論文 한국정밀공학회지 제10권 제3호 1993.9
Journal of the Korean Society of Precision Engineering Vol. 10. No.3 Sep. 1993.

신경회로망을 이용한 이동로보트의 위치 추정에 관한 연구

김 재 희*, 이 재 철*, 조 형 석**

A Study on Estimation of a Mobile Robot's Position Using Neural Network

Jae-H Kim*, Jae-C Lee*, Hyung-S Cho**

ABSTRACT

For navigation of a mobile robot, it is one of the essential tasks to find out its current position. Dead reckonining is the most frequently used method to estimate its position. However conventional dead reckoner is prone to give us false information on the robot position especially when the wheels are slipping. This paper proposes an improved dead reckoning scheme using neural networks. The network detects the instance of wheel slipping and estimates the linear velocity of the wheel; thus it calculates current position and heading angle of a mobile robot. The structure and variables of the neural network are chosen in consideration of slip motion characteristics. A series of experiments are performed to train the networks and to investigate the performance of the improved dead reckoning system.

Key Words : Position Estimation(위치추정), Mobile Robot(이동로보트), Dead Reckoning(사산법), Neural Network(신경회로망), Wheel Slipping(바퀴미끄럼)

1. 서 론

인간의 기능을 기계로 하여금 수행시키고자 하는 노력은 실로 오래전부터 경주되어 왔다. 고정된 위치에서 주어진 과업을 수행하는 일련의 로보트를 비롯하여 이제는 그 로보트가 공간을 이동하면서 직무를 수행토록 하기 위하여 많은 연구가 수행되고 있다. 평지나 굴곡지형 또는 수중에서 작업할 수 있는 로보트 및 우주 공간이나 지구 아닌 다른 혹성에서 인간의 역할을 대신하는 로보트 등 그 응용분야는 실로 다양하다.

본 연구에서는 주로 평지를 주행하는 이동로보트에 관해서 살펴 보기로 한다. 이러한 이동로보트는 공장자동화를 위한 무인 반송차(automated guidance vehicle), 사무빌딩 내에서의 문서전달로보트(document delivery robot), 병원에서의 간호로보트(nursing robot), 방문객이나 맹인을 안내하는 로보트, 원자력발전소의 오염지역 내의 이상 유무를 감시하는 감시용로보트 등 그 응용분야에 따라 여러가지 임무를 띠고 있다. 이와같은 이동로보트가 그 임무를 수행하기 위하여 주어진 경로를 정확하게 주행하려면 여러가지 요소기술이 필요하지만 그중에서도 특히 로보트의 현위치를 스스로 추정하는 일은 필수적이라 할 수 있다.

이동로보트가 스스로의 위치를 알아내는 방법은 여러가지가 사용되고 있다. 그중에서 차륜형 이동로보트의 경우 바퀴의 회전량을 감지하여 현재의 위치를 환산하는 사산법(dead reckoning : DR)은 그 알고리즘이 간단하고 구현하기 편리하며 빠른 시간내에 처리할 수 있어서 가장 널리 사용되고 있다[1-5]. Myer[6]은 아나로그 컴

* 한국원자력연구소 핵인공지능연구실(정회원)
** 한국과학기술원 정밀공학과(정회원)

퓨터회로를 만들어 이 방법을 구현한 바 있으며 Tsukuba 대학에서 만든 Yamabico 이동로보트도 역시 휠 인코더를 장착하여 이 사산법에 의해 로보트의 현위치를 추정하였다[7].

그러나 여러가지 오차 요인 때문에 이 사산법으로 로보트의 정확한 현위치를 구하기에는 어려움이 있다. 이러한 오차를 발생케 하는 요인은 크게 두가지로 나눌 수 있다. 즉 바이어스 형태(bias type) 오차와 랜덤(random) 오차가 그것이다. 바이어스 형태의 오차들은 규칙적이고 반복적이이서 일관성 있는 위치 추정 오차를 발생시킨다. 그 한예가 바퀴의 마모 때문에 바퀴의 반경이 변해서 발생하는 오차이다. 이 오차는 바퀴재질의 탄성을 무시한다면 변수추정기(parameter idenifier)를 도입하거나 로보트 보정(calibration)을 통하여 보상될 수 있을 것이다. 랜덤 오차는 영 평균(zero mean) 특성을 갖는 불규칙한 오차로서 바닥의 조도(roughness)가 낮아서 발생하는 바퀴의 미끄럼 등이 이에 속한다. 즉 바닥과 바퀴간의 마찰 계수가 매우 낮아 최소한계를 넘어서면 미끄럼이 발생하게 된다. 이 오차는 거의 측정 불가능하여 사산기(dead-reckoner)의 성능을 매우 저하시키게 된다. 이러한 오차외에도 디지탈 컴퓨터를 사용할 때 quantization error가 발생하는데 이는 샘플링 시간(sampling time), data truncation 정도 등에 따라 그 영향이 좌우된다. 또 바닥이 기울어 바퀴가 굴곡 있는 곳을 지나게 되면 바퀴의 이동거리와 지도상의 이동거리가 다르게 되어 오차가 발생하게 될 것이다. 이상에서 살펴본 바와 같이 사산법은 다양한 오차 요인을 내재하고 있다. 그런데 바이어스 형태 오차는 변수추정기 등을 통하여 보상될 수 있을 것으로 보여지며 경사로 인한 오차도 경사계를 사용하여 보상이 가능할 것으로 생각하여 본 연구에서는 다루지 않는다. 본 연구에서는 바퀴의 미끄럼 때문에 발생하는 오차를 보상하는 방법을 대하여만 생각하기로 한다.

바퀴의 미끄럼을 탐지하고 그것을 보상하여 로보트의 현 위치를 추정하는 방법은 몇가지로 생각할 수 있다. 그 하나는 이동로보트의 하부에 여러개의 바퀴와 휠 인코더를 설치하여 각 인코더로부터 얻은 정보를 비교 분석하여 미끄럼으로 인해 잘못 측정된 정보를 버리고 정상적인 정보만을 가지고 로보트의 위치를 추정하는 방법이다. 그러나 바퀴 전부가 미끄러지는 경우에는 옳은 정보를 구별해 낼 수 없고 소형로보트의 경우 많은 바퀴와 인코더를 설치하기가 어려운 단점이 있다. 이 방법외에 적응학습 알고리즘(adaptive learning algorithm)[8]이 제안된 바 있다. 이 방법은 로보트의 절대 위치를 측정할 수 있는 센서(sensor)를 사용하여 사산기에 의한 로보트 현위치와 비교해 봄으로서 미끄럼을 탐지하고 보상하는 방법이다. 그러나 이 방법은 바퀴 미끄럼이 발생한 즉시 감지하지 않고 간헐적으로 설치되어 있는 랜드마크(landmark)가 측정 범위안에 들어와서 절대위치 센서가 측정할 수 있을 때까지 기다려야하므로 비효율적이다.

