Regular paper
Journal of the KIECS. pp. 637-644, vol. 14, no. 4, Aug. 31. 2019, t. 96, pISSN 1975-8170 | eISSN 2288-2189
http://dx.doi.org/10.13067/JKIECS.2019.14.4.637

일반 그래프 최적화를 활용한 그래프 기반 SLAM 구현

고낙용* · 정준혁** · 정다빈***

The Implementation of Graph-based SLAM Using General Graph Optimization

Nak-Yong Ko* · Jun-Hyuk Chung** · Da-Bin Jeong***

요 약

본 논문은 일반 그래프 최적화(g2o, General Graph Optimization)를 사용하여 그래프 기반 SLAM을 구현한 결과를 기술한다. 일반 그래프 최적화는 SLAM을 노드와 엣지의 그래프를 통하여 표현한다. 노드는 시간에 따른 로봇의 위치를 나타내며, 엣지는 노드들 사이의 구속 조건을 나타낸다. 구속 조건은 센서에 의한 측정값에 의해 결정된다. 일반 그래프 최적화는 구속 조건에 의해 결정되는 성능지표를 최적화하여 SLAM 문제를 해결한다. 실현된 일반 그래프 최적화 방법을 SLAM 방법의 성능 시험용으로 공개된 실험 데이터를 사용하여 검증하였다.

ABSTRACT

This paper describes an implementation of a graph-based simultaneous localization and mapping(SLAM) method called the General Graph Optimization. The General Graph Optimization formulates the SLAM problem using nodes and edges. The nodes represent the location and attitude of a robot in time sequence, and the edge between the nodes depict the constraint between the nodes. The constraints are imposed by sensor measurements. The General Graph Optimization solves the problem by optimizing the performance index determined by the constraints. The implementation is verified using the measurement data sets which are open for test of various SLAM methods.

키워드

SLAM, Invariant Extended Kalman Filter, Real Time, Mobile Robot
슬램, 불변 확장 칼만 필터, 실시간, 이동 로봇

Ⅰ. 서 론

로봇이 자율 주행을 하기 위해서는 주변 환경에 대한 지도 정보와 로봇의 위치 정보에 대하여 판단하는 기능이 필요하다. 주변 환경에 대한 지도 정보가 주어진다면 현재 위치를 추정하면서 자율 주행을 할 수 있다. 주변 환경에 대한 지도 정보가 없을 경우 사전 탐사를 통하여 위치를 추정함과 동시에 지도를 작성한다. 이렇게 위치 추정과 지도 작성을 동시에 하는 것을 SLAM(: Simultaneous localization and mapping)이라고 한다[1-3]. 최근 많이 발표되는 SLAM 관련 논문들은 칼만 필터나 파티클 필터와 같이 필터링을 이용해서 접근하는 방식보다 최소 자승

* 교신저자 : 조선대학교 전자공학부
** 조선대학교 대학원 제어계측공학과(junchi0925@gmail.com)
*** 조선대학교 대학원 전자공학과(jjdabin@naver.com)
· 접 수 일 : 2019. 06. 05
· 수정완료일 : 2019. 07. 10
· 게재확정일 : 2019. 08. 15
· Received : Jun. 05, 2019, Revised : Jul. 10, 2019, Accepted : Aug. 15, 2019
· Corresponding Author : Nak-Yong Ko
Dept. Electronics engineering, Chosun University.
Email : nyko@chosun.ac.kr

법에 기반으로 한 접근법을 주로 이루고 있다. 또 최근의 로봇은 다양한 SLAM 방법을 이용하기 때문에 기존의 접근법에 비해 위치와 지도를 보다 정확한 추정이 가능하게 한다[4].

