'93 한국자동제어학술회의논문집 (1993.10.20~22)

적응 제어기를 이용한 자율 운반체 제어

이기성, 신동호°
홍익대학교 공과대학 전기·제어공학과

Control of a Mobile Robot using a Self-tuning Controller

Keeseong Lee, Dong Ho Shin°
Department of Electrical and Control Engineering, Hong-Ik University

ABSTRACT

The control of the motion of a mobile robot is studied. The driving and steering motor assembly is located in the front of the mobile robot. The position of the mobile robot is determined by the steering angle and driving distance. For the controller design, a time-series multivariate model of the autoregressive exogenous (ARX) type is used to describe the input-output relation. The discounted least square method is used to estimate parameters of the time-series model. A self-tuning controller is so designed that the position of the center of the mobile robot track the given trajectory. Simulation result controlled by a self-tuning controller is presented to illustrate the approach.

1. 서론

자율 운반체의 운동 제어에 대한 연구를 하였다. 자율 운반체는 운반체 내부에 있는 Battery로 부터 얻어진 에너지를 제어기에 의해서 2개 이상의 Motor에 조절 공급됨으로써 원하는 위치로 Guide Line의 도움없이 이동이 가능하도록 설계 되어 있다. 자율 운반체는 바퀴에 설치되어 있는 타코메터나 인코더 등의 도움을 받아 Dead-reckoning 방법에 의해 현재의 위치를 추적 할수 있고 여러가지 센서 (CCD Camera, Infrared Sensor, Ultrasound Sensor 등)의 도움으로 주위의 방해물을 피한후 원하는 Trajectory를 따라 움직일수도 있다. 이 자율 주행체는 단독으로 사용이 될 뿐아니라 본체위에 Flexible Manipulator를 설치하여 지능적인 부품 수송(Station과 Station 사이) 및 Assembly 작업(Station 내에서)을 수행할 수 있다. 이러한 자율 운반체는 유연한 생산 관리 시스템으로써 생산성 향상과 물류 처리 공정의 유연성을 제공할수 있다. 자율 운반체의 이동을 위해서 여기서 고려한 시스템에는 앞쪽에 2개의 모터, 구동 모터와 조향 모터, 가 한개의 Assembly에 설치 되어 구동과 조향을 동시에 할수 있다. 한편 자율 운반체의 뒷쪽에는 구동은 하지 않는 2개의 Castor가 단지 무게 중심을 맞추기 위해서 설치 되어 있다.

기존의 제어기는 제어하려는 물체의 운동 방정식을 알고 있을 때만 가능하다. 그러나 자율 운반체의 운동 방정식의 경우 비선형을 나타냄으로써 복잡하고 또한 운동 방정식을 정확히 구하지 못하는 경우가 많다. 제어기로 널리 사용이 되는 PID(Proportional, Integral, Differential) 제어의 경우 사용이 되는 이득의 값은 한개의 값으로 고정이 되어 있다. 이러한 고정된 이득은 외부 환경의 변화에 대응 할수 없게 되어 많은 경우 제어를 실패 하게 된다. 반면, 여기서 제안하는 적응 제어기의 경우 [1,2,3,4,7] 자율 운반체의 운동방정식을 모르거나 부분적으로 아는 경우에도 사용이 가능하고 외부 환경의 변화(취급하려는 무게의 변화 및 이동하는 바닥의 기울기 변화 등)에 대해서도 제어를 수행 할수 있는 적응 제어를 사용하였다. 자율 운반체의 운동 방정식이 유도되고 적응 제어기를 이용한 시뮬레이션 결과도 보여 준다.

2. 시스템 개요

2.1 시스템 구조

Mobile Robot는 Robot 내부에 있는 Battery에서 얻어진 에너지를 Motor에 조절 공급 됨으로써 구동이 된다. Robot 앞쪽에 위치한 구동파트는 구동 Motor와 조향 Motor로써 구성이 되어 있다. Robot 뒷쪽은 구동을 하지 않으며 단지 무게 중심을 맞추기 위해서 설치되어 있다.

