동역학 기반의 지능 힘제어 방식을 이용한 이동 로봇의 장애물 회피에 대한 연구

Collision Avoidance of a Mobile Robot Using Intelligent Force Control Algorithm Based on Robot Dynamics

장 은 수, 정 슬*
(Eun Soo Jang and Seul Jung)

Abstract : In this paper, a new collision avoidance algorithm based on the dynamic model of a mobile robot is proposed. In order to avoid obstacles on the path of a mobile robot, intelligent force control is used to regulate accurate distance between a robot and an obstacle. Since uncertainties from robot and environment dynamics degrade the performance of a collision avoidance task, neural network is used to compensate for uncertainties so that the collision avoidance can be performed intelligently. Simulation studies are conducted to confirm the proposed collision avoidance tracking control algorithm.

Keywords : mobile robot, collision avoidance, force control, path planning, neural network

I. 서론

이동 로봇은 일반 산업현장에서 사용되고 있는 산업로봇과 달리 이동성을 가지고 있는 특징 때문에 다 방면에 걸쳐서 효과적으로 사용되고 있다. 이동 로봇의 주 연구인 자율주행에 관한 연구는 장애물 회피에서부터 위치측정(localization), 경로 계획과 제어 등의 이론적인 연구를 거쳐 실제적인 자율 주행 기반의 작업 수행까지 많은 발전을 가져왔다. 과거에는 엔코더와 같은 주행거리(odometry) 센서에 의한 오측(dead reckoning)에 의해서 부정확한 이동성을 가진 로봇이 주를 이루었으나, 최근에는 다중 및 다종 센서 융합에 의한 정확성을 갖는 로봇에 대한 연구가 대부분이다[1, 2].

이동 로봇의 자율적인 움직임을 위해서 장애물 회피나 위치 측정은 가장 기본적이고 중요한 기술이라 할 수 있다. 전역 위치 측정(global localization)을 위해서는 GPS나 비콘과 같은 절대 센서를 사용해야 하고 주변 환경을 인식하기 위해서는 자이로나 엔코더와 같은 국부적인 센서를 사용하는 방법이 있다. 효율적인 위치 측정을 위해서는 두 방법을 모두 사용하는 것이 바람직하다. 실내에서는 주로 주변 환경의 맵핑에 의한 위치측정, 장애물 회피, 그리고 경로계획이 이루어진다[2, 3]. 주변의 환경을 모델링하기 위한 수단으로 점유 격자를 사용하는 가상 역장(Virtual Force Field) 방법과, 벡터장 히스토그램(Vector Field Histogram) 방법, 그리고 인공전위계 방법(Artificial Potential Field) 등이 있다[4-8]. 이 방법들은 필드의 세기에 따라 목표점까지 정확한 장애물 표현이 가능하고 환경의 변화에 민감하게 반응하는 장점이 있다.

하지만 이동로봇이 장애물 회피하는 이전의 대부분의 연구에 있어서 단순히 기구학 모델에 의존하고 있어 대부분 이동 로봇의 동력학적 특성을 반영하지 못하여 과도한 회피 동작이나 진동이 발생하는 등의 특성을 무시되었다. 본 논문에서 제안하는 힘제어 기반의 장애물 회피 알고리즘은 이동 로봇의 동적 특성을 고려하여 장애물과 일정한 거리를 유지하면서 장애물을 회피하고 이동하기 때문에 문제점인 과도한 회피 동작이나 진동이 발생할 경우 능동적으로 대처할 수가 있다.

본 논문은 장님 이동 로봇에 대한 연구의 연장으로 힘센서 대신에 보다 더 정밀한 센서인 레이저 스캐너 센서를 사용함으로써 실질적인 자율 이동 로봇의 장애물 회피에 적용 가능함을 시뮬레이션을 통해 검증 하고자 한다. 이전의 연구에서는 힘센서에 의존하여 일정한 힘으로 벽을 누르면서 벽을 따라 움직이는 장님 이동 로봇을 제안하였고, 이동 로봇의 기구학 및 동력학 분석에 기초하여 하이브리드 힘제어 방식을 구성하여 보았다[9-10]. 하이브리드 힘제어 방식은 임피던스 힘제어 방식과 함께 대표적인 제어 방식으로 알려져 있다[10-11]. 또한, 로봇이나 환경으로부터 발생하는 불확실성 때문에 발생하는 오차를 줄이기 위해서 지능적인 힘제어 방식을 도입하였다. 지능 힘제어 방식은 신경회로망을 기준입력에 보상함으로써 동적인 불확실성을 보상하는 입력 보상 방식을 사용 하였다[12-13].

본 논문에서 제안하는 지능 힘제어 기반의 장애물 회피 알고리즘을 검증하기 위해 가상의 사각형 모양의 장애물과 원 모양의 장애물을 각각 한 개와 두 개씩을 두어 시뮬레이션 환경을 구성하였다. 로봇의 초기 위치와 최종 목표지점을 정하여 로봇이 장애물과 일정한 거리를 유지하면서 장애물을 회피 및 장애물 추종, 그리고 최단거리 경로로 로봇이 이동하는 것을 시뮬레이션을 통해 확인하였다.

II. 이동 로봇 기구학

그림 1은 이동 로봇과 환경과의 관계를 가상적으로 나타낸다. 실제 환경에 가상적으로 힘을 적용함으로써 발생한 힘의 관계를 나타낸다.

* 책임저자(Corresponding Author)
논문접수 : 2004. 6. 6., 채택확정 : 2004. 7. 18.
장은수, 정 슬 : 충남대학교 메카트로닉스공학과
(cdcmp@hanmail.net/jungs@cnu.ac.kr)

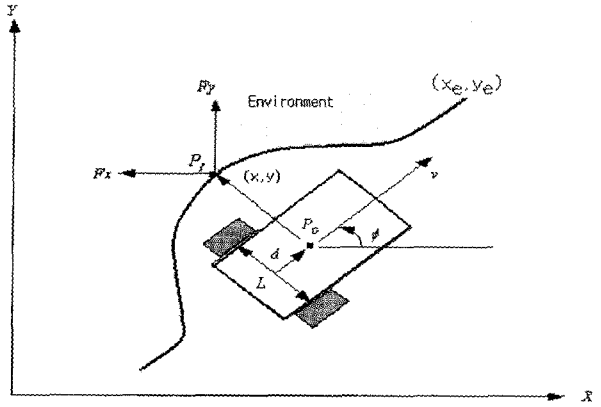


그림 1. 이동 로봇의 기구학.
Fig. 1. Kinematics of a mobile robot.