SLAM에는 대표적으로 필터링(Filtering) 방법과 평활화(Smoothing) 방법이 있다. 먼저 필터링 방법은 실시간으로 데이터를 얻어 현재 주행 위치와 지도를 작성하는 방법이다. 필터링의 방법에는 EKF(: Extended Kalman filter) SLAM, IEKF(: Invariant Extended Kalman filter) SLAM, UKF(: Unscented Kalman filter) SLAM 방법 등이 있다[5-6]. 평활화 방법은 로봇이 주행을 완료한 후 얻은 데이터들을 사용하여 위치를 추정하고 지도를 작성하는 방법이다. 평활화 방법으로는 그래프 SLAM, 그래프 기반 SLAM 방법 등이 있다[7-8]. 그래프 기반 SLAM의 방법은 그래프 구성과 그래프 최적화 과정으로 되어 있다. 그래프 구성 과정은 이동체의 위치에 대해서 노드와 엣지를 구성하는 과정이다. 그래프 최적화 과정은 그래프 구성 과정에서 생성된 노드들과 엣지들을 최적화를 수행한다. 이 부분에서 노드는 이동체가 지나간 위치 또는 특징 위치를 나타내고, 엣지는 수신된 센서 정보를 바탕으로 노드와 노드 사이의 구속 조건을 의미한다[9].

정확한 SLAM을 위해서라면 로봇은 자신의 현재 위치를 정확히 인지해야 하고, 이동할 시 누적되는 오차를 줄여야만 한다. 위의 오차를 줄이는 방법으로는 대표적으로 루프 결합(Loop Closing) 방법이 있다. 하지만 이 방법은 오차룰 크게 줄일 수 있긴 하지만 계산량의 문제 때문에 실시간 처리를 하는 과정에 있어서 문제가 있다. 위의 문제를 해결하는 방법으로는 희소 행렬을 이용하여 계산량 문제를 해결하는 방법이 있다. 희소 행렬을 사용한 대표적인 방법으로는 ISAM(: Incremental Smoothing and Mapping)과 일반 그래프 최적화(g2o, General Graph Optimization)가 있다. ISAM과 일반 그래프 최적화는 루프 최적화라는 방법을 사용한다[10-13]. 최근 이용이 활발하게 증가하고 있는 도구는 C++로 작성된 일반 그래프 최적화 라이브러리다. 본 논문에서는 일반 그래프 최적화를 사용하여 그래프 기반 SLAM을 구현하였다. 본 논문의 2장에서는 그래프 기반 SLAM 방법을 기술한다. 3장에서는 일반 그래프 최적화에 대해 기술한다. 4장에서는 시뮬레이션 한 결과를 그림으로 나타내고 기술한다. 마지막으로 5장에서는 결론으로 마무리는 맺는다.

II. 그래프 기반 SLAM

이 그래프 기반 SLAM은 그래프 구성 과정과 그래프 최적화 과정으로 구성된다. 그래프 구성 과정은 이동 로봇의 위치에 대한 노드와 엣지를 구성하는 과정이다. 그래프 최적화 과정은 그래프 구성에서 생성된 노드들과 엣지들의 최적화를 수행한다. 여기서 노드는 이동 로봇이 지나간 위치 및 로봇의 자세를 나타내고, 엣지는 수신된 센서 정보로 얻어진 노드 간의 관계를 의미한다. 그림 1은 노드와 엣지의 관계를 나타낸 그림이다.

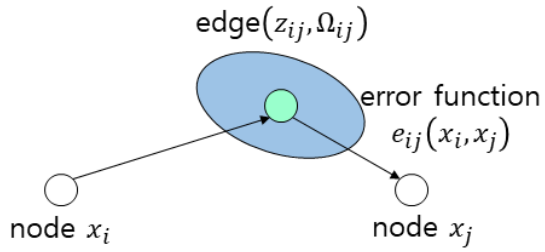


그림 1. 노드와 엣지
Fig. 1 Nodes and edges

i, j는 노드 인덱스를 의미하고 zij는 i번째 노드에서 바라본 j번째 노드의 가상 측정값을 의미한다. Ω ij는 i번째 노드와 j번째 노드의 정보 행렬을 의미하며 eij는 i번째 노드와 j번째 노드의 오차 함수를 의미한다. 여기서 오차 함수는 가상 측정값 zij와 예측한 노드 xi와 xj의 관계 zij_hat의 차이를 의미한다.

