위치 추정, 충돌 회피, 동작 계획이 융합된 이동 로봇의 자율주행 기술 구현

노성우* · 고낙용** · 김태균***

Implementing Autonomous Navigation of a Mobile Robot Integrating Localization, Obstacle Avoidance and Path Planning

Sung-woo Noh* · Nak-yong Ko** · Tae-gyun Kim***

요 약

본 논문은 실내 이동로봇의 자율주행 방법을 적용한 결과를 기술한다. 구체적으로 지도생성, 위치추정, 장애물 회피, 경로계획에 대해서 설명한다. 기하학적 지도는 위치추정과 경로계획에 이용된다. 위치 추정을 위해서 지도 정보를 이용하여 센서 데이터를 계산하고 이를 실제 센서 데이터와 비교한다. 위치 추정에는 몬테 카를로 위치 추정 방법을 사용한다. 인공 전위계를 사용하여 장애물로부터의 척력과 목표 위치로의 인력을 구하여 장물을 피한다. 경로계획을 위해 다익스트라 알고리즘을 이용하여 로봇의 출발 위치에서 목표 위치까지의 최단거리 경로를 구한다. 이러한 방법들이 통합하여 자율 주행 방법을 실제로 구현하여 실험하였다. 실제 실험을 통하여 제안한 방법이 로봇을 안전하게 자율주행하게 함을 확인하였다.

ABSTRACT

This paper presents an implementation of autonomous navigation of a mobile robot indoors. It explains methods for map building, localization, obstacle avoidance and path planning. Geometric map is used for localization and path planning. The localization method calculates sensor data based on the map for comparison with the real sensor data. Monte Carlo Localization(MCL) method is adopted for estimation of the robot position. For obstacle avoidance, an artificial potential field generates repulsive and attractive force to the robot. Dijkstra algorithm plans the shortest distance path from a start position to a goal point. The methods integrate into autonomous navigation method and implemented for indoor navigation. The experiments show that the proposed method works well for safe autonomous navigation.

키워드

autonomous navigation, mobile robot, map building, localization, obstacle avoidance, path planning

Ⅰ. 서 론

앞으로 가정과 회사에서 이동 로봇은 가족이나 회사 구성원의 한 일원으로 등장하게 될 날이 멀지 않았다. 최근에는 박물관이나, 공공 기관에서 방문객들에게 원하는 정보를 제공하고 전시물을 홍보 및 소개하는 안내로봇의 개발이 활발하게 이루어지고 있다. 또한 위험한 군사지역에서 사람이 탑승하지 않고 원

* 조선대학교 정보통신공학과(nswking0212@naver.com)
** 교신저자 : 조선대학교 제어계측공학과(nyko@chosun.ac.kr)
*** 조선대학교 제어계측공학과(ktg9114@naver.com)
접수일자 : 2010. 12. 06 심사(수정)일자 : 2011. 01. 04 게재확정일자 : 2011. 02. 09

하는 지점까지 이동 할 수 있는 무인자동차가 곧 상용화 될 예정이다. 이처럼 이동 로봇은 단순 반복적인 작업을 하던 산업용 로봇과 달리 사람과 같은 환경을 공유하며 인간 친화적인 지능형 서비스 로봇으로 발전하고 있다. 이를 위해서 이동로봇은 자율 주행이 수행 되어야 한다. 자율주행에 관한 연구 분야는 지도생성(Map Building), 위치파악(Localization), 장애물 회피(Obstacle Avoidance), 경로계획(Path Planning)이 주된 연구 분야이다[1][2]. 자율주행을 위해서는 이러한 요소기술들이 절적하게 통합되어야 한다.

본 논문에서는 자율주행을 위해서 구현된 방법들에 대해서 설명한다. Ⅱ장에서는 맵 작성하는 방법에 대해서 설명을 하고 Ⅲ장에서는 위치추정 알고리즘 중에서 MCL(Monte Carlo Localization)에 대해서 설명한다. Ⅳ장에서는 장애물 회피를 위해서 사용한 포텐셜 필드방법에 대해서 설명을 하고 Ⅴ장에서는 이동로봇이 시작점에서 목표점으로 갈수 있는 경로 계획 방법 중에서 다익스트라 알고리즘을 제시한다. Ⅵ장에서는 구현한 기술요소들을 통합하여 실제 로봇에 통합된 알고리즘의 성능을 실험 및 검증을 한다. 마지막으로 결과 및 차후 연구 진행 방향에 대해서 기술한다.

