동적프로그래밍을 이용한 자율이동로봇의 동작계획

Motion Planning of Autonomous Mobile Robot using Dynamic Programming

윤 희 상, 박 태 형*
(Hee-Sang Yoon and Tae-Hyoung Park)

Abstract: We propose a motion planning method for autonomous mobile robots. In order to minimize traveling time, a smooth path and a time optimal velocity profile should be generated under kinematic and dynamic constraints. In this paper, we develop an effective and practical method to generate a good solution with lower computation time. The initial path is obtained from voronoi diagram by Dijkstra's algorithm. Then the path is improved by changing the graph and path simultaneously. We apply the dynamic programming algorithm into the stage of improvement. Simulation results are presented to verify the performance of the proposed method.

Keywords: motion planning, path planning, trajectory planning, autonomous mobile robots, dynamic programming

I. 서론

자율이동로봇은 주변 환경으로부터 자신의 위치를 인지하여 목적지까지 스스로 이동하는 자율이동기능이 포함된 로봇이다. 자율이동로봇은 현재의 위치를 센서로부터 인식하여 위치를 측정하고, 측정된 위치로부터 목적지까지 동작계획(motion planning)을 통해 로봇을 제어하는 제어기술로 구분된다[1].

자율이동로봇의 주행성능을 높이기 위해서는 동작계획시 장애물을 회피하여 이동 가능한 경로를 생성하는 경로계획(path planning)과 주어진 경로로부터 최적의 가감속으로 속도프로파일을 생성하는 궤적계획(trajectory planning)이 중요하다[1]. 자율이동로봇의 동작계획문제는 로봇의 구조 및 크기에 대한 기구학 제약조건과 모터의 최대속도 및 토크 등의 동력학 제약조건을 함께 고려하여 로봇이 이동 가능한 최적의 경로 및 궤적을 생성해야하는 최적화 문제이다[2,3]

자율이동로봇의 동작계획문제는 경로계획문제와 궤적계획문제로 분리하여 단계별로 적용하는 계층적방법[2-4]과 궤환을 통해 해를 개선해 나가는 궤환적방법[5-7]이 연구되었다. 계층적방법은 경로계획에서 경로를 생성하고, 궤적계획으로 속도프로파일을 생성한다. 궤환적방법은 동력학을 고려한 연구와 고려하지 않은 연구로 구분된다. 동력학을 고려하지 않고, 가시도그래프(visibility graph)[1]와 같은 방법으로 직선 경로를 생성하고, 경로의 연결지점은 원호(circle arc)[2]나 클로소이드호(clothoid arc)[3]를 이용하여 부드럽게 변환한 연구가 있다. 그러나 동력학 특성을 고려하지 못하여 궤적계획시 사다리꼴형태의 등가감속으로만 나타내었다. 동력학을 고려한 연구로는 경로계획시 컴퓨터 그래픽에 사용되는 스플라인을 이용하여 경로를 생성하는 연구가 있다. [4]는 경로생성을 위해 3차스플라인을 사용하여 가속도 제약에 따른 곡선경로와 속도프로파일을 생성하였다. 그러나 계층적방법으로 접근한 방법들은 각 단계별로 해를 구하므로, 이동시간 측면에서 고려할 때 최적에 가까운 해를 찾는 것은 어렵다.

궤환적방법은 계층적방법과 달리 경로계획과 궤적계획의 해를 궤환을 통해 개선해 나가는 방법이다. 궤환적방법은 계층적 방법에 비해 많은 계산시간이 필요로 하나 해를 개선시킴으로써 이동시간이 짧은 경로를 생성할 수 있다. [5,6]은 B-스플라인으로 곡선을 생성하고, B-스플라인의 조정점을 변경하였다. 그러나 모든 B-스플라인의 조정점을 이동시켜 이동시간이 짧은 곡선경로를 탐색하경로를 탐색하 많은 계산시간이 필요하다. 이를 보완하여 [7]은 이동시간이 짧은 곡선경로 탐색을 위해 최적화 알고리즘 중 하나인 SA (Simulated Annealing)를 이용하여 B-스플라인의 조정점을 변경하여 곡선 경로를 생성하였다. 그러나 SA특성인 초기값에 따라 지역적 최소(local minimum)문제에 빠질 수 있으며, 이전 연구에 비해 감소하기는 했지만 반복횟수에 따라 계산시간이 증가하는 문제가 있다.

본 논문은 자율이동로봇의 국지적인 최적화 기법인 동적프로그래밍(DP: Dynamic Programming) 알고리즘을 사용하여 최소이동시간을 목표로 하는 궤환적구조의 동작계획방법을 제안한다. DP는 문제를 분할할 수 있는 특수한 경우에서의 최적화 기법으로서, 다양한 분야의 최적화문제의 해를 구하기 위해 사용되고 있다[8-10]. 본 논문에서는 DP를 이용하여 탐색 공간내에 동력학 제약조건을 충족하는 부분적인 해를 구하여, 탐색공간내의 전체 해를 구하는데 사용한다. 제안하는 동작계획방법은 초기경로생성과 경로개선단계로 구성된다. 초기경로생성단계에서는 빠른 계산을 위해 보르노이 다이어그램(voronoi diagram)과 탐색알고리즘인 딕스트라알고리즘(dijkstra's algorithm)으로 직선경로의 경유점

* 책임저자(Corresponding Author)
논문접수: 2009. 8. 19., 수정: 2009. 9. 28., 채택확정: 2009. 11. 4.
윤희상: 충북대학교 대학원 제어로봇공학과(nada621@cbnu.ac.kr)
박태형: 충북대학교 전자공학부(taehpark@cbnu.ac.kr)
※ 본 논문은 2008년도 충북대학교 학술연구지원사업의 연구비지원에 의하여 연구되었음.

을 생성한다. 경유점들에 대해 국지적인 곡선생성방법인 에르미트보간(hermite interpolation)을 이용하여 곡선경로로 변경한다. 경로개선단계에서는 DP 알고리즘을 반복 적용하여 이동시간이 짧은 경로로 개선시킨다. 시뮬레이션을 통해 개선된 경로를 초기경로와 비교하여 성능을 평가한다.

II. 문제 정의

1. 로봇 모델

자율이동로봇의 기구학 및 동력학 모델을 정의하고, 이로부터 이동거리에 대한 파라미터모델과 제약조건을 유도한다. 본 논문에서는 두 개의 구동바퀴로 구성된 차동구동방식의 자율이동로봇을 대상으로 한다. 그림 1(a)는 자율이동로봇의 기구학 모델을 나타낸다. 로봇의 무게중심의 위치와 자세를 (x,y,θ)로 정의하고, 로봇의 좌우바퀴의 반지름을 r, 중심과의 거리를 L이라 할 때, 바퀴가 미끄러짐이 없는 경우 자율이동로봇의 기구학은 다음과 같다.

$$\dot{x}=v\cos\theta=\frac{r}{2}cos\theta(\dot{\phi}_r+\dot{\phi}_l) \tag{1}$$

$$\dot{y}=v\sin\theta=\frac{r}{2}sin\theta(\dot{\phi}_r+\dot{\phi}_l) \tag{2}$$

$$\dot{\theta}=\frac{r}{2L}(\dot{\phi}_r-\dot{\phi}_l) \tag{3}$$

여기서 $\dot{\phi}_r$은 오른쪽 바퀴의 속도 이며, $\dot{\phi}_l$는 왼쪽 바퀴의 속도이다.

