Applied estimation Graph SLAM

Zhengming Zhu

December 31, 2023

1 Part1 - The SLAM problem

1.1 SLAM modeling

1.How would you model a GPS measurement (in a simple but unrealistic way)? How many of the state vectors does it depend on? (hint: look at eq. (4) to see an example of how models might look)
Since the GPS can provide basic information including Latitude, Longitude, and Altitude. To be a simple but unrealistic way, we can model it as:

$$z_k = Hx_t + \eta$$

Where H is observation matrix and η is noise. So, we assume a 2D model, the state can be represented as $x_t = [p_x, p_y, \theta]$, the H relies on position but not orientation. Therefore:

$$H = \begin{matrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{matrix}$$

2.How would you model a compass measurement? The compass mainly provide orientation information of robot, so:

$$z_k = Hx'_t + \nu$$

$$\text{where} H = \begin{matrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{matrix}$$

$$x_t = [p_x, p_y, \theta]$$

3.How would you model a gyroscope measurement? A 'gyro' gives a reading of angular velocity or the rate of change of an angle. Hint: One way is to extend the robot state vector to accommodate direct measurements of a part of the state with the gyro? How would that change the g above? How would you still also use the information from the wheel rotations? **Extend the state vector:** Because gyroscope mainly measure the angular velocity of robot, we assume: $x_t = [p_x, p_y, \theta, v_x, v_y, \omega]$, then:

$$z_k = Hx'_t + \nu$$

$$\text{where} H = \begin{matrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{matrix}$$

$$x_t = [p_x, p_y, \theta, v_x, v_y, \omega]$$

Not extend the state vector: we can get the angle change with the time step, so:

$$h(x) = x_\theta + \omega \times \Delta t$$

$$z_k = h(x) + \eta$$

1.2 SLAM as a Probabilistic Graphical Model

4.We assume ϵ in our planar motion model is given by a normal (Gaussian) distribution with zero mean and diagonal covariance matrix R. Using the model of g in eq.(2), what explicitly is $p(x_i|x_{i-1}, u_i)$? So, when the control vector input:

$$g(x_{i-1}, u_t) = (x_{i-1} + (\Delta s)\cos(\theta_{i-1}, y_{i-1} + (\Delta s)\sin(\theta_{i-1}), \theta_{i-1} + \Delta\theta)^T$$

$$p(x_i|x_{i-1}, u_i) = \frac{1}{\sqrt{2\pi R}} \exp(-\frac{1}{2}(x-g)R^{-1}(x-g)^T)$$

5.If the landmark measurements give the distance to the landmark with a variance of $1.0m^2$, what is the explicit model of $p(z_k|x_{p(k)}, m_{f(k)})$? Because the measurement is the distance, so:

$$\overline{z_k} = \sqrt{(x_{p(k)} - m_x)^2 + (y_{p(k)} - m_y)^2}\sigma \sim N(0,1)$$

$$p(x_i|x_{i-1}, u_i) = \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{1}{2}(\frac{z_k - \overline{z_k}}{\sigma})^2)$$

6.Draw a figure of the SLAM problem as a Bayesian belief network, factor graph, and Markov random field with at least four robot poses and three landmarks.

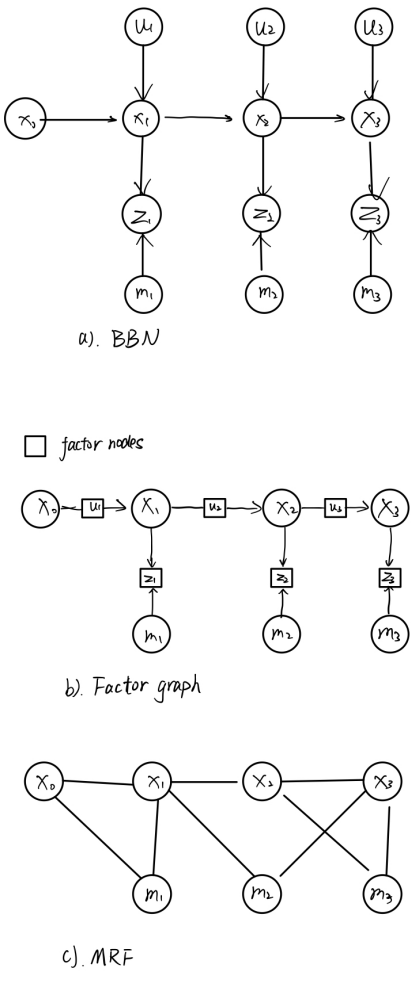

Figure 1: SLAM representation

7.What assumptions are made in Equation (3)? 1.The prior distribution of every state is independent of each other. 2.The measurement model is independent. 3.The motion model is independent.

