Computer Vision 2 — Exercise 2

Extended Kalman Filter & Particle Filter

M.Sc. Francis Engelmann, Dr. Jörg Stückler
engelmann@vision.rwth-aachen.de, stueckler@vision.rwth-aachen.de

RWTH Aachen University, Computer Vision Group
http://www.vision.rwth-aachen.de
Content Exercise 2

• **Question 1: Extended Kalman Filter**
  - Compared to basic KF
  - Unicycle motion model
  - Nonlinearities & Jacobians

• **Question 2: Particle Filter**
  - Compared to general KF
  - SIR Algorithm
  - Implementation Details
Question 1: Extended Kalman Filter
Q1: EKF - Compared to basic KF

- **Kalman Filter**

  - Assumption: linear model
    \[
    x_t = D_t x_{t-1} + \varepsilon_t \\
    y_t = M_t x_t + \delta_t
    \]
  
  - Prediction step
    \[
    x_t^- = D_t x_{t-1}^+ \\
    \Sigma_t^- = D_t \Sigma_{t-1}^+ D_t^T + \Sigma_{d_t}
    \]
  
  - Correction step
    \[
    K_t = \Sigma_t^- M_t^T (M_t \Sigma_t^- M_t^T + \Sigma_{m_t})^{-1} \\
    x_t^+ = x_t^- + K_t (y_t - M_t x_t^-) \\
    \Sigma_t^+ = (I - K_t M_t) \Sigma_t^-
    \]

- **Extended Kalman Filter**

  - Nonlinear model
    \[
    x_t = g(x_{t-1}) + \varepsilon_t \\
    y_t = h(x_t) + \delta_t
    \]
  
  - Prediction step
    \[
    x_t^- = g(x_{t-1}^+) \\
    \Sigma_t^- = G_t \Sigma_{t-1}^+ G_t^T + \Sigma_{d_t}
    \]
  
  - Correction step
    \[
    K_t = \Sigma_t^- H_t^T (H_t \Sigma_t^- H_t^T + \Sigma_{m_t})^{-1} \\
    x_t^+ = x_t^- + K_t (y_t - h(x_t^-)) \\
    \Sigma_t^+ = (I - K_t H_t) \Sigma_t^-
    \]
Q1: EKF - Unicycle Motion Model

- The "unicycle" motion model is an approximation often used for bicycle or car motions.

- State vector:

\[ x_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \\ v_t \end{bmatrix} \]

- position x
- position y
- orientation angle
- velocity

Example Trajectory
Q1: EKF

a) Point out, which steps contain nonlinearities and give the definition of the functions $g$ and $h$.

**Dynamic Model**

\[
x_t = g(x_{t-1}) + \epsilon_t
\]

\[
x_t = \begin{bmatrix} x_t \\ y_t \\ \theta_t \\ v_t \end{bmatrix},
\]

\[
x_{t+1} = \begin{bmatrix} x_t + \Delta tv_t \cos \theta_t + \epsilon_x \\ y_t + \Delta tv_t \sin \theta_t + \epsilon_y \\ \theta_t + \epsilon_\theta \\ v_t + \epsilon_v \end{bmatrix}
\]

$\begin{align*}
g & : \mathbb{R}^4 \to \mathbb{R}^4, \\
\begin{bmatrix} x_t \\ y_t \\ \theta_t \\ v_t \end{bmatrix} & \mapsto \begin{bmatrix} x_t + \Delta tv_t \cos \theta_t \\ y_t + \Delta tv_t \sin \theta_t \\ \theta_t \\ v_t \end{bmatrix}
\end{align*}$

**Measurement Model**

\[
y_t = h(x_t) + \delta_t
\]

\[
y_t = \begin{bmatrix} x_t \\ y_t \end{bmatrix}
\]

$\begin{align*}
h & : \mathbb{R}^4 \to \mathbb{R}^2, \\
\begin{bmatrix} x_t \\ y_t \\ \theta_t \\ v_t \end{bmatrix} & \mapsto \begin{bmatrix} x_t \\ y_t \end{bmatrix}
\end{align*}$
Q1: EKF b)

b) Compute the Jacobians $G_t$ and $H_t$ of $g(x)$ and $h(x)$ respectively.

$g : \mathbb{R}^4 \rightarrow \mathbb{R}^4, \begin{bmatrix} x_t \\ y_t \\ \theta_t \\ v_t \end{bmatrix} \rightarrow \begin{bmatrix} x_t + \Delta t v_t \cos \theta_t \\ y_t + \Delta t v_t \sin \theta_t \\ \theta_t \\ v_t \end{bmatrix}$

$h : \mathbb{R}^4 \rightarrow \mathbb{R}^2, \begin{bmatrix} x_t \\ y_t \\ \theta_t \\ v_t \end{bmatrix} \rightarrow \begin{bmatrix} x_t \\ y_t \end{bmatrix}$

