2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Madrid, Spain, October 1-5, 2018

Scan Similarity-based Pose Graph Construction Method for Graph SLAM

Wonsok Yoo[1], Hanjun Kim[1], Hyunki Hong[1], and Beom H. Lee[2]

Abstract— **Scan similarity-based pose graph construction method for graph SLAM is proposed. To perform delicate pose graph SLAM, front-end that constructs a graph as well as back-end that optimizes the constructed graph is an important task. Generally, there is an error accumulation phenomenon during the odometry estimation process. This paper focuses on the method of creating a high quality graph by suggesting ways to improve the graph accuracy since the accumulated errors in the graph might degrade the performance of the entire graph SLAM. We deal with one of our previous works, dynamic keyframe selection technique, based on scan similarity computation method more precisely and suggest a loop closure detection method by exploiting previously proposed 2-D laser scan descriptor. To verify objective performance of the proposed method, the experimental results of the odometry estimation are shown by using the benchmark dataset and the real world dataset. Additionally, results of the pose graph SLAM are shown for the real world dataset which include the loop clorues.**

I. INTRODUCTION

Simultaneous Localization and Mapping (SLAM) problems have been researched by many researchers over two decades in robotics community and now has various applications ranging from exploration to autonomous driving [1]. SLAM problem approaches can be classified either as filtering or smoothing [2]. The first things include Kalman (or information) filters [3] and particle filters [4] approaches. The second smoothing approaches which are also called graph-based approaches estimate the full robot trajectory by using the full set of measurements [5]. Thus, these approaches are also known as full SLAM. The robot poses are represented by nodes and the measurement information is encoded by the edges. These graph representation is generally referred as pose graph. Both SLAM approaches have pros and cons regarding the accuracy and the computational resources. Of course the full SLAM approaches show better accuracy but require high computational power. Though these approaches are known to the community since the work of Lu and Milios [5], they have become more popular as efficient algorithms for solving the optimization problems are now available [6].

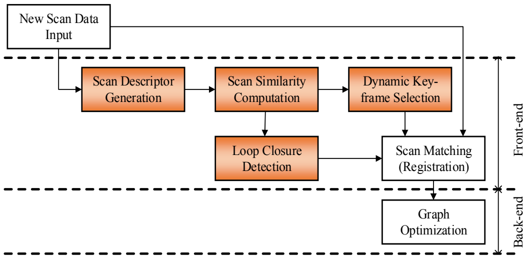

New Scan Data Input
Scan Descriptor Generation
Scan Similarity Computation
Dynamic Key-frame Selection
Loop Closure Detection
Scan Matching (Registration)
Graph Optimization
Front-end
Back-end

Fig. 1. The overall block diagram of the entire proposed method: main contributions of this paper are colored with orange.

To achieve high accuracy of the graph-based SLAM, front-end that construct a graph as well as back-end that optimize the constructed graph is an important task. Graph optimization is useless if the input graph contains fatal error information. Depending on the sensor usage, there are many graph construction methods. First of all, it can be divided by proprioceptive sensors and exteroceptive sensors [7]. The proprioceptive sensors include wheel encoder and IMU. These proprioceptive sensor senses own interactions with the world. On the other hand, the exteroceptive sensors senses the structure of the environment directly. These kinds of sensors include vision sensors and ranging sensors such as sonar and laser range finder (or LiDAR). There exist many researches from vision-based methods [8] to laser-based methods [9], [10] including hybrid method [11]. Most of the previous graph construction researches usually focused on the enhancement of the registration algorithm between two frame data. Conventionally, the graph construction methods have the local error accumulation problem. This causes the estimated robot poses to diverge since the error remains in the updated pose; the reference at the next step. Thus, reference frame selection is an important problem to enhance the accuracy of graph construction. As colored with orange in Fig. 1, this paper mainly focuses on the front-end of the entire graph SLAM framework especially on pair selection method not the registration technique.

In this paper, we deal with the 2-D LiDAR sensor platform. Building upon our previous work [12], we compared the performance of the odometry estimation based on the dynamic keyframe selection technique with the conventional *frame-to-frame(f2f)* [13] and *frame-to-keyframe(f2k)* methods [14]. This time, we used the benchmark dataset and the real world dataset instead of the simulation dataset. In addition, loop closure detection method is proposed by exploiting the 2-D laser scan descriptor and the scan similarity. Finally, pose graph SLAM results are shown by optimizing the constructed graph with g2o [15].

