Proceedings of the 11th
KACC, October 1996

Distance Profile Histogram과 뉴럴네트워크를 이용한 이동로보트의 주행제어

신무승 김현태 이성렬° 박민용
연세대학교 전자공학과

Abstract

본 논문은 새로운 지역 경로 계획 알고리듬으로 DPH(Distance Profile Histogram)방법과 뉴럴네트워크를 사용한 주행 방법을 제안한다. DPH방법은 격자형 환경 모델을 기반으로 장애물의 존재 유무와 거리정보와 같은 장애물의 기하학적 배치정보를 사용하게 된다. 또한 긴 장애물이나 막힘상황(Dead end)과 같이 지역 경로 계획만으로는 회피하기 어려운 상황에서는 뉴럴네트워크에 의해 학습된 정보에 의해 주행하는 방법을 사용했다.

1. 서론

이동로보트가 미지의 환경에서 안전한 주행을 하기 위해서는 환경을 정확하게 파악하고 분석하여 실시간으로 안전한 주행을 하기위한 알고리듬이 필수적이다. 본 논문에서 사용되는 초음파 센서는 감지되는 물체의 방향이 부정확하고, 근접한 다른 초음파 센서에 의해 영향을 받거나 센서의 중심축을 기준으로 약 15° 이상 기울어진 면은 감지가 되지 않는 단점이 있다. 위의 단점들을 최대한 극복하기 위해 본 논문에 사용된 로보트는 초음파 센서 9개를 로보트 전면에 원형 배치를 하고 서로 다른 샘플 주기를 사용함으로써 대화 현상이 발생하는 것을 최대한 억제 하였다. 이렇게 얻어진 환경에 대한 정보에 의해 목표점까지 장애물을 안전하게 회피할 수 있는 경로를 생성하고 발생된 경로를 따라 로보트가 주행을 하게 된다. 이에 관한 연구는 다음과 같다.

- 모서리 검출 방법[1]
- 위치 힘장 방법[2]
- 벡터장 히스토그램 방법[3]

그러나 이러한 방법들은 장애물의 기하학적 특성을 이용하지 않는다. 이를 위해서 본 논문에서는 장애물의 기하학적 배치정보를 사용하는 DPH(Distance Profile Histogram)방법을 기반으로 일반적으로 지역 경로 계획만으로는 회피하기 어려운 막힘상황은 뉴럴네트워크를 사용하여 미리 정의된 환경의 프로토타입을 학습시킨후 주행하는 방법을 제안한다.

2. DPH(Dsitance Profile Histogram)알고리듬

2.1 격자형 환경 모델의 구성

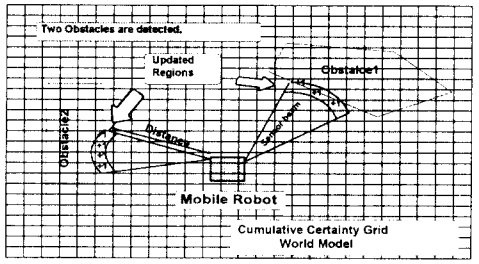

그림 2.1 센서 정보로 갱신되는 환경영역

본 논문에서 사용는 격자형 환경 모델은 빠른 시간내에 확실성 지도를 얻기 위하여 단순화된 CCG(Cumulative Certainty Grid)방법을 이용한다.

2.2 DPH(Distance Profile Histogram)

DPH는 다음에 설명하는 몇 개의 단계를 거쳐 만들어진다. 먼저 그림 2.2와 같이 전체 환경 지도 위에서 로보트의 현재 위치를 중심으로 반경 R의 Active Circle을 설정한다.