장애물과 레이저 센서와의 접촉점 P_f 에서의 속도 벡터들은 다음과 같이 로봇의 선속도, v와 각속도, w로 나타내어 질 수 있다[9].

$$\begin{bmatrix}\dot{x}\\ \dot{y}\\ \dot{\phi}\end{bmatrix}=\begin{bmatrix}\cos\phi & -d\cdot\sin\phi-a\cdot\cos\phi\\ \sin\phi & d\cdot\cos\phi-a\cdot\sin\phi\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ w\end{bmatrix} \tag{1}$$

여기서, ϕ 는 좌표에 대한 로봇의 방향 각, d 는 구동축의 중심에서 로봇의 무게 중심까지 거리이고 a 는 무게 중심에서 접촉점까지의 거리이다. 로봇의 선속도와 각속도는 다음과 같이 로봇 양쪽 바퀴의 회전 속도로 나타낼 수 있다.

$$\begin{bmatrix}v\\ w\end{bmatrix}=\begin{bmatrix}\frac{r}{2} & \frac{r}{2}\\ \frac{r}{L} & -\frac{r}{L}\end{bmatrix}\begin{bmatrix}\dot{\theta}_R\\ \dot{\theta}_L\end{bmatrix} \tag{2}$$

여기서 r은 바퀴의 반지름이고 L은 바퀴 양 축 간의 거리, 그리고 $\dot{\theta}_R$ 은 오른쪽 바퀴의 각속도이고 $\dot{\theta}_L$ 은 왼쪽 바퀴의 각속도이다. (1)과 (2)를 합하면 자코비안과 $\dot{\theta}$ 에 의하여 $\dot{Z}$ 를 구할 수 있다.

$$\dot{Z}=J\dot{\theta}\quad,\ J=\begin{bmatrix}j_{11} & j_{12}\\ j_{21} & j_{22}\\ j_{31} & j_{32}\end{bmatrix} \tag{3}$$

$Z\equiv[xy\phi]^T$ 이고, 자코비안(J)의 각 원소는 다음과 같다.

$$\begin{aligned}j_{11}&=\frac{r}{2}\cos\phi-\frac{r}{L}(d\cdot\sin\phi+a\cdot\cos\phi)\\ j_{12}&=\frac{r}{2}\cos\phi+\frac{r}{L}(d\cdot\sin\phi+a\cdot\cos\phi)\\ j_{21}&=\frac{r}{2}\sin\phi+\frac{r}{L}(d\cdot\cos\phi-a\cdot\sin\phi)\end{aligned} \tag{4}$$

$$j_{22}=\frac{r}{2}\sin\phi-\frac{r}{L}(d\cdot\cos\phi-a\cdot\sin\phi)$$

$$j_{31}=\frac{r}{L},\ j_{32}=-\frac{r}{L}$$

III. 하이브리드 힘제어 알고리즘

하이브리드 힘제어 방식은 위치제어와 힘제어 공간을 나누어 제어하는 방식으로 로봇의 힘제어 방식으로 잘 알려져 있다[10]. 조인트 공간에서의 힘제어의 이동로봇의 동역학 식은 다음과 같이 나타내어질 수 있다[15].

$$D(\theta)\ddot{\theta}+C(\theta,\dot{\theta})\dot{\theta}+J^T(\theta)h=\tau \tag{5}$$

여기서 $\theta=[\theta_R\theta_L]^T$ 이고 $h=\begin{bmatrix}f\\ \mu\end{bmatrix}$, f 는 외부로부터의 힘벡터, μ 는 외부 모멘트이고 τ 는 바퀴의 구동 토크이다. (3)의 자코비안 관계를 통해 각속도는 다음과 같다.

$$\dot{\theta}=J^{-1}\dot{Z}, \tag{6}$$

여기서 **Z** 는 위치벡터로 $Z\equiv[xy\phi]^T$ 이다. 하지만 (6)에서 자코비안 행렬의 역행렬을 구할 수 없으므로 아래와 같은 슈도 역행렬을 사용한다.

$$\dot{\theta}=J^{\#-1}\dot{Z} \tag{7}$$

여기서 $J^{\#-1}=(J^TJ)^{-1}J^T$ 이다. (7)을 미분함으로써 가속도텀을 구할 수 있다.

$$\begin{aligned}\ddot{\theta}&=J^{\#-1}(\ddot{Z}-\dot{J}J^{\#-1}\dot{Z})\\ &=(J^TJ)^{-1}J^T(\ddot{Z}-\dot{J}(J^TJ)^{-1}J^T\dot{Z})\end{aligned}, \tag{8}$$

여기서 $\dot{Z}=\begin{bmatrix}\dot{x} & \dot{y} & \dot{\phi}\end{bmatrix}^T$ 그리고 $\dot{\theta}=\begin{bmatrix}\dot{\theta}_R & \dot{\theta}_L\end{bmatrix}^T$.
따라서 제어 법칙은 다음과 같다.

$$\tau=\hat{D}(\theta)u+\hat{C}(\theta,\dot{\theta})\dot{\theta}+J^T(\theta)h \tag{9}$$

여기서 **u**는 제어 입력 벡터를 나타내고 $\hat{D},\hat{C}$ 는 각각 로봇 동역학 모델 ***D***, ***C*** 의 평가치 행렬을 나타낸다. 하이브리드 힘제어 알고리즘에서 제어 입력 **u**는 위치제어 입력과 힘제어 입력으로 나뉘어지며 스위칭을 통해 선택적으로 활성화된다. 먼저 힘제어 방향에 있어서 제어입력을 알아보자. 출력 벡터 **F**를 다음과 같이 정의하자.

$$F\equiv\begin{bmatrix}f_x & f_y & \phi\end{bmatrix}^T, \tag{10}$$

여기서 x 방향과 y 방향의 힘은 각각 다음과 같이 정의된다.

$$f_x = k_e \delta x$$
$$f_y = k_e \delta y \quad (11)$$

여기서 $\delta x = x - x_e$, $\delta y = y - y_e$ 이고 x_e, y_e 는 장애물의 위치, 그리고 x, y 는 접촉점의 위치, k_e 는 가상 장애물의 강성도를 나타낸다. (10)과 (11)로 부터 힘벡터는 다음과 같은 행렬로 표현되어 질 수 있다.

$$\begin{aligned} F &= K_e \Delta Z \\ &= K_e (Z - Z_e) \\ &= \begin{bmatrix} k_e & 0 & 0 \\ 0 & k_e & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x - x_e \\ y - y_e \\ \phi \end{bmatrix} \end{aligned} \quad (12)$$

일반적으로 $\dot{Z}_e = 0, \ddot{Z}_e = 0$ 이므로 (12)로 부터

$$\ddot{Z} = K_e^{-1} \ddot{F} \quad (13)$$

(13)을 (8)에 대입하면 가속도를 얻게 된다.

$$\begin{aligned} \ddot{\theta} &= J^{\#-1}(\ddot{Z} - \dot{J}\dot{J}^{-1}\dot{Z}) \\ &= J^{\#-1} K_e^{-1}(\ddot{F} - \dot{J}\dot{J}^{\#-1}\dot{F}) \end{aligned} \quad (14)$$

(14)의 가속도 텀을 제어 입력으로 사용하여 $u = \ddot{\theta}$ 를 (9)에 대입하면 다음과 같은 제어법칙을 얻게 된다.

$$\begin{aligned} \tau &= \hat{D}(q)u + J^T(q)h \\ &= \hat{D}[J^{\#-1} K_e^{-1}(\ddot{F} - \dot{J}\dot{J}^{\#-1}\dot{F})] + \hat{C} + J^T h \end{aligned} \quad (15)$$

힘제어 방향에서의 제어입력 $\ddot{F}$ 은 다음과 같이 PD 제어기 형태로 설정된다.

