Self Driving Cars 2023-Fall
Homework 3
312605001 機器人碩士 歐庭維

1. Code explain

<ANS>

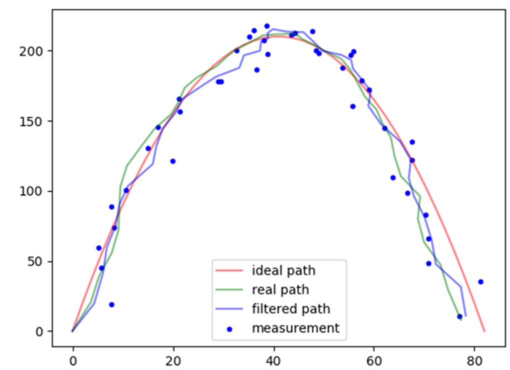
```python
1  """
2  @author: OU,TING-WEI @ M.S. in Robotics
3  date : 10-24-2023
4  Self-Driving-Cars HW3 ( NYCU FALL-2023 )
5  """
6  import numpy as np
7  
8  class KalmanFilter:
9      def __init__(self, x=0, y=0, yaw=0):
10         # State [x, y, yaw]
11         self.state = np.array([x, y, yaw])
12 
13         # Transition matrix
14         self.A = np.identity(3)  # Identity matrix for state transition
15         self.B = np.identity(3)  # Identity matrix for control input
16 
17         # State covariance matrix
18         self.S = np.identity(3) * 1  # Initialize state covariance matrix
19 
20         # Observation matrix
21         self.C = np.array([[1, 0, 0],
22                            [0, 1, 0]])  # Maps the state to the measurement space
23 
24         # State transition error
25         self.R = np.identity(3)  # Process noise covariance matrix
26 
27         # Measurement error
28         self.Q = np.identity(2) * 3  # Measurement noise covariance matrix
29 
30     def predict(self, u):
31         # Prediction step
32         self.state = self.A @ self.state + self.B @ u
33         self.S = self.A @ self.S @ self.A.T + self.R  # Update state covariance
34 
35     def update(self, z):
36         # Update step
37         K = self.S @ self.C.T @ np.linalg.inv(self.C @ self.S @ self.C.T + self.Q)
38         self.state = self.state + K @ (z - self.C @ self.state)
39         self.S = (np.identity(3) - K @ self.C) @ self.S  # Update state covariance
40         return self.state, self.S  # Return the updated state and covariance
```

1: **Algorithm Kalman_filter**$(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)$**:**
2: $\bar{\mu}_t = A_t\, \mu_{t-1} + B_t\, u_t$
3: $\bar{\Sigma}_t = A_t\, \Sigma_{t-1}\, A_t^T + R_t$
4: $K_t = \bar{\Sigma}_t\, C_t^T (C_t\, \bar{\Sigma}_t\, C_t^T + Q_t)^{-1}$
5: $\mu_t = \bar{\mu}_t + K_t(z_t - C_t\, \bar{\mu}_t)$
6: $\Sigma_t = (I - K_t\, C_t)\, \bar{\Sigma}_t$
7: return μ_t, Σ_t

Step 2 & 3 → Line 30 ~ 33
Step 4 ~ 7 → Line 35 ~ 40

2. Filtered path result

<ANS>



3. How you design the observation matrix (C)?

<ANS>

The function of the Observation matrix is to map the system state space into the observation space, allowing us to compare the estimation and the measurement.
The robot state is [x, y, yaw] and the Measurement (z) is Position of [x, y], so we can design the observation matrix

$$C = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Use this matrix, we can map the robot state [x, y, yew] to measurement [x, y].

4. How you design the covariance matrices(Q, R)?

<ANS>
Given:

Control term (u): displacement of the robot and change in yaw [delta_x, delta_y, delta_yaw] (with Gaussian noise added to delta_x, delta_y, and delta_yaw, having a mean of 0 and a variance of 1, respectively).

Measurement (z): Position [x, y] (with Gaussian noise added to x and y, having a mean of 0 and a variance of 3, respectively).

The covariance matrix is as follows:

$$\mathbf{cov}(X,Y) = \begin{bmatrix} \mathrm{cov}(x_1,y_1) & \mathrm{cov}(x_1,y_2) & \cdots & \mathrm{cov}(x_1,y_n) \\ \mathrm{cov}(x_2,y_1) & \mathrm{cov}(x_2,y_2) & \cdots & \mathrm{cov}(x_2,y_n) \\ \vdots & \vdots & \ddots & \vdots \\ \mathrm{cov}(x_m,y_1) & \mathrm{cov}(x_m,y_2) & \cdots & \mathrm{cov}(x_m,y_n) \end{bmatrix}$$

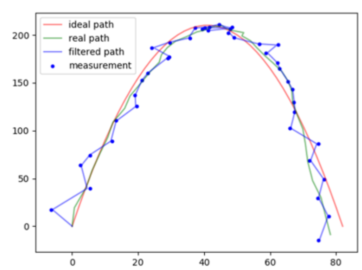
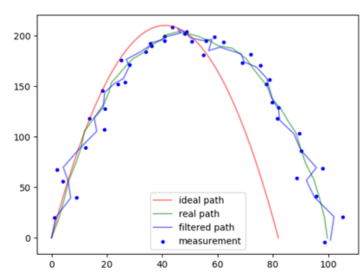
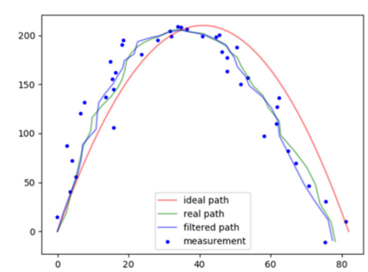
When X and Y are independent, the covariance matrix becomes the identity matrix. In the case of the robot state where x, y, and yaw are independent of each other, we can define R and Q as follows:

→ We can choose $R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

→ We can choose $Q = \begin{bmatrix} 3 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 3 \end{bmatrix}$

5. How will the value Q and R arrect the output of Kalman Filter?

<ANS>
Q (Measurement Noise Covariance) is represents the uncertainty in the measurements. It reflects the accuracy of the sensor or measurement. If Q is set to a higher value, the filter will trust the predicted state more and be less influenced by the measurement state.

Q = 0.1	Q = 1	Q = 10
		

R (Process Noise Covariance) is represents the uncertainty in the process model. It accounts for how the state is expected to change over time due to the underlying dynamics of the system. If R is set to a higher value, it implies that we trust the measurement value more, and the filter will be less responsive to changes in the process model.

R = 0.1	R = 1	R = 10