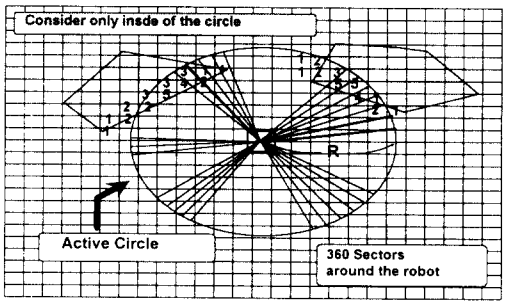

그림 2.2 Active Circle의 설정과 섹터 분할

이 Circle은 1° 간격으로 분할되는 섹터로 나누어진다. 이제 Active Circle안의 각 섹터의 COD(Cumulative Obstacle Density)와 DP(Distance Profile)를 다음의 식으로 구한다.

$$COD(i)=\sum_{k=1}^{n} C_i(k)/n$$

$$DP(i)=\mathrm{Min}[\mathrm{distance}(\mathrm{center\ of}\ C_i(k),\mathrm{center\ of\ robot})] \quad (2.1)$$

, for all k in i-th setor

여기서 $C_i(k)$는 i번째 섹터와 겹치는 k번째 CCG의 값이다. 위에서 구한 COD는 이산적인 특성을 가지므로 다음의 이동 평균법에 의해 평활화(smoothing)하고 임계함수를 이용하여 이치화한다.

$$COD(i)=\frac{\sum_{k=-n}^{n} COD(i+k)}{2n+1} \quad (2.2)$$

$$COD(i)=f_{th}[COD(i)]$$

$$\text{where } f_{th}(k)=\begin{cases}0 & k<\text{threshdd}\\ 1 & k\geq\text{threshold}\end{cases}$$

$$DPH(i)=COD(i)*DP(i) \quad \text{where } i=1\sim360$$

여기서 n은 10이고 k는 4를 사용하였다. 이제 COD와 DP를 곱하여 최종적인 DPH를 식 2.2와 같이 얻는다.따라서 DPH는 장애물이 점유하고 있는 섹터와 점유하고 있는 거리정보를 동시에 표현하는 히스토그램이 된다.

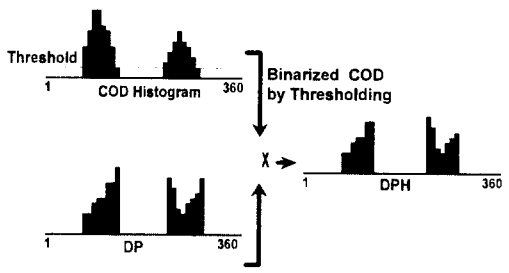

그림 2.3 COD와 DP를 이용한 DPH의 생성

3 지역 경로 계획 알고리듬

3.1 중간 경유점(subgoal) 설정

복수개의 장애물이 있을때 로보트가 안전한 주행을 하기 위해 DPH를 이용한 중간 경유점을 설정한다. 그림 3-4에서와 같이 DPH에는 몇개의 비어있는 영역과 점유된 영역이 존재하게 된다. 먼저 비어있는 영역을 찾고 그 영역의 크기가 너무 작으면 버리고 다음 비어있는 영역중 목표점에 가장 가까운 방향의 영역을 목표영역으로 정한다. 선택된 영역에서 그림 3-4에서와 같이 목표영역의 양쪽 끝 섹터의 각 θ_R ,θ_L을 구하고 두 장애물의 양 끝점과 두점의 중점 (Mid_X ,Mid_Y)를 다음의 식으로 구한다.

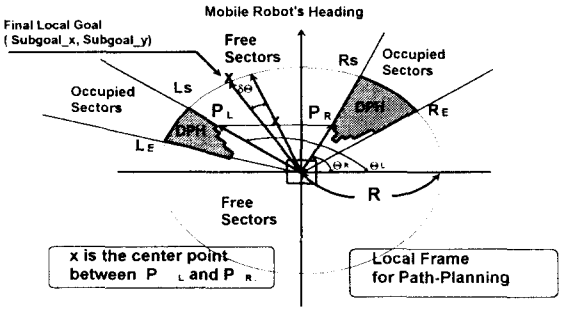

그림 3.1 DPH를 이용한 기하학적 지역 경로 계획

$$Mid_x = \frac{P_R(x)+P_L(x)}{2}$$
$$P_R(x)=\cos(\theta_R)*DPH(\theta_R),\quad P_L(x)=\cos(\theta_L)*DPH(\theta_L)$$
$$Mid_y = \frac{P_R(y)+P_L(y)}{2}$$
$$P_R(y)=\sin(\theta_R)*DPH(\theta_R),\quad P_L(y)=\sin(\theta_L)*DPH(\theta_L) \qquad (3.1)$$