$$u = \ddot{F} = \ddot{F}_d + K_D(\dot{F}_d - \dot{F}) + K_P(F_d - F) \quad (16)$$

(16)을 (15)에 대입하면 다음과 같은 제어 법칙을 얻게 된다.

$$\begin{aligned} \tau = \hat{D} J^{\#-1} K_e^{-1} [\ddot{F}_d + K_D(\dot{F}_d - \dot{F}) - \\ \dot{J}\dot{J}^{\#-1}\dot{F} + K_P(F_d - F)] + \hat{C} + J^T h \end{aligned} \quad (17)$$

위치 제어 방향의 제어 법칙도 마찬가지로 PD 제어 형태의 방법에 의해 유도 할 수 있다.

$$u = \ddot{Z} = \ddot{Z}_d + K_D(\dot{Z}_d - \dot{Z}) + K_P(Z_d - Z) \quad (18)$$

위치제어에서는 $h = 0$ 이므로 (18)을 (9)에 대입하면 다음과 같은 제어 법칙을 얻게 된다.

$$\tau = D J^{\#-1}[Z_d + K_D(\dot{Z}_d - Z) - \dot{J}\dot{J}^{\#-1}\dot{Z} + K_P(Z_d - Z)] + \hat{C} \quad (19)$$

하이브리드 힘제어 방식은 위치제어 방향과 힘제어 방향을 선택행렬에 의해 스위칭하므로 전체적인 제어법칙은 선택행렬 **S** 를 포함하게 된다. 위치제어를 위해서는

$S = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$ 그리고 힘제어는 $S = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$ 을 선택하도록 제어법칙을 구성하면 전체적인 제어법칙은 다음과 같다.

$$\begin{aligned} \tau = S[\hat{D} J^{\#-1}\{\ddot{Z} + K_D(\dot{Z}_d - \dot{Z}) - \dot{J}\dot{J}^{\#-1}\dot{Z} + K_P(Z_d - Z)\} + \hat{C}] \\ + (I - S)[\hat{D} J^{*-1} K_e^{-1}\{\ddot{F} + K_D(\dot{F}_d - \dot{F}) - \dot{J}\dot{J}^{\#-1}\dot{F} + K_P(F_d - F)\} + \hat{C} + J^T h] \end{aligned} \quad (20)$$

여기서 $k_e = \begin{bmatrix} k_e & 0 & 0 \\ 0 & k_e & 0 \\ 0 & 0 & 1 \end{bmatrix}$ $K_D = \begin{bmatrix} K_{Dx} & 0 & 0 \\ 0 & K_{Dy} & 0 \\ 0 & 0 & K_{D\phi} \end{bmatrix}$, $K_P = \begin{bmatrix} K_{Px} & 0 & 0 \\ 0 & K_{Py} & 0 \\ 0 & 0 & K_{P\phi} \end{bmatrix}$

로봇의 동적 모델이 정확하고 불확실성이 없다고 가정할 경우에, 즉 $\hat{D} = D, \hat{C} = C$ 이면, 오차 방정식은 다음과 같다.

$$\ddot{e} + K_D\dot{e} + K_P e = 0 \quad (21)$$

여기서 $e = F_d - F$ or $e = Z_d - Z$. 하지만 로봇의 동역학을 정확하게 알 수 없고 비선형성 동적 특성이 존재하므로 (21)을 만족하지 못하게 된다. 따라서 이러한 불확실성을 없애기 위해 지능적인 알고리즘이 필요하다. (20)의 제어법칙을 블록도로 나타내면 그림 2와 같다.

IV. 지능 하이브리드 힘제어

일반적으로 (21)을 만족하기란 매우 어렵다. 따라서 여기서는 신경회로망을 사용하여 불확실성을 보상하여 오차를 줄이는 알고리즘을 제안한다. 제안하는 방식은 입력 보상 방식으로 신경망 제어기는 제어 회로 밖에서 선 필터를 사용하여

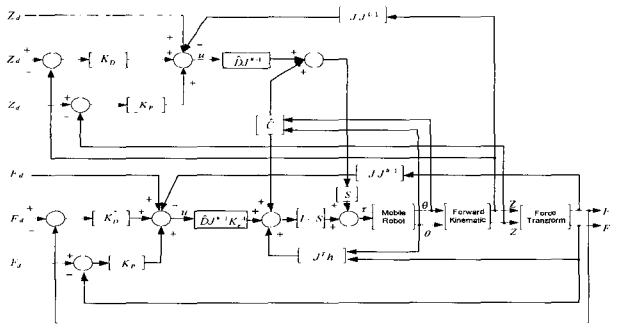


그림 2. 하이브리드 힘제어 블록도.
Fig. 2. Hybrid forced control block diagram.

보상을 실행 한다[12-13]. PID 제어기로 들어오는 오차의 입력에 신경망의 출력값과 더해져서 오차를 점점 줄여 나가게 된다. 이러한 방식을 사용함으로써 기존의 제어기를 수정하지 않고 시스템 외부에서 내부의 불확실성을 보상할 수 있다.
위치제어와 힘제어 부분에 각각 신경회로망이 사용되며, 신경회로망의 보상 신호 Φ는 입력 경로에 더해진다. 먼저 힘제어 부분의 보상을 살펴보면

$$\ddot{F} = \ddot{F}_d + K_D(\dot{F}_d - \dot{F}) + K_P(F_d - F + \Phi_f) \quad (22)$$

여기서 Φ_f 는 신경회로망의 보상 신호이다.
(22)를 (15)에 대입하면 다음과 같은 제어 법칙을 얻게 된다.

$$\tau = \hat{D}J^{\#-1}K_e^{-1}[\ddot{F}_d + K_D(\dot{F}_d - \dot{F}) - \dot{J}J^{\#-1}\dot{F} + K_P(F_d - F + \Phi_f)] + \hat{C} + J^T h \quad (23)$$

마찬가지로 위치 제어일 때에는 위치에 신경회로망 출력을 더하면 다음과 같다.

$$\ddot{Z} = \ddot{Z}_d + K_D(\dot{Z}_d - \dot{Z}) + K_P(Z_d - Z + \Phi_p) \quad (24)$$

(24)를 (15)에 대입하면 다음과 같은 위치제어 법칙을 얻게 된다.

$$\tau = \hat{D}J^{\#-1}[Z_d + K_D(\dot{Z}_d - Z) - \dot{J}J^{\#-1}\dot{Z} + K_P(Z_d - Z + \Phi_p)] + \hat{C} \quad (25)$$

선택 행렬 **S**를 포함하는 전체적인 제어 법칙을 구하면 다음과 같다.

$$\tau = S[\hat{D}J^{\#-1}\{\ddot{Z} + K_D(\dot{Z}_d - \dot{Z}) - \dot{J}J^{\#-1}\dot{Z} + K_P(Z_d - Z + \Phi_p)\} + \hat{C}] \quad (26)$$
$$+(I-S)[\hat{D}J^{*-1}K_e^{-1}\{\ddot{F} + K_D(\dot{F}_d - \dot{F}) - \dot{J}J^{\#-1}\dot{F} + K_P(F_d - F + \Phi_f)\} + \hat{C} + J^T h]$$

여기서,

$$k_e = \begin{bmatrix} k_e & 0 & 0 \\ 0 & k_e & 0 \\ 0 & 0 & 1 \end{bmatrix} K_D = \begin{bmatrix} K_{Dx} & 0 & 0 \\ 0 & K_{Dy} & 0 \\ 0 & 0 & K_{D\theta} \end{bmatrix}, K_P = \begin{bmatrix} K_{Px} & 0 & 0 \\ 0 & K_{Py} & 0 \\ 0 & 0 & K_{P\theta} \end{bmatrix}$$

그림 3은 지능 힘제어 블록 다이아그램을 나타낸다.

