Journal of Korean Institute of Intelligent Systems, Vol. 23, No. 4, August 2013, pp. 332-340
http://dx.doi.org/10.5391/JKIIS.2013.23.4.332

이동 로봇 위치 추정 및 시뮬레이션 프로그래밍 툴킷

Programming Toolkit for Localization and Simulation of a Mobile Robot

정석기* · 김태균** · 고낙용***†
Seok Ki Jeong*, Tae Gyun Kim**, and Nak Yong Ko***†

*조선대학교 제어계측공학과
**한국해양과학기술원 해양시스템연구부
***† 조선대학교 제어계측로봇공학과
† Dept. Control, Instrumentation, and Robot Engineering, Chosun University

요 약

본 논문은 실제 환경과 모의실험에서 이동 로봇의 위치 추정과 자율주행 구현을 위한 프로그래밍 툴킷에 대해 서술한다. 기존에 사용되고 있는 라이브러리들은 복잡성과 유용성의 결함으로 사용에 어려움이 있다. 제안된 툴킷은 추측항법, 운동 모델, 측정 모델, 그리고 방향 또는 지향각의 연산을 위한 툴킷들로 구성된다. 추측 항법과 운동 모델은 차륜 구동 로봇과 전, 후륜 속도에 의한 이륜차 로봇에 대해 다룬다. 툴킷들은 실제 환경과 모의실험에서의 자율주행을 위해 사용 가능하다. 툴킷의 사용가능성은 모의실험의 결과와 실제 실험의 결과를 보임으로써 증명한다. 제안된 툴킷은 이동 로봇의 위치추정, 지도 작성, 그리고 장애물 회피와 같은 자율주행의 구성 기술을 위한 알고리즘의 검사에 사용할 수 있을 것으로 기대된다.

키워드 : 이동 로봇, 위치 추정 라이브러리, 로봇 시뮬레이션, 차륜 구동 로봇, 이륜차 로봇 모델

Abstract

This paper reports a programming toolkit for implementing localization and navigation of a mobile robot both in real world and simulation. Many of the previous function libraries are difficult to use because of their complexity or lack of usability. The proposed toolkit consist of functions for dead reckoning, motion model, measurement model, and operations on directions or heading angles. The dead reckoning and motion model deals with differential drive robot and bicycle type robot driven by front wheel or rear wheel. The functions can be used for navigation in both real environment and simulation. To prove the feasibility of the toolkit, simulation results are shown along with the results in real environment. It is expected the proposed toolkit is used for test of algorithms for mobile robot navigation such as localization, map building, and obstacle avoidance.

Key Words : Mobile Robot, Localization Library, Robot Simulation, Differential Drive Robot, Bicycle Robot Model

1. 서 론

이동 로봇은 자율주행을 위해 지도 작성, 위치 인식 기술, 경로 계획 기술, 그리고 회피기술들이 필요하다. 자율주행 요소 기술 중 위치 인식 기술은 가장 기본적이고 필수적인 기술이다. 로봇이 스스로의 위치를 알지 못하면 지도상에 어디에 존재하고 있는지 알 수 없고, 목적지까지의 경로 설정이 불가능해 주행을 수행하기 어렵다. 그러므로 이동 로봇의 위치 인식은 필수적 핵심 기술이다[1]. 위치 추정 기술은 연구자 또는 적용하려는 로봇에 따라 다양한 방법으로 개발되었다. 본 논문은 기존 라이브러리들과 달리 개발된 툴킷에 필요한 기본 요소들과 툴킷들에 대한 이론적 배경을 설명으로 이해를 돕는다. 본 연구를 통해 주행 기술의 위치 추정에 관련된 툴킷과 툴킷을 적용한 시뮬레이션 구현에 대해 설명한다.

기존의 이동 로봇의 자율주행을 위한 라이브러리는 uRON (Universal Robot Navigation)[2], CARMEN(Carnegie Mellon Robot Navigation Toolkit)[2-3], MRPT(Mobile Robot Programming Toolkit)[2][4-5], Robotics Toolbox for Matlab[6-7], 그리고 Karto SDK[2][8] 등이 있다. 각각의 라이브러리는 적용된 알고리즘, 작업환경, 적용되는 로봇, 사용방법, 기본 구성 요소 등의 제약으로 사용자가 쉽게 습득하

접수일자: 2013년 2월 5일
심사(수정)일자: 2013년 4월 11일
게재확정일자 : 2013년 4월 12일
† Corresponding author
이 논문은 2013년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. NRF-2013R1A1A4A01012469)에서 지원하여 연구하였습니다. 연구비 지원에 감사합니다.
This research was supported by Basic Science Research Program through the National Research Foundation of Korea(NRF) funded by the Ministry of Education(NRF-2013R1A1A4A01012469).

고 적용하기에 어려움을 가진다.

CARMEN은 이동 로봇 제어를 위한 소프트웨어의 공개 소스 모음집이다. 이 툴킷은 구동부와 센서 제어, 센서 정보 기록, 장애물 회피, 위치 추정, 경로 계획, 그리고 지도 작성의 기초적인 자율주행의 기술들을 고안한 모듈 방식의 소프트웨어다[3]. 그러나 CARMEN은 IPC(Inter Process Communication)를 다룰 줄 알아야 한다는 단점이 있다. MRPT는 라이브러리 구조가 복잡해 사용자가 필요한 기술을 검색하기 어렵다. 또한 사용자가 함수를 찾더라도 상호 작용되는 함수 및 입출력 정보들에 대한 분석을 단점으로 가진다.

Robotics Toolbox for Matlab은 Matlab 프로그램을 사용해 모의실험을 할 수 있는 라이브러리이다. 이 라이브러리는 추측 항법, 확장 칼만 필터 알고리즘을 적용한 위치 추정, 주행 경로를 오차 없이 주행하는 가정으로 랜드 마크(Land Mark)의 위치를 찾는 지도 작성, 확장 칼만 필터를 적용한 SLAM(Simultaneous Localization and Mapping), 그리고 몬테카를로 알고리즘(Monte Carlo Algorithm)을 적용한 위치 추정 등을 모의실험을 할 수 있도록 개발됐다. 이와 같이 Robotics Toolbox for Matlab은 위치 추정 모의실험을 다양한 방법을 통해 시도할 수 있는 장점이 있다. 하지만 사용자의 목적에 부합되는 모의실험을 위해서 사용자가 원하는 로봇의 구성요소, 지도 정보, 장애물 등을 위해 소스 코드의 분석과 수정을 필요로 한다[9].

본 논문은 기존 라이브러리와 차별성을 위해 위치 추정 툴킷들의 배경 이론 및 입출력 정보를 기술한다. 기술된 내용을 통해 사용자는 위치 추정 기술을 습득하고 사용 가능하며 위치 추정 알고리즘 구현 시 시간소모를 줄이는 데 도움이 된다. 시뮬레이션 툴킷은 개발된 위치 추정 툴킷들을 사용해 구현했다. 로봇이 작동하는 중 소프트웨어 또는 하드웨어의 결함으로 심각한 파손 및 손실로 비용과 시간 등에 피해를 입을 수 있다. 시뮬레이션은 이런 문제를 사전에 방지하기 위해 필요하다[10]. 본 연구를 통해 구현된 시뮬레이션 툴킷은 언급된 실제 상황에서의 문제를 해결하기 위해 개발되었다.

