논문 2021-16-38

실내 자율주행에 적합한 SLAM과 전역경로생성 방법을 적용한 휠체어로봇 구현

(Implementation of Wheelchair Robot Applying SLAM and Global Path Planning Methods Suitable for Indoor Autonomous Driving)

백 수 진, 김 아 현, 김 종 욱*
(Su-Jin Baek, A-Hyeon Kim, Jong-Wook Kim)

Abstract : This paper presents how to create a 3D map and solve problems related to generating a global path planning for navigation. Map creation and localization were performed using the RTAB-Map package to create a 3D map of the environment. In addition, when the target point is within the obstacle space, the problem of not generating a global path was solved using the asr__navfn package. The performance of the proposed system is validated through experiments with a wheelchair-type robot.

Keywords : Autonomous Mobile Robot, Visual SLAM, Navigation, Global Path Planning

I. 서 론

최근 코로나19 사태로 인한 비대면 수요가 증가하고 있고, 자동화 가전제품 보급 증가와 인건비 상승 등으로 가정용 로봇에 대한 수요가 증가하고 있다. 또한 자율주행 로봇에 관한 연구가 활발히 진행되고 있는데, 로봇이 가정이나 사무실 같은 실내에서 자유롭게 이동하기 위해서는 자율주행 기술이 필요하다.

자율주행이란 로봇이나 사용자의 개입 없이 스스로 판단하여 목적지까지 주행하는 것을 말한다. 이를 위해서는 주행하려는 환경의 2차원 또는 3차원지도가 필요하다. 2차원지도는 평면으로 지도를 만들기 때문에 2차원 센서에 인식되지 않은 장애물은 매핑이 되지 않는다. 그래서 가정 내에 있을 수 있는 식탁이나 책상과 같은 경우에 다리만을 장애물로 매핑하게 된다. 그러므로 가정내 객체의 3차원 형상을 인식할 수 있어야 한다.

또한 로봇이 주행 중 전원부족이나 비정상 상황으로 인해 리부팅 후 현 위치에서 주행 알고리즘 실행 시 특별한 조작없이 자동으로 실행되어야 한다. 이를 위해서는 지도의 좌표계가 로봇을 기준으로 생성될 필요가 있다.

로봇이 환경정보가 없는 공간에서 자율주행으로 목적지까지 안전하게 도착하기 위해 LiDAR, 카메라 등의 센서를 이용하여 공간의 지도를 작성하는 동시에 로봇 자신의 위치를 추정하는 기술을 SLAM (Simultaneous Localization and Mapping) [1]이라고 한다. SLAM 알고리즘에는 Filter-Based SLAM [2], Vision-Based SLAM [3], Graph-Based SLAM [4] 등이 있다.

Filter-Based SLAM은 칼만필터, 파티클필터 같은 필터를 사용한 SLAM기법이다. 칼만필터를 사용하는 방식은 로봇의 상태를 추정하기 위한 센서 데이터가 가우시안 분포를 따를 때만 사용할 수 있다. 파티클필터를 사용하는 방식은 가우시안이 아닌 임의의 분포를 다루기 위한 접근 방법이고, 샘플이라는 입자들로 구성되는데 이 샘플의 개수가 충분하지 않으면 정확하게 위치 추정이 되지 않을 수 있다 [5].

Vision-Based SLAM은 영상센서를 사용하는 방식이다 [6]. 3D 맵을 작성할 수 있지만, 영상센서만을 사용하여 맵을 작성할 수 있다. 그러나 영상센서는 시야가 좁기 때문에 영상센서만을 사용하여 매핑을 하면 누락되는 데이터가 발생할 수 있다.

Graph-Based SLAM은 LiDAR와 영상센서 등을 사용하여 환경의 3D 포인트 클라우드를 생성하거나 2D 그리드 맵을 생성한다. Vision-Based SLAM 방식과 유사하지만, LiDAR 센서를 사용할 수 있다는 장점이 있다. 따라서 본 논문에서는 영상센서와 LiDAR 센서를 모두 사용할 수 있는 Graph-Based SLAM을 사용한다.