*This research was supported by a grant to Bio-Mimetic Robot Research Center Funded by Defense Acquisition Program Administration, and by Agency for Defense Development (UD160027ID), and in part by ASRI in Seoul National University

[1]Wonsok Yoo, Hanjun Kim, and HyunKi Hong, are with the Department of Electrical and Computer Engineering, Seoul National University, Republic of Korea yoowon0@snu.ac.kr

[2]Beom H. Lee, a Fellow of IEEE, is a Professor with the Department of Electrical and Computer Engineering, Seoul National University, Republic of Korea bhlee@snu.ac.kr

978-1-5386-8093-3/18/$31.00 ©2018 IEEE

II. RELATED WORK

The most naïve *f2f* pose graph construction method [13] updates the robot poses by using sequential frames. In general, registration results include small errors even in case of successful alignment because two scan data that obtained at different time step do not contain physically identical points. Thus, local drifts are inevitable in case of the *f2f* method. To alleviate local drift phenomenon, a *f2k* method has been proposed but still two problems are remained [16], [17]. First, once improperly updated keyframe is used, the pose estimation after that keyframe will diverge. Second, previously proposed keyframe update criteria, which can be applied to laser odometry estimation method, such as certain number of frames and time intervals [16], [17] are heuristic (displacement of the estimated pose between current and previous keyframes [18] is not always reasonable choice). Therefore, results of the odometry estimation are not guaranteed to be successful with the *f2k* method.

Also, many loop closing solutions (laser scan, RGB and both) are suggested to enhance the SLAM results. Loop closing which only relies on laser scan data is very challenging. Fast Laser Interest Region Transform (FLIRT) has been proposed [19] and the author developed place recognition algorithm using the FLIRT [20]. Also, the laser scan based loop closing, which uses machine learning, has been proposed [21]. This approach presented the features which are invariant to viewpoint and trained 360 degrees of laser scan data to detect the loop closure regardless of the viewpoint. Another loop closing method covering the large scale environment has been suggested by representing the geometrical relations into the pose invariant histograms [22]. Additionally, real-time loop closing method has been proposed by matching all scans to nearby submaps [23].

III. SIMILARITY OF 2-D LIDAR SCAN

A. Definition of the scan descriptor and similarity

To compute the similarity of two scan data, a scan descriptor which can represent each scan compressively is required. An arbitrary corner scan data of the time step k is shown in Fig. 2 (a)-(c). Let the points set in the kth scan $S_k = \left\{\mathbf{p}_b^k \mid b = 1, 2, \cdots, n_k\right\}$, where n_k is the number of the points in the scan data. For the ith point $\mathbf{p}_i^k = \left[p_{i,x}^k \quad p_{i,y}^k\right]'$ of the scan data, the nearest neighbors set is defined by

$$N_i^k = \left\{\mathbf{p}_{b'}^k \mid b' = i - \left\lfloor \frac{m_k}{2} \right\rfloor, \cdots, i, \cdots, i + \left\lfloor \frac{m_k - 1}{2} \right\rfloor \right\}, \quad (1)$$

where m_k is the number of nearest neighbors and b' is the integer index of the point data $\mathbf{p}_{b'}^k$. Here, the floor function $\lfloor \cdot \rfloor$ is defined as

$$\lfloor x \rfloor = \max\left\{m \in \mathbb{Z} \mid m \leq x\right\}, \quad (2)$$

where x is real numbers, m is integers and $\mathbb{Z}$ is the set of integers. Now, using the nearest neighbor set defined with (1) and (2), the structural information d_i^k around the point $\mathbf{p}_i^k$ can be described by

$$d_i^k = \frac{1}{\left|N_i^k\right|} \sum_{i'} \left\|\mathbf{e}_{i'}^k\right\|_2^2, \quad (3)$$

where i' satisfies $\mathbf{p}_{i'}^k \in N_i^k$ and $\mathbf{e}_{i'}^k$ is the error vector from the fitted line using N_i^k to the ith scan point. The above value (3) becomes a large near at corners or complex environments and a small around line data such as corridors. From Fig. 1 (d), the error vector of the point $\mathbf{p}_i^k$ can be represented by $\mathbf{e}_i^k = \left(\mathbf{n}_i^k \cdot \mathbf{p}_i^k - \mathbf{n}_i^k \cdot \mathbf{p}_c^k\right)\mathbf{n}_i^k$, where $\mathbf{n}_i^k$ is the normal vector of the line fitted by nearest neighbors set N_i^k and $\mathbf{p}_c^k$ is the virtual mean point on the fitted line. Thus, (3) can be represented by