자율이동로봇의 동력학 모델은 일반적인 머니퓰레이터 로봇과 유사하나, 논홀로노믹(non-holonomic) 구속조건을 포함하는 형태로 변형된다[5,12]. 본 논문에서 사용한 자율이동로봇의 동력학모델은 참고문헌[5]에 유도과정이 있으며, 오른쪽, 왼쪽 바퀴의 토크 τ_1, τ_2에 대한 동력학식은 다음과 같다.

$$A\cos\theta\ddot{x}+A\sin\theta\ddot{y}+B\ddot{\theta}-Csin\theta\dot{x}\dot{\theta}+Ccos\theta\dot{y}\dot{\theta}=\tau_1 \tag{4}$$

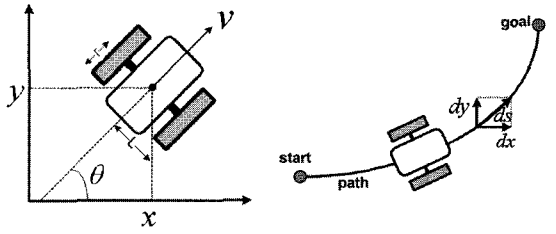
$$A\cos\theta\ddot{x}+A\sin\theta\ddot{y}-B\ddot{\theta}-Csin\theta\dot{x}\dot{\theta}+Ccos\theta\dot{y}\dot{\theta}=\tau_2 \tag{5}$$

여기서 $A=\frac{2I_y+mr^2}{2r}$, $B=\frac{2I_yL^2+I_zr^2}{2rL}$, $C=\frac{I_y}{r}$이며, m은 로봇의 무게, I_y, I_z는 y 및 z축방향의 관성모멘트이다.

자율이동로봇의 이동경로를 이동거리를 나타내는 거리 파라미터 s로 표현할 수 있다. 거리 파라미터 s는 기구학 및 동력학 모델을 단순화 할 수 있다. 2차원 좌표를 거리 파라미터 s로 $x=x(s)$, $y=y(s)$와 같이 나타낼 수 있다. 또한 s를 (4), (5)에 대입하여 속도 $\dot{s}=\frac{ds}{dt}$, 가속도 $\ddot{s}=\frac{d^2s}{dt^2}$, 회전속도 $\frac{d\theta}{ds}$ 및 회전가속도 $\frac{d^2\theta}{ds^2}$로 변환하여 정리하면,

$$(A+B\frac{d\theta}{ds})\ddot{s}+B\frac{d^2\theta}{ds^2}\dot{s}^2=\tau_1 \tag{6}$$

$$(A-B\frac{d\theta}{ds})\ddot{s}-B\frac{d^2\theta}{ds^2}\dot{s}^2=\tau_2 \tag{7}$$



(a) Kinematic model. (b) Path parameter model.

그림 1. 로봇 모델.

Fig. 1. The robot model of AMR.

와 같이 거리 파라미터 s로 동력학 모델을 표현할 수 있다.

자율이동로봇은 제한된 모터의 출력에 의해 최대속도 및 최대토크에 제한을 가진다. 따라서 로봇의 최대속도를 $v_{\max}$, 최대토크를 $\tau_{\max}$라 할 때, 로봇은 다음과 같은 속도 및 토크에 관한 제약조건을 가진다[12].

$$|v_r|,|v_l|\le v_{\max}, \tag{8}$$

$$|\tau_1|,|\tau_2|\le \tau_{\max} \tag{9}$$

또한 로봇이 경로를 따라 회전 시 제한된 회전속도 및 회전가속도에 의해 좌우바퀴의 속도 및 가속도에 제한을 가진다. 거리 파라미터 s로 표현된 이동경로는 (8),(9)의 최대속도, 가속도 조건을 충족하는 최대허용속도 $\dot{s}_{\max}$ 및 최대허용가속도 $\ddot{s}_{\max}$를 가진다. 경로 s에서 임의의 지점 s_k에서 (8)을 만족하는 좌우바퀴에 대한 최대허용속도 $\dot{s}_{\max}$는 다음과 같이 정리된다.

$$\dot{s}_{\max}=\min_{i=1,2}\sqrt{\frac{sgn(Q_i)v_{\max}}{Q_i}} \tag{10}$$

단, i가 1일 때 오른쪽, 2일 때 왼쪽 바퀴의 속도를 나타내며, Q_i는 $Q_1=1+L\frac{d\theta}{ds}$, $Q_2=1-L\frac{d\theta}{ds}$이다. 또한 s_k에서 좌우바퀴의 최대허용가속도 $\ddot{s}_{\max}$는 다음과 같다.

$$\ddot{s}_{\max}=\min_{i=1,2}\left(\frac{sgn(R_i)\tau_{\max}-T_i\dot{s}^2}{R_i}\right) \tag{11}$$

여기서 R_i, T_i는 다음과 같다.

$$R_1=A+B\frac{d\theta}{ds},\quad R_2=A-B\frac{d\theta}{ds} \tag{12}$$

$$T_1=B\frac{d^2\theta}{ds^2},\quad T_2=-B\frac{d^2\theta}{ds^2} \tag{13}$$

따라서 자율이동로봇의 이동경로는 (8)~(11)의 속도 및 가속도 제약조건을 충족해야 한다.

2. 동작계획 문제

본 논문에서는 자율이동로봇의 동작계획문제를 경로계획과 궤적계획문제로 분리 하여 접근한다. 먼저 경로계획문제를 정의한다. 로봇은 2차원 평면에서 일정영역을 차지하지만, 문제를 단순화하기 위해 점의 이동으로 표현한다. 또한 장애물도 로봇의 영역이 포함되어 있다고 가정한다. 시작점

을 $s \in R^2$, 목표점을 $g \in R^2$라 정의 하고, N개의 장애물 $O_1, \cdots, O_N$이 존재할 때, 로봇의 경로 $\boldsymbol{V}$는 시작점과 목표점 사이의 직선으로 연결된 경유점 $v_j \in R^2 (j = 1, \cdots M)$의 집합으로 표현할 수 있다[11].

$$\boldsymbol{V} = \{v_1, v_2, \cdots v_M\} \tag{14}$$

직선으로 연결된 경유점들은 실제 로봇이 이동이 가능하도록, 보간 및 근사 방법들을 이용하여 부드러운 곡선 형태로 표현이 가능하다. 경로계획문제는 시작점과 목표점사이에서 장애물과 충돌이 발생하지 않는 경유점 $\boldsymbol{V}$를 찾고, 이 경유점으로 부터 로봇이 주행 가능하도록 부드러운 곡선경로를 생성하는 문제이다.

