한 분석은 미비한 실정이다. 이 논문에서는 진화결과 구성된 제어기의 동작 메커니즘을 오토메타를 통하여 체계적으로 분석하였다. 그 결과 진화방식은 주어진 문제를 적절히 해결할 수 있는 제어기를 만들어 낼 수 있음을 알 수 있었다. 이때 주어진 문제를 여러 부문제로 세분하고 각 부문제를 해결하는 적절한 메커니즘을 획득하였으며 이러한 메커니즘의 상호작용을 통하여 주어진 전체 문제를 해결하고 있음을 알 수 있었다.

ABSTRACT

It is difficult to construct a controller for a mobile robot so that it can adapt appropriately in dynamically changing environment. To solve this problem, extensive research has been actively performed to construct a controller by evolutionary method, but few results has come out about the behavioral mechanism of the evolutionarily constructed controller. This paper attemps to systematically analyze the mechanism of the controller constructed by evolution. As a result, we have found that evolution can produce a controller that can solve a given problem properly, where given problem is divided into several sub problems and adequate mechanisms emerge for each sub problem. The whole problem has been solved through the complicated interactions of these mechanisms.

1. 서 론

상태 공간의 표현을 이용하여 현재위치를 인지하고 경로를 계획한 후 모터를 구동시키는 전통적인 인공지능의 접근방식으로는 끊임없이 변화하는 환경에 적절히 대처할 수 있는 이동로봇의 제어기를 구축하기는 어렵다. 이를 해결하기 위해서 Brooks[2] 등은 이제까지의 전통적인 인공 지능 접근 방식에 반기를 들고, 센서로부터 즉각적인 반응에 의해 모터를 구동시키는 행동기반 인공 지능의 접근 방식을 로보틱스에 적용하고 있다. 표현을 전적으로 무시하는 극단을 취한 문제도 있기는 하지만, 곤충이나 쥐가 미로를 찾는데 그러한 표현이나 경로계획을 수행하지는 않을 것이라는 데에서 그 타당성을 찾아 볼 수 있다.

이러한 접근 방식의 핵심은 가능한 한 로봇의 이동에 도움이 되는 정보를 설계자가 부여하지 않고 로봇 스스로가 자율적으로 학습해 나가도록 함으로써 변화하는 환경 속에서 로봇 스스로 환경에 적응하는 능력을 가질 수 있도록 하자는 데에 있다. 이의 한 방법으로서 행동기반 인공지능적 접근 방식을 취하여 로봇 스스로가 환경에 적응하는 능력을 갖는 제어 시스템은 이미 구축된 바 있는데[1], 이 논문에서는 진화결과 획득한 제어 메커니즘을 오토메타를 통해 분석함으로써 로봇이 주어진 환경에 적절히 대응하는 과정을 보이고 그러한 적응적 행동이 진화과정 중에서 로봇이 획득한 여러 행동전략의 상호 협동을 통해서 얻어진다는 사실을 보인다.

이 논문의 구성은 다음과 같다. 2장에서는 이 논문에서 모델로 사용한 이동로봇과 그 시뮬레이터에 대하여 살펴보고 3장에서는 이를 제어하기 위하여 사용한 퍼지 제어기와 이의 진화적 구성 방법에 대하여 살펴본다. 4장에서는 실험결과 및 진화결과 구성된

※이 논문은 1997년 한국학술진흥재단의 공모과제 연구비에 의하여 연구되었음.

제어기의 실험적 분석을 행하고 5장에서는 결론을 내린다.

2. 행동기반 이동 로봇 : Khepera

Khepera 로봇은 그림 1과 같이 주변 장애물과의 거리를 감지할 수 있는 8개의 적외선 감지 센서를 몸체의 전, 후, 좌, 우에 갖고 있으며, 이와 쌍을 이루어 빛을 감지할 수 있는 8개의 감지 센서를 장착하고 있다. 또한 두 개의 바퀴가 있어 어느 방향으로나 이동이 가능하다. Khepera 시뮬레이터[7]는 실제 로봇의 행동을 제어할 수 있도록 설계된 것으로 이를 사용하여 얻은 제어기는 실제 로봇에 쉽게 장착할 수 있도록 설계되었다.