만약, 목표영역이 그림 3.2 에서와 같이 탐색 최대 영역의 크기 θ_{max}보다 큰 경우에는, 오른쪽 영역일 때 DPH(θ_{max}) = DPH(θ_R)를, 왼쪽 영역일 때는 DPH(θ_{max}) = DPH(θ_L)을 할당한다. 이것은 장애물이 없는 영역이 매우 클 때 로봇이 지나치게 많이 진행 방향을 바꾸는 것을 방지해준다.
또한 좀더 안전한 주행을 위해 장애물의 존재 정도를 고려하기 위해 $\delta\theta$를 사용하여 최종적인 경유점을 결정한다.

$$subgoal_X = R\cos(\delta\theta+\theta_{Mid})$$
$$subgoal_Y = R\sin(\delta\theta+\theta_{Mid}) \qquad (3.2)$$
여기서 $\theta_{sub} = \tan^{-1}(Mid_Y/Mid_X)$

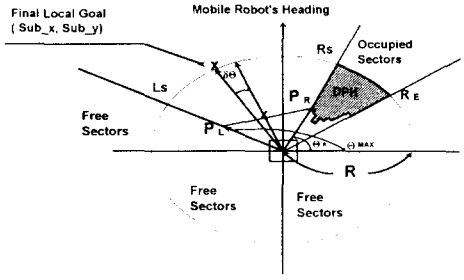

그림 3.2 한쪽영역만 장애물이 점유하고 있는 경우

그러나 위에서 설명한 지역 경로 계획 알고리듬을 반드시 실행해야만 하는 것은 아니다. 목표점으로 곧장 진행하여도 충돌의 위험이 없는 경우에는 지역 경유점을 설정하지 않고 곧장 목표점을 향해 진행한다. 즉 목표점 방향의 좌우로 각각 60° 이상의 점유되지 않은 섹터가 존재하는 경우와 모든 장애물이 목표지점보다 멀리 있는 경우에도 목표점으로 직진하게 된다.

3.2 속도 제어

안전한 주행을 위해 로봇 주변의 장애물이 위치한 상황에 따라서 주행 속도를 변화시켜야 한다. 로봇의 주행속도 변화는 다음과 같다.

$$V = V_{min} + (V_{max}-V_{min})*\frac{K_V}{R} \quad \text{여기서, } R=1.5$$
$$K_V = \frac{\sum_{i=S_R}^{S_L} f(DPH(i))}{S_L - S_R} \quad \text{여기서, } f(j)=\begin{cases} R & j=0 \\ j & j\neq 0 \end{cases} \qquad (3.3)$$

4. 뉴럴네트워크에 의한 학습 주행

4.1 Backpropagation의 구조및 입력 패턴의 양자화

Backpropagation은 많은 뉴럴네트워크 알고리듬중에서 비선형 문제를 해결하는데 있어 우수한 성능을 보여왔며 비선형 특징을 지니며 때로는 휴리스틱(Heuristic)한 방법이 요구되는 이동로보트와 같은 시스템을 제어하는데 많이 사용되어 왔다.[5][6][7] 로보트 주행시 회피하기 어려운 상황에 대해 예외처리를 담당하는 뉴럴네트워크를 학습시키기 위해서는 여러가지 복잡한 환경들의 프로토타입(prototype)을 설정할 필요가 있다. 그림 4-1은 이들 환경들에 대한 프로토타입을 나타내고 있다.

주행환경의 프로토타입은 그림 4-1에 나타낸것과 같이 크게 8가지 상황이 존재한다. 이들 8가지 환경은 뉴럴네트워크로 학습을 시켜 각각의 환경을 인지하도록 해야 한다.

본 논문에서는 주행 환경의 패턴을 인식하고 적절한 행동을 해야하기 때문에 뉴럴네트워크중에서 Backpropagation을 사용한다. 그림 4.2는 본 논문에서 사용한 뉴럴네트워크를 보여 주고 있다. 입력 계층으로 초음파의 거리 정보가 입력이 된다.