부드러운 곡선경로를 따라 자율이동로봇이 안정적으로 빠르게 주행하기위해서는, 동력학 제약조건을 고려하여 로봇의 좌우바퀴의 속도를 결정하는 속도프로파일을 생성한다. 속도프로파일을 생성하면 주어진 경로에 대해 이동시간을 구할 수 있다. 경로의 총 이동거리가 s_e일 경우, 로봇의 동력학 제약조건을 고려한 최적의 속도 프로파일이 $\hat{s}$이라 할 때, 전체 경로에 대해 로봇의 이동시간은 다음과 같다[13].

$$\boldsymbol{J} = \int_0^{s_e} \frac{dt}{ds} ds = \int_0^{s_e} \frac{1}{\dot{\hat{s}}} ds \tag{15}$$

여기서 $\boldsymbol{J}$는 이동시간으로 경로에 대한 비용이다. 궤적계획문제는 로봇 모델에서의 속도 및 가속도 제약조건 (10),(11)을 충족하고, 비용 $\boldsymbol{J}$를 최소로 하는 속도프로파일 $\hat{s}$을 찾는 최적화 문제이다.

따라서 동작계획은 경로계획으로 경로를 생성 하고, 궤적계획으로 속도 프로파일을 생성하여, 비용 $\boldsymbol{J}$가 최소인 해를 찾는 문제이다.

III. 동작 계획 방법

본 논문에서는 자율이동로봇의 동작계획문제에 국지적 최적화 기법을 적용한다. 또한 동작계획문제를 경로계획과 궤적계획으로 따로 적용하는 것이 아닌, 경로계획으로 생성된 경로를 궤적계획으로 이동시간을 평가하여, 반복적으로 경로를 개선하여 이동시간을 줄이는 궤환적방법이다. 제안하는 동작계획방법은 초기경로를 생성하고, 경로를 개선하는 단계로 구성된다.

1. 초기경로생성

1.1 그래프 구성

장애물이 주어진 2차원 공간에서 장애물과 충돌 없는 경로를 생성하기위해 보르노이 다이어그램[1]을 이용한다. 보르노이 다이어그램은 장애물과의 거리가 같은 점들의 집합으로 나타낸다. 이 집합 중 분기점을 노드(node)로 지정하고, 노드들 사이의 연결선을 아크(arc)로 지정하여 그래프를 구성한다. 그림 2는 노드와 아크로 구성된 보르노이 다이어그램을 나타낸다.

노드와 아크로 구성된 그래프는 장애물을 회피하는 경로에 대한 정보이며, 경로탐색을 통해 시작점과 목표점까지의 초기경로를 생성한다. 경로탐색은 대표적인 방법인 딕스트라알고리즘을 적용하여 시작점에서 목표점까지 경유점들로 구성된 직선경로 $\boldsymbol{V}$를 구한다.

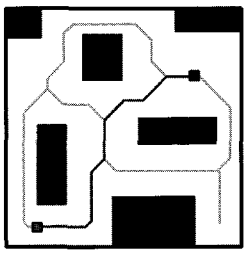
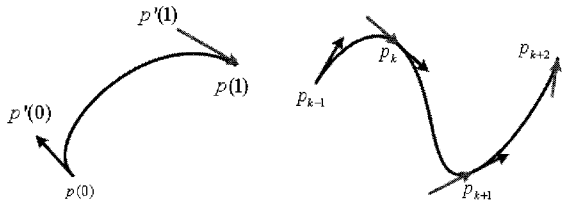


그림 2. 보르노이 다이어그램.
Fig. 2. Voronoi diagram.



(a) Two points. (b) Four points.

그림 3. 에르미트 보간.
Fig. 3. Hermite interpolation.

1.2 곡선생성

경로 $\boldsymbol{V}$는 직선으로 연결된 경로이다. 자율이동로봇은 직선으로 연결된 지점에서 회전 시 무한대의 가속도로 인해 이동하지 못한다. 로봇이 이동하기위해서는 경로가 곡선형태로 부드러워야한다. 따라서 직선경로 $\boldsymbol{V}$의 경유점들을 이용하여 직선경로를 곡선경로로 바꾸는 방법이 필요하다. 본 논문에서는 곡선생성을 위해 에르미트 보간을 사용한다. 에르미트 보간은 그림 3(a)와 같이 두 점과 각 점에서의 접선 벡터가 주어졌을 때, 두 점 사이를 곡선으로 연결하는 방법이다. 에르미트 보간은 그림 3(b)와 같이 연결점(joint points)에서 접선 벡터를 같게 하여 다수의 점에 대해 연속적인 곡선생성이 가능하다.

2차원 공간에서 곡선은 $p(u) = [x(u)\ y(u)]$ (단, $0 \le u \le 1$)와 같이 곡선 파라미터 u로 표현한다. 곡선은 3차다항식으로 표현 가능하며 다음과 같다[14].

$$P(u) = a_3 u^3 + a_2 u^2 + a_1 u + a_0 \text{ 단, } 0 \le u \le 1 \tag{16}$$

에르미트 보간은 양 끝점 즉 $u = 0$, $u = 1$일 때의 접선 벡터을 두어 (16)의 상수 a_1, a_2, a_3, a_4 정한다. 직선경로 $\boldsymbol{V}$의 임의의 경유점 v_j $(j = 1, \cdots M-1)$에 대해 에르미트 보간을 적용하면 다음과 같다[15].

$$p_j(u) = [u^3\ u^2\ u\ 1] \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_j \\ v_{j+1} \\ v'_j \\ v'_{j+1} \end{bmatrix} \tag{17}$$

(17)은 두 점에 대한 에르미트 보간을 이용하여 곡선을 생성한다. 모든 경유점에 대해 적용하기 위해서는 각 경유

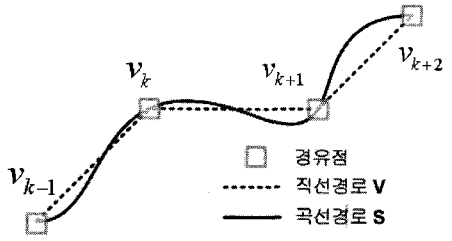


그림 4. 경로 비교.

Fig. 4. Comparison of paths.

점에서의 접선벡터가 필요하다. 시작점과 목표점은 로봇의 자세로 접선벡터를 구할 수 있으며, 시작점과 목표점을 제외한 나머지 경유점에서의 접선벡터는 자신을 기준으로 인접 경유점들의 방향을 이용하여 다음과 같이 계산한다.

$$\begin{cases} v'_j = (1-\xi)(v_{j+1} - v_{j-1}) \\ v'_{j+1} = (1-\xi)(v_{j+2} - v_j) \end{cases} \text{ 단, } 0 \le \xi \le 1 \qquad (18)$$

여기서 ξ는 곡선의 장력을 결정하는 상수이다.

