퍼지 및 지능시스템학회 논문지 2005, Vol. 15, No. 5, pp. 626-631

Dynamic Programming을 적용한 트리구조 미로내의 목표물 탐색 알고리즘

Target Object Search Algorithm under Dynamic Programming in the Tree-Type Maze

이동훈 · 윤한얼 · 심귀보

Dong-Hoon Lee, Han-Ul Yoon, and Kwee-Bo Sim

중앙대학교 전자전기공학부

요 약

본 논문에서는 Dynamic Programming(DP)을 적용한 트리구조 미로내의 목표물 탐색 알고리즘을 구현한다. DP는 큰 문제를 이루는 작은 문제들을 먼저 해결하고 작은 문제들의 최적해를 이용하여 순환적으로 큰 문제를 해결한다. 먼저 실험을 위해 적외선 센서를 부착한 소형 이동 로봇과, 'Y'형태로 갈라진 길을 연결한 트리 구조의 미로 환경을 구성한다. 실험에서는 두 개의 서로 다른 알고리즘 - 좌수법, DP - 을 사용하여 목표물 탐색을 시도한다. 마지막으로 위 실험을 통해 DP를 미로 탐색문제에 적용했을 때의 성능을 검증한다.

Abstract

This paper presents the target object search algorithm under dynamic programming (DP) in the Tree-type maze. We organized an experimental environment with the concatenation Y-shape diverged way, small mobile robot, and a target object. By the principle of optimality, the backbone of DP, an agent recognizes that a given whole problem can be solved if the values of the best solution of certain ancillary problem can be determined according to the principle of optimality. In experiment, we used two different control algorithms: a left-handed method and DP. Finally we verified the efficiency of DP in the practical application using a real robot.

Key Words : Dynamic Programming, Left-handed method, Small mobile robot, Tree-type maze.

1. 서 론

현재 로봇 시스템은 공장 자동화 및 우주탐사, 군사 등 광범위한 분야에서 응용되고 있다. 특히 퍼지, 신경회로망, 유전 알고리즘, 강화학습 등의 인공지능 이론은 로봇에게 자율성을 부여하여 로봇 혼자서도 주어진 임무를 수행할 수 있는 단계에 이르렀다. 비단 특수 목적의 로봇 분야뿐만 아니라 청소, 의료보조, 정원관리 등 일상생활에서 응용될 수 있는 연구도 진행되고 있다. 이와 같이 수많은 로봇 중 이동 로봇은 21세기에 가장 성장할 것으로 예상되는 분야 중 하나이지만, 아직 실생활과 산업계에서는 큰 비중을 차지하지 못하고 있다. 이러한 현실의 가장 큰 원인은 agent가 수많은 불확실성을 다룰 수 있는 지능을 확보하지 못하고 있기 때문이다 [1][2].

본 논문에서는 상기에 언급한 것과 같은 문제점을 보완하는 한 가지 방법으로 자율 이동 로봇의 알고리즘에 Dynamic Programming(DP)을 사용하여 'Y'형 미로 블록을 탐색 하는 DP를 적용한 미로탐색알고리즘을 제안한다. 본 논문에서 제안한 DP를 적용한 미로탐색 알고리즘의 실험을 위해 소형 이동 로봇을 제작하였고, 제작된 이동 로봇에 좌수법과 DP를 적용한 미로탐색 알고리즘을 각각 적용하여 목표물 탐색을 하고, 이 실험을 통한 결과를 통하여 DP를 적용한 미로탐색 알고리즘의 성능을 확인하였다.

본 논문의 구성은 다음과 같다. 2장에서는 DP의 기본 개념과 본 논문에서 실험할 미로환경의 한 예를 통하여 agent가 어떻게 goal을 탐색 할 것인지에 대하여 기술하고, 3장에서는 DP를 agent에 적용 하여 목표물을 탐색 하였을 때 예상되는 결과 및 실험에 사용되는 알고리즘에 대하여 언급한다. 4장에서는 실험을 위해 제작된 agent의 구성 및 각 파트별 기능에 대하여 설명하고, 5장에서는 좌수법과 DP를 이용한 미로탐색 알고리즘을 agent에 적용한 후 각각 실험을 한 후 그 실험결과에 대하여 언급한다. 6장에서는 결론 및 향후 과제에 대하여 논한다.

