Journal of the Korean Institute
of Industrial Engineers
Vol.22, No.3, September, 1996

SOM(Self-Organization Map)을 이용한 로보트 매니퓰레이터 충돌회피 경로계획

Collision-Free Path Planning for Robot Manipulator using SOM

이종우*, 이종태*

RHEE JONG WOO*, RHEE JONG TAE*

Abstract

The basic function of an industrial robot system is to move objects in the workspace fast and accurately. One difficulty in performing this function is that the path of robot should be programmed to avoid the collision with obstacles, that is, tools, or facilities. This path planning requires much off-line programming time.

In this study, a SOM technique to find the collision-free path of robot in real time is developed. That is, the collision-free map is obtained through SOM learning and a collision-free path is found using the map in real time during the robot operation. A learning procedure to obtain the map and an algorithm to find a short path using the map is developed and simulated. Finally, a path smoothing method to stabilize the motion of robot is suggested.

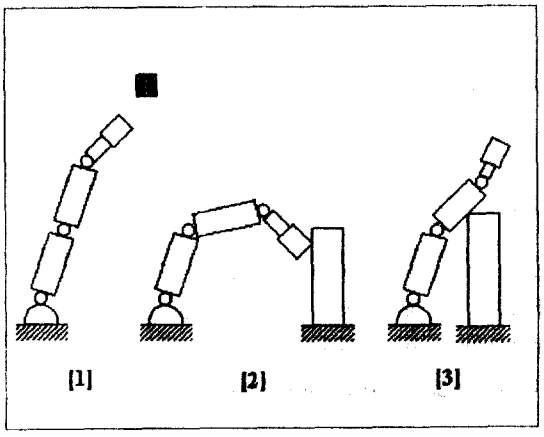
제 1 장 서 론

1-1. 연구 배경

최근 산업현장에서는 생산성 향상과 공장 자동화 추세에 따라 여러 작업공정에 로보트의 이용이 증가하고 있다[3][4]. 로보트 시스템의 중요한 목적 중 하나는 복잡한 작업영역내에 존재하는 물체와 충돌없이 작업물을 신속하고 정확하게 한 장소에서 다른 장소로 옮기는 것이다[8]. 로보트의 작업환경에는 생산에 필요한 각종 공구 및 장치, 설비들이 존재하며 이러한 요소들은 로보트와 충돌을 일으키는 장애물이 될 수 있다. 이러한 로보트의 이용에 있어서의 어려움 중 하나는 작업

* 동국대학교 산업공학과

영역내의 장애물, 즉 각종 공구, 시설 등의 물체와의 충돌을 피할 수 있도록 프로그램 되어야 하며, 이를 위해서 많은 off-line programming time이 소요된다는 것이다. 그림 1은 불가능한 로보트 운동상황을 나타내고 있다[14].



[1] 작업물이 로보트의 작동범위를 초과하는 경우
[2] 로보트가 잡고 있는 물체와 장애물이 충돌하는 경우
[3] 로보트의 링크(link)부분과 장애물이 충돌하는 경우

그림 1. 불가능한 로보트 운동 상황

장애물이 존재하는 곳에서는 로보트와 장애물이 충돌을 일으키지 않도록 하는 경로를 결정할 필요가 있으며, 이를 충돌회피 경로계획이라 한다.

로보트 충돌회피 경로계획은 로보트와 장애물이 서로 충돌을 일으키지 않는 안전한 경로를 찾아야 한다는 것과 그 경로가 가능한 다른 경로보다 짧아야 하며, 로보트 팔 이동에 있어서 무리하게 부하를 주는 심한 변동이 없어야 한다는 세가지 측면이 고려되어야 한다.

장애물 충돌회피 경로계획 문제에 대하여 최근 많은 연구가 수행되어졌는데 이들 연구들은 크게 가설검정방법(hypothesis and test), 벌점부여방법(penalty function), 자유공간명시법(explicit free space) 세가지 방법으로 구분된다[13].

가설검정방법(hypothesize and test)에서는 미리 가정된 경로에 따라 움직이는 물체에 의해 얻어지는 궤적(swept volume)을 계산한다. 즉, 이 궤적과 장애물의 포개짐을 검사하여, 겹치는 부분이 있으면 새로운 경로를 가정하고 다시 시작하여 반복하여 충돌회피경로를 찾는 방법이다. 이 방법은 간단하다는 장점이 있으나, 궤적과 장애물과의 충돌여부를 판단하기가 용이하지 않고, 찾은 경로의 우수성을 보장하기 어렵다는 단점이 있다.

벌점부여방법(penalty function)은 장애물과 로보트 사이의 거리에 따라 벌점을 주는 방법으로 일반적으로 충돌을 일으킬 때의 현상에 대한 벌점은 무한대이며 로보트 팔이 장애물로부터 거리가 멀어질수록 벌점은 급격히 감소한다. 전체 벌점함수는 개개의 장애물로부터 벌점들을 더하고 가장 최단경로로부터 편차에 대한 벌점을 더함으로써 계산된다. 최단경로는 부분적으로 벌점이 작은 것을 조합함으로써 얻어진다. 그러나 이 방법은 다관절 로보트(multi-link robot)같은 경우에 앤드 이펙터(end-effector)가 충돌을 피하더라도 각각의 링크(link)는 장애물을 피하지 못할 수 있다는 단점을 가지고 있으며, 이로 인해서 단지 구형(circular or spherical)의 로보트에서만 적용될 수 있다.

양자유공간방법(explicit free space)은 충돌이 일어나지 않는 자유공간을 파악하고 이를

정확히 표현하여 이 공간내에서 시작점과 목표점을 연결하는 경로를 찾는 방법이다. 이 방법은 주어진 작업환경으로부터 자유공간을 구함으로써 주어진 제약조건을 만족하는 최적경로를 찾을 수 있으며 안전경로의 존재여부도 정확히 알 수 있으나 계산시간이 길어진다는 단점이 있다.

Lozano-perez는 배치공간(configuration space) 개념을 도입하여 충돌회피 경로계획을 이동물체에 대한 장애물의 확장개념으로 파악하여 장애물 회피문제 해결을 시도하였다[18]. 그러나 이 방법은 포인트 로보트(point robot)와 같은 작은 주행로보트나, 직교좌표용 로보트 등에는 쉽게 적용할 수 있으나 관절용 로보트의 경우, 매 순간마다 로보트의 자세가 변하게 되어 이 방법을 적용하기 어렵다. 따라서 배치공간(configuration space)을 이용한 충돌회피 경로계획 방법을 관절형 로보트에 적용하기 위한 방법으로서 작업공간내에 장애물을 관절공간으로 매핑(mapping)하는 방법이 제안되기도 하였다[14][22].

1-2. 연구 목적 및 내용

선행연구들에서의 충돌회피 경로설계에서는 충돌회피 경로를 발견하기 위한 수리적인 알고리즘의 개발과 시뮬레이터 개발이 주를 이루고 있다. 그러나 대부분의 연구들이 다수의 복잡한 행렬등식 계산으로 인해서 실시간 응용이 곤란하다고 할 수 있다[2][9].