본 논문의 2장에서는 차륜 구동 로봇과 이륜차 로봇 모델에 적용 가능토록 구현된 추측 항법과 운동 모델 툴킷, 외부 환경 정보를 인지하기 위한 측정 모델 툴킷, 방향의 연산에 대한 툴킷을 설명한다. 3장에서는 위치 추정 툴킷들을 적용한 시뮬레이션 툴킷에 대해 서술한다. 4장에서는 시뮬레이션 툴킷과 실제 실험에 대한 실험을 기술한다. 5장에서는 본 논문의 결론을 맺는다.

2. 위치 추정 툴킷

2.1 추측 항법

추측 항법은 로봇의 이전 위치 정보와 로봇의 내부 정보를 이용하여 시간 간격 후의 로봇의 위치를 추정하는 방법이다. 식 (1)은 시각 t에서 로봇의 위치를 나타내는 위치 정보인 x_t, y_t, θ_t를 갖는 X_t에 대한 식이다. x_t, y_t는 직교좌표상의 로봇의 좌표 값이다. 그리고 θ_t는 로봇의 방향이 직교좌표의 x축과 이루는 사잇각을 나타낸다.

$$X_t = [x_t\, y_t\, \theta_t]^T \tag{1}$$

로봇의 내부 정보를 이용한 로봇의 속도 정보는 모터의 엔코더 데이터를 이용하여 구할 수 있다. 획득한 속도 정보와 주행 시간으로 위치를 추정할 수 있다. 본 절에서는 이륜차 로봇과 차륜 구동 로봇 대한 추측 항법에 대해 기술한다.

그림 1은 차륜 구동 로봇의 구조와 운동에 대해 나타낸다. 로봇의 두 바퀴의 반지름은 같고, 두 바퀴의 중심 사이 거리는 d, 좌측 바퀴의 속도와 우측 바퀴의 속도는 v_l, v_r임을 보이고 있다. 로봇의 주행 속도는 v, 회전 속도는 w로 나타내고 있다. *ICR(instantaneous center of rotation)*은 이동 로봇이 일정 시간 동안 v와 w의 속도로 주행한 경우 이전 위치와 현재 위치에서 로봇의 두 바퀴의 중심을 지나는 연장선이 교차하는 지점이다[11]. 좌우 바퀴의 이동 속도는 라디안 단위의 중심각과 반지름의 곱이 호의 길이와 같음을 이용해 구한다. w는 로봇의 회전 속도이다. w는 v_l와 v_r의 차와 R을 이용해 구할 수 있다. v는 v_l와 v_r의 합을 반으로 나눈 값이다. 로봇 좌우 바퀴의 각속도 w_l, w_r과 두 모터의 중심 사이의 거리 d를 획득 가능한 경우 직진 속도 v와 회전 속도 w를 구할 수 있다.

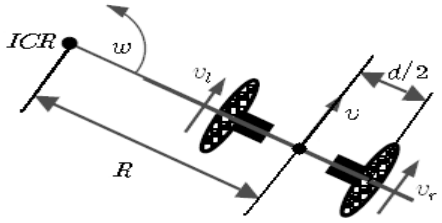


그림 1. 차륜 구동 이동 로봇의 이동
Fig. 1. Motion of a differential drive mobile robot

표 1. 차륜 구동 로봇의 추측 항법 툴킷
Table 1. Dead reckoning toolkit of differential drive robot

Algorithm ***DR_DD***$(X_{t-1},\ v,\ w,\ \Delta t)$	
1:	*if* $w \neq 0$
2:	$x_t = x_{t-1} - \frac{v}{w}(\sin\theta_{t-1} - \sin(\theta_{t-1} + w\Delta t))$
3:	$y_t = y_{t-1} + \frac{v}{w}(\cos\theta_{t-1} - \cos(\theta_{t-1} + w\Delta t))$
4:	$\theta_t = \theta_{t-1} + w\Delta t$
5:	*otherwise*
6:	$x_t = x_{t-1} + v\cos\theta_{t-1}\Delta t$
7:	$y_t = y_{t-1} + v\sin\theta_{t-1}\Delta t$
8:	$\theta_t = \theta_{t-1}$
9:	*return* X_t

표 1은 차륜 구동 로봇의 추측 항법 툴킷인 *DR_DD*의 의사코드를 나타낸다. 구현된 툴킷은 입력 정보로 이전 위치 정보 X_{t-1}, 직진 속도 v와 회전 속도 w, 그리고 시간 간격 Δt를 입력받는다. 툴킷의 결과로 추정된 위치 정보 X_t를 반환한다.

이륜차 로봇은 그림 2와 같이 나타낼 수 있다. 그림 2에서는 뒷바퀴의 좌표 정보인 x, y와 x축과 로봇 몸체가 이루는 각인 로봇의 지향 각 θ, 로봇몸체와 앞바퀴의 사잇각 ϕ에 대해 나타낸다[12].

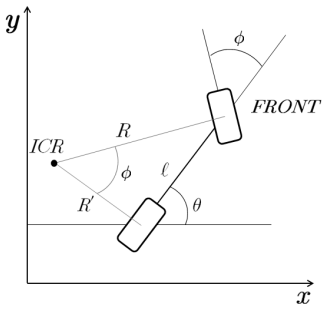


그림 2. 이륜차형 이동로봇의 기구학
Fig. 2. Kinematics of bicycle type mobile robot

표 2. 이륜차 로봇의 추측 항법 툴킷
Table 2. Dead reckoning toolkit of bicycle type robot

Toolkit Name	***DR_BF***
Inputs	X_{t-1}, v_f, ϕ, ℓ, Δt
Process	Apply *DR_DD* toolkit after calculating w using ϕ
Results	X_t
Toolkit Name	***DR_BR***
Inputs	X_{t-1}, v_r, ϕ, ℓ, Δt
Process	Apply *DR_DD* toolkit after calculating w using ϕ
Results	X_t

표 2는 앞바퀴 또는 뒷바퀴의 속도를 이용한 이륜차 로봇의 추측 항법을 나타낸다. 툴킷 명칭은 *DR_BF*와 *DR_BR*이라 정의한다. 툴킷 사용을 위한 입력 정보는 이전 위치 정보 X_{t-1}와 앞바퀴의 속도 정보 v_f 또는 v_r, 로봇의 몸체와 앞바퀴의 사잇각 ϕ, 로봇의 앞바퀴와 뒷바퀴 사이거리 ℓ, 그리고 시간 간격 Δt가 있다. θ의 변화인 $\dot{\theta}$는 로봇 몸체의 방향과 x축과의 사잇각의 변화를 나타낸다. 이 변화는 차륜 구동 로봇의 회전 속도인 w와 같으므로 *DR_DD* 툴킷의 재사용이 가능하다. 각 툴킷의 결과는 재사용된 *DR_DD* 툴킷에서 획득한 현재의 추정 위치 정보 X_t를 반환한다. 이륜차 로봇은 비홀로노믹 시스템이며, 이는 적분 불가능한 구속조건으로부터 도출되는 시스템을 의미한다[13]. x축과 로봇 방향이 이루는 사잇각의 변화인 $\dot{\theta}$는 비홀로노믹 시스템을 통해 구할 수 있다.