각 경유점에서 (16)을 이용하여 생성한 곡선 p_j를 전체 곡선 경로 $\boldsymbol{P}$로 나타낼 수 있다.

$$\boldsymbol{P} = \{p_1, p_2 \cdots p_{M-1}\} \qquad (19)$$

에르미트 보간을 이용하여 생성한 곡선경로 $\boldsymbol{P}$는 곡선 파라미터 u로 나타낸 부분곡선들의 집합이다. 그러나 앞서 로봇 모델링에서는 경로를 거리 파라미터 s로 표현하였다. 곡선경로 $\boldsymbol{P}$를 로봇 모델링에서의 동력학식 및 제약조건 등을 적용하기 위해 거리 파라미터 s로 변환한다. 곡선경로 $\boldsymbol{P}$에서 j번째 곡선 $p_j(u) = [x_j(u)\ y_j(u)]$에 대해 거리 파라미터 $s_j(u)$는 다음과 같이 나타낼 수 있다.

$$s_j(u) = \begin{cases} s_{j-1}(1) + \sqrt{x_j(u)^2 + y_j(u)^2}, & u = 0 \\ s_j(u - \Delta u) + \sqrt{x_j(u)^2 + y_j(u)^2}, & u \neq 0 \end{cases} \qquad (20)$$

곡선 파라미터 u로 표현된 곡선경로 $\boldsymbol{P}$를 (20)을 이용하여 거리 파라미터로 변환하면 다음과 같이 곡선경로 $\boldsymbol{S}$로 나타낼 수 있다.

$$\boldsymbol{S} = \{s_1, s_2 \cdots s_{M-1}\} \qquad (21)$$

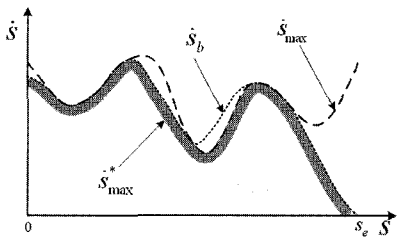
그림 4는 직선경로 $\boldsymbol{V}$와 $\boldsymbol{V}$의 경유점을 에르미트보간으로 곡선으로 변환한 곡선경로 $\boldsymbol{S}$를 나타낸다.

1.3 궤적계획

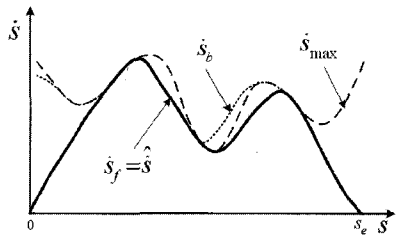
곡선경로 $\boldsymbol{S}$에 대해 로봇의 제약조건을 고려한 이동 가능한 속도 프로파일을 궤적계획알고리즘으로 구한다. 본 논문에서 사용한 궤적계획알고리즘[12]은 다음과 같은 단계를 가진다.

Step 1: 경로 $\boldsymbol{S}$에 대해 (10)을 이용하여 최대허용속도 $\dot{s}_{\max}$을 구한다(그림 5(a)).

Step 2: 경유점 카운터를 $k = M-1 \cdots 1$로 설정하고, (11)을 이용하여 최대허용가속도 $\ddot{s}_{\max}$를 계산한다. $\ddot{s}_{\max}$를 벗어나지 않도록 후진 방향($s_{M-1} \rightarrow s_1$)의 속도프로파일 $\dot{s}_b$를 구한다(그림 5(a)).



(a) Backward velocity profile



(b) Forward velocity profile

그림 5. 궤적계획.

Fig. 5. Trajectory planning.

$$\dot{s}_b(k) = \sqrt{\dot{s}_b(k-1)^2 + 2\ddot{s}_{\max}(k)ds} \qquad (22)$$

Step 3: 새로운 최대허용속도 $\dot{s}^*_{\max}$는 다음과 같이 설정한다(그림 5(a)).

$$\dot{s}^*_{\max} = \min(\dot{s}_{\max}, \dot{s}_b) \qquad (23)$$

Step 4: 경유점 카운터를 $k = 1, \cdots M-1$ 까지 역으로 설정하고, 전진 방향($s_1 \rightarrow s_{M-1}$)의 속도프로파일 $\dot{s}_f$를 다음과 같이 구한다(그림 5(b)).

$$\dot{s}_f(k) = \sqrt{\dot{s}_f(k-1)^2 + 2\ddot{s}_{\max}(k)ds} \qquad (24)$$

여기서 $\dot{s}_f$가 시작점에서 목표점 까지 가속 및 감속이 고려된 속도 프로파일 $\hat{\dot{s}}$이다. 속도 프로파일 $\hat{\dot{s}}$에 대해 (15)를 이용하여, 이동시간을 구하면 경로 $\boldsymbol{S}$에 대한 비용 $\boldsymbol{J}$를 구할 수 있다.

2. 경로개선

보르노이 다이어그램과 에르미트보간을 이용하여 생성된 곡선경로 $\boldsymbol{S}$는 시작점에서 목표점까지 부드러운 곡선을 생성한다. 그러나 곡선경로 $\boldsymbol{S}$를 구성하는 경유점 $\boldsymbol{V}$의 위치가 다른 곳에 위치할 경우, 이동시간이 단축될 수 있다. 경로개선단계는 곡선경로 $\boldsymbol{S}$의 경유점 $\boldsymbol{V}$의 위치를 변경하여 이동시간이 짧은 경로로 개선하는 단계이다.

경유점들에 대한 각 부분구간의 곡선을 $s_k(v_k, v_{k+1})$이라 하고, 이때의 이동시간을 J_k라 할 때, 곡선경로 $\boldsymbol{S}$에 대한 전체 이동시간 $J(S)$는 다음과 같이 나타낼 수 있다[9].

$$J(S) = J_1[s_1(v_1, v_2)] + J_2[s_2(v_2, v_3)] \cdots + J_{M-1}[s_{M-1}(v_{M-1}, v_M)] \qquad (25)$$

여기서 각 구간에서 경유점들을 이동시켜 생성된 구간의 곡선경로가 이동시간을 단축시킬 수 있다. 만일 각 구간의 최소이동시간을 가지는 경유점을 찾는다면, 전체경로의 이

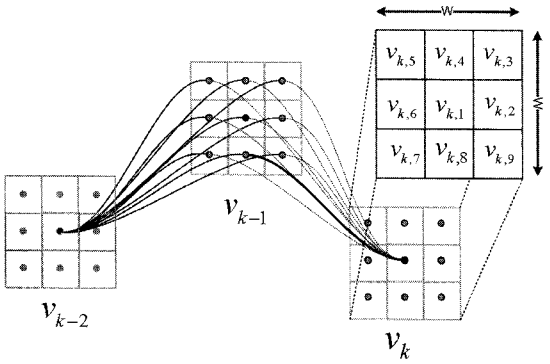


그림 6. 탐색창.