본 연구에서는 장애물이 존재하는 작업 공간에서 로보트가 장애물과 충돌없이 움직일 수 있는 경로를 실시간에 구하기 위한 SOM의 응용방안을 제시한다. 즉, 로보트가 장애물과 충돌없이 움직일 수 있는 경로 map을 SOM의 학습을 통해 구축하고 이를 이용하여 실시간에 충돌회피경로를 설정할 수 있도록 하고자 한다. map의 작성을 위해서는 off-line pro-gramming이 필요하나 동일한 작업환경의 경우 1회만 요구되며 이 map을 이용하여 로보트의 경로계획을 실시간에 할 수 있도록 함으로써 다양한 작업을 수행해야 하는 제조환경에서 프로그래밍시간을 최소하고 생산성을 향상시킬 수 있도록 하고자 하는 것이다.

충돌회피경로 map은 로보트 팔의 유효(feasible)한 위치를 관절 좌표공간상의 점으로 표시함과 동시에 이러한 점들의 배열이 로보트 팔의 이동상의 이웃한 연결관계를 나타내는 것으로서 로보트 팔이 장애물과 충돌이 없이 목표지점으로 신속히 움직일 수 있는 경로를 쉽게 발견할 수 있도록 하는 것이다. 본 연구에서는 2관절 로보트를 대상으로 simulation을 통하여 최적학습조건을 갖는 SOM network을 이용, 충돌회피경로 map을 구축하고, 이 map으로부터 실시간에 로보트를 정확하게 이동목표점으로 신속히 이동할 수 있는 충돌회피 경로를 찾는 방법을 개발한다. 또한 얻어진 경로를 point-to-point polynomials 공식으로 보다 유연하게(smoothing) 할 수 있는 방안을 제시한다[5].

제 2 장 SCARA 로보트 매니퓰레이터의 운동학적 모델링

본 연구에서 대상으로 하는 SCARA 로보트는 그림 2와 같이 세개의 평행인 회전관절을 가지고 있어서 평면내에서 방위를 잡을 수 있으며, 네번째의 미끄럼(prismatic) 관절

은 앤드 이펙터(end-effecter)를 평면에 수직하게 움직이게 한다. 그림에서 점선은 로보트 매니퓰레이터가 움직일 수 있는 가능 공간을 표시해 주고 있다.

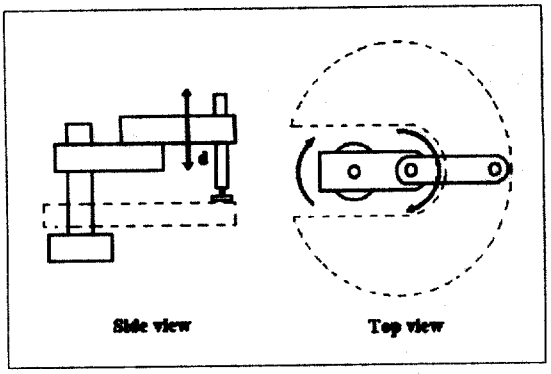


그림 2. SCARA 매니퓰레이터 구조

2-1. 정방향 기구학(forward kinematics)

정방향 기구학은 한 조의 관절각이 주어졌을 때, 기저계에 대한 공구의 위치와 방위를 계산하는 정적 기하학이다. 공구의 위치와 방향은 각 링크계간의 공간적 관계와 관절(joint)의 회전 중심점으로부터의 각도를 이용하여 계산할 수 있다.

2-2. 역방향 기구학(inverse kinematics)

역방향 기구학(inverse kinematics)에서는 공구의 작업공간상의 위치로부터 관련된 관절변수(각)의 해집합을 구한다. 만약 그러한 해가 존재하지 않는다면 로보트 공구의 위치와 방위는 작업공간밖에 있음을 의미하게 된다. 관절변수의 해(solution)를 얻기 위해서는 대수적인 방법(algebraic method)과 기하학적 방법(geometric method)이 사용되는데, 본 연구에서는 기하학적 역방향 기구학(geometric inverse kinematics)을 이용하여 관절각을 구하는 방법을 이용하였다[6]. 다관절 로보트에 있어서 주어진 공구위치에 대응되는 관절각의 해는 여러개가 있을 수 있으며 각각 충돌회피경로 map의 구성을 위해 네트워크의 입력치로 이용된다.

제 3 장 SOM network을 이용한 로보트 경로계획

3-1. 로보트 팔 운영문제에서의 신경회로망 기존연구

Kohonen은 1984년에 SOM Network을 이용하여 로보트 팔의 역방향 기구학 문제(inverse kinematics problem)의 해를 구하는 방법을 제안하였다[12][15][17][19]. 고전적인 로보트 역방향 기구학 문제 해결방법은 등식을 풀기 위한 많은 계산량 때문에 실시간에 해답을 얻기가 어렵지만 신경망은 역방향 기구학의 입력과 출력관계를 학습함으로써 주어진 입력값에 대해 단순한 계산만으로 출력값을 구할 수 있는 장점이 있다[21].

Gues, Ahmad[8]은 역전파학습을 이용하여 정방향 기구학(forward kinematics)을 해결하고자 하였다. Pan, Yang과 Qi[21]은 역방향 기구학 문제 해결을 위해 SOFM(Self-Organization Feature mapping)을 이용하였다.

J. Heikomen, P. Koikkalainen, E. Oja[16]의 연구에서는 모빌 로보트(mobile robot)의 운행중 센서를 이용하여 장애물을 인식하고 충돌이 일어나지 않는 경로를 발견한 다음 SOM 매핑으로 로보트를 control하여 운행하는 방안이 제시되었다.

Jodan[18]은 역전파학습을 이용하여 기구학

문제를 해결하고자 하였다. 이 연구에서는 2차원상의 2관절 로보트의 좌표변형(coordinate transformation) 매핑의 개념으로 해를 구하였으며 관련된 수학적 문제를 해결하는데 신경망을 이용하였으나 현실적인 문제를 해결하기에는 문제에 대한 정의가 다소 부족하였다[9].

3-2. SOM Network의 구성

자기조직화 지도망(self organizing map network)은 비지도학습(unsupervised learning)을 사용하는 신경회로망의 일종으로써, 1984년 코호넨(T. Kohonen)에 의해 제시된 경쟁학습 모델(competitive learning model)이다[1][7][8][22]. 경쟁 학습은 입력된 정보에 대해서 많은 유니트 중에서 승자 유니트를 결정하여 승자 유니트의 연결 가중치만을 입력된 정보에 대응하여 조절하는 학습방법이다. SOM Netwo-rk의 학습방법은 이웃(neighborhood)이라는 개념을 사용하고 있는 점에서 일반경쟁 학습방법과 차이가 있다. 즉, SOM Network에서는 승자 유니트 뿐만아니라 승자 유니트의 이웃 유니트들의 연결 가중치도 함께 조절된다. 즉,승자 유니트 뿐만 아니라 그것의 이웃 유니트들도 해당 입력 패턴에 반응하여 자기조직화가 이루어진다. 유니트들의 이웃 관계는 SOM Network의 output node들의 배열에 의해 결정되므로 SOM Network에서는 입력 패턴들을 output node들에 의한 위상학적인 공간으로 매핑(mapping)할 수 있는 능력을 갖게 된다. 학습이 진행됨에 따라 이웃의 범위는 줄어들게 되고 학습과정이 완료된다.

그림 3은 SOM Network의 구성을 나타내는 그림이다.