2.2 운동 모델

추측 항법은 로봇의 내부 정보만으로 로봇의 위치를 추정하는 방법이다. 로봇이 이동 시 내부 또는 외부의 오류로 실제 이동 거리와 추측 항법의 추정 위치는 오차를 갖게 된다. 그리고 시간이 지남에 따라 로봇의 추정 위치는 누적된 오차를 가진다. 로봇 동작의 오차는 모터의 엔코더로부터 얻게 되는 속도 정보의 오차, 바퀴와 바닥면의 마찰력에 의한 미끄러짐에 대한 오차, 그리고 바퀴의 공기압으로 인한 바퀴 반지름의 오차 등이 있다. 운동 모델에 적용한 속도 정보의 오차는 정규 확률분포를 이용한다.

표 3. 정규 분포 툴킷
Table 3. Normal distribution toolkit

Algorithm ***NormalDistribution(b)***	
1:	*return* $\frac{1}{2}\sum_{i=1}^{12} rand(b,-b)$

정규 확률 분포의 확률 변수를 구하는 툴킷의 명칭을 *NormalDistribution*이라 정의한다. 표 3은 이 툴킷의 의사코드를 나타낸다. 표 3의 b는 실제 실험을 통해 얻은 운동 오차들의 표준편차를 구해서 적용한다. *rand*함수는 b에서 $-b$까지 균등 분포를 따르는 임의의 값을 출력하는 함수다. 식 (2)는 오차 정보들을 가지는 A를 나타낸다. A는 로봇의 속도 정보와 로봇의 방향에 대한 오차를 가진다.

$$A = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4\ \alpha_5\ \alpha_6] \tag{2}$$

표 4. 차륜 구동 로봇의 운동 모델 툴킷
Table 4. Motion model toolkit of differential drive robot

Algorithm ***MM_DD***$(X_{t-1},\ v,\ w,\ \Delta t,\ A)$	
1:	$\hat{v} = v + sample(\alpha_1 \lvert v^2 \rvert + \alpha_2 \lvert w^2 \rvert)$
2:	$\hat{w} = v + sample(\alpha_3 \lvert v^2 \rvert + \alpha_4 \lvert w^2 \rvert)$
3:	$\hat{\gamma} = sample(\alpha_5 \lvert v^2 \rvert + \alpha_6 \lvert w^2 \rvert)$
4:	$x_t = x_{t-1} - \frac{\hat{v}}{\hat{w}}\left(\sin\theta_{t-1} - \sin\left(\theta_{t-1} + \hat{w}\Delta t\right)\right)$
5:	$y_t = y_{t-1} + \frac{\hat{v}}{\hat{w}}\left(\cos\theta_{t-1} - \cos\left(\theta_{t-1} + \hat{w}\Delta t\right)\right)$
6:	$\theta_t = \theta_{t-1} + \hat{w}\Delta t + \hat{\gamma}\Delta t$
9:	*return* X_t

표 4의 차륜 구동 로봇의 운동 모델 툴킷은 *MM_DD*라 정의하고 의사코드로 표현한다[14]. 툴킷의 입력 정보는 이전 위치 정보 X_{t-1}과 속도 정보 v와 w, 시간 간격 Δt, 그리고 오차 정보인 A를 입력받는다. *MM_DD* 툴킷의 출력은 위치 정보 X_t를 반환한다. 표 4의 *sample*함수는 *NormalDistribution* 툴킷을 의미한다.

표 5. 이륜차 로봇의 운동 모델 툴킷
Table 5. Motion model toolkit of bicycle type robot

Toolkit Name	***MM_BF***
Inputs	X_{t-1}, v_f, ϕ, ℓ, Δt, A
Process	Apply *MM_DD* toolkit after calculating w using ϕ
Results	X_t
Toolkit Name	***MM_BR***
Inputs	X_{t-1}, v_r, ϕ, ℓ, Δt, A
Process	Apply *MM_DD* toolkit after calculating w using ϕ
Results	X_t

표 5는 전륜, 후륜의 속도를 이용한 이륜차 로봇의 툴킷을 나타낸다. 전륜, 후륜 속도를 이용한 이륜차 로봇의 운동 모델 툴킷은 *MM_BF*와 *MM_BR*이라 정의한다. 툴킷의 입력 정보는 이전 위치 정보 X_{t-1}와 전륜 속도 정보 v_f 또는 후륜 속도 정보 v_r, 로봇 몸체와 앞바퀴의 사잇각 ϕ, 로봇의 앞뒤 바퀴의 사이거리 ℓ, 시간 간격 Δt, 그리고 운동 에러 A가 있다. 이륜차 로봇의 지향 각 변화인 $\dot{\theta}$는 차륜 구동 로봇의 회전 속도인 w와 같다. 그러므로 *MM_DD* 툴킷은 *MM_BF*와 *MM_BR* 툴킷에서 재사용 가능하다. *MM_BF*와 *MM_BR*의 출력 값은 연산된 위치 정보 X_t를 반환한다.

2.3 측정 모델

본 절에서는 거리 측정 센서에 대한 측정 모델 툴킷에 대해 다룬다. 실제 로봇의 센서를 통해 측정된 거리 값과 추측 항법을 통한 추정 위치에서 계산된 거리 값을 이용한다. 거리 값들을 확률분포에 적용해 실제 위치에 대한 추정 위치의 신뢰도를 연산한다. 이 과정을 측정 모델이라 한다. 표 6은 측정 모델 툴킷에 대해 나타낸다. 본 연구로 개발된 측정 모델 툴킷은 3개이다. 개발된 툴킷은 공통으로 사용되는 툴킷, 센서의 매개체가 직진성을 가지는 경우의 툴킷, 그리고 센서의 매개체가 회절 성질을 가지는 경우의 툴킷이 있다.

측정 모델 툴킷은 센서 정보와 계산된 정보를 4가지 확률분포를 사용해 각 확률분포의 확률 값을 구한다. 측정 모델의 확률 분포를 따르는 확률 값은 4개의 확률 값에 가중치들을 곱하고 모두 더하여 얻을 수 있다. 가중치는 z_{whit}, z_{wmax}, 그리고 z_{wrand}가 있고, 센서의 매개체에 따른 측정 모델에 대한 가중치 z_{wshort}와 z_{wlong}이 있다. 가중치들의 합은 z_{whit}, z_{wmax}, z_{wrand}, 그리고 z_{wshort} 또는 z_{wlong}을 더한 값으로 결과가 1이 되어야 한다.