이 시뮬레이터는 0~1023까지의 값을 거리 감지 센서로부터 얻을 수 있으며, 거리 감지 센서의 값이 0인 경우는 아무런 장애물도 감지되지 않은 상태를 의미하고, 값이 1023인 경우는 장애물이 아주 가까이 있다는 것을 의미한다. 그 사이의 값들은 감지 센서와 장애물 사이의 거리에 반비례하여 증가한다. 거리 감지 센서와 한 쌍으로 되어 있는 빛 감지 센서는 50~500까지의 밝기를 감지할 수 있으며 센서 값이 500인 경우는 광원에서 아주 멀리 떨어져 어두운 경우이고, 센서값이 50인 경우는 광원에 아주 가까이 있어 밝은 경우이다. 본 논문에서는 거리 감지 센서로부터의 입력만을 사용한다. 또 바퀴는 좌, 우 모두 -10에서 +10까지의 속력을 낼 수 있다.

3. 진화형 퍼지제어기

퍼지 제어 시스템[3,9]의 규칙은 IF(조건부) THEN (결론부)로 표현되는데, 본 논문에서 조건부는 8개의 거리 감지 센서값을 소속 함수에 의해서 표현한 변수들의 조건을 명시하고, 결론부는 이에 따른 두 모터의 값을 또다시 소속 함수에 의해 표현한 것이다. 소속 함수로는 삼각형 모양을 사용하였다.

i번째 규칙의 j번째 입력 변수에 설정된 퍼지 소속 함수를 I_{ij}라하고 x_j를 j번째 입력변수, 삼각형의 꼭지점을 c_{ij}, 밑변의 양끝점을 a_{ij}, b_{ij}라 할 때 소속 함수 I_{ij}는 식 (1)과 같이 정의된다.

$$I_{ij} = \begin{cases} \dfrac{1}{(c_{ij}-a_{ij})}(x_j - a_{ij}), & \text{만약} \quad a_{ij} \le x_j \le c_{ij} \\ \dfrac{1}{(c_{ij}-b_{ij})}(x_j - b_{ij}), & \text{만약} \quad c_{ij} \le x_j \le b_{ij} \\ 0 \qquad\qquad , & \text{기타} \end{cases} \tag{1}$$

퍼지 추론은 Correlation Minimum 방법[6]을 사용하였다. i번째 규칙의 퍼지 추론값 μ_i는 n이 입력변수의 수 일 때 식 (2)와 같이 정의된다.

$$\mu_i = \mathrm{MIN}(I_{i0}(x_0), I_{i1}(x_1), \cdots, I_{ij}(x_n)) \tag{2}$$

역퍼지화는 식 (3)과 같이 간략화된 Centroid Defuzzification[6] 방법을 사용하였다.

$$y^*_k = \frac{\sum_{i=0}^{m} \mu_i \bar{y_i}}{\sum_{i=0}^{m} \mu_i}, \quad m = \text{규칙의 수} \tag{3}$$

이 논문에서 사용한 퍼지화 및 역퍼지화 과정의 예가 그림 2에 나타나 있다. 이 그림에서 퍼지 시스템은 8개의 입력을 받고 2개의 출력을 내보낸다. 각 입력 변수 X_0~X_7의 값이 그림과 같을 때 입력 변수가 각 규칙에 형성된 조건을 만족하는 정도 중에서 최소값을 취하는 과정이 Correlation Minimum을 이용한 퍼지 추론 과정이다.

