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

1. Code explain

<ANS>

```python
"""
@author: OU,TING-WEI @ M.S. in Robotics
date : 10-24-2023
Self-Driving-Cars HW3 ( NYCU FALL-2023 )
"""
import numpy as np

class KalmanFilter:
    def __init__(self, x=0, y=0, yaw=0):
        # State [x, y, yaw]
        self.state = np.array([x, y, yaw])

        # Transition matrix
        self.A = np.identity(3)  # Identity matrix for state transition
        self.B = np.identity(3)  # Identity matrix for control input

        # State covariance matrix
        self.S = np.identity(3) * 1  # Initialize state covariance matrix

        # Observation matrix
        self.C = np.array([[1, 0, 0],
                           [0, 1, 0]])  # Maps the state to the measurement space

        # State transition error
        self.R = np.identity(3)  # Process noise covariance matrix

        # Measurement error
        self.Q = np.identity(2) * 3  # Measurement noise covariance matrix

    def predict(self, u):
        # Prediction step
        self.state = self.A @ self.state + self.B @ u
        self.S = self.A @ self.S @ self.A.T + self.R  # Update state covariance

    def update(self, z):
        # Update step
        K = self.S @ self.C.T @ np.linalg.inv(self.C @ self.S @ self.C.T + self.Q)
        self.state = self.state + K @ (z - self.C @ self.state)
        self.S = (np.identity(3) - K @ self.C) @ self.S  # Update state covariance
        return self.state, self.S  # Return the updated state and covariance
```

Algorithm Kalman_filter($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):

$\bar{\mu}_t = A_t\ \mu_{t-1} + B_t\ u_t$

$\bar{\Sigma}_t = A_t\ \Sigma_{t-1}\ A_t^T + R_t$

$K_t = \bar{\Sigma}_t\ C_t^T (C_t\ \bar{\Sigma}_t\ C_t^T + Q_t)^{-1}$

$\mu_t = \bar{\mu}_t + K_t(z_t - C_t\ \bar{\mu}_t)$

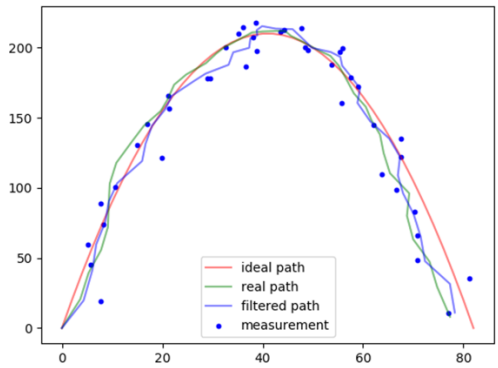
$\Sigma_t = (I - K_t\ C_t)\ \bar{\Sigma}_t$

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 [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).

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

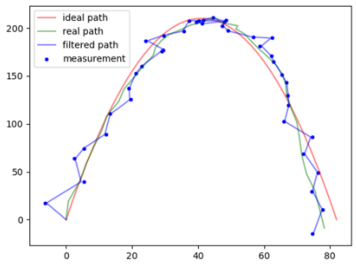
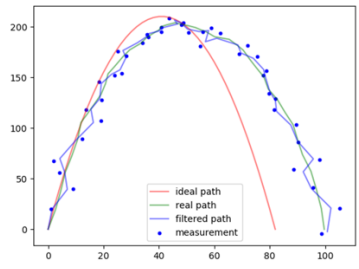
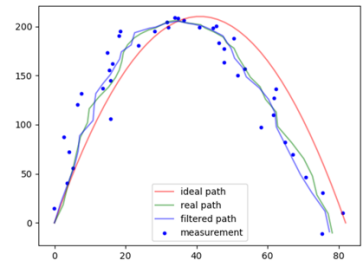
Measurement (z): Position [x, y] (with Gaussian noise added to x and y, having a mean of 0 and a variance of 3, respectively).

→ 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 (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 Q is set to a higher value, it implies that we trust the process model more, and the filter will be more responsive to changes in the predicted state.

Q = 0.1	Q = 1	Q = 10
		

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

R = 0.1	R = 1	R = 10