大韓機械學會論文集 第17卷 第2號, pp. 267~276, 1993.

〈論 文〉

신경최적화 회로를 이용한 로봇의 장애물 회피에 관한 연구

조용재* · 정낙영** · 한창수***

(1992년 6월 9일 접수)

A Study on the Obstacle Avoidance of a Robot Manipulator by Using the Neural Optimization Network

Yong-Jae Cho, Nak-Young Chong and Chang-Soo Han

Key Words: Obstacle Avoidance(장애물 회피), Trajectory Planning(궤적 계획), Neural Optimization Network(신경최적화 회로), Redundant(여유 자유도)

Abstract

This paper discusses the neural network application in the study on the obstacle avoidance of robot manipulator during the trajectory planning. The collision problem of two robot manipulators which are simultaneously moving in the same workspace is investigated. Instead of the traditional modeling method, this paper introduces that of the neural optimization network using the Hopfield network. The parallel processing based on the calculation of joint angle in the cartesian coordinate with constrained condition shows the possibility of real time control. The problem of the falling into the local minima is cleared by the adaptive weight factor control using the temperature adding method. Computer simulations are shown for the verification.

1. 서 론

최근 산업 현장에서는 생산성 향상과 공장 자동화의 추세에 따라, 여러 작업 공정에 로봇의 사용이 점차 늘어나고 있다. 이에 따라서 로봇의 응용 기술 또한 다양하여져서 간단한 조립, 운반, 우주에서의 활용, 심지어는 의료에 이르기까지 여러 방면으로 많은 연구가 이루어지고 있다. 로봇을 이용하여 어떤 작업을 수행할 때 가장 중요하게 다루어야 할 문제중의 하나는 그 로봇의 공간상의 경로를 계획하는 것과 로봇의 첨단부가 이 경로를 따라 이동할 때 각 관절들이 움직여 가야할 공간상의 궤적을 계획하는 문제가 될 것이다.

*정회원, 한라중공업 우주항공/자동화부
**정회원, 한양대학교 기계설계과 대학원
***정회원, 한양대학교 기계공학과

로봇의 운동 경로 계획과 궤적 계획은 조인트 운동의 기구학적 제한 조건 등으로 인하여 쉽지 않으며 이와 더불어 작업 공간상의 장애물을 동시에 고려하고 여기에 여유 자유도(redundant)를 가질 경우까지 고려하면 더욱 복잡한 문제가 된다.[1][2][3] 이에 대한 연구로서, Yoshikawa[4]는 충돌 회피를 안전한 방향(충돌에서 벗어나는 방향)으로 향하는 속도 방향으로 벡터 Z를 정의하여 수행하였으나, 작업 환경에 대한 선험적인 지식이 요구되므로, 장애물이 움직이는 경우에 대해서는 해결해 줄 수 없었다.

Maciejewski[5]는 매 시간마다 장애물에서의 최소 근접거리에 있는 매니퓰레이터의 지점(point)을 확인하고, 그 지점에 장애물 표면으로부터 멀어지는 방향으로 속도 명령(velocity command)을 주어 이동 장애물에 대한 충돌 회피까지도 수행하였으나, 자코비안 행렬(jacobian matrix)을 이용한

해석 과정에서 가상 역행렬(pseudo inverse) 계산이 요구되어 근사해(approximation solution)로 구해지게 된다. 이러한 문제의 파악 및 서술이 복잡한 경우나 변수의 수가 아주 많은 경우 등에 있어서 생체 신경을 모델링한 신경회로망(neural network)을 이용하게 되면, 정확성면에서, 그리고 장애물 변화와 같은 외란에 있어서 큰 장점을 가지게 된다. 이에 대한 연구로서, Graf와 Lalonde[6]는 장애물 회피에 관한 문제를 기존의 자코비안을 이용한 방법이 아닌, 생체 신경을 모델링한 신경회로망중 코호넨 회로(kohonen network)를 이용하여 다루어 주었다. 이 회로의 특징은 자율학습(unsupervised learning)으로 매니퓰레이터에 대한 기구학적 및 작업 환경에 대한 구속 조건을 자동적으로 학습할 수 있는 능력을 가지고 있고, 연결 노드의 수를 증가시켜 줄수록 매니퓰레이터의 복잡성과 정확성이 증가되지만, 상대적으로 학습 시간의 증가를 가져오고, 한 지점을 찾아가는 경로 계획(path planning) 문제에 국한되어, 궤적 계획(trajectory planning)까지는 미치지 못하고 있다. 이 외에도 신경회로망을 이용한 매니퓰레이터의 제어에 관한 연구는 Grossberg,[7] Grossberg와 Kuperstein[8], Kuperstein[9] 등 많이 이루어지고 있는데, 본 연구에서 다루고자 하는 모델은 Hopfield Network을 바탕으로 한 신경최적화 회로로서, A/D 변환기, 신호 판단회로, 선형 프로그램밍 회로 등에 구현된 바 있다.[12]

본 연구에서는 우선 네트워크의 기본 이론을 소개하고, 이를 바탕으로 로봇의 궤적계획에 이용하기 위한 변수, 구속조건, 에너지를 정의하며, 네트워크가 국부적 최소 상태에 빠지는 것을 방지하기 위하여 Temperature Adding을 이용한 적절한 가중치 조절로 작업 공간상에 장애물이 정지하고 있는 경우와 장애물이 움직이는 경우, 같은 작업 공간상에서 동시에 움직이는 두 대의 로봇들의 충돌회피문제 등 여러 종류의 궤적 계획 문제에 적용시켜 보고자 한다. 이와같이 궤적계획 문제를 신경회로를 이용하여 다루게 되면, 신경회로망의 장점으로서 링크수나 장애물의 변화등 상황의 변화에 따르는 복잡한 모델링 전개가 필요 없어지고, 여유자유도를 가지는 경우에도 별도의 성능지수를 위한 인위적인 조작을 요구하는 알고리즘의 개발이 필요없이 스스로가 최소의 에너지 상태를 찾아가게 되며, 병렬식(parallel processing) 계산방식으로 각 링크의 위치를 동시에 구할 수 있게 되어 실시간 제어의 가능성을 제시하여 준다.