2.2 적응 제어기 (Self-tuning Controller)

기존의 Mobile Robot의 제어기는 제어 하려는 물체의 운동 방정식을 알고 있을 때만 가능 하다. 그러나 Mobile Robot의 구조는 Mobile Robot의 운동 방정식이 비선형을 나타냄으로써 복잡하고 대부분의 경우 운동 방정식을 정확히 구하지 못하는 경우가 많다. 또한 널리 사용이 되는 PID 제어의 경우 사용이 되는 이득의 값은 고정이 되어 있다. 이러한 고정된 이득은 외부 환경의 변화에 대응 할수 없게 되어 많은경우 제어를 실패하게 된다. 반면 여기서 제안하는 적응 제어기의 경우 Mobile Robot의 운동방정식을 모르거나 부분적으로 아는 경우에도 사용 할수 있고 외부 환경의 변화(취급 하려는 물체의 변화 및 Floor의 기울기 변화 등)에 대해서도 제어를 수행 할수 있다. 그림 1에서는 적응 제어기를 보여준다. 적응 제어기는 두 부분으로 구성이 된다. 첫 부분은 Parameter 계산부분이고 다음 부분은 제어 부분이다. Parameter 계산부분은 Discounted Least Square Estimation 방법에 의해서 시스템의 Parameter를 계산하고 제어부분은 Estimation된 Parameter를 이용하여 Self-tuning Control Algorithm에 의해 제어가 수행

이 된다. 많은 Self-tuning 제어의 목적은 특정 부분 (예: Mobile Robot의 무게중심 등) 이 외부에서 주어진 Trajectory를 따라 가게 만드는 것이다.

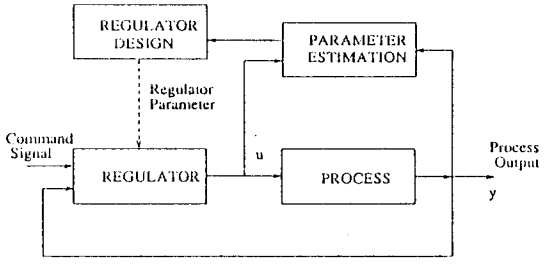

그림 1. Self-tuning Controller

Controller 설계를 위해 Autoregressive Exogenous (ARX) 모델을 이용하여 시스템의 입력-출력 관계를 나타 낼수 있다. 이 ARX 시스템 입-출력 관계는

$$q(k) = A(z^{-1})q(k) + B(z^{-1})u(k-1) + e(k) \qquad (1)$$

여기서 k는 Sampling 시간을 나타내며 1,2,3,.....등의 값을 갖는다. 또한 z는 Backward Shift Operator이며 $z^{-1}q(k) = q(k-1)$ 이다. 벡터 q(k)는 시스템의 위치나 시스템 Parameter를 나타내며 입력은 시간이 (k-1)T인 경우 u(k-1)이고 e(k)는 System Error를 나타낸다. Polynomial Matrix인 $A(z^{-1})$와 $B(z^{-1})$는 다음과 같이 정의 될수 있다.

$$A(z^{-1}) = A_1 z^{-1} + \cdots + A_n z^{-n}$$
$$B(z^{-1}) = B_0 + B_1 z^{-1} + \cdots + B_{n-1} z^{-(n-1)} \qquad (2)$$

여기서 n은 모델의 차수를 나타내는 양의 정수를 나타내고 Matrix A와 B의 Dimension은 n x n이다. Error Vector인 e(k)는 평균값은 영이고 IID(Independent, Identically Distributed) 인 Gaussian Random Noise 벡터이다.