Ⅱ. 맵 작성

이동로봇의 지도 작성은 이동로봇이 다양한 환경에서 주어지는 목표점까지 자율적으로 이동하기 위해서는 환경에 대한 지도정보가 필요하다. 이동로봇의 지도 작성에 대한 분류는 크게 Geometric map과 Topological map으로 구분하여 볼 수 있다. Geometric map은 로봇주위의 지형을 표현하는데 있어서 주위의 지형을 절대적인 기하학 관계를 이용한다. 이 방법은 물체가 점유된 cell 의 모든 절대 위치로부터 지도를 완성하는 방법이다. Topological map의 경우 센서로부터 관찰된 특징 점의 관계를 기록하고 Node와 Edge로서 표현된다. Node의 경우 관찰된 특징 점을, Edge는 그 특징점 사이의 관계를 나타낸다[3][4].

본 논문에서는 맵 작성의 방법적인 측면 보다 자율주행을 위한 방법 중에 하나로써 지도정보가 필요로 하였다. 따라서 다음 그림1과 같이 실제 도면을 이용하여 맵을 작성하였다.

그림 1. 실제 도면을 이용한 로봇 지도 작성
Fig. 1 Robot Map Building Using Actual Drawing

그림 2. 프로그램에서 사용된 로봇 지도
Fig. 2 Robot Map Used the Program

그림2는 실제 도면을 가지고 각각의 특징 점을 축출하여 프로그램에 적용한 로봇지도이다.

Ⅲ. 위치 추정

이동로봇에 있어 위치추정 기술은 필수적인 기술요소중 하나로 이동로봇이 주어진 환경에서 안내, 물건전달, 주변감시등 작업을 올바르게 수행하기 위해서는 이동로봇은 스스로 자신의 위치를 추정 알 수 있어야 한다[5]. 로봇의 위치 추정을 위해 해결하여야 할 가장 큰 문제점은 센서 신호의 불확실성과 로봇 동작의 불확실성이 있는 조건에서 로봇의 위치를 추정하여야 한다는 점이다. 모든 센서 신호는 정도의 차이는 있지만 반드시 불확실성을 포함하고 있다. 불확실성이 적은 센서는 상대적으로 고가이며 신호 처리에 많은 연산 작업을 필요로 한다. 경제적으로 적당한 가격이고 계산량에서 실시간 신호 처리가 가능한 센서를 사

용하면서 불확실성에 따른 문제를 해결할 수 있어야 한다. 로봇 동작의 불확실성도 고가의 복잡한 시스템을 사용하면 줄일 수 있으나, 로봇 동작의 불확실성은 로봇 자체만의 문제에 의해 발생하는 것이 아니고 로봇 동작 환경에 의해서도 발생하므로 이를 제거하는 것은 불가능하다. 만일 로봇 동작의 불확실성이 없다면 dead-reckoning 만으로도 로봇의 위치를 정확히 추정할 수 있지만, 로봇 동작의 불확실성으로 인하여 dead-reckoning만으로는 실제 사용가능한 위치 추정이 불가능하다고 할 수 있다.

이동로봇의 위치추정에 방법은 데드레크닝, 삼각측량, 칼만 필터, 확장 칼만 필터, MCL 알고리즘등이 있다. 본 논문에선 MCL 알고리즘에 대해서 기술한다.

3.1 MCL을 이용한 위치 추정

MCL 알고리즘은 기존 이동로봇의 위치추정 방법인 데드레크닝(Dead Reckoning), 베이스 필터(Bayes Filter) 그리고 삼각측량(Trialteration)방법들의 문제점들을 확률론적 방법으로 해결 할 수 있는 장점을 지니고 있다.

MCL 알고리즘은 이동로봇의 위치정보를 파티클로 나타낸다. MCL 알고리즘은 전역 위치추정이 가능하고, 확률이 높은 위치에 계산을 집중시킬 수 있는 장점을 지니고 있다.

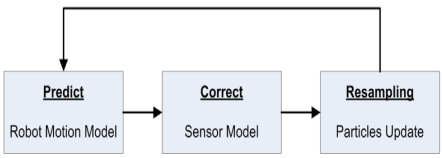

그림 3. MCL 알고리즘 흐름도
Fig. 3 MCL Algorithm Flow Chart

위의 그림3과 같이 MCL 알고리즘은 이동로봇의 모션을 예측하는 “Predict”부, 획득된 센서정보를 통해 파티클의 신뢰도를 생성하는 “Correct”부, 그리고 파티클의 신뢰도를 가지고 파티클을 재생성하는 “Resamping”부로 나눠볼 수 있다.

