TP3: Simultaneous Localization and Mapping using Extended Kalman Filter

Navigation pour les systèmes autonomes

ROB312

Yu WANG

October 15, 2021

1 Introduction

In this practical work, we will study a Simultaneous Localization and Mapping (SLAM) method that builds a map of an unknown environment using an Extended Kalman Filter (EKF).

1.1 Mapping with backtracking

The goal of mapping with backtracking is propagating new information on current pose to the previous poses. In order to achieve this goal:

- memorize relations between robot poses and map elements to be able to correct them
- Possible with scan maps or landmarks maps

The general framework is shown as (1.1) (Bayesian filtering on robot pose and map elements Possible with scan maps or landmarks maps).

$$Bel(x_t, c_t) = \eta p(y_t|x_t, c_t) \iint p(x_t|x_{t-1}, u_{t-1}))Bel(x_{t-1}, c_{t-1})dx_{t-1}dc_{t-1} \tag{1.1}$$

Generally, there are many ways to approximate and estimate Bel (x, c), we will study two in depth: the extended Kalman filter (this TP) and the particulate filter (the next TP).

1.2 Principle of EKF SLAM

For each landmark that is sensed, if it is already in the map, the Kalman filter will be used to estimate its pose and the robot's pose. The formula is shown as (1.2).

$$\begin{aligned} x_t^* &= f(X_{t-1}, u_t) \\ P_t^* &= A.\hat{P}_{t-1}.A^T + B.Q.B^T \\ Y_t^* &= h(x_t^*) \\ K &= P_t^* H^T.(H.P_t^*.H^T + P_Y)^{-1} \\ \hat{X}_t &= x_t^* + K(Y_t - Y_t^*) \\ \hat{P}_t &= P_t^* - KHP_t^* \end{aligned} \tag{1.2}$$

If it is not in the map, it will be added to the state vector. If necessary, X with absolute iconic pose and P with correct co-variance will be expanded by calculating J_r and J_y.

1.3 Introduction of TP

The provided code simulates the robot moving along a given trajectory in an environment composed of punctual landmarks. A simple extended Kalman filtering method using the direction and distance perception of these landmarks is realized, and the result is shown in Figure 1.1.

In the left figure, the red curve is the estimated position of EKF, the green curve is the odometer trajectory, and the black curve is the real trajectory. In the figure on the right, the blue line is the error between the estimated position and the true position, and the red line is the co-variance.

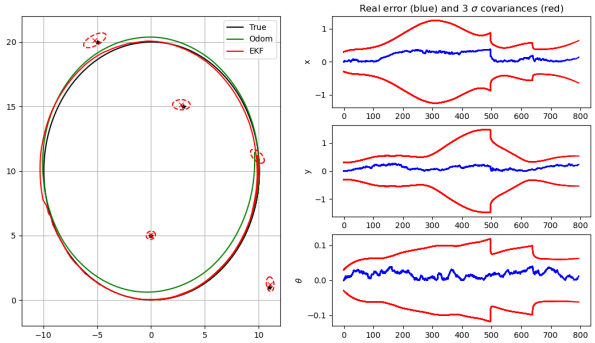

Figure 1: Result of the original condition

2 Influence of the environment

For this question, use the default parameters of the provided code. By default, the data association is assumed to be known, i.e. for each perceived landmark, the corresponding landmark in the map is identified without ambiguities. In particular, this makes it possible to properly manage loop closures, even when the error in the map is very severe.

In this section, we will firstly analyze the influence of the environment on the performance of the algorithm. We will analyze the number of positions of landmarks and the robot trajectory in question 1 and the Mahalanobis distance is used in question 2.

2.1 Question 1 : Modify the number and position of landmarks and the robot trajectory

We use different numbers and positions of landmarks and robot trajectories in the following three cases. Among them, the parameter *yaw_rate* is related to the length of the loop, and *Landmarks* is the definition of landmarks. It should be noted that landmarks that are too close may cause some problems, so landmarks should be selected reasonably.

2.1.1 Case 1 : a short loop and a dense map with many landmarks inside the robot perception radius

In this case, we choose a larger *yaw_rate* (0.2) to obtain a shorter cycle and we create more landmarks (10) to increase the density of the map.