식(2)의 모르는 Parameter를 구하기 위해서 Vector인 $\beta(k)$와 $\alpha(k)$를 다음과 같이 정의 할수 있다.

$$\beta(k)=[\beta_1(k) \cdots \beta_m(k)]=[A_1 \cdots A_n ; B_0\ B_1 \cdots B_{n-1}]'$$
$$\alpha(k)=[q'(k) \cdots q'(k-n); u'(k-1) \cdots u'(k-n)]' \qquad (3)$$

여기서 ' 는 Transpose를 나타낸다. 식(3)을 이용하여 식(1)을 다시쓰면

$$q(k) = \beta'(k)\,\alpha(k) + e(k) \qquad (4)$$

System Parameter를 결정하는 여러 방법중 Explicit Recursive Identification Method를 사용하였다. Discounted Least Square Parameter Estimation 식은 다음과 같다.

$$\beta_i(k) = \beta_i(k-1) + S(k)\alpha(k)[q_i(k) - \beta_i'(k-1)\alpha(k)]$$
$$S(k) = \frac{1}{\lambda}\left[S(k-1) - \frac{S(k-1)\alpha(k)\alpha'(k)S(k-1)}{\lambda + \alpha'(k)S(k-1)\alpha(k)}\right] \qquad (5)$$

여기서 $q_i(k)$는 i 번째 System 변수의 측정된 값이고 λ는 0에서 1사이의 값을 갖는 Forgetting Factor이다. S(k)는 (2mn x 2mn) Symmetric Matrix 이다. 실제에 있어서 Parameter의 값을 알지 못하기 때문에 그들의 값은 예측 값을 사용 한다. 이렇게 얻어진 식은 Self-tuning 제어를 구성 하는데 사용이 된다.

Self-tuning 제어기를 제작하는 경우 제일 먼저 고려해야 하는 점은 각 Mobile Robot에 대한 모델을 정의해야 한다. 각 Mobile Robot의 운동 방정식을 고려 하여 모델의 차수를 정해야한다. 이 Controller를 실현하기 위해서 LQG Self-tuning 제어 알고리즘을 이용 한다. LQG-제어 알고리즘의 목적은 Mobile Robot의 중심이 원하는 Trajectory인 $(q^d(k))$를 (d는 desired를 나타냄) 따라 가도록 하는 것이다. 이 제어기는 적당한 가중치를 가진 위치 오차와 입력 에너지로 구성된 함수를 최소화 함으로써 결정이 된다.

$$I(u(k)) = E[\|q(k+1)-q^d(k+1)\|_Q^2 + \|u(k)\|_R^2 \mid \phi(k)] \qquad (6)$$

여기서 $\|\cdot\|_R$ 은 가중치 R을 가진 Generalized Norm을 나타내며 $\|u\|_R^2$ 는 $u'Ru$이다. R과 Q는 각각 Positive Definite하고 Positive Semi-definite한 Matrix이다. Expectation 기호인 E는 Sampling 시간 kT까지의 (시간이 kT일때 포함) 자료에 의해서 정해진다. Self-tuning Controller는 시스템의 식과 가능한 입력을 만족하면서 식(6)의 값을 최소화 하는 것이다. 여기서 Parameter의 값은 예측된 값으로 대치된다.

식(6)을 최소화 하는 식은 수학적으로 다음과 같이 표현 될수 있다.

$$Ru(k) + B_0'Q[q(k+1|k) - q^d(k+1)] = 0 \qquad (7)$$

여기서 $q(k+1|k)$는 Least Square Error의 관점에서 볼때 과거의 자료에 의해서 (Sampling 시간 kT를 포함한) q(k+1)의 한단계앞 예측치이다. 식 (7) 의 해는 Self-tuning Controller를 구성한다. 각 Sampling 시간마다 Parameter와 Controller 이득을 결정한다.

제안하는 Controller의 구조는 그림 2 에서 보여 준다. Self-tuning Controller의 구조는 두 부분으로 구성이 되어 있다. 첫 부분은 Mobile Robot와 Computer와의 Interface이고 두번째 부분은 그림 1에서 나타낸 Controller를 나타 내고 있다. Interface의 경우는 각 Motor의 정보(위치 정보; 속도 정보 등)를 받아들이고 제어에 사용 되는 값(Motor Voltage 등)을 서로 교환하기 위한 회로로 구성이 되어있다. Controller는 PC를 이용하여 제어에 필요한 계산을 담당하게 되어 있다. 이 연구에서는 Mobile Robot의 Model 과 Controller 연구를 하였다.