센서의 매개체가 직진성을 가지는 경우 측정 모델 툴킷은 대표적으로 레이저 거리 측정 센서를 사용할 때 적용한다. 센서의 매개체가 회절의 성질을 가지는 경우 측정 모델 툴킷은 대표적으로 초음파 센서를 이용할 때 적용한다. 공통 측정 모델 툴킷은 공통으로 사용되는 확률분포의 확률 값을 구하는 툴킷으로 명칭은 *MM_Common*이라 정의한다. *MM_Common* 툴킷은 입력 정보로 실제 센서에서 획득한 센서 정보 z_t^k, 추정 위치에서 계산된 정보 z_t^{k*}, 가중치 Z, 가우시안 확률분포에 사용될 표준편차σ_{hit}, 센서의 측정 가능 최댓값 z_{max}를 입력받는다. 식 (3)은 가중치 Z로 z_{whit}, z_{wmax}, 그리고 z_{wrand}를 가진다. 확률분포들에 대한 가중치는 조합된 확률분포에 기여하는 수치를 나타낸다.

$$Z = [z_{whit}\ z_{wmax}\ z_{wrand}] \tag{3}$$

*MM_TR*은 직진성을 가지는 매개체를 이용한 센서의 측정 모델 툴킷의 명칭이다. 툴킷의 입력 정보는 실제 센서의 측정 정보 z_t^k, 추정 위치에서 계산된 정보 z_t^{k*}, 4개 확률모델의 가중치인 Z와 z_{wshort}, 가우시안 확률분포에 사용될 표준편차 σ_{hit}, 지수 확률분포에 사용되는 λ_{short}, 그리고 센서의 측정 가능 최댓값 z_{max}를 입력받는다. 4개의 확률분포가 조합된 확률분포의 확률 값을 구하기 위해 *MM_TR* 툴킷 내에서 *MM_Common*을 호출한다. 툴킷의 결과는 조합된 확률분포의 확률 값이다.

*MM_R*은 회절 성질을 가지는 매개체를 이용한 센서의 측정 모델 툴킷의 명칭이다. 툴킷의 입력으로 센서의 측정 정보 z_t^k, 계산된 정보 z_t^{k*}, 4개 확률모델의 가중치인 Z와 z_{wlong}, 가우시안 확률분포에 사용될 표준편차 σ_{hit}, 지수 확률분포에 사용될 λ_{long}, 그리고 센서의 측정 가능 최댓값 z_{max}를 입력받는다. 툴킷을 사용한 결과 값은 조합된 확률분포의 확률 값이다.

표 6. 측정 모델 툴킷
Table 6. Measurement model toolkit

Toolkit Name	***MM_TR***
Inputs	$z_t^k, z_t^{k*}, Z, z_{wshort}, \sigma_{hit}, \lambda_{short}, z_{max}$
Process	Calculate probability according to exponential probability distribution. Multiply weighting to the probability. Added the weighted probability to the result of *MM_Common* result.
Results	$p_{short} z_{wshort} + p$
Toolkit Name	***MM_R***
Inputs	$z_t^k, z_t^{k*}, Z, z_{wlong}, \sigma_{hit}, \lambda_{long}, z_{max}$
Process	Calculate probability according to exponential probability distribution. Multiply weighting to the probability. Added the weighted probability to the result of *MM_Common* result.
Results	$p_{long} z_{wlong} + p$
Toolkit Name	***MM_Common***
Inputs	$z_t^k,\ z_t^{k*},\ Z,\ \sigma_{hit},\ z_{max}$
Process	Calculate the error probabilities due to Gaussian noise, sensor failure, and random measurement. Multiply weight value to each probability. Add the weighted probabilities.
Results	$p_{hit} z_{whit} + p_{max} z_{max} + p_{rand} z_{wrand}$

2.4 방향 관련 연산

로봇의 방향은 기준 축을 중심으로 반시계방향의 π까지는 양수로, 시계방향의 π까지는 음수로 표현한다. 방향의 연산은 부호의 차이로 오류가 발생할 수 있다. 로봇 방향의 연산을 위해 본 연구에서는 두 방향의 합과 차를 구하는 툴킷과 방향들의 대푯값을 구하는 툴킷을 개발했다.

두 방향의 합을 구하는 툴킷의 명칭은 *AngleSum*으로 정의한다. 입력 정보는 두 방향 값을 입력받는다. 툴킷의 실행 결과 값은 두 방향의 합을 반환한다. 방향의 차를 구하는 툴킷의 명칭은 *AngleDifference*라 정의한다. 툴킷의 출력은 입력된 두 방향의 사잇각을 반환한다. 반환되는 사잇각은 입력받은 두 방향의 사잇각 중 작은 사잇각이다. 툴킷의 출력 값은 첫 번째 입력 정보를 기준으로 반환 값의 부호가 달라진다. 첫 번째로 입력받은 방향을 기준으로 두 번째로 입력

된 방향이 반시계방향에 위치한다면 반환되는 값은 -부호를 갖게 되고, 시계방향에 위치한다면 반환되는 값의 부호는 +이다.

방향들의 대푯값을 구하는 툴킷의 명칭은 *AngleAvg*라 정의한다. 툴킷의 입력 정보는 방향들 θ_n과 방향의 개수인 N을 입력받는다. 입력받은 방향들의 대푯값을 구하기 위해 크기가 1인 단위벡터들의 합을 구하는 방법을 이용한다. 모든 방향의 총합의 크기에 입력받은 방향의 총 개수로 나눈 값은 입력된 방향들이 평균 방향에 밀집된 분포도와 같다. 모든 방향이 같은 방향이라면 분포도는 1이 되고, 모든 방향이 고르게 분포되어 있다면 0에 가까운 값을 얻는다. 툴킷의 출력으로는 대표 방향과 분포도를 반환한다. 표 7은 방향 연산에 관한 툴킷을 나타낸다.

표 7. 방향 연산 툴킷
Table 7. Toolkit for operations on direction

Toolkit Name	***AngleSum***
Inputs	θ_1, θ_2 (two orientations to be added)
Process	limit the sum of θ_1 and θ_2 within the range of $-\pi$ to π
Results	θ
Toolkit Name	***AngleDifferece***
Inputs	θ_1, θ_2
Process	limit the difference of θ_1 and θ_2 within the range of $-\pi$ to π
Results	θ
Toolkit Name	***AngleAvg***
Inputs	θ_n, N
Process	Obtained the sum of unit vectors representing orientations(θ_n). Divide the result by the number of orientations(N).
Results	θ_{avg}, $distribution$

3. 시뮬레이션 툴킷

본 장에서는 개발된 시뮬레이션 툴킷에 대해 설명한다. 2장에서 설명한 위치 추정 툴킷을 시뮬레이션 툴킷에 적용하여 개발했다. 본 연구로 시뮬레이션 툴킷은 사용자가 시뮬레이션 툴킷에 위치 추정 툴킷의 요소들을 직접 입력하고 설정할 수 있도록 개발했다.

본 연구에서는 시뮬레이션 툴킷에서 사용되는 가상의 로봇을 현실의 실제 로봇으로 가정하고 구현했다. 그러므로 본 장에서 시뮬레이션에서의 가상 로봇을 실제 로봇이라 표현한다. 시뮬레이션 툴킷은 Microsoft Visual Studio C++ 6.0 개발도구의 MFC(Microsoft foundation class)를 사용해 구현했다[15]. 그림 3은 MFC 기반으로 개발된 시뮬레이션 툴킷의 CMCLDlg 클래스에 대한 구조를 나타낸다.