Fig. 6. Searching windows.

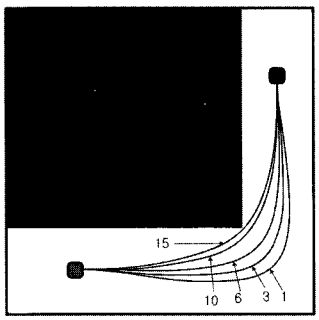


그림 7. 경로개선 예.

Fig. 7. Example of path improvement.

동시간을 줄일 수 있다.

이동시간이 단축시키는 경로를 탐색하기 위해 DP 알고리즘을 이용한다. DP는 전체 문제를 부분문제로 분할하고, 각 문제의 해를 구하여, 이를 다시 결합하여 전체문제의 해를 구하는 방법이다. DP는 최적성이 입증된 방법으로, 다양한 분야의 최적화 문제에 활용되고 있다[8-10]. DP는 탐색평면의 차원에 따라 curse of dimensionality문제를 가지나 제안하는 DP는 2차원 문제로 그 효과가 크지 않다.

DP를 동작계획에 적용하기 위해 초기경로생성 시 경유점 $\boldsymbol{V}=\{v_1, v_2 \cdots v_k\}$를 부분문제로 분할하기위한 스테이지로(stage)로 정의한다. 시작점 v_1과 목표점 v_k를 제외한 나머지 스테이지를 중심으로 최적해 탐색을 위해 그림 6과 같이 $W\times W$개의 탐색창을 구성한다. 탐색창의 W^2개의 점들은 한 스테이지에서 경로 탐색을 위한 스테이트(state)로 정의한다. k번째 경유점 v_k에 대해 스테이트는 $v_{k,1}, \cdots v_{k,W^2}$로 표시한다. DP에서 부분구간의 곡선경로 생성은 초기경로의 곡선생성과 같이 에르미트보간을 이용한다. 임의의 스테이지 $v_{k,\alpha}$와 $v_{(k+1,\beta)}$사이의 곡선은 $p_{(k,\alpha)(k+1,\beta)}$로 표시하고, 이 때 이동거리에 대한 파라미터 경로는 $s_{(k,\alpha)(k+1\beta)}$ 표시한다. $s_{(k,\alpha)(k+1\beta)}$는 궤적계획알고리즘을 적용하여 구간별 이동시간을 계산할 수 있다. 시작점부터 경유점 $v_{k,\alpha}$까지의 최소이동시간은 $\boldsymbol{J}^*_{(k,\alpha)}$이며, 이전 스테이지와의 이동시간은 $\boldsymbol{J}_{(k,\beta)}$로 표시한다.

곡선경로 $s_1, \cdots, s_{M-1}$에 대해 구간별 순차적인 최소이동시간은 다음과 같다.

$$\begin{aligned} J^*_{1,\alpha} &= \min\left[J_1(s_1, s_{2,\alpha})\right] \\ J^*_{2,\alpha} &= \min_{\beta\in W^2}\left[J^*_{1,\alpha} + J_{2,\alpha}(s_{2,\beta}, s_{3,\alpha})\right] \\ &\vdots \\ J^*_{M-1,\alpha} &= \min_{\beta\in W^2}\left[J^*_{M-2,\alpha} + J_{M-1,\alpha}(s_{M-1,\beta}, s_{M\alpha})\right] \end{aligned} \tag{26}$$

(15)에서 최적에 가까운 해를 구하기 위해서는 각 스테이지에서의 이동시간이 최소가 되는 스테이트를 탐색해야 한다. 여기서 탐색창의 크기에 따라 해의 최적성 여부를 판별할 수 있다. 이상적인 경우 지도의 크기가 $M_x\times M_y$인 일 때 탐색창의 크기와 지도의 크기가 같은 경우, 즉 $W^2 = M_x \times M_y \in R^2$일 때, 전역탐색으로 최적에 가까운 해를 구할 수 있다. 그러나 해를 구하기 위해 많은 계산시간이 필요하다. 스테이지의 수가 M개일 때, 탐색창의 스테이트 수는 W^2이며, 계산시간은 $O(MW^4) = O[M(M_xM_y)^2]$이다. 따라서 탐색창의 크기 W가 커지면, 탐색영역이 4제곱에 비례하여 증가한다.

전역탐색의 경우 탐색 영역이 전체공간으로 최적에 가까운해를 찾지만 많은 연산이 필요하다. 반면에 탐색영역이 작으면, 연산이 줄어 계산시간을 줄일 수 있지만, 최적해를 보장하지 못한다. 따라서 탐색창의 크기를 줄이고, 알고리즘을 반복적으로 적용하면, 탐색영역이 줄어 전역탐색에 비해 빠르게 최적에 근사한 해를 구할 수 있다.

본 논문에서는 DP를 동작계획의 경로개선에 적용하기 위해 탐색창의 크기를 지도크기보다 작게 설정하고, 반복적용하여 경로를 개선한다. 알고리즘의 종료는 더 이상 이동시간이 줄어들지 않는 경우에 종료한다. DP를 이용한 동작계획의 경로개선 알고리즘은 다음과 같다.

Step 1: 스테이지 k=1 에 대하여 최소이동시간 $\boldsymbol{J}^*_{(1,1)}$=0으로 설정한다.

Step 2: 스테이지를 $k=2,\cdots,M-1$ 로 증가시킨다.

Step 3: 각 스테이지의 스테이트 $v_{k,\alpha}$에 대해 (17)을 이용하여 곡선 $p_{(k,\alpha)(k+1,\beta)}$를 생성하고, 이를 (18)으로 거리파라미터 $s_{(k,\alpha)(k+1\beta)}$로 변환한다.

Step 4: $s_{(k,\alpha)(k+1\beta)}$에 대해 궤적계획알고리즘을 적용하여, 구간 이동시간 $\boldsymbol{J}_{(k,\alpha)(k+1,\beta)}$를 계산한다. 최소이동시간 $\boldsymbol{J}^*_{(k,\alpha)}$는 다음과 같이 계산하여 저장한다.

$$\boldsymbol{J}^*_{k,\alpha} = \min_{\beta\in W^2}\left(\boldsymbol{J}^*_{k-1,\beta} + \boldsymbol{J}_{(k-1,\beta)(k,\alpha)}\right) \tag{27}$$

$\boldsymbol{J}^*_{(k,\alpha)}$가 생성되는 β 값을 지시자 $\pi_{k,\alpha}$ 로 저장한다.