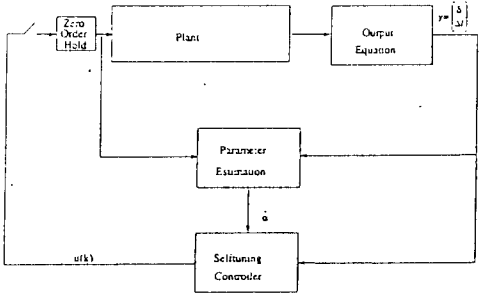

그림 2. Controller의 구조

2.3 Tricycle Type의 Mobile Robot의 구조

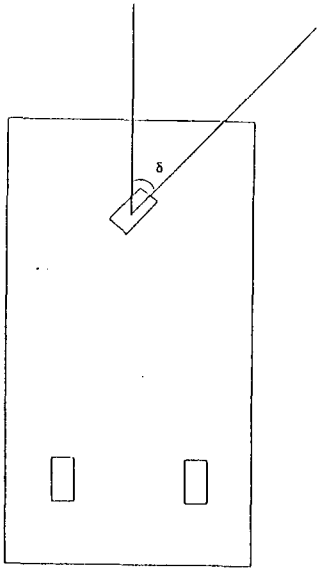

그림 3. Mobile Robot 구조

여기서 δ는 조향 Motor의 조향 각도이다. World 좌표계 (XY)에서 Mobile Robot의 위치와 방향을 결정하기 위해서 Mobile Robot의 Local 좌표계 (xy)는 Mobile Robot의 무게 중심에 위치한다.

Mobile Robot의 위치는 Mobile Robot에 설치된 Sensor들의 값에 의해서 World 좌표계 값이 결정 된다. 만약 Sensor가 구동되는 앞 두바퀴에 설치되어 있고 뒷바퀴는 단순히 무게 중심을 맞추기 위한 경우 앞 두바퀴에 설치된 Sensor는 Sampling 시간동안 구동 Motor의 이동 거리와 조향 Motor의 조향 각도의 값을 측정 할수 있다. Sampling 시간이 작다고 가정하는 경우 Sampling 시간 동안 Mobile Robot의 움직임은 작다고 가정할수 있다. 이 경우, Mobile Robot의 무게 중심의 위치는 Decomposition Method에 의해 다음과 같이 결정된다.

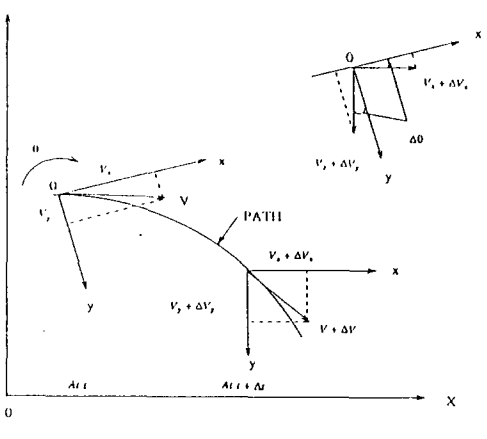

그림 4. World 좌표계와 Local 좌표계

Heading 각도와 이동시 그리는 원의 반경은 기하학적 구조에 의해서 결정이 된다. S를 뒷바퀴 사이의 중심점과 앞바퀴 사이의 중심점과의 거리, δ를 조향 각도라고 하면 앞바퀴의 회전 반경 R_k는 다음과 같이 표현이 된다.

$$R_k = \left| \frac{S}{\cos(90^\circ - |\delta|)} \right| \qquad (8)$$

또한 Heading 각도를 θ라고 정의할때 Heading Angle의 변화 $\Delta\theta$ 는

$$\Delta\theta = sgn(\delta)\frac{l_1}{R_k} \qquad (9)$$

여기서 l_1 은 Sampling Time 동안 움직인 거리이고

$$sgn(x) = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} \quad \text{이다.} \qquad (10)$$

Mobile Robot 무게 중심의 위치의 변화는 다음의 두 경우로 나누어 칠 수 있다:

(i) $\delta = 0$ (직진의 경우)
(ii) $\delta \neq 0$ (회전의 경우).