접수일자 : 2005년 8월 26일
완료일자 : 2005년 10월 7일
본 연구는 산업자원부의 뇌신경정보학연구사업의 '뇌정보처리에 기반한 감각정보 융합 및 인간행위 모델 개발'의 연구비 지원으로 수행되었습니다. 연구비 지원에 감사드립니다.

2. Dynamic Programming의 개념 및 실제 사용 예

2.1 Dynamic Programming의 기본적인 개념

Dynamic Programming(DP)의 특징은 첫째로 문제의 순환적인 성질을 이용한다는 것이다. DP는 큰 문제를 이루는

작은 문제들을 먼저 해결하고 작은 문제들의 최적해를 이용하여 순환적으로 큰 문제를 해결한다. 여기서 순환적이라는 의미는 어떤 작은 문제의 해가 그보다 큰 문제의 해를 구하는데 사용되고 또 이렇게 구해진 큰 문제의 해는 그 보다 더 큰 문제의 해를 구하는데 이용되는 것이다. 둘째 DP에서는 중복된 계산을 하지 않는다. DP는 문제의 순환적인 성질 때문에 이전에 계산되어졌던 작은 문제의 해가 다른 어딘가에 필요하게 되는데 이를 위해 DP에서는 이미 해결된 작은 문제들의 해를 저장 공간에 저장하게 된다. 그리고 이렇게 저장된 해들이 다시 필요할 때 마다 해를 얻기 위해 다시 문제를 재계산하지 않고 테이블의 참조를 통해서 중복된 계산을 피하게 된다. 셋째로 DP가 어느 문제에나 적용될 수 있는 것은 아니다. 주어진 문제가 최적화의 원칙을 만족해야만 DP를 적용할 수 있다. 최적화의 원칙이란 어떤 문제에 대한 해가 최적일 때 그 해를 구성하는 작은 문제들의 해 역시 최적이어야 한다는 것이다. DP의 방법 자체가 큰 문제의 최적 해를 작은 문제의 최적해들을 이용하여 구하기 때문에 만약 큰 문제의 최적해가 작은 문제들의 최적화의 해들로 구성되지 않는다면 이 문제는 DP를 적용할 수 없다. DP 문제는 반드시 이 최적화의 원칙을 만족해야 하기 때문에 문제를 해결하려고 하기 전에 해당 문제가 DP를 적용하여 풀릴 수 있는 문제인가를 먼저 검토해 보아야 한다 [3][4][5][6].

2.2 DP의 실제 사용 예

일반적으로 Dynamic Programming(DP)은 최적의 value function, 최적의 policy function, recurrence relation과 boundary condition으로 이루어져 있다. 그림 1은 간단한 cost path problem을 설명하는데 도움이 되는 실례이다.

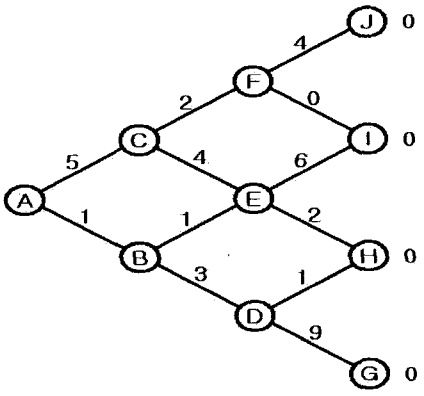

그림 1. 간단한 비용 경로 문제.
Fig. 1. A simple cost path problem.

우리는 agent가 action을 선택하여 minimum cost를 축적함으로서 goal state $g \in G \in S$ 로 주행 하도록 agent를 학습시켜야 한다. 여기서 G는 goal state의 집합, S는 모든 state의 집합, g는 goal이다. 그림 1에서 최적의 value function $V(s)$는 "*states*에서 *goal g*로가는 최소의 *value*"이고, 순환관계에 의하여 state s의 minimum cost는 가장 잘 선택 된 action a의 cost와 같다 [3]. 그리고 다음 state의 minimum cost는 action a에 의하여 지시되어진다.

$$V(s)=\min_{a\in actions}cost(s,a)+V(next(s,a)),\ \forall s\in S-G \quad (1)$$

위 수식에서 $cost(s,a)$는 action a에 의해 지시 되는 state의 cost를 나타내고, $next(s,a)$는 action a 에 의하여 지시하는 state s의 다음 state의 cost를 나타낸다 [4]. goal state value를 정의하면 다음 식과 같다.

$$V(s)=0,\forall s\in G. \quad (2)$$

방정식 (2)는 boundary condition을 나타낸다. 최적의 policy function $\pi(s) = a\ such\ that\ V(s) = \min a\in actions\ cost(s,a) + V(next(s,a))$ 이다 [5].