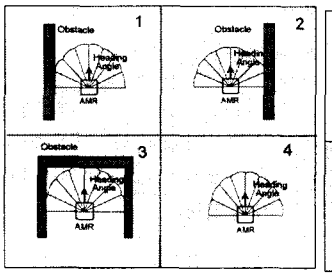

그림 4.1 주행 환경의 프로토타입(a)

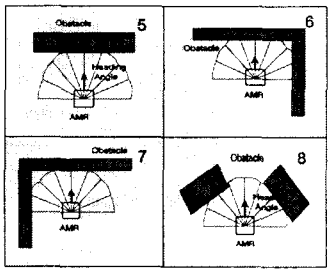

그림 4.1 주행 환경의 프로토타입(b)

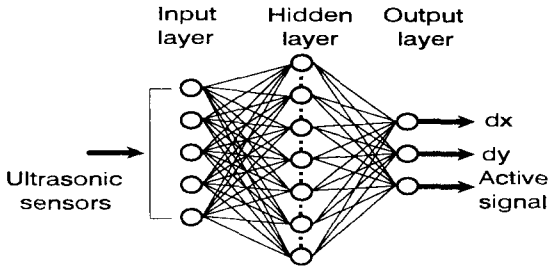

그림 4.2 주행환경 인식을 위한 뉴럴 네트워크

실제 로보트에 사용되는 초음파 센서는 15°간격 으로 총 9개로 구성되어 있고 로보트의 주행 방향을 기준으로 가장 오른편에 있는 센서를 1번으로 하여 시계 방향으로 9번까지 번호를 붙인다. 여기서 입력의 갯수를 줄이고 데이타의 처리를 용이하게 하기위해서 입력거리에 대한 양자화가 필요하다. 이를 위해 중앙에 위치한 5번 센서를 제외하고 오른쪽부터 2개의 센서를 하나의 입력으로 처리하고 입력되는 거리정보는 위험정도에 따라 다음과 같이 양자화 한다.

D1 = 90cm < 감지거리 < 200cm

D2 = 15cm ~ 감지거리 ~ 90cm
D3 = 15cm ~ 감지거리 ~ 45cm
여기서 D1=0, D2=1, D3=2의 값에 해당된다.
이렇게 함으로써 환경 인식 능력은 어느정도 감소는 하지만 너무 많아지는 데이타의 양을 줄일 수 있고 좀더 정확한 출력을 기대할 수 있다. 그림 4.3은 입출력 데이타에 대해 뉴럴네트워크를 통해 학습을 행했을 때의 오차를 보여주고 있다.

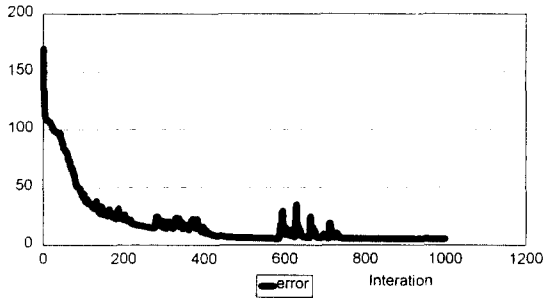

그림 4.3 학습 오차

그림 4.3에 나타나있듯이 학습 오차는 완전히 0으로 수렴하지 않는 이유는 오차의 제곱의 합으로 표시를 하였기 때문이다.

4.2 DPH와 뉴럴네트워크에 의한 주행 제어

장애물의 기하학적 정보를 사용하는 DPH방법은 초음파 센서의 부정확한 특성을 어느 정도 극복이 가능하지만 긴 장애물이나 막힘 상황(Dead end)상황에서는 충돌의 위험성이 증가된다.

그림 4.4는 긴 벽과 같은 장애물이 가로 막고 있는 상황에서 DPH방법으로만으로 주행을 한 결과이다. 그림에 나타나 있는 것과 같이 로보트에 가깝게 위치한 벽은 감지가 되나 어느정도 이상 기울어져 있는 벽은 감지가 되지 않기 때문에 장애물이 없다고 판단하고 주행 하게되므로 장애물에 충돌하는 결과를 보이고 있다.