$G_t = \frac{\partial g(x)}{\partial x} = \begin{bmatrix} \frac{\partial g_1}{\partial x_t} & \frac{\partial g_1}{\partial y_t} & \frac{\partial g_1}{\partial \theta_t} & \frac{\partial g_1}{\partial v_t} \\ \frac{\partial g_2}{\partial x_t} & \frac{\partial g_2}{\partial y_t} & \frac{\partial g_2}{\partial \theta_t} & \frac{\partial g_2}{\partial v_t} \\ \frac{\partial g_3}{\partial x_t} & \frac{\partial g_3}{\partial y_t} & \frac{\partial g_3}{\partial \theta_t} & \frac{\partial g_3}{\partial v_t} \\ \frac{\partial g_4}{\partial x_t} & \frac{\partial g_4}{\partial y_t} & \frac{\partial g_4}{\partial \theta_t} & \frac{\partial g_4}{\partial v_t} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\Delta t v_t \sin \theta_t & \Delta t \cos \theta_t \\ 0 & 1 & \Delta t v_t \cos \theta_t & \Delta t \sin \theta_t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

$H_t = \frac{\partial h(x)}{\partial x} = \begin{bmatrix} \frac{\partial x_t}{\partial x_t} & \frac{\partial x_t}{\partial y_t} & \frac{\partial x_t}{\partial \theta_t} & \frac{\partial x_t}{\partial v_t} \\ \frac{\partial y_t}{\partial x_t} & \frac{\partial y_t}{\partial y_t} & \frac{\partial y_t}{\partial \theta_t} & \frac{\partial y_t}{\partial v_t} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$
Q1: EKF c)

c) Implement extended Kalman filter.

- Generate measurements by adding Gaussian noise to original state at each time step.
- Use EKF to estimate original trajectory based on noisy measurements.
Q1: EKF c)

c) Implement extended Kalman filter.
Q1: EKF c)

c) Implement extended Kalman filter.

- Influence of $\sum_{d_t}$ and $\sum_{m_t}$

Filtered trajectory from noisy measurements

\[
\sigma_d = \text{eye}(4) \times 0.01;
\]
\[
\sigma_m = \text{eye}(2) \times 0.5;
\]

Filtered trajectory from noisy measurements

\[
\sigma_d = \text{eye}(4) \times 0.00001;
\]
\[
\sigma_m = \text{eye}(2) \times 0.05;
\]
Question 2: Particle Filter
Q2: Particle Filter

- What is different from particle filters to Kalman filters?
  - **Kalman Filter**: all probability distributions are normal distributions.
  - **Particle Filter**: any distribution is possible.
  - In particular, this allows us to model multiple hypothesis for the state.
Q2: Particle Filter

• What is different from particle filters to Kalman filters?
  – In particular, this allows us to model **multiple** hypothesis for the state.

**Kalman Filter:** Single Mode

**Particle Filter:** Multiple Modes
Q2: Particle Filter

- **SIR** (Sampling Importance Resampling)

  
  Set of particles (samples from posterior at time t)

  
  Function: \( [x_t] = SIR([x_{t-1}, y_t]) \)

  \( \bar{x}_t = x_t = 0 \)

  for \( i = 1:N \)

  \( \text{Sample } x^i_t \sim p(x_t|x^i_{t-1}) \)

  \( w^i_t = p(y_t|x^i_t) \)

  end

  for \( i = 1:N \)

  \( \text{Draw } i \text{ with probability } \propto w^i_t \)

  \( \text{Add } x^i_t \text{ to } x_t \)

  end

  Number of particles

  Particle

  State transition distribution, according to dynamic model
  here: constant position model

  Resampling

  Particles \( x \)

  Detections \( y \)
Q2: Particle Filter

- **SIR** (Sampling Importance Resampling)

```matlab
function \( \tilde{X}_t = SIR[X_{t-1}, y_t] \)
\( \tilde{X}_t = X_t = \emptyset \)
for \( i = 1:N \)
  a) Sample \( x_i^t \sim p(x_t|x_{t-1}^t) \)
  \( w_i^t = p(y_t|x_i^t) \)
  b) \( w_i^t \)
  end
for \( i = 1:N \)
  c) Draw i with probability \( \propto w_i^t \)
     Add \( x_i^t \) to \( X_t \)
  end
```

*generate_particles.m*
*compute_particle_likelihood.m*
*inverse_transform_sampling.m*
Q2: Particle Filter a)

- Generate particles (samples)
  - Draw random samples from a 2D Normal distribution.
  - MATLAB's `randn` returns normally distributed samples.
Q2: Particle Filter a)

- **Generate particles (samples)**
  - Draw random samples from a 2D Normal distribution.
  - MATLAB’s \texttt{randn} returns normally distributed samples.
Q2: Particle Filter b)

- **Compute particle likelihood (weights)**
  - How likely does a particle correspond to a detection?
  - Measure: Parzen density estimation with Gaussian kernel

\[
    w^i_t = \sum_j \exp \left( -\frac{1}{2} \left( \frac{y^j_x - x^i_x}{\sigma^2_x} + \frac{y^j_y - x^i_y}{\sigma^2_y} \right) \right)
\]
Q2: Particle Filter - Resampling

- **Inverse transform sampling**
  - **Goal**: resample N particles from existing set of N particles, favor particles with larger weight.

1. From discrete particle distribution compute cumulative distribution. Each bin corresponds to a particle, the height of the bin corresponds to the weight.
2. Sample $u$ from uniform distribution between 0 and 1
3. Look up bin in cumulative distribution and pick resulting particle $x_j$
Q2: Particle Filter - Result