우리가 찾고자 하는 목표지점인 goal을 그림 1에서 Ⓗ, start position을 Ⓐ라고 가정하자. 첫 번째 state에서 agent가 Ⓐ에서 Ⓑ로 action을 취한다면, 그것은 방정식 1과 2에 의하여 산출되어 진다.

$$\begin{aligned}V(s_0)&=\min_{actions}cost(s_0,down)+V(next(s_1,a_{next}))\\&=\min_{a\in action}1+V(next(s_1,a_{next}))\end{aligned} \quad (3)$$

여기서, s_0, s_1, s_2, s_3는 S의 원소이다. $V(s_0)$를 정의하기 위해서 $V(next(s_1, a\ next))$를 결정하는 것이 필요하다. 두 번째로 agent가 Ⓑ에서 Ⓔ로 이동한다면 식 4와 같이 산출된다.

$$\begin{aligned}V(s_1)&=\min_{a\in actions}cost(s_1,up)+V(next(s_2,a_{next}))\\&=\min_{a\in actions}1+V(next(s_2,a_{next}))\end{aligned} \quad (4)$$

세 번째로, agent가 Ⓔ에서 Ⓗ로 이동한다면,

$$\begin{aligned}V(s_2)&=\min_{a\in actions}cost(s_2,down)+V(next(s_3,a_{next}))\\&=\min_{a\in actions}1+V(next(s_3,a_{next}))\end{aligned} \quad (5)$$

마지막 state가 0값이면 agent는 종결조건을 만족하기 때문에 s_3를 goal state로 인식하고 식6과 같다.

$$V(s_3)=0 \quad (6)$$

따라서 방정식 3, 4, 5 에 따라 $s_0 \rightarrow s_3$까지의 minimum cost는 식 7과 같다.

$$V(s_0\rightarrow s_3)=1+1+2+0=4. \quad (7)$$

우리가 Ⓗ를 goal state로 가정하였기 때문에 agent의 학습이 완료 되었다.

3. Dynamic Programming를 적용한 미로탐색 알고리즘

이번 장에서는 트리형태의 미로 안에서 goal을 찾기 위해 Dynamic Programming(DP)을 적용한 탐색 알고리즘을 소개한다. 그림 2는 본 논에서 제안한 DP를 이용한 미로탐색 알고리즘을 실험을 하기 위한 미로 환경의 한 예이다.

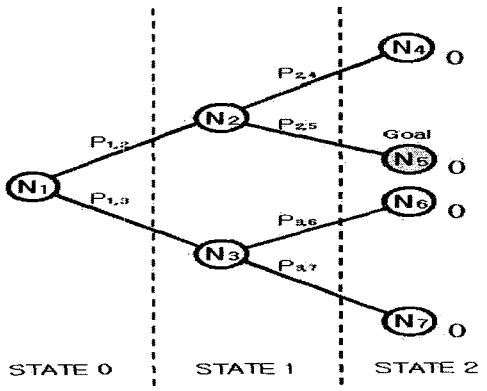

그림 2. 트리형 미로 환경의 예.
Fig. 2. An example of the tree-type maze environment.