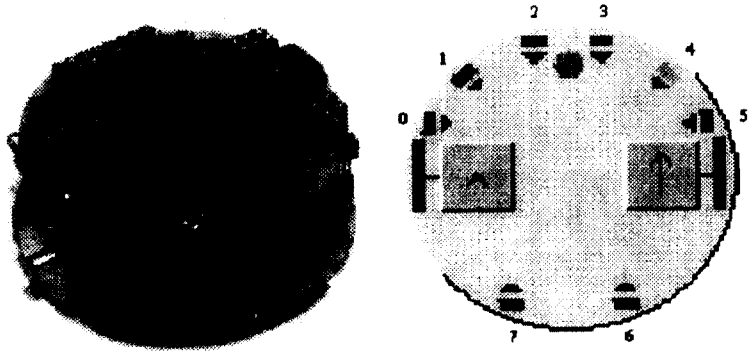

그림 1. 케프라 로봇 및 간략형 표현

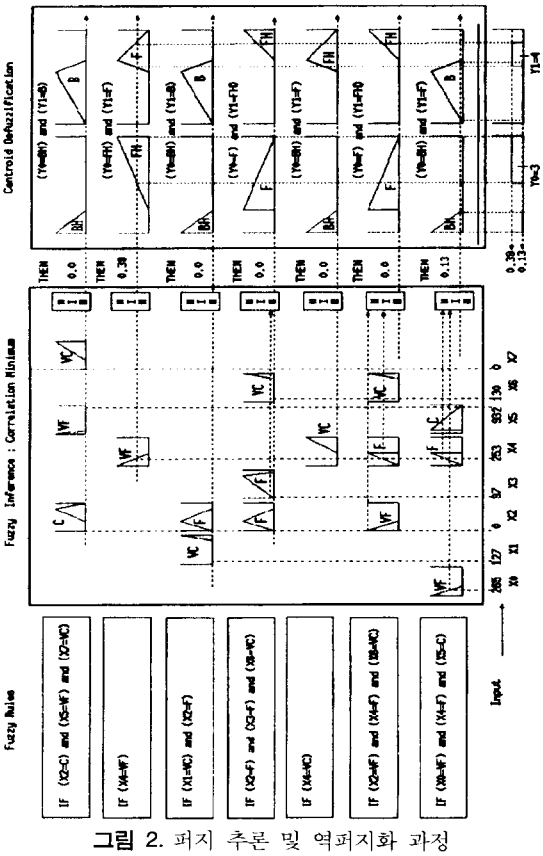

그림 2. 퍼지 추론 및 역퍼지화 과정

그림 2에서는 두 번째 규칙(0.39)과 일곱 번째 규칙(0.13)의 퍼지 추론값이 0을 초과하였고 나머지 규칙들은 0인 추론값을 내고 있다. 이 퍼지 추론값을 출력변수에 형성된 퍼지 집합의 중심점 값에 곱하고 그 무게 중심을 잡는 과정이 식 (3)을 이용한 역퍼지화 과정인데, 그림 3에서는 각각 3과 4의 출력값을 내고 있다.

$$y_0^* = \frac{0.13 \times (-8.76) + 0.39 \times 5.0}{(0.13 + 0.39)} = 3 \qquad (4)$$

$$y_1^* = \frac{0.13 \times (-3.68) + 0.39 \times 4.0}{(0.13 + 0.39)} = 4 \qquad (5)$$

Khepera 로봇은 최대 8개의 거리감지 센서를 입력 변수로, 두 개의 모터에 대한 출력을 출력변수로 갖는 다수의 퍼지 규칙으로 제어되는데, 규칙의 수나 각 규칙에서 사용되는 입력 변수의 수 및 소속 함수의 형태 등을 적절히 결정하기 위하여 유전자 알고리즘을 사용하였다.

생물의 진화과정을 모방한 유전자 알고리즘[5]은

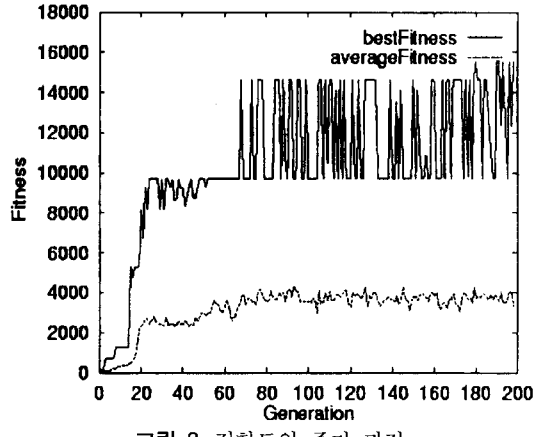

그림 3. 적합도의 증가 과정

기본적으로 문제를 해결하기 위해서 해답의 표현(유전자형)을 집단으로 유지한다. 이들로부터 교차나 돌연변이라는 연산자를 통해 새로운 유전자형을 형성하고 이 중 문제해결을 위한 적합도가 높은 것들을 선택하여 새로운 집단을 형성하면서 점차적으로 문제를 해결해 나간다.

이 논문에서는 Khepera 로봇을 제어하는 퍼지 시스템의 매개 변수를 유전자형으로 표현하고 이들 중 적합도가 높은 것들을 교차와 돌연변이시켜 새로운 유전자 코드를 만든 후 이들을 다시 퍼지 시스템에 적용하는 방식으로 최적의 매개 변수를 구했다. 자세한 인코딩 방법은 참고 문헌[1]을 참조하기 바란다.