본 논문에서는 로봇용 소프트웨어 라이브러리 및 애플리케이션을 구축할 수 있는 ROS (Robot Operating System) [7]를 기반으로 가정이나 사무실 같은 실내에서 자율주행하는 로봇에 적합한 SLAM과 네비게이션 패키지를 검토하여 구현하였다. Graph-Based SLAM 중에서 로봇을 초기위치가 아닌 곳에 두고 SLAM을 실행하여도 이전처럼 수동으로 맞

*Corresponding Author (kjwook@dau.ac.kr)
Received: Oct. 16, 2021, Revised: Nov. 17, 2021, Accepted: Dec. 9, 2021.
S.-J. Baek, A.-H. Kim: Donga-A University (Master Student)
J.-W. Kim: Donga-A University (Prof.)
※ 본 논문은 산업통상자원부 산업기술혁신사업 '가변형 밀착구조를 가진 신체약자 생활자립형 서비스 로봇 개발(No. 20004720)'의 지원을 받아 연구되었음.

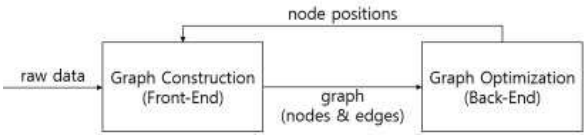

그림 1. 그래프 기반 SLAM 파이프라인
Fig. 1. Graph Based SLAM System Pipeline

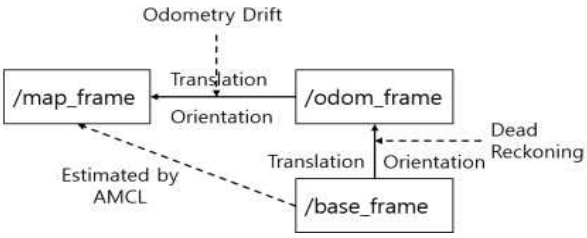

그림 2. AMCL Localization 구조
Fig. 2. AMCL Localization Structure

춰주지 않아도 되는 RTAB-Map (Real-Time Appearance-Based Mapping)을 사용하고 [8, 9], 장애물이 위치한 지점을 목적지로 입력해도 목적지에 가깝게 전역주행경로를 생성할 수 있는 asr_navfn 패키지를 사용했다 [10, 11].

본 논문의 구성은 다음과 같다. 2장에서는 Visual SLAM에 대해 설명하고, 3장에서는 네비게이션 알고리즘인 asr_navfun에 대해 설명한다. 4장에서는 본 논문에서 구현한 패키지를 휠체어로봇에 적용하여 자율주행을 실시한 결과를 설명하고, 5장에서 결론을 제시한다.

Ⅱ. RTAB-Map

그림 1은 Graph-Based SLAM 파이프라인을 나타낸다. Front-End로 센서의 원 (raw) 데이터가 입력으로 들어오면 센서 데이터들의 차이를 통해 로봇의 위치를 계산한다. 이 때, 센서 데이터의 노이즈로 인한 오류가 누적되어 전체적인 궤적이 어긋나게 되는데, PGO (Pose Graph Optimization) 알고리즘을 사용하여 Back-End에서 Loop Closing을 함으로써 기존에 누적된 오류를 제거하고 노드들의 위치를 최적화한다 [12]. 이 과정에서 발생한 Loop Closing은 RGB-D 센서에 의해 탐지된 이미지가 이전에 방문한 위치인지를 결정하기 위해 bag-of-words 기법을 사용한다 [13].

로봇 자율주행에는 지도 생성뿐만 아니라 로봇의 위치를 아는 것도 중요하다. 위치인식 (Localization)은 미지의 공간에서 로봇이 어디에 위치하고 있는지를 예측하는 방법이다. 로봇의 위치인식에 가장 많이 사용되는 알고리즘인 AMCL (Adaptive Monte Carlo Localization)은 베이즈 필터를 기반으로 하는 파티클 필터를 사용하여 로봇의 위치를 추정하며, 파티클 필터는 로봇 pose의 불확실성을 확률로 나타내기 위해 가중치를 갖는 샘플인 파티클을 사용한다 [14].