그림 2의 미로 환경을 agent가 DP을 사용하여 탐색 한다면 agent는 시작점 N_1에서 랜덤 선택 방식으로 $P_{1,2}$와 $P_{1,3}$의 두 가지 action을 취할 수 있다. 여기서 agent는 두 개의 table을 생성하는데 그 첫 번째는 각 action의 값을 저장하고, 다른 하나는 state가 종결지점인지를 저장한다. 여기서 agent가 $P_{1,3}$의 action을 취했다면 agent는 state N_3으로 이동하고 N_3의 state에서 두 가지 조건을 검사 하는데, 그 첫 번째는 N_3의 state가 "goal state인지 아닌지" 확인하는 것이고, 두 번째는 "boundary condition인지 아닌지"를 확인 하는 것이다. N_3의 state가 두 가지 조건이 아닌 것이 확인되면 $P_{1,3}$의 action에 값 50을 할당한다. N_3의 state에서 조건은 "goal ≠ Yes", "boundary condition ≠ Yes" 이므로, 또다시 랜덤하게 action을 선택하여 다음 state로 이동하게 된다. 예를 들어 $P_{3,6}$의 action을 취했다고 하면, N_6의 state로 이동하게 되고 이전과 같은 방식으로 N_6의 state가 "goal인지 아닌지", "종결조건인지 아닌지"를 확인한다. 이번 경우에는 "goal ≠ Yes", "boundary condition = Yes" 이므로, $P_{3,6}$의 action에는 "100"의 값을 할당한다. "goal ≠ Yes", "boundary condition = Yes"인 경우에는 이전 state로 이동하고 탐색하지 않은 반대편의 갈림길로 이동함으로 N_3의 state에서 $P_{3,7}$의 action을 선택하게 된다. 여기서도 "goal ≠ Yes", "boundary condition = Yes"인 경우이기 때문에 $P_{3,7}$의 action에는 "100"의 값을 할당한다. 여기서 state N_3으로 올수 있는 유일한 action은 $P_{1,3}$이기 때문에 $P_{1,3}$또한 "100"의 값을 할당한다. N_1의 state에서 $P_{1,3}$의 action은 "100"이 할당되었음으로 아직 값이 할당되지 않은 $P_{1,2}$의 action을 선택하여 state N_2로 이동하고 앞서 말한 것과 같은 방식으로 N_2의 state가 "goal = Yes or goal ≠ Yes?", "boundary condition = Yes or boundary condition ≠ Yes?" 인지를 확인하고 action에 값을 할당한다. 마지막으로 agent가 action $P_{3,5}$를 선택 하였을 때 agent는 state N_5에 도달하고, "goal = Yes", "boundary condition = Yes"로 식 2의 조건을 만족하기 때문에 $P_{2,5}$의 action에는 "0"값을 할당하고, 그 반대편 action인 $P_{2,4}$의 action에는 "100"값을 할당한다. 또한 state N_2로 올수 있는 유일한 길인 action $P_{1,2}$의 값을 "0"을 할당한다. 이와 같이 agent의 첫 번째 주행에서 각각의 action과 state의 값을 모두 할당하였기 때문에 두 번째 주행에서 agent는 값이 "0"인 action을 선택하고, 종결지점인지만을 확인하면 시작위치에서 최단거리로 goal지점에 도달할 수 있다. agent가 action을 선택할 때 랜덤 선택 방식을 사용하기 때문에 goal을 찾을 확률 $P_{x,y}$는 식 8과 같다.

$$P_{x,y} = 100/2^{n+1} \tag{8}$$

여기서 $x = n^{th} state$, $y = (n+1)^{th} state$ 이다.

4. 실험용 이동로봇

본 논문에서 제안한 Dynamic Programming(DP)을 적용한 미로탐색 알고리즘의 실험을 위하여 소형 이동로봇을 제작하였다.

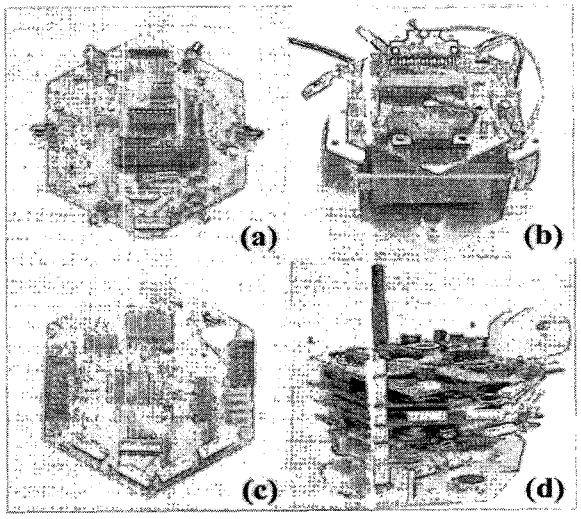

그림 3. (a) 센서 구동부, (b) 모터 구동부
(c) 메인 제어부 (d) 로봇 외형.
Fig. 3. Sensor emitter-detector pair, (b) motor driver (c) a main controller, and (d) the appearance of robot.

