MINGI JEONG
F00422M
Mingi.Jeong.GR@dartmouth.edu

1 Description

1.1 Introduction

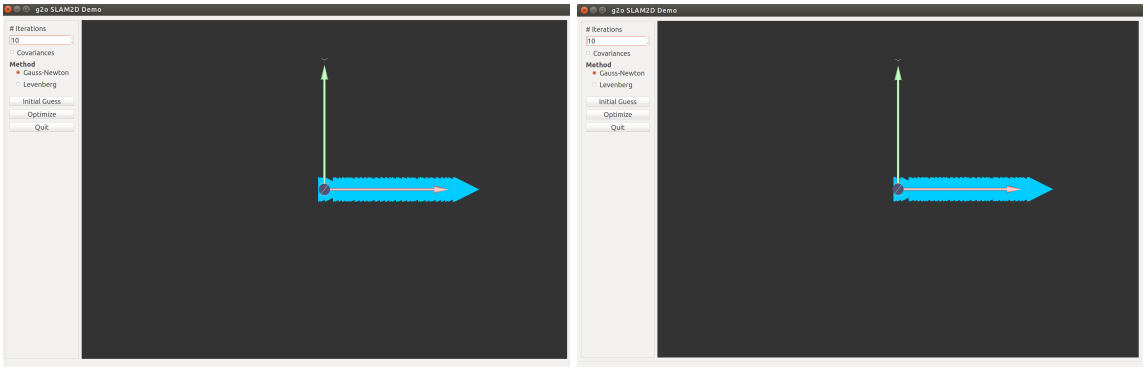
This program was made to implement the codes of Rosbot's state graph optimization based on g2o library. Assuming that the robot is following the movement in 1-D system from the same rosbag file as PA-2, the graph optimization algorithm return the estimate as g2o format consisting of nodes and edges. Based on constructed graph(`pose`) and measurement (lidar scan), the program initiates the optimization. For the launch file, you can pass saving path for data as well as other parameters such as measurements and covariances. Only one launch file can do everything needed in this assignment. The parameters for the graph optimization are based on the initial belief and measurement errors. Finally a user can check the result by using slam2d_g2o as well as matplotlib.

1.2 How my program works

- Please refer to the readme on github `https://github.com/MingiJeong/graph_optimization_mg_cs169`
- rosbag file link `https://drive.google.com/open?id=1onVc88q--nnUFm7Kv0Schqn1i-yoMVre`
- After all configuration (including designation of saving path) is done with download, you can just launch "roslaunch graph_optimization_mg_cs169 graph_optimization.launch.
- I made a delay for plotting node (15 sec) since the rosbag file last aroung 12 sec. You will see plots popping up and data (g2o, csv, pdf plot) saved in a designated folder.

1.3 Design Decision

- Basically, this program assumes that the robot navigates in 1-D world even if the odom data is technically based on 2-D inputs (x,y) (actually, it contains 3-D, but the robot moves in 2-D on a plane; that's why I mentioned 2-D). Therefore, in a same way in PA-2, I used 1-D accumulated distance from starting point as of **"state"** we are interested in attaining by the graph optimization [1].
- A system model using `pose` is based on data inputted by a wheel odometry. The poses of the robot itself become nodes of this algorithm. In addition, the pose differences (Euclidean distance) compose the edges between the nodes made of the poses.
- On the other hand, the differences between scan data Z_t-Z_{t+1} become the other edges connecting the existing nodes [2].
- The initial time 0 is based on the first time when cmd_vel was published.
- To construct the graph as 2-D, I had to input g2o format poses such as g2o(x, y, theta). In terms of edges, I used 3 x 3 identity matrix and changed (0,0) index to the values as inverse of covariance matrix Q, R from PA-2.
- The lidar scan data are from the straightforawrd measurement as of index 0. Some outliers when the lidar detects infinity are excluded for connecting edges [1].
- For time synchronization of the signals, I measured time differences and interpolated. That is, the poses are adjusted to the scan time series so that I can compare those two data under a normalized condition.
- All in all, the data are stored as csv and g2o format. The ground truth is only saved as csv.
- As soon as the data saving is completed, the plotter node in the launch file initiate visualization of plots: path comparison and error comparison.



(a) (b)

Figure 1: Graph as g2o format (a) before optimization based on pose (b) after optimization inclusive of scan