2. 이 론

2.1 Hopfield Network 및 신경최적화 회로

Hopfield Network은 1980년 초에 John Hopfield에 의하여 고안된 회로로, 그 기본 구조는 Fig. 1과 같이 다른 회로와는 달리 출력값이 다시 입력으로 들어가 출력값이 변하지 않을때까지 이 반복을 계속 수행하는 것이다. 즉 다른 회로들은 단일 방향으로 작동하는 정적인 것에 비하여 이 회로는 시간에 따라서 내부 상태가 동적으로 변화하는 것이다. Hopfield Network의 특성중의 하나는 작동과정이 회로의 에너지 함수로 설명되어질 수 있다는 것이다. 이것은 그림상으로 보면 쉽게 이해할 수 있는데, Fig. 2는 3개의 패턴을 나타내는 Hopfield Network의 에너지 함수를 보여주고 있다. 여기에 불완전한 입력이 들어올 경우는 이 에너지 곡선상의 임의의 지점에 공이 하나 주어진 것으로 생각할 수 있는데, 공을 놓으면 이 공은 골짜기를 타고 내려가서 더 이상 내려갈 수 없는 지점

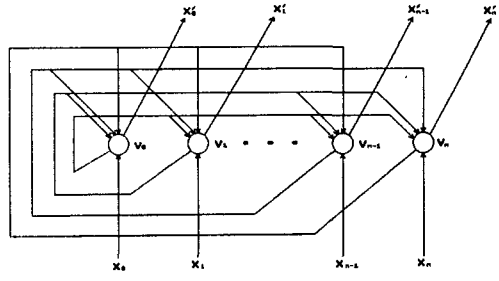

Fig. 1 Basic structure of Hopfield network

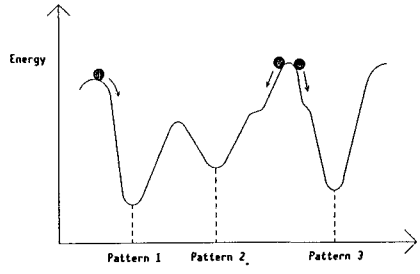

Fig. 2 Energy curve of Hopfield network with three pattern

에 도달하게 되어 이 지점이 바로 기억된 완전한 패턴임을 뜻하는 것이다. 이와 같이 Hopfield Network이 에너지 곡선상의 임의의 지점에서 시작하여 상태를 계속 바꾸어 나가면서 최소점에 도달하는 특성은 일반적인 최적화 문제(optimization problem)를 해결하는 것과 매우 유사한 것이다. 즉, 어떠한 최적화 문제를 에너지 곡선으로 나타낼 수 만 있다면, 일반적인 최적화 문제도 이 회로를 이용하여 해결해 줄 수 있어, 원래의 목적인 연상기억과 더불어 최적화 문제라는 또 하나의 응용분야에 대한 가능성을 제시하여 주고 있는 것이다. 그러면 일반적인 최적화 문제를 어떻게 회로로 구성하는지를 살펴보겠다.

일반적으로 선형 최적화 문제는

$$\text{목적함수} ; \Phi = \underline{A} \cdot \underline{V} \tag{1}$$
$$\text{구속조건} ; \underline{D_j} \cdot \underline{V} \geq B_j (j=1,2,\cdots,M)$$

과 같이 표현된다.

여기서, $\underline{A}$: N-dimensional Vector of Coefficint

$\underline{V}$: N-dimensional Vector of Design Variable

$\underline{D_j}$: N-dimensional Vector of Coefficient

B_j : Boundary Condition $(j=1, 2, \cdots, M)$

이렇게 구성된 최적화 문제를 회로로 구성하면 Fig. 3과 같이 된다.

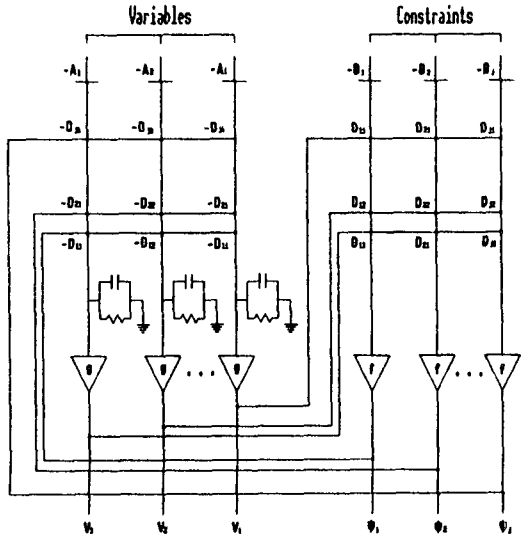

Fig. 3 Neural optimization network

Fig. 3을 보면 왼쪽편에 설계변수(design variable)가 되는 N개의 출력 V_i와 오른쪽편에 구속조건을 나타내주는 M개의 출력 Ψ_j로 나타나 있다. 오른편의 j번째 출력단의 Ψ_j는 i번째 변수에 대한 구속조건 계수(constraint coefficient) $\langle -D_{ji} \rangle$가 곱하여져서 왼쪽편의 입력단으로 들어간다. 그리고 각각의 V_i에는 입력 Capacity C_i와 입력 Resistor R_i가 병렬로 연결되어 있다. 또한 M개의 Ψ_j는 왼쪽편에서 넘어오는 N개의 변수 V_i에 D_{ji}가 곱하여진 것과 경계치가 되는 상수 B_j가 합해져서 나온다.