그림 3은 본 논문의 실험을 위해 제작된 소형 이동 로봇의 각 파트별 보드와 외관을 보여준다. 실험을 위하여 제작된 소형 이동로봇은 메인 제어부와 전원부, 모터 및 센서 구동부로 구성되어 있다 [7]. 좁은 미로에서도 무리없는 이동을 위하여 가로 : 10cm, 세로 : 12cm, 높이 18cm의 소형으로 제작 하였고, 360° 사각이 없는 전 방향 감지를 위해 6축으로 적외선 센서를 부착하였다 [8]. 7.2V의 리튬이온 전지 2개를 사용하여 정전압을 공급하고, 모터 구동부는 NMB PG25L-024 stepping motor(drive voltage 12V, drive method 2-2 Phase, 0.495 step angle)를 사용한다 [7][8]. 메인제어부에서는 센서에서 입력되는 신호를 입력받아 현재 로봇의 위치를 판단하고, agent가 주행할 때 구간별 Cost를 결정한다. 그림. 3a는 6축으로 적외선 센서를 부착한 센서 구동부이고, 그림. 3b는 메인 제어부이다. 그림. 3c는 리튬이온 전지를 부착한 로봇 바디와 모터 구동부의 모습이다. 이 실험용 이동로봇은 조립형으로서 모터 구동부 위로 각 보드가 조립되며 그림. 3d는 각 보드별 조립이 완료된 모습이다. 제작된 이동로봇은 기능별 보드 조립형으로 제작하여 필요시에

메모리 확장 및 교체, 통신기능 추가 등 여러 형태로 변형이 가능 하도록 설계 되었다. 또한 각 보드별로 전원선 및 통신선이 호환되도록 설계하여 사용하지 않는 보드는 분리하지 않아도 다른 보드에 영향이 미치지 않도록 하였다 [8][9].

5. Dynamic Programming를 적용한 미로탐색 알고리즘과 좌수법의 목표물 탐색 실험.

5.1 실험 환경 및 시나리오

본 논문의 실험에서 agent로 시작점에서부터 goal 탐색을 시도 하였다. Dynamic Programming(DP)을 적용한 미로탐색 알고리즘으로 atent가 미로를 탐색 할 때 종결지점에서 목표물인 goal 지점과 그 이외의 종결지점을 구분 할 수 있도록 goal 지점에는 흰색을 칠하고 그 이외의 종결지점에는 모두 검정색을 칠하였다. 미로 벽은 흰색 하드 보드지를 사용 하였다.

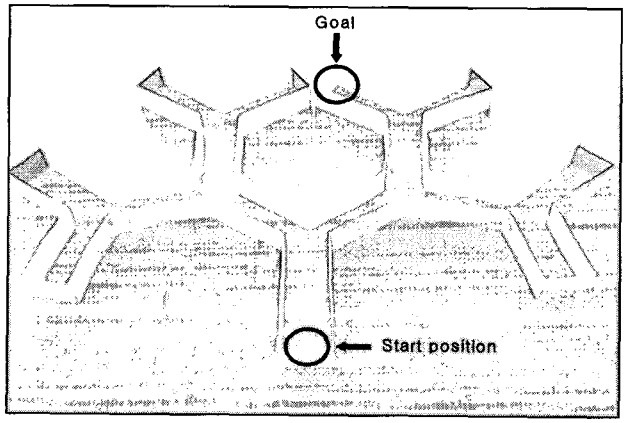

그림 4. 미로환경 외관.
Fig. 4. The appearance of mage environment.

그림4는 본 논문의 실험을 위한 'Y'형 미로 이다. 화살표가 가리키는 지점이 목표물인 Goal지점과 agent가 처음 주행을 시작할 위치인 start position이다. 먼저 좌수법으로 agent가 미로를 탐색 하도록 하고, 다음으로 본 논문에서 제안한 DP를 적용한 미로탐색 알고리즘으로 agent가 미로를 탐색 하도록 한다. 그 다음 좌수법으로 미로탐색을 마친 agent으로 2차 주행을 하고, 마지막으로 DP를 적용한 미로탐색 알고리즘으로 목표물을 찾은 agent을 다시 시작점에서부터 목표물을 탐색 하도록 한다. 모든 탐색 후 3장에서 예상했던 결과를 실제 실험을 통하여 확인한다.

5.2 실험결과

첫 번째로, 좌 수법을 agent에 적용하여 목표물 탐색을 시도하였다. 3장에서 예상되었던 것과 같이 좌 수법은 왼쪽 벽을 무조건 따라가기 때문에 상당한 시간을 소요한 후에 결국에는 goal 지점에 도달할 수 있었다.

그림 5는 agent가 좌수법으로 'Y'형 미로를 탐색할 때 agent의 주행 경로를 보여준다. 두 번째로, Dynamic Programming(DP)을 agent에 적용하여 미로탐색 실험을 하였다.