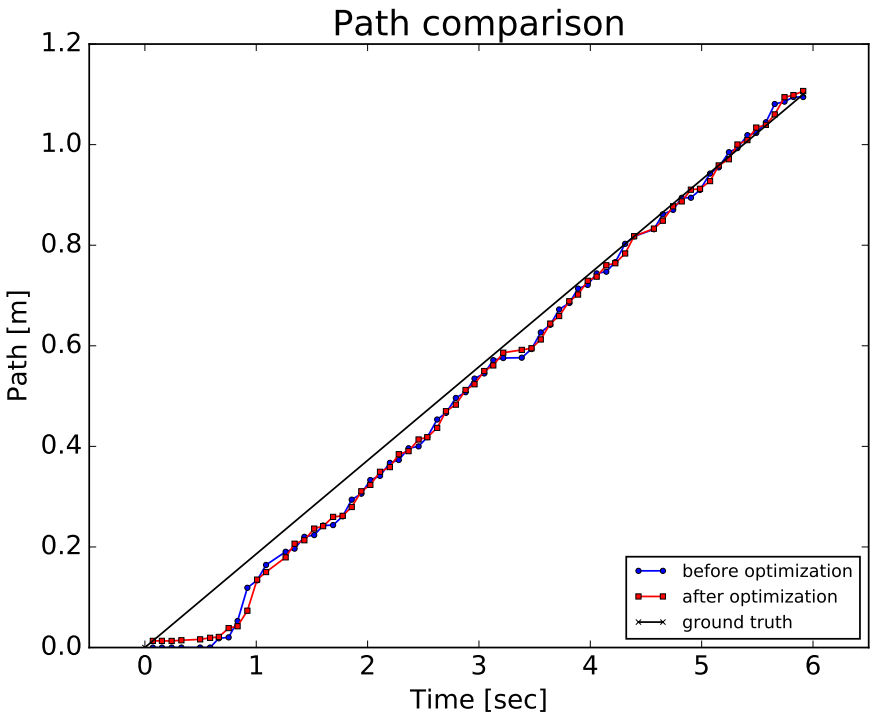


Figure 2: Result of path comparison between before optimization and after optimization

2 Evaluation

2.1 Performance

- As for an information matrix of constraints by either poses or scans, I chose the values same as PA-2. In other words, 1/Q and 1/R value was inputted to the information matrix.
- The first several time stamps show that the optimized algorithm has some more errors than non-optimized one. It is because there is an offset distance from the wall even if I measured it by a ruler.
- However, after a few nodes, the optimized algorithm follows the path well.
- For the end time, the optimized graph has a bit lower error than the non-optimized graph has. Nonetheless, the comprehensive analysis should be conducted to evaluate the entire performance.

2.2 Result

- Results are as shown in and Fig. 2 and Fig. 3.
- After changing parameters such as the inverse of covariance matrix, we will be able to get different results.

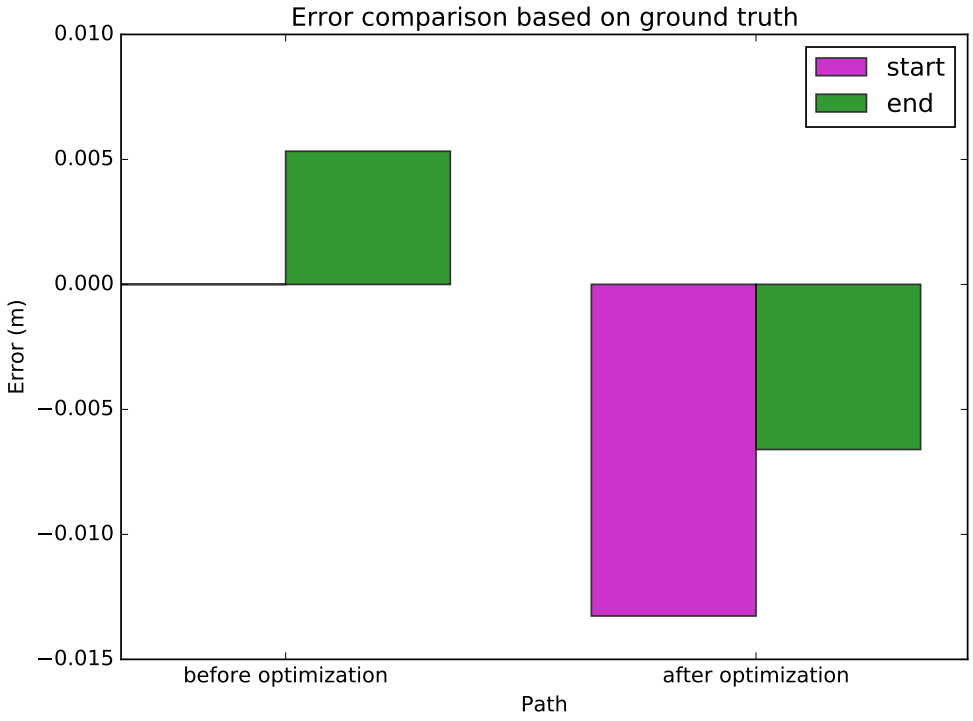


Figure 3: Result of error comparison between before optimization and after optimization with respect to ground truth

References

[1] JEONG, M. Kalman filter. https://github.com/MingiJeong/kalman_filter_mg_cs169. Accessed on 2020-02-07.

[2] RAINER KUEMMERLE, GIORGIO GRISETTI, H. S. K. K. W. B. g2opy. https://github.com/uoip/g2opy. Accessed on 2020-02-07.