그림4는 MCL 알고리즘을 나타낸 것이다. 그림4의 MCL 알고리즘을 보면 Sample Motion Model, Measurement Model, Resampling 3단계로 구분된다. 다음 그림에서 Line(3)의 Sample Motion Model은 이전 샘플링에서 생성된 각각의 파티클 위치를 기준으로 현재 속도명령인 직진속도와 회전속도에 의해 각각의 파티클 위치를 예측하는 과정이다.

$$
\begin{aligned}
&\textit{Algorithm MCL}(X_{t-1}, u_t, z_t, m)\\
&\{\\
&1: \quad \bar{X}_t = X_t = \phi\\
&2: \quad \textit{for } m = 1 \textit{ to } M \textit{ do}\\
&3: \quad\quad x_t^{[m]} = \textit{sample\_motion\_model}(u_t, x_{t-1}^{[m]})\\
&4: \quad\quad \omega_t^{[m]} = \textit{measurement\_model}(z_t, x_t^{[m]}, m)\\
&5: \quad \textit{endfor}\\
&6: \quad \textit{for } m = 1 \textit{ to } M \textit{ do}\\
&7: \quad\quad \textit{draw } i \textit{ with probability} \propto \omega_t^{[i]}\\
&8: \quad\quad \textit{add } x_t^{[i]} \textit{ to } X_t\\
&9: \quad \textit{endfor}\\
&10: \quad \textit{return } X_t\\
&\}
\end{aligned}
$$

그림 4. MCL 알고리즘
Fig. 4 MCL Algorithm

Line(4)에서 Measurement Model은 추정신뢰도를 계산하는 부분으로서 로봇의 현재 위치(pose)에서 수신된 센서정보와 지도정보를 가지고 파티클의 신뢰도를 계산한다. Line(6)~(9)은 파티클에 대한 신뢰도를 기반으로 추정 파티클들을 선별하는 과정이다. 이러한 과정을 매 샘플링마다 수행하여 파티클들의 분포에 의해 로봇의 위치가 추정된다.

3.2 MCL 알고리즘 모션 모델

모션 모델은 이동로봇에 대한 직진속도와 회전속도의 명령에 따른 파티클 위치를 예측하는 알고리즘이다. MCL 알고리즘은 이동로봇이 위치할 확률을 파티클들을 통해 표현한다.

아래 그림5에서 Sample Motion Model Velocity는 이전 파티클들이 현재 속도 정보에 대응하여 파티클들이 이동하는 모션모델 알고리즘으로 파라미터에 따라 파티클들에 불확실성을 추가하여 이동로봇에 대한 동작을 나타낸다. 즉, 모션 모델은 직진, 회전 속도 정보와 더불어 $\alpha 1 \sim \alpha 6$ 의 파라미터에 따라 파티클의 동작에 대해 수학적으로 표현하기 위한 모델이다.

그림5의 u_t는 이동로봇의 속도정보, x_{t-1}은 시각 $t-1$에서의 파티클의 위치(x, y, θ)를 나타낸다. Sample Normal distribution 함수는 속도명령 파라미터인 $\alpha 1 \sim \alpha 6$에 대한 파티클 동작의 불확실성을 추가하는

알고리즘이다.

$$
\begin{aligned}
&Sample\_Motion\_Model\_Velocity(u_t, x_{t-1})\\
&\{\\
&\quad \hat{v} = v + sample(\alpha_1 v^2 + \alpha_2 \omega^2)\\
&\quad \hat{\omega} = w + sample(\alpha_3 v^2 + \alpha_4 \omega^2)\\
&\quad \hat{\gamma} = sample(\alpha_5 v^2 + \alpha_6 \omega^2)\\
&\quad x' = x - \frac{\hat{v}}{\hat{\omega}}\sin\theta + \frac{\hat{v}}{\hat{\omega}}\sin(\theta + \hat{\omega}\Delta t)\\
&\quad y' = y + \frac{\hat{v}}{\hat{\omega}}\cos\theta - \frac{\hat{v}}{\hat{\omega}}\cos(\theta + \hat{\omega}\Delta t)\\
&\quad \theta' = \theta + \hat{\omega}\Delta t + \hat{\gamma}\Delta t\\
&\quad return \quad x_t = (x', y', \theta')^T\\
&\}
\end{aligned}
$$

$$
\begin{aligned}
&Sample\_Normal\_distribution(b^2)\\
&\{\\
&\quad return \quad \frac{1}{2}\sum_{i=1}^{12} random(-b, b)\\
&\}
\end{aligned}
$$

그림 5. MCL 모션 모델
Fig. 5 MCL Motion Model

3.3 MCL 알고리즘의 센서모델

센서모델 알고리즘은 Sample Motion Model에서 생성하였던 각각의 샘플들에 대해 가중치를 부여하는 알고리즘으로 4개의 분포도(Measurement Noise, Unexpected Object, Sensing Failure, Unexplainable Measurement)로 구성된다.