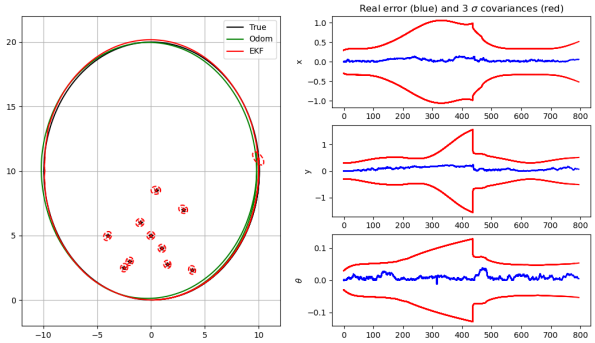

Figure 2: Result of the case 1

Figure 2 shows a high-quality map. The black curve (true trajectory) and the red curve (estimated curve) overlap very well, and the location of each landmark

is well located. However, it can be found that landmarks close to the starting position are better positioned than landmarks far from the starting position, because the corresponding uncertainty is smaller. The figure on the right also shows this, where errors accumulate during the cycle and are corrected at the starting point, therefore uncertainty.

In short loops and dense maps, the robot can keep many landmarks within its perception radius during movement. Through data association and matrix update, a good map can be obtained.

2.1.2 Case 2 : a long loop and a dense map with many landmarks all along the loop

In this case, set the value of the *yaw_rate* to 0.1 to get a longer loop, and add landmarks along the loop.

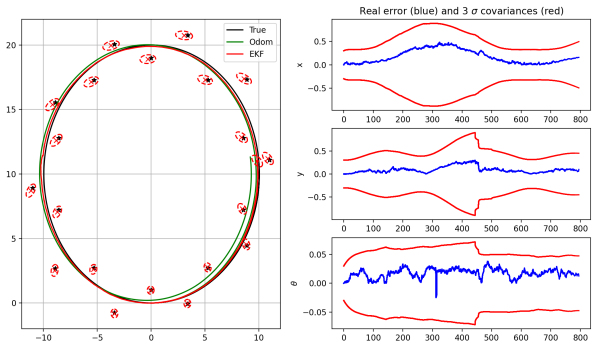

Figure 3: Result of the case 2

We can see from the Figure 3 that the quality of the map is also very good, and the red curve (estimated position) and black curve (real position) overlap very well. The location of each landmark is also well positioned. It can still be found that landmarks near the starting position are better positioned than landmarks far from the starting position. In the figure on the right, errors and uncertainties also accumulate during the cycle and are corrected at the starting point.

Large-scale perception (10) allows the robot to retain many landmarks within its perception radius during movement, which ultimately provides us with a good map.

2.1.3 Case 3 : a long loop and a sparse map with only few landmarks near the start position

In this case, there are only a few landmarks near the starting location. In Figure 4, we can see that although the landmarks in the map have been located, because they are close to the starting position, the three curves no longer overlap. The larger error and larger uncertainty in the figure on the right also indicate this. The lack of landmarks makes it impossible for the robot to correct and update its predictions, which leads to this phenomenon. Until the starting position is approached again, the estimated curve can be corrected again.

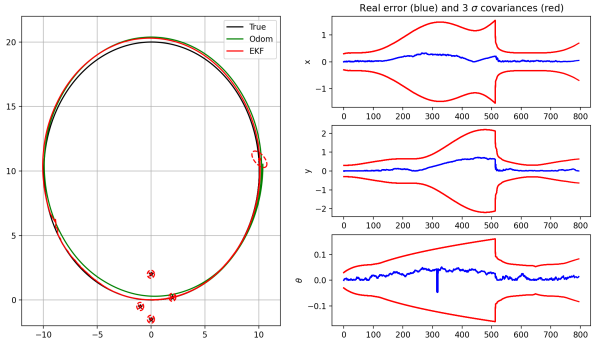

Figure 4: Result of the case 3

2.2 Question 2 : Answer the same question when the data association is performed using the Mahalanobis distance

In this section, we study the case when the data association is performed using the Mahalanobis distance ($KNOWN\_DATA\_ASSOCIATION = 0$). The parameter M_DIST_TH should be modified according to the environment.