(i) $\delta = 0$ 인 경우

Local Coordinate Frame을 기준으로한 Mobile Robot 무게 중심의 변화량 ΔP 는

$$\Delta P = [\ \Delta P_x \ \ \Delta P_y\ ]' = [\ 0 \ \ l_1\ ]' \qquad (11)$$

여기서 ΔP_x 와 ΔP_y는 각각 Local 좌표계를 기준으로 x 방향과 y 방향의 위치의 변화를 나타낸다.

(ii) $\delta \neq 0$ 인 경우 (그림 5와 6)

Local Coordinate Frame을 기준으로한 Mobile Robot 무게 중심의 변화량, ΔP는

$$\Delta P = [\ \Delta P_x \ \ \Delta P_y\ ]' \qquad (12)$$

여기서

$$\Delta P_x = -\ sgn(\delta)[R_k \sin(90^\circ - sgn(\delta)\ |\delta|) - \frac{L_1}{2}]\ (1-\cos\Delta\theta)$$

$$\Delta P_y = [R_k \sin(90^\circ - |\delta|) - sgn(\delta)\ \frac{L_1}{2}]\ |\sin\Delta\theta|$$

여기서 ΔP_x 와 ΔP_y는 각각 Local 좌표계를 기준으로 x 방향과 y 방향의 위치의 변화를 나타내고 L_1 은 두 뒷바퀴 사이의 거리이다.

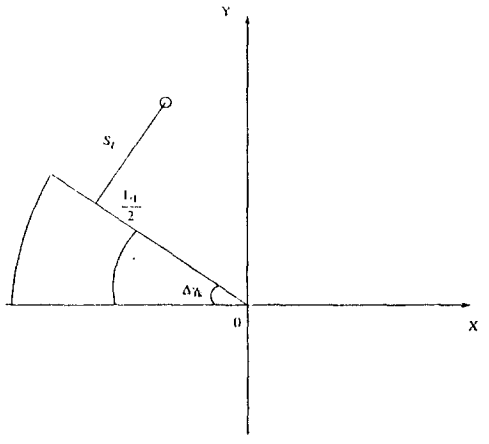

그림 5. $\delta > 0$ 일때

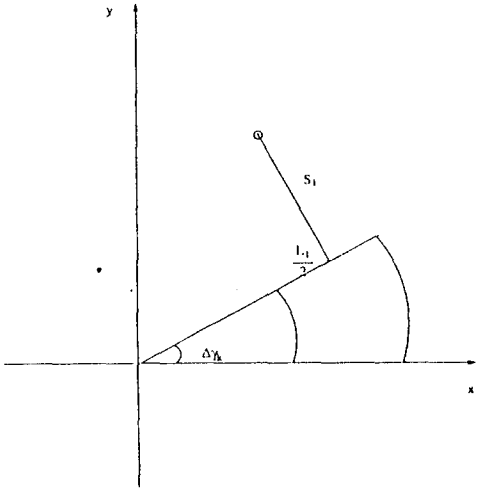

그림 6. $\delta < 0$ 일때

2.4 World Coordinate Frame에서의 Mobile Robot 무게 중심의 위치

World Coordinate Frame에서의 Mobile Robot의 무게 중심의 위치, P(k), 는 Local Coordinate Frame에서 계산된 $\Delta P(k)$ 에 의해서 계산이 된다. World Coordinate Frame에서의 Mobile Robot의 Heading Angle, $\theta(k)$, 는 k 번째 샘플링 후에 각 샘플링 시에 Heading Angle의 변화 합에 의해서 결정된다. 즉, Heading Angle의 값은 다음과 같이 결정된다.

$$\theta(k) = \sum_{i=1}^{k} \Delta\theta(i) + \theta(0) \qquad (13)$$

여기서 $\Delta\theta(i)$ 는 식(9)로 부터 얻어지며 $\theta(0)$ 는 Mobile Robot의 초기 Heading Angle이다.