V_i는 g라는 임의의 선형 단조 증가함수로 구성된 증폭기의 출력단이고, Ψ_j는 식(2)와 같이 구성된 증폭기 f의 출력단이다.

$$\Psi_j = f(U_i) \tag{2}$$
여기서, $U_j = \underline{D_j} \cdot \underline{V} - \underline{B_j}$

$$f(Z) = \begin{cases} 0, & Z \geq 0 \\ -Z, & Z < 0 \end{cases}$$

그러면, 이 회로에서 시간 t에 대한 V_i의 회로 방정식을 구하여 보면

$$C_i \frac{dU_i}{dt} = -A_i - \frac{U_i}{R_i} - \sum_j D_{ji} f(U_j)$$
$$= -A_i - \frac{U_i}{R_i} - \sum_j D_{ji} f(\underline{D_j} \cdot \underline{V} - B_j) \tag{3}$$
$$(i=1, 2, \cdots, N,\ j=1, 2, \cdots, M)$$

와 같이 표시할 수 있다.[12]

그리고, 이 회로의 에너지는

$$E = (A \cdot V) + \sum_j F(\underline{D_j} \cdot \underline{V} - B_j)$$
$$+ \sum_i \frac{1}{R} \int_0^{V_i} g^{-1}(V)\, dV \tag{4}$$
여기서, $f(Z) = \frac{dF(Z)}{dZ}$

와 같이 나타난다.

에너지 E를 시간에 대하여 미분하여 주면

$$\frac{dE}{dt} = \sum_i A_i \frac{dV_i}{dt} + \sum_j \sum_i D_{ji} f(\underline{D_j} \cdot \underline{V} - B_j) \frac{dV_i}{dt}$$
$$+ \sum_i \frac{U_i}{R_i} \frac{dV_i}{dt} = \sum_i \frac{dV_i}{dt} \left[A_i + \sum_j D_{ji} \right.$$
$$\left. f(\underline{D_j} \cdot \underline{V} - B_j) + \frac{U_i}{R_i} \right] \tag{5}$$

와 같이 된다.

식(5)에 식(3)을 대입하여 정리하면 다음과 같이 된다.

$$\frac{dE}{dt}=-\sum_i C_i\frac{dV_i}{dt}\frac{dU_i}{dt}$$

$$\left[\frac{dU_i}{dt}=U_i\frac{dV_i}{dt}=g^{-1}(V_i)\frac{dV_i}{dt}\right]$$

$$=-\sum_i C_i\ g^{-1}(V_i)\left(\frac{dV_i}{dt}\right)^2 \tag{6}$$

여기서 C_i는 양의 값을 갖고, $g^{-1}(V_i)$는 단조 증가함수, 그러므로

$$\frac{dE}{dt}\leq 0 \tag{7}$$

가 되어 에너지 E는 단조 감소함수가 되어, Fig. 3의 회로는 그 시스템의 시간이 경과함에 따라서 상태공간(state space)상에서의 에너지의 최소값을 찾아가게 되는 것이다. 다음 절에서는 이러한 회로를 로봇의 궤적계획 문제에 이용하기 위하여 기본이 되는 모델링 전개에 대하여 살펴보겠다.

2.2 궤적 계획을 위한 기본 모델링

본 연구에서 다루게 될 관절각의 궤적계획 문제를 간략하게 설명하면, 로봇 매니퓰레이터의 첨단부가 장애물이 있는 작업 환경하에서 다음과 같은 구속 조건들 1) 링크 길이를 유지하며, 2) 기준점(base)과 각 링크들 그리고 첨단부(end-effector) 간의 안전거리 및 상대적 조건등을 유지하고, 3) 장애물 회피를 만족시켜 주면서 목표점(target point)을 찾아가는 것으로 정의한다.

이 세가지 구속조건은 다음과 같이 수식화될 수 있다.

$$SV_i^i=SC_i^i\ \ (i=1, 2, \cdots, N) \tag{8}$$

$$SV_i^j\geq SC_i^j\ \ (j=1, 2, \cdots, N-1,\ i=1, 2, \cdots, N,\ i>j) \tag{9}$$

$$PV_i\geq PC_i\ \ (i=1, 2, \cdots, N-1) \tag{10}$$

$$HV_i^j\geq HC_i^j\ \ (i=1, 2, \cdots, L,\ i=1, 2, \cdots, N) \tag{11}$$

여기에서 SC, PC, HC는 원하는 링크 길이와 안전영역(keep-off region)을 나타내는 상수이고, SV, PV, HV는 K-dimensional의 이상적인 매니퓰레이터상에서 다음과 같이 구성된다.

$$SV_i^1=\left(\sum_{k=1}^{k}(L_{i,\,k}-B_k)^2\right)^{1/2} \tag{12}$$

$$(i=1, 2, \cdots, N,\ k=1, 2, \cdots, K)$$

$$SV_i^j=\left(\sum_{k=1}^{k}(L_{i,\,k}-L_{j,k})^2\right)^{1/2} \tag{13}$$

$$(j=1, 2, \cdots, M,\ i=2, 3, \cdots, N,\ N\geq M,\ k=1, 2, \cdots, K)$$

$$PV_i=\left(\sum_{k=1}^{k}(L_{i,\,k}-T_k)^2\right)^{1/2} \tag{14}$$

$$(i=1, 2, \cdots, N,\ k=1, 2, \cdots, K)$$

$$HV_i^j=\left(\sum_{k=1}^{k}(L_{i,\,k}-O_{j,k})^2\right)^{1/2} \tag{15}$$

$$(j=2, 3, \cdots, L,\ i=2, 3, \cdots, N,\ k=1, 2, \cdots, K)$$

여기서

N : Number of Link

M : Dummy Number

k : Number of Dimension

B_k : Base Position $(k=1, 2, \cdots, K)$

$L_{i,k}$: Joint Position of Movable Link $(i=1, 2, \cdots, N,\ k=1, 2, \cdots, K)$

$L_{N,K}$: End Effector $(k=1, 2, \cdots, K)$

T_k : Target Position $(K=1, 2, \cdots, K)$

$O_{i,k}$: Obstacle Position $(i=1, 2, \cdots, L,\ k=1, 2, \cdots, N)$

이러한 구속조건을 만족시켜 주면서 $PV_N=0$이 될 때의 $L_{i,k}$를 구하게 되면 그 매니퓰레이터의 움직임에 대응하는 해를 구할 수 있다.