Measurement Noise는 Z_t^{k*}를 평균, σhit의 표준편차를 갖는 식 (1)과 같이 가우시안 분포로 모델링 될 수 있다. η는 정규분포 $N(z_t^k; z_t^{k*}, \sigma_t^k)$의 합이 항상 1이 되도록 해주는 변수이다.

$$
p_{hit}(z_t^k \mid x_t, m) = \begin{cases} \eta N(z_t^k; z_t^{k*}, \sigma^2) & if\ 0 \le z_t^k \le z_{max} \\ 0 & otherwise \end{cases}
$$

$$
N(z_t^k; z_t^{k*}, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma_{hit}^2}} e^{-\frac{(z_t^k - z_t^{k*})^2}{2\sigma_{hit}^2}} \tag{1}
$$

Unexpected Object는 다음 수식2처럼 주변 환경에 이동체가 감지될 확률로 지수 확률분포로 표현할 수 있다.

$$
p_{short}(z_t^k \mid x_t, m) = \begin{cases} \eta \lambda_{short} e^{-\lambda_{short} z_t^k} & if\ 0 \le z_t^k \le z_t^{k*} \\ 0 & otherwise \end{cases} \tag{2}
$$

만일 센서의 신호가 어떤 알 수 없는 이유에 의해 측정되지 않았다면 센서의 값은 그 최대값으로 나타나게 된다. 예를 들자면, 초음파센서의 경우에서는 물체에 반사되는 각도에 따라서 이런 경우가 많이 발생할 수 있으며, 레이저 센서의 경우에서는 유리로 되어 있는 벽을 감지할 경우 레이저가 투과하고 돌아오지 못하는 경우가 생길 수 있다. 랜덤한 에러는 센서의 거리 값의 모든 영역에 존재하는 것으로 가정한다.

Sensing Failure는 다음식3과 같이 센서가 감지하지 못했을 경우를 표현되어진다.

$$
p_{max}(z_t^k \mid x_t, m) = I(z = z_{max}) \begin{cases} 1 & if\ z = z_{max} \\ 0 & otherwise \end{cases} \tag{3}
$$

Unexplainable Measurement의 경우 벽의 반사나 혼선으로 인한 센서의 값이 잘못된 확률분포를 나타낸 것으로 아래 식4와 같이 표현한다.

$$
p_{rand}(z_t^k \mid x_t, m) = \begin{cases} \dfrac{1}{z_{max}} & if\ 0 \le z_t^k \le z_{max} \\ 0 & otherwise \end{cases} \tag{4}
$$

아래의 5의 식은 각각의 센서모델 알고리즘 하나의 확률분포도로 표현한 것으로 4개의 분포도를 적용하였을 경우, 그림8과 같은 결과를 얻을 수 있다.

$$
p(z_t^k \mid x_t, m) = \begin{pmatrix} z_{hit} \\ z_{short} \\ z_{max} \\ z_{ramd} \end{pmatrix}^T \cdot \begin{pmatrix} p_{hit}(z_t^k \mid x_t, m) \\ p_{short}(z_t^k \mid x_t, m) \\ p_{max}(z_t^k \mid x_t, m) \\ p_{rand}(z_t^k \mid x_t, m) \end{pmatrix}
$$

$$
cf)\ z_{hit} + z_{short} + z_{max} + z_{rand} = 1 \tag{5}
$$

그림6은 이동로봇의 센서값을 확률변수로 하여 파티클과 지도정보를 바탕으로 신뢰도를 추출하는 알고리즘이다.

$Weight\_of\_OneParticle\_Acquisition\_Model(z_t, x_t, m)$
{
  $q = 1$
  for $k = 1$ *to* K *do*
    compute z_t^{k*} *for the measurement* z_t^k *using ray casting*
    $p = z_{hit} \cdot p_{hit}(z_t^k | x_t^k, m) + z_{short} \cdot p_{short}(z_t^k | x_t^k, m)$
      $+ z_{max} \cdot p_{max}(z_t^k | x_t^k, m) + z_{rand} \cdot p_{rand}(z_t^k | x_t^k, m)$
    $q = q \cdot p$
  endfor
}

z_t : *Sensor Data*
x_t : *Particle Information*
m : *Map Information*

그림 6. MCL 센서 모델
Fig. 6 MCL Sensor Model

3.4 MCL 알고리즘의 리샘플링 모델

리샘플링은 센서신호에 근거해 이동로봇의 위치를 추정하는 과정이다. 다음 알고리즘과 같이 룰렛 방법[6]을 적용하여 파티클의 신뢰도에 따라 파티클을 복제하거나 제거하는 과정이다.

```
Roulette_Method_Particle_Select(Pn, Pb)
{
  Draw Roullet(Pb)
  for 1 to Pn do
      Shoot an Arrow to the Roulette
      Draw Selected Particle
  end for
}
```