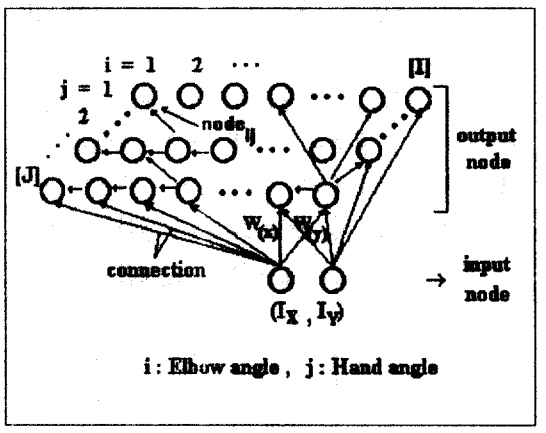


그림 3. SOM Network의 구성

그림 3에서 보는 것과 같이 본 연구에서 사용된 SOM Network는 2개의 input node와 2차원상에 배열된 다수의 output node, 그리고 각 input node와 output node사이의 연결 가중치(weight)로 구성된다. 한쌍의 input node와 각 output node사이의 연결가중치는 로보트 팔이 취할 수 있는 elbow angle과 hand angle을 나타낸다. output node의 배열은 각 output node에 대한 elbow angle과 hand angle들의 이웃 관계를 나타낸다. 각 output node의 연결가중치는 자기조직화 학습과정을 통해 입력 패턴에 적응하여 변하게 된다.

3-4. SOM Network 학습에 의한 map 구축

앞서 설명한 SOM Network에서의 각 input node들과 각 output node 사이의 한쌍의 연결 가중치값은 두개의 관절이 취할 수 있는 각도 벡터(vector)로서 이들 벡터들은 관절좌표계상에서의 로보트의 유효위치를 나타내게

된다. 이러한 유효위치들은 SOM Network의 학습에 의해 로보트 작업영역의 충돌회피공간의 위치점들을 가르키는 map을 구성하게 된다.

SOM의 자기조직화를 위해서는 로보트의 유효위치에 해당하는 'HAND ANGLE'과 'ELBOW ANGLE'이 입력되어야 하며 이들 입력치에 대해 자기조직화를 수행해야 한다.

그림 4에서는 작업현장의 로보트 위치와 관절각의 관계를 보여주고 있다. 작업장내에서 로보트 팔의 끝점이 위치할 수 있는 임의의 점에 대응되는 end point로부터 관절각도를 구하기 위해서 역방향 기구학(inverse kinem-atics)을 이용한다. 관절각 벡터는 2개의 해가 존재할 수 있다.

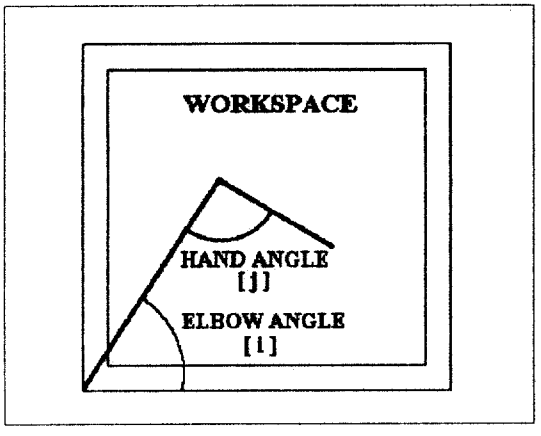


그림 4. ROBOT ARM의 구성

그림 5의 경우 end point에 대응되는 관절각도값은 (α_1, β_1)과 (α_2, β_2)가 된다.

그림 6에서는 관절각이 입력치로서 이용되지 않는 경우를 보여준다.

(α_1, β_1)와 (α_2, β_2)를 각도좌표계에 나타내면 그림 7과 같다.

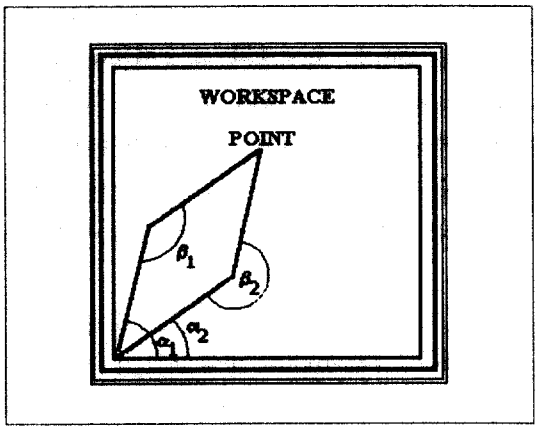
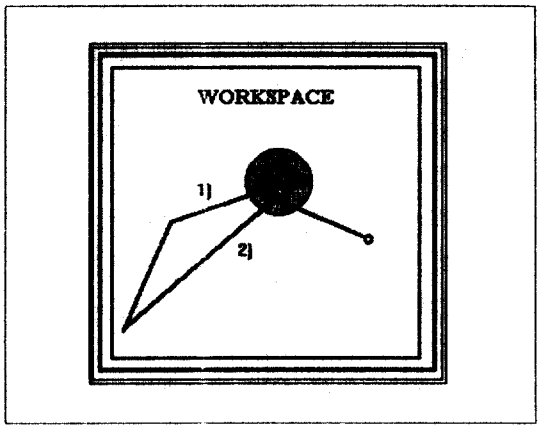


그림 5. end point에 대응되는 관절각



1] 끝점이 장애물에 부딪히는 경우
2] 관절이 장애물에 부딪히는 경우

그림 6. 관절각이 입력치로 사용되지 않는 경우

유효한 관절각도 입력벡터에 대한 자기조직화 과정은 다음과 같다.

(I_X, I_Y)를 elbow angle과 hand angle의 입력값, $W_{X,ij}$, $W_{Y,ij}$를 input node X와 node ij 및 input node Y와 node ij 사이의 가중치값 이라할때승자노드$node_{ij}$는 $|(I_X, I_Y) - (W_{X,ij}, W_{Y,ij})|$가 최소인 node 이다. 이 때 각 output node

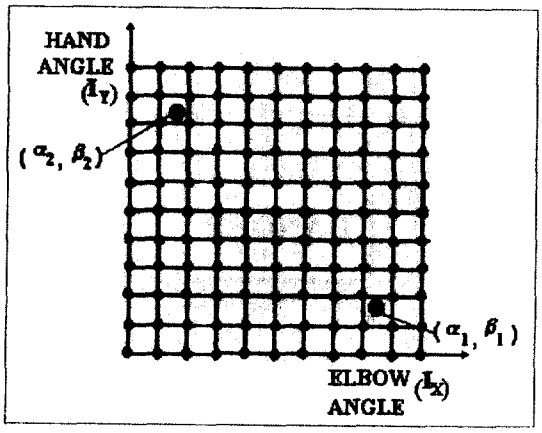


그림 7. 관절각도 좌표계상의 입력 vector

에 연결된 가중치 벡터 ($W_{X,ij}$, $W_{Y,ij}$)는 다음 학습식에 의해 변해 가면서 입력패턴에 대한 자기조직화가 이루어진다.

$$W_{X,ij}(t+1) = W_{X,ij}(t) + \eta \times (I_X - W_{X,ij}(t)) \times e^{-(\Delta ij/\sigma)} \quad (1)$$

$$W_{Y,ij}(t+1) = W_{Y,ij}(t) + \eta \times (I_Y - W_{Y,ij}(t)) \times e^{-(\Delta ij/\sigma)} \quad (2)$$

위 학습식에서 Δij 는 SOM Network의 승자 node 와 node ij 의 거리이며, η 는 학습률, σ 는 output node의 이웃 node 간의 공간적 제약의 강도를 나타내는 파라메타를 나타낸다.