$$\begin{aligned} d_i^k &= \frac{1}{\left|N_i^k\right|} \sum_{i'} \left|\mathbf{n}_i^k \cdot \mathbf{p}_{i'}^k - \mathbf{n}_i^k \cdot \mathbf{p}_c^k\right|^2 \\ &= \left(\mathbf{n}_i^k\right)^T \mathbf{C}_i^k \mathbf{n}_i^k, \end{aligned} \quad (4)$$

where i' satisfies $p_{i'}^k \in N_i^k$, $\mathbf{n}_i^k$ is the normal vector, and $\mathbf{C}_i^k$ denotes the covariance matrix of those points in N_i^k. From (4), the normal vector $\mathbf{n}_i^k$ can be substituted by the eigenvector $\mathbf{v}_{i,min}^k$ of the $\mathbf{C}_i^k$ which is corresponding to the minimum eigenvalue $\lambda_{i,min}^k$. Therefore, (4) can be simply represented by

$$\begin{aligned} d_i^k &= \left(\mathbf{v}_{i,min}^k\right)^T \mathbf{C}_i^k \mathbf{v}_{i,min}^k \\ &= \lambda_{i,min}^k. \end{aligned} \quad (5)$$

Based on the above (3), (4) and (5), the kth scan is defined by following vector:

$$\mathbf{D}^k = \left[d_1^k \quad d_2^k \quad \cdots \quad d_{n_k}^k\right]^T. \quad (6)$$

Now, the similarity between two scans S_u and S_v can be computed with the cross correlation coefficient by

$$\rho\left(\mathbf{D}^u, \mathbf{D}^v\right) = \frac{1}{n_u - 1} \sum_{l=1}^{n_u} \left(\frac{d_l^u - \mu_{\mathbf{D}^u}}{\sigma_{\mathbf{D}^u}}\right)\left(\frac{d_l^v - \mu_{\mathbf{D}^v}}{\sigma_{\mathbf{D}^v}}\right), \quad (7)$$

where $\mu_{\mathbf{D}^{(\cdot)}}$ and $\sigma_{\mathbf{D}^{(\cdot)}}$ are the mean and standard deviation of $\mathbf{D}^{(\cdot)}$ respectively. The suggested similarity computation method will be applied to both the dynamic keyframe selection technique and loop closure detection.

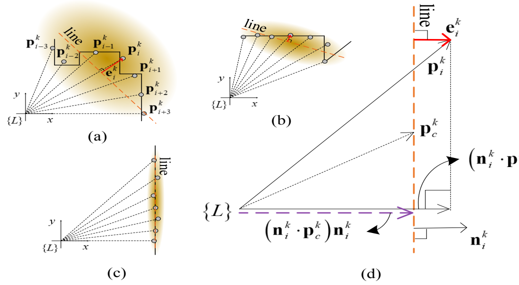


Fig. 2. Description of the scan descriptor element: (a)-(c) Line fitting with nearest neighbors set N_i^k of a point $\mathbf{p}_b^k$ and its error $\mathbf{e}_i^k$; (d) Simple error representation by using the normal vector $\mathbf{n}_i^k$ of fitted line and the virtual mean point $\mathbf{p}_c^k$.

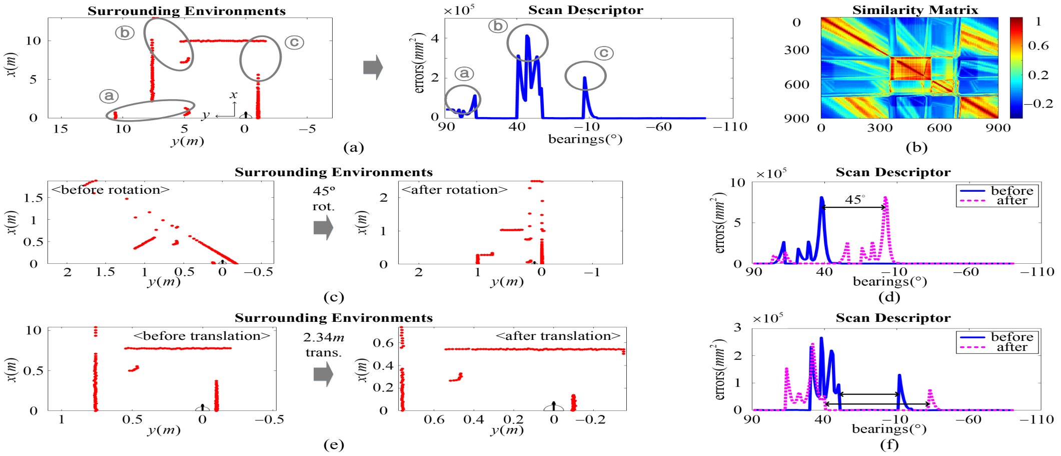


Fig. 3. Validation of the proposed scan descriptor: (a) The left plot shows the location of the robot at $(0,0)$ and the red dots represent the scan data points of the surrounding environments. The right plot shows the scan descriptor of the left scan data. The correspondences are marked by ⓐ, ⓑ and ⓒ; (b) Example of the similarity matrix for an arbitrary dataset; (c) Scan data before and after the 45° rotational motion; (d) Variation of the scan descriptor before and after the 45° rotational motion; (e) Scan data before and after the $2.34m$ translational motion; (f) Variation of the scan descriptor before and after the $2.34m$ translational motion.