2.3 네트워크 구현

앞 절에서 모델링된 문제를 2.1절의 회로로 구현시키기 위하여 식(8)-(11)의 구속조건을 이용하여 다음과 같은 에너지를 정의하여 준다.

$$E(S)_i^i=\frac{1}{2}(SV_i^i-SC_i^i)^2\ \ (i=1, 2, \cdots, N)$$

$$E(S)_i^j=\frac{1}{2}F(SV_i^j-SC_i^j)\ \ (j=1, 2, \cdots, N-1,\ i=1, 2, \cdots, N,\ i>j)$$

$$E(P)_i=\frac{1}{2}F(PV_i-PC_i)\ \ (i=1, 2, \cdots, N-1)$$

$$E(P)_N=\frac{1}{2}PV_N^2$$

$$E(O)_i^j=\frac{1}{2}F(HV_i^j-HC_i^j)\ \ (j=1, 2, \cdots, L,\ i=1, 2, \cdots, N)$$

$$E(G)_{i,k}=\frac{1}{R_{i,k}}\int_0^{L_{i,k}}g^{-1}(L)\,dL\ \ (i=1, 2, \cdots, N,\ k=1, 2, \cdots, K) \tag{16}$$

여기서, $g(Z)=Z$

$$F(Z)=\begin{cases}0, & Z\geq 0\\ Z^2, & Z<0\end{cases}$$

여기서 함수 $F(Z)$가 의미하는 것은 Z가 양의값을 갖을 때는, 주어진 구속 조건을 만족시키고 있어

출력을 0으로 내보내고, 음의 값을 갖을 때는 Z의 제곱을 취해주어 양의 값을 내보내 전체 에너지에서 구속조건 위반 인자로 들어가게 되는 것이다. 식(16)과 같이 구성된 에너지들을 하나로 묶어 전체에너지의 시간 미분값을 구한것이 부록에 나와있다. 이 미분 방정식은 앞의 식(7)의 $dE/dt \leq 0$을 만족시켜 주어 결국 이 에너지 함수는 $L_{i,k}$라는 상태변수를 갖는 하나의 단조감소 함수가 되어, 여기서 목적 함수의 역할을 해주는 $E(P)_N$이 나머지 구속 조건들을 만족시켜가며 0이 될때 매니퓰레이터의 첨단부가 목표점을 찾아가는 것이다. 이에 대한 해석 방법으로는 Fig. 3의 회로 소자들을 직접 구성하는 하드웨어적인 해석법과 부록에 유도된 미분 방정식을 프로그램적으로 구성하는 소프트웨어적인 방법이 있는데, 본 연구에서는 소프트웨어적인 방법을 사용하였고, 이 때의 미분방정식의 Solver로는 IMSL의 Runge-kutta Verner 5-6 order를 이용하였다.

2.4 Temperature Adding을 이용한 가중치 조절

앞 절에서와 같이 신경 최적화 회로를 로봇의 궤적 계획에 적용시킬 때 이 회로가 국부적 최소상태에 빠져서 목표점을 제대로 찾아가지 못하는 경우가 발생하였는데, 이는 장애물이 있을 경우 구속조건들간의 상호 연관 관계에 의하여 생기는 것으로, 초기위치(initial configuration)에 따라 장애물 근처에서 진동이 발생하거나 또는 교착상태(deadlock State)에 빠지는 경우이다. 이것은 Hopfield Network을 최적화 문제에 이용할 때 전역적 최소값(global minimum)에 대한 보장이 없기 때문이다. 이에 대한 해결책으로 제안된 것이 Temperature Adding으로서, 구속 조건의 범위에 해당하는 안전 영역(keep-off region), 즉 장애물의 크기를 가상적으로 늘려줌으로써 전체 에너지를 증가시켜 국부적 최소상태에서 빠져나올 수 있도록 하였다. 본 연구에서는 이러한 원리를 이용한 적절한 가중치(weight factor) 조절을 위하여 두 가지 알고리즘을 제안하였는데, 알고리즘 I은 가중치 조절을 위한 기본 알고리즘이 되는 것으로, 그 조절방법은 전체 가중치의 합을 1로 했을 때 각 가중치들의 상대값들을 부록에 유도된 각 에너지 항들에 곱해주는 것이다. 알고리즘 II는 국부 최소값 방지를 위해 Temperature Adding의 원리를 이용한 것으로, 알고리즘 I의 2단계와 3단계 사이에 알고리즘 II를 추가시켜 주는 것이다. 여기서 장애물과의 안전 영역 G와 Q의 증가로 구속조건 위반인자(Constraint Violation)가 늘어나 전체 에너지가 증가되어 국부적 최소상태에서 벗어날 수 있게 된다. α와 β값은 안전 영역(keep-off region) G와 Q를 증가시키는 계수의 역할을 하게 되는데, 그 수치의 결정은 국부 최소값을 벗어날 수 있는 임의의 임계값 이상이면 만족하지만 너무 커지면 매니퓰레이터의 관절각(joint angle)의 변화가 심해지므로 적절한 값의 결정을 위해 기존의 최적화 기법인 Golden-section method의 Line search의 원리를 이용하여 최소의 임계값을 갖도록 결정하여 주었다.[18]

ALGORITHM I

1단계 : 각 가중치와 에러 범위를 초기화한다.

2단계 : 해석.

3단계 : 각 조인트에 대한 장애물 확인, 침범하면 $WF(2, L)$의 비율을 증가시켜 2단계로 보낸다.

4단계 : 각 링크에 대한 장애물 확인, 침범하면 $WF(3, L)$의 비율을 증가시켜 3단계로 보낸다.

5단계 : 각 링크 길이 확인, 변하면 $WF(1, L)$의 비율을 증가시켜 2단계로 보낸다.

6단계 : 목표점에 대한 에러 범위 확인, 크면 $WF(4, 1)$ 비율을 증가시켜 2단계로 보내고, 작으면 다음 목표점 지정하고 1단계로 보낸다.

ALGORITHM II

1단계 : 전체 에너지를 확인, 에너지가 0이 아닌 값으로 변화가 없거나 진동할 경우는 2단계로 보내고, 그렇지 않으면 알고리즘 II를 빠져나온다.

2단계 : 각 조인트에 대한 장애물 확인, 침범하면 $Q=\alpha \cdot R$로 놓고, $WWF(4,2)$와 $WWF(L, K)$를 증가시켜 Golden-section subroutine을 거쳐 알고리즘 I의 해석 부분으로 보내다.