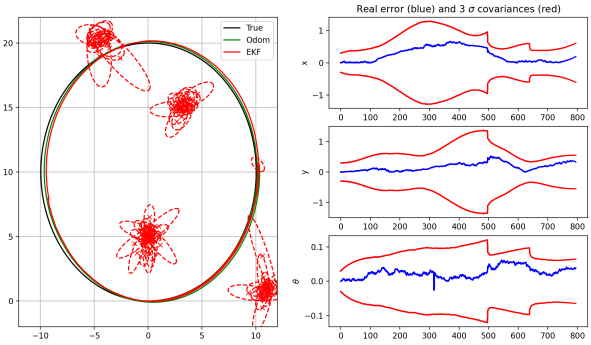

Figure 5: Result with $M\_DIST\_TH = 1$

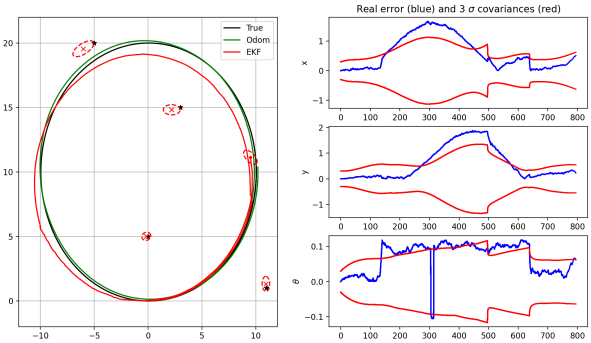

Figure 6: Result with $M\_DIST\_TH = 100$

M_DIST_TH is the Mahalanobis distance threshold for data association. It can be seen from Figure 5 that when the threshold is too small, more points than reality will be misidentified as landmarks. However, when the threshold is too large, some landmarks will be ignored, and there will be a large error between the real position and the estimated position, as shown in Figure 6. Therefore, an appropriate threshold should be selected to ensure the normal function of the algorithm. Here we set the threshold to 9.

2.2.1 Case 1 : a short loop and a dense map with many landmarks inside the robot perception radius

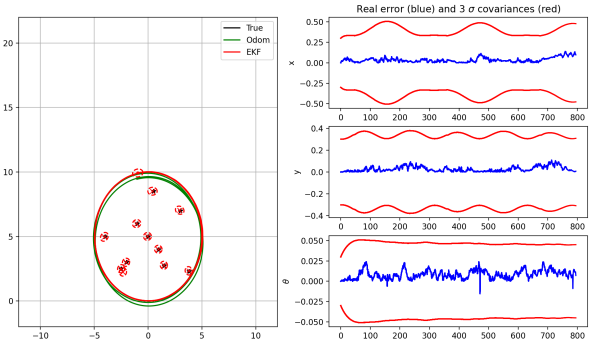

Figure 7: Result of the case 1

2.2.2 Case 2 : a long loop and a dense map with many landmarks all along the loop

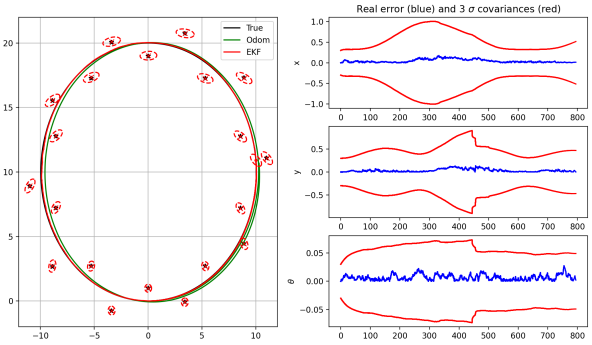

Figure 8: Result of the case 2

2.2.3 Case 3 : a long loop and a sparse map with only few landmarks near the start position

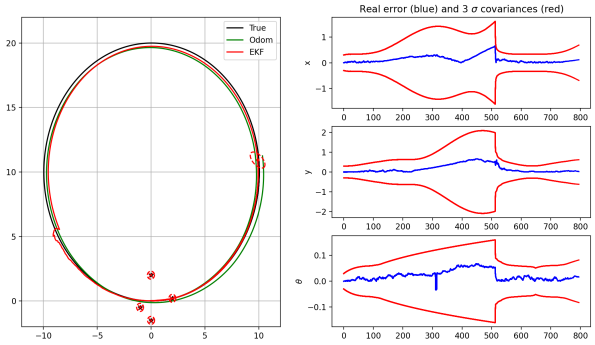

Figure 9: Result of the case 3

When a suitable threshold is chosen, the figures obtained are very similar to those obtained in question 1. When there are enough landmarks in the sensing area of the robot, the prediction can be corrected and updated to obtain a good map, as shown in Figure 7 and 8.

However, when there are not enough landmarks, the black curve (true position) and the red curve (estimated position) no longer coincide, which is also shown by the large error and uncertainty in the right panel of Figure 9.

3 Probabilistic models

In this section, we keep the configuration with unknown data association ($KNOWN\_DATA\_ASSOCIATION = 0$), and the environment with large loops and sparse mapping. According to question 2, the Mahalanobis distance threshold for data association is set to 9.