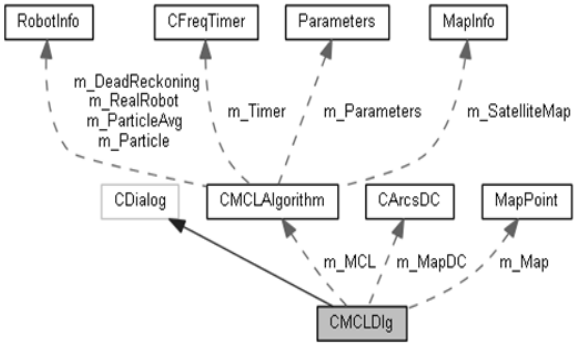


그림 3. 시뮬레이션 툴킷의 CMCLDlg클래스 멤버 구조
Fig. 3. CMCLDlg class member structure in simulation toolkit

CMCLDlg 클래스는 몬테카를로 알고리즘[16-17]을 위한 클래스인 CMCLAlgorithm 클래스의 변수 m_MCL을 생성하고 사용한다. CMCLDlg 클래스는 몬테카를로 알고리즘을 m_MCL 변수를 통해 사용할 수 있다.

RobotInfo 구조체는 위치 정보, 센서의 측정 값, 측정 모델의 결과인 신뢰도를 저장 가능하도록 구현했다. CMCLAlgorithm 클래스는 RobotInfo 구조체 변수인 m_DeadReckoning, m_RealRobot, m_ParticleAvg, m_Particle 변수들을 가진다. m_RealRobot은 실제 로봇의 위치를 표현하고 측정 센서 값을 저장하기 위한 변수다. m_DeadReckoning 변수는 추측 항법에 따른 추정 위치를 표시하기 위해 생성했다. m_Particle과 m_ParticleAvg는 몬테카를로 알고리즘의 적용을 위해 생성했다. m_Particle 변수는 파티클의 위치 정보, 계산된 정보, 그리고 측정 모델의 연산결과인 신뢰도를 저장한다. CMCLAlgorithm 클래스의 RobotInfo 구조체 변수들은 CMCDlg 클래스의 CMCLAlgorithm 클래스 변수인 m_MCL을 통해 관리할 수 있다.

시뮬레이션 툴킷에서 시간간격은 로봇의 위치를 도출하기 위해 필요하다. CFreqTimer 클래스는 시간 간격을 구하기 위해 사용되었다. Parameters 구조체는 2장에서 설명된 운동 모델의 오차 정보 A를 저장하기 위해 만들었다. 시뮬레이션 툴킷에서 센서의 측정 값은 초음파 송신기가 외부에 설치되고 로봇에 수신기가 설치되어 송신기에서 수신기까지의 거리 값임을 가정한다. 송신기의 위치 정보는 2차원 좌표 정보를 가지는 MapInfo 구조체를 사용했다. 개발된 시뮬레이션 툴킷은 회절의 성질을 가지는 초음파 센서를 사용함을 가정하므로 *MM_R* 측정 모델 툴킷을 사용한다.

2장에서 설명된 위치 추정 툴킷들은 CMCLAlgorith 클래스에 구현되었다. 2장에서 소개한 툴킷의 명칭은 시뮬레이션 툴킷에 적용 시 명칭을 다르게 구현했다. 그림 4는 CMCLAlgorithm 클래스에 사용된 위치 추정 툴킷을 나타내고, 이 함수들을 2장에 소개한 위치 추정 툴킷과 비교한다.

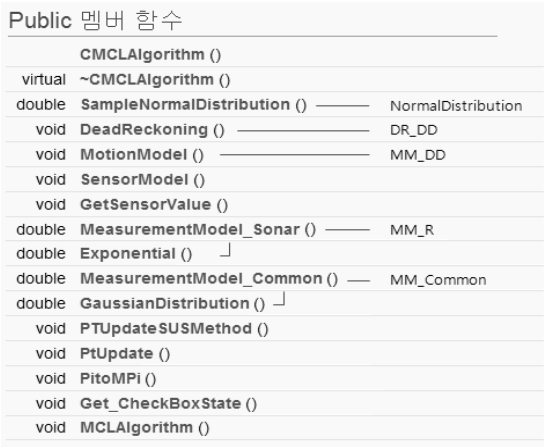


그림 4. CMCLAlgorithm클래스에 사용된 위치 추정 툴킷
Fig. 4. Localization toolkit in CMCLAlgorithm class

2장에서 소개한 위치 추정 툴킷 중 *NormalDistribution*은 시뮬레이션 툴킷에서 *SampleNormalDistribution*으로 명칭이 바뀌어 구현됐다. *DR_DD*는 *DeadReckoning*, *MM_DD*는 *MotionModel*, *MM_Common*은 *MeasurementModel_Common*으로 바뀌었다. *MM_R*은 *MeasurementModel_Sonar*로 바뀌었다. 2장에서 소개한 바와 같이 *MM_R*은 *MM_Common*을 재사용하는 툴킷이다. *MeasurementModel_Sonar*는 *MM_R*과 같이 *Measurement Model_Common*을 재사용한다.

그림 5는 시뮬레이션 툴킷의 실행화면이다. 개발된 시뮬레이션 툴킷은 2장에서 설명된 차륜 구동 로봇을 적용했다. 오른쪽 화면은 위에서 아래를 내려다본 2차원 평면으로 표현한 지도다. 초음파송신기 1의 위치는 원점이다. 원점을 기준으로 1 m간격을 점선으로 나타냈다. 초음파송신기 4개는 지도의 가장자리에 위치한다. 적색 원은 실제 로봇, 청색 원은 추측 항법을 통해 획득한 추정 위치를 나타낸다. 연두색 선은 실제 로봇과 추정 위치의 오차 적용 여부를 알아보기 위한 주행 경로를 나타낸다.

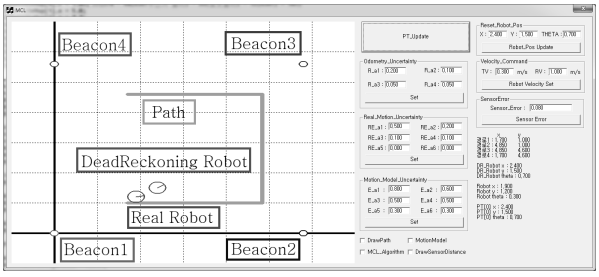


그림 5. 시뮬레이션 툴킷 실행 화면
Fig. 5. Simulation toolkit view

그림 6은 시뮬레이션 툴킷 화면 중 위치 추정 툴킷에 적용되는 요소들을 사용자가 입력하고 설정가능한 작업공간을 나타낸다. 실제 환경의 로봇은 외부 또는 내부의 오류로 오차가 적용되어 사용자의 속도 명령을 따르는 이상적인 움직임을 보이지 않는다. 시뮬레이션 툴킷의 실제 로봇 운동은 실제 환경의 로봇 운동처럼 사용자가 명령한 속도 명령에 내부 또는 외부 오차가 적용돼야 한다. Real_Motion_Uncertainty의 입력 칸은 사용자가 시뮬레이션 툴킷의 실제 로봇에 적용될 내부 또는 외부 오차를 입력하고 설정할 수 있다. 내부 또는 외부 오차를 적용하기 위해 *MM_DD* 툴킷을 사용한다. 적용된 오차는 사용자의 속도 명령에 대한 로봇의 운동의 불확실성이다.