B. Validation of the scan descriptor

Fig. 3 (a) shows an arbitrary scan and its descriptor. The left plot describes robot pose and the scan data of the surrounding environments. The right plot depicts the scan descriptor of the left scan. The horizontal axis of the right plot means the bearing of the sensor rays which ranges from $-90°$ to $90°$. Since data points of the scan are obtained per degrees, the number of the scan points is 181, which is same with the dimension of (6). Each correspondence is marked by ⓐ, ⓑ and ⓒ with gray ellipses in Fig. 3 (a). As defined in (3)-(5), the element of the proposed scan descriptor value is large near at the corners or complex environments and is small near at lines such as walls.

The shape variation of the descriptor according to the rotational and translational motion of the robot is shown in Fig. 3 (c), (d) and (e), (f). In case of rotational motion, the descriptor shifts along the bearing axis, and in case of translational motion, it stretches along the bearing axis. With these properties, the similarity computed by using (7). The similarity drops are depicted in Fig. 3 (b) for an arbitrary dataset. From this figure, it can be verified that the similarity between two scan data drops as robot moves.

IV. POSE GRAPH CONSTRUCTION METHOD

A. State transition model of pose graph construction

Generally, the state transition model is represented by

$$\hat{\mathbf{x}}_{k+1} = A_k\hat{\mathbf{x}}_k + B_k\mathbf{u}_k + \mathbf{w}_k, \tag{8}$$

where $\mathbf{u}_k$ means the control input between $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$ and $\mathbf{w}_k$ is the error. In case of scan matching algorithms, there is no control input and the errors originate from the measurement noise which follows the Gaussian if the favorable alignment is assumed. Thus, the state transition model can be represented by

$$\hat{\mathbf{x}}_{k+1} = H_{k+1}^{k}\hat{\mathbf{x}}_k + \mathbf{w}, \tag{9}$$

where H_{k+1}^{k} is the homogeneous form true transformation matrix between $\mathbf{x}_k$ and $\mathbf{x}_{k+1}$. Now, this form can be represented recursively as

$$\begin{aligned} \hat{\mathbf{x}}_k &= H_k^{k-1}\hat{\mathbf{x}}_{k-1} + \mathbf{w} \\ &\vdots \\ \hat{\mathbf{x}}_1 &= H_1^0\mathbf{x}_0 + \mathbf{w}. \end{aligned} \tag{10}$$

Substituting (10) to (9) then

$$\begin{aligned} \hat{\mathbf{x}}_{k+1} = &H_{k+1}^{k}H_k^{k-1}\cdots H_1^0\mathbf{x}_0 \\ &+ (H_{k+1}^{k}H_k^{k-1}\cdots H_2^1 + H_{k+1}^{k}H_k^{k-1} \\ &\cdots H_3^2 + \cdots + H_{k+1}^{k}H_k^{k-1} + H_{k+1}^{k} + \mathbf{1})\mathbf{w}. \end{aligned} \tag{11}$$

From (11), it can be verified that the errors are accumulated as the number of registration increases.

B. Dynamic keyframe selection technique

Proposing dynamic keyframe selection technique determines appropriate reference scan every time step to alleviate the error accumulation. The scan matching algorithms generally show analogous registration errors if the similarity of two scans is more than a certain degree. The closest scan data to the initial frame's time step among reference candidates is likely to have the highest certainty. Therefore, the k'th scan having the highest certainty which exceeds the user defined similarity threshold τ among the reference candidates is selected as the keyframe by

$$k^* = \min_{k' \in M_c} k', \tag{12}$$

where

$$M_c = \left\{ v' \in \mathbb{Z}^+ \mid \rho(\mathbf{D}^c, \mathbf{D}^{v'}) \geq \tau,\, v' = c - m_r, \ldots, c-1 \right\}. \tag{13}$$

The entire process is described in Algorithm 1. Since the scan similarity (7) ranges from -1 to 1, the *max* value