AMCL은 그림 2와 같이 프레임을 변환하여 로봇의 위치가 map의 위치로 변환되므로, 로봇을 초기위치에 두지 않고 프로그램을 실행하면 수동으로 초기위치를 맞춰주어야 한다. 그러나 RTAB-Map은 map 프레임을 odom 프레임으로 변환하여 로봇의 위치에 map의 위치를 맞추므로, 로봇을 초기위치가 아닌 곳에 두고 프로그램을 실행하여도 이전처럼 수동으로 맞춰주지 않아도 된다.

그림 3은 ROS Navigation 노드 구조에서 RTAB-Map을 적용하였을 때 노드 구조를 나타낸 것이다. 기존에는 map 업데이트를 위해 map_server 패키지를 사용하고, localization을 위해 amcl 패키지를 사용한다. 그러나 여기서는 map_server 대신 rtabmap 노드를 사용하고, amcl 노드 대신 루프 폐쇄 감지 접근 방식과 근접 감지 접근 방식을 사용하여 로봇의 위치를 추정한다. 여기서 루프 폐쇄 감지 접근 방식은 2.1절에서 설명하였고, 근접 감지 접근 방식 [15]은 360° LiDAR 센서와 같이 시야가 넓은 센서를 같이 사용하여 카메라가 감지할 수 없는 영역에서의 odometry 오류를 보정한다.

레이저 스캔은 카메라만큼 물체 식별을 잘 하지 못하기 때문에 근접 감지는 로봇의 추정 위치 주변에 위치한 local map의 노드로 제한된다.

Ⅲ. ASR navfn

로봇이 자율 주행을 하기 위해서는 출발지부터 목적지까지의 전체 경로를 생성하는 전역경로계획 이 필요하다. ROS

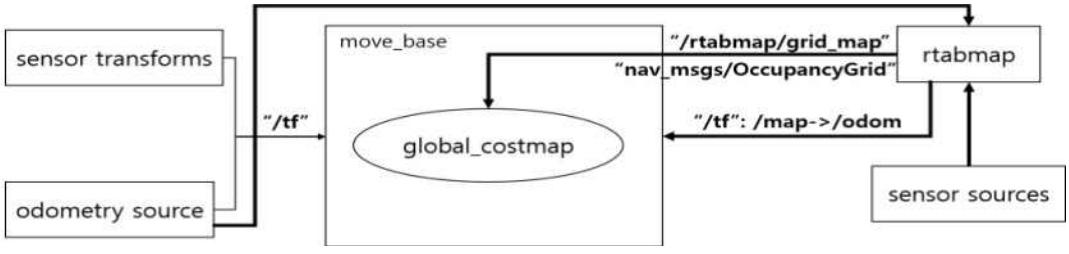

그림 3. RTAB-MAP을 적용한 Navigation 노드구조
Fig. 3. Structure of Navigation node applied RTAB-MAP