주어진 유전자형이 얼마나 원하는 해답에 가까운지를 결정하기 위한 적합도는 다음과 같이 결정하였다.

$$\text{적합도} = \alpha(\text{충돌 회수}) + \beta(\text{움직인 거리}) + \gamma(\text{규칙의 개수}) + \delta(\text{소속 함수의 개수}) + \varepsilon(\text{주요 지점 통과 회수})$$

단, α= -3, β= 1, γ= -100, δ= -10, ε=500

적합도의 각 계수는 여러 번의 실험을 통하여 결정하였고 움직인 거리가 크고, 도중 도중의 주요지점을 많이 지날수록 적합도가 커지도록 하였다. 반면에 로봇이 벽에 충돌한 회수가 크다거나 제어를 위한 규칙의 개수 및 소속 함수의 개수가 클수록 적합도를 떨어뜨리도록 하였다.

4. 실험 결과 및 분석

Khepera 로봇의 시뮬레이터[7]는 C++ 언어로 프로그램 되었으며, 실험은 SUN SparcStation 10과 Pentium 133 MHz 칩을 장착한 PC에서 실행하였다. 먼저

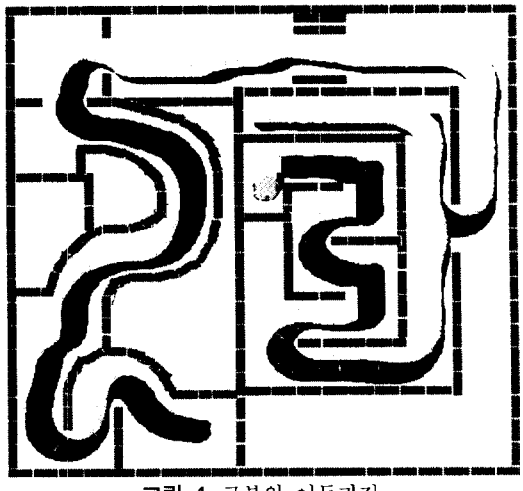

그림 4. 로봇의 이동과정.

초기에 200개의 유전자형을 임의로 만들고 각각에 대한 퍼지 제어기를 구축한 후 로봇을 구동시켜 최대 5000 센서 샘플링 시간동안 수행시키면서 각 개체의 적합도를 계산하였다.

그림 3은 세대에 따른 적합도의 변화를 보여준다. 초기에는 적합도가 진동하면서 상승하다가 64세대 이후 급진적으로 높은 적합도를 갖는 돌연변이적 개체가 나타났다가 사라지곤 하였다. 우수개체 보존 전략을 사용하였는데도 우수개체가 사라지는 경향이 있는 것은 시뮬레이션 환경을 실제 물리적 환경과 비슷하게 하기 위하여 노이즈를 많이 첨가하는데다 입력변수에 형성된 퍼지집합을 4가지로 제한하였기 때문이다.

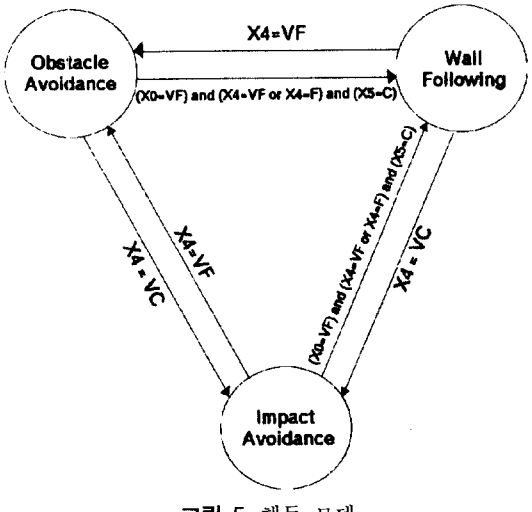

그림 5. 행동 모델

67~70세대에서 본 실험의 최종 목표 지점에 도달하는 개체가 처음으로 발생하였는데 이는 67번째 세대의 86번째 개체로서 총 10개의 규칙 중 7개의 유효한 규칙으로 구성되었다. 최종적으로 형성된 규칙에 대한 자세한 사항은 참고 문헌[1]을 참조하기 바란다.