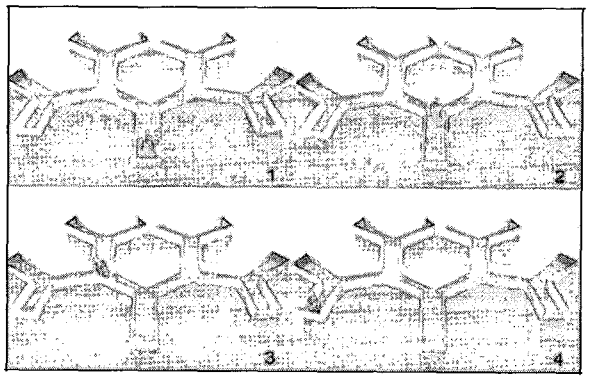

그림 5. 좌수법을 적용한 goal 탐색.
Fig. 5. Ther robot is searching the goal under the left-handed method.

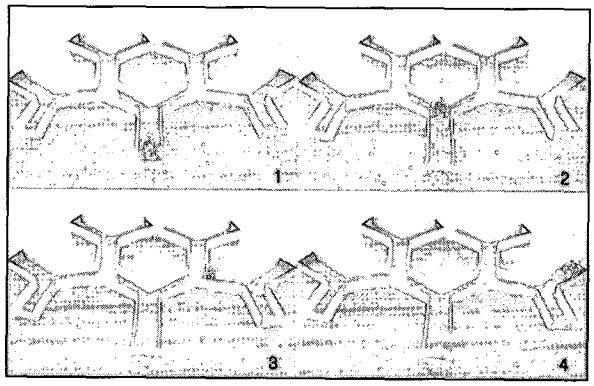

그림 6. DP를 적용한 goal 탐색.
Fig. 6. The mobile robot is searching the goal under DP.

이동 로봇이 시작점에서부터 탐색을 하면서 각 state마다 랜덤하게 action을 취하고 이 action에 의해 지시되어진 다음 state의 값에 따라 현재 위치의 state와 이전의 state사이의 action에 대해 점수를 부여하고 결국에는 골 지점을 찾아 갈 수 있었다. 세 번째로, 처음 좌수법으로 미로탐색을 마친 로봇으로 미로탐색을 다시 실험하였다. 예상했던 대로 그림 5와 동일한 방향으로 동일한 시간을 소비하고 결국에는 목표지점을 찾아갔다. 마지막으로, DP를 적용하여 목표물을 탐색을 완료한 agent를 시작점에서 다시 구동 하여 실험을 하였다.

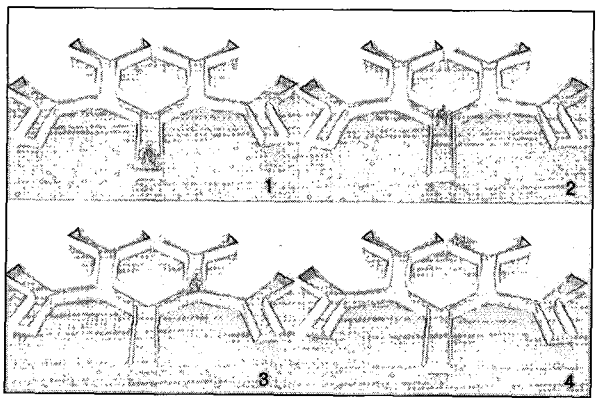

그림 7. DP를 적용한 goal 탐색완료.
Fig. 7. The mobile robot is search completed the goal under DP.

DP를 적용한 agent는 이전에 실험을 통하여 현재 미로내의 각 state와 action의 값을 모두 저장하고 있기 때문에 'Y' 형 미로내의 state와 state의 구역별 최적의 부분 주행로를 결정하고 이 부분 주행로들의 결합으로 agent가 가장 최적화된 주행을 할 수 있었다. 그림 7은 agent가 DP를 사용하여 목표물을 탐색 한 후 두 번째로 목표물을 탐색 하였을 때의 모습을 보여준다.

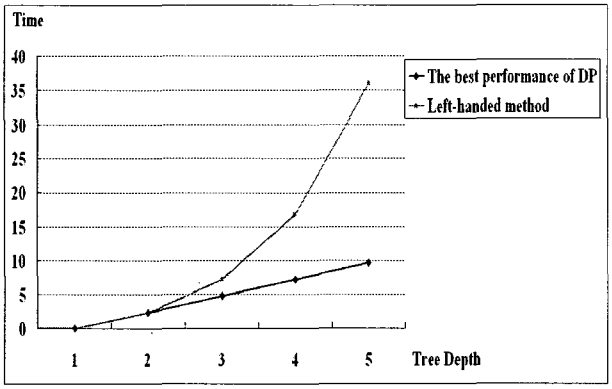

그림 8. DP와 좌수법의 성능비교.
Fig. 8. The performance of DP over the left-handed method.