Step 5: 최종 스테이지 $k=M-1$ 에 대하여, 시작점부터 목표점까지의 최소누적거리 $\boldsymbol{J}^*_{M,1}$ 를 (9) 와 같이 구하고 지시자 $\pi_{M,1}$ 을 저장한다.

Step 6: 스테이지를 $k=M-1,\cdots,2$ 로 감소시키면서 지시자를 역 추적하며, 각 스테이지에서 최소이동시간을 가지는 지점 v^*_k 을 찾는다. 개선된 경유점 $\boldsymbol{V'}$ 는 각 스테이지의 최적 격자점으로 다음과 같이 구성된다.

$$V' = \{v_1, v_2^*, \cdots, v_{M-1}^*, v_M\} \quad (28)$$

Step 7: 새로운 경유점에 대해 이동시간이 일정값 이상 개선이 없을 경우 Step 1부터 Step 6의 과정을 반복한다.

그림 6은 k번째 스테이지의 첫번째 스테이트에서 이전 $k-1$번째 스테이지의 스테이트들에서 곡선을 생성하여 곡선들 중 이동시간이 최소가 되는 경로가 선택되어 부분해를 구하는 것을 보여준다. 그림 7은 부분해로부터 전체해를 구하는 DP 알고리즘을 적용하여, 반복횟수의 증가에 따라 이동시간이 줄어드는 방향으로 경로가 개선되는 과정을 보여준다.

IV. 시뮬레이션

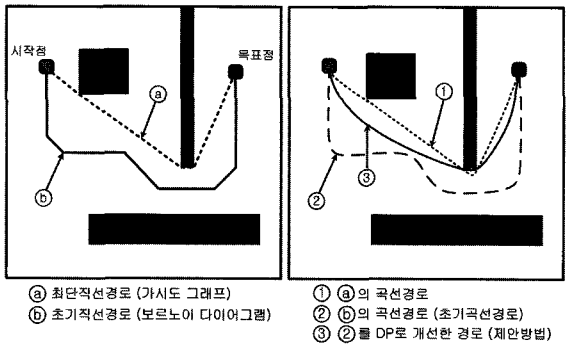
자율이동로봇의 동작계획 방법의 성능 평가를 위해 시뮬레이션을 수행하였다. 알고리즘은 IBM-PC (Intel Core2 CPU 1.83GHz)에서 MS-Windows XP상에서 Matlab 7.3을 이용하여 구현하였다.

로봇의 크기, 무게, 바퀴 관성과 같은 물리적인 변수들과 토크 및 속도에 대한 제약조건을 표 1에 나타내었다. 시뮬레이션은 20[m]×20[m] 평면에 장애물이 배치된 지도를 사용하였다. 단위격자의 크기는 100[mm]×100[mm]로 설정하였다. 성능 검증을 위해 다음의 시뮬레이션을 수행하였다. 첫 번째로 최단 경로 생성방법인 가시도그래프로 초기경로

표 1. 시뮬레이션 모델 변수.

Table 1. Variables of the simulation model.

r	0.1 $[m]$	m_1	50.0 $[kg]$
L	0.75 $[m]$	m_2	1.0 $[kg]$
I	27.17$[kg \cdot m^2]$	I_y	0.005 $[kg \cdot m^2]$
$\lvert\tau_{\max}\rvert$	≤ 1.0 $[N \cdot m]$	$\lvert v_{\max}\rvert$	≤ 1.0 $[m/s]$



(a) Initial path. (b) Initial and improved curve paths.

그림 8. 경로계획 비교결과.

Fig. 8. Comparative results of path planning.

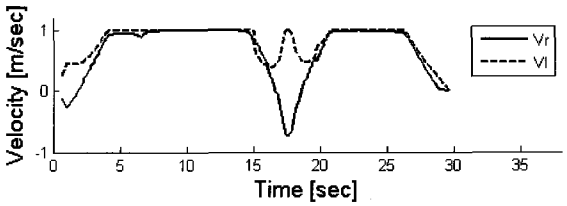
표 2. 시뮬레이션 결과 비교.

Table 2. Comparison of simulation results.

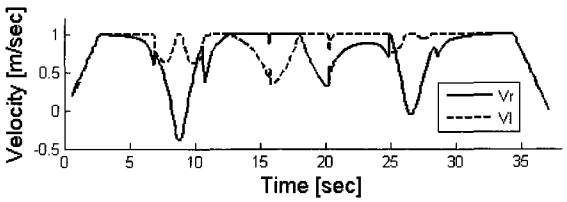
	가시도그래프	보르노이 다이어그램	DP를 이용한 개선경로
이동거리	21.5 [m]	29.1 [m]	22.1 [m]
이동시간	29.63 [sec]	37.09 [sec]	27.17 [sec]

를 생성한 경로를 곡선경로로 변경하여, 제안한 방법과 이동시간, 이동거리, 속도 및 토크 등을 비교하였다. 두 번째로 DP에서 사용되는 탐색창의 크기 및 반복횟수에 따라 개선된 경로를 비교하여, DP의 설정변수에 따른 특성을 비교하였다.

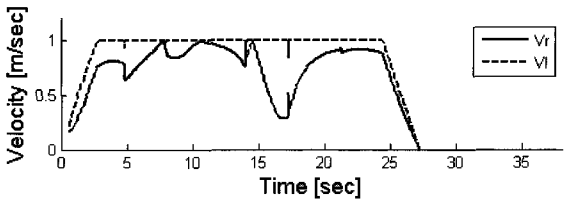
최단경로를 생성하는 가시도그래프와 비교를 위해, DP에서의 탐색창의 크기는 $W=3$으로 설정하였으며, DP알고리즘의 종료는 이동시간의 개선이 0.1초 이하일 경우 알고리즘을 종료하도록 하였다. 시작점은 (4.5[m], 3.1[m], $-\pi/2$), 목표점은 (5.8[m], 17.7[m], $\pi/2$)일 때, 그림 8(a)는 가시도그



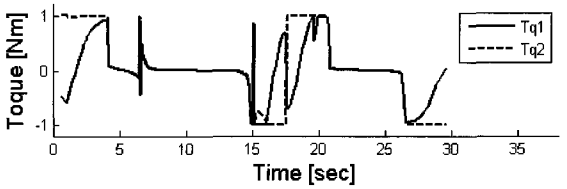
(a) Velocities of V-graph path.



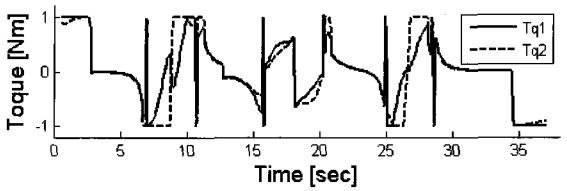
(b) Velocities of initial path.