Real_MotionUncertainty가 적용된 로봇의 속도 정보는 실제 환경 로봇의 실제 주행 속도로 간주한다. 사용자는 정확한 내부 및 외부 오차를 획득할 수 없으므로 정확한 로봇의 속도를 얻을 수 없다. 사용자가 획득 가능한 로봇의 속도 정보는 내부 오차가 적용된 내부 정보를 통한 속도 정보다. 이 속도 정보는 실제 환경의 로봇에 비교하면 엔코더 정보를 사용한 속도 정보와 같다. 이 과정은 그림 7에서 나타낸다.

Odometry_Uncertainty는 실제 로봇의 내부 정보에 적용하기 위한 불확실성이다. 사용자는 획득한 로봇 내부 정보를 이용한 속도 정보를 추측 항법에 사용하고 추정 위치를 구할 수 있다. 로봇의 내부 오차를 적용한 속도 정보를 *DR_DD* 툴킷에 사용해 추정 위치를 얻을 수 있다. 그림 8은 *DR_DD* 툴킷을 이용해 실제 로봇의 이동경로와 추측 항법을 통해 획득한 추정 위치의 경로를 나타낸 그림이다. DrawPath 체크박스를 이용해 실제 로봇, 추측 항법을 통한 추정 위치, 그리고 파티클들에 대한 경로를 그릴 수 있다. Motion_Model_Uncertainty는 몬테카를로 알고리즘 사용 시 내부 속도 정보에 적용되어 파티클들의 위치 정보 획득에 사용된다.

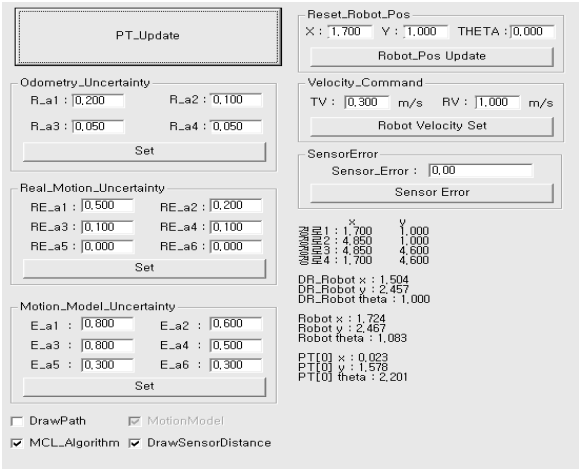


그림 6. 시뮬레이션 툴킷 실행 화면 - 변수 입력
Fig. 6. Simulation toolkit view - parameter input

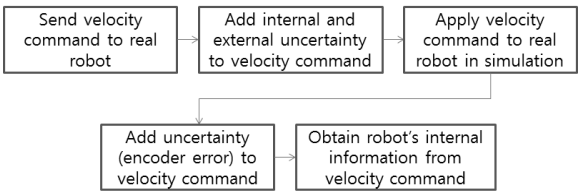


그림 7. 시뮬레이션 툴킷에서 로봇 속도 정보 획득 과정
Fig. 7. Velocity calculation procedure in the simulation toolkit

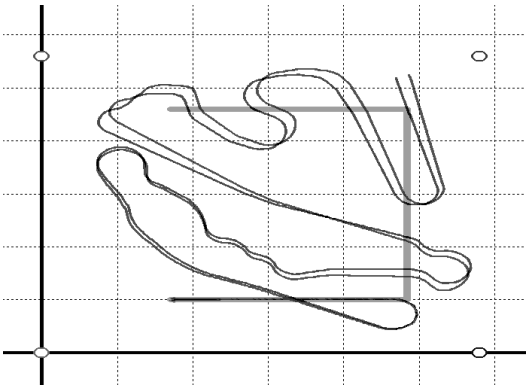


그림 8. 실제 로봇 경로와 추측 항법을 사용한 추정 위치 경로
Fig. 8. Trajectory of real robot and trajectory estimated by deadreckoning

몬테카를로 알고리즘을 이용한 파티클 필터 방법에서 운동 모델은 파티클들에 각각 다른 오차를 적용하기 위한 단계다. 파티클의 위치는 *MM_DD* 툴킷을 사용해 구할 수 있다. MotionModel 체크박스를 체크상태로 만들고 로봇을 제어하면 파티클들이 각각의 다른 운동을 보임을 알 수 있다. 그림 9는 운동 모델이 적용된 파티클들의 운동을 나타낸다.

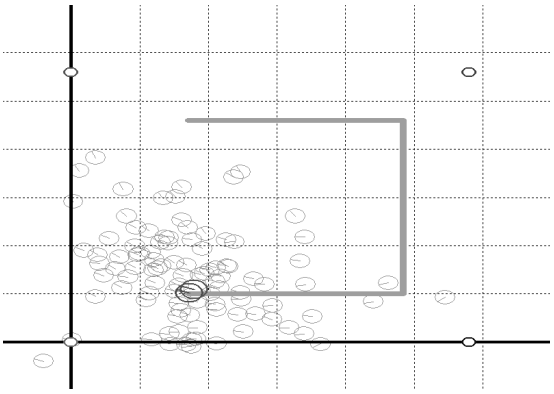


그림 9. 운동 모델을 적용한 파티클의 운동
Fig. 9. Particle movement according to motion model

MCL_Algorithm 체크박스는 몬테카를로 알고리즘을 실행하기 위한 것이다. 파티클들의 위치는 *MM_DD* 툴킷을 이용해 구한다. 운동 모델로 구한 파티클의 위치에서 실제 로봇의 센서 값과 파티클에서 계산된 값을 *MM_R* 툴킷에 적용해 파티클의 신뢰도를 구한다. 파티클의 신뢰도는 파티클의 위치를 다시 선정하는 단계에 이용된다. 몬테카를로 알고리즘을 이용한 위치 추정은 위 과정을 반복하며 파티클들의 평균 위치를 알아내는 것이다.

그림 10은 MCL_Algorithm체크박스에 체크하여 몬테카를로 알고리즘을 사용한 화면을 나타낸다. 주황색 원은 파티클들의 평균 위치와 방향의 대푯값을 나타낸다. 파티클들의 방향의 대푯값은 *AngleAvg* 툴킷을 이용해 구한다.