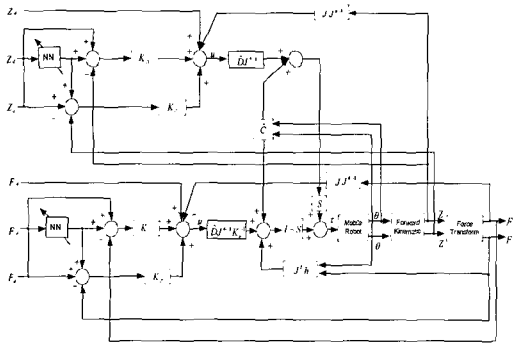


그림 3. 지능 하이브리드 힘제어 블록도.
Fig 3. Intelligent hybrid force control block diagram.

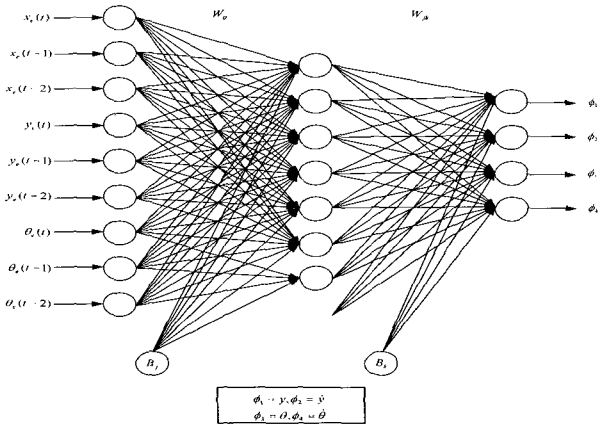


그림 4. 신경망 구조.
Fig 4. The structure of neural network.

V. 학습 알고리즘

역전파 알고리즘을 사용하여 시스템 자코비안이 필요 없이 실시간 학습이 가능하도록 하였다. 그림 4는 신경회로망의 구조를 나타낸다.

입력층 9개, 은닉층 7개, 출력층 4개로 이루어져 있다. 신경망 입력은 x, y, θ의 현재 오차와 이전오차, 그 이전 오차를 사용하였다. 신경망 출력은 로봇의 위치 y 와 헤딩 각 θ에 각각 더해 주었다. 그리고 뉴런의 비선형 함수는 아래와 같은 Tangent hyperbolic 함수를 사용하였다.

$$f(x) = \frac{1 - \exp(-x)}{1 + \exp(-x)} \quad (27)$$

신경망의 출력은 다음과 같이 하였다.

$$\phi = \phi_y + \phi_\theta \quad (28)$$

$$\phi_y = k_{py}\phi_1 + k_{dy}\phi_2 \quad (29)$$

$$\phi_\theta = k_{p\theta}\phi_3 + k_{d\theta}\phi_4 \quad (30)$$

신경망을 학습하는 학습 신호는 (31)과 같이 설정하고, 목적함수는 (32)와 같이 설정하여 목적함수가 0 이 되도록 신경망을 학습시킨다.

$$t_n = k_{py}e_y + k_{dy}\dot{e}_y + k_{p\theta}e_\theta + k_{d\theta}\dot{e}_\theta \quad (31)$$

$$E = \frac{1}{2}t_n^2 \quad (32)$$

(32)를 가중치에 대해서 미분하면 목적함수 E의 그래디언트를 (33)처럼 구할 수 있다.

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial t_n}\frac{\partial t_n}{\partial w} = t_n\frac{\partial t_n}{\partial w} = -t_n\frac{\partial \phi}{\partial w} = -t_n\left(\frac{\partial \phi_y}{\partial w} + \frac{\partial \phi_\theta}{\partial w}\right) \quad (33)$$

$$\frac{\partial \phi_y}{\partial w} = k_{py}\frac{\partial \phi_1}{\partial w} + k_{dy}\frac{\partial \phi_2}{\partial w} \quad (34)$$

$$\frac{\partial \phi_\theta}{\partial w} = k_{p\theta}\frac{\partial \phi_3}{\partial w} + k_{d\theta}\frac{\partial \phi_4}{\partial w} \quad (35)$$

위에서 구한 (33), (34), (35)를 이용하여 역전파 알고리즘에 사용하면 다음과 같다.

$$\Delta w(t) = \eta \frac{\partial E}{\partial w} + \alpha \Delta w(t-1) \tag{36}$$

$$w(t+1) = w(t) + \Delta w(t) \tag{37}$$

여기서, η 는 학습률이고 α 는 운동량 상수이다.

VI. 레이저 센서 및 장애물 추종 알고리즘

본 논문에서는 실제 레이저 센서와 같은 센서 정보를 얻기 위해서 가상의 센서 정보를 만들었다. 레이저 센서 모델로는 SICK 사의 LMS200을 사용하였고 0.5° 간격으로 180° 스캔을 하며 한 번 스캔에 총 361개의 센서 데이터가 입력된다. 그러나 보다 간단하게 가상의 센서 정보를 얻기 위해서 0.5° 간격을 5° 간격으로 줄여 한 번 스캔에 입력되는 데이터를 37개로 축소하였다. 입력되는 각각의 데이터의 정보에 의해서 이동로봇이 장애물을 만났을 때 회피 및 일정한 거리 간격을 두고 추종하게 된다.

본 실험에서는 레이저 센서라는 직접적인 거리 값이 입력되는 가상의 센서를 사용했기 때문에 힘센서를 사용했을 때의 접촉점의 좌표인 x, y, x_e, y_e 를 센서 거리값의 오차로 바로 f_x 와 f_y 에 적용하였다. 여기서 x, y 는 레이저 센서의 거리 값이고, x_e, y_e 는 원하는 경로 값이다. 여기서, 원하는 경로와 장애물 사이의 거리는 70cm로 설정하였으며, 각각의 센서 값이 미리 정해놓은 70cm의 거리보다 커지면 장애물이 없는걸 판단하여 목표지점으로 이동한다. 각각의 센서값이 70cm에 근접하면 장애물이 있다는 것을 식별하여, 로봇의 헤딩 각이 바뀌게 된다. 센서의 직접적인 거리값에 의해 구하여진 거리오차는 로봇과 원하는 경로와의 거리 오차로 표현된다. 그림 5 는 일정한 거리를 유지하며 움직이는 로봇을 나타낸다.