$$e_{ij}(X_i, X_j) = Z_{ij} - \hat{Z}_{ij}(X_i, X_j) \quad (1)$$

노드들의 집합 x를 구하기 위해 각 노드들에 대한 확률 밀도 함수를 구한다. 각 노드들의 확률 밀도 함수를 곱하여 가장 큰 값 x_star를 구한다. 각 노드들의 확률 밀도 함수을 곱한 결과에 negative log를 취해주면 상수 부분을 제외하면 식(4)과 같이 negative log likelihood F(x)가 나오게 된다.

negative log를 취했으므로 F(x)가 가장 작아지는 x_star를 구하면 된다.

$$pdf(x) = |2\pi\sum|^{-\frac{1}{2}} e^{-e^{T}ohm\frac{e}{2}} \quad (2)$$

$$x^{*} = \max imize\left(\Pi\,|2\pi\sum|^{-\frac{1}{2}} e^{-e^{T}ohm\frac{e}{2}}\right) \quad (3)$$

$$F(x) = \sum_{<i,j>\in C} e_{ij}^{T} ohm_{ij} e_{ij} \quad (4)$$

$$x^{*} = \arg\min F(x) \quad (5)$$

노드들의 집합 x를 최신화하기 위해 Δx를 구한다. Δx를 구하기 위해 F(x)에 x+Δx를 적용하여 구한다. 테일러 급수 전개를 사용하여 2차식까지 전개하여 식(6)과 같은 결과 식을 구한다. F(x+Δx)가 최소가 되는 Δx 값을 구하기 위하여 F(x+Δx)를 Δx에 대해 미분하여 0이 되는 값을 구한다. 식(6)의 맨 마지막 식을 미분하여 0이 되는 식을 구하면 식(7)과 같은 선형 시스템이 구해진다. 이 선형 시스템을 풀어서 Δx를 구하여 x를 최신화한다. 여기서 H는 확률론적 정보 행렬을 의미하며 b는 계수 벡터를 의미한다.

$$\begin{aligned} F(\tilde{x}+\triangle x) &= \sum_{<i,j>\in C} F_{ij}(\tilde{x}+\triangle x) \\ &\simeq \sum_{<i,j>\in C} c_{ij} + 2b_{ij}\triangle x + \triangle x^{T} H_{ij} \triangle x \\ &= c + 2b\triangle x + \triangle x^{T} H \triangle x \end{aligned} \quad (6)$$

$$H\triangle x^{*} = -b \quad (7)$$

Ⅲ. 일반 그래프 최적화

일반 그래프 최적화는 Rainer Kummerle가 개발한 그래프 기반 비선형 오차 함수를 최적화하기 위한 C++ 프레임워크이다. 일반 그래프 최적화는 광범위한 문제의 해결에 쉽게 적용할 수 있는 장점이 있다. 선형 시스템을 해결하기 위한 도구이며 최적화 문제를 해결해준다. SLAM 및 BA(: Bundle Adjustment)방법에 사용할 수 있다. linear solver로 CSparse, CHOLMOD, PCG 등이 있다. 그림 2은 일반 그래프 최적화 프로그램을 실행한 그림이다. optimizer 부분에서 linear solver를 선택할 수 있다. 그리고 iterations 부분에서는 반복횟수를 정할 수 있다. 일반 그래프 최적화의 2D data format은 그림 3의 (a)와 같다. VERTEX_SE2는 초기 노드 정보를 의미한다. FIX는 고정하려는 노드를 의미한다. 마지막으로 EDGE_SE2는 엣지 정보를 의미한다. 2차원 정보이므로 로봇의 2차원 좌표 x, y를 사용하며 자세 정보는 로봇의 헤딩 정보인 θ를 사용한다. 엣지 정보의 A11~A33은 3*3인 정보 행렬를 의미한다. 일반 그래프 최적화의 3D data format은 그림 3의 (b)와 같다. 2D data format과 유사하게 VERTEX_SE3:QUAT는 초기 노드 정보를 의미한다. FIX는 고정하려는 노드를 의미한다. 마지막으로 EDGE_SE3:QUAT는 엣지 정보를 의미한다. 3차원 정보이므로 로봇의 3차원 좌표 x, y, z를 사용하며 자세 정보는 quaternion을 사용한다. 엣지 정보의 A11~A66은 6*6인 정보 행렬를 의미한다.