Algorithm 1: Dynamic keyframe selection

```
   Input: $m_r, \tau, S_{k-m_r}, S_2, ..., S_k$
   Output: $c'$
1  Initialize $max \leftarrow -2$
2  Compute $\mathbf{D}^k$ from $S_k$                ▷ The $k$-th scan descriptor
3  for $i \leftarrow k - m_r$ to $k-1$ do
4      Compute $\mathbf{D}^i$ from $S_i$            ▷ The $i$-th scan descriptor
5      if $\rho(\mathbf{D}^k, \mathbf{D}^i) \geq \tau$ then
6          $c' \leftarrow i$
7          break             ▷ Break when similarity exceeds $\tau$
8      else
9          if $\rho(\mathbf{D}^k, \mathbf{D}^i) > max$ then
10             $max \leftarrow \rho(\mathbf{D}^k, \mathbf{D}^i)$
11             $c' \leftarrow i$          ▷ To cope with exceptions
12 return $c'$              ▷ Return the final reference scan index
```

Algorithm 2: The # of representatives determination

```
   Input: $S_k, v_{t-1}, \omega_{t-1}, t_s, \delta_{t,max}, \delta_{r,max}$
   Output: $m_r$
1  Initialize $m_r \leftarrow 1$, $t_r \leftarrow 1$, $\delta_t \leftarrow v_{t-1} \cdot t_s$, $\delta_r \leftarrow \omega_{t-1} \cdot t_s$
2  $r_{min} \leftarrow minimum(r_b^k \mid b = 1, 2, ..., n_k)$
3  if $|\omega_{t-1} \cdot t_s| > 0.2$ AND $|\omega_{t-2} \cdot t_s| \leq 0.2$ then
4      $t_r \leftarrow t - 1$            ▷ Rotation starting time logging
5  while true do
6      if $\delta_t \geq \delta_{t,max}$ OR $|\delta_r| \geq \delta_{r,max}$ then
7          break            ▷ Until maximum motion variations
8      $\delta_t \leftarrow \delta_t + \|\hat{\mathbf{x}}_{k-m_r} - \hat{\mathbf{x}}_{k-(m_r+1)}\|_2$
9      $\delta_r \leftarrow \delta_r + (\hat{\theta}_{k-m_r} - \hat{\theta}_{k-(m_r+1)})$
10     $m_r \leftarrow m_r + 1$
11     if $(t - t_r) \leq m_r$ then
12         break   ▷ When iteration reaches rotation starting time
13 return $m_r$
```

which represents the maximum similarity is initialized to -2. A value of τ that is too large will result in a short interval registration pair, leading to error accumulation, and vice versa. To deal with the exceptional $M_k = \emptyset$ case, the maximum similarity value is recorded and the time index which brought the maximum value is used as a substitute for the reference index.

The proposing technique requires the number of basic model candidates m_r in line 3. Naïvely, constant m_r can be used when the robot moves at the same speed. However, constant m_r cannot cover the movement of the robot with speed variation. Thus, m_r is set by counting up until the angular displacement δ_r from the current time step to the previously recorded time step exceeds user defined angle $\delta_{r,max}$ or the translational displacement δ_t exceeds the following:

$$\delta_{t,max} = \begin{cases} \alpha \tan^{-1}\left(\frac{r_{min}-\gamma_{th}}{100}\right) + \varepsilon & \text{if } r_{min} \geq \gamma_{th}, \\ \varepsilon & \text{otherwise,} \end{cases} \tag{14}$$

where r_{min} is the median value of the lower 5% of the distance information of the current scan data and $\alpha, \gamma_{th}, \varepsilon$ are hyper parameters. The value $\delta_{t,max}$ is dependent on r_{min} since short range data disappear quickly in the FOV of the LiDAR compared with the long range data. Sometimes, the short range data cause occlusions so that common region between scan data becomes small as robot moves. Consequently, small $\delta_{t,max}$ can prevent registration failure by sustaining enough common data points. The α and ε are concerned with the maximum and minimum of the curve respectively and γ_{th} is the threshold that decides the cutting point of the range data. The entire procedure for determining the number of representatives m_r is described in Algorithm 2.