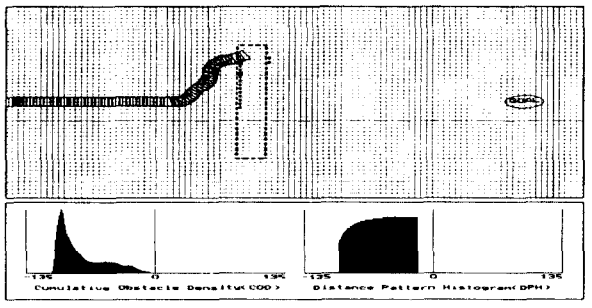

그림 4.4 긴 장애물이 있는 상황에서
DPH방법에 의한 주행 결과

따라서 이러한 상황을 극복할 수 있는 방법으로 본 논문에서는 긴 장애물, 막힘 상황(Dead end), 위험한 상황을 회피하기 위한 방법으로 뉴럴네트워크를 사용하여 이들 상황을 학습 시킨후 상황에 따라 적절한 행동을 할 수 있게 한다.

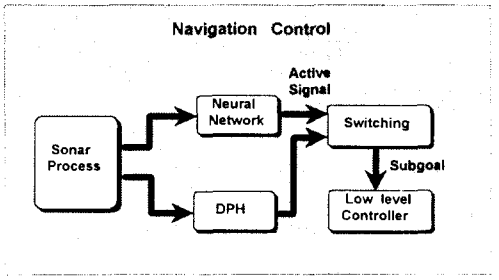

그림 4.5 최상위 제어 시스템

그림 4.5는 DPH와 뉴럴네트워크로 구성된 제어기를 나타내고 있다. 뉴럴네트워크의 출력은 active 신호와 로보트를 중심축으로하고 진행 방향의 x축의 양의 방향으로하는 국부 좌표계에서 로보트의 중심점으로부터 로보트가 지나야할 중간 경유점까지의 x의 변위값과 y의 변위값을 출력으로 한다. 그림 4.5에서 나타낸 것과 같이 센서 정보 처리부로부터 센서의 작동 유무와 거리정보는 DPH 알고리듬을 사용하는 제어기와 뉴럴네트워크를 사용하는 제어기에 병렬로 입력이 된다. 이 때 뉴럴네트워크를 사용하는 제어기는 학습된 정보에 의해 입력되는 센서 정보에 해당하는 환경 패턴을 분류하게 된다. 그리고 일반적으로 긴 벽이나 막힘 상황과 같이 DPH 방법만으로 회피하기 어려운 상황이 되면 학습된 정보에 의해 출력 에서 active 신호는 1이 되고 그렇지 않으면 0이 된다. active 신호가 1이면 뉴럴네트워크의 출력을 active 신호가 0이면 DPH의 출력을 스위칭을 하여 중간 계층의 위치 제어기로 경유점을 보낸다.

4.3 스위칭(Switchig) 제어

본 알고리듬은 DPH방법과 뉴럴네트워크의 학습에 의한 패턴 주행의 적절한 전이를 통해 보다 효율적인 주행을 행하는것에 기반을 두고 있다. 그러나 이 두 알고리듬은 서로 중간 경유점을 발생시키는 방법이 상이하기 때문에 서로 전이되는 순간에 발생되는 중간 경유점에 많은 변화가 일어날 수 있다. 따라서 불필요한 동작 또는 오동작이 발생할 수 있다. 특히 로보트가 목표점으로 향해가고 있는 상황에서는 커다란 문제점이 발생하지는 않지만 막힘상황과 같이 어려운 장애물을 회피하기 위해서는 목표점의 반대 방향으로 가야하는 상황이 생기는 경우와 긴 장애물을 지나고 회전시 센서의 특성상 벽이 감지가 되지않는 문제가 발생한다.

그림 4.6에 나타낸 것과 같이 패턴주행에 의해 목표점과 반대방향으로 로보트가 진행하는 상황에서 막힘상황을 막 벗어나면 알고리듬이 DPH방법으로 전환 된다. 이때 센서에 감지되는 장애물이 없는 경우 중간경유점은 목표점이 되고 로보트는 목표점을 향하여 회전을 하게 된다. 그러나 목표점이 로보트가 지나온 방향으로 향해 있으면 로보트가 지나왔던 곳으로 다시 주행하는 결과가 나올 수 있다.