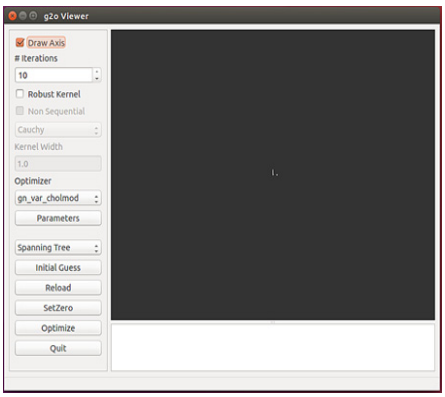


그림 2. 일반 그래프 최적화 Viewer
Fig. 2 General graph optimization viewer

```
VERTEX_SE2 'node number' 'x' 'y' 'θ'
FIX 'node number'
EDGE_SE2 'node i' 'node j' 'dx' 'dy' 'dθ' 'A11'
         'A12' 'A13' 'A22' 'A23' 'A33'
```

(a) 2D data format

```
VERTEX_SE3:QUAT 'node number' 'x' 'y' 'z' 'qx'
                'qy' 'qz' 'qw'
FIX 'node number'
EDGE_SE3:QUAT 'node i' 'node j' 'dx' 'dy' 'dz'
              'dqx' 'dqy' 'dqz' 'dqw' 'A11
              A12 ... A16 A22 A23 ... A26
              A33 A34 ... A36 A44 A45 A46
              A55 A56 A66'
```