그림 4는 이 개체의 움직임을 보여주는 것이다. 직접적으로 각각의 제어를 위한 규칙을 의도적으로 만들어주지 않았음에도 불구하고 목표지점까지 도달하는데 필요한 우회전이나 좌회전, 180도 회전 등을 적절히 수행함을 볼 수 있다. 이들 각각의 행동에 사용된 주요한 규칙들은 다음과 같다.

규칙 2	IF (x_4=VF) and (y_0=FH) THEN (y_1=F)
규칙 5	IF (x_4=VC) and (y_0=BH) THEN (y_1=F)
규칙 7	IF (x_0=VF), (x_4=F) and (x_5=C) THEN (y_0=BH) and (y_1=F)

로봇이 목표지점까지 도달하기 위해서 여러 부문제를 해결하여야 하는데 분석결과 로봇은 그림 5와 같은 행동 모델을 이용하여 부문제들을 해결하고 있었다. 'Obstacle Avoidance' 상태는 규칙 2의 만족도가 0을 초과하였을 경우 활성화되고, 'Wall Following' 상태는 규칙 2와 7의 만족도가 0을 초과하였을 경우 활성화되며, 'Impact Avoidance' 상태는 규칙 5의 만족도가 0을 초과하였을 경우 활성화된다.

4.1 좌회전

(a) 단계 1 (b) 단계 2 (c) 단계 24 (d) 단계 50

그림 6. 좌회전

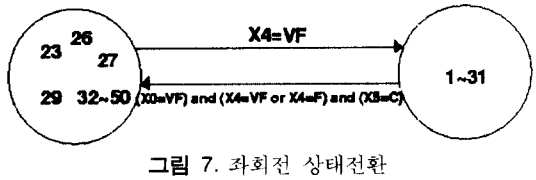

그림 7. 좌회전 상태전환

(a) 단계 1 (b) 단계 4 (c) 단계 36 (d) 단계 41

그림 8. 우회전

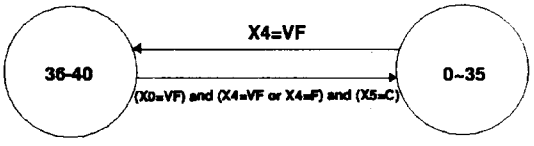

그림 9. 우회전 상태 전환

그림 6은 로봇이 좌회전을 하는 모습이고 그림 7은 같은 기간에 로봇의 상태전환을 표시한 것이다.

전반부에는(1~31) 규칙 2와 7의 만족도가 0을 초과하는 경우 활성화되는 'Wall Following' 상태에서 제어가 이루어지며 후반부에는(32~50) 규칙 2의 만족도가 0을 초과하여 활성화되는 'Obstacle Avoidance' 상태에서 제어가 이루어지고 있어, 결과적으로 이 두 가지 상태의 협동 작업으로 주어진 상황을 진행하고 있다. 특히 단계 23, 36, 27, 29 등에서는 'Wall Following' 상태에서 'Obstacle Avoidance' 상태로의 전환이 일어나고 있는데 이는 그림 6에서 볼 수 있듯이 좌회전 중간에 로봇의 왼쪽에 놓여진 장애물을 감지한 결과로 이 장애물을 회피하기 위하여 'Obstacle Avoidance' 상태로의 전환이 잠깐 잠깐 일어난 것이다.

4.2 우회전

좌회전 상황과 마찬가지로 우회전 상황에서도 로봇은 'Wall Following' 상태와 'Obstacle Avoidance' 상태를 적절히 활용하여 회전하고 있음을 그림 8~9를 통하여 확인할 수 있다. 특이한 점은 좌회전과 우회전이 서로 상반되는 상황임에도 불구하고 'Wall Following'과 'Obstacle Avoidance' 두 가지 상태를 똑같이 이용하여 같은 방식으로 부문제를 해결하고 있다는 점이다. 단순한 원칙으로 여러 가지 복잡한 상황에 쓰일 수 있도록 진화과정을 통하여 창발성을 획득하였음을 볼 수 있다.

4.3 직진

그림 10은 주어진 상황에서 로봇이 직진하고 있는 모습을 단계적으로 보인 것이다. 직진은 로봇에게 있어서 가장 기본적이고 중요하므로 진화과정을 통하여 반드시 획득해야할 행동 중의 하나이다. 로봇은 그림 10의 단계 1에서 단계 97까지 움직이는 동안 규칙 2를 이용한 'Obstacle Avoidance'(규칙 2) 상태에서 움직이고 있다.