3.1 Question 3 : Change the estimated noise values

3.1.1 Case 1 : Q = Q_sim/10, Py = Py_sim/10

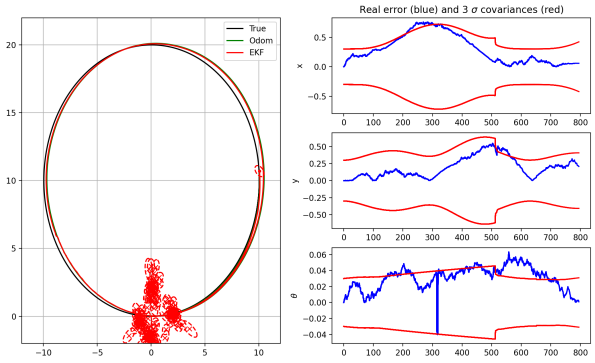

Figure 10: Result of the case 1

It can be seen from Figure 10 that when Q and Py are smaller than Q_Sim and Py_Sim, the location of the landmark will be misidentified. The black curve (true position) and the red curve (estimated position) do not overlap, and the error between the real position and the estimated position is very large.

3.1.2 Case 2 : Q = Q_sim, Py = Py_sim

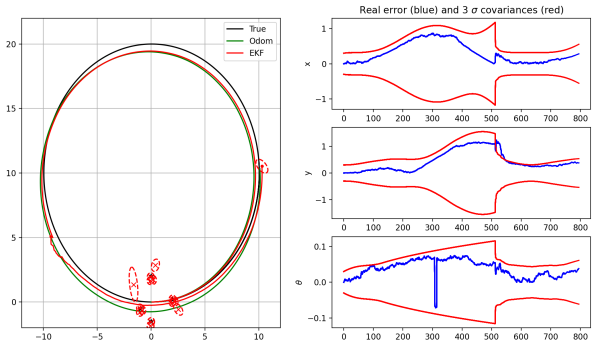

Figure 11: Result of the case 2

It can be seen from Figure 11, when Q and Py are equal to Q_Sim and Py_Sim, we can find that the location of the landmark is better recognized than before, but it is still not very good. In this case, the error is small, but

the uncertainty increases. The black curve (true position) and the red curve (estimated position) overlap better.

3.1.3 Case 3 : Q = 10Q_sim, Py = 10Py_sim

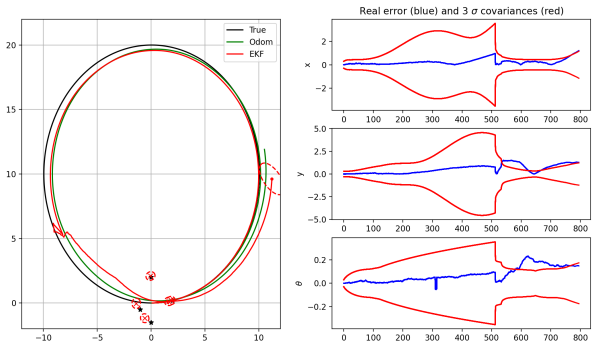

Figure 12: Result of the case 3

It can be seen from Figure 12, when Q and Py are much larger than Q_Sim and Py_Sim, we find that the location of the landmark is well recognized and the black curve (true position) and the red curve (estimated position) overlap better. In this case, the error is also small, but the uncertainty is indeed very large.

3.1.4 Best configuration : Q = 2Q_sim, Py = 2Py_sim

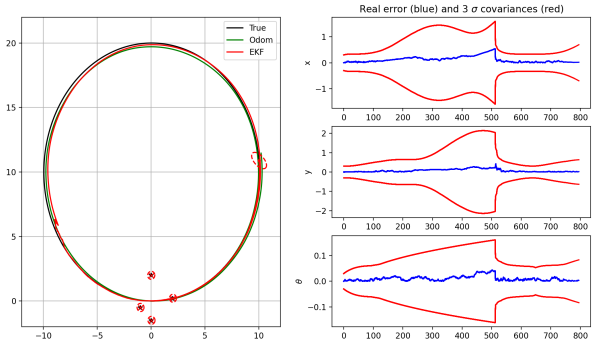

Figure 13: Best configuration

From Figure 10, Figure 11 and Figure 12, we can find that compared with

Q_Sim and Py_Sim, the larger Q_Sim and Py_Sim, the better the recognition of landmarks, and the smaller the error between the real position and the estimated position. However, uncertainty has increased. So the best configuration is that Q and Py are a bit larger than Q_Sim and Py_Sim. Just like the situation in Figure 13, the landmark positioning is relatively good, the corresponding uncertainty is relatively small, and the error is not too large.