각 식 (1), (2)는 지수함수의 성질상 SOM Network 상에서 승자 node와 가까운 거리에 있는 node에 훨씬 큰 영향을 끼치고 있음을 나타내고 있다. 이는 승자노드와 이웃한 정도에 따라 가중치 벡터(weight vector)의 학습정도에 차이에 있음을 말한다.

[그림 8]은 관절각도 좌표계에서 입력값에 대한 학습과정을 보여주고 있다. 그림에서는 입력된 관절각도 vector와 SOM Network의 output node의 연결 가중치 벡터(weight vector)들의 위치를 관절각도좌표계에서 나타내고 있다.

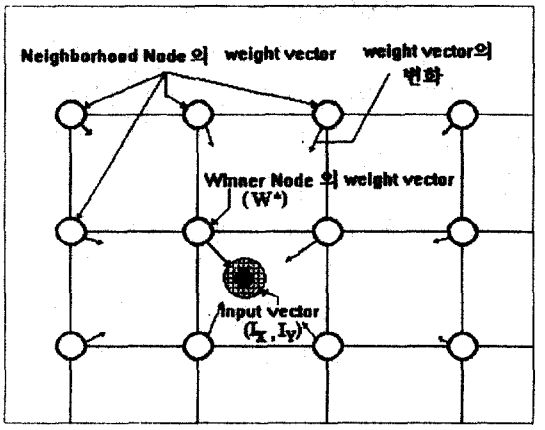


그림 8. 입력값에 대한 학습 과정

위 그림에서는 입력된 input에 대해서 승자 node의 가중치 벡터와 이웃 node의 가중치 벡터가 움직이면서 자기조직화 되고 있는 것을 보여 준다. 모든 output node의 이웃정도는 SOM Network상의 승자 node들의 거리에 의해 결정된다. 식 (1)과 (2)의 σ 는 학습이 진행되면서 점차 감소하게 된다. 즉, σ 가 감소하면서 승자 node의 이웃범위가 감소하는 효과를 갖게 된다.

3-5. SOM Network의 학습단계

본 연구에서는 SOM Network의 학습단계(learning)를 3단계로 나누어 수행하였다. 3개의 학습결과를 이용하여 충돌회피경로를 설정하는 방법은 뒤에서 설명하도록 한다.

① 1차 SOM Network 학습

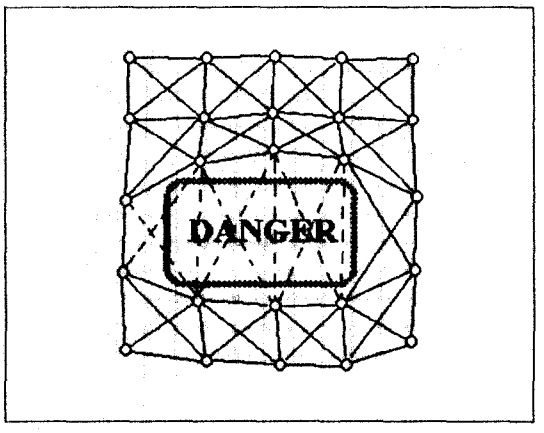
1차 학습단계에서는 로보트의 작업영역에 장애물이 없다고 가정하고 모든 작업영역의 모든 위치를 로보트의 유효한 end point로 간주하고 학습을 통해 자기 조직화한다. 이때 1차 학습결과로 얻어진 map을 1차 map이라 부르기로 한다.

② 2차 SOM Network 학습

2차 학습단계에서는 로보트 작업영역상의 장애물을 고려하여 작업영역중 로보트의 end point가 장애물과 충돌하지 않는 위치점을 대응되는 관절각도로 변환하여 입력 vector로 사용한다. 여기서 end point에 대응되는 관절각도 vector는 자기 조직화를 위한 입력값으로 사용되기 전에 유효성을 검토한다. 즉, 그 관절 각도 vector에 따른 로보트 팔의 link들의 위치가 장애물과 충돌하는 경우인지 조사하여 충돌하는 것으로 판명되는 경우 자기조직화를 위한 입력 vector로 사용하지 않는다. 이렇게 함으로써 자기조직화 결과 얻어지는 map은 로보트의 end point 뿐만 아니라 link들이 장애물과 충돌하지 않는 관절각도 vector의 공간을 구성하게 된다. 즉 학습이 완료되면 output node의 가중치 벡터로 구성된 map은 로보트 작업공간의 유효 위치점들을 로보트의 관절각도 벡터들로 매핑(mapping)한 것이 된다. 본 연구에서는 2차 학습결과 얻어진 map을 2차 map이라 부르기로 한다.

그림 9는 2차 학습결과 얻어진 map의 형태를 나타내고 있다.

그림에서 가중치 벡터가 주변으로 흩어진 지역은 바로 장애물과 충돌을 일으키는 관절각도 벡터의 영역이다. 뒤에서 설명하겠으나 로보트의 운동은 기본적으로 map상의 가중치 벡터 사이에서 관절의 각도를 변화시켜감으로써 이루어지게 된다. 이때 각도의 변화는 가중치 벡터 사이를 linear하게 움직이도록 하는 것이 편리할 것이다. 이 경우 이웃한 가중치 벡터 사이의 거리가 짧을수록 그 구간은 안전구간일 확률이 크다. 완전한 map을 만들기 위해서는 이웃한 가중치 벡터 사이의 직선구간에 충돌가능지역이 있는지를 조사하고 충돌가능구간인지 안전구간인지 표시하는 것이 필요하다.



○ : 가중치 벡터 (관절각)
——— : 가능한 로보트 이동경로
– — – : 이동이 허용되지 않는 구간
DANGER : 충돌가능지역

그림 9. 2차 학습 후의 map

③ 3차 SOM Network 학습

뒷절에서 설명할 로보트 경로결정 과정에서 그림 9에서 점선으로 표시된, 이동이 허용되지 않는 구간사이를 로보트 관절이 이동해야하는 경우가 생길 수 있다. 이 경우 점선구간사이의 안전한 이동경로를 발견하기위한 사전학습이 필요한데 이것이 3차 SOM Network 학습이며 이 결과로 만들어진 map

을 3차 map이라 한다. 본 연구에서는 점선구간에 대해 각각 5개의 가중치 벡터 노드를 추가로 구성하여, 자기조직화 학습을 실시하여 충돌회피경로가 발견할될때까지 가중치 벡터를 학습시키는 방법을 선택하였다. 그림 10은 3차 학습 후의 map을 보여주고 있다. 3차 map을 2차 map상의 충돌가능지역을 제외하고는 같은 모양을 취한다.