(a) 단계 1 (b) 단계 6 (c) 단계 14 (d) 단계 97

그림 10. 직진

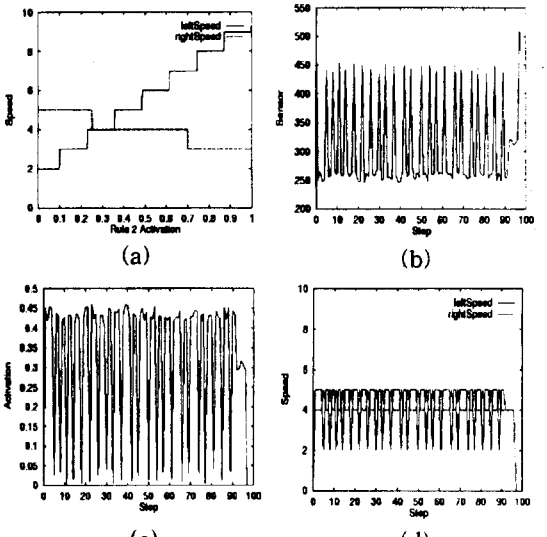

그림 11. 직진. (a) 규칙 2의 만족도에 따른 로봇의 좌·우 속력, (b) 단계별 X_4 센서값, (c) 단계별 규칙 2의 만족도, (d) 단계별 로봇의 좌우 속력

그림 11(a)에서 보면 'Obstacle Avoidance' 상태를 활성화시키는 규칙 2는 만족도에 따라서 직진, 좌회전, 우회전 등을 하도록 구성되었음을 알 수 있다. 규칙 2의 만족도가 0.23 이하이면 오른쪽 모터의 속력은 5를 유지하고 왼쪽 모터의 속력은 그보다 작으므로 로봇은 좌회전을 수행한다. 또한 만족도 0.23~0.35에서는 양쪽 모터의 속력이 4로 동일하므로 직진을 하고, 0.35 이상의 만족도에선 우회전을 수행한다.

그림 10에서 언뜻 보기에 로봇은 직진만을 수행하는 것처럼 보이고 이 경우에 그림 11(a)에서 알 수 있듯이 규칙 2의 만족도가 0.23~0.35 정도인 것으로 추측되나 그림 11(b)에서 보면 규칙 2의 조건으로 참여하고 있는 X_4 센서값이 크게 진동하고 있는 것을 볼 수 있다. 따라서 그림 11(c)와 같이 단계 0~97까지 움직이는 동안 규칙 2의 만족도가 0.025 근처와 0.425 근처를 진동하게 되며 결국 좌회전과 우회전을 번갈아 하게 된다. 이는 그림 10의 직진행동이 실제로는 양쪽 모터의 속력을 동일하게 하여 이동하고 있는 것이 아니라는 것을 짐작케 한다. 실제로 단계별 로봇의 좌. 우 모터의 속력을

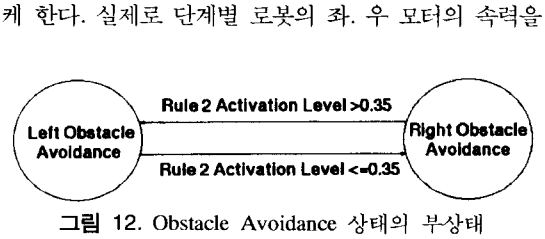

그림 12. Obstacle Avoidance 상태의 부상태

보여준 그림 11(d)에서 로봇은 좌회전과 우회전을 번갈아 하고 있다는 것을 알 수 있다.

결국 로봇은 이와 같은 동작을 반복적으로 수행함으로써, 즉 오른쪽 벽에 가까이 가면(X_4 센서값이 크다 → 규칙 2의 만족도가 낮다) 좌회전을 하고 왼쪽 벽에 가까이 가면(X_4 센서값이 작다 → 규칙 2의 만족도가 높다) 우회전을 하는 동작을 반복적으로 수행함으로써 그림 10과 같이 언뜻 보기에 직진을 하는 듯한 동작을 구현하고 있는 것이다.