이 실험에서 확인한 바와 같이 각각의 알고리즘이 goal 지점을 찾은 후 두 번째 탐색을 할 때에는 좌수법을 사용한 agent는 처음과 동일한 방법으로 동일한 시간을 소비하며 골 지점을 찾는 반면에 DP를 적용한 미로탐색 알고리즘으로 미로를 탐색한 agent는 한번 지나간 자리의 값을 모두 기억하고 있기 때문에 시작점에서 출발하자마자 곧바로 goal 지점을 찾아 가는 것을 확인할 수 있었다. 이번 실험 에서 좌수법과 DP을 적용한 agent의 첫 번째 실험에서는 DP를 적용한 agent가 좌수법을 적용한 agent보다 빠른 시간에 목표지점에 도달할 수 있었지만 처음 agent가 미로를 탐색 할 때, 모든 경우에 DP를 적용한 agent가 좋은 성능을 보이는 것은 아니다. DP를 적용한 agent가 갈림길에서 선택한 방향은 랜덤하게 선택한 방향이기 때문에 첫 주행 시에는 좌수법보다 더 나쁜 성능을 보일 때도 있었다. 하지만 본 논문에서 제안한 'Y'형 미로내의 목표물 탐색 알고리즘의 가장 핵심적인 부분은 처음 미로탐색을 마친 agent의 다음 주행의 성능을 평가 하는 것으로 이 부분에 대해서는 충분히 만족할만한 결과를 얻었다.

그림 8은 'Y'형 블록이 증가함에 따라 좌수법을 적용한 agent와 DP를 적용한 agent의 성능을 비교 분석한 것이다. 좌수법을 적용한 agent는 'Y'형 미로가 증가 할수록 탐색 시간은 2에 지수 승으로 증가 하였다. 첫 주행에서는 DP를 적용한 agent도 좌수법과 비슷한 시간을 소요하면서 'Y'형 미로를 탐색 하였다. DP의 첫 주행에서 action을 선택하는 방법이 랜덤 선택 방법이기 때문에 경우에 따라 좌수법보다 효율이 좋을 수도 있고 나쁠 수도 있다. 하지만 Fig. 8에 나타난 것처럼 두 번째 주행을 했을 때에는 좌수법을 이용한 방법과 DP를 이용한 방법은 현격히 차이가 났다. 좌수법을 사용한 agent는 처음 주행한 것과 동일한 결과가 나오지만 DP를 사용한 agent는 시작점에서 목표물까지의 가장 빠른 길로 주행을 하였다. 트리의 개수가 2^n-1개 증가 할 때의 예상 시뮬레이션을 해 본 결과 각각의 알고리즘을 적용한 agent의 처음 탐색 시간은 트리 개수가 2^n-1 개 증가 할 때마다 2에 지수 승으로 증가 하였다. 하지만 두 번째 탐색에서는 좌수법을 적용한 agent는 이전과 동일한 시간을 소비한 반면 DP를 적용한 atent는 한 depth만 증가 할 뿐이었다.

6. 결론 및 향후과제

본 논문에서는 선험적 지식이 없는 'Y'형 미로 공간에서의 goal 탐색을 위해 Dynamic Programming(DP)을 적용한 미로탐색 알고리즘을 제안하였다. 또한 실험을 위해 실제 소형 이동 로봇을 제작하고, 제작된 로 을 통해 이 알고리즘이 임의의 공간에서의 처음 탐색 주행 이후에는 한번 선택한 action에 대한 모든 값을 기억하여 시작점에서 출발하자마자 곧바로 목표 지점을 찾아 가는 것을 확인 할 수 있었다. 이 실험의 결과를 통해 본 논문에서 제안한 DP를 적용한 미로탐색 알고리즘의 성능을 확인할 수 있었다. 본 논문에서는 DP를 적용한 알고리즘을 보다 쉽게 설명하기 위하여 이진트리로 미로를 구성하여 실험을 하였지만 본 논문에서 제안한 방법을 이용하면 어떤 구조의 미로든지 목표지점을 찾아 가는 것이 가능 하다.