그림 7. MCL 리샘플링 모델
Fig. 7 MCL Resampling Model

Ⅳ. 장애물 회피

이동로봇이 정해진 목표지점에 이동하기 위해서는 이동로봇의 이동경로 상에 존재하는 장애물을 인지하여 회피를 하여야 한다. 특히 이동 로봇이 작업하는 환경이 다양한 형태의 장애물이 산재한 공간일 경우, 이에 대한 효과적인 장애물 검출과 회피가 필요하다[7]. 장애물 회피 알고리즘은 센서의 특징을 잘 반영하여야 하고, 신속하며 정확한 장애물 데이터를 얻어내어 그 결과를 알고리즘에 효율적으로 사용하여야 한다[8]. 현재에도 장애물 회피 방법에 대한 여러 연구가 활발히 진행되어지고 있고 있다. 본 장에서는 전위계 함수(Potential Filed)방법에 대해서 설명한다.

4.1 전위계 함수 방법

전위계 함수(Potential Field Method)의 주요한 원리는 로봇이 존재하는 직교 좌표계로 표현된 공간에서 점 또는 단위 셀(Cell)을 설정하는 방법으로서, Khatib[9]와 Arkin[10]의 사례에서 찾아볼 수 있다. 이 점 또는 단위 셀은 장애물이 검출된 격자로부터 로봇을 밀어내는 척력, 그리고 목표점으로부터 로봇에 작용하는 인력의 정도 값을 지니고 있다. 로봇은 이와 같은 점 또는 셀로부터 생성된 가상의 힘 벡터를 모두 합성한 합성 벡터의 방향으로 진행하게 된다. 그림 8은 포텐셜 필드적용 방법에 대해서 나타내었다.

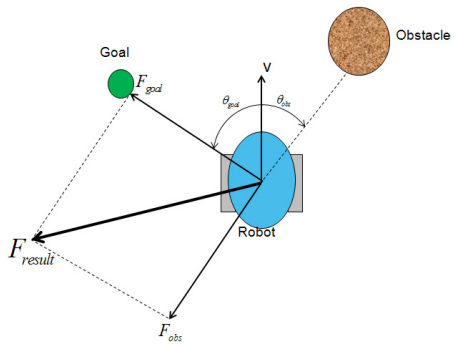

그림 8. 포텐셜 필드 방법
Fig. 8 Potential Field Method

여기서 척력은 장애물과 거리가 멀면 멀수록 힘은 작아져야하며, 인력은 거리에 비례해서 커져야 한다. 우선 척력의 식을 나타내면 식6과 같다.

$$\overrightarrow{F_{obs}} = -A \cdot \sum_{i=1}^{n} \frac{1}{d_i^2} \cdot \hat{r_i} \quad (6)$$

A는 constant scaling factor이며 실험을 통하여 값을 결정해준다. n은 장애물의 수, d_i는 장애물i와 로봇 사이의 거리, $\hat{r_i}$는 로봇에서 장애물로의 단위 방향 벡터이다. 이와 같이 장애물 등의 척력은 거리가 가까워질수록 무한대로 증가하게 된다.

인력에 관한 식을 나타내면 식7과 같다.

$$\overrightarrow{F_{goal}} = B \bullet d^2 \bullet \hat{r} \quad (7)$$

여기서 B는 역시 constant scaling factor이며 실험적으로 적절한 값을 결정해준다. d는 로봇과 목표 지점 사이의 거리, $\hat{r}$은 로봇에서 목표 지점으로의 단위 방향 벡터이다.

Ⅴ. 경로 계획

이동로봇의 경로 계획은 이동 로봇이 정해진 지도상에서 주어진 초기 위치로부터 목적지까지 찾아가는 경로를 말한다. 이처럼 주어진 공간상의 시작점에서 목표점까지 경로를 얻는 문제는 로봇 분야에서 많은 연구가 진행되어져 오고 있다[11].

경로계획 방법은 크게 전역 경로계획과 지역 경로계획으로 나눌 수 있다. 전역 경로계획은 미리 주어진 장애물 지도를 기반으로 출발점에서 목표점까지 장애물과의 충돌을 피하면서 가장 빠르게 갈 수 있는 경로를 찾는 방법이다. 전역 경로계획은 크게 Road Map방법, Cell Decomposition 방법, Potential Method 방법 등이 있다. 지역 경로계획은 지도가 없는 미지의 환경을 이동하거나, 이미 작성된 지도를 이용한 전역 경로계획에 따라서 이동로봇이 이동할 때 지도에 나와 있지 않은 장애물이나 이동 장애물을 피하기 위하여 실시간 센서 정보를 이용하여 국부적으로 경로를 재생성하는 방법이다. 제안된 방법들은 버그 알고리즘(Bug algorithms), 벡터 필드 히스토그램 방법(Vector Filed Histogram) 등이 있다. 본 절에서는 Cell Decomposition방법을 이용하고 최단거리를 구하는 방법 중의 하나인 다익스트라(Dijkstra) 알고리즘을 제안한다.