3단계 : 각 링크에 대한 장애물 확인, 침범하면 $G=\beta \cdot R$로 놓고, $WWF(4,2)$와 $WFF(L, K, J)$를 증가시켜 Golden-section subroutine을 거쳐 알고리즘 I의 해석 부분으로 보낸다.

4단계 : 2단계나 3단계에서 해당되는 $WWF(L, K)$, $WFF(L, K, J)$를 알고리즘 I에서 다음 목표점으로 넘어갈 때까지 고정시킨다.

여기서,

$WF(1, L)$: Weight Factor for Link Length
$WF(2, L)$: Weight Factor for Obstacle Avoidance of Joint
$WF(3, L)$: Weight Factor for Obstacle Avoidance of Link
$WF(4, 1)$: Weight Factor for Searching Target
$WWF(4, 2)$: Control Factor of Target Weight
$WWF(L, K)$: Control Factor of Joint Weight
$WWF(L, K, J)$: Control Factor of Link Weight
L : Number of Link
K : Number of Obstacle
J : Number of Sect Link
R : Radius of Obstacle
Q, G : Keep-off Regions

3. 시뮬레이션 및 결과 고찰

3.1 하나의 목표점을 찾아가는 경우

이 문제는 로봇이 어떤 지점에서 다른 지점으로 물건을 움직이는 경우에 해당되는 것으로, Fig. 4를 보면 7개의 링크로 구성된 매니퓰레이터가 목표점(0, 100)의 지점을 찾아가는 경우이고, Fig. 5는 이 때에 (60, 60) 지점에 반지름 20 cm의 장애물이 있을 경우 이 장애물을 피하면서 찾아가는 것을 보여주고 있다.

3.2 첨단부의 경로가 주어진 경우

(1) 고정 장애물일 경우

예제 1) 임의의 3개의 링크로 구성된 매니퓰레이터가 첨단부의 경로 $X(t)=20+10\cdot t$, $Y(t)=40-2\cdot t$로 주어졌을 경우와 (30, 60) 지점에 반지름 7 cm의 장애물이 있을 경우

예제 2) 임의의 3개의 링크로 구성된 매니퓰레이터가 첨단부의 경로 $X(t)=22+4.8t$, $Y(t)=16-3.5\cdot t$로 주어졌을 경우와 (15, 27)지점에 반지름 7 cm의 장애물이 있을 겨우

Fig. 6과 Fig. 7은 예제 1의 경우에 해당하는 시뮬레이션 결과로, Fig. 6은 장애물이 없을 때이고, Fig. 7은 Fig. 6에서 궤적상에 장애물이 있을 경우로 충돌 회피가 원만하게 수행됨을 볼 수 있다. 또 그때의 관절 속도(Joint Velocity)는 Fig. 8과 Fig. 9에 나타나 있는데, Fig. 9를 보면 장애물에 대한 충돌회피가 일어나는 0.4초에서 2.0초 구간에서 관절각의 속도변화가 있음을 보여주고 있다. Fig. 10, Fig. 11, Fig. 12는 예제 2의 경우에 해당하는 시뮬레이션 결과로, Fig. 10은 장애물이 없을 경우

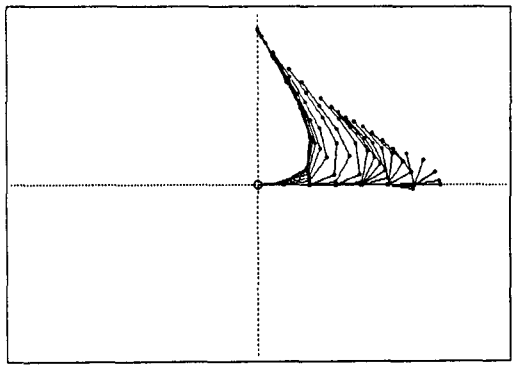

Fig. 4 Simulation result without obstacle

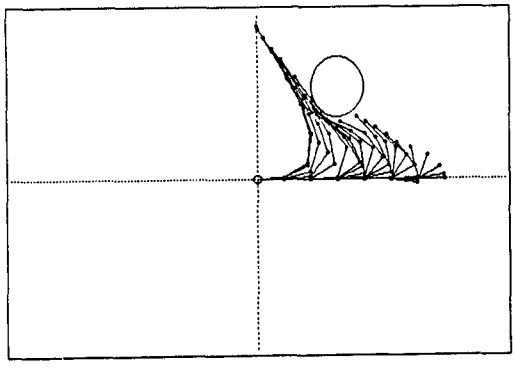

Fig. 5 Simulation result with obstacle

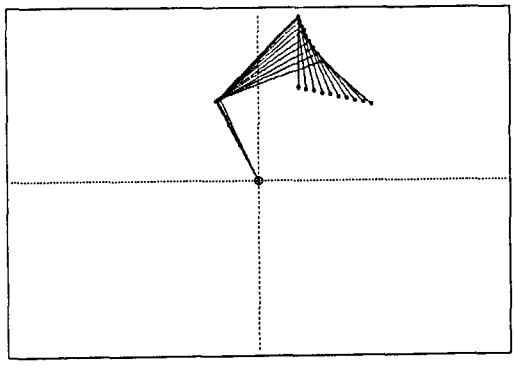

Fig. 6 Simulation result without obstacle of (ex 1)

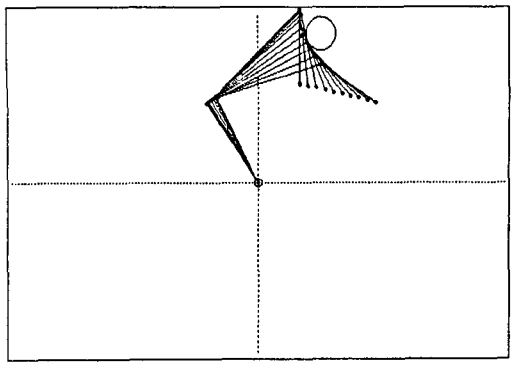

Fig. 7 Simulation result with obstacle of (ex 1)