VII. 단거리 경로 계획

본 논문에서 사용한 단거리 경로 계획은 사람이 장애물을 봤을 때, 어느쪽으로 가야 가장 빨리 가야 하는 지와 흡사하다. 단거리 경로 계획 알고리즘을 적용하기 위해서는 우선

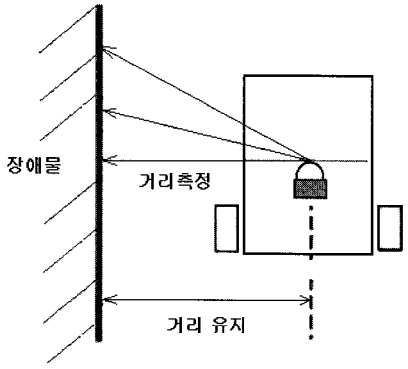


그림 5. 거리를 유지하면 움직이는 로봇.
Fig. 5. Tracking robot of maintaining distance.

장애물을 감지 하여야 한다[9]. 그림 6에서와 같이 스캔을 처음 시작하는 점을 시작점으로 정하고, 스캔이 끝나는 점을 끝점으로 정했다. 결과적으로 이 두 점을 이용하면 장애물의 대략적인 모양이 나오게 된다.

시작점과 끝점을 찾기 위해서는 레이저 센서가 현재 얻은 값과 1step 이전에 얻은 값에 의해서 구별 되고 하나의 clustering을 형성하게 된다.

$$\mathrm{d} = \sqrt{(x_m - x_{m-1})^2 + (y_m - y_{m-1})^2} \quad (\mathrm{m} = 1,2,3) \tag{38}$$

(38)에서 x, y 의 좌표는 레이저 센서의 입력 좌표이다. 임계값은 레이저 센서의 분해능을 감안하여 50cm로 두었다. 레이저 센서는 반 시계 방향부터 스캔한다. 따라서 입력되는 레이저 값과 그 이전의 레이저 값을 (38)에서 계산으로 구하여진 d를 임계값과 비교하여 작으면 시작점이 정해 진다. 그리고, 그 다음을 확인하여 d가 임계값 보다 커지면 그 지점에서 끝점이 정해진다.

그림 6은 레이저 센서의 시작점과 끝점을 이용하여 로봇의 방향성을 도식적으로 나타낸 그림이다. 끼인 각 및 각 지점까지의 거리는 간단한 삼각함수 공식으로 구할 수 있다. 먼저 각 α_1, α_3 는 다음과 같이 구할 수 있다.

$$\alpha_1 = \tan^{-1}\left(\frac{e_y - R_y}{e_x - R_x}\right) - desired\_\theta \tag{39}$$

$$\alpha_3 = desired\_\theta - \tan^{-1}\left(\frac{s_y - R_y}{s_x - R_x}\right)$$

시작점과 끝점과의 거리 L은

$$L = \sqrt{a^2 + b^2 - 2ab\cos(\alpha_1 + \alpha_3)} \tag{40}$$

각 α_2, α_4 는 다음과 같이 구한다.

$$\alpha_4 = \cos^{-1}\left(\frac{-b^2 + (a^2 + L^2)}{2aL}\right) \tag{41}$$

$$\alpha_2 = \pi - (\alpha_4 + \alpha_1 + \alpha_3)$$

거리 d_x 는

$$d_x = \frac{a}{\sin(\pi - (\alpha_4 + \alpha_3))}\sin\alpha_4 \tag{42}$$

거리 L_r, L_l 은 다음과 같이 구한다.

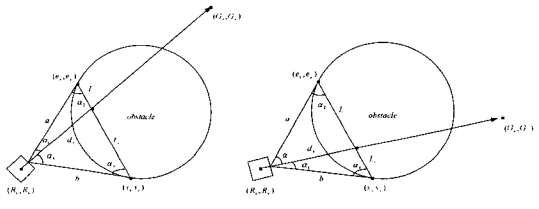


그림 6. 단거리 경로 계획 1.
Fig. 6. Short distance path plan 1.

거리 L_r, L_l 은 다음과 같이 구한다.

$$L_r = \frac{d_x}{\sin\alpha_4}\sin\alpha_3,\ L_l = L - L_r \tag{43}$$

R_x, R_y 는 로봇의 직교좌표 상의 현재 좌표이고, s_x, s_y 는 시작 점의 x, y 좌표가 되겠고, e_x, e_y 는 끝점의 x, y 좌표 이다. (43)에 의해서 L_l 과 L_r 의 길이와 α_1 과 α_3 의 각도를 비교하여 서로 작은 값이 나오는 곳으로 로봇의 방향이 결정된다.

VIII. 시뮬레이션 결과

1. 실험 환경

시뮬레이션은 위에서 실험한 동일한 환경에서 실험을 하였다. PD 이득값과 로봇의 이동 속도를 달리 주어서 비선형 텀이 가해진 상황이라고 가정하여 실험을 하였다. 각각의 변수들은 표 3과 같이 주어졌다. 여기서 η, α 는 신경회로망의 학습율과 모멘텀의 변수이고 K_P, K_D 는 PD 제어기의 변수이다. 그리고 로봇이 이동 중 위치제어와 힘제어의 두 가지 제어를 감안하여 신경망을 위치와 힘 두 부분으로 사용 하였다.

2. 시뮬레이션 결과

시뮬레이션은 하이브리드 힘제어만 사용했을 경우와 신경망을 추가했을 경우로 나누었다. 장애물의 실험 환경은 한 개의 긴 사각형 장애물과 두 개의 사각형 장애물, 그리고 두 개의 원 장애물과 원과 사각 장애물이 같이 있을 때의 경우를 실험하였다.

1) 한 개의 사각형 장애물

아래 그림 7과 8은 각각 사각형 장애물이 있을 경우를 나타낸다. 그림 7에서 보다 그림 8에서 신경회로망이 보상하므로 인해 거리를 잘 유지하는 것을 볼 수 있다.

2) 두개의 사각형 장애물

그림 9와 그림 10은 두개의 사각형 장애물이 연속적으로 있을 경우를 나타낸다. 하나의 장애물을 회피한 뒤에 목적지로 향하여 가다가 또 다른 장애물을 만나 회피해 가는 것을 볼 수 있다. 추종 결과는 서로 비슷하게 보이지만 신경망을 사용한 그림 10의 경우 거리가 잘 유지됨을 볼 수 있다. 또한 그림 9와 10에서 또 다른 장애물이 나타났을 경우에 동적인

표 3. 변수 설정 값.
Table 3. Parameter values.

	설정값	변수	설정값
K_{PX}	90	K_{DX}	30
K_{PY}	70	K_{DY}	20
$K_{P\theta}$	40	$K_{D\theta}$	10
Position η	0.05	Force η	0.03
Position α	0.5	Force α	0.3

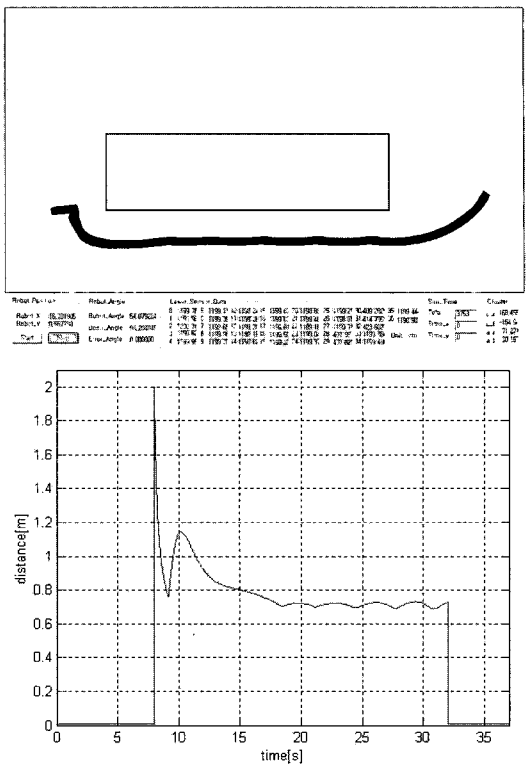


그림 7. 하이브리드 힘제어 방식.
Fig 7. Hybrid force control.