World Coordinate Frame에서 왼쪽 뒷 바퀴의 위치를 결정하기 위해서 바로 전 Sampling Time의 Local Coordinate Frame에서의 위치 변화, $\Delta P(k)$, 가 World Coordinate Frame의 값으로 Mapping 된다. 이것은 $\Delta P(k)$에 Transformation Matrix A(k-1) 를 적용 함으로 정해진다 즉,

$$\begin{aligned} P(k) &= [\ P_x(k)\ \ P_y(k)\ ]' \\ &= P(k-1) + A(k-1)\Delta P(k) \end{aligned} \qquad (14)$$

여기서 P(k) 는 k 번째 샘플링 기간의 마지막에 고정된 World Coordinate Frame 에서 왼쪽 뒷바퀴의 위치이다. (2x2) Matrix 인 A(k-1) 는 Local Coordinate Frame 에서의 위치와 World Coordinate Frame 에서의 위치와의 관계를 나타낸다. A(k-1) 는 다음의 식으로 표현이 된다.

$$A(k-1) = \begin{bmatrix} \cos\theta(k-1) & \sin\theta(k-1) \\ -\sin\theta(k-1) & \cos\theta(k-1) \end{bmatrix} \qquad (15)$$

여기서 $\theta(k-1)$ 은 k-1 번째 샘플링 후의 Heading Angle이다.

Mobile Robot 무게 중심의 위치, $P_c(k)$, 는 다음의 식에 의해서 정해진다.

$$P_c(k) = P(k) + A(k)P_0 \qquad (16)$$

여기서 A(k) 는 식 (15)에서 k-1 대신 k를 치환 했을때의 값이고 P_0 는 Local Coordinate Frame 을 기준하여 Robot의 무게 중심 위치이다. 즉,

$$P_0 = [\frac{L_1}{2}\ \ S_1]' \qquad (17)$$

여기서 S_1 은 Mobile Robot의 무게 중심으로 부터 두 뒷바퀴를 연결하는 축의 중심과의 거리를 나타낸다.

2.5 Mobile Robot 의 Dynamic Model

Mobile Robot 무게 중심의 속도, $V(t)$, 는 시간이 t 일때 Local Coordinate Frame x와 y 축을 따라 두개의 성분 V_x 와 V_y 로 나뉘어 질수 있다. Mobile Robot 가 회전을 하는 경우 Mobile Robot 가 향해가는 방향 뿐 아니라 속도도 변한다. x 축에 평행한 속도의 변화량은 다음과 같이 구할수 있다

$$\begin{aligned} \delta V_x &= (V_x+\Delta V_x)\cos\theta\Delta\theta - V_x - (V_y+\Delta V_y)\sin\Delta\theta \\ &= V_x\cos\Delta\theta + \Delta V_x\cos\Delta\theta - V_x - V_y\sin\Delta\theta - \Delta V_y\sin\Delta\theta \end{aligned} \qquad (18)$$

$\Delta\theta$가 작다고 가정하면 $\cos(\Delta\theta)\approx 1$ 와 $\sin(\Delta\theta)\approx\Delta\theta$ 이고 또 2차항을 무시하면 식(18)은 다음과 같이 표현이 된다.

$$\delta V_x = \Delta V_x - V_y\Delta\theta \qquad (19)$$

Mobile Robot 의 Dynamic Model 은 다음과 같이 표현된다 [5,6].

$$I\ddot{\theta} = l_f P_f \sin\delta$$
$$m(\dot{V}_y + V_x\dot{\theta}) = P_f \sin\delta$$
$$m(\dot{V}_x - V_y\dot{\theta}) = P_f \cos\delta \qquad (20)$$

여기서 l_f는 전륜중심과 무게중심과의 거리이고 l_r는 후륜중심과 무게중심과의 거리이다. 여기서 Slip Angle 효과는 무시하였다. 위 식에서의 Input은 Longitudinal Force, P_f, 와 Steering Angle, δ, 이다. World Coordinate Frame 에서 Mobile Robot 무게 중심의 X 방향과 Y 방향 성분 속도, $\dot{P}_X$와 $\dot{P}_Y$,는 다음과 같이 나타낸다.

$$\dot{P}_X = -V_y\sin\theta + V_x\cos\theta$$
$$\dot{P}_Y = V_y\cos\theta + V_x\sin\theta \qquad (21)$$