5.1 다익스트라 알고리즘

Dijkstra의 최단 경로 알고리즘[12]은 네트워크에서 하나의 시작 정점으로부터 모든 다른 정점까지의 최단 경로를 찾는 알고리즘이다. 최단경로는 경로의 길이 순으로 구해진다. 먼저 집합 S를 시작 정점 v로부터의 최단경로가 이미 발견된 점점들의 집합이라고 한다. 아래의 그림9와 같이 Dijkstra의 알고리즘에서는 시작 정점에서 집합 에 있는 정점만을 거쳐서 다른 정점으로 가는 최단거리를 기록하는 배열이 반드시 있어야 한다. 이 1차원 배열을 distance라고 한다. 시작 정점을 v이라 하면 distance[v] = 0이고 다른 정점에 대한 distance 값은 시작정점과 해당 정점간의 가중치 값이 된다. 가중치는 보통 인접 행렬에 저장되므로 인접 행렬을 weight이라 하면 distance[w] = weight[v][w]가 된다. 정점v에서 정점w로의 직접 간선이 없을 경우에는 무한대의 값을 저장한다. 시작단계에서는 아직 최단경로가 발견된 정점이 없으므로 S={v}일 것이다. 즉 처음에는 시작 정점 v를 제외하고는 최단거리가 알려진 정점이 없다. 알고리즘이 진행되면서 최단거리가 발견되는 정점들이 S 에 하나씩 추가될 것이다.

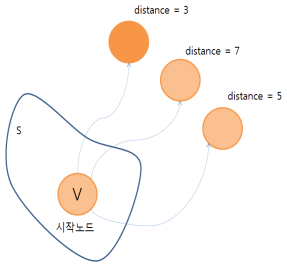

그림 9. 다익스트라 알고리즘 1단계
Fig. 9 Dijkstra Algorithm 1Step

그림9와 같이 알고리즘의 1 단계에서 S안에 있지 않은 정점 중에서 가장 distance 값이 작은 정점을 S에 추가 한다.

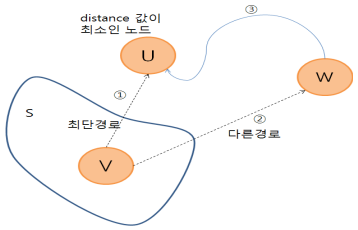

그림 10. 다익스트라 알고리즘에서의 최단 경로 증명
Fig. 10 The shortest path in Dijkstra's algorithm proof

그 이유를 다음 그림10에서 보면, 현재 S에 들어있지 않는 정점 중에서 distance 값이 가장 작은 정

점을 u라고 하자. 그러면 시작 정점 v에서 정점 u까지의 최단거리는 경로 1이 된다. 만약 짧은 경로, 예를 들어 정점 w를 거쳐서 가는 가상적인 더 짧은 경로가 있다고 가정해 보자. 그러면 정점 v에서의 정점 u까지의 거리는 정점 v에서 정점 w까지의 거리 2와 정점 w에서 정점 u로 가는 거리 3을 합한 값이 될 것이다.

그러나 경로 2는 경로 1보다 항상 길 수 밖에 없다. 현재 distance 값이 가장 작은 정점은 u이기 때문이다. 다른 정점은 정점 u까지의 거리보다 항상 더 길 것이다. 따라서 매 단계에서 집합 S에 속하지 않는 정점 중에서 distance 값이 가장 작은 정점들을 추가해나가면 시작 정점 V에서 모든 정점까지의 최단거리를 구할 수 있다.

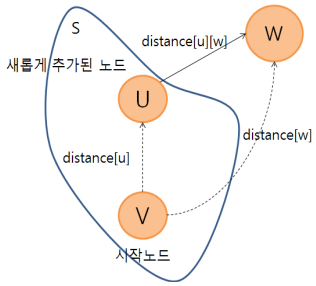

그림 11. 다익스트라 거리 값 갱신
Fig. 11 Dijkstra Distance value Update

그림 11과 같이 새로운 정점 u가 S에 추가되면, S에 있지 않은 다른 정점들의 distance 값을 수정한다. 새로 추가된 정점 u를 거쳐서 정점까지 가는 거리와 기존의 거리를 비교하여 더 작은 거리로 distance 값을 수정한다. 즉 다음과 같은 식8을 이용한다.