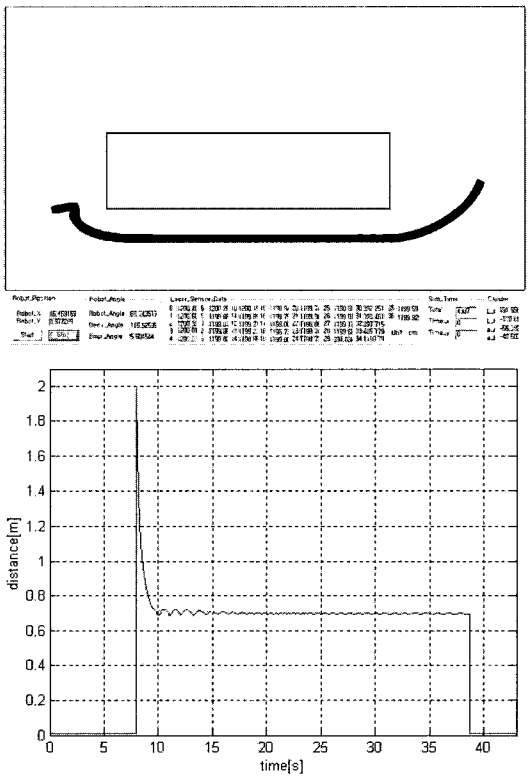


그림 8. 지능 하이브리드 힘제어 방식.
Fig 8. Intelligent hybrid force control.

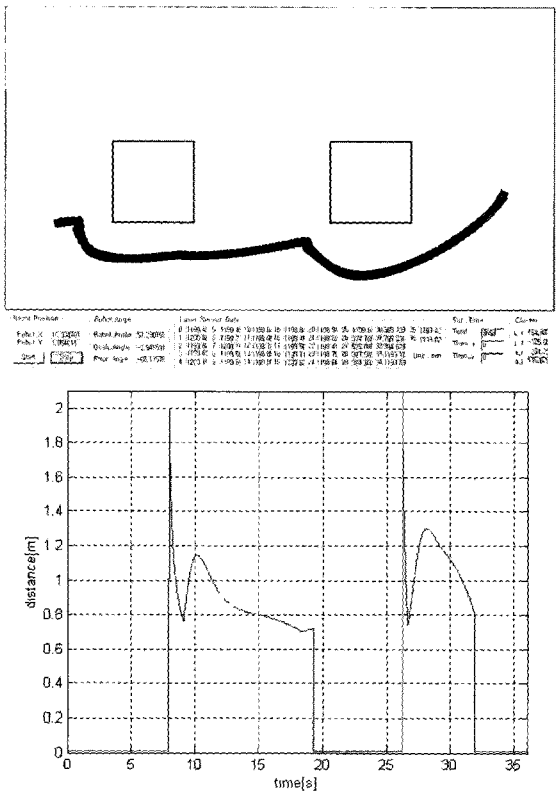


그림 9. 하이브리드 힘제어 방식.

Fig 9. Hybrid force control.

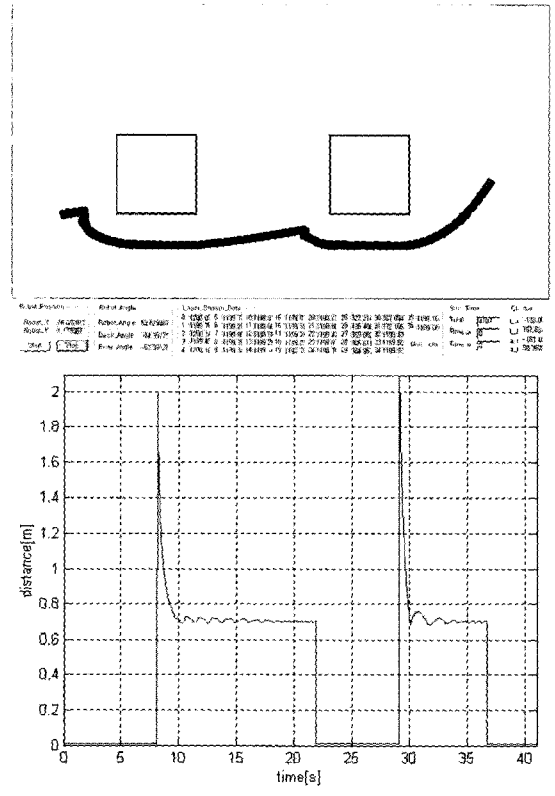


그림 10. 지능 하이브리드 힘제어 방식.

Fig 10. Intelligent hybrid force control.

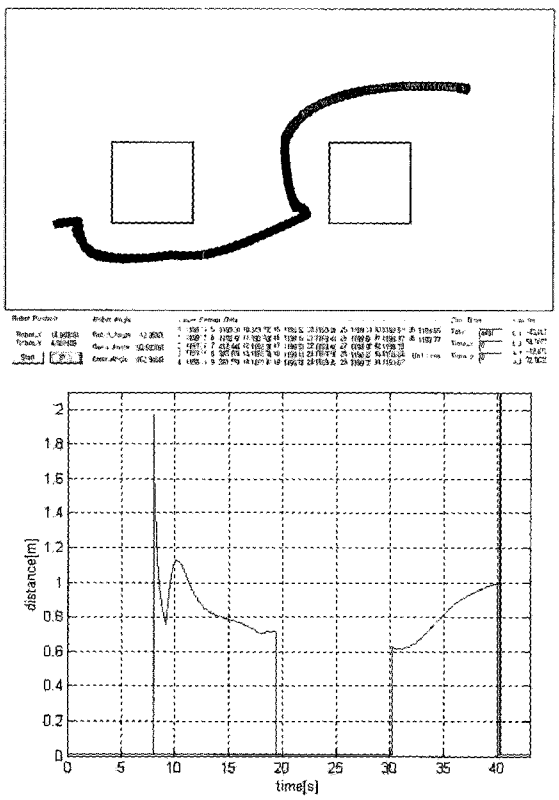


그림 11. 하이브리드 힘제어 방식.

Fig 11. Hybrid force control.