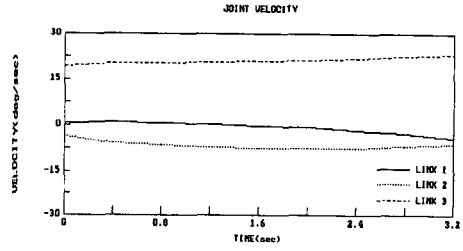

Fig. 8 Joint velocity without obstacle of (ex 1)

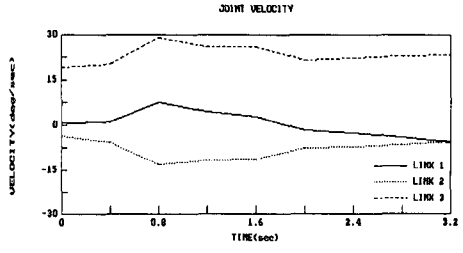

Fig. 9 Joint velocity with obstacle of (ex 1)

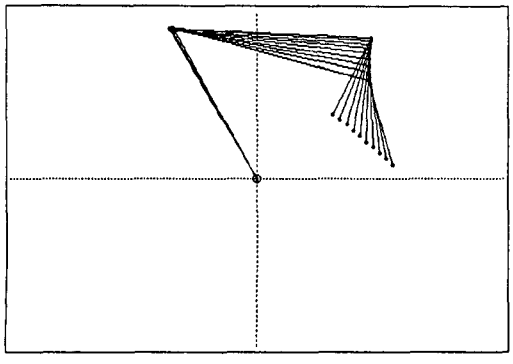

Fig. 10 Simulation result without obstacle of (ex 2)

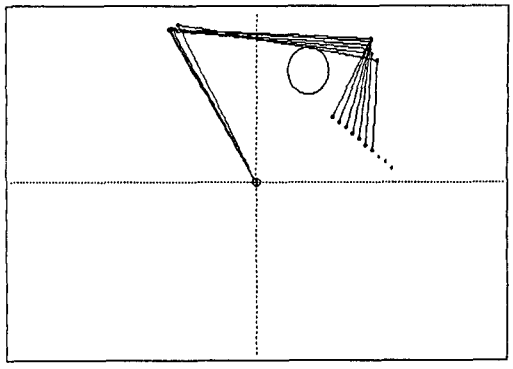

Fig. 11 Simulation result with obstacle of (ex 2) (local minimum)

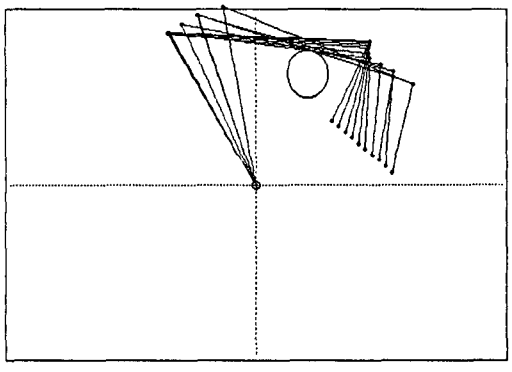

Fig. 12 Simulation result with obstacle of (ex 2)

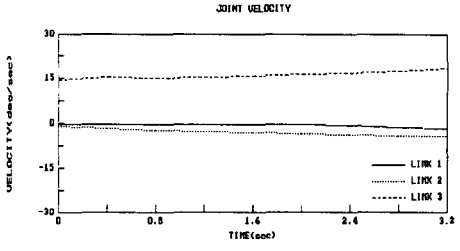

Fig. 13 Joint velocity without obstacle of (ex 2)

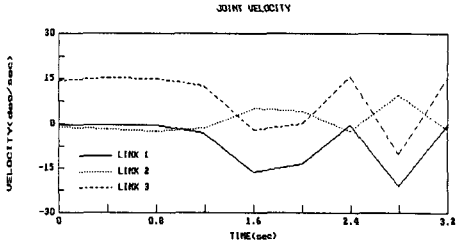

Fig. 14 Joint velocity with obstacle of (ex 2)

이며, 이 궤적상에 장애물이 있을 경우는 Fig. 11과 같이 국부적 최소상태에 빠져서 더 이상 진행을 못하게 되는데, 2.4절에서 소개한 알고리즘 II를 적용시키면 Fig. 12에서와 같이 국부적 최소상태에서 빠져나와 원하는 대로 수행함을 볼 수 있다. 그리고 그때의 관절속도는 Fig. 13과 Fig. 14에 나타나 있는데, Fig. 14에서 보면 역시 충돌회피가 일어나는 1.4초에서 3.2초 구간에서 많은 속도의 변화를 보여주고 있다. 그리고 Table. 1은 알고리즘 II가 적용되었을 때 시뮬레이티드 어닐링에서의 강도의 역할을 하여주는 α와 β값을 구한 것으로, 여기서는 α는 변화가 없고, β만 영향을 미치고 있다.

(2) 장애물이 이동할 경우

예제 3) 첨단부의 경로가 $X(t)=-30\cdot(\cos((\pi/10)\cdot(t+7.5)))+20$, $Y(t)=30\cdot(\sin((\pi/10)\cdot(t+7.5)))+22$이고, 반지름 5 cm의 원형 장애물도 같은 시간 t의 함수 $X(t)=40-5\cdot t$, $Y(t)=(1/40)\cdot((X(t)-20)^2)+30$으로 움질일 경우

예제 4) 반지름 5 cm의 원형 장애물이 $X(t)=40$, $Y(t)=3\cdot t$의 함수로 움직이고, 첨단부의 경로가 이 장애물의 원주상에 있을 경우

Fig. 15는 예제 3에 해당하고, Fig. 16은 예제 4에 해당하는 것으로, 이와 같이 장애물이 움직일 경우도 장애물을 피하며 주어진 경로를 따라 잘 수행함을 볼 수 있다.