본 연구에서는 신경회로망을 이용하여 바퀴의 미끄럼을 탐지하고 보상함으로서 종래의 사산기에 의한 위치추정을 개선하였다. 이동로보트의 미끄럼 현상을 고려하여 신경회로망의 입출력과 구조를 정하였으며, 일련의 실험데이터를 사용하여 신경회로망을 훈련시켰다. 이러한 훈련된 신경회로망 추정 시스템을 경험하지 않은 경우에 대하여 적용시켜 그 결과를 고찰함으로써 제안된 시스템의 성능과 효능을 평가하였다.

2. 시스템 개요

본 연구에서 사용한 이동로보트 LCAR와 구동 계통을 소개하기로 한다. 그림 1에서 보는 바와 같이 KAIST 자동화 실험에서 제작한 LCAR(Laboratory for Control and Automation Robot)는 두개의 구동

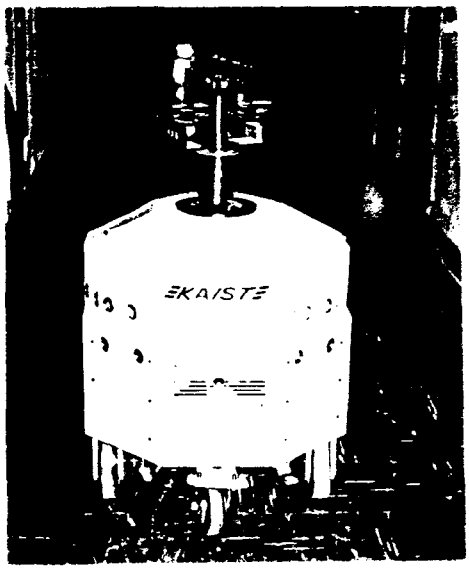


Fig. 1 The mobile robot LCAR

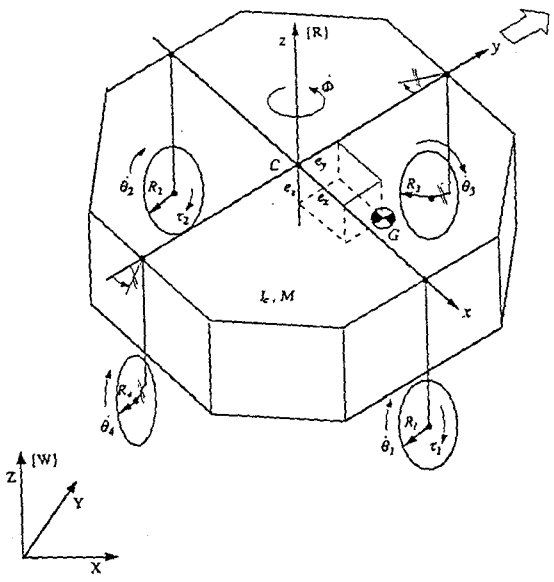


Fig. 2 The model of the mobile robot, LCAR

바퀴가 양쪽에 설치되어 있고 앞뒤에 caster wheel이 하나씩 달려 있다. 두개의 구동바퀴의 속도차이에 의해 로보트는 앞뒤로 움직이고 방향을 변화시키며 주행할 수 있다. DC servo motor와 콘트롤러가 바퀴를 제어하고 각각에 인코더가 부착되어 이로부터 바퀴의 회전량을 측정한다. 주 컴퓨터는 IBM AT가 사용되고 모든 관리제어를 담당한다. 로보트의 중량(M)은 112kg이고 관성모멘트(I_c)는 7.1kgm^{2}이다. 그림 2에서 보는 바와 같이 로보트 좌표계 {R}은 로보트 몸체 중심에 놓고, 기준 좌표계 {W}를 X-Y-Z로 표시하기로 한다. 로보트의 무게 중심은 {R} 좌표계상의 (e_x, e_y, e_z)에 편심되어 위치한다. 로보트 몸체의 방향각(heading angle)을 ϕ로 표시하고 기준 좌표계상에서 로보트의 현위치를 (X, Y)로 표시하기로 한다. 각각의 바퀴의 회전속도를 $\dot{\theta}_i$로 표시하고 여기서 첨자 i는 각 바퀴를 나타내며 오른쪽 구동 바퀴를 1, 왼쪽 바퀴를 2로 정한다. 바퀴의 선속도를 $\dot{y}_i$로 표시하고 이는 바퀴 회전축이 로보트 몸체의 y축 방향으로 직선 운동하는 속도를 나타낸다.