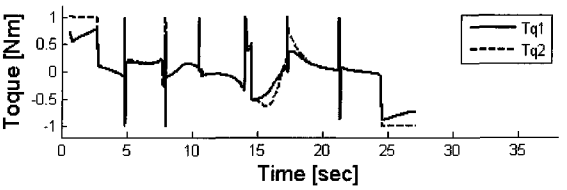
(c) Velocities of improved path(proposed).



(d) Toques of V-graph path.



(e) Toques of initial path.



(f) Toques of improved path(proposed).

그림 9. 좌우바퀴의 속도 및 토크.

Fig. 9. Toques and velocities of wheel.

래프로 구한 최단 직선 경로와 본 논문에서 사용하는 보르노이 다이어그램으로 구한 초기 직선 경로를 나타내며, 그림 8(b)는 최단 직선 경로를 곡선화한 경로와 초기 직선 경로를 곡선화한 경로 및 이를 DP를 이용하여 개선한 경로의 결과를 보여준다. 그림 8(b)의 경로들의 이동거리 및 이동시간은 표 2에 나타내었다. 표 2에서 최단경로를 생성하는 가시도 그래프는 보르노이 다이어그램으로 생성한 경로 및 제안한 개선 경로에 비해 이동거리가 짧다. 그러나 이동시간은 제안한 개선경로가 27.17초로 가시도 그래프에 의한 경로의 29.63초보다 더 빠른 것을 확인할 수 있다. 이는 DP를 이용하여 경로를 개선하는 제안 방법이 최단 경로 생성 후 곡선 경로를 생성하는 방법보다 이동시간 측면에서 좀 더 빠른 경로를 생성하는 것을 볼 수 있다.

그림 9는 그림 8(b) 경로에 대해 각 경로의 이동시간에 대한 속도 및 토크에 관한 그래프를 나타낸다. 그림 9(a), (b), (c)를 보면 개선된 (c)의 경우가 속도 곡선이 부드럽고, 좌우바퀴가 높은 속도를 유지하는 것을 확인 할 수 있다. 또한 토크는 개선된 경로인 (f)의 토크 그래프가 좌우바퀴의 토크가 급격한 차이 없이 최대토크를 보다 잘 사용한 것을 볼 수 있다.

다음으로 그림 10과 같은 지도를 사용하여 탐색창의 크기 및 반복횟수에 따른 성능비교를 하였다. 시작점은 (18.5[m], 2.7[m], 0°)이며, 목표점은 (5.8[m], 17.7[m], 0°)이다. 먼저 탐색창의 크기에 따라 개선된 경로의 이동시간 및 DP의 계산시간을 비교하였다. DP의 반복횟수는 1회로 제한하였으며, 탐색창의 크기를 $W=3,5,7,9,11$로 증가시키면서, 개선된 경로들의 이동시간 및 계산시간을 비교하였다. 표 3은 탐색창의 크기에 따른 이동시간 및 계산시간을 나타낸다. 탐색창의 크기가 가장 작은 3과 가장 큰 11일 때를 비교하면, 이동시간은 29.23초에서 26.82초로 약 8.2%에 해당하는 2.41초가 개선되었다. 그러나 계산시간은 0.86초에서 135.48초로 약 158배가 증가하였다. 탐색창의 크기를 증가

표 3. 탐색창 크기에 따른 시뮬레이션 결과.

Table 3. Comparison of computing and traveling time as a size of searching window.

No.	탐색창 크기	이동시간 [sec]	계산시간 [sec]
1	3	29.23	0.86
2	5	28.35	5.98
3	7	27.61	23.81
4	9	27.22	60.75
5	11	26.82	135.48

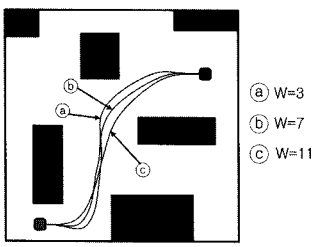


그림 10. 탐색창 크기에 따른 경로.

Fig. 10. Comparison of paths by size of searching window.

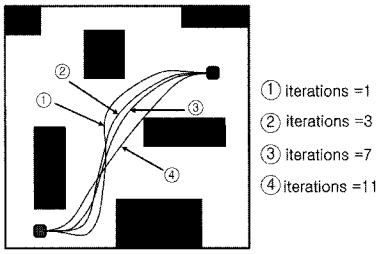


그림 11. 반복횟수에 따른 경로 비교.

Fig. 11. Comparison of paths by iterations.

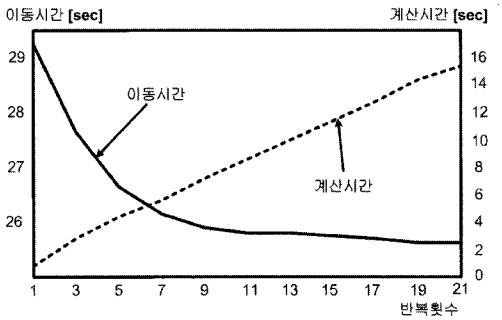


그림 12. 반복횟수에 따른 이동시간 및 계산시간.

Fig. 12. Comparison of computing and traveling time as a function of iterations.

시킬수록 경로는 개선되나, 계산시간은 탐색창의 크기에 따라 급격히 증가하는 것을 볼 수 있다.

다음으로 반복횟수에 따른 경로의 개선정도를 비교하였다. 탐색창의 크기는 $W=3$로 고정하고 반복횟수를 1~25회로 증가시키면서 이동시간 및 계산시간을 비교하였다. 그림 12는 반복횟수 증가에 따른 이동시간 및 계산시간을 나타낸다. 반복횟수의 증가에 따라 이동시간이 줄어드는 것을 볼 수 있으나, 7회 부터는 그 감소폭이 줄어든 것을 볼 수 있다. 또한 계산시간은 1회에 0.86초에서 반복횟수에 비례하여 증가하였다.

탐색창의 크기 및 반복횟수에 따른 경로는 그림 10과 그림 11에 나타내었다. 그림 10의 ⓐ,ⓑ,ⓒ는 탐색창의 크기가 3,7,11 일 때, 개선된 경로를 보여주며, 그림 11의 ①,②,③,④는 반복횟수가 1,3,7,11 일 때의 경로를 나타낸다. ⓒ,③은 이동시간이 각각 26.82초, 26.16초로이며 개선된 경로가 서로 유사하다. 그러나 계산시간은 ⓒ가 135.48초, ③이 5.69초로 많은 차이가 발생한다. 따라서 탐색창의 크기보다는 반복횟수를 증가시키는 것이 계산시간이 빠르고, 이동시간이 짧은 경로로 개선하는데 유리한 것을 확인할 수 있다. 또한 DP의 반복횟수가 증가함에 따라 개선율이 떨어지므로 개선율의 하한값을 설정하여 개선율이 하한값 이하일 경우 DP 알고리즘을 종료하여 효율적인 경로생성이 가능하다.