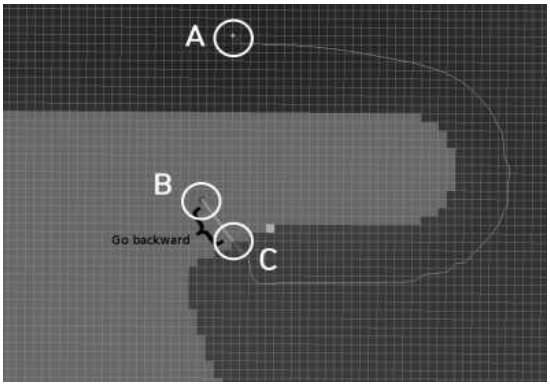

그림 4. asr_navfn 경로 설정
Fig. 4. Setting path of asr_navfn

그림 5. 휠체어 로봇
Fig. 5. Wheelchair robot

에서는 전역경로계획 (global path planning) 알고리즘으로서 Navfn 패키지를 기본적으로 제공한다 [16]. Navfn은 ROS에서 자율 주행을 위한 오픈소스인 move_base노드의 Global planner 플러그인으로 사용되며, 시작 지점과 도착 지점 사이의 글로벌 경로를 최소 비용으로 찾기 위해 하나의 시작 노드로부터 모든 다른 노드까지의 최단 경로를 찾는 Dijkstra [17] 알고리즘을 사용한다. 하지만 Navfn 패키지는 목적지나 서비스 대상 객체가 글로벌 맵 상의 장애물 공간 내에 설정될 때(탁자위에 있는 물건 가져오기 등), 글로벌 경로를 생성하지 못하고 공간을 비우기 위해 복구 동작을 하는 rotate-recovery [18] 패키지로 인해 제자리 회전을 하게 된다. 가정 내에서는 가구의 배치가 자주 바뀔 뿐만 아니라 목적지가 잘못 입력됐을 때 규모가 큰 휠체어 로봇이 제자리 회전을 하는 건 다소 위협적이다. 그래서 본 논문은 장애물이 위치한 지점을 목적지로 입력해도 목적지에 가깝게 주행경로를 생성할 수 있는 asr_navfn 패키지를 사용하였다.

asr_navfn 패키지는 Navfn global planner의 업데이트된 버전으로, 목적지에 장애물이 있는 경우 계산을 중단하는 대신 Dijkstra 알고리즘을 사용하여 목적지까지의 최단 경로를 계산하고, 그 후 장애물이 있으면 장애물이 없는 지점을 찾을 때까지 경로를 목적지에서 뒤로 이동한다.

그림 4는 navfn을 통해 배열형태로 생성된 global plan 경로에서 비용 값이 장애물이 있다고 판단되지 않을 때까지 배열의 마지막 요소를 삭제하여 적절한 도착지점을 찾는 모습이다. A 위치가 현재 로봇이 있는 위치이고, B위치가 사용자가 지정한 목적지이며, B지점은 도달할 수 없는 영역이어서 C지점까지 목적지를 변경하는 모습이다.

Ⅳ. 휠체어 로봇

본 논문에서는 상기 패키지들을 휠체어 타입의 로봇에 구현하였다. 그림 5는 휠체어 타입의 로봇 사진이고, 표 1에 로봇의 하드웨어 구성을 정리했다. 로봇이 이동하기 위해 로봇의 중앙에는 roboclaw 2×30A 모터 드라이버로 구동시키는 모터가 있고, 모터에는 로봇이 자신의 위치를 계산할수 있도록 엔코더가 부착되어 있으며, 앞뒤에는 캐스터 휠이 있다.

표 1. 하드웨어 구성
Table 1. Hardware configuration

Components	Model	Size (mm)
main controller	Intel NUC 10i5FNKN (NVME 250GB, RAM 8GB)	117 × 112 × 51
motor driver	roboclaw 2×30A	74 × 52 × 17
LiDAR	ydlidar G2	72 × 72 × 41
RGB-D camera	realsense d435i	90 × 25 × 25

네비게이션 시 장애물을 감지하기 위해 2D LiDAR 센서와 RGB-D 카메라가 사용된다. LiDAR는 ydlidar G2 모델로, 전면부 좌측과 우측 그리고 후면부 중앙에 총 3개를 부착하여 360도로 주변을 인식할 수 있도록 하였고, 라이다가 인식할 수 없는 3차원 환경을 인식하기 위해 전면 우측에 Intel사의 RealSense d435i RGB-D 카메라를 부착하였다.

로봇의 메인 컨트롤러는 Intel사의 NUC 10i5FNKN 보드를 사용하였다. 로봇 구동의 전체적인 시스템은 Ubuntu 18.04 운영체제의 ROS Melodic 버전 기반으로 구성하였다.

Ⅴ. 실험 결과

그림 6 (a)는 건물 복도를 가정 내라고 설정하고 상자를 이용해 가정 내에 있을 수 있는 장애물인 책상 형태의 장애물을 두었다. 이는 2D 센서를 사용하여 매핑했을 때와 3D 센서를 사용하여 매핑했을 때를 비교하기 위함이다. 그림 6 (b)는 SLAM gmapping 패키지와 2D LiDAR 센서를 사용하여 2D 지도를 작성한 모습이다. 그림 6 (b)를 보면, LiDAR 센서는 장애물의 아래 부분만 인식하여 윗부분은 매핑이 되지 않은 모습을 볼 수 있다. 그러나 그림 6 (c)를 보면,