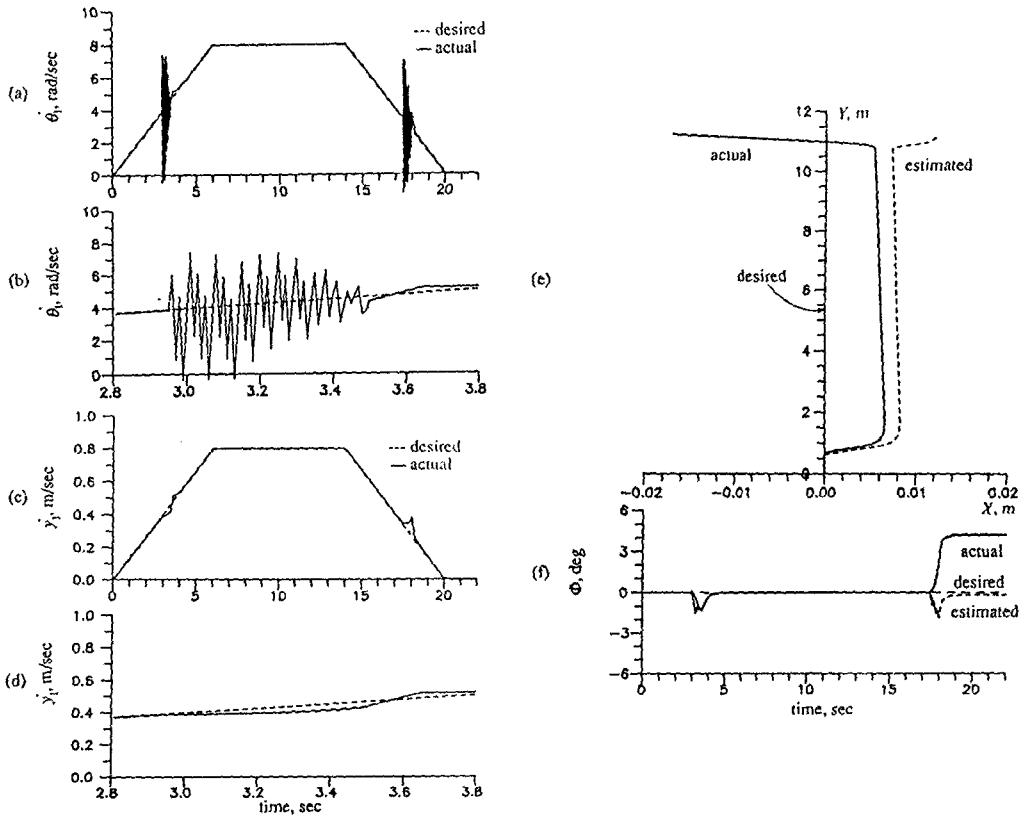


Fig. 3 Slip motion of the right wheel : straight line trajectory

3. 신경회로망을 이용한 미끄럼 보상기

이동로보트의 바퀴를 일반적인 PID 제어 하였을 경우 평상시 좋은 제어 성능을 보이던 이 제어 시스템이 바닥이 매우 미끄러운 지역에 당도하여서는 바퀴의 평시와는 다소 다른 특성을 보이게 된다. 즉 그림 3에서 보는 바와 같이 바퀴를 사다리꼴의 형태의 속도 profile을 따라서 가속, 등속, 감속시킬 경우 미끄럼이 없는 구간에서는 목적치를 잘 추종하지만 미끄러운 구간에서는 바퀴의 회전속도가 매우 불균일하게 된다. 이 그림은 2.5~4초, 9~11초, 17.5~18.5초 구간에 미끄러운 구간이 있다고 가정하고 이동로보트의 미끄럼 모델을 사용하여 시뮬레이션 한 결과인데[9, 10], 가속과 감속 구간에서 미끄럼이 발생하는 현상을 보여 주고 있다. 이론적으로 바퀴의 미끄럼은 바퀴에 가해진 구동 토오크와 바퀴와 바닥간의 마찰계수에 크게 관계 되는데, 등속구간의 경우 제어기에서 바퀴 구동 토오크가 매우 적게 산출되어 미끄럼이 거의 발생되지 않는다. 그림 3(b)는 (a)를 확대하여 보여주는 그림이다. 한편 미끄럼이 발생하는 구간에서도 바퀴의 선속도는 로보트 전체의 관성 때문에 완만한 변화를 보이게 된다(그림 3(c)). 일반적으로 가속구간에서 미끄러질 때는 참조 입력(reference input)을 따라가지 못하고, 감속 구간에서는 미처 감속이 되지 못하는 경향을 보여준다. 그림 3(d)는 (c)를 확대하여 보여주는 그림이다. 이와같은 상황에서 단지 바퀴의 회전속도만을 측정하여 사산법에 의해 로보트의 위치를 산출하면 실제 로보트의 위치와 많이 달라지게 된다. 그림 3(e)는 사산법에 의한 추정 궤적과 실제 궤적을 비교하여 보여주고 있다. 그림 3(f)는 로보트의 방향각에 대하여 추정치와 실제치를 보여준다. 본 연구에서는 이와같은 현상을 배경으로 하여 미끄럼 발생을 탐지하고 그때 바퀴의 선속도를 추정하는 방법을 신경회로망을 이용하여 구현하였다.

3.1 미끄럼 탐지

위에서 살펴본 바와 같이 바퀴가 미끄러질 때의 거동은 정상적인 경우와 다름을 알 수 있다. 이것은 통상적으로 오차를 근간으로 한 제어기를 사용하였을 경우에는 거의 유사한 거동을 보여주게 된다. 그림 3에서 보는 바와 같이 바퀴의 각속도의 변화량, 즉 각가속도를 추출해 봄으로써 바퀴가 미끄러지고 있는지를 눈으로 보아 추측할 수 있다. 비슷하게 생각해 볼 때 신경회로망이 이 역할을 담당해 줄 수 있을 것으로 판단된다. 한편 미끄럼을 탐지하기 위한 또 하나의 방법으로서 바퀴의 회전가속도 θ_i에 관한 일련의 데이터를 가지고 산술적인 판별식을 적용할 수도 있을 것이다. 그러나 이 판별식은 각각의 로보트의 특성에 따라 조정해 주어야 할 것이다. 결국 신경회로망을 이용한 미끄럼 탐지 방법은 이 조정자의 역활을 신경회로망의 훈련 과정을 통하여 대치한다고 생각할 수도 있다.

그림 4에서 보는 바와 같이 미끄럼 탐지를 위한 신경회로망의 입력으로서 바퀴의 회전가속도 $\ddot{\theta}_i$과 희망회전가속도 $\ddot{\theta}_i^d$를 선정하였다. 같은 조건상에서 미끄러질 때도 $\ddot{\theta}_i^d$에 따라 다소 다른 거동을 나타내기 때문에 $\ddot{\theta}_i^d$도 입력변수로 두기로 한다. 출력으로서는 slip이면 1, nonslip이면 0을 출력하도록 훈련시킨다. 그림 4에서 위첨자 *는 target value를 의미하여, ε은 target value와 신경회로망의 출력간의 오차인데 이 ε이 최소

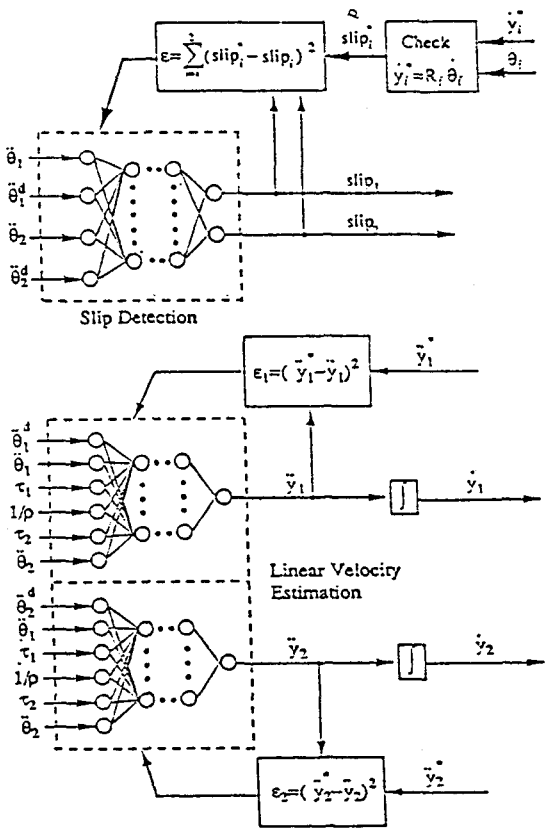


Fig. 4 Structure of neural networks for slip detection and linear velocity estimation