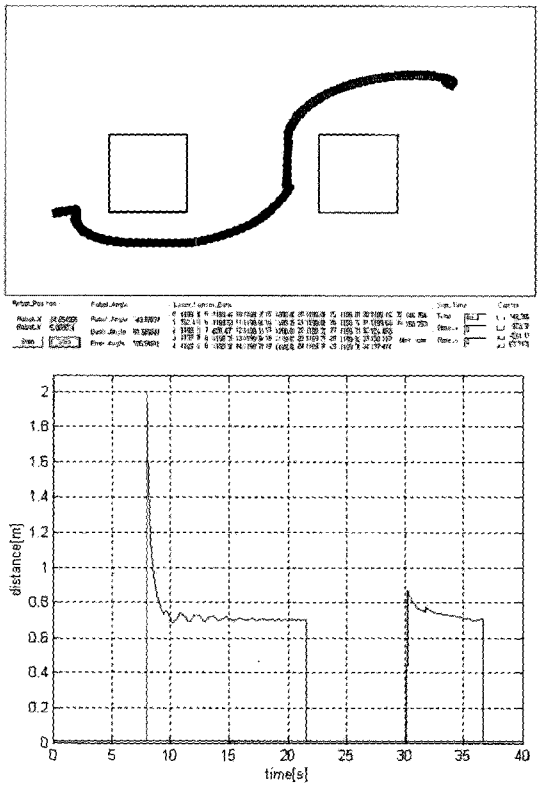


그림 12. 지능 하이브리드 힘제어 방식.

Fig 12. Intelligent hybrid force control.

요소에 의해 다소의 오버슈트가 발생함을 볼 수 있다. 그림 10의 경우에 더 잘 추종하는 것을 볼 수 있다.

이번에는 목적지가 다를 경우 로봇이 최단 거리를 찾아가는지 알아보았다. 그림 11과 12에서 목적지까지 최단 거리로 움직이는 것을 볼 수 있다. 그림 11의 경우 로봇이 안으로 들어갔다가 다시 나오는 동적인 현상을 볼 수 있으며, 0.7m 보다 크게 추종하는 것을 볼 수 있다. 그림 12의 경우, 거리 추종이 우수함을 볼 수 있다.

3) 두개의 원형 장애물

아래 그림은 위의 예제와 목표점이 다를 경우에 로봇이 단 거리를 찾아 움직이는 것을 보여준다.

그림 13과 14를 비교해 보면 그림 13의 경우 두 번째 장애물의 경우에 잘 추종하지 못하였으나 그림 14에서 보면 신경회로망을 사용했을 경우에는 잘 추종하는 것을 볼 수 있다.

4) 사각형 + 원 장애물

또 다른 실험으로 사각형과 원 장애물을 연속적으로 회피하는 것을 하였다. 그림 15에서는 기준 거리 0.7m를 초과하여 추종하고 있으며, 그림 16의 경우에는 기준 거리를 잘 추종하는 것을 볼 수 있다.

그림 15의 경우에도 기준 거리보다 다소 크게 추종하는 것을 볼 수 있으며, 두 번째 원의 장애물의 경우에는 잘 추종하지 못하는 것을 볼 수 있다.

하지만 그림 16의 경우, 신경망을 사용하면 기준 거리를

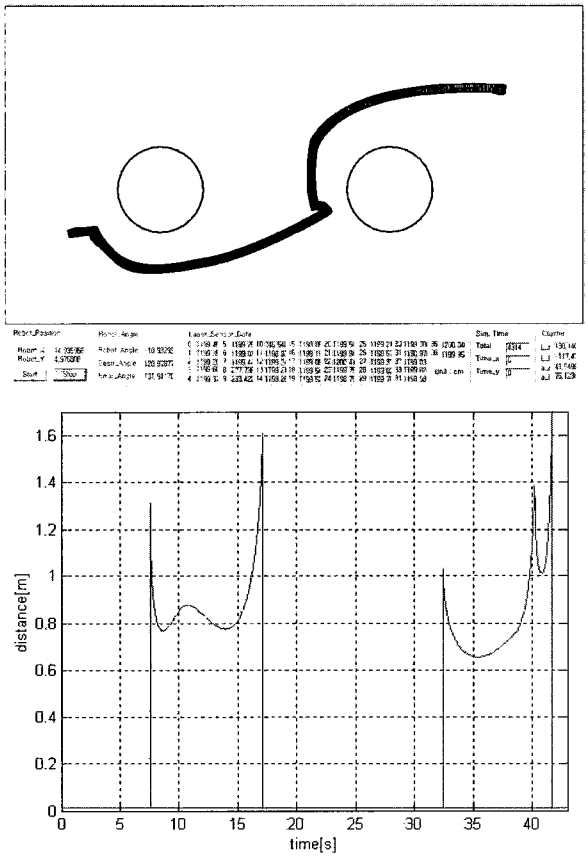


그림 13. 하이브리드 힘제어 방식.

Fig 13. Hybrid force control.

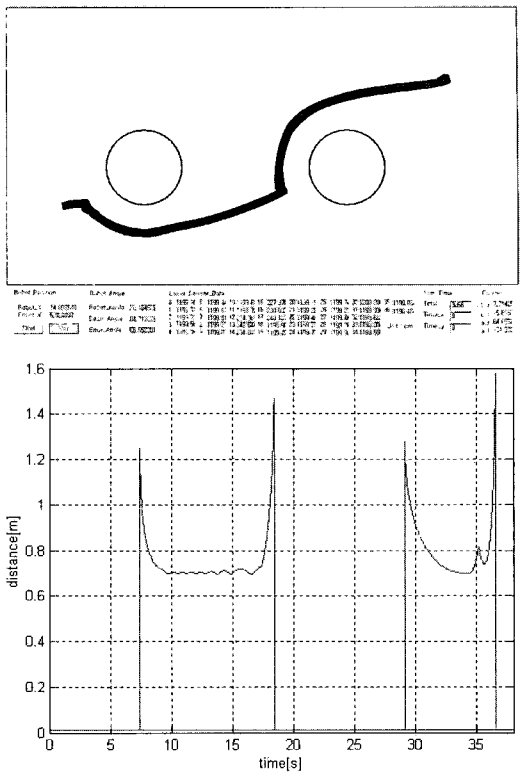


그림 14. 지능 하이브리드 힘제어 방식.

Fig 14. Intelligent hybrid force control.

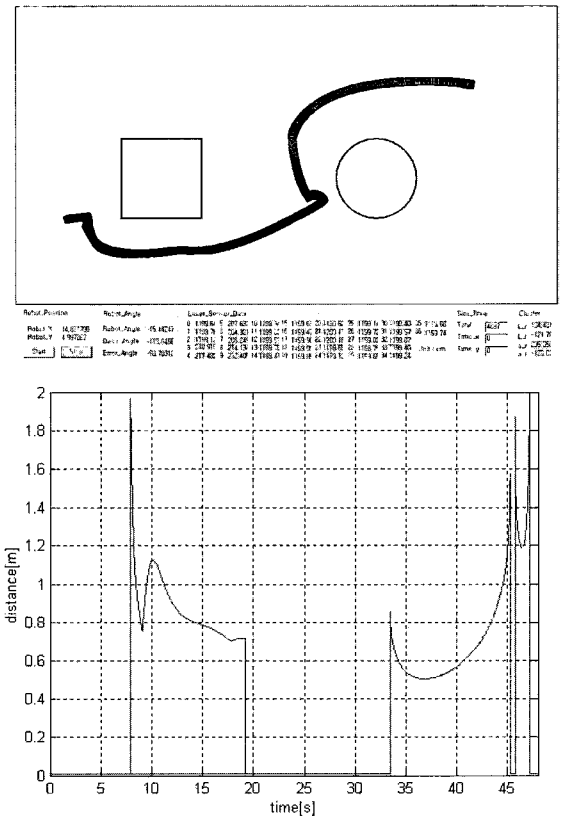


그림 15. 하이브리드 힘제어 방식.