1.3 GraphSLAM Graph construction

8.What happens to the structure of Ω when the robot sees a previously seen landmark after a long time (aka loop closure)? Hint, take a look at Figure (1). Can (10) be solved in linear time then?
If robot has seen a previously seen landmarks after some time has passed. In that case, features in the environment will be linked to multiple states, which is far away the diagonal. As mentioned above, when there is no loop closure, the variables in Ω can be reordered to make it band-diagonal and thus it becomes simple to solve in linear time way. But now, the matrix is becoming more complicated.

9.Show that linearizing the motion equation for one pose to pose factor gives rise to the following terms to be added to Ω and ξ at the appropriate places: What are G and $\overline{\mu}_{i-1}$? Hint: Consider adding a new term to the log likelihood and then compare linear and quadratic terms with the form of Eq. (8).
To linearizing g by Taylor Series approximation , we can write it as :

$$g(u_i, x_{i-1}) = g(u_i, \overline{\mu}_{i-1}) + G(x_{i-1} - \overline{\mu}_{i-1})$$

The G is the Jacobian of the motion model and the $\overline{\mu}_{i-1}$ is the estimation of the state at x_{i-1}.
As we have learned that the log MLE representation as follows:

$$\ln p(x_i|x_{i-1}, u_i) = const + \frac{1}{2}(x_i - g(u_i, x_{i-1}))^T R^{-1}(x_i - g(u_i, x_{i-1}))$$

Put the linearized equation into it:

$$\ln p(x_i|x_{i-1}, u_i) = const + \frac{1}{2}x_i^T R^{-1} x_i + \frac{1}{2}x_{i-1}^T G^T R^{-1} G x_{i-1} - x_i^T R^{-1} G x_{i-1}$$
$$-x_{i-1}^T G^T R^{-1}(G\overline{\mu}_{i-1} - g(u_i, \overline{\mu}_{i-1})) + x_i^T R^{-1}(G\overline{\mu}_{i-1} - g(u_i, \overline{\mu}_{i-1}))$$

So, we can get:

$$\Delta\Omega_{i-1,i-1} = G^T R^{-1} G$$
$$\Delta\Omega_{i,i} = R^{-1}$$
$$\Delta\Omega_{i,i-1} = R^{-1} G$$
$$\Delta\xi_{i-1} = G^T R^{-1}(G\overline{\mu}_{i-1} - g(u_i, \overline{\mu}_{i-1}))$$
$$\Delta\xi_i = R^{-1}(G\overline{\mu}_{i-1} - g(u_i, \overline{\mu}_{i-1}))$$

1.4 Solve the system

10.How can we set a convergence condition to stop the algorithm?
There are some ways to set the convergence condition:
Delta Threshold: Set a threshold during iteration, and check the difference between current estimated state value and last state value, if it is less than the threshold, we can assume that it converges.
Residual Threshold: We can check the objective function that we optimize, whether it is less than a threshold. If so, we can assume it converge.

11.What are the factors that might influence the number of iterations required by the algorithm?

1. Initial value
2. The uncertainty of motion model and measurement model
3. The complexity of the environment

12.When might this implementation of a GraphSLAM solution fail to converge?

1. Bad initial state
2. Too large uncertainty of motion model and measurement model
3. Too complex environment
4. Too poor collected information and processing

2 Part2

2.1 Characterize each of the five data sets:

2.1.1 Dataset1

- Small number of landmarks.
- No local minima or weak minima, the result can converge to the ground truth.
- The density of the map and the sensor range do not effect the convergence much.
- The noise of motion model and measurement model do not effect the convergence much even though with bit uncertainty like $R = diag(0.03, 0.03, 3) \quad Q = diag(0.03, 3, 0.003)$.
- We set the initial state as $[5, 5, 2]$, the start point do not change the convergence much.