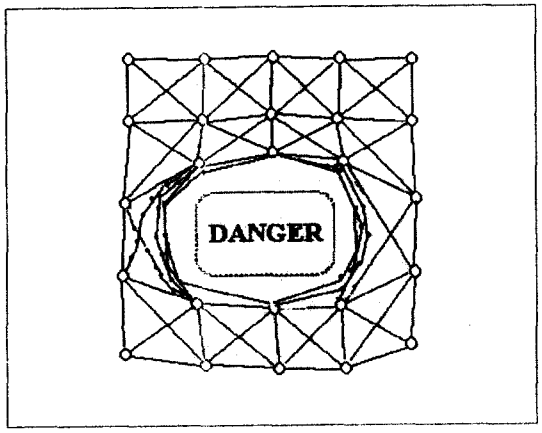


그림 10. 3차 학습 후의 map

3-6. 충돌회피경로 설계 알고리즘

3-6-1. WORMBODY 경로

먼저 본 연구에서 사용하고자 하는 WORMBODY 경로의 개념을 설명한다. 'WORMBODY'란 관절각도 좌표계상에서 로보트가 거치게 될 가중치 벡터위치들을 연결한 것이다 WORMBODY를 결정하기 위해서는 학습과정에서 구한 1차 map, 2차 map, 3차 map을 이용한다. 먼저, 로보트의 현재 위치와 목표점을 기준으로 첫번째 거쳐야할 가중치 벡터 (first weight vector) 위치와 마지막으로 거쳐야 할 벡터(last weight vector)위치를 2차 map상에서 찾는다. first weight vector는 2차 map상에서 표시된 현재 로보트 관절의 위치와 가장 가까운 가중치 벡터의 위치이며 last weight vector는 로보트의 목표위치에서 가장 가까운 가중치 벡터의 위치이다. 여기서 로보트의 e-nd point를 기준으로 한 최종목표점에 해당하는 관절의 위치는 2관절 로보트의 경우 최대 2개가 있을 수 있는데 본 연구에서는 관절각 변동필요량이 작은 관절의 위치를 최종 목표점으로 선택하였다. 다음으로, first weight vector와 last weight vector에 대응되는 가중치 벡터들을 1차 map에서 찾고 이 두개의 가중치 벡터들을 직선으로 연결한 후 이에 잘 fitting되는 일련의 가중치 벡터들을 구한다. 이러한 가중치 벡터들을 선택하는 방법은 몇가지가 있으나 본 연구에서는 first weight vector와 last weight vector를 연결한 직선이 두 개의 이웃하는 가중치 벡터 사이를 가로지를 때 그 두개의 가중치 벡터 중에서 그 직선과 가까운 것을 선택하는 방법을 취하였다. 그림 11은 first weight vector와 last weight vector를 1차 map에서 직선으로 연결하고 선택된 가중치 벡터들을 보여주고 있다. 이와같이 1차 map에서 선택된 가중치 벡터들에 대응하는 가중치 벡터들을 2차 map에서 찾고 이들을 연결하면 기본적인 WORMBODY가 구해지게 된다. 그림 12에서는 이의 개념이 도해적으로 나타나 있다. 이러한 WORMBODY는 로보트가 충돌지역을 가능한 짧고 유연하게 우회할 수 있는 경로가 된다.

3-6-2. 3차 map을 이용한 WORMBODY 완성

위와 같이 구한 WORMBODY는 아직 완전하지 못한다. 그 이유는 WORMBODY상의

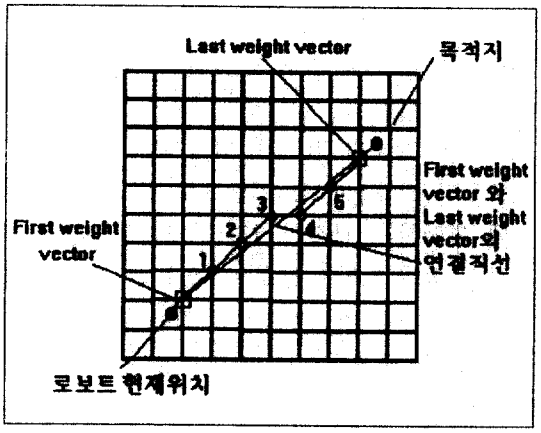


그림 11. 1차 map 상의 기본경로

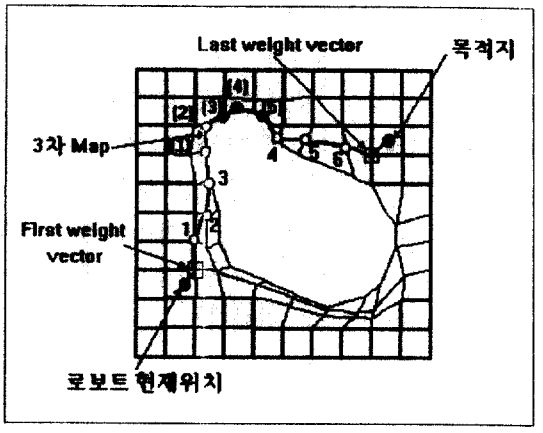


그림 13. WORMBODY 완성

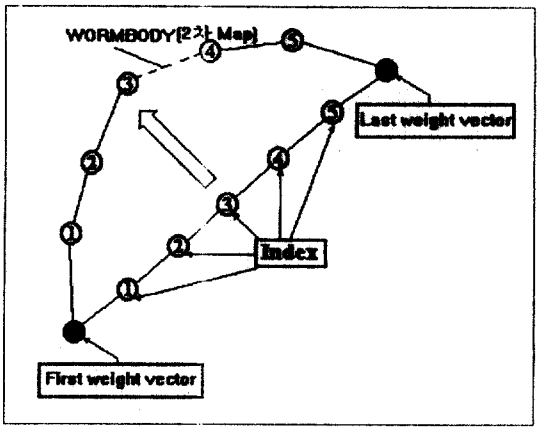


그림 12. WORMBODY 검색시의 개념적 도해
(점선은 2차 map상에서 나타난 충돌구간을 표시함)

가중치 벡터 연결구간에 장애물과의 충돌지역이 포함할 수 있기 때문이다. 즉 그림 9에서 점선으로 표시된 부분이 포함될 수 있는데 이렇게 충돌구간의 로보트 경로를 3차 map에서 구한 경로를 이용하여 보완하게 된다. 그림 13은 이 개념을 나타내고 있다.

3-6-3. point-to-point polynomial을 이용한 유연한 경로 설계

매니퓰레이터의 운동의 유연성은 로보트의 수명과 작업성과의 우수성에 영향을 미친다. 관절각의 변이가 심한 운행은 로보트 관절의 마모를 가속화시켜 작업 정확도를 떨어뜨릴 수 있다. 보다 유연한 경로를 얻기 위해서는 경유점들간의 경로에 대하여, 일종의 시간적 · 공간적 제약을 부과하여야만 한다[8].

본 연구에서는 대부분의 충돌회피 경로문제 연구에서와 같이 시간에 대한 제약은 고려하지 않고 충돌회피 경로에 대한 공간적 제약만을 고려하여 보다 유연한 관절이동경로를 구할 수 있는 방안을 제시한다. 부드러운 경로를 위해서는 curve smoothing이 필요한데 이를 위해 Point-to-point polynomial을 이용할 수 있다[22]. Point-to-point polynomial에는 여러가지 방법에 있는데, 본 연구에서는 cubic p-olynomial을 이용하였다.

비모수 cubic polynomials 방정식은 다음과

같다.

$$y = C_1 + C_2 x + C_3 x^2 + C_4 x^3$$

계수 C_1, C_2, C_3, C_4 는 각 데이타 점들을 통해서 결정되어 진다.

다음 그림 14는 다섯개의 polynomial과 8개의 점들을 이용하여 각 점을 연결하는 직선을 부드러운 곡선으로 만드는 것을 예로 보여 주고 있다.