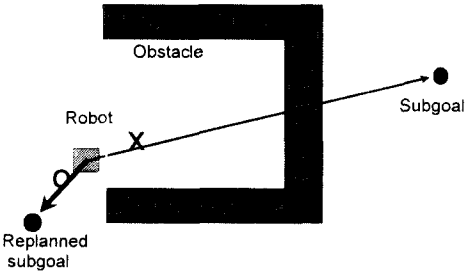

그림 4.6 스위칭 제어

따라서 로보트가 막힘상황을 완전히 빠져 나올 수 있도록 적절한 제어가 필요하다. 이를 위해서 센서 정보에 의해 감지되는 장애물의 기울기를 계산을 하고 이 기울기 값이 연속적으로 감지가 되면 긴 장애물이라고 판단을하고 이를 벗어나면 다음과 같은 방법으로 중간경유점을 발생시켜 문제점을 해결한다.

$$\begin{aligned} next\_x &= \delta x\cos\theta - \delta y\sin\theta + current\_x \\ next\_y &= \delta x\sin\theta + \delta y\cos\theta + current\_y \end{aligned} \quad (4.1)$$

여기서 (next_x, next_y)는 중간 경유점, (current_x, current_y)는 로보트의 현 위치 δx, δy는 로보트를 기준으로 하는 국부 좌표계에서의 중간 경유점으로 각각 2.0, 1.5의 값을 갖는다. 또한 θ는 로보트의 진행각으로서 로보트의 위치가 장애물의 오른편에 있으면 양의 값을 왼편에 있으면 음의 값을 갖는다.

5.1 주행 환경 및 실험

표 5.1은 실험에 사용된 실험 환경을 나타내고 있다.

주행 공간의 크기	10 m × 5 m
격자 크기	0.1 m × 0.1 m
최고 주행 속도	0.7 m/s
최고 주행 가속도	0.25 m/s^2
구동 바퀴사이 거리	0.33 m
Active 원의 반경	1.5 m
Kd	10.0
샘플 주기	0.01 s
COD의 스레스홀드	4
입력 노드 개수	5 개
중간 노드 개수	20 개
출력 노드 개수	5 개

표 5.1 실험 환경

주행 실험은 가로 10m, 세로 5m의 공간내에서 미지의 장애물이 존재하는 상황에서의 주행 결과이다. 그림 5.1과 5.2는 막힘 상황(Dead end)이 존재하는 환경에서의 로보트의 주행 결과이다. 로보트의 초기 상태는 (2, 3, 0°)이고 목표점은 (9, 2.5)이다.