Fig 15. Hybrid force control.

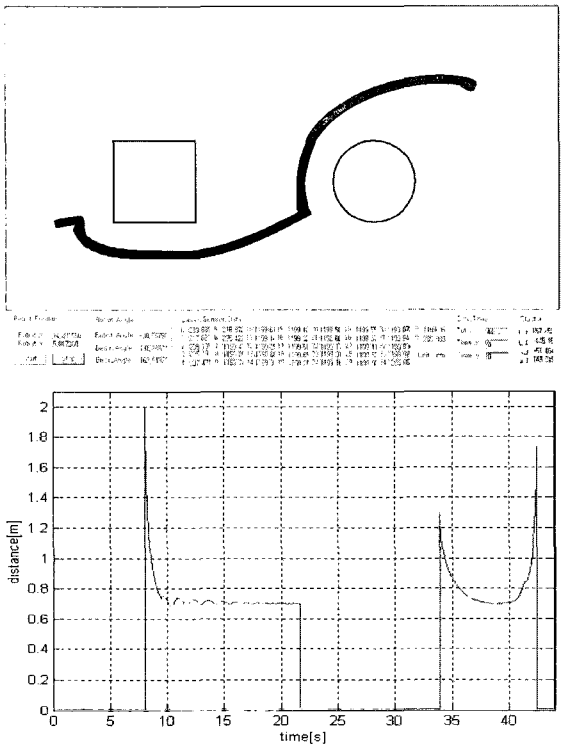


그림 16. 지능 하이브리드 힘제어 방식.
Fig 16. Intelligent hybrid force control.

잘 유지하면서 추종하는 것을 볼 수 있다. 두 번째 나타난 원의 장애물의 경우에서도 잘 추종하는 것을 볼 수 있다.

V. 결론

본 논문에서는 힘제어 알고리즘을 기반으로 이동 로봇의 가장 기본적인 기능인 장애물 회피 및 단거리 경로에 의한 목적지까지의 움직임을 시뮬레이션을 통해 검증 하였다. 이전에 제안된 장애물 회피 알고리즘 중 단점이었던 과도 회피 현상 등을 장애물과 로봇과의 동적인 관계, 즉 위치와 힘과의 관계를 제어하는 힘제어 알고리즘을 적용하여 보다 안정적인 이동로봇의 경로추종을 구현할 수가 있었다. 로봇은 장애물이 있을 경우 성공적으로 단거리를 찾아 목표점에 도달함을 볼 수 있었다. 그리고 로봇의 이동 속도와 제어기의 이득값을 달리 했을 때 나타나는 불확실성에 의한 거리 추종 오차를 신경망을 사용한 제어기의 입력 보상 방식에 의해서 오차를 줄일 수 있었다.

참고문헌

[1] J. Borenstein, H. R. Everett, and L. Feng, "Navigating Mobile Robots: System and Technique", A. K Peters, 1996.

[2] J. A. Castellanos and J. D. Tardos, "Mobile Robot Localization and Map Building", Kluwer Academic Publishers, 1999.

[3] Z. Xu, J. Liu, Z. Xiang, H. Li, "Map Building for Indoor Environment with Laser Range Scanner", *IEEE Intl. Conference on Intelligent Transportation Systems,* pp. 136-140, 2002.

[4] J. Borenstein, "The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots", *IEEE Trans, on robotics and automation, pp. 278-288,* vol. 7, no. 3, 1991.

[5] O. Khatib, "Real-time Obstacle Avoidance for Manipulators and Mobile Robots", *International Journal of Robotics Research,* vol. 5, no. 1, pp 90-98, 1986.

[6] J. M. Lee "Tangible Tele-operation of a Mobile Robot with Force-reflection", Sino-Korea Symposium on Intelligent System, pp. 88-93, 2003.

[7] H. Haddad, M. Khatib, S. Lacroix and R. Chatila. "Reactive Navigation in Outdoor Environments using Potential Fields", *IEEE Intl. Conference on Robotics & Automation,* pp. 1232-1237, 1998.

[8] J. L. Martinez, A. Pozo-Ruz, "Object Following and Obstacle Avoidance Using a Laser Scanner in the Outdoor Mobile Robot Auriga-Alpa", *IEEE/RSJ Intl. Conference on Intelligent Robotics & Systems,* pp. 204-209, 1998.

[9] 전풍우, 정 슬, "장님 이동로봇의 힘제어 : 분석, 시뮬레이션 및 실험", 제어· 자동화· 시스템공학회 논문지, 제9권 10호, pp. 798-807, 2003년 10월.

[10] M. Raibert and J. Craig, "Hybrid Position/Force Control of Manipulators", *ASME Journal. of Dynamic Systems, Measurements, and Control,* vol. 102, pp. 126-133, 1981.

[11] N. Hogan, "Impedance Control : An Approach to Manipulator, Part i, ii, iii", *ASME Journal of Dynamics Systems, Measurements, and Control,* vol. 3, pp. 1-24, 1985.

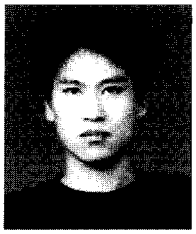
[12] 조현택, 정 슬, "속도추정 기반의 2 자유도 도립진자의 안정화를 위한 입력보상 방식의 분산 신경망 제어기에 관한 실험적 연구", *Journal of Control, Automation, and Systems Engineering,* 2004.

[13] 정 슬, "인공지능시스템" 충남대학교 출판부 2004.

[14] 정 슬, "로봇 공학: Matlab 및 SIMULINK 응용" 충남대학교 출판부, 2004.

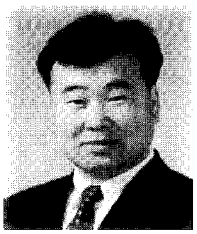
[15] N. Sarkar, X. Yun, and V. Kumar, "Control of mechanical systems with rolling constraints: Application to dynamic control of mobile robots," *International Journal of Robotics Research,* vol. 31, pp. 55-69, February, 1994.

[16] 황경훈, "Dynamics Obstacle Avoidance using Fuzzy and AGPM for Trap Recovery of Mobile Robot", 성균관대 석사 학위 논문 2002.



장 은 수

1976년 2월16일생. 2002년 2월 경일대학교 제어계측공학과 학사 졸업. 현재 충남대학교 메카트로닉스공학과 석사과정. 관심분야는 이동로봇의 장애물 회피 알고리즘, 임베디드 리눅스 시스템.



정 슬

1964 년 9 월 11 일생. 1988 년 미국 웨인 주립대 전기 및 컴퓨터 공학과 졸업. 1991 년 미국 캘리포니아대 전기공학과 석사. 동대학 박사. 1997 년~현재 충남대학교 메카트로닉스공학과 부교수. 관심분야는 지능 제어 알고리즘 및 하드웨어 구현, 로봇의 인간의 지능적인 상호 작용, 필드 로봇 시스템.