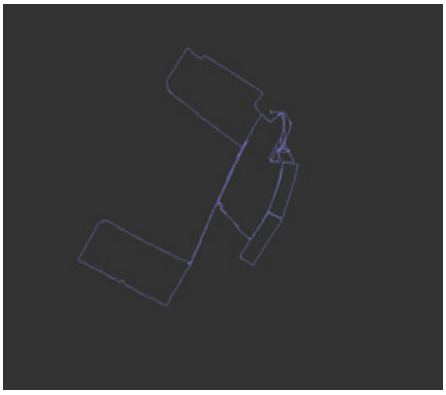
(b) 3D data format

그림 3. 일반 그래프 최적화 데이터 형태
Fig. 3 일반 그래프 최적화 data format

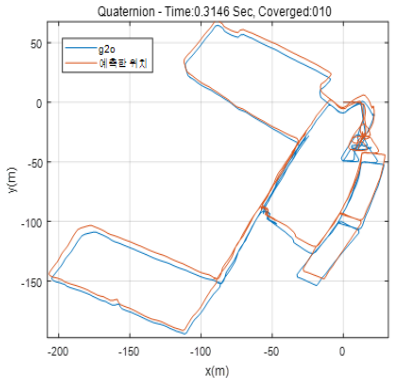
Ⅳ. 시뮬레이션 및 결과

본 절에서는 제안한 그래프 기반 SLAM 방법을 일반 그래프 최적화에 의해 구현하고 매트랩 기반으로 직접 프로그래밍한 시뮬레이션의 결과와 비교한다. 실험 데이터는 2D data와 3D data를 사용했다. 2D data는 IPB-Photogrammetry Lab의 'Intel Research Lab' dataset과 'MIT-CSAIL' dataset을 사용했다. 3D data는 MIT 대학의 Luca Carlone의 'Garage' dataset을 사용하였으며, 취리히 연방공과대학(ETH Zurich) Autonomous System Lab의 'Wood autumn' dataset과 'Stairs' dataset을 ICP(Iterative closest point) 방법에 적용하여 엣지 정보를 생성하여 사용했다. 그림 4는 'MIT-CSAIL' dataset을 일반 그래프 최적화 및 매트랩으로 구동한 결과이다. 그림 4의 (a)는 초기 노드들의 위치를 나타내며 (b)는 10번 최적화한 결과이다. (c)는 매트랩으로 10번 최적화한 결과로 청색 그래프는 일반 그래프 최적화의 결과이며 적색 그래프는 매트랩의 결과이다. 일반 그래프 최적화는 iteration 한번당 0.0011초가 걸렸고 매트랩은 0.3146초 걸렸다. 그림 5는 'Intel Research Lab' dataset을 일반 그래프 최적화 및 매트랩으로 구동한 결과이다. 그림 5의 (a)는 초기 노드들의 위치를 나타내며 (b)는 10번 최적화한 결과이다. (c)는 매트랩으로 10번 최적화한 결과로 청색 그래프는 일반 그래프 최적화의 결과이며 적색 그래프는 매트랩의 결과이다. 일반 그래프 최적화는 iteration 한번당 0.0021초가 걸렸고 매트랩은 0.6875초 걸렸다. 그림 6은 'Wood autumn' dataset을 일반 그래프 최적화 및 매트랩으로 구동한 결과이다. 그림 6의 (a)는 초기 노드들의 위치를 나타내며 (b)는 50번 최적화한 결과이다. (c)는 매트랩으로 50번 최적화한 결과로 녹색 그래프는 일반 그래프 최적화의 결과이며 적색 그래프는 매트랩의 결과이며 청색 그래프는 ground truth, 흑색 그래프는 ICP 결과로 dead-reckoning한 결과이다. 일반 그래프 최적화는 iteration 한번당 0.0002초가 걸렸고 매트랩은 0.0283초 걸렸다. 그림 7은 'Stairs' dataset을 일반 그래프 최적화 및 매트랩으로 구동한 결과이다. 그림 7의 (a)는 초기 노드들의 위치를 나타내며 (b)는 50번 최적화한 결과이다. (c)는 매트랩으로 50번 최적화한 결과로 녹색 그래프는 일반 그래프 최적화의 결과이며 적색 그래프는 매트랩의 결과이며 청색 그래프는 ground truth, 흑색 그래프는 ICP 결과로 dead-reckoning한 결과이다. 일반 그래프 최적화는 iteration 한번당 0.0003초가 걸렸고 매트랩은 0.0250초 걸렸다. 'Wood autumn' dataset과 'Stairs' dataset의 경우 ground truth와 비교하였을 때 정확도가 떨어지는 것을 확인할 수 있다. 이는 ICP 방법에 의해 만들어진 엣지 정보들의 신뢰성이 낮아서 위치 추정 결과의 정확성이 떨어진 것으로 분석된다. 그림 8은 'Garage' dataset을 일반 그래프 최적화 및 매트랩으로 구동한 결과이다. 그림 8의 (a)는 초기 노드들의 위치를 나타내며 (b)는 50번 최적화한 결과이다. (c)는 매트랩으로 50번 최적화한 결과로 청색 그래프는 일반 그래프 최적화의 결과이며 적색 그래프는 매트랩의 결과이다. 일반 그래프 최적화는 iteration 한번당 0.0235초가 걸렸고 매트랩은 10.1745초 걸렸다. 직접 만든 매트랩 시뮬레이션에서 'Garage' dataset의 경우 노드들의 집합인 x_bre를 최신화할 때 필요한 Δx의 값이 매우 커지는 현상이 발생하여 Δx의 값을 0.005배 하여 최신화하였다.