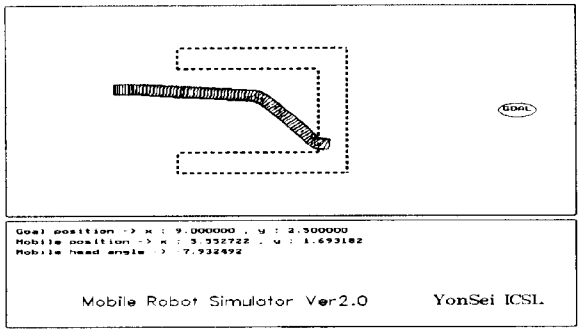

그림 5.1 막힘상황에서 DPH만을 이용한 주행 결과

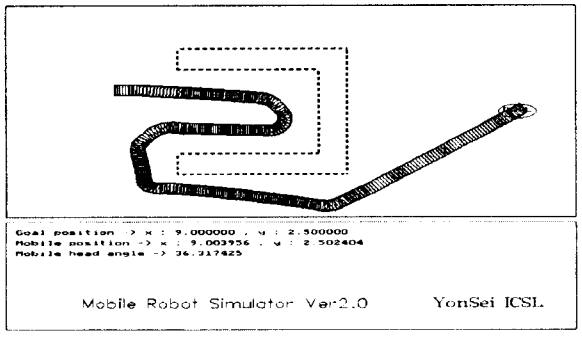

그림 5.2 막힘상황에서 뉴럴네트워크를 이용한 주행

그림5.1은 DPH방법만을 사용하여 주행한 결과이다. 로보트의 진행 방향으로 볼때 주행이 시작된 후 먼저 왼쪽의 벽과 정면의 벽을 감지하게 되어 로보트는 오른쪽으로 회전을 하게 된다. 이때 오른쪽의 벽이 감지가 되지만 초음파 센서의 특성상 정면의 벽과 오른쪽의 벽이 만나는 코너 부분는 감지가 되지 않고 로보트는 장애물이 없다고 판단을 하고 그 방향으로 주행을 하게 된다. 따라서 두 벽이 만나는 구석은 미처 초음파 센서에 감지가 되기 전에 로보트가 장애물과 충돌을 하는 상황이 발생하게 된다. 그림 5.2는 그림 5.1과 똑같은 주행환경에서 뉴럴네트워크를 사용하여 주행한 결과이다. 미리 위와 같은 환경이 학습되어 있기 때문에 안전하고 빠르게 목적지까지 도달함을 알 수 있다. 오른쪽 벽을 통과하면 학습 주행에서 DPH주행으로 스위칭이 되는데 이때 센서에 감지되는 장애물이 없기 때문에 로보트는 목표점을 향하게 되고 급격한 회전이 발생하여 장애물을 빠져나가지 못하게 되는데 이는 4장에서 설명한 스위칭 제어를 통해 극복했음을 알 수 있다. 그림 5.3은 다수의 장애물이 불규칙적으로 존재하는 상황이다.

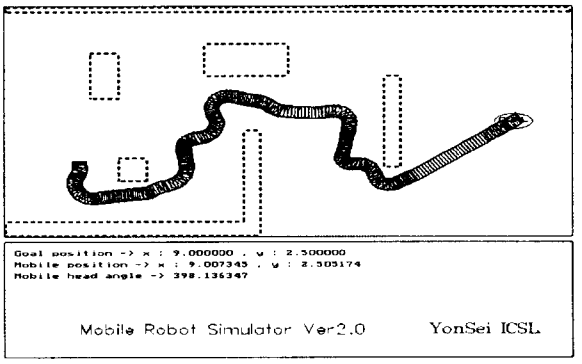

그림 5.3 복잡한 장애물이 있는 상황에서의 주행

제 6 장 결론

본 논문에서는 미지의 환경에서 안전하게 목표점까지 주행할 수 있는 지역 경로 계획으로 DPH방법과 좀 더 복잡한 환경에서는 뉴럴의 학습을 통해 주행하는 방법을 제안하였다. 제안된 방법에 의해 일반적으로 회피하기 어려운 환경에서도 뉴럴네트워크를 이용하여 안전하게 회피하였다. 기존의 방법들은 막힘 상황을 벗어나기 위해서 멈추거나 지역 경로 계획에서 전역 경로 계획으로 전환하여 경로 계획을 다시 해야 하는 번거로움이 있지만 제안한 방법은 수행중인 지역 경로 계획하에서 온라인(on-line)으로 DPH와 뉴럴사이를 스위칭해줌으로써 기존의 방법에 비해 빠른 처리 속도를 보여주고 있다.

참고문헌

[1] Koren, Y., Borenstein, J., "Obstacle Avoidance with Ultrasonic Sensors," IEEE Journal of Robotics and Automation, Vol. 4, NO. 2, pp.213-218 April, 1988.

[2] Koren, Y., Borenstein, J., "Real-Time Obstacle Avoidance for Fast Mobile Robots," IEEE Transaction of Systems, Man, and Cybernetics, Vol. 19, NO. 5 pp.179-1186, 1989.

[3] Koren, Y., Borenstein, J., "The Vector Feild Histogram - Fast Obstacle Avoidance for Mobile Robots," IEEE Trans. Robotics and Automation, Vol. 7, NO. 3, pp.78-288, 1991.

[4] A. Elfes, "A Sonar-based mapping and navigation system," Technical Report, The Robotics Inst., Carnegie-Mellon Univ., pp.25-30, 1985

[5] M.Sekiguchi, S.Nagata, K.Asakawa, "Behavior Control for a Mobile Robot by Dual-Hierarchical Neural Network," IEEE/RSJ International Workshop on Intelligent Robots and Systems, pp.122-127, 1989

[6] P.K.Chande, G.N.Sharma, N.Dagdee, "An Obstacle Avoidance Experiment on a Mobile Robot Using Artificial Neural Network," Proceedings of the 31th SICE Annual Conference International Session, pp.951-954, 1992

[7] Prabir K. Pal, Asim Kar, "Mobile Robot Navigation Using a Neural Net," IEEE International Conference on Robotics and Automation, pp.1503-1508, 1995