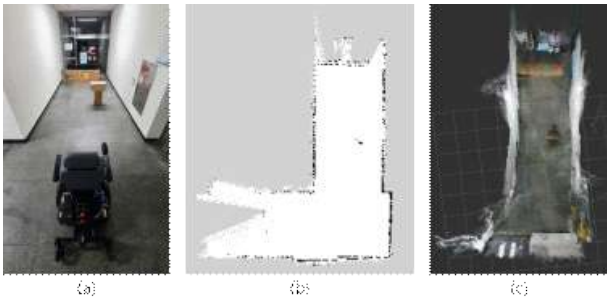

그림 6. (a) 환경 (b) 2D 지도 (c) 3D 지도
Fig. 6. (a) Environment (b) 2D map (c) 3D map

그림 7. Navfn을 사용한 내비게이션
Fig. 7. Navigation using Navfn

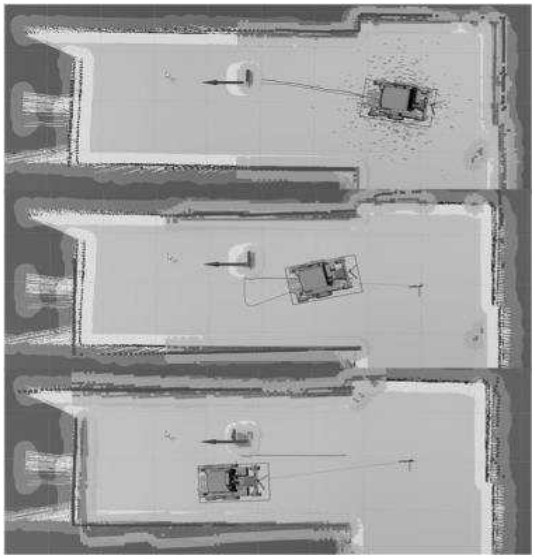

그림 8. asr_navfn을 사용한 자율주행
Fig. 8. Navigation using asr_navfn

RTAB-Map과 RGBD 카메라를 사용하여 3D 지도를 작성하고 RGBD 카메라는 3차원 포인트 클라우드를 생성하여 장애물 전체가 매핑이 된 것을 확인할 수 있다.

그림 7은 2D 지도에서 기존의 global planner 패키지인 Navfn을 move_base노드의 글로벌 플래너 플러그인으로 사용하여 자율주행을 한 모습으로, 장애물이 있는 위치를 목적지로 설정하면 주행경로를 생성하지 못하는 모습을 볼 수 있다.

그림 9. asr_navfn을 사용한 자율주행 영상 캡쳐한 모습
Fig. 9. Captured navigation video using asr_navfn

그림 8은 2D 지도에서 기존의 Navfn 패키지를 보완한 asr_navfn을 글로벌 플래너 플러그인으로 사용하여 자율주행을 한 모습으로, 장애물이 있는 위치에 목적지를 설정해도 목적지와 근접한 장소에 도착하는 모습을 확인할 수 있다.

그림 9는 2D 지도에서 실제 주행하는 모습의 영상을 캡쳐한 것으로, 그림 9와 같이 목적지 근처에 잘 도착하는 모습을 확인할 수 있다.

그림 7과 그림 9의 결과를 비교하면, 로봇이 위치한 지점을 (0,0), 목적지의 좌표점을 (3.2,-0.5)로 설정한 경우, navfn 패키지를 사용하면 목적지까지 도달거리가 3.24m이고 asr_navfn 패키지를 사용하면 약 1m가 되므로 asr_navfn 패키지가 경로 생성 성능이 크게 향상된 것을 확인할 수 있다.

Ⅵ. 결 론

본 논문에서는 휠체어 타입의 가정용 로봇에 기존의 map_server 패키지를 통해 생성했던 2D 맵을 Graph-Based SLAM인 RTAB-Map을 적용하여 3D 맵으로 생성하고, 장애물이 있는 위치에 목적지를 설정해도 Global plan을 생성할 수 있는 asr_navfn 패키지를 적용하여 자율주행을 수행할 수 있었다.

