실내 자율주행을 위한 ROS 기반 이동 로봇의 경로 계획 방법

백지훈, 김상훈*
*한경대학교 전기전자제어공학과
e-mail:kimsh@hknu.ac.kr

A path planning method for indoor Self-driving robot based on ROS

Ji-Hoon Baek, Sang-Hoon Kim
Dept of Electrical, Electronic and Control, Hankyong National University

요 약

본 논문은 Linux ubuntu에서 로봇 개발 플랫폼 ROS(Robot Operating System)을 이용하여 실내 자율주행 관련 패키지와 LRF센서를 사용한 경로탐색을 하기까지의 과정 그리고 향후의 설계 방안에 대해 다룬다.

1. 서론

4차 산업혁명을 주도하는 지능로봇은 분야를 막론하고 활발한 연구가 진행되고 있다. 그 중 자율주행 분야는 로봇이 목적지까지 이동하기 위해 주어진 환경에 대하여 스스로 판단하여 이동할 수 있도록 한다.

그 과정으로는 로봇의 위치 인식, 지도 작성, 경로 탐색, 장애물 회피 등이 있는데 이 중에 어느 하나 쉬운 것이 없다. 하지만 Linux의 우분투를 기반으로 하는 Robot operating system(이하 ROS)은 센서 값, 노드간의 출력 값 등의 데이터를 메시지의 형태로 간단하게 주고받을 수 있고 기존에 만들어진 자율주행에 관련된 패키지가 오픈소스로 공개되어 있어 사용자로 하여금 쉽게 로봇을 개발할 수 있는 환경을 제공 한다.

이를 이용하여 본 논문에서는 먼저 자율주행의 경로탐색까지의 과정을 기술하려 한다. SBC(Single Board Computer)로는 Odroid XU4를 이용하여 우분투 16.04 버전에서 ROS kinetic버전을 사용하였으며 mapping 및 장애물 인식을 위해 LRF(Laser Range Finder)센서 HOKUYO사의 URG_04LX모델을 사용하였다.

2. 본론

2.1. 실내 자율주행의 기본 4요소

모바일 로봇이 자율적으로 목적지까지 장애물에 부딪히지 않고 무사히 도착하기 위해서는 다음과 같은 4가지 요소를 필요로 한다.

- 지도 작성(Mapping)

시중에 판매되는 차량용 navigator는 정확한 지도가 포함되어 운전자가 원하는 위치까지의 경로를 계산할 수 있다. 하지만 임의의 실내에서 사용 될 모바일 로봇은 따로 지도를 만들어 사용할 필요가 있다.

여기서 사용되는 기법이 SLAM(Simultaneous localization and mapping : 동시적 위치 추정 및 지도 작성)인데 이는 로봇이 미지의 임의 공간을 이동하면서 주변을 감지하며 현재 위치를 추정함과 동시에 지도를 작성하는 방법이다

- 위치 인식(Localization)

지도가 있다고 하더라도 로봇의 위치를 알 수 없다면 지도는 의미 없는 데이터가 돼버린다. 그렇기 때문에 로봇은 지도상에서 자신의 위치를 추정할 수 있는 능력을 갖춰야 한다.

위치를 인식하는 방법으로는 절대적 위치 인식 방법과 상대적 위치 인식 방법으로 나뉠 수 있는데 실외의 경우엔 GPS를 이용하여 절대적 위치를 파악할 수 있지만 실내에서는 GPS를 사용할 수 없기 때문에 바퀴의 엔코더를 이용하여 주행량을 계산하고 관성 센서를 이용하여 주행량의 오차를 줄이는 상대적 위치 인식 방법이 주로 사용된다. 다른 방법으로는 2D 거리정보를 통해 로봇의 위치를 확률적으로 나타내는 몬테카를로 위치추정(Monte Carlo Localization) 알고리즘, 위치 좌표에 대한 정보를 가진 마커를 사용하여 로봇이 마커를 인식하였을 때의 마커에 대한 상대적인 거리를 계산하여 위치를 인식하는 방법 등을 사용할 수 있다.