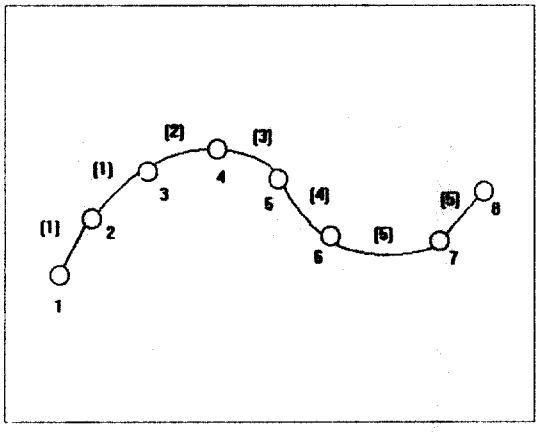


그림 14. cubic polynomials을 사용한 p-t-p smoothing의 예

제 4 장 수행도 평가

4-1. 모의실험

4-1-1. SOM Network을 이용한 충돌회피 매핑

본 절에서는 앞서 설명한 1차 map , 2차 map 학습, 3차 map의 학습과 WORMBODY 개념을 이용한 충돌회피경로 설계, 그리고 point-to-point polynomial 곡선을 이용한 로보트 경로 smoothing의 시뮬레이션 과정을 모의실험하고 수행도를 평가한다. 로보트 형태는 2관절 SCARA형 로보트이며 작업공간내 장애물들은 원통모양의 기둥으로 가정한다. 그리고, 로보트의 두번째 링크(link)와 앤드이펙터(end-effector)는 Hand Arm과 함께 모델링한다. 모든 로보트 운동과정에서의 관절의 변이속도는 일정하다고 가정한다.

그림 15는 모의실험을 위한 컴퓨터의 화면을 보여주고 있는데, 반복횟수표시부분, 로보트팔의 작업공간상 위치, 그리고 관절각좌표계산의 위치를 보여주고 있다. output node의 갯수는 25×25 로 하였다. 로보트 1축은 30° 에서 330° 까지, 2축도 30° 에서 330° 까지로 관절각의 범위를 제한하였다.

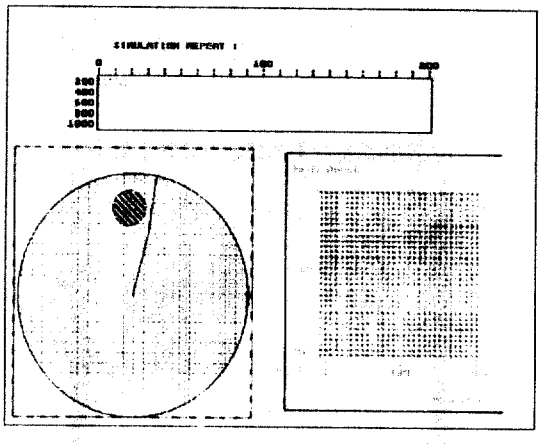


그림 15. simulation 전체 그림

아래의 모의실험과정은 장애물의 갯수가 1개인 경우를 예로 보인 것이다. 그림 16은 학습결과로 관절각 좌표계상에 output node들의 가중치 벡터위치를 나타낸 1차 map의 학습결과를 보여준다.

그림 17부터 그림 22까지는 2차 map 학습과정을 반복회수의 증가에 따라 보여주고 있

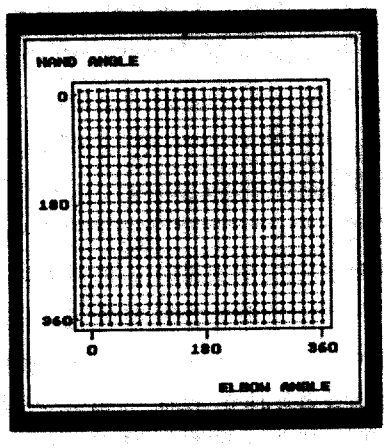


그림 16. 1차 map

다.

그림 23은 3차 SOM Network 학습결과 만들어진 3차 map을 나타낸다. 3차 map은 2차 map의 충돌지역이 보완된 것이다.

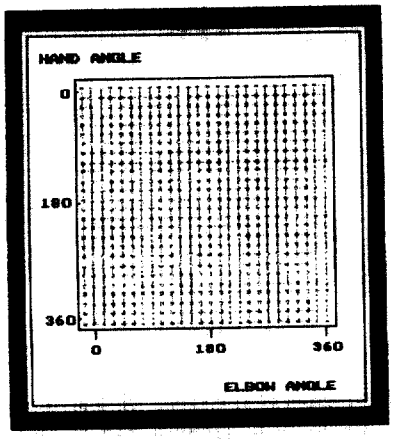


그림 17. 2차 map학습과정
(반복횟수 = 0 회)

그림 24부터 그림 29까지는 주어진 1차, 2차, 3차 map, 그리고 curve smoothing 기법을 이용하여 로보트의 운행경로인 WORMBODY

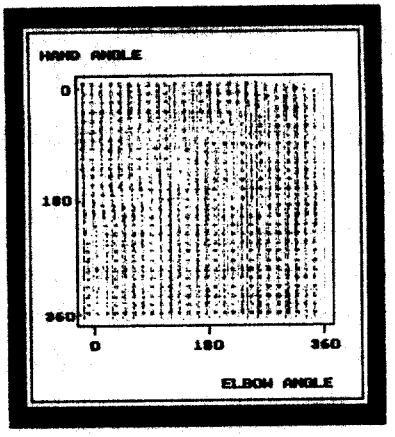


그림 18. 2차 map학습과정
(반복횟수 = 300 회)

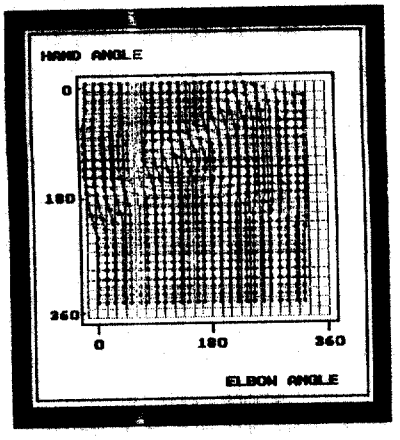


그림 19. 2차 map학습과정
(반복횟수 = 800 회)

를 구하는 과정을 보여주고 있다. 그림에서 굵은 선으로 표시된 WORMBODY는 로보트가 거쳐야 할 관절각 좌표계상의 weight vector를 연결하고, 이를 point-to-point 곡선을 이용하여 smoothing 한 것이다.