서론에서 언급한바와 같이 이동 로봇은 21세기에 가장 성장할 것으로 예상되는 분야 중 하나이지만, 이동로봇의 수많은 불확실성을 다룰 수 있는 지능을 확보하지 못해 실생활 및 산업계에서 큰 비중을 차지하지 못하고 있는 것이 사실이다 [2]. 실생활과 산업계에서 이동로봇의 활발한 성장을 위해서는 본 논문에서 제안한 DP를 적용한 미로탐색 알고리즘과 같이 강화 학습 및 다른 알고리즘도 컴퓨터상의 시뮬레이션이 아닌 하드웨어 상에서 실제로 구현이 되어 여러 복잡한 현실 세계에서 그 성능을 검증 하여야 할 것이다.

참 고 문 헌

[1] 이원창, 옥진삼, 강근택, "웹을 이용한 이동로봇의 원격제어," *한국동력기계공학회지*, vol. 4, no. 4, pp. 78-83, 2000.

[2] 정완균, "이동로봇의 위치인식기술", *한국통신학회지*, vol. 21, no. 10, pp. 67-75, 2004.

[3] J.-S. R. Jang, C.-T. Sun, and E. Mizutani, *Neuro-Fuzzy and Soft Computing*, Prentice Hall, 1997.

[4] C. W. Xu, "Fuzzy model identification and self-learning for dynamic systems," *IEEE Trans. Syst. Man. Cybern.*, Vol. 17, No. 4, pp. 683-689, 1987.

[5] H. Takagi and M. Sugeno, "Fuzzy identification of systems and its application to modeling and control," *IEEE Trans. on Sys. Man and Cybern.*, Vol. 15, pp. 116-132, 1985.

[6] J. H. Holland, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley: MA, 1989.

[7] H.-U. Yoon, S.-H. Whang, D.-W. Kim, and K.-B. Sim, "Strategy of cooperative behaviors for distributed autonomous robotics systems", *Proc.*

of the 10th International Symposium on Artificial Life and Robotics, pp. 151-154, 2005.
[8] H.-U. Yoon and K.-B. Sim, "Hexagon-based Qlearning for object search with multiple robots", *Proc. of the 1st International Conference on Natural Computation*, pp. 713-722,. 2005.
[9] 이동훈, 김대욱, 심귀보, "DARS 로봇에서의 영상전송 시스템 개발," *한국퍼지 및 지능 시스템 학회*, vol. 15, no. 1, pp. 229-232, 2005.

저 자 소 개

이동훈(Dong-Hoon Lee)
2005년 : 순천대학교 전기제어과 공학사
2005년 ~ 현재 : 중앙대학교 대학원 전자전기공학부석사과정

관심분야 : Machine learning, 로보틱스, 이동로봇제어, 홈네트워크시스템

Phone : +82-2-820-5319
E-mail : sky52929@wm.cau.ac.kr

윤한얼(Han-Ul Yoon)
2004년 : 중앙대학교 전자전기공학부 공학사
2004년~현재 : 동대학원 전자전기공학부 석사과정

관심분야 : Processor architecture, DARS, Reinforcement learning
Phone : +82-2-820-5319
E-mail : huyoon@wm.cau.ac.kr

심귀보(Kwee-Bo Sim)
1984년 : 중앙대학교 전자공학과 공학사
1986년 : 동대학원 전자공학과 공학석사
1990년 : The University of Tokyo 전자공학과 공학박사
1991년 ~ 현재 : 중앙대학교 전자전기공학부 교수
2003년 ~ 2004년 : 일본계측자동제어학회(SICE) 이사
2000년 ~ 2004년 : 제어자동화시스템공학회 이사 및 (현) 지능시스템연구회 회장
2003년 ~ 2004년 : 한국퍼지 및 지능시스템학회 부회장
2002년 ~ 현재 : 중앙대학교 산학연컨소시엄센터 센터장 및 기술이전센터 소장
2005년 ~ 현재 : 한국퍼지 및 지능시스템학회 수석부회장
관심분야 : 인공생명, 지능로봇, 지능시스템, 다개체시스템, 학습 및 적응알고리즘, 소프트 컴퓨팅(신경망, 퍼지, 진화연산), 인공면역시스템, 침입탐지시스템, 진화하드웨어, 인공두뇌, 지능형 홈 및 홈네트워킹, 유비쿼터스 컴퓨팅 등
Phone : +82-2-820-5319
Fax : +82-2-817-0553
E-mail : kbsim@cau.ac.kr
Homepage URL : http://alife.cau.ac.kr