향후에는 주행하는 도중 사용자가 개입할 수 있도록 하는 알고리즘을 추가하여 사용자가 local map에 있는 장애물의 위치에 목적지를 설정하여도 목적지 근처에 가서 도착할 수 있도록 개선할 계획이다.

References

[1] https://ko.wikipedia.org/wiki/SLAM

[2] N. Kwak, B.-H. Lee, K. Yokoi, "Result Representation of Rao-Blackwellized Particle Filter for Mobile Robot SLAM," Journal of Korea Robotics Society, Vol. 3, No. 4, pp. 308-314, 2008 (in Korean).

[3] H. S. Kim, J. W. Kam, S. S. Hwang, "An Evaluation System to Determine the Completeness of a Space map Obtained by Visual SLAM," Korea Multimedia Society, Vol. 22, No. 4, pp. 417-423, 2019 (in Korean).

[4] S. Thrun, M. Montemerlo, "The Graph SLAM Algorithm with Applications to Large-scale Mapping of Urban Structures," International Journal of Robotics Research, Vol. 25, No. 5-6, pp. 403-429, 2006.

[5] J. B. Song, S. Y. Hwang, "Past and State-of-the-art SLAM Technologies," Journal of Institute of Control, Robotics and Systems, Vol. 20, No. 3, pp. 372-379, 2014 (in Korean).

[6] R. Mur-Artal, J. M. M. Montiel, J. D. Tardos. "ORB-SLAM: a Versatile and Accurate Monocular SLAM System." IEEE Tran. Robotics, Vol. 31. No. 5, pp. 1147-1163, 2015

[7] https://www.ros.org/

[8] https://github.com/introlab/rtabmap_ros

[9] N. Ragot, R. Khemmar, A. Pokala, R. Rossi, J. Y. Ertaud, "Benchmark of Visual Slam Algorithms: Orb-slam2 vs Rtab-map," International Conference on Emerging Security Technologies, pp. 1-6, 2019.

[10] https://github.com/asr-ros/asr_navfn

[11] M. Al-Nuaimi, S. Wibowo, H. Qu, J. Aitken, S. Veres, "Hybrid Verification Technique for Decision-making of Self-driving Vehicles," Journal of Sensor and Actuator Networks, Vol. 10, No. 3, pp. 42, 2021.

[12] https://edward0im.github.io/engineering/2020/09/08/pose-graph-optimization/

[13] https://en.wikipedia.org/wiki/Bag-of-words_model_in_computer_vision

[14] S. Byeon, J. Park, "A Resampling Method to Improve the Performance of AMCL Under a few Features Environment." Workshop of Korean Institute of Electrical Engineers, pp. 162-165, 2021 (in Korean).

[15] M. Labbé, F. Michaud, "Long-term Online Multi-session Graph-based SPLAM with Memory Management," Autonomous Robots, Vol. 42, No. 6, pp. 1133-1150, 2018.

[16] http://wiki.ros.org/navfn

[17] https://mattlee.tistory.com/50

[18] http://wiki.ros.org/rotate_recovery

Su-Jin Baek (백 수 진)

2020 Electronics Engineering from Dong-A University, Busan, Republic of Korea (B.S.)

Field of Interests: Robotics, Embedded system

Email: qortkdrnr@naver.com

A-Hyeon Kim (김 아 현)

2020 Electronics Engineering from Dong-A University, Busan, Republic of Korea (B.S.)

Field of Interests: Robotics, Embedded system

Email: rladkgus124@naver.com

Jong-Wook Kim (김 종 욱)

1997 Electronics and Electrical Engineering from POSTECH, Pohang, Republic of Korea (B.S.)

2000 Electronics and Electrical Engineering from POSTECH, Pohang, Republic of Korea (M.S.)

2004 Electronic and Electrical Engineering from POSTECH, Pohang, Republic of Korea (Ph.D.)

2006~Electronics Engineering, in Dong-A University (Prof.)

Career:

2021~Vice President of Korean Institute of Intelligent Systems

Field of Interests: Embedded system, Optimization algorithm, Robotics and Artificial intelligence

Email: kjwook@dau.ac.kr