(a) optimize 전



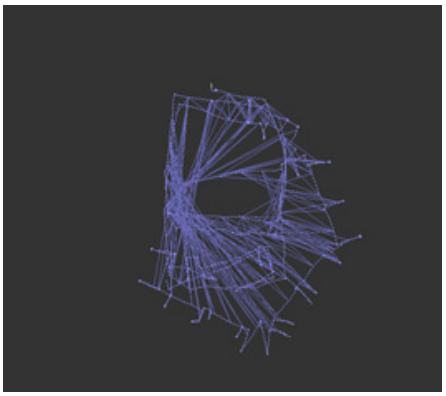
(b) optimize 후



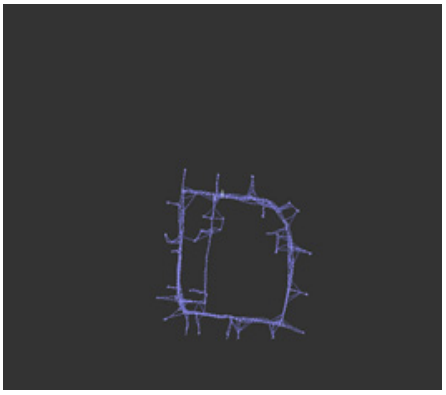
(c) 매트랩 구동 결과

그림 4. 'MIT-CSAIL' 일반 그래프 최적화 및 매트랩 구동 결과

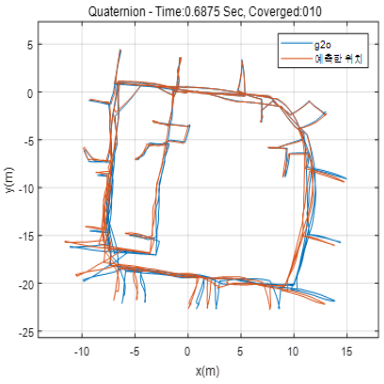
Fig. 4 'MIT-CSAIL' General graph optimization and matlab implementation results



(a) optimize 전



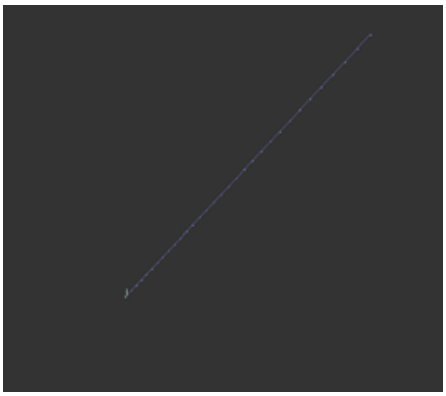
(b) optimize 후



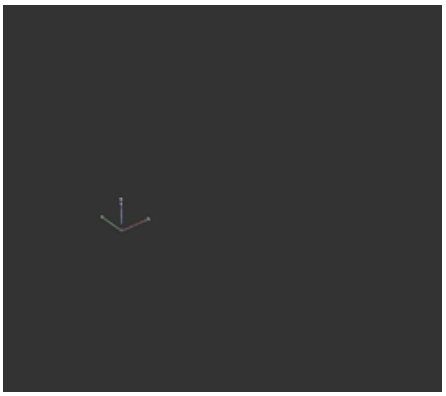
(c) 매트랩 구동 결과

그림 5. 'Intel Research Lab' 일반 그래프 최적화 및 매트랩 구동 결과

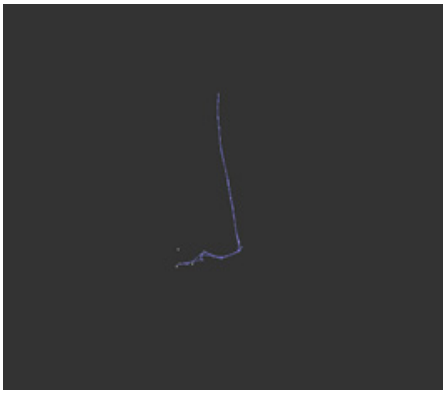
Fig. 5 'Intel Research Lab' General graph optimization and matlab implementation results