C. Loop closure detection using 2-D laser scan descriptor

During the odometry estimation process, the 2-D laser scan descriptors are pre-computed. By using those values, loop closures can be detected performing registration between current scan and loop closure candidate scan data. The loop closure candidate scan set is defined as

$$M_{lc} = \left\{ v' \in \mathbb{Z}^+ \mid \rho(\mathbf{D}^c, \mathbf{D}^{v'}) \geq \tau_{lc}, \left\| \hat{\mathbf{x}}'_c - \hat{\mathbf{x}}'_{v'} \right\|_2 \leq \gamma_{lc} \right\}, \tag{15}$$

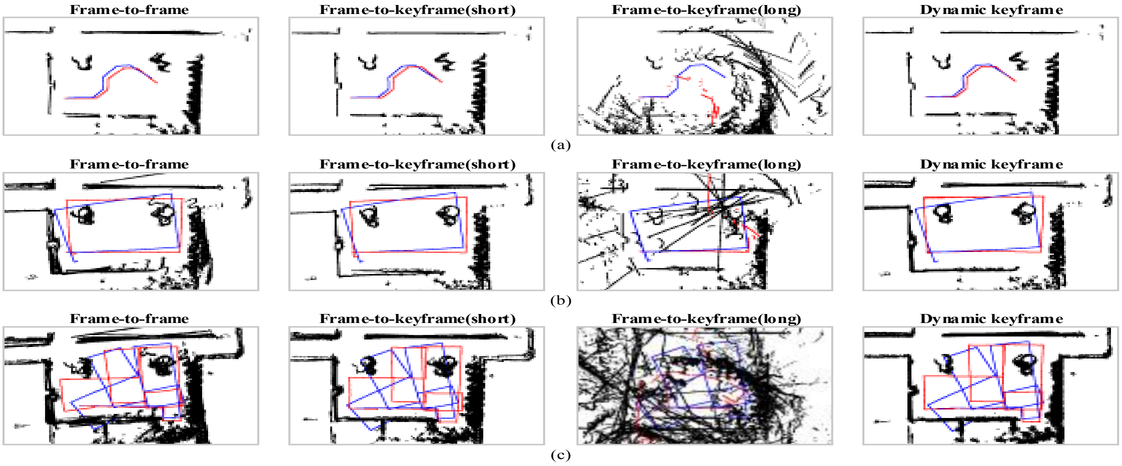
where $\hat{\mathbf{x}}'_c$ and $\hat{\mathbf{x}}'_{v'}$ denote the estimated robot poses at current time step c and v' respectively, and τ_{lc}, γ_{lc} are loop closure similarity and range threshold respectively.

V. EXPERIMENTAL RESULTS

A. Experimental results of the odometry estimation

First, experiments of the odometry estimation with the Bicocca benchmark dataset [24] have been performed to verify objective performance of the proposed method. We used 'Bicocca-2009-02-25b' dataset among five dataset. This dataset is a large scale environment with a total travel distance of 774 meters. Since the drift phenomenon is inevitable during the odometry estimation process, the dataset is divided into 5 segments. Performance of the proposed *frame-to-dynamic keyframe(f2d)* method is compared with the conventional *f2f* and *f2k* methods. In case of the *f2k* method, two keyframe update critera have been used. The proposed method was implemented by C++ on a PC with Intel Core i7-3770 quad-core 3.4GHz, 8GB memory. The input scan data are registered by using generalized-ICP [9]. The $f2k(short)$ and $f2k(long)$ methods have updated the keyframe every 100mm or 1°, and 500mm or 15°, respectively. As shown in Table I, it can be verified that the proposed *f2d* method has shown the least RMS of Absolute Trajectory Error(ATE) for all the five segments compared with the previous *f2f* and *f2k* methods. The average computation time is shown in Table II. On the other hand, the other three methods took more computation time since they need more computation power to select keyframes. However, the proposed method has shown more accurate results with similar computation time to the *f2k* methods.

Moreover, the odometry estimation experiments are conducted using the real world dataset which are collected at the 3rd floor of Automation and Systems Research Institute (ASRI), Seoul National University. This is challenging environments with occlusions caused by two pillars, various



(a)

(b)

(c)

Fig. 4. Odometry estimation results of *f2f*, *f2k(short)*, *f2k(long)* and the proposed *f2d* methods with real world dataset: (a) SNU ASRI No. 1 dataset; (b) SNU ASRI No. 2 dataset; (c) SNU ASRI No. 3 dataset.