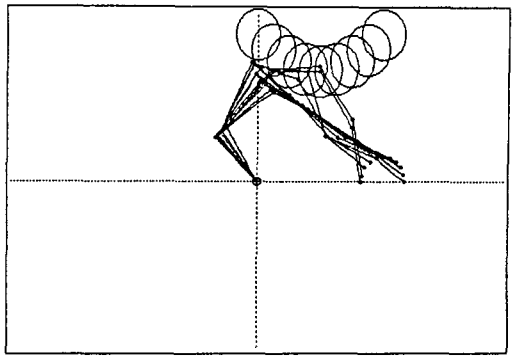

Fig. 15 Simulation result with moving obstacle of (ex 3)

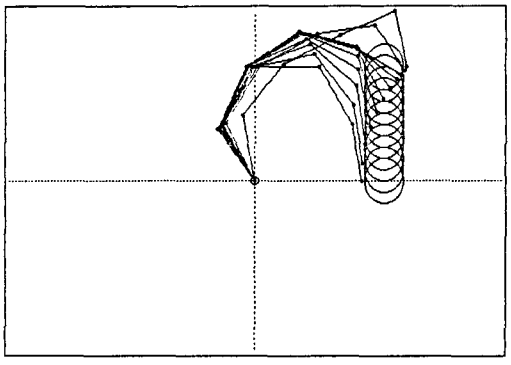

Fig. 16 Simulation result with moving obstacle of (ex 4)

3.3 Dual-arm Type일 경우

이 경우는 실제 산업현장에서 가장 많이 접하게 되는 같은 작업 공간상에서 주어진 경로를 따라 두 대의 로봇이 동시에 작업을 하는 것으로, 여기서는 한 로봇을 주 로봇(master), 다른 한 로봇은 종

Table 1 Results of the simulated annealing

Time	α	β
1.6	1	1.22909
2.0	1	1.64728
3.2	1	2.15061

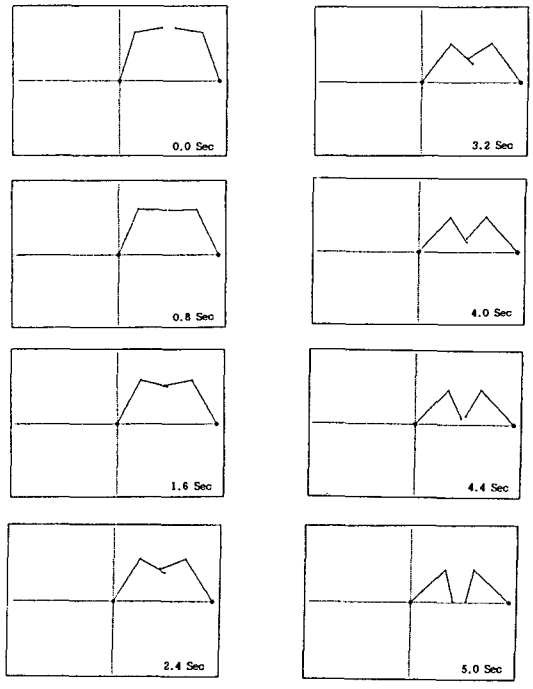

Fig. 17 Simulation results of the dual-arm manipulator

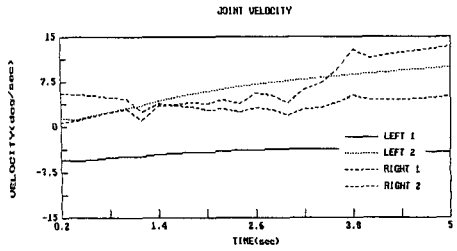

Fig. 18 Joint velocity of dual-arm manipulator

Table 2 Time-delay of collision-avoidance

Time(master)	Time-delay(slave)
1.2	-0.0999
1.4	-0.12471
1.6	-0.15651
1.8	-0.19526
2.0	-0.24796
2.2	-0.24796
2.4	-0.35884
2.6	-0.38356
2.8	-0.42592
3.0	-0.51062
3.2	-0.53826
3.4	-0.54826
3.6	-0.50826
3.8	-0.39974

로봇(slave)으로 지정하여 주로봇은 주어진 경로를 따라가고, 종 로봇은 주 로봇과의 충돌 회피를 위한 시간 지연을 최소로하며 따라가게 된다. 시뮬레이션 결과는 Fig. 17에 나와있으며, Fig. 18은 이때의 관절 속도의 변화를 보여주고 있고, Table 2는 충돌 회피를 위한 종 로봇의 시간 지연을 구한 것이다.

4. 결 론

작업 공간상에 정지 또는 이동 장애물이 존재하는 경우나 다른 로봇이 동시에 움직이는 경우에 있어서 장애물을 피하면서 로봇이 주어진 작업을 하는데 필요한 관절각의 궤적 계획문제를 Hopfield Network을 이용하여 해석함으로써 다음과 같은 결론을 얻을 수 있었다.

(1) 신경회로망의 장점으로서, 링크수나 장애물의 변화 등 상황 변화에 따르는 별도의 복잡한 모델링 과정이 필요없이 관절각의 궤적계획을 효과적으로 해결할 수 있었고, 각 관절들의 위치로부터 관절각들을 동시에 구할 수 있으므로 병렬식(parallel processing) 계산방식에 의한 실시간 제어의 가능성을 제시하여 주었다.

(2) Temperature Adding을 이용한 가중치 조절로 기존의 최적화 기법을 이용할 경우 발생할 수 있는 국부적 최소상태에 빠지는 문제, 즉 Dead-Lock 상태에 빠지는 문제를 새로운 가상 장애물을 만들지 않고 해결할 수 있었다.

(3) 로봇이 여유 자유도를 가지고 있는 경우에 있어서, 별도의 성능 지수를 정의하는 등의 인위적인 조작을 요구하는 알고리즘 개발이 필요없이 로봇 스스로가 최소의 에너지 상태로 찾아가게 함으로써 시간에 따른 관절각들의 궤적을 찾아낼 수 있었다.

(4) 추후 연구과제로는 동역학적 특성을 추가시키는 것과 실제로 이 회로를 하드웨어적으로 구현시키는 것이 있겠다.

참고문헌

(1) Kikuo, F. and Hanan, S., 1989, "Time-Minimal Paths among Moving obstacles," Proc. IEEE intl. Conf on Robotics & Automation, Vol 2, pp 1110~1115.

(2) Klein, C.A. and Huang, C., 1983, "Review of Pseudo Inverse Control for Kinematically Redundant Manipulator," IEEE Trans on Syst, Man and Cybernatics, Vol. SMC-13, pp. 245~250.