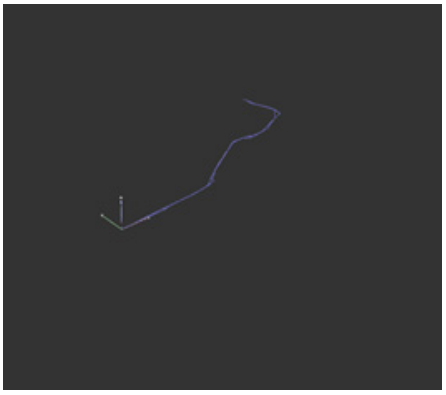
(a) optimize 전



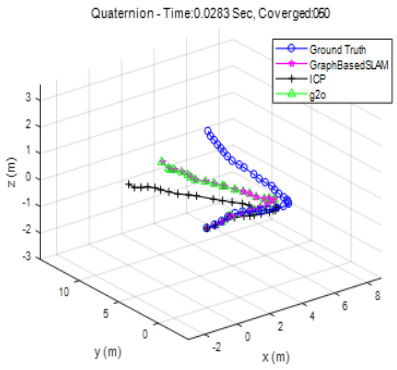
(a) optimize 전



(b) optimize 후



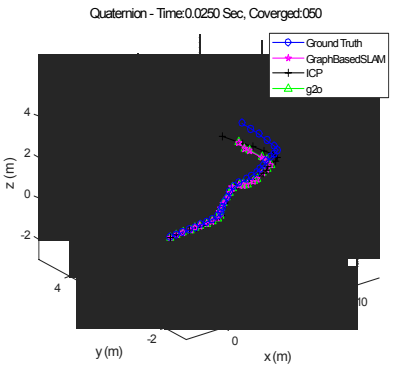
(b) optimize 후



(c) 매트랩 구동 결과

그림 6. 'Wood autumn' 일반 그래프 최적화 및 매트랩 구동 결과

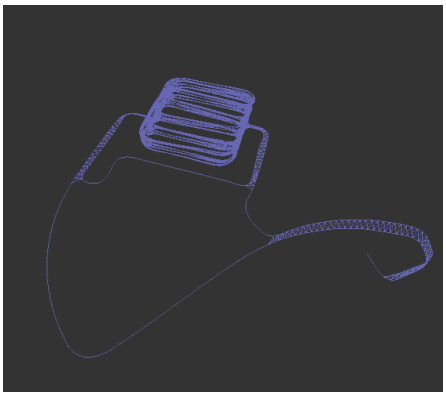
Fig. 6 'Wood autumn' General graph optimization and matlab implementation results



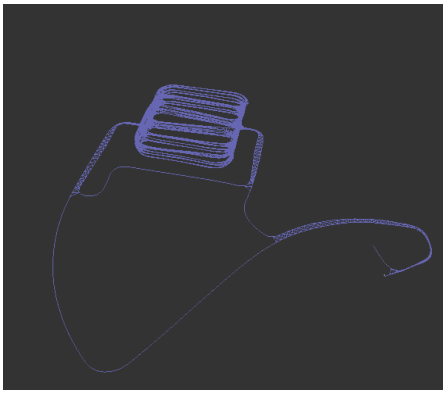
(c) 매트랩 구동 결과

그림 7. 'Stairs' 일반 그래프 최적화 및 매트랩 구동 결과

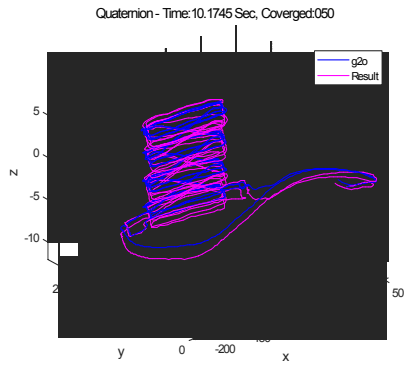
Fig. 7 'Stairs' General graph optimization and matlab implementation results



(a) optimize 전



(b) optimize 후



(c) 매트랩 구동 결과

그림 8. 'Garage' 일반 그래프 최적화 및 매트랩 구동 결과

Fig. 8 'Garage' General graph optimization and matlab implementation results

Ⅴ. 결 론

본 연구에서는 평활화 방법인 그래프 기반 SLAM 방법을 일반 그래프 최적화에 의해 구현되었다. 일반 그래프 최적화에서 구현한 그래프 기반 SLAM 방법이 ICP 특징점 추정에 의한 Dead-Reckoning 추정 방법보다 좋은 결과가 나왔다. 이러한 결과가 매트랩을 이용한 시뮬레이션 결과와 비슷한 결과가 나온 것을 확인할 수 있다. 하지만, Ground truth와 비교하였을 때 오차가 있다는 것을 확인할 수 있다. ICP 방법에 의해 만들어진 엣지 정보들의 신뢰성이 낮아서 위치 추정 결과의 정확성이 떨어진 것으로 분석된다. 향후, 노드들 사이의 변환을 구하는 다른 방법을 사용하여 엣지 정보들의 신뢰성을 높여 정확한 위치 추정을 수행할 예정이다. 또한, 다른 측정 데이터들을 사용하여 SLAM을 구연하는 연구를 수행할 예정이다.

