Exercise 07 for MA-INF 2201 Computer Vision WS25/26
Submission on 11.01.2026

Overview

This exercise focuses on state estimation and tracking, combining theoretical foundations with practical implementation. Use the numpy and matplotlib libraries for implementation and visualization. Provide detailed comments for your implementation.

1. **Theoretical Exercise:** Prove the Chapman-Kolmogorov relation *(4 points)*:

$$\begin{aligned}\Pr(\mathbf{w}_t \mid \mathbf{x}_{1:t-1}) &= \int \Pr(\mathbf{w}_t \mid \mathbf{w}_{t-1})\, \Pr(\mathbf{w}_{t-1} \mid \mathbf{x}_{1:t-1})\, d\mathbf{w}_{t-1} \\ &= \int \mathcal{N}_{\mathbf{w}_t}[\boldsymbol{\mu}_p + \boldsymbol{\Psi}\mathbf{w}_{t-1}, \boldsymbol{\Sigma}_p]\, \mathcal{N}_{\mathbf{w}_{t-1}}[\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_{t-1}]\, d\mathbf{w}_{t-1} \\ &= \mathcal{N}_{\mathbf{w}_t}[\boldsymbol{\mu}_p + \boldsymbol{\Psi}\boldsymbol{\mu}_{t-1}, \boldsymbol{\Sigma}_p + \boldsymbol{\Psi}\boldsymbol{\Sigma}_{t-1}\boldsymbol{\Psi}^T]\end{aligned}$$

2. **Kalman Filter Implementation:** Consider a 2D signal with acceleration. The state vector includes position, velocity, and acceleration components: $\mathbf{x} = [x, y, v_x, v_y, a_x, a_y]^T$.

   Data: The observation data is provided in 'data/observations.npy' file which contains noisy 2D position measurements file which can be loaded using:

```
observations = np.load('observations.npy')  # Shape: (N, 2)
```

   System Model:

   - Time step: $\Delta t = 0.1$ seconds
   - State transition matrix Ψ incorporating acceleration terms:

$$\Psi = \begin{bmatrix} 1 & 0 & \Delta t & 0 & \frac{\Delta t^2}{2} & 0 \\ 0 & 1 & 0 & \Delta t & 0 & \frac{\Delta t^2}{2} \\ 0 & 0 & 1 & 0 & \Delta t & 0 \\ 0 & 0 & 0 & 1 & 0 & \Delta t \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

   - Process Noise Covariance (Σ_p): Diagonal matrix with process noise parameter $sp = 0.001$
   - Measurement Matrix (Φ) observing only positions:

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

   - Measurement Noise Covariance (Σ_m): Diagonal matrix with measurement noise parameter $sm = 0.05$

2.1. What should be the time evolution equation? Explain each term. *(1 point)*

2.2. What should be the measurement equation? Explain each term.*(1 point)*

2.3. Implement a Kalman filter to track the signal: Initialize the state vector with $\mathbf{x}_0 = [-10, -150, 1, -2, 0, 0]^T$. Implement the prediction step (time update). and the correction step (measurement update). *(6 points)*

2.4. Visualize the observations vs. filtered estimates. *(2 points)*

3. **Fixed-Lag Smoothing:** Extend your Kalman filter implementation to perform fixed-lag smoothing for improved state estimation.

3.1. Implement the smoother with a lag of 5 time steps. *(6 points)*

3.2. Analysis of lag size effects. *(4 points)*

4. **Unicycle Motion Model**: Consider a unicycle motion model. The state vector in this model is represented by 4 elements: $\mathbf{x}_t = [x_t, y_t, \theta_t, v_t]^T$, where x_t and y_t represent the $2D$ position at time t, while θ_t and v_t represent angle and velocity, respectively.

System Model:

- The time evolution equation for this model is:

$$x_{t+1} = g(x_t) + \epsilon_t \tag{1}$$

$$x_{t+1} = \begin{bmatrix} x_t + \Delta t v_t cos(\theta_t) + \epsilon_x \\ y_t + \Delta t v_t sin(\theta_t) + \epsilon_y \\ \theta_t + \epsilon_\theta \\ v_t + \epsilon_v \end{bmatrix} \tag{2}$$

- The measurement equation for this model is:

$$m_t = h(x_t) + \delta_t \tag{3}$$

$$m_t = \begin{bmatrix} x_t + \delta_x \\ y_t + \delta_y \end{bmatrix} \tag{4}$$

4.1. Which components in g and h contain non-linearities? *(0.5 points)*

4.2. Compute the Jacobian of $g(x)$ *(1.5 points)*

4.3. Compute the Jacobian of $h(x)$ *(1.5 points)*

4.4. Generate $T = 200$ states from the time evolution model. Set the parameters as follows: $\Delta_t = 0.1$, $x_0 = [0, 0, 0, 1]$, $v_t = v_{t-1}$, $\theta_t = \theta_{t-1} + 0.6\sin(0.2t\Delta_t) \cdot \Delta_t$ and $\epsilon_t = [0.001, 0.001, 0.001, 0.001]$. *(0.5 points)*

4.5. Generate observations from these states using the measurement equation. Use $\delta_t = [0.005, 0.005]$. *(0.5 points)*

4.6. Implement the Extended Kalman filter. Why is it better to use the Extended Kalman filter in this case? Explain. *(8 points)*

4.7. Visualize states, observations, and filtered estimates. *(1.5 point)*

4.8. Increase δ_t to $\delta_t = [0.05, 0.05]$. What effect do you observe? Why is it the case? *(2 points)*

Submission Guidelines

- Submit your implementation in a Python file named solution.py.
- Include clear documentation and comments in your code.
- Provide theoretical answers and analysis in `answers.pdf`. (Please limit your answers to no more than 15 lines.)