가 되도록 신경회로망의 가중치(weight)들을 조정해 나간다. 즉 오차역전파법(Error Back Propagation)을 사용하였다. 예를들어 미끄럼 탐지 회로망에서 $\dot{y}_i$*는 신경망을 훈련시키기 위하여 별도 실험을 통하여 얻어진 바퀴의 선속도이고 $\dot{\theta}_i$는 바퀴의 회전속도인데 $\dot{y}_i$*와 $R_i\dot{\theta}_i$를 비교하여 같으면 미끄럼이 없는 경우이므로 $Slip_i$*를 0으로 출력하고, 다르면 미끄럼이 발생한 경우이므로 $Slip_i$*를 1로 출력한다. 그래서 $Slip_i$*와 실제 신경회로망의 출력 $Slip_i$간의 오차 ε이 최소가 되도록 신경망의 가중치(weight)를 수정시킨다. 이와같은 방법으로 훈련이 모두 끝나면 실제로 $\ddot{\theta}_i$과 $\ddot{\theta}_i^d$의 수치를 입력하여 $Slip_i$를 출력시킨다.

3.2 선속도 추정 시스템

미끄럼이 발생했을 때 사산법을 사용하여 정확한 로보트의 위치를 산출하려면 바퀴의 선속도를 추정해 내야만 한다. 그러나 미끄럼이 발생했을 때 바퀴의 선속도는 바퀴의 회전속도와 관련이 없다. 문헌[9]에서 미끄럼 현상 모델링을 통하여 얻어진 내용을 참조하면 바퀴의 선가속도 $\ddot{y}_i$는 바퀴의 구동 토오크 τ_i, 바퀴의 회전 가속도 $\ddot{\theta}_i$와 함수관계를 가짐을 알 수 있다. 그 이외에 실제 로봇을 주행시켰을 때 희망 회전가속도 $\ddot{\theta}_i^d$와 로보트의 주행 회전반경 ρ도 선가속도에 영향을 보여준다. 그리하여 선속도 추정 회로망의 출력 $\ddot{y}_i$을 내기 위한 입력으로서 τ_1, τ_2, $\ddot{\theta}_1$, $\ddot{\theta}_2$, $\ddot{\theta}_i^d$와 $1/\rho$를 선정한다. 이 신경회로망을 통하여 $\ddot{y}_i$를 출력한뒤 이 값을 누적 가산하여 바퀴의 선속도 $\dot{y}_i$를 산출한다. 움직이는 물체의 미끄럼 문제는 그 계통의 동특성과 관련되어 힘(force) 또는 가속도와 유관하다. 따라서 이 신경회로망의 입력과 출력 변수도 힘 또는 가속도 단위라야 상관 관계를 가질 수 있다. 그러므로 선속도 $\dot{y}_i$를 추정하려면 일단 신경회로망을 통하여 선가속도 $\ddot{y}_i$를 추정한 후 적분하여 선속도 $\dot{y}_i$를 산출하여야 좋은 추정 성능을 나타낼 수 있다.

3.3 미끄럼 보상을 통한 개선된 사산법

이상에서 미끄럼 탐지 신경회로망과 선속도 추정 신경망을 설명하였다. 이 두 신경망을 사용하여 실제 로보트 주행시의 위치 추정 방법을 생각해 보자. 그림 5에서 보는 바와 같이 slip이 일어나지 않을 경우의 바퀴의 선속도 $\dot{y}_i$은 바퀴의 반경 R_i에 바퀴의 회전속도 $\dot{\theta}_i$을 곱한값 $\dot{y}^{ns}$이 된다. 반면 미끄럼 발생했을 때는 선속도 추정 회로망이 작동하여 $\dot{y}_i$를 출력해 낸다. 이 시스템을 통하여 결정된 양 바퀴의 선속도 $\dot{y}_i$를 사용하여 사산기는 기준 좌표계상에서의 로보트의 현위치 X, Y와 방향각 Φ를 산출해 낸다.

4. 실험 및 고찰

전장에서 제안한 개선된 사산법을 시험하기 위하여 일련의 실험을 수행하였다. 미끄럼 모델을 이용한 컴퓨터 시뮬레이션 결과는 문헌[9]에 자세히 서술되어 있다.

4.1 실험장치

2장에서 소개한 이동로보트 LCAR를 사용하여 실험을 수행하였다. 그림 6에서 보는 바와 같이 이 실험장치는 로보트 몸체와 IBM AT 컴퓨터, SONY CCD 카메라, 이미지 프로세서, 디지탈 콘트롤러, 서보 앰프 및 DC 서보 모터로 구성되어 있다. 실험은 먼저 신경회로망을 훈련시키기 위하여 샘플 데이터를 수집하여야 하고 이

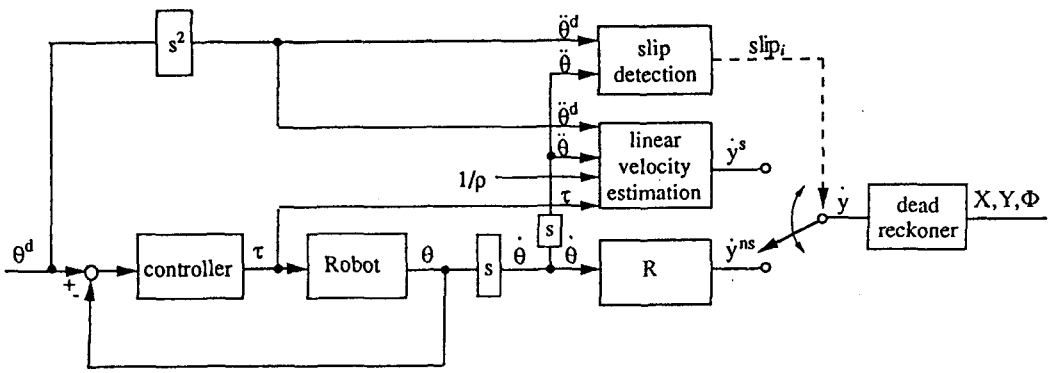


Fig. 5 Position estimation using neural network : slip detection and linear velocity estimation