이는 로봇이 'Obstacle Avoidance' 제어 상태에서 그림 10의 단계 1에서 97까지 움직이는 동안에 그림 12와 같은 'Obstacle Avoidance'의 부상태의 전환을 통하여 움직임을 의미한다. 예를 들어 그림 11(c)의 단계 1을 보면 규칙 2의 만족도가 대략 0.425 정도라는 것을 볼 수 있고 이는 'Left Obstacle Avoidance' 상태에서 오른쪽 모터는 4정도의 속력으로, 왼쪽 모터는 5정도의 속력으로(그림 11(d) 참조) 우회전을 통하여 왼쪽 장애물에서 멀어진다. 반면에 규칙 2의 만족도가 0.025정도인 단계 6이 되면 'Left Obstacle Avoidance' 상태에서 'Right Obstacle Avoidance' 상태로 전환하여 왼쪽 모터는 2, 오른쪽 모터는 5정도의 속력으로(그림 11(d) 참조) 좌회전을 통하여 오른쪽 장애물에서 멀어진다. 결국 'Left Obstacle Avoidance' 동작과 'Right Obstacle Avoidance' 동작이 반복적으로 작용하여 그림 10과 같이 행동함을 알 수 있다.

(a) 단계 1 (b) 단계 31 (c) 단계 41 (d) 단계 61

그림 13. 돌아나오기

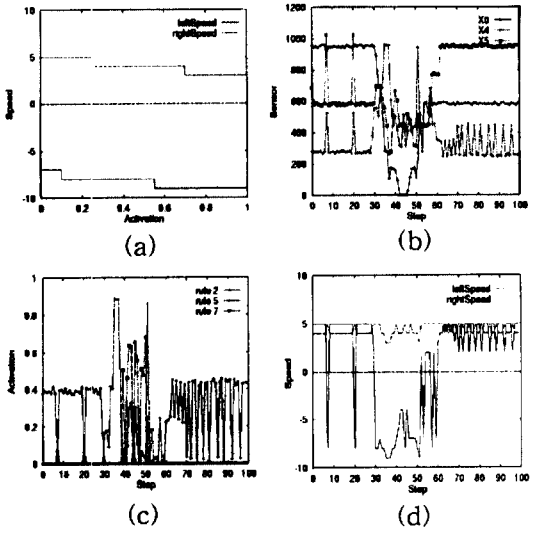

그림 14. 돌아 나오기. (a) 규칙 5의 만족도에 따른 로봇의 좌우 속력, (b) 단계별 센서값, (c) 단계별 규칙의 만족도. (d) 단계별 속력변화

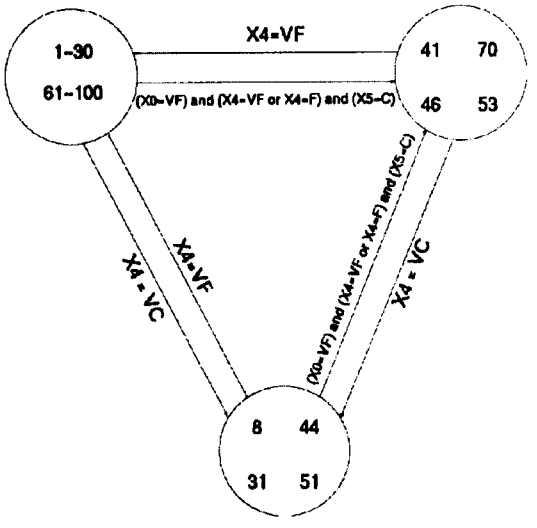

그림 15. 돌아나오기 상태전환

4.4 돌아 나오기

그림 13은 지속적인 탐색을 가능하게 하는 중요한 행동의 하나인 막다른 골목을 빠져 나오고 있는 과정을 보인 것이다. 이 과정에서 로봇은 그림 15와 같이 세 가지 상태 모두를 사용하였으며 특히 막다른 골목에 다다랐을 경우에는 'Impact Avoidance' 상태와 'Wall Following' 상태를 주로 사용하고 있다. 'Impact Avoidance' 상태는 규칙 5로 활성화되며 'Wall Following' 상태는 규칙 2와 7로 활성화된다.

그림 14(a)는 규칙 5의 만족도에 따른 로봇의좌. 우 모터의 속력을 나타낸 것이다. 이 그림에서 알 수 있듯이 규칙 5가 활성화되기만 하면 좌회전을 하도록 구성되어 있다. 그림 14(b)는 각 단계별 센서값을 그림 14(c)는 각 단계별 규칙의 만족도를 보여주고 있고 그림 14(d)는 각 단계별 로봇의 좌. 우 모터의 속도를 보여준다.