4 Undelayed initialization

In this section, we will only use the orientation of the landmarks instead of their distance to perform SLAM. An undelayed initialization method will be used to initialize several landmarks in the cone corresponding to the perception direction, and then remove the useless landmarks. This method helps us solve the problem that it is difficult to estimate their location from a single perception.

4.1 Implementation Sola et al. (2005)

We look for a safe way to fill the conic-shaped ray with the minimum number of Gaussian-shaped distributions. For that, we define $p(s)$ as a geometric series with $\alpha_j = \alpha =$ constant as (4.1). An overview of the series with its parameters is shown in Figure 14.

$$p(s) = \sum_{j=1}^{N_g} c_i \Gamma(s - \beta^{j-1} s_1, (\beta^{j-1}\sigma_1)^2) \tag{4.1}$$

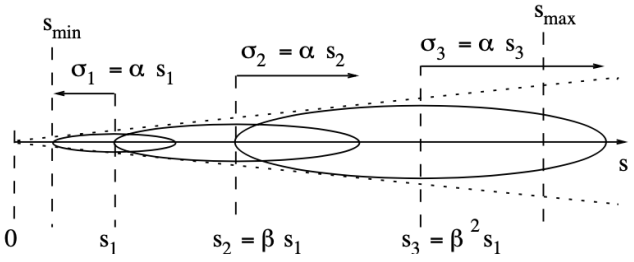

Figure 14: The conic Ray: a geometric series of Gaussian distributions

From the bounds $[s_{min}, s_{max}]$, and the choice of the ratio α and the geometric base β, we need to determine the first term (s_1, σ_1) and the number of terms N_g. We impose the conditions $s_1 - \sigma_1 = s_{min}$ and $s_{N_g} + \sigma_{N_g} \geq s_{max}$ to get (4.2).

$$
\begin{aligned}
s_1 &= (1-\alpha)^{-1} s_{min} \\
\sigma_1 &= \alpha s_1 \\
N_g &= 1 + ceil\left[log_\beta\left(\frac{1-\alpha}{1+\alpha}\frac{s_{max}}{s_{min}}\right)\right]
\end{aligned}
\qquad s \tag{4.2}
$$

where $ceil(x)$ is the next integer to x and the geometric base β determines the sparseness of the series.

Then we want to choose the Gaussian in the ray that best represents the real landmark, while using at the same time the angular information this ray provides. It consists of three main operations: the inclusion of all the members of the ray into the map; the subsequent updates using Federated Information Sharing; and the successive pruning of bad members. We update the landmark hypothesis by stacking iteratively one by one according to (4.3) and (4.4), and the related index has also changed.

$$
\hat{\mathbf{X}}^+ = \begin{bmatrix} \hat{\mathbf{X}} \\ \hat{\mathbf{x}}_p^1 \\ \vdots \\ \hat{\mathbf{x}}_p^{N_g} \end{bmatrix} \tag{4.3}
$$

$$
\mathbf{P}^+ = \begin{bmatrix} \mathbf{P} & \mathbf{P}_{pX}^{1\,T} & \cdots & \mathbf{P}_{pX}^{N_g\,T} \\ \mathbf{P}_{pX}^1 & \mathbf{P}_{pp}^1 & & \\ \vdots & & \ddots & \\ \mathbf{P}_{pX}^{N_g} & & & \mathbf{P}_{pp}^{N_g} \end{bmatrix} \tag{4.4}
$$

4.2 Result

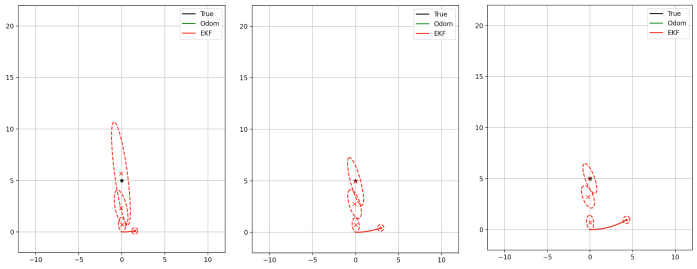

Figure 15: Result of the undelayed initialization

As shown in Figure 15, we have completed the test of the algorithm of bearing

only SLAM. The appearance and update process of the detected landmarks are clearly displayed.

References

Sola, J., Monin, A., Devy, M., and Lemaire, T. (2005). Undelayed initialization in bearing only slam. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2499–2504.