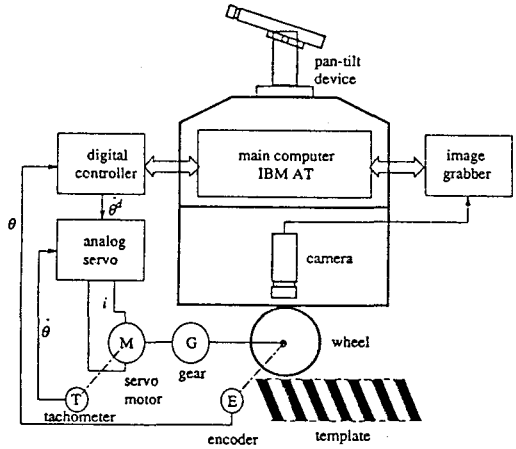
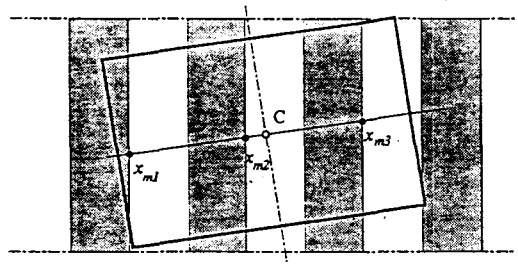
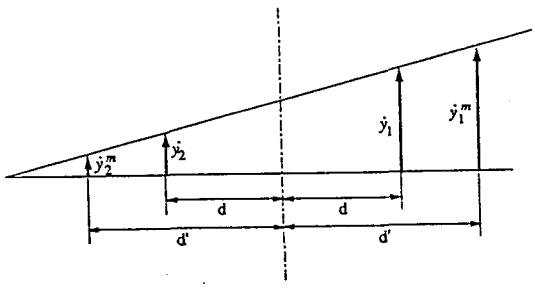


Fig. 6 Schematic diagram of the experimental system

데이터로 신경회로망을 훈련시킨 뒤, 실제 로봇을 주행시키면서 이 신경회로망 시스템을 가동하여 로보트의 현위치를 추정하는 두단계로 수행되었다. 신경회로망을 훈련시키기 위하여는 바퀴의 각속도 $\dot{\theta}_i$과 바퀴의 선속도 $\dot{y}_i$ 그리고 바퀴에 전달되는 토오크 τ_i를 측정하여야 한다. 회전 각속도는 바퀴에 부착되어 있는 인코더를 읽어서 측정하고 토오크는 PID 제어기에서 출력되는 제어



(a) displacement of the camera center



(b) geometric relation of the linear velocities

Fig. 7 Illustrations of the visual measurement of the linear velocity

입력으로 대체 하였다. 그런데 이동로보트와 같이 서서히 이동하는 물체의 선속도를 측정하기란 쉬운 일이 아니다. 여러가지 센서가 시험되었지만 정확한 속도와 가속도를 측정할 수 없었다. 스트레인 게이지 형태의 상업용 가속도계를 사용할 경우에도 이동로보트의 가속도는 실제로 매우 작은 값이어서 신뢰성 있는 측정이 불가능하였다. 그리하여 비교적 제한적인 구간에서 속도를 측정하는 한 방법으로 CCD 카메라를 사용하였다. 즉 로보트가 이동할 구간의 바닥에 일정한 간격으로 막대가 그려진 템플레이트(template)를 부착하고 로보트의 바퀴상단에 CCD 카메라를 설치한 뒤 카메라에 맺힌 영상을 실시간으로 분석함으로서 바퀴의 선속도를 계산하였다. 그림 7은 이 방법을 간략히 설명해 주고 있다. 여기서 카메라의 광축(optical axis)은 실제로 선속도를 측정하고자 하는 점의 위치보다 바깥으로 나와 있으므로 기하학적으로 환산해 주어야 한다. 즉,

$$\dot{y}_1 = \frac{\dot{y}_1^m + \dot{y}_2^m}{2} + \frac{d}{2d'}(\dot{y}_1^m - \dot{y}_2^m) \qquad (1)$$

$$\dot{y}_2 = \frac{\dot{y}_1^m + \dot{y}_2^m}{2} - \frac{d}{2d'}(\dot{y}_1^m - \dot{y}_2^m) \qquad (2)$$

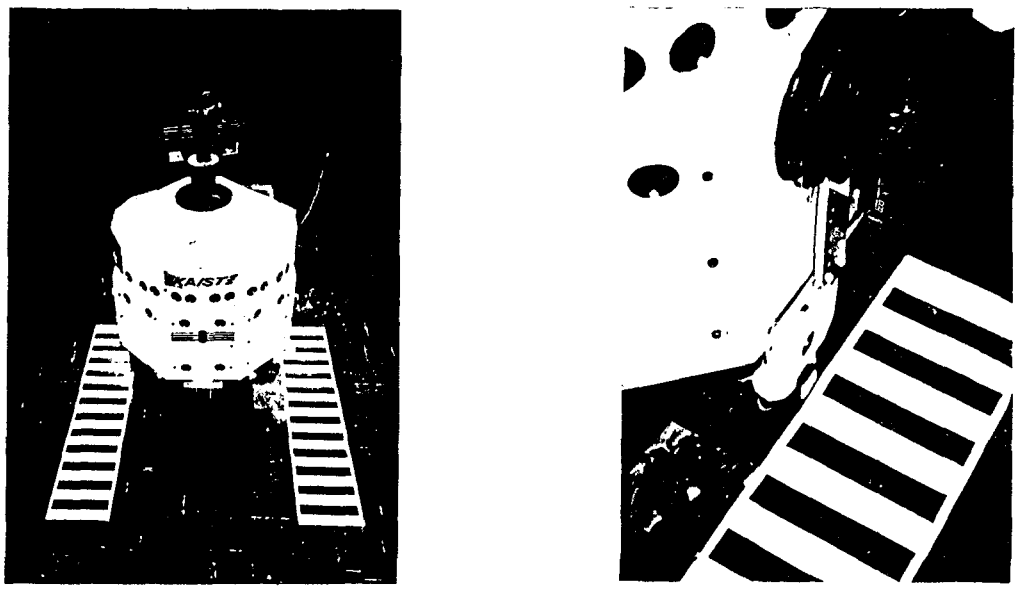
여기서 $\dot{y}^m$은 카메라에 의해 측정된 선속도이고 d'은 카메라 중심축과 로보트 몸체의 중심간의 거리이고 d는 바퀴 중심선과 로보트 몸체 중심간의 거리이다. 바닥의 템플레이트를 조명하기 위하여 할로겐 램프(12V, 50W)를 바퀴 근처에 장착하였다. 미끄러운 구간은 윤활유 위에 비닐을 덮어 인위적으로 설치하였다.