$$distance[w] = \min(distance[w], distance[u] + weight[u][w]) \quad (8)$$

그림 12는 네트워크에서 하나의 시작 정점으로부터 모든 다른 정점까지의 최단 경로를 찾는 Dijkstra 알고리즘이다. 최단경로는 경로의 길이 순으로 구해진다.

```
shortestPath(G,v)
  S ← v
  for 각정점 w ∈ G do
    distance[w] ← weight[v][w];
  while 모든정점이 S에포함되지않으면 do
      u ← 집합 S에 속하지 않는정점중에서최소 distance 정점;
      S ← S ∪ u
      for u에인접하고 S에있지않은각정점 z do
        if distance[u] + weight[u][z] < distance[z]
          then distance[z] ← distance[u] + weight[u][z];
```

그림 12. 다익스트라 알고리즘
Fig. 12 Dijkstra Algorithm

Ⅵ. 실험 및 고찰

본 실험에서는 실제 로봇을 이용하여 로봇이 시작점에서 목표점까지 이동하는 동작을 보인다. 지도 정보는 실제 조선대학교 전자정보공과대학 6층 도면을 이용하여 파일에 실제도면의 특징점을 저장하여 지도를 불러오는 방식을 이용한다. 본 실험에서 사용한 거리센서는 Sick사의 LMS200을 사용하였고 이동로봇은 레드원테크놀러지(주)에서 제작한 IRONC 로봇을 이용하였다. 표1은 실제 로봇이 동작을 위해 필요한 환경 정보 인식을 나타내었다.

표 1. 이동로봇 환경정보
Table 1. Environment Information for Navigation

파티클 수	1500개
LRF 장애물 감지 거리	55 cm
센서 최대 거리	30 m
지도 크기(가로*세로)	9944*1896 cm
이동장애물 유무	유
이동로봇 왕복횟수	5회

표1에서 나온바와 같이 LRF센서 장애물 감지거리를 55cm로 상대적으로 짧은 거리를 인식하도록 하였는데 그 이유는 멀리에서부터 장애물을 인식하여 회피를 하면 좋으나 실질적으로는 센서길이를 길게 하면 문 사이를 빠져나갈 수 가 없다. 따라서 상대적으로 장애물 감지가 로봇 가까이에서 감지되도록 거리

를 짧게 하였다. 위의 그림 13는 실제 로봇과 구현된 동작 프로그램을 보인다. (a)는 실제 로봇를 나타내고 (b)는 메인화면으로 카메라 뷰와 로봇의 동작모드를 나타낸다. (c)는 LRF센서 정보를 나타내고 (d)는 로봇이 이동 및 경로 생성을 보여주는 Map 화면이다.

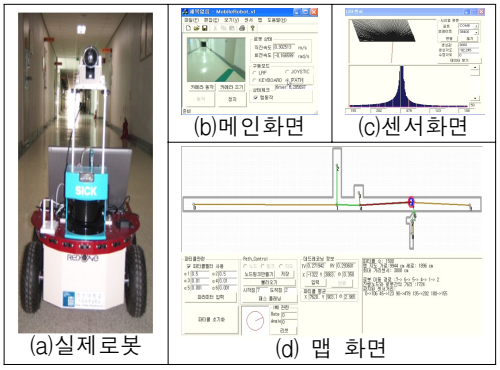

그림 13. 실제 이동로봇과 프로그램
Fig. 13 Real Robot and Program

로봇은 MCL알고리즘을 통하여 현재 위치를 추정하고 시작점과 목표점을 입력하여주면 녹색선분으로 경로가 생성되어 장애물을 회피하면서 목표점으로 이동한다.

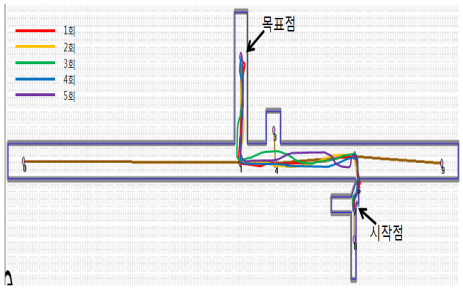

그림 14. 이동로봇의 주행 경로
Fig. 14 Mobile Robot Navigation Path

그림14는 실제 로봇이 주행하면서 목표점까지 이동시에 나타나는 경로 추종화면이다. 본 실험에서는 5번의 반복을 실시하였다.

그림 15에서는 이동로봇이 주행 중 1~4번 주행은 경로에 변함이 없이 추종을 하였으나 보라색으로 표시된 화면은 갑작스런 동적 장애물을 만났을 경우 물체를 회피하면서 경로 추종을 하는 것을 나타낸다.