위 시스템의 원하는 Output은 World Coordinate Frame 에서 Mobile Robot 의 무게 중심 위치, $P = [\,P_X \;\; P_Y\,]'$ 를 나타낸다. Dynamic Model을 유도하기 위해서 전륜과 후륜의 Vertical Loads, F_{zf} 와 F_{zr}는 다음과 같은 값을 갖는다.

$$F_{zf} = [mgl_r - P_f h] / (l_f + l_r)$$
$$F_{zr} = [mgl_f + P_f h] / (l_f + l_r) \qquad (22)$$

구동 Motor와 조향 Motor의 Dynamic Model 은 각각 다음과 같은 값을 갖는다.

$$K_1u_1/R_1 = J_1\ddot{\theta}_1 + (K_1K_{1b}/R_1 + B_1)\dot{\theta}_1 + \mu F_{zf} r$$
$$K_2u_2/R_2 = J_2\ddot{\delta} + (K_2K_{2b}/R_2 + B_2)\dot{\delta} + \mu F_{zf}\frac{L}{4} \qquad (23)$$

2.6 Simulation 결과

사용되는 제어 변수는 구동 Motor의 구동 거리와 조향 Motor의 각도인 두 변수를 사용한다. 본 연구에서는 식(1)의 q를 다음과 같이 정의하였다.

$$q = [\;\delta \;\; \Delta l\;]' \qquad (24)$$

여기서 δ는 조향각을 나타내고 Δl은 샘플링 시간동안 움직이는 거리를 나타낸다. 따라서 원하는 값 δ^d 와 Δl^d 는 다음과 같이 정해진다.

$$\delta^d(k+1) = atan2((P_Y^d(k+1)-P_Y(k)),(P_X^d(k+1)-P_X(k)))$$
$$\Delta l^d(k+1) = [[P_X^d(k+1)-P_X(k)]^2 + [P_Y^d(k+1)-P_Y(k)]^2]^{1/2} \qquad (25)$$

여기서 P_X^d 와 P_Y^d 는 World Coordinate Frame에서의 원하는 X와 Y축상의 궤적을 나타낸다.

Simulation을 위해서 퍼듀대학에서 제작된 Mobile Robot의 수치를 사용하였다.

$$I = 2.471\ kgm^2;\ h = 13.97\ cm\ ;\ g = 9.8\ m/sec$$
$$l_f = 28.2\ cm\ ;\ l_r = 32.77\ cm\ ;\ m = 54.8\ kg \qquad (26)$$

실험에 의해서 구동과 조향 모터의 Dynamic Model은 다음과 같다.

$$2.89u_1 = \dot{\theta}_1 + \frac{1}{15}\ddot{\theta}_1 + 0.0044F_{zf}$$
$$1.54u_2 = \dot{\delta} + \frac{1}{5.6}\ddot{\delta} + 0.072F_{zf} \qquad (27)$$

Mobile Robot의 운동은 위에서 주어진 식에 의해서 Simulation이 이루어졌다. Simulation을 위해서 IMSL 프로그램의 DIVPRK 서브루틴을 사용하였다. DIVPRK 서브루틴은 5차의 Runge-Kutta 알고리즘을 사용하였다.

Mobile Robot 제어를 위한 Self-tuning Controller는 Mobile Robot 의 무게 중심이 미리 정해진 Trajectory를 따라 가도록 설계가 되어 있다. 조향각 $\delta(k)$ 는 과거의 조향각과 과거에 조향 모터에 주어진 입력에 의해 결정된다. 또한, 이동거리 $\Delta l(k)$ 는 k 번째 샘플링 시간 동안 움직인 거리이며 과거 샘플링 기간동안 이동거리와 구동 모터에 주어진 입력에 결정된다. 사용된 δ 와 Δl 에 대한 Autoregressive Model은 다음과 같다.

$$q(k) = (A_1z^{-1} + A_2z^{-2} + A_3z^{-3})q(k) + (B_0 + B_1z^{-1} + B_2z^{-2})u(k-1) + e(k) \qquad (28)$$