- 경로 계획(Path planning)

로봇이 맵 정보를 가지고 있고 자신의 위치 또한 알고 있다면 경로를 계획할 수 있다. 이 때 로봇은 맵 정보만을 가지고 경로를 계산하는 전역 경로 탐색(Global path planning)을 수행하고 실제 환경에서 로봇의 중심으로 일

부 지역을 대상으로 한 국부 경로 탐색(local path planning)을 수행하여 로봇이 목적지 까지 도달할 수 있도록 한다.

- 장애물 회피(Obstacle avoidance)

로봇이 목적지까지 안전하게 도착하기 위해서 로봇은 장애물과의 거리를 계측할 필요가 있다. 이 때 거리를 계측하는 방법으로 LRF센서, 초음파 센서 혹은 Depth camera와 같은 디바이스를 이용하여 장애물까지의 거리를 계측하고 이 정보를 통해 장애물을 회피하여 기동할 수 있다.

2.2 ROS 용어 정리

본문의 2.3의 내용에 대한 이해를 돕기 위해 ROS 공식 카페의 내용을 인용하여 ROS의 기본이 되는 용어들을 아래와 같이 간단히 정리하였으니 참고하길 바란다[1].

- 마스터(master)

마스터는 노드와 노드사이의 연결 및 메시지 통신을 위한 네임 서버와 같은 역할을 한다. roscore가 실행 명령어이며, 마스터를 실행하게 되면 각 노드들의 이름을 등록하고 필요에 따라 정보를 받을 수 있다. 마스터가 없이는 노드간의 접속, 토픽과 서비스와 같은 메시지 통신을 할 수 없다.

- 노드(node)

ROS에서 최소 단위의 실행 프로세서를 의미한다. 노드는 생성과 함께 노드 이름, publisher 이름, subscribe 이름, topic 이름, 서비스 이름, 메시지 형태, URI 주소 및 포트를 등록하여 이를 기반으로 각 노드는 노드끼리 토픽 및 서비스를 이용하여 메시지를 주고받을 수 있다.

- 패키지(package)

ROS는 패키지를 단위로 각각의 응용 프로그램들이 개발된다. 패키지는 최소한 하나 이상의 노드를 포함하고 있다.

- 토픽(topic)

발행자 노드가 하나의 이야깃거리에 대해서 토픽이라는 이름으로 마스터에 등록한 후 이야깃거리에 대한 이야기를 메시지 형태로 발행한다. 이 메시지를 수신받기 원하는 subscriber 노드는 마스터에 등록된 토픽의 이름에 해당되는 publisher 노드의 정보를 받는다. 이를 기반으로 구독자 노드는 발행자 노드와 직접 연결하여 메시지를 송/수신 또는 요청/응답 받게 된다.

- 메시지(masage)

노드는 메시지를 통해 노드간의 데이터를 주고받는다. 메시지는 integer, floating point, boolean와 같은 변수 형태이다.

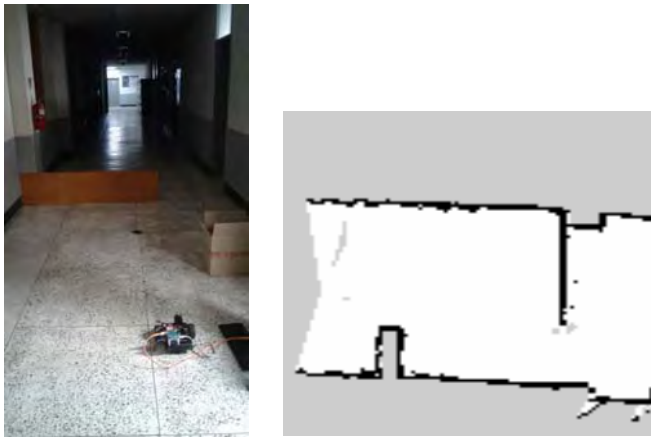
- 퍼블리셔(publisher)

publisher 노드는 발행을 하기 위해 토픽을 포함한 자신의 정보들을 마스터에 등록하고, 수신을 원하는 subscriber 노드에게 메시지를 보내게 된다.