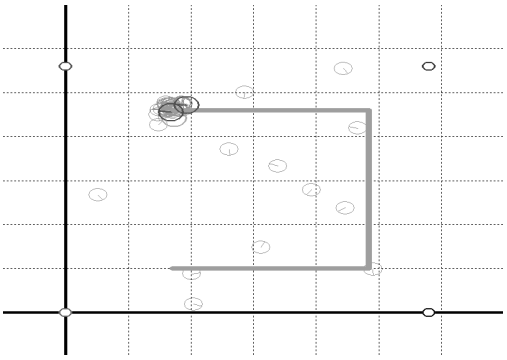


그림 10. 몬테카를로 알고리즘 적용
Fig. 10. Applying MCL algorithm

그림 11은 DrawSensorDistance 체크박스에 체크된 경우의 그림이다. 이 그림은 로봇 위치에서 초음파 송신기까지의 거리 값을 구해 지도상에 표현한 그림이다. 이 거리 값을 센서의 거리 측정값으로 정하고 파티클들의 연산된 거리 값을 이용해 *MM_R* 툴킷을 통해 파티클들의 신뢰도를 구한다. 로봇에서 초음파 송신기들로부터 측정된 거리 값들은 각 초음파 송신기의 색과 같은 색으로 표현한다. 사용자는 SensorError의 입력란에 센서 오차를 입력하여 시뮬레이션에 센서 오차를 반영하도록 할 수 있다. 그림 11은 오차 값으로 0.5를 입력하여 나타낸 그림이다. 센서 값은 센서 오차를 표준편차, 계산된 거리 값을 평균으로 적용한 정규분포에서 획득한 확률 변수 값을 사용한다.

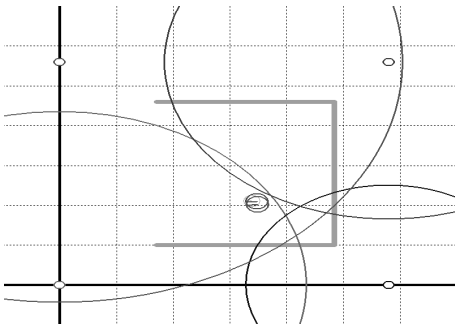


그림 11. 센서 측정 값 표현
Fig. 11. Description of senosr measurement

4. 실험

본 실험에서는 실제 로봇의 주행 중 저장된 정보들을 이용한 추측 항법의 추정 위치 경로와 시뮬레이션 툴킷을 이용한 추측 항법의 추정 위치 경로를 비교한다. 그리고 시뮬레이션 툴킷에서 몬테카를로 알고리즘의 위치 추정 성능을 보인다. 마지막으로 실제 환경에서 이동 로봇에 위치 추정 툴킷들을 적용한 실험에 대해 다룬다.

실험은 조선대학교 전자정보공과대학 6122호 실험실과 6층 복도에서 수행되었다. 실험에 사용된 이동 로봇은 레드원테크놀러(주)의 NRLab 02 로봇이다. 그림 12는 저장된 정보들 중 속도 정보를 이용해 추측 항법의 추정 위치를 나타낸 그림이다. 녹색 선은 실험실에 표시된 경로를 나타낸다. 청색 선은 추측 항법을 통한 추정 위치의 경로를 나타

낸다. 이 실험으로 내부 속도 정보를 이용한 추측 항법의 추정 위치는 실제 로봇의 위치로 판단하기 어려움을 알 수 있다. 그림 13은 시뮬레이션 툴킷에서 이동 경로를 따라 실제 로봇을 이동시키며 추측 항법을 통해 획득한 추정 위치의 경로를 나타낸 그림이다. 시뮬레이션 툴킷에 적용한 오차 정보들은 표 8과 같다. 그림 13을 통해 시뮬레이션 툴킷에서 추측 항법을 통한 추정 위치 정보는 내부 또는 외부의 오차 누적이 지속됨을 볼 수 있다. 그림 13을 통해 추측 항법을 통한 추정 위치 정보는 실제 로봇의 위치로 판단하기에 어려움을 보인다.

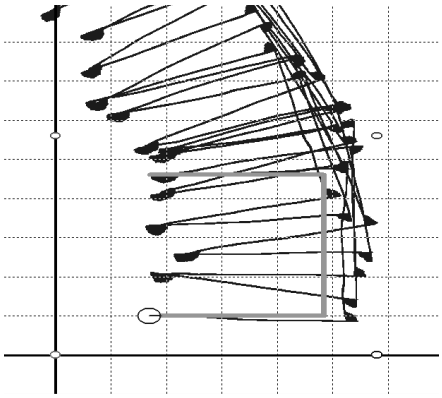


그림 12. 실제 환경에서 추측 항법을 이용한 추정 위치 경로
Fig. 12. Trajectory estimated by deadreckoning in real experiment

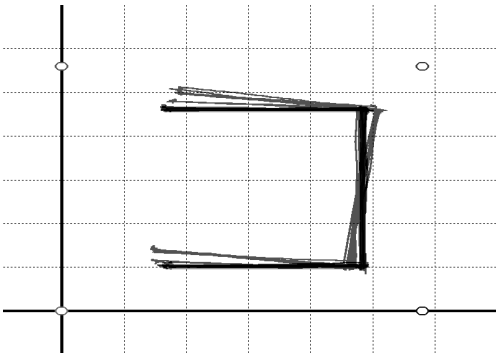


그림 13. 시뮬레이션에서 추측 항법을 이용한 추정 위치 경로
Fig. 13. Path of the estimated position through dead reckoning on simulation

표 8. 시뮬레이션 툴킷에 적용된 오차 정보
Table 8. Uncertainty applied to simulation

Real_Motion_Uncertainty			
RE_a1	0.5	RE_a2	0.2
RE_a3	0.1	RE_a4	0.1
RE_a5	0.0	RE_a6	0.0
Odometry_Uncertainty			
R_a1	0.2	R_a2	0.2
R_a3	0.05	R_a4	0.05
Motion_Model_Uncertainty			
E_a1	0.8	E_a2	0.6
E_a3	0.5	E_a4	0.5
E_a5	0.3	E_a6	0.3

그림 14는 실제 환경에서 몬테카를로 알고리즘을 사용하여 로봇의 위치를 추정하며 자율주행하는 로봇을 나타낸다. 그림 14의 좌측 그림은 파티클들을 청색, 파티클 대푯값은 분홍색, 그리고 주행 경로는 녹색으로 표현한다. 우측 그림은 실제 환경에서 좌측 그림의 해당지점을 지나고 있음을 보이기 위한 실제 환경에 사진이다. 로봇의 주행 환경은 건물도면을 이용해 프로그램 상에 지도를 작성했다. 몬테카를로 알고리즘을 통해 획득한 대푯값이 실제 로봇의 위치와 유사함을 볼 수 있다.

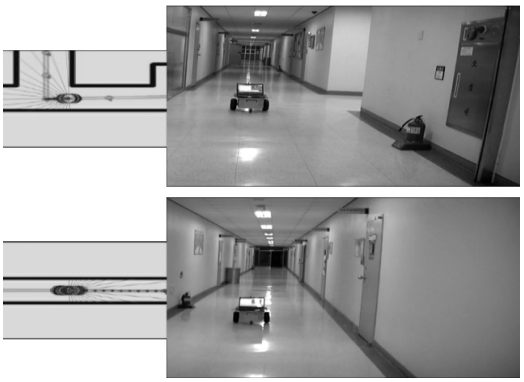


그림 14. 주행 로봇에 위치 추정 툴킷 적용 및 실험
Fig. 14. Applying toolkit for localization of a mobile robot and experiment

시뮬레이션 툴킷을 통해 본 연구로 개발된 위치 추정 툴킷들의 이용에 문제점이 없음을 보였다. 또한 실제 로봇의 몬테카를로 위치 추정 알고리즘을 이용한 실제 실험을 통해 개발된 툴킷들을 검증했다.