여기서 사용된 파라메터는

$$A_1 = \begin{bmatrix} a_{11}^1 & 0 \\ 0 & a_{22}^1 \end{bmatrix} \quad A_2 = \begin{bmatrix} a_{11}^2 & 0 \\ 0 & a_{22}^2 \end{bmatrix} \quad A_3 = \begin{bmatrix} a_{11}^3 & 0 \\ 0 & 0 \end{bmatrix}$$
$$B_0 = \begin{bmatrix} b_{11}^0 & 0 \\ 0 & b_{22}^0 \end{bmatrix} \quad B_1 = \begin{bmatrix} b_{11}^1 & 0 \\ 0 & b_{22}^1 \end{bmatrix} \quad B_2 = \begin{bmatrix} b_{11}^2 & 0 \\ 0 & 0 \end{bmatrix} \qquad (29)$$

Mobile Robot가 원하는 평면상의 Trajectory를 따라 가기를 원하는 경우 q의 시간에 따르는 원하는 값인 $\{q^d\}$ 이 주어진다. 식(4)와 (5)에 의해서 β와 S를 구한후 그 값을 이용하여 식(7)에 의한 적응 제어기가 성공적으로 Mobile Robot를 제어 할수 있음을 Simulation 결과로 그림 7에서 보여준다.

3. 결론

본 논문에서는 self-tuning Algorithm을 이용한 제어기를 제안했다. 이러한 적응 제어기는 외부의 환경변화에 대해서 제어상에 사용이 되는 이득들을 자동으로 변화를 시켜 스스로 외부환경에 적응을 할수 있는 능력을 가지고 있다. 적응 제어를 이용한 Simulation의 결과를 보여 준다.

4. 참고 문헌

[1] K.J. Astrom, P. Eykhoff, "System Identification - A Survey," *Automatica*, pp. 123-162, Vol. 7, Pergamon Press, 1971

[2] K.J. Astrom, B. Wittenmark, "On Self-tuning Regulator," *Automatica*, pp. 185-199, Vol. 9, Pergamon Press, 1973

[3] K.J. Astrom, U. Borisson, L. Ljung and B. Wittenmark, "Theory and Application of Self-tuning Regulators," *Automatica*, pp. 457-476, Vol 13, Pergamon Press, 1977

[4] A.J. Koivo, T-H Guo, "Adaptive Linear Controller for Robotic Manipulator," *IEEE Transaction on Automatic Control*, pp. 162-171, Vol. AC-28, No.2, 1983

[5] G.W. Sandor, A.G. Erdman, *Advanced Mechanism Design: Analysis and Synthesis*, Prentice Hall, Englewood Cliffs, N.J., 1984

[6] H. Hatwal, E.C. Mikulcik, "Some Inverse Solutions to an Automobile Path Tracking Problem with Input Control of Steering and Brakes," *Vehicle System Dynamics*, pp 61-71, 1986

[7] A.J. Koivo, K.S. Lee, "Self-tuning Control of a Two-link Manipulator with a Flexible Forearm," *International Journal of Robotics Research*, Vol. 11, No. 4, Augut 1992

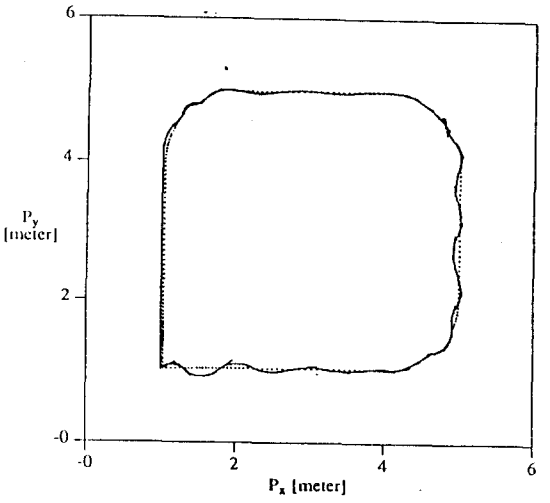

그림 7. Simulation 결과