2.1.2 Dataset2

- The map has more landmarks and it is not symmetrical.
- No local minima or weak minima according to the global error from the simulation result.
- The denser map make the algorithm need more iteration to converge.
- The noise of motion model and measurement model do not effect the convergence much even though with bit uncertainty like $R = diag(0.03, 0.03, 3) \quad Q = diag(0.03, 3, 0.003)$
- The start point does not effect the convergence much, it might increase the iteration number if setting a far start point, we set the initial state as $[5, 5, 3]$, the algorithm also converges after 13 iteration.

2.1.3 Dataset3

- The map has 5 landmarks, but the trajectory is complex.
- Yes, there is weak minima during the process.
- The density of the map is not effect the convergence much.
- Yes, the noise also effect the convergence much.
- The start point does not effect the convergence much according to what are mentioned above.

2.1.4 Dataset4

- The map has 10 landmarks.
- Yes, there is local minima during the process.
- The density of the map is not effect the convergence much.
- This dataset has so large noise and effect the convergence much.
- The start point does not effect the convergence much according to what are mentioned above.

2.1.5 Dataset5

- The map has 40 landmarks.
- Yes, there is local minima during the process.
- The density of the map is not effect the convergence much.
- This dataset has so large noise and effect the convergence much.
- The start point does not effect the convergence much according to what are mentioned above.

When is the linearize system a reasonable approximation or not?
When the full estimation of the robot state and map feature is close to the ground truth, which means the global error is less than the threshold. Therefore, it will be a reasonable approximation.

What happens when the robot closes a loop returning to see a landmark from the beginning?
The robot close the loop and see the landmark form the beginning will add more constrains to the information matrix, which reduces the uncertainty of the estimation.

2.2 Save images of the best final map for each data set and include with your report.

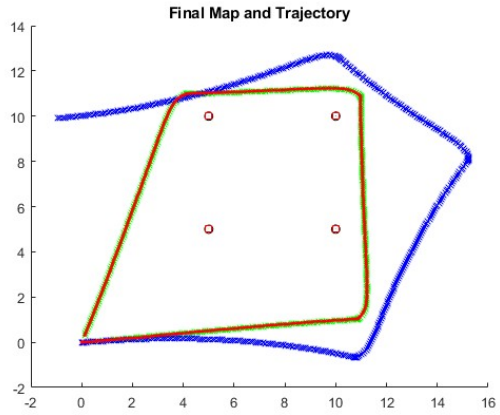

Figure 2: Best final map of dataset1

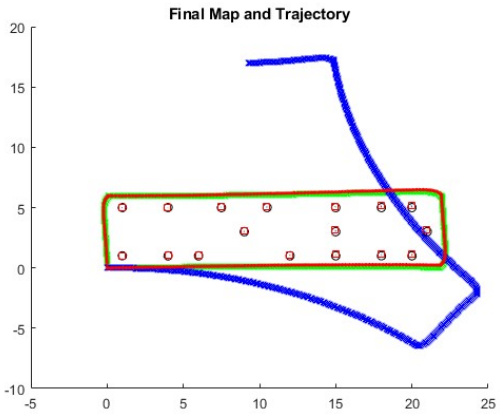

Figure 3: Best final map of dataset1

2.3 Suggest some ways that the method could be improved:

Deal with outliers and spurious measurements: We can set a threshold to check the difference between previous data value and current data value. If the difference is larger than the threshold, we can assume it is a outlier or spurious.
Avoid oscillation and overshooting:We can set a maximum limitation for the update step when optimizing. In that way, the algorithm can converge in a smooth and stable way.
Data association:We can use Maximum Likelihood Estimation to help associate the feature and the measurement.

2.4 By changing the values of Q and R you should be able to get a third data set to work:

The algorithm also follows the Bayes rule, so changing the values of Q and R means that how we assign the weight of the motion model and the measurement model. After several times modification, I rely more on the motion model with respect to the position, and rely more on the measurement model with respect to the angle, so the Q and R is as follows:

$$R = \begin{matrix} 0.0005^2 & 0 & 0 \\ 0 & 0.0005^2 & 0 \\ 0 & 0 & 1 \end{matrix}$$

$$Q = \begin{matrix} 0.01^2 & 0 & 0 \\ 0 & 0.5^2 & 0 \\ 0 & 0 & 0.001 \end{matrix}$$

I get the convergence result at approximately 60 times update, the mean absolute error=(0.090806, 0.080914, 0.025020) and mean error(x, y, theta)=(-0.071937, 0.073989, 0.012912).