4.2 실험방법 및 절차

실험은 두 단계에 걸쳐 수행되었다. 즉 신경망을 훈련시키기 위하여 샘플 패턴(sample pattern)을 수집하는 과정과 이 데이터로 신경망을 훈련시킨 뒤 미끄럼 탐지와 선속도 보상을 수행하여 로보트가 실제로 주행하는 동안 로보트의 현 위치를 추정하는 과정으로 이루어져 있다. 실험중 로보트가 미리 설치된 템플레이트를 크게 벗어나면 카메라 시스템이 실제로 선속도를 측정할 수 없으므로 실험은 비교적 제한적으로 실시되었다. 표 1에서 보는 바와 같이 로보트의 주행경로, 최대 선속도, 가속정도와 미끄럼 구간(왼쪽 바퀴 혹은 오른쪽 바퀴)을 18가지로 달리하면서, 매 주행마다 20쌍(set)의 신경회로망 입출력 표본 데이터들을 수집하였다. 신

Table 1. Experimental condition for network training

no	path	$\dot{y}_{max}$ [m/sec]	$\ddot{y}_{max}$ [m/sec^2]	accel. time t_r [sec]	slipping wheel
1	straight	0.3	0.100	3	right
2	〃	0.3	0.060	5	left
3	〃	0.3	0.060	5	both
4	〃	0.2	0.050	4	right
5	〃	0.2	0.067	3	left
6	〃	0.2	0.067	3	both
7	〃	0.1	0.020	5	right
8	〃	0.1	0.033	3	right
9	〃	0.1	0.025	4	left
10	〃	0.1	0.050	2	left
11	〃	0.1	0.025	4	both
12	〃	0.1	0.033	3	both
13	curve (ρ=10m)	0.1	0.020	5	right
14	〃	0.1	0.025	4	left
15	〃	0.1	0.025	4	both
16	curve (ρ=10m)	0.1	0.025	4	right
17	〃	0.1	0.033	3	left
18	〃	0.1	0.020	5	both



(a) overview (b) velocity measurement

Fig. 8 Experimental setup

경회로망을 학습시키는데 사용하였다. 로보트의 속도는 사다리꼴 형태를 유지하고 로보트의 속도는 최대 0.3m/sec 이내에서 실험하였다. 이것은 CCD 카메라의 노출 특성 때문에 급히 이동하면 영상이 희미해지기 때문이다. 그림 8은 실험장치의 모습을 보여준다.

미끄럼 탐지와 선속도 추정 회로망으로는 다층 퍼셉트론(multilayered perceptron)[11]이 사용되었다. 이 신경망의 구조와 학습조건은 여러가지 경우에 대하여 시험한 결과 비교적 수렴성이 양호한 경우를 택하였다. 미끄럼 탐지 신경회로망은 한개의 hidden layer와 세개의 노드(node)로 하였으며 학습률(learning rate) η는 0.1, 모멘텀율(momentum rate) α은 0.2로 하여 학습시켰다. 한편 선속도 추정 회로망은 두개의 hidden layer와 첫층에는 3개, 둘째층은 2개의 노드로 하였으며 학습률 η는 0.1, α은 0.2로 하였다. 이 결과 미끄럼 탐지 회로망은 40회의 iteration 후 0.05의 오차 이내로 수렴하였으며, 선속도 추정 회로망은 50회의 iteration 후 0.001의 오차로 수렴하여 더 이상 변화가 없었다.

이 시스템을 실제로 구현함에 있어서 바퀴의 각가속도 $\ddot{\theta}_i$와 선가속도 $\ddot{y}_i$를 측정하여야 하는데 휠 인코더와 CCD 카메라를 사용하는 경우 정확치가 않다. 그래서 각가속도 $\ddot{\theta}_i$을 $\dot{\theta}_i(k-1)$와 $\dot{\theta}_i(k)$로 대치하고 선가속도 $\ddot{y}_i$를 $\dot{y}_i(K-1)$와 $\dot{y}_i(k)$로부터 산출하였다. 여기서 k는 샘플링수를 의미한다. 결과적으로 미끄럼 탐지를 위한 입력변수는 $\dot{\theta}_1(k-1)$, $\dot{\theta}_1(k)$, $\ddot{\theta}_1^d$, $\dot{\theta}_2(k-1)$, $\dot{\theta}_2(k)$와 $\ddot{\theta}_2^d$가 된다. 같은 방법으로 선속도 추정 회로망의 입력변수는 $\dot{\theta}_1(k-1)$, $\dot{\theta}_1(k)$, $\dot{\theta}_2(k-1)$, $\dot{\theta}_2(k)$, $\ddot{\theta}_1^d$, $\ddot{\theta}_2^d$, τ_1, τ_2와 $1/\rho$가 된다.

4.3 결과 및 고찰

그림 9는 실제 로보트가 미끄러질 때 바퀴의 거동을 표시하였다. 이 경우 로보트는 최대 0.3m/sec 속도와 최대 가속도 0.1m/sec^{2}로 직선운동을 할 때 왼쪽 바퀴가 미끄러진 경우에 대한 결과이다. 이 실험에서 각 실험치는 매 250msec마다 측정된 값이다. 이는 사용된 영상시스템이 영상을 잡아 처리하는데 약 100msec가 걸리고 양쪽 바퀴를 처리하여야 하기 때문이다. 이처럼 샘플링 시간이 다소 길기 때문에 로보트의 운동을 제어하기가 쉽지 않은 단점이 발생한다. 그래서 feedforward term을 삽입한 PID 제어를 통하여 유연한 운동을 유도하였다. 즉 제어 입력 τ_i는