- 서브스크리버(subscriber)

subscriber 노드는 구독을 하기 위해 토픽을 포함한 자신의 정보를 마스터에 등록하고 구독하고자 하는 토픽을 발행하는 publisher 노드로부터 메시지를 받게 된다.

2.3 Hector SLAM 패키지



(그림 1) 실제 환경과 Hector SLAM 패키지를 통해 맵핑된 지도

경로를 계획하기 위해 필요한 맵을 작성하기 위해 본 논문에서는 ROS에서 제공하는 Hector SLAM 패키지를 사용하였다. 그러나 Odroid_XU4에서는 CPU의 연산속도가 알고리즘 처리 속도를 따라가지 못하는 문제가 있어 파라미터를 수정하여 맵을 작성하였다.

2.4 내비게이션 패키지

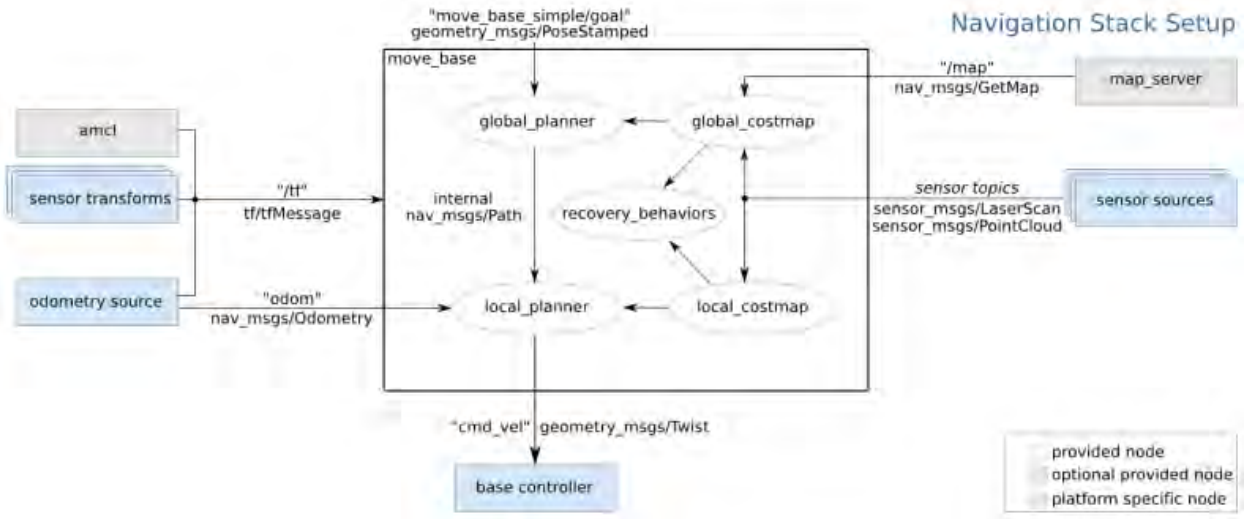
Hector SLAM을 통해 맵 정보를 가지고 있다면 이제 내비게이션 패키지를 통해 맵 상의 로봇의 위치를 추정하고 자율주행을 위한 경로 탐색을 할 수 있다.

맵 상에서 로봇의 위치 추정은 amcl 노드에서 [2.1.2 위치추정]에 언급된 몬테카를로 위치 추정 알고리즘에서 적은 수의 샘플수를 사용하여 수행시간을 줄인 AMCL(Adaptive Monte Carlo Localization) 알고리즘이 사용되었다.

경로 탐색의 과정은 move_base 그룹의 노드들을 통해 처리된다. move_base 그룹은 [그림 2]의 큰 네모 칸으로 둘러싸인 영역을 의미한다.

2.4.1 전체 경로 탐색

먼저 전체 경로 탐색은 map_server 노드로부터 /map 토픽을 통한 맵 정보와 LRF센서 관련 노드의 /scan 토픽을 통한 2D 거리정보를 global_costmap 노드에 공급하여 장애물 영역, 장애물과 충돌이 예상되는 영역, 로봇이 이동 가능한 영역 총 3영역으로 계산된 지도를 만든다. 그 다음으로 global_planner 노드에서 costmap을 공급받아 로봇이 충돌하지 않는 영역을 중점적으로 전체 경로를 탐색한다. 이 때 사용된 알고리즘은 ROS wiki에 따르면 Dijkstra 알고리즘을 유연하게 대체하여 만든 알고리즘이라 한다.