그림 14(b)에서 단계 31~60정도까지 로봇이 막다른 골목에 처해 있다는 사실을 센서값의 급격한 변화가 일어나는 것으로 알 수 있다. 그림 15는 주요 단계별 상태 전환을 그림으로 나타낸 것으로 로봇은 단계 30 이전에는 주로 'Obstacle Avoidance' 상태에서 제어되다가 왼쪽 장애물에 가까이 간 단계 30~60 사이에는 규칙 5와 규칙 2,7이 반복적으로 활성화되어

'Impact Avoidance' 상태와 'Wall Following' 상태를 반복하게 된다.

5. 결 론

본 논문에서는 진화형 이동로봇의 제어기를 체계적으로 분석하여 보았다. 최종적으로 구성된 퍼지 시스템은 7개의 규칙으로 이루어졌으며 그 중 3개의 규칙만을 사용하여 목표 지점까지 도달 할 수 있었다. 분석 결과 로봇이 주어진 문제를 해결하는 데 필요한 여러 가지 부문제들을 적절히 해결해 내는 메커니즘을 진화 과정을 통하여 발견해 냈으며 설계자가 제어에 필요한 지식을 명시적으로 표현해 주지 않았음에도 불구하고 최종적으로 구성된 퍼지 규칙에 이러한 문제 해결에 대한 능력을 지니고 있음을 확인할 수 있었다.

로봇은 이러한 주어진 문제를 내부적으로 3가지의 상태전환을 통하여 문제를 해결하고 있었다. 또한 로봇은 주어진 문제를 여러 가지의 부문제로 분류하고 각각의 부문제를 해결할 수 있는 능력을 지녔으며 이러한 부문제 해결 능력이 전체적으로 주어진 문제를 풀 수 있게 하는 바탕이 되고 있다. 어떤 부문제는 하나의 규칙만을 적용하여도 해결이 가능한 문제도 있었지만 대부분의 경우 구성된 규칙 몇 가지의 상호 협동을 통해 해결되고 있었는데 이는 로봇이 각각의 규칙을 상호 독립적으로 사용할 뿐만 아니라 복합적이고 상호 연관적으로 사용하고 있음을 알 수 있다.

참고문헌

[1] 이승익, 조성배, "행동기반 로봇의 퍼지제어기 설계를 위한 진화형 접근 방식," 정보과학회 논문지(B), **24**(12), pp.1400-1407, 1997.

[2] Brooks, R., "A Robust Layered Control System for a Mobile Robot," *IEEE Trans, Robotics and Automation*, **2**(1), Mar. 1986.

[3] Cox, E., *The Fuzzy Systems Hand Book*, AP Professional, 1994.

[4] Dorigo, M., Chnepf, U. S., "Genetics-based Machine Learning and Behavior Based Robotics : A New Synthesis," *IEEE Trans. Systems, Man, and Cybernetics*, **23**(1), pp.141-154, 1993.

[5] Goldberg, D. E., *Genetic Algorithms in Search Optimization & Machine Learning*, Addison-Wesley, 1989.

[6] Kosko, B., *Neural Networks and Fuzzy systems*, Prentice-Hall, 1992.

[7] Michel, O., *Khepera Simulator version 1.0 User Manual*, 1995.

[8] Miglino, O., Lund, H. H. and Nolfi, S., "Evolving Mobile Robots in Simulated and Real Environments," *Artificial Life*, **2**(4), pp.417-434, 1995.

[9] Zadeh, L. A., "Making Computers Think Like People," *IEEE Spectrum*, pp.26-32, 1984.

이 승 익(Seung-Ik Lee)

1995년 : 연세대학교 컴퓨터과학과 졸업(학사)
1997년 : 연세대학교 컴퓨터과학과 졸업(석사)
1997년~현재 : 연세대학교 컴퓨터과학과 박사과정
관심분야 : 퍼지, 진화연산, 인공생명

조 성 배(Sung-Bae Cho)

1988년 : 연세대학교 컴퓨터과학과 졸업(학사)
1990년 : 한국과학기술원 졸업(석사)
1993년 : 한국과학기술원 졸업(박사)
1995년 : 현재 연세대학교 컴퓨터과학과 교수
관심분야 : 신경망, 진화연산, 인공생명