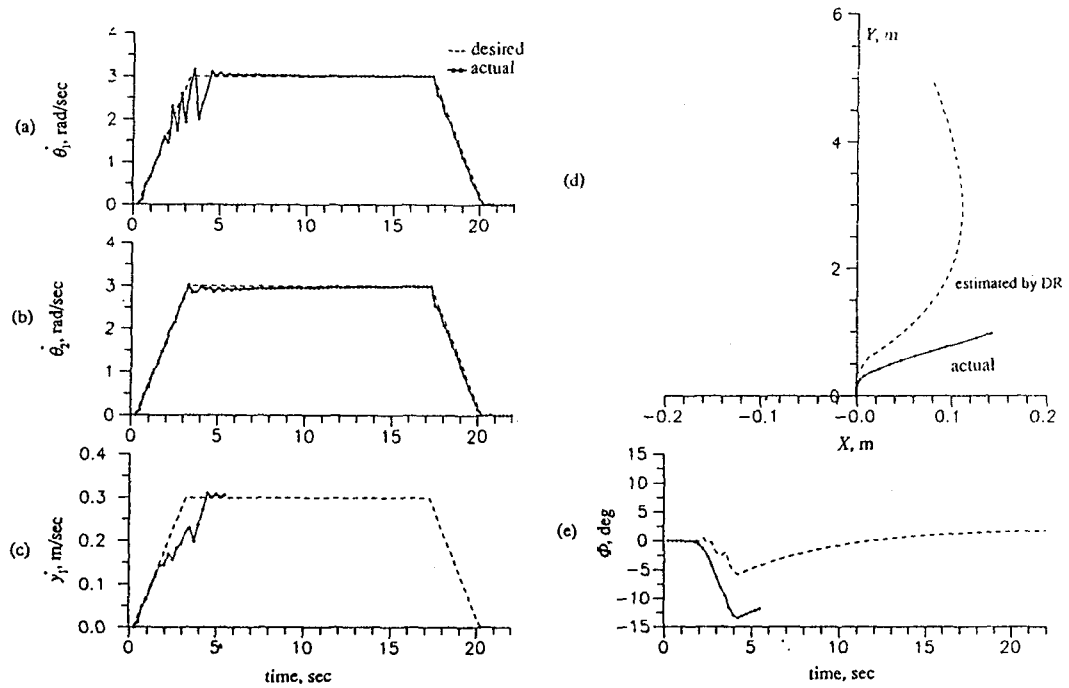


Fig. 9 Experimental phenomena of the wheel slipping

$$\tau_i = K_p(\theta_i^d - \theta_i) + K_d(\dot{\theta}_i^d - \dot{\theta}_i) + K_f\dot{\theta}_i^d \quad (3)$$

로 정하였다. 여기서 각각의 게인은 LCAR의 단순모델을 사용하여 컴퓨터 시뮬레이션을 통하여 출력반응(output response)을 살펴보고 오버슈트(overshoot)가 없고 전체 오차가 가장 적은 경우의 게인을 택하였다. 본 실험에서는 $K_p=0.32$, $K_d=1.65$, $K_f=0.2$로 하였다. 한편 실험결과 샘플링 시간이 긴 덕택으로 오히려 측정된 각속도와 선속도에는 잡음(noise)이 덜 나타났다.

그림 9(a)와 (b)는 휠 인코더로부터 측정한 바퀴의 각속도이다. 가속구간에서 바닥이 미끄러워 왼쪽 바퀴가 미끄러짐으로 인해 각속도가 불균일하게 변하고 있음을 보여주고 있다. 그림 9(c)는 CCD 카메라를 이용하여 바퀴의 선속도를 측정한 결과인데 5.5초 이후는 바닥에 미리 붙여둔 템플레이트 영역을 벗어나 카메라가 영상을 얻을 수 없어서 선속도 측정이 불가능 하였다. 이로 인해 (d)와 (e)에서도 실제(actual) 궤적이 일부만 나타나 있다. 그림 10과 11에서도 이와같은 이유로 실제 데이터가 초기에만 보여져 있다. 측정된 일련의 샘플 패턴을 사용하여 신경회로망을 훈련시킨 뒤 실제로 로보트를 이동시키면서 이 훈련된 회로망을 가동시켜 로보트의 위치를 추정하였다. 그림 10은 최대속도 0.28m/sec, 최대가속도 0.07m/sec^{2}로 직선운동 시키던 도중 왼쪽 바퀴가 미끄러진 경우이다. 미끄러지는 동안 왼쪽 바퀴의 각속도는 매우 불균일하게 나타나고 선속도는 약간 다른 형태를 보여준다. 이 경우 실제 로보트의 이동 궤적과 신경망에 의해 추정된 궤적 그리고 단순히 사산법을 사용하였을 때의 이동 궤적을 그림 10(d)에서 보여준다. 신경망에 의해 추정된 궤적이 실제 궤적에 좀 더 가깝게 나타나 있다. 그림 11은 로보트가 곡선 궤적을 이동하던중 왼쪽 바퀴가 미끄러진 경우를 보여주고 있다. 이 경우 회전 반경은 12m이고 최대속도 0.25m/sec, 최대가속도 0.071m/sec^{2}로 주행하였다.

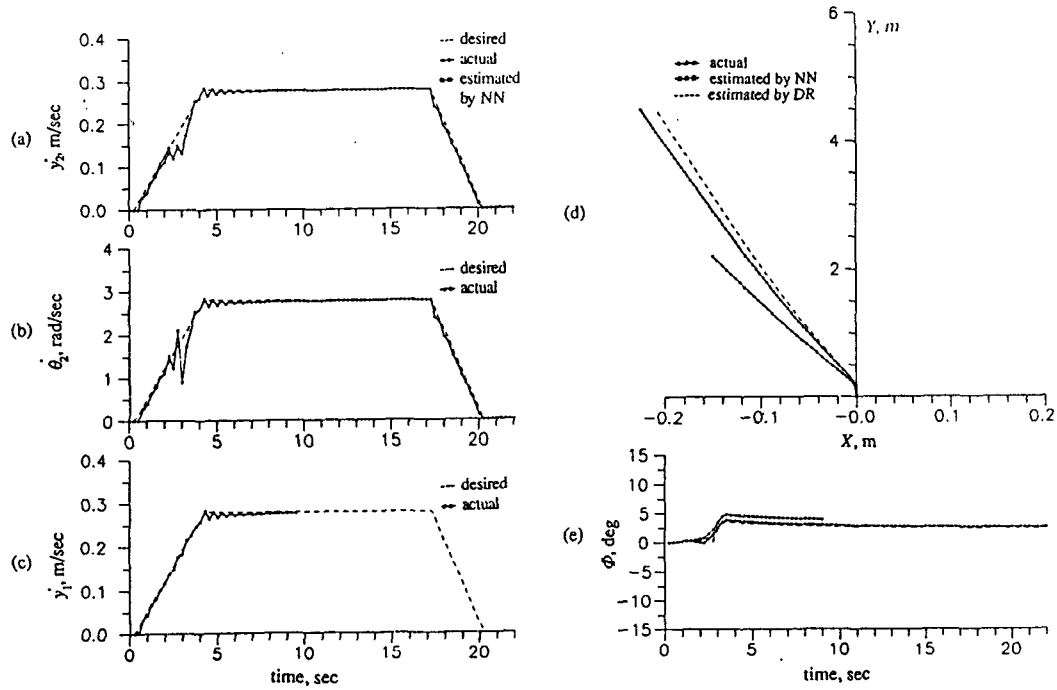


Fig. 10 Experimental results on the position estimation using trained neural network : Straight line trajectory