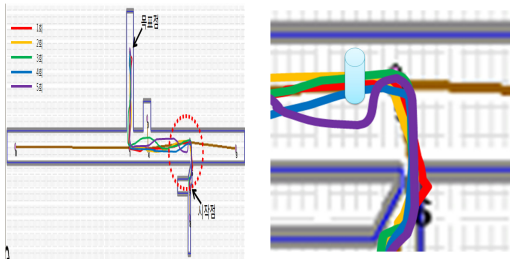

그림 15. 장애물 발생시 이동로봇 주행
Fig. 15 Mobile Robot Navigation In case of Obstacle

Ⅶ. 결 론

본 논문에서는 자율이동로봇의 주행을 시뮬레이터가 아닌 실제 로봇을 이용하여 동작됨을 보였다. 맵 작성, 위치추정, 장애물 회피, 경로계획, 경로 추정 방법들에 대해서 만족할 만한 결과를 보였다. 또한, 프로그램에서는 다양한 구동모드를 형성하여 조이스틱, 물체회피, 키보드제어, 네비게이션 모드로 동작 할 수 있고 원격제어를 통하여 현재 로봇이 이동하고 있는 지점이 어디인지를 바로 확인 할 수 있다.

향후 연구에서는 맵 제작을 레이저 센서를 이용하여 지도를 제작 할 수 있는 Map Building에 대해서 연구를 하고 각각의 제안된 알고리즘의 성능평가를 검증하여 로봇의 안전한 주행을 할 수 있도록 진행할 예정이다.

감사의 글

본 연구는 교육과학기술부, 한국연구재단의 2010년 지역혁신인력양성사업(과제명: 로봇의 자율주행 요소기술 상용화 및 인력양성, No. I00061)의 지원에 의해 이루어짐.

참고 문헌

[1] S. Thrun, “Learning metric-topological maps for indoor mobile robot navigation,” Artificial Intelligence, vol. 99, no. 1, pp. 21 - 71, 1998.
[2] Borenstein, J,. Everett, B,.and Feng, L.,

"Where am I? Sensors and Methods for Mobile Robot Positioning." pp. 130-131, 1996.
[3] Moravec, H.P. and Elfes, A., "High Resolution Maps from Wide Angle Sonal." Proceeding of the IEEE conference on Robotics and Automation, Wasington D.C., pp. 116-121, 1985.
[4] Borenstein, J. and Koren, Y., 1991, "Histogramic in-motion mapping for mobile robot obstacle avoidance." IEEE Journal on Robotics and Automation, Vol. 7, No. 4, pp. 525-539, 1991.
[5] Borenstein, J. and Feng, L., where am I? Sensors and Method for Autonomous Mobile Robot Positioning-1995 Edition, 1995c.
[6] 김태균, "유비쿼터스 센서 환경에서의 이동로봇 위치추정," 조선대학교 석사학위 논문, 2009.
[7] 김성철, "자율 이동 로봇의 장애물 회피를 위한 안전 방향 구간 탐색 방법," 조선대학교 박사학위 논문, 2002.
[8] Ko, N. Y. and Simmons, R. G., "The Lane-curvature Method for Local Obstacle Avoidance," International Conference in Intelligent Robots and Systems (IROS 1998), Victoria, B.C., Canada, Oct. 13-17, 1998.
[9] O.Khatib, "Real-Time Obstacle Avodance for Manipulators and Mobile Robots." IEEE International Conference on Robotics and Automation, pp. 500-505, 1985.
[10] Arkin, R. C., "Motor Schema-Based Mobile Robot Navigation," The International Journal of Robotics Research: 92-112, 1989.
[11] N.C. Tsourveloudis, K.P. Valavanis, and T. Hebert, "Autonomous Vehicle Navigation Utilizing Electrostatic Potential Fields and Fuzzy Logic," IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION, VOL. 17, NO. 4, pp. 490-497, 2001.
[12] Data Structures in C 언어로 쉽게 풀어쓴 자료구조, 천인국. p426-436

저자 소개

노성우(Sung-woo Noh)

2007년 조선대학교 제어계측공학과 졸업(공학사)
2010년 조선대학교 대학원 제어계측공학과 졸업(공학석사)
2010년~현재 조선대학교 대학원 정보통신공학과 (박사과정)
※ 관심분야 : 이동로봇, 경로계획, Microprocessor

고낙용(Nak-yong Ko)

1985년 서울대학교 제어계측공학과 졸업(공학사)
1987년 서울대학교 대학원 제어계측공학과 졸업(공학석사)
1993년 서울대학교 대학원 제어계측공학과 졸업 (공학박사)
1997~1998, 2004~2005 미국 Carnegie Mellon Univ. Visiting research scientist
1992년~현재 조선대학교 제어계측로봇공학과 교수
※ 관심분야 : 지상로봇과 수중로봇의 자율주행

김태균(Tae-gyun Kim)

2007년 조선대학교 제어계측공학과 졸업(공학사)
2009년 조선대학교 대학원 제어계측공학과 졸업(공학석사)
2009년~현재 조선대학교 대학원 제어계측학과 (박사과정)
※ 관심분야 : 이동로봇, 수중로봇, 위치추정