결과적으로 본 시뮬레이션 결과는 탐색창의 크기로 최소로 하고 DP를 반복적으로 적용하는 것이 이동시간 단축과 계산시간에 유리한 것을 확인할 수 있었다. 또한 이동경로는 경로계획과 궤적계획을 DP를 이용하여 궤환적 구조로 개선하였으며, 이동시간이 빠른 경로 생성 및 속도프로파일을 생성하여 동작계획 시 최소이동시간 동작계획에 근사한

방법임을 입증한다. 그러나 제안한 방법 또한 반복적으로 해를 개선시키므로, 계산시간이 증가할 수 있으며 이는 지도의 단위격자의 크기를 적절히 설정하여 수렴속도를 빠르게 하거나, 반복횟수를 고정하여 효과를 줄일 수 있다.

V. 결론

DP를 이용하여 자율이동로봇의 동작계획방법을 제안하였다. 제안한 방법은 궤환적구조를 가진 방법으로 이동시간이 줄어드는 방향으로 경로를 반복적으로 개선한다. 기존의 궤환적방법은 최소시간 경로 생성을 위해 초기경로 생성 후 전체 공간에 대해 랜덤하게 반복적으로 탐색하였다. 따라서 수렴하기 위해서는 계산시간이 많이 걸린다. 제안한 방법은 보르노이 다이어그램으로 장애물과의 거리가 최대인 초기 직선 경로를 생성하고, 최적화 알고리즘인 DP알고리즘으로 경로를 개선하였다. 또한 DP의 탐색창 크기는 작게 설정 하여 탐색공간을 줄이고, 줄어든 탐색공간에 대해 반복 적용하여, 효율적으로 경로를 개선하여 최소시간 해에 근접하는 방법이다.

DP의 특성상 스테이지가 증가하여 탐색영역이 커지면, 계산시간이 증가하나, 제안한 방법은 부분적인 경로생성이 가능하여 시스템 성능에 따라 전체 또는 부분적으로 경로를 생성하여 계산시간 문제의 효과를 줄일 수 있다. 제안한 방법은 다양한 종류의 자율이동로봇에 적용가능하며, 특히 동력학을 고려한 부드럽고 빠르게 이동하는 로봇에 효과적으로 적용 될 수 있으리라 기대된다. 추후 연구를 통하여, 이동하는 장애물에 대한 회피 및 실시간 동작계획을 연구를 통해 해법을 제시하고자 한다.

참고문헌

[1] R. Siegwart and I. R. Nourbakhsh, "Introduction to autonomous mobile robots," The MIT Press, 2004.

[2] Y. H. Liu and S. Arimoto, "Finding the shortest path of a disc among polygonal obstacles using a radius-independent Graph," *IEEE Trans. on Robotics & Automation*, vol. 11, no. 5, pp. 682-691, 1995.

[3] T. Fraichard and A. Scheuer, "From reeds and shepp's to continuous-curvature paths," *IEEE Trans. on Robotics*, vol. 20, no. 6, pp. 1025-1035, 2004.

[4] M. Lepetic, G. Klancar, I. Skrjanc, D. Matko, and B. Potocnik, "Time optimal path planning considering acceleration limits," *Robotics & Autonomous Systems*, vol. 45, no. 3, pp. 199-210, 2003.

[5] M. Yamamoto, M. Iwamura, and A. Mohri, "Quasi-time-optimal motion planning of mobile platforms in the presence of obstacles," *IEEE/RSJ Int. Conf. On Intelligent Robot and Systems*, vol. 4, pp. 2958-2963, 1999.

[6] Z. Shiller and Y. R. Gwo, "Dynamic motion planning of autonomous vehicles," *IEEE Trans. on Robotics & Automation*, vol. 7, no. 2, pp. 241-249, 1991.

[7] M. Haddad, T. Chettibi, S. Hanchi, and H. E. Lehtihe, "A random-profile approach for trajectory planning of wheeled mobile robots," *European Journal of Mechanics A/Solids*, vol. 26, no. 3, pp. 519-540, 2007.

[8] A. A. Amini, T. E. Weymouth, and R. C. Jain, "Using dynamic programming for solving variational problems in vision," *IEEE Trans. on Pattern Analysis & Machine Intelligence*, vol. 12, no. 9, pp. 855-866, 1990.

[9] D. Geiger, A. Gupta, L. A. Costa, and J. Vlontzos, "Dynamic programming for detecting, tracking, and matching deformable contours," *IEEE Trans. on Pattern Analysis & Machine Intelligence*, vol. 17, no. 3, pp. 294-302, 1991.

[10] T. H. Park and B. H. Lee, "An approach to robot motion analysis and planning for conveyor tracking," *IEEE Trans. on Systems, Man, and Cybernetics,* vol. 22, no. 2, pp. 378-384, 1992.

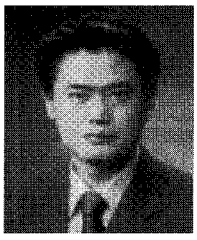
[11] 윤희상, 유진오, 박태형, "자율이동로봇을 위한 동적 경로 계획 방법," 제어 · 로봇 · 시스템학회 논문지, 제 14권 제4호, pp. 392-398, 2008.

[12] W. Wu, H. Chen, and P. Y. Woo, "Time optimal path planning for a wheeled mobile robot," *Journal of Robotic Systems*, vol. 17, no. 11, pp. 585-591, 2000.

[13] J. E. Bobrow, "Optimal robot path planning using the minimum-time criterion," *IEEE J. of Robotics & Automation*, vol. 4, no. 4, pp. 443-450, 1988.

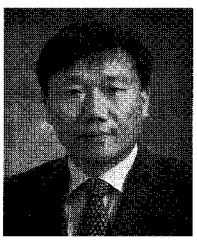
[14] A. Piazzi, C. G. L. Bianco, and M. Romano, "η^3-splines for the smooth path generation of wheeled mobile robots," *IEEE Trans. on Robotics*, vol. 23, no. 5, pp. 1089-1095, 2007.

[15] V. B. Anand, *Computer Graphics & Geometric Modeling for Engineers*, J. Wiley, 1993.



윤 희 상

2004년 충북대 전기전자컴퓨터공학부 졸업. 2006년 동 대학원 제어계측공학과(석사). 2006년~2007년 충북대 부설 유비쿼터스 바이오정보기술 연구센터 주임연구원. 2007년~현재 충북대 제어계측공학과 박사과정. 관심분야는 로보틱스, 영상처리, 임베디드시스템 등.



박 태 형

1988년 서울대 제어계측공학과 졸업. 1992년 동 대학원 제어계측공학과(석사). 1994년 동 대학원 제어계측공학과(박사). 1994년~1997년 삼성 테크윈(주) 정밀기기연구소 선임 연구원. 1997년~현재 충북대 전기전자컴퓨터공학부 교수. 2000년~2001년 Univ. of Toronto 방문교수. 관심분야는 로보틱스 및 자동화, 전자조립 및 검사 시스템 등.