TABLE I

RMS OF ATE COMPARISON ('BICOCCA-2009-02-25B') [UNIT: *mm*]

Dataset	*f2f*	*f2k(short)*	*f2k(long)*	*f2d*
1st segment	1436	1091	29821	1080
2nd segment	3022	3578	23240	263
3rd segment	2022	565	11754	394
4th segment	4560	3254	46905	1287
5th segment	2544	1928	28026	788

TABLE II

AVG. COMPUTATION TIME ('BICOCCA-2009-02-25B') [UNIT: *sec.*]

Dataset	*f2f*	*f2k(short)*	*f2k(long)*	*f2d*
1st segment	0.031	0.076	0.088	0.064
2nd segment	0.029	0.070	0.093	0.071
3rd segment	0.028	0.058	0.077	0.072
4th segment	0.024	0.045	0.078	0.076
5th segment	0.024	0.056	0.079	0.080

TABLE III

ERROR COMPARISON OF THE ODOMETRY ESTIMATION [UNIT: *mm*]

Dataset	*f2f*	*f2k(short)*	*f2k(long)*	*f2d*
ASRI No.1	503	443	3329	404
ASRI No.2	380	376	7523	211
ASRI No.3	436	715	8225	369

TABLE IV

AVG. COMPUTATION TIME OF THE ODOMETRY ESTIMATION [UNIT: *sec.*]

Dataset	*f2f*	*f2k(short)*	*f2k(long)*	*f2d*
ASRI No. 1	0.030	0.036	0.047	0.040
ASRI No. 2	0.030	0.040	0.044	0.046
ASRI No. 3	0.030	0.036	0.043	0.043

entrance exhibits, and glass walls which the LiDAR shows weakness. Three different real world scan dataset which has different robot path were used to evaluate the performance of the proposed graph construction method. Each dataset is named as ASRI No. 1(no loop closure), ASRI No. 2(single loop closure) and ASRI No. 3(five loop closures). Fig. 4 shows the odometry estimation result of the conventional methods and the proposed method using the three dataset. Here, the blue points represent the estimated robot poses by wheel encoders and the red points represent the estimated poses through scan data registration. The black points are scan data which are plotted at the estimated poses. From these results, the proposed dynamic keyframe method has shown enhancement compared with the conventional methods. The enhancement grows as the traveled distance increases. For all the dataset, the results of $f2k(long)$ method have shown complete failure because of the fixed and long keyframe update criteria

Finally, to verify the performance of the proposed method, the errors between the estimated poses and the near-ground truth poses are measured. Since it is almost impossible to obtain the ground truth data in the real world, the corner poses in the robot paths are measured and used as the near-ground truth data. The error and the average computation time for the above experiment results of the pose graph construction are presented in Table III and Table IV, respectively. From these results, it can be verified that the proposed *f2d* method has shown the least errors for all the three dataset with slight longer average computation time compared with the previous *f2f* and *f2k* methods.

B. Experimental results of the pose graph SLAM

Odometry is estimated based on the proposed dynamic keyframe selection technique and the loop closures are detected by exploiting the scan descriptor and similarity computation method. The graphs are optimized using the g2o [15]. Also, the information matrix between two poses are computed with Hessian method [25].

In Fig. 5, near-ground truth is represented by green '×' symbol. The wheel odometry trajectories are plotted by orange dotted lines and the results of the trajectories with the proposed pose graph construction method are plotted by blue dashed lines. The red solid lines show the trajectories

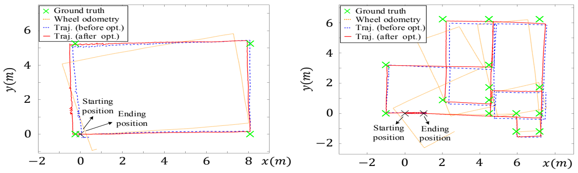


Fig. 5. Pose graph SLAM results with real world dataset.

TABLE V

ERROR COMPARISON OF THE POSE GRAPH SLAM [UNIT: *mm*]

Dataset	Wheel odo.	*f2d* odo.	*f2d* SLAM
ASRI No.2	1057	211	182
ASRI No.3	929	369	237

of the graph-based SLAM results after the optimization. Of course, the errors are decreased after the optimization and these are described in Table V.

VI. CONCLUSION

This paper presented pose graph construction method for graph SLAM with 2-D LiDAR sensor. To construct high quality pose graph, scan similarity calculation method using scan descriptor was presented and its validation was performed with arbitrary scan data. The error accumulation problem was described with state transition model of the pose graph construction. By using the scan similarity calculation method, the dynamic keyframe selection technique which can reduce the error accumulation of the pose graph generation is described. Moreover, the loop closure detection method has proposed by exploiting the scan similarity computation method. Through the experiments, the performance of the odometry estimation based on dynamic keyframe selection technique was verified. Finally, the results of the pose graph SLAM were shown by applying the proposed loop closure detection method and graph optimization.

Further research on loop closure detection should be done to overcome the limitation regarding the robot path. After making up the loop closure detection method, we are planning to compare the proposed graph SLAM algorithm with other state-of-art SLAM algorithms.