감사의 글

이 논문은 2018학년도 조선대학교 학술연구비의 지원을 받아 연구되었음.

References

[1] J. Do and W. Lee, "ROS Based Remote Control of an Autonomous Mobile Robot and Visual SLAM For Parking Lot Management," *Institute of Control Robotics and Systems*, vol. 24, no. 11, Nov. 2018, pp. 1088-1093.

[2] G. Song, N. Ko, and H. Choi, "Attitude Estimation of Unmanned Vehicles Using Unscented Kalman Filter," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 14, no. 1, Feb. 2019, pp. 265-274.

[3] T. Kim and S. Hwang, "Cascade AOA Estimation Using Uniform Rectangular Array Antenna," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 13, no. 5, Oct. 2018, pp. 923-930.

[4] S. Mo, D. Yu, J. Park, and K. Chong, "Robust Mobile-Robot Localization for Indoor

SLAM," *Institute of Control Robotics and Systems*, vol. 22, no. 4, Apr. 2016, pp. 301-306.
[5] A. Barrau and S. Bonnabel, "An EKF-SLAM algorithm with consistency properties," *arXiv preprint arXiv*:1510.06263, 2015.
[6] T. Kim, N. Ko, and S. Noh, "Simultaneous Estimation of Landmark Location and Robot Pose Using Particle Filter Method," *International Journal of Fuzzy Logic and Intelligent Systems*, vol. 22, no. 3, 2012, pp. 353-360.
[7] S. Thrun and M. Montemerlo, "The graph SLAM algorithm with applications to large-scale mapping of urban structures," *Int. Journal of Robotics Research*, vol. 25, no. 5, 2006.
[8] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, "A tutorial on graph-based SLAM," *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, 2011, pp. 31-43.
[9] Y. Heo, J. Lim, and S. Lee, "EKF-based SLAM using sonar salient feature and line feature for mobile robots," *Journal of the Korean Society for Precision Engineering (in Korean)*, vol. 28, no. 10, 2011, pp. 1174-1180.
[10] M. Kaess, A. Ranganathan, and F. Dellaert, "iSAM: Incremental Smoothing and Mapping," *IEEE Transactions on Robotics (TRO)*, vol 24, no. 6, 2008, pp. 1365-1378.
[11] G. Grisetti, C. Stachniss, and W. Burgard, "Nonlinear Constraint Network Optimization for Efficient Map Learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 3, 2009, pp. 428-439.
[12] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g2o: A General Framework for graph optimization," *Proceedings on 2011 IEEE International Conference on Robotics and Automation (ICRA)*, 2011, pp. 3607-3613.
[13] S. Lee and B. Kim, "3D Simultaneous Localization and Map Building (SLAM) using a 2D laser range finder based on vertical/horizontal planar polygons," *Journal of institute of Control, Robotics and Systems (in Korean)*, vol. 20, no. 11, 2014, pp. 1153-1163.

저자 소개

고낙용(Nak-Yong Ko)

1985년 서울대학교 제어계측공학과 졸업(공학사)
1987년 서울대학교 대학원 제어계측학과 졸업(공학석사)
1993년 서울대학교 대학원 제어계측공학과 졸업(공학박사)
1997년, 2005년 Carnegie Mellon Univ., Visiting Research Scientist
1992. - 현재: 조선대학교 전자공학부 교수
※ 관심분야 : 로봇공학, 내비게이션, 추정, 제어

정준혁(Jun-Hyuk Chung)

2017년 조선대학교 제어계측로봇공학과 졸업(공학사)
2019년 조선대학교 대학원 제어계측공학과 졸업(공학석사)
※ 관심분야 : 로봇공학, 내비게이션, 추정

정다빈(Da-Bin Jeong)

2019년 조선대학교 전자공학과 졸업(공학사)
2019년 - 현재: 조선대학교 대학원 전자공학과 석사과정(석·박사 통합과정)
※ 관심분야 : 로봇공학, 내비게이션, 추정