5. 결론

본 연구에서는 이동 로봇 위치 추정과 시뮬레이션 툴킷을 개발했다. 툴킷은 로봇의 운동 모델, 센서의 측정 모델, 방향 연산, 차륜 구동 로봇과 이륜차 로봇의 데드 레크닝을 위한 툴킷들로 구성된다. 이 툴킷들을 사용하여 이동 로봇의 자율주행을 위한 시뮬레이션 툴킷 및 실제 로봇에 적용 가능한 프로그램의 개발이 가능하다. 본 연구에서는 시뮬레이션 툴킷을 통해 위치 추정 알고리즘인 몬테 카를로 위치 추정 알고리즘을 구현하고 그 성능을 검증했고, 이 툴킷을 이용한 시뮬레이션 결과와 실제 환경에서 실험한 결과를 비교하여 툴킷의 유용성을 보였다.

제안된 툴킷은 특정한 로봇을 대상으로 하지 않고 일반적인 이륜차 로봇 모델과 차륜 구동 로봇 모델을 대상으로 하여 범용성을 높였다. 특히 로봇 운동의 불확실성과 센서 측정의 불확실성을 다양한 변수를 사용하여 사용자가 조절할 수 있게 했다. 그리고 실제 사용하는 로봇과 센서의 성능을 모사할 수 있다. 본 연구에서 제한된 툴킷은 로봇의 위치 추정, 장애물 회피 알고리즘, 지도 작성 알고리즘, 경로 추적 알고리즘, 그리고 경로 계획 알고리즘 등 자율주행 관련 알고리즘들을 구현하고, 시뮬레이션을 통하여 성능을 검증하는데 사용될 수 있다.

본 연구에서 개발된 툴킷은 사용자가 쉽게 이해하여 사

용할 수 있고, 향후 개발된 다양한 함수들을 추가하여 확장할 수 있다. 또한 현재 개발된 함수들을 사용하여 위치 추정, 지도 작성, 그리고 장애물 회피 알고리즘 등 자율주행 요소 기술 알고리즘을 실현하여 모듈형 응용 프로그램으로 제공한다면, 사용자가 기술적인 내용을 이해하지 못하더라도 이들 모듈들을 사용하여 원하는 작업을 실현 할 수 있을 것으로 기대된다.

References

[1] Jens-Steffen Gutmann, Ethan Eade, Philip Fong and Mario E. Munich, "Vector Field SLAM-Localization by Learning the Spatial Variation of Continuous Signals," *IEEE Transactions on robotics*, vol. 28, no. 3, pp. 650-667, Jun. 2012.

[2] W.P. Yu, S.L. Choi, J.Y. Lee, and S.H. Park, "Robot Navigation Technology and Its Standardization Trends," *Electronics and Telecommunications Trends,* vol. 26, no.6, pp. 108-119, Dec. 2011.

[3] http://carmen.sourceforge.net/home.html

[4] Jose Luls Blanco Claraco, *Development of Scientific Applications with the Mobile Robot Programming Toolkit*, http://www.mrpt.org/downloads/mrpt-book.pdf, Oct. 2010.

[5] http://www.mrpt.org/

[6] Peter Corke, *Robotics, Vision and Control*, Springer, 2011.

[7] http://www.petercorke.com/RVC/

[8] http://www.kartorobotics.com/

[9] Seok Ki Jeong, "Development of Function Library for Mobile Robot Localization," *Chosun Univ. master's thesis,* 2013.

[10] Dong Jin Seo, Nak Yong Ko, Sewoong Jung, and Jongbae Lee, "Network Based Robot Simulator Implementing Uncertainties in Robot Motion and Sensing," *The Journal of Korea Robotics Society,* vol. 5, no. 1, pp. 23-31, Mar. 2010.

[11] Julius Maximilian Univeresitat Wurzburg, *Kinematics of a car-like moile robot*, March 2003.

[12] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo, *Robotics*, Springer, 2009.

[13] Taek_Kun Nam and Chol-Seong Kim, "A Posture Control for Underwater Vehicle with Nonholonomic Constraint," *Journal of Korean Navigation and Port Research,* vol. 28, no. 6, pp. 469-474, 2004.

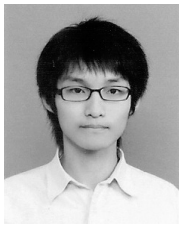
[14] Sebastian Thrun, Wolfram Burgard, Dieter Fox, *Probabilistic Robotics*, The MIT Press, Aug. 2005.

[15] 최호성, *열혈강의 visual c++ 2008 mfc 윈도우 프로그래밍*, 프리렉, Mar. 2009.

[16] Nak Yong Ko and Tae Gyun Kim, "Indoor Localization of a Mobile Robot Using External Sensor," *Journal of Institute of Control, Robotics and Systems* , vol. 16, no. 5, pp. 420-427, May 2010.

[17] Yuefeng Wang, Dan Wu, Sepideh Seifzadeh, Jingxi Chen, "A Moving Grid Cell Based MCL Algorithm for Mobile Robot Localization," *Robotics and Biomimetics (ROBIO), 2009 IEEE International Conference on*, pp. 2445-2450, Dec. 2009.

저 자 소 개



정석기(Seok Ki Jeong)

2011년 : 호남대학교 전자공학과 공학사
2013년 : 조선대학교 대학원 제어계측공학과 공학석사
2013～현재 : 조선대학교 대학원 제어계측공학과 박사과정

관심분야 : 이동 로봇, 자율주행
Phone : +82-62-230-7766
E-mail : seokki@chosun.kr

김태균(Tae Gyun Kim)

2007년 : 조선대학교 제어계측공학과 공학사
2009년 : 조선대학교 대학원 제어계측공학과 공학석사
2013년 : 조선대학교 대학원 제어계측학과 공학박사
2013년～현재 : 한국해양과학기술원 해양시스템연구부 연구원

관심분야 : 이동 로봇, 수중로봇, 자율주행
Phone : +82-42-866-3865
E-mail : tgkim@kiost.ac

고낙용(Nak Yong Ko)

1985년 : 서울대학교 제어계측공학과 공학사
1987년 : 서울대학교 대학원 제어계측공학과 공학석사
1993년 : 서울대학교 대학원 제어계측공학과 공학박사
1997～1998년, 2004～2005 : 미국 Carnegie Mellon Univ. Visiting research scientist
1992～현재 : 조선대학교 제어계측로봇공학과 교수

관심분야 : 지상로봇과 수중로봇의 자율주행
Phone : +82-62-230-7108
E-mail : nyko@chosun.ac.kr