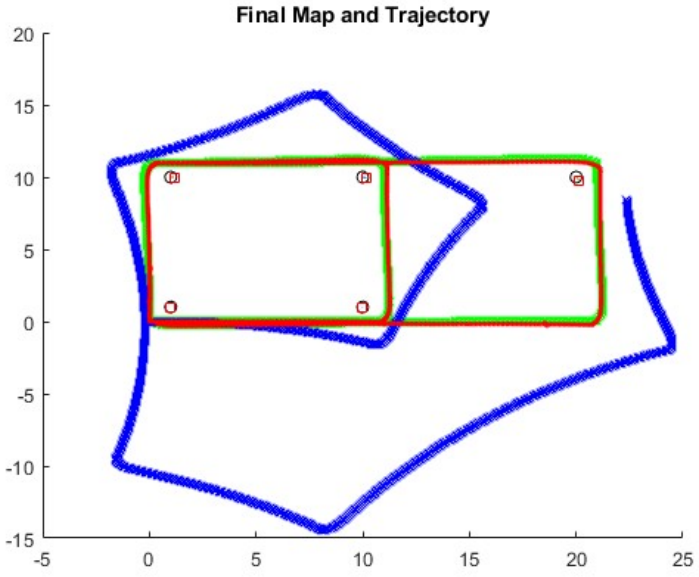

Figure 4: Best final map of dataset3

2.5 By splitting the graph up into smaller parts and and solving the first part then adding the second using the found solution for the first part as a starting approximation you can solve an additional data set.

In the practical code, the algorithm can divide the graph optimization problem into 8 parts, and it solve the first part and the second part use the solved part as current estimated state, which really can help speed up and smooth the convergence.

2.6 Implement some (non-trivial) way of improving the performance and show the improvement

To improve the performance on the more complex and noisy dataset, I implement outlier detection and process. As a result, the algorithm get significant improvement performance with respect to dataset4. The more details as follows:

Outlier detection on state estimation:We focus on the solved robot pose every iteration directly, because in the every iteration, the input state would be the previous estimation. So, we skip the might wrong measurement or motion during the optimization, instead, we find the outlier estimation after every graph solving. If the difference between the current estimation and the previous estimation is larger than a threshold, we can assume it is a outlier.

Dynamic threshold:In practice, we do not know how to set an appropriate threshold. Therefore, we applied the dynamic threshold with sliding window, to be more specific, the model records recent estimation errors during optimization:

$$\Theta_{dynamic} = \frac{1}{windowSize} \sum_{i=1}^{windowSize} (\hat{x_i} - x_i) + \lambda \sqrt{\frac{1}{windowSize} \sum_{i=1}^{windowSize} (\hat{x_i} - x_i)^2}$$

In this case, we set a dynamic threshold after referencing the mean and standard deviation with a flexible parameter.

Process outliers:Since the environment of the last two datasets are circle, which has smooth motion change. So, we skip the inappropriate estimation by keep the last estimation and input it to next iteration.