(3) Ballieul, J., 1985, "Kinematics Programming Alternatives for Redundant Manipulators," Intl Proc. IEEE Conf on Robotics and Automation, pp. 722~728.

(4) Yoshikawa, T., 1984, "Analysis and Control of Robot Manipulators with Redundancy," Robotics Research : The first international symposium, ed. M.Bracdy and R.Paul. Cambridge : MIT Press, pp. 735~747.

(5) Maciejewski, A.A. and Klein, C.A., 1985,

"Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environments," Intl. J. Robotics Research. vol. 4, No. 8, pp. 109~116.

(6) Graf, H.D. and Lalonde, R.W., 1989, "A Neural Controller for Collision-Free Movement of General Robot Manipulators," IEEE. Intl. Conf on Neural Network, Vol. 1, pp. 77~84.

(7) Grossberg, S. and Bullock, D., 1987, "A Neural Network Architecture for Automatic Trajectory Formation and Coordination of Multiple Effectors during Variable-Speed Arm Movements," Proc. IEEE. Conf on Neural Network, Vol. 4, pp. 559~566.

(8) Grossberg, S. and Kuperstein, M., 1986, "Neural Dynamics of Adaptive Sensory-Motor Control," North-Holland.

(9) Kuperstein, M., 1987, "Adaptive Visual-Motor Coordination in Multijoint Robots Using Parallel Architecture," IEEE, Intl. Conf on Robotics and Automation, pp. 1595~1602.

(10) Hofield, J.J. and Tank, D.W., 1985, "Neural Computation of Decisions Optimization Problems." Biological Cybern, Vol. 52. pp. 141~152.

(11) Hopfield, J.J. and Tank, D.W., 1985, "Collective Computation with Continuous Variables," Disordered Systems and Biological Organization.

(12) Tank, D.W. and Hopfiled, J.J., 1986, "Simple 'Neural' Optimization Network : An A/D Converter, Signal Decision Dircuit, and a Linear Programming Circuit," IEEE Trans on CAS, Vol. CAS-33, No. 5, pp. 533~541.

(13) Beale, R. and Jackson, T., 1990, "Neural Computing : An Introduction," MIT Press.

(14) John, H., Anders, K. and Richard, G.P., 1991, "Introduction to the Theory of Neural Computation," Addison-Wesley Publishing Company.

(15) Pogio, T., 1985, "Early Vision : From Computational Structure to Algorithms and Parallel Hardware," Computer Vision, Graphics and Image Processing, Vol. 31, pp. 139~155.

(16) Tsutsumi, K., 1987, "Neural Computation and Learning Strategy for Manipulator Position Contro," Proc. IEEE. Intl. Conf. Networks. Vol. 4, pp. 525~534.

(17) Yamamoto, M., Ozaki, H. and Mohri, A., 1988, "Planning of Manipulator Joint Trajectories by an Iterative Method," Robotical, Vol. 6, pp. 101~105.

(18) Jasbir, S.A., 1989, "Introduction to Optimum Design," McGraw-Hill Book Company.

(19) Craig, J.J., 1989, "Introduction to Robotics : Mechanics and Control," Addison-Wesley Publishing Comkpany.

(20) Paul, R.P., 1981, "Robot Manipulators : Mathematics, Programming, and Contro," MIT Press.

부 록

$$-C_{1,k}\frac{d}{dt}U_{1,k}=(L_{1,k}-B_k)\cdot(1-\frac{SC_1^1}{SV_1^1})$$
$$-(L_{2,k}-L_{1,k})\cdot(1-\frac{SC_2^2}{SV_2^2})+\sum_{j=2}^{N}(L_{1,k}-L_{j,k})\cdot f(1$$
$$-\frac{SC_1^j}{SV_1^j})+(L_{1,k}-T_k)\cdot f(1-\frac{PC_1}{PV_1})+\sum_{j=1}^{L}(L_{1,k}$$
$$-O_{j,k})\cdot f(1-\frac{HC_1^j}{HV_1^j})+\frac{U_{1,k}}{R_{1,k}} \tag{A1}$$

$$-C_{i,k}\frac{d}{dt}U_{1,k}=(L_{1,k}-L_{i-1k})\cdot(1-\frac{SC_i^i}{SV_i^i})$$
$$-(L_{i+1,k}-L_{i,k})\cdot(1-\frac{SC_{i+1}^{i+1}}{SV_{i+1}^{i+1}})+\sum_{j=1}^{i-1}(L_{i,k}$$
$$-L_{j,k})\cdot f(1-\frac{SC_i^j}{SV_i^j})+\sum_{j=i+1}^{N}(L_{j,k}-L_{i,k})\cdot f(1-\frac{SC_i^j}{SV_i^j})$$
$$+\sum_{j=2}^{N}(L_{1,k}-L_{j,k})\cdot f(1-\frac{SC_1^j}{SV_1^j})+(L_{1,k}-T_k)\cdot f(1$$
$$-\frac{PC_1}{PV_1})+(L_{i,k}-T_k)\cdot f(1-\frac{PC_i}{PV_i})+\sum_{j=1}^{L}(L_{i,k}$$
$$-O_{j,k})\cdot f(1-\frac{HC_i^j}{HV_i^j})+\frac{U_{1,k}}{R_{1,k}}$$
$$(i=1, 2, \cdots, N-1) \tag{A2}$$

$$-C_{N,k}\frac{d}{dt}U_{N,k}=(L_{N,k}-L_{N-1,k})$$
$$\cdot(1-\frac{SC_N^N}{SV_N^N})+\sum_{j=1}^{N-1}(L_{N,k}-L_{j,k})\cdot f(1-\frac{SC_N^j}{SV_N^j})$$
$$+(L_{N\cdot k}-T_k)+\sum_{j=1}^{L}(L_{N,k}-O_{j,k})\cdot f(1-\frac{HV_N^j}{HC_N^j})$$
$$+\frac{U_{N,k}}{R_{N,k}} \tag{A3}$$

여기서, $(k=1, 2, \cdots, k)$

$$U_{i,k}=V_{i,k},\quad f(Z)=\begin{cases}0, & Z\geq 0\\ Z, & Z<0\end{cases}$$