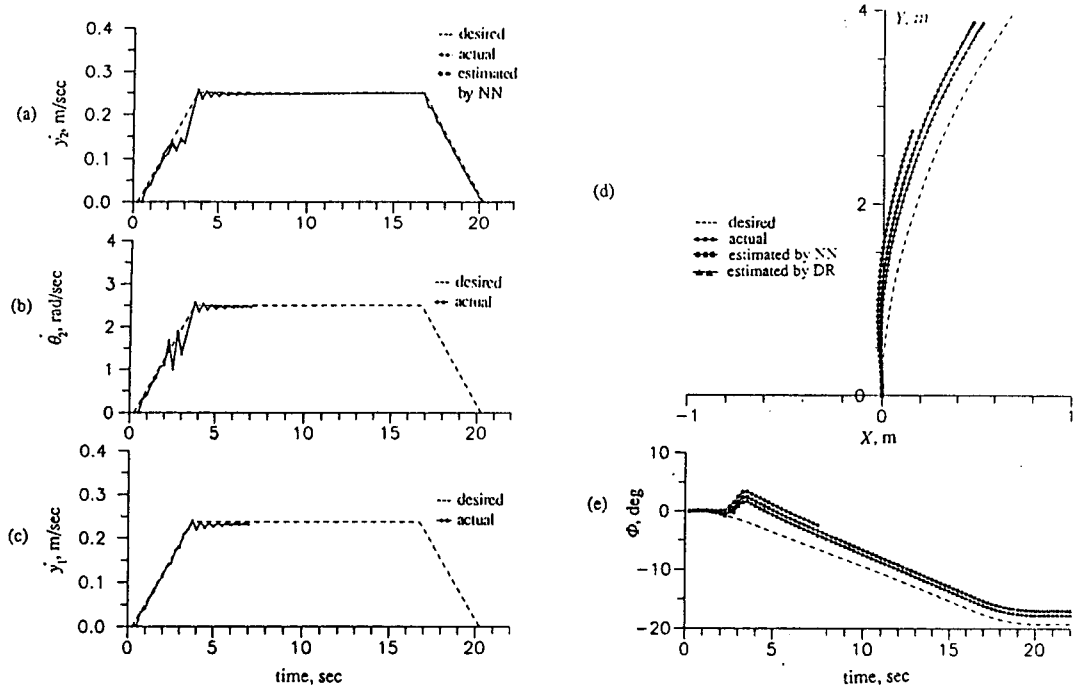


Fig. 11 Experimental results on the position estimation using trained neural network : Curved line trajectory

5. 결 론

일반적으로 사산법은 이동로보트의 위치를 산출하는데 가장 널리 사용되고 있지만 보상되어야 할 결점이 있는데 그중에서도 바퀴의 미끄럼에 의한 오차 발생은 매우 치명적이라 할 수 있다. 본 연구에서는 신경회로망을 이용하여 바퀴의 미끄럼을 탐지하고 바퀴의 선속도를 추정하는 방법을 제안하였다. 이 연구를 통하여 얻어진 결과를 정리하면 다음과 같다.

첫째, 일반적으로 상용센서를 사용하여 서서히 움직이는 로보트의 선속도를 측정하기는 매우 어렵다. 본 실험에서 바닥에 설치한 마크의 영상을 이용한 로보트 선속도 측정은 제한적이기는 하지만 소기의 목적을 달성할 수 있는 한 방법임이 실증되었다.

둘째, 제안된 방법을 실제 로보트에 구현함에 있어 입출력 파라미터를 정확하게 측정하기는 쉽지 않다. 그러나 신경회로망이 지닌 고유한 오차 포용능력(fault tolerance)으로 인하여 샘플 패턴의 부정확도를 잘 극복하여 기존의 결과보다 향상된 추정 성능을 보여준다.

참고문헌

1. T. Tsumura, N. Fujiwara, T. Shirakawa and M. Hashimoto, "An Experimental System for Automatic Guidance of Roboted Vehicle Following the Route Stored in Memory", Proc. 11th Int'l Symposium on Industrial Robots. Tokyo Japan, pp. 187~194 (Oct. 1991).
2. T. Hongo, H. Arkawa, G. Sugimoto, K. Tange and Y. Yamamoto, "An Automatic Guidance System of a Self-Controlled Vehicle", IEEE Trans. Industrial Electronics, Vol. IE-34, No. 1, pp. 5~10 (Feb. 1987).
3. H. P. Moravec, "The Stanford cart and the CMU rover", Proc. IEEE, Vol. 71, No. 7, pp. 872~884 (1983).
4. M. Julliere, L. Marce and H. Place, "A guidance system for a mobile robot", Proc.

13th Int'l Symposium on Industrial Robots and Robot 7, Tokyo Japan, Vol. 2, pp. 58~68 (Apr. 1983).

5. A. R. Johnston, T. Assefi and L. Y. Lai, "Automated vehicle guidance using discrete reference markers", IEEE Trans. Vehicular Technology, Vol. VT-28, No. 1, pp. 95~106 (1979).
6. J. H. Myer, "VEPOL-A Vehicular Planimetric Dead-Reckoning Computer", IEEE Transaction on Vehicular Technology, Vol. 20, No. 2 (Aug. 1971).
7. S. Yuta, Y. Kanayama, T. Yajima and S. Shimmura, "An Implementation of MICHI-A Locomotion Command System for Intelligent Mobile Robots", Proc. Int'l Conf. on Advanced Robotics, Tokyo Japan, pp. 127~134 (Sept. 1985).
8. S. R. Lee, B. Lapkin and S. Dickerson, "AGV guidance : Position estimation, control and adaptation," MHRC Fall Monitor's Meeting, Atlanta, USA (Oct. 1988)
9. Jae H. Kim, "A Study on the Position Estimation of a Wheeled Mobile Robot", Ph. D. Dissertation, DPE86815, KAIST (Aug. 1992).
10. Jae H. Kim and Hyung S. Cho, "An Improved Dead Reckoning Scheme for a Mobile Robot using Neural Network", MECHATRONICS, Vol. 3, No. 4 (1993).
11. Y. H. Pao, Adaptive Pattern Recognition and Neural Networks. Addison-Wesley, New York (1989).