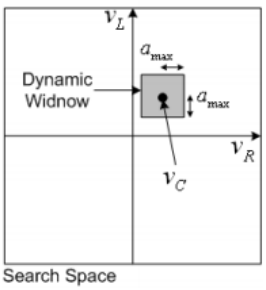


(그림 2) Navigation 패키지의 move_base그룹의 노드 및 토픽 정보[4]

2.4.2 국부 경로 탐색

다음으로 국부 경로 탐색은 전체 경로, 센서 노드를 통한 2D 거리 정보(/scan 토픽), sensor transforms 노드를 통해 로봇에서 LRF센서의 상대적 위치에 따라 상대적인 데이터를 절대적 데이터로 바꿔주기 위한 정보 (/tf 토픽), 엔코더와 IMU 센서를 통해 계측한 로봇의 주행거리로 추정한 로봇의 상대적 위치 정보(/odom 토픽)를 통해 국부 경로를 계획할 수 있다.

먼저 local_costmap노드에서 /scan 노드를 통한 2D 거리 정보를 바탕으로 costmap을 만든다. local costmap은 global costmap과 사용하는 목적은 다르지만 동일한 표현 방식의 지도이다. 다음으로 local_planner 노드에 전역 경로, 로봇의 위치 정보, local costmap을 공급받아 DWA(Dynamic Window Approach)알고리즘을 통하여 지역 경로를 계획한다. DWA 알고리즘은 로봇이 이동 가능한 최대속도 범위를 [그림 3]와 같이 탐색 공간(search space)으로 정의하고, 로봇의 현재속도 v_c를 기준으로 최대가속도 범위내에 있는 영역을 동적윈도우로 설정한다. 또한 동적윈도우내의 좌우바퀴의 속도에 대해 목적함수(objective function)를 통해 계산하여 최대가 되는 선속도

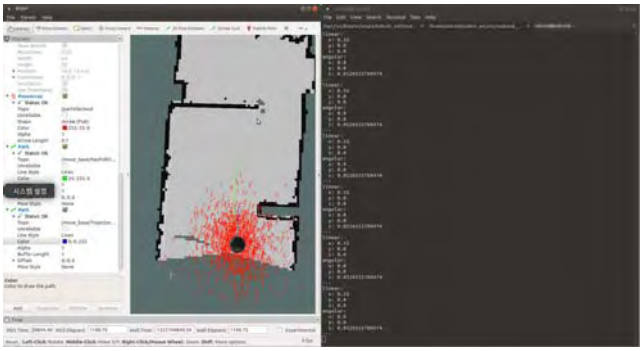


(그림 3) 탐색 경로

와 각속도를 찾아 로봇을 제어하는 방법이다[2] ROS에서는 목적함수를 통해 출력되는 선속도와 각속도를 /cmd_vel 이라는 토픽을 통해 서브스크리버 노드에 전달된다. 본 논문에서는 DWA알고리즘의 구체적인 내용들에 대해서는 생략한다.

2.4.3 내비게이션 패키지 실행

위 과정은 하나의 패키지에서 실행되므로 navigation패키지를 사용하기 위해서는 맵 정보, 2D 거리 정보, 위치 정보 등을 메시지 형태로 제공받아야 패키지가 실행된다. 그러나 본 논문에서는 실험에 LRF센서만을 이용한 실험을 진행하여 /odom 토픽을 통해 상대위치 정보를 local planner 노드에 발행할 수 없는 등 제약사항이 있어 가상 로봇 kobuki로 대체하여 패키지를 사용하였다. 따라서 본 논문에서는 고정된 위치에서 전체 경로와 국부 경로만을