Dataset4

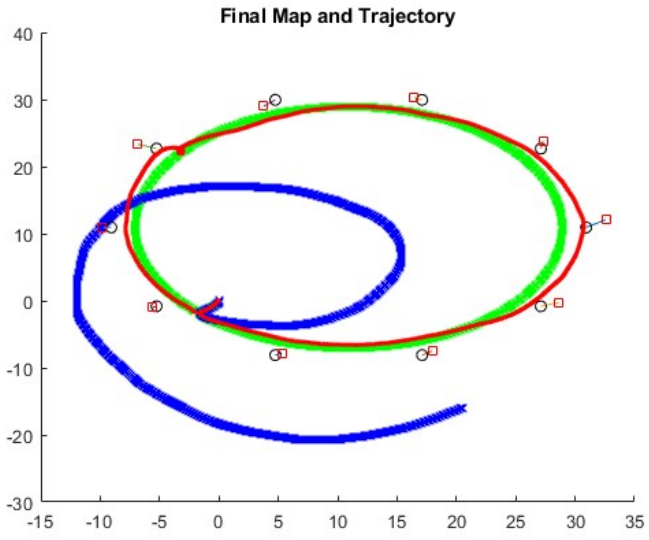

Figure 5: Best final map of dataset4

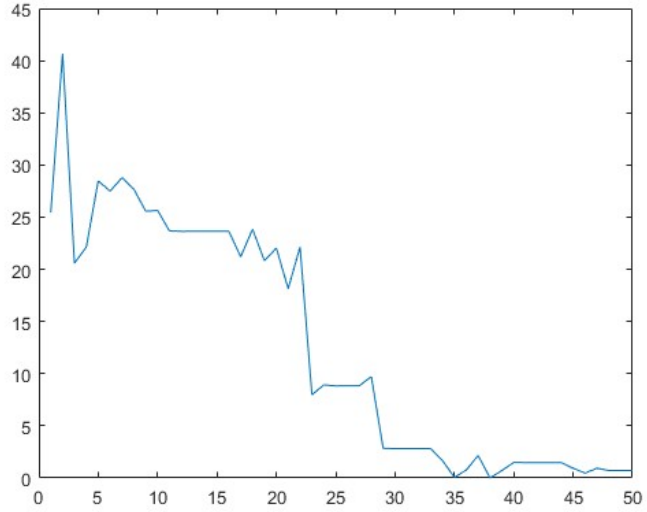

Figure 6: Global error of dataset4

As the figure shows, we get a good estimation after 35 or more iteration.

Moreover, this method also helps find the minima when optimizing dataset5. Since there is no odometry information, the algorithm will stick into the minima, because so the initial estimate can not be made beyond the features seen at the start. As we can conclude from the figures, the global error goes stable after 100 or more times iteration, which means the algorithm gets into the local optimal solution.

Dataset5

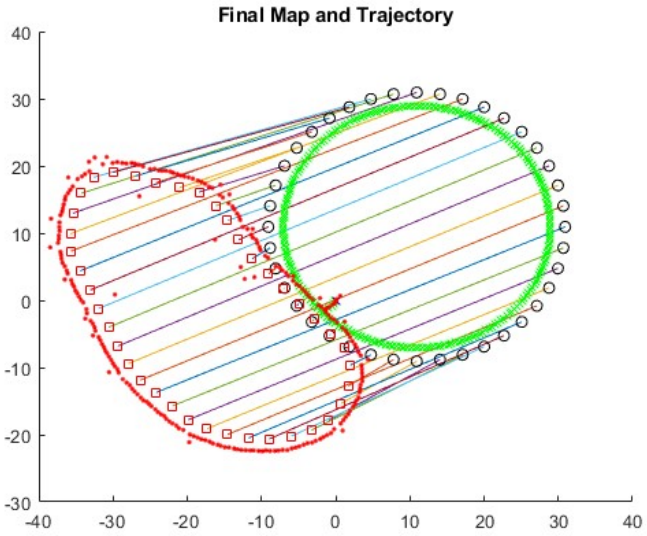

Figure 7: Best final map of dataset5

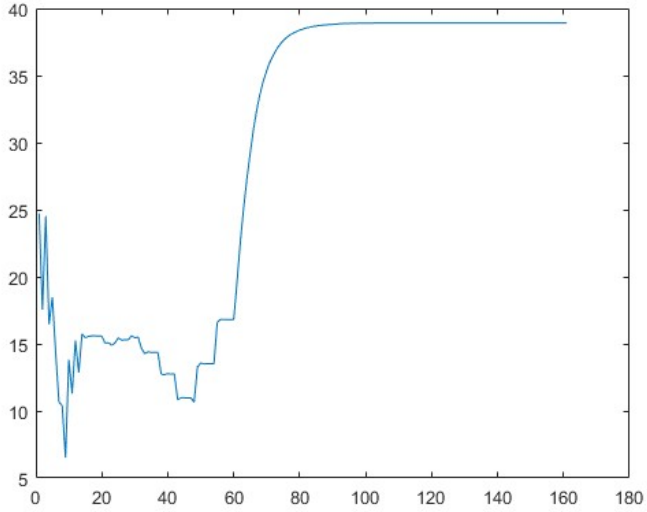

Figure 8: Global error of dataset5