REFERENCES

[1] G. Bresson, Z. Alsayed, L. Yu, and S. Glaser, “Simultaneous localization and mapping: A survey of current trends in autonomous driving,” *IEEE Transactions on Intelligent Vehicles*, vol. 2, no. 3, pp. 194–220, 2017.

[2] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard, “A tutorial on graph-based slam,” *IEEE Intelligent Transportation Systems Magazine*, vol. 2, no. 4, pp. 31–43, 2010.

[3] S. Thrun, D. Koller, Z. Ghahramani, H. Durrant-Whyte, and A. Y. Ng, “Simultaneous mapping and localization with sparse extended information filters: Theory and initial results,” *International Journal of Robotics Research*, vol. 23, no. 7–8, pp. 693–716, 2004.

[4] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit, “FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges,” in *Proceedings of the International Joint Conference on Artificial Intelligence*, (Acapulco, Mexico), pp. 1151–1156, 2003.

[5] F. Lu and E. Milios, “Globally consistent range scan alignment for environment mapping,” *Autonomous Robots*, vol. 4, no. 4, pp. 333–349, 1997.

[6] N. Sünderhauf and P. Protzel, “Towards a robust back-end for pose graph slam,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Saint Paul, USA), pp. 1254–1261, 2012.

[7] T. Duckett and U. Nehmzow, “Mobile robot self-localisation and measurement of performance in middle-scale environments,” *Robotics and Autonomous Systems*, vol. 24, no. 1-2, pp. 57–69, 1998.

[8] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha, “Visual simultaneous localization and mapping: a survey,” *Artificial Intelligence Review*, vol. 43, no. 1, pp. 55–81, 2015.

[9] A. V. Segal, D. Haehnel, and S. Thrun, “Generalized-ICP,” in *Proceedings of Robotics: Science and Systems*, (Seattle, USA), 2009.

[10] J. Zhang and S. Singh, “Visual-lidar odometry and mapping: Low-drift, robust, and fast,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Seattle, Washington), pp. 2174–2181, IEEE, 2015.

[11] M. Korn, M. Holzkothen, and J. Pauli, “Color Supported Generalized-ICP,” in *Proceedings of International Conference on Computer Vision Theory and Applications VISAPP*, (Lisbon, Portugal), pp. 592–599, 2014.

[12] W. S. Yoo and B. H. Lee, “Dynamic key-frame selection technique for enhanced odometry estimation based on laser scan similarity comparison,” *Electronics Letters*, vol. 53, no. 13, pp. 852–854, 2017.

[13] K. Konolige and M. Agrawal, “Frame-frame matching for realtime consistent visual mapping,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Roma, Italy), pp. 2803–2810, 2007.

[14] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, “Keyframe-based visual–inertial odometry using nonlinear optimization,” *International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[15] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g 2 o: A general framework for graph optimization,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Shanghai, China), pp. 3607–3613, 2011.

[16] D. Nister, O. Naroditsky, and J. Bergen, “Visual odometry,” in *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, (Washington, USA), pp. I–652–I–659, 2004.

[17] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *Proceedings of IEEE and ACM International Symposium on Mixed and Augmented Reality*, (Nara, Japan), pp. 225–234, 2007.

[18] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, “Keyframe-based dense planar slam,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Singapore), pp. 5110–5117, 2017.

[19] G. D. Tipaldi and K. O. Arras, “Flirt-interest regions for 2d range data,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Anchorage, USA), pp. 3616–3622, 2010.

[20] G. D. Tipaldi, L. Spinello, and W. Burgard, “Geometrical flirt phrases for large scale place recognition in 2d range data,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Karlsruhe, Germany), pp. 2693–2698, 2013.

[21] K. Granstrom, J. Callmer, F. Ramos, and J. Nieto, “Learning to detect loop closure from range data,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Kobe, Japan), pp. 15–22, 2009.

[22] M. Himstedt, J. Frost, S. Hellbach, H.-J. Böhme, and E. Maehle, “Large scale place recognition in 2d lidar scans using geometrical landmark relations,” in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, (Chicago, USA), pp. 5030–5035, 2014.

[23] W. Hess, D. Kohler, H. Rapp, and D. Andor, “Real-time loop closure in 2d lidar slam,” in *Proceedings of IEEE International Conference on Robotics and Automation*, (Stockholm, Sweden), pp. 1271–1278, 2016.

[24] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, and J. D. Tardos, “Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets,” in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems Workshop on Benchmarks in Robotics Research*, 2006.

[25] O. Bengtsson and A.-J. Baerveldt, “Localization in changing environments-estimation of a covariance matrix for the idc algorithm,” in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 4, (Maui, USA), pp. 1931–1937, 2001.