(그림 4) Rviz를 통해 시각화한 내비게이션 패키지와 /cmd_vel 토픽에서 공급되는 메시지

관측하였다. 실행된 내비게이션 패키지는 Rviz를 통해 토픽에 대한 정보들을 [그림 4]와 같이 시각화 할 수 있다. [그림 4]의 장애물 근처의 흰 점들은 LRF 센서를 통해 계측된 실제 환경에서의 장애물에 대한 정보이고 로봇을 주위로 빨간색의 화살표는 AMCL노드를 통해 계산된 로봇의 위치할 확률을 입자 형태로 나타낸 것이다. 그리고 로봇에서 나오는 초록색 선은 global_planner를 통한 전체 경로를 나타내며 파란색 선은 local_planner를 통해 국부 경로를 나타내는 /cmd_vel 토픽의 선속도와 각속도를 시

각화 하여 나타낸 것이다.

2.5 자율주행 실험

내비게이션 패키지의 Local_planner노드를 통해 얻은 /cmd_vel 토픽의 Linear velocity와 angular velocity값을 이용하여 모바일 로봇의 Atmega128과 UART 통신을 이용하여 좌우 바퀴의 PWM을 제어하여 자율주행을 해보고자 하였으나 이 값들을 이용하여 실제 모바일 로봇에 적용시켜 로봇을 구동시켜 본 결과 바퀴를 회전시키기 위한 최소의 힘, 바닥의 마찰계수 등의 요인들에 의하여 로봇이 회전을 해야 할 때 원하는 움직임을 보이지 않았다. (본 논문에서는 한백전자의 HBE-RoboCAR 모델의 뒷바퀴를 제거하여 사용하였다.)

(그림 5) 실험에서 사용된 HBE-RoboCAR

3. 결론 및 연구 방향

ROS kinetic 버전에서 제공하는 패키지를 이용하여 맵을 제작하고 내비게이션 패키지를 이용하여 최종적으로 경로를 탐색하는 과정까지 실험을 할 수 있었다. 그러나 실제 환경에서의 자율주행은 모터의 힘, 바닥면의 마찰계수 등의 요인으로 인해 로봇을 원하는 데로 제어 할 수 없었다. 따라서 위 문제를 해결하기 위해 하드웨어 설계 시 로봇이 회전하는 움직임을 보다 잘 컨트롤 할 수 있는 조향장치와 차동 기어를 이용한 후륜 구동 형태로 모바일 플랫폼을 설계하고, 경로추종 알고리즘을 적용하여 지역 경로 계획을 따를 수 있도록 연구를 진행할 계획이다.

또한 로봇이 실내에서 자율주행을 하며 로봇은 자신의 위치를 주행거리를 계산하는 상대적인 계측과 파티클 필터를 이용하여 2d 거리센서의 데이터와 맵 정보를 비교하여 맵 상의 위치를 보정해주는 기법이 사용되었다. 그러나 상대적인 계측 방법은 오차율이 누적될수록 점점 더 오차가 커지게 되고 주변의 장애물이 계측되지 않을 정도의 넓은 공간에 있다고 가정을 하게 된다면 파티클 필터를 통한 맵 상의 위치 보정이 불가능하다. 이와 같은 문제를 해결하기 위해 임의의 지점에 QR코드를 배치하여 영상처리를 통해 QR코드가 가진 맵 상의 절대 좌표와 QR코드로부터 로봇까지의 상대적인 거리를 계산하여 맵상의 정확한 위치를 보상받을 수 있는 거점을 만들어 주는 방법을 적용시켜 로봇이 좀 더 정확한 위치를 알 수 있도록 연구를 진행할 계획이다.

참 고 문 헌

[1] 오픈소스 소프트웨어 & 하드웨어: 로봇 기술 공유 카페 (오로카), "로봇 운영체제 강좌 : ROS 용어 정리|작성자 표윤석", http://cafe.naver.com/openrt/2466, 2018-03-22

[2] D. Fox, W. Burgard, S. Thrun, "The dynamic window approach to collision avoidance," IEEE Robotics and Automation Magazine, pp. 23-33, 1997.

[3] 윤희상, 박태형. "수정된 전역 DWA에 의한 자율이동로봇의 경로계획." 전기학회논문지, 60.2 (2011.2): 389-397.

[4] ROS.org, "move_base", http://wiki.ros.org/move_base, 2018-03-29