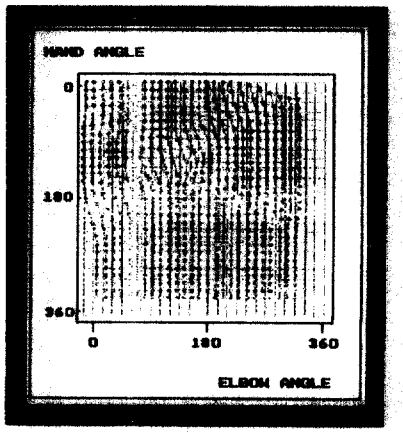


그림 20. 2차 map학습과정
(반복횟수 = 1000 회)

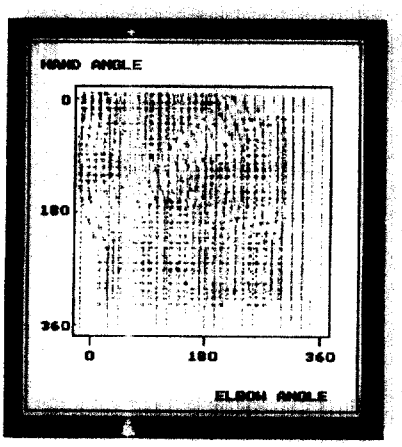


그림 21. 2차 map학습과정
(반복횟수 = 1500 회)

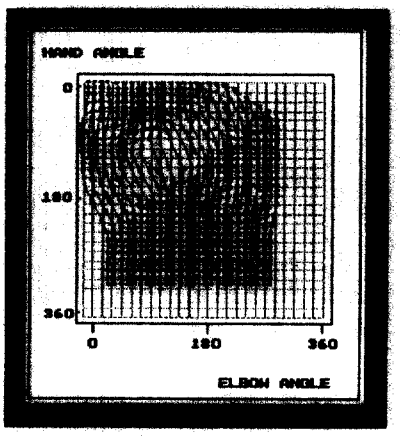


그림 22. 2차 map학습과정
(반복횟수 = 2000 회)

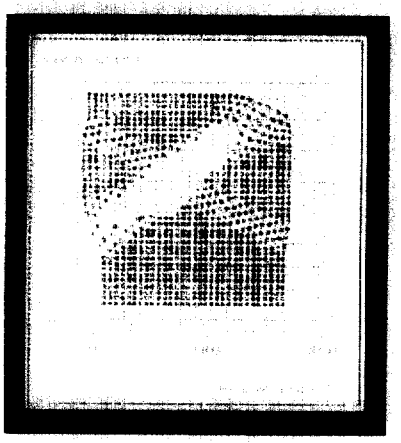


그림 23. 3차 map

4-2. 민감도 분석

4-2-1. 2차원 node 수에 대한 학습속도의 차이

표 1에서는 2차원 node 수에 대한 학습 속도의 차이를 나타내고 있다. 반복수 1000, 2000, 3000회에 대하여 15번씩 모의실험을 한 결과의 평균을 보여주고 있다.

모의실험 결과수치를 통해서 node가 늘어남에 따라서 학습 속도가 점차 증가한다는 것을 알 수 있다. 이러한 학습속도의 증가는 node의 갯수와 비교할 때 크다고 할 수 없으며 신경망 문제해결을 위한 병렬처리 컴퓨터가 사용될 경우 전혀 문제가 되지 않을 것이다.

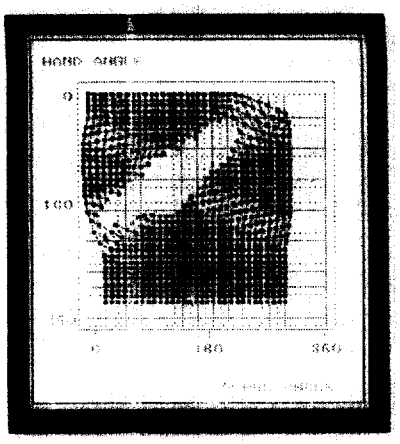


그림 24. WORMBODY를 이용한 충돌회피 경로 (1단계)

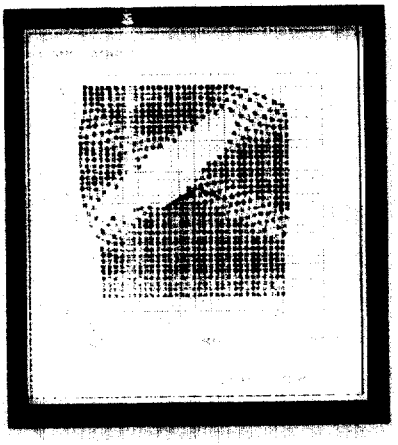


그림 25. WORMBODY를 이용한 충돌회피 경로 (2단계)

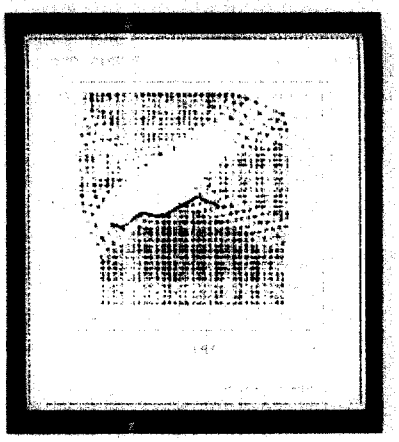


그림 26. WORMBODY를 이용한 충돌회피 경로 (3단계)

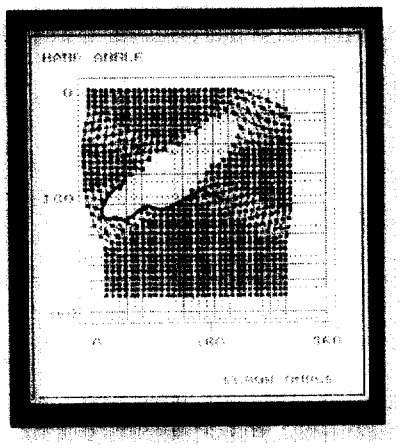


그림 27. WORMBODY를 이용한 충돌회피 경로 (4단계)

4-2-2. 장애물의 증가에 대한 학습의 차이

장애물이 추가되는 경우의 학습속도의 차이는 30×30 output node을 가진 로보트를 기준으로 장애물이 1개, 2개, 3개, 4개, 5개인 경우를 모의실험 하였고, 학습과정상의 반복 횟수는 1000번으로 고정하였다. 표 2는 장애물 추가에 대한 모의실험 결과로서, 15번 반복 실험결과의 평균값으로 학습속도를 표시하였다.

결과에서와 같이 장애물이 추가되는 경우에도 총학습시간은 별 차이를 보이지 않았다 (학습속도의 평균에 대한 분산값이 약 4.43초인 것으로부터 설명됨). 즉, 장애물이 추가되어도 학습시간은 거의 차이가 없으며 결과적으로 off-line programming에 걸리는 시간은 별로 차이가 없다는 것을 알 수 있다.

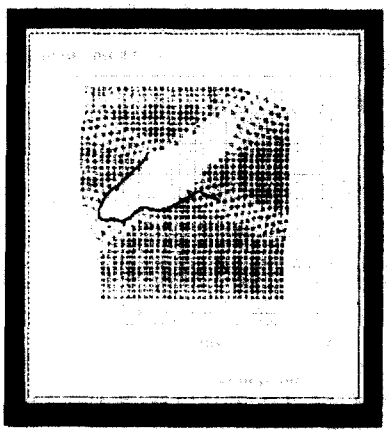


그림 28. WORMBODY를 이용한 충돌회피 경로 (5단계)

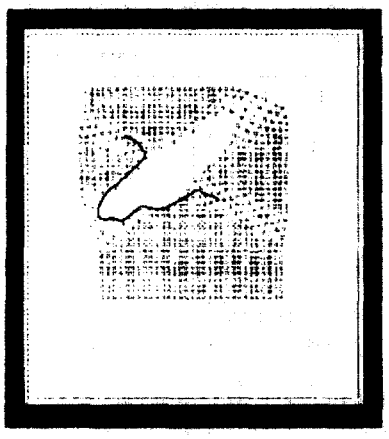


그림 29. WORMBODY를 이용한 충돌회피 경로 (6단계)

표 1. 2차원 node 수에 대한 학습 속도의 차이(pentium-60 PC 사용)

(단위 : 초)

	node	10×10	20×20	30×30	40×40	50×50
반	1000	132.88	246.97	372.19	492.08	643.92
복	2000	298.37	415.59	698.42	983.11	1405.44
수	3000	384.75	612.15	983.66	1475.58	2104.08

학습결과로 얻어진 map을 이용한 충돌회

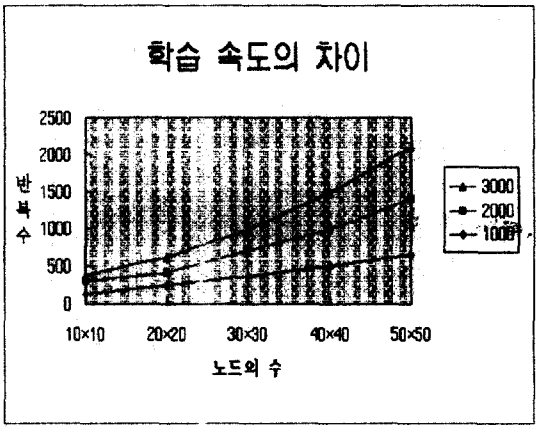


그림 30. 학습속도의 차이

표 2. 장애물이 추가되는 경우 학습속도의 차이(pentium-60 PC 사용)

(단위 : 초)

장애물 수	학 습 시 간
1	378.87
2	370.12
3	380.66
4	377.40
5	381.01
평 균	377.61
표준 편차	4.43

피경로설정 과정에 걸리는 시간은 거의 무시할 정도이기 때문에 민감도 분석을 생략하기로 한다.

제 5 장 결론 및 추후 연구방향

본 연구에서는 2축을 가진 SCARA 로보트의 작업공간에서 로보트 동작을 방해하는 각 장애물의 충돌을 피하면서 원하는 위치로 로보트의 동작을 제어할 수 있는 map을 구성

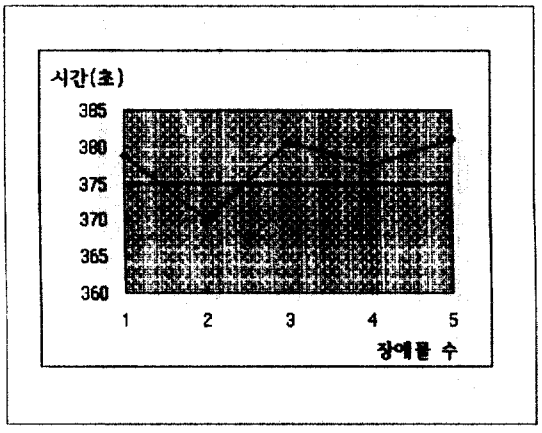


그림 31. 장애물이 추가되는 경우 학습속도의 차이

하기 위한 SOM Network 학습을 이용하였다. 또한 학습된 map을 이용하여 로보트운동경로를 결정할 수있는 방법을 개발하였다. 또한 결정된 로보트 운동경로를 point-to-point 곡선을 이용하여 유연한 경로로 만들 수 있음을 보였다.

학습단계에서 구한 map을 이용하여 이를 통해 실시간에 로보트 경로를 찾을 수 있다. 로보트 작업의 준비시간(setup time)을 단축하여 전체 작업의 lead time을 단축할 수 있을 것이다.

보다 현실적인 로보트 운영상황에서는 작업장의 장애물이 미리 주어진 위치가 아니라 작업진행과정에서 발생하는 경우, 그 장애물에 대한 충돌회피경로를 실시간에 찾아야 하는 문제가 발생한다. 이 경우 SOM Network의 자기조직화를 얼마나 신속하게 할 수 있는가 하는 것이 실제로 충돌회피 경로 발견의 관건이 되어 이는 차후 연구과제가 될 것이다. 또한 3축 이상의 로보트의 경우 충돌회피경로 발견을 위해서는 SOM Network의 output node들의 특별한 배열구조를 필요로 할 것이다. 이를 위한 연구 역시 다음의 과제가 될 것이다.

참 고 문 헌

[1] 김대수, "신경망 이론과 응용(I)," 하이테크정보, 1993

[2] 김주필, "다관절 로보트를 위한 충돌회피 경로 및 제어프로그램 생성," 성균관대학교 산업공학과 석사학위논문, 1995

[3] 김태윤, 김홍복 공저, "메카트로닉스 산업용 로보트," 생능사, 1991

[4] 남궁재찬, "ROBOT 공학의 기술," 기전연구사, 1990

[5] 류성렬 역, "C 언어로 작성한 최신 알고리즘(Computer Algorithms in C)," 다다미디어, 1994

[6] 오석찬, 이종우, 이종태, "공정의 이상유무를 진단하기 위한 최적 SOM(Self-Organi- zation Map) 설계," 추계 대한산업공학회논문집, p374-378, 1994

[7] 이상원, "학습하는 기계 신경망," Ohm 사, 1993

[8] Ahmad, Ziauddin Guez, Allon, , "Solution to The Inverse Kinematics Problem in Robotics by Neural Networks," ICNN, 1988

[9] Aylor, Stephen, Rabelo, Luis Sema-Alptekin, "Artificial Neural Networks For Robotics Coordinate Transformation," Computers ind. Vol.22, No.4, pp. 481-493, 1992

[10] Craig, John J., "Introduction to Robotics and Control, Second Edition," ADDISON-

WESLEY, 1990

[11] Dewey, Bruce R., "Computer Graphics for Engineers," HARPER & ROW, P- UBLISHERS, Inc, 1988

[12] Freeman, James A., Skapura, David M., "Neural Networks, Algorithms, Applications, and Programming Techniques," ADDISON-WESLEY, 1991

[13] Fu, K. S., Gonzalez, R. C., and C.S.G. Lee, "Robotics: Control, Sensing Vision and Intelligence," McGraw-Hill, 1987

[14] Gilles, Ir. W., "Universal Computer Program for seeking a Collision-Free Path for an Industrial Robot," 15th ISIR, 1985

[15] Hecht-Nielsen, Robert "Neuralcomputing," ADDISON-WESLEY, 1990

[16] Heikkonen, J.,Koikkalainen, P. E. Oja, "Self-Organizing Maps for Collision-free Navigation," World Congress on Neural Network,Vol 3, pp. 141-144, 1993

[17] Horne, Bill, Jamshidi, M. & Vadiee, Nader "Neural Networks in Robotics : A Survey," J. Intelligent and Robotic System, Vol. 3, pp. 51-66, 1990

[18] Josin, G. Neural-network heuristics Byte Oct. pp. 183-184, 1987

[19] Kohonen, T. "Self-Organizating and Associative Memory," Springer-Verlag, Berlin, 1984

[20] Lozano-Perez, Tomas "A Simple Motion-Planning Algorithm General Robot Manipulators," IEEE, Journal of Robotics and Automation, Vol. RA-3, No. 3, pp. 224-238, June 1987

[21] Pan, San-Yih, Yang, Jackson C. S., Gui -Zhong Qi, "A Neural Network Methodology to Solve Robotic Manipulator's Inverse Kinematic Problem," World Congress on Neural Network, Vol 3, pp. 194-197, 1993

[22] Red, W. E. Hung-Viot Truong-Cao, "Configuration Maps for Robot Path Planning in Two Dimensions," Transaction of the ASME, Vol. 107, pp. 292-298, 1985

95년 12월 최초 접수, 96년 7월 최종 수정