Probabilistic Robotics

Bayes Filter Implementations

Gaussian filters
Markov ⇔ Kalman Filter Localization

• **Markov localization**
  - localization starting from any unknown position
  - recovers from ambiguous situation.
  - However, to update the probability of all positions within the whole state space at any time requires a discrete representation of the space (grid). The required memory and calculation power can thus become very important if a fine grid is used.

• **Kalman filter localization**
  - tracks the robot and is inherently very precise and efficient.
  - However, if the uncertainty of the robot becomes too large (e.g. collision with an object), the Kalman filter will fail and the position is definitively lost.
Kalman Filter Localization

System

System error source

Control

System state
(desired but not known)

Measuring devices

Measurement error sources

Kalman filter

Optimal estimate of system state

Observed measurement
Bayes Filter Reminder

\[
Bel(x_t) = \eta \ P(z_t | x_t) \int P(x_t | u_t, x_{t-1}) \ Bel(x_{t-1}) \ dx_{t-1}
\]

1. Algorithm **Bayes_filter** (Bel(x), d):
2. \( \eta = 0 \)
3. If \( d \) is a perceptual data item \( z \) then
4. For all \( x \) do
5. \( Bel'(x) = P(z | x)Bel(x) \)
6. \( \eta = \eta + Bel'(x) \)
7. For all \( x \) do
8. \( Bel'(x) = \eta^{-1}Bel'(x) \)
9. Else if \( d \) is an action data item \( u \) then
10. For all \( x \) do
11. \( Bel'(x) = \int P(x | u, x') \ Bel(x') \ dx' \)
12. Return \( Bel'(x) \)
Bayes Filter Reminder

- **Prediction**
  \[
  \overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \, bel(x_{t-1}) \, dx_{t-1}
  \]

- **Correction**
  \[
  bel(x_t) = \eta \, p(z_t \mid x_t) \overline{bel}(x_t)
  \]
Kalman Filter

• Bayes filter with Gaussians
• Developed in the late 1950's
• Most relevant Bayes filter variant in practice
• Applications range from economics, weather forecasting, satellite navigation to robotics and many more.

• The Kalman filter "algorithm" is a couple of matrix multiplications!
Gaussians

\[ p(x) \sim N(\mu, \sigma^2) : \]

\[ p(x) = \frac{1}{\sqrt{2\pi} \sigma} e^{-\frac{1}{2} \frac{(x-\mu)^2}{\sigma^2}} \]

**Univariate**

\[ p(\mathbf{x}) \sim N(\mathbf{\mu}, \Sigma) : \]

\[ p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mathbf{\mu})' \Sigma^{-1} (\mathbf{x}-\mathbf{\mu})} \]

**Multivariate**
Gaussians

1D

2D

\[
C = \begin{bmatrix} 0.020 & 0.013 \\ 0.013 & 0.020 \end{bmatrix}
\]

\[
\lambda_1 = 0.007 \\
\lambda_2 = 0.033
\]

\[
\rho = \frac{\sigma_{XY}}{\sigma_X \sigma_Y} = 0.673
\]

3D

Video
Properties of Gaussians

• Univariate

\[ X \sim N(\mu, \sigma^2) \]
\[ Y = aX + b \]
\[ \Rightarrow \quad Y \sim N(a\mu + b, a^2\sigma^2) \]

\[ X_1 \sim N(\mu_1, \sigma_1^2) \]
\[ X_2 \sim N(\mu_2, \sigma_2^2) \]
\[ \Rightarrow \quad p(X_1) \cdot p(X_2) \sim N\left(\frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2} \mu_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} \mu_2, \quad \frac{1}{\sigma_1^{-2} + \sigma_2^{-2}}\right) \]

• Multivariate

\[ X \sim N(\mu, \Sigma) \]
\[ Y = AX + B \]
\[ \Rightarrow \quad Y \sim N(A\mu + B, A\Sigma A^T) \]

\[ X_1 \sim N(\mu_1, \Sigma_1) \]
\[ X_2 \sim N(\mu_2, \Sigma_2) \]
\[ \Rightarrow \quad p(X_1) \cdot p(X_2) \sim N\left(\frac{\Sigma_2}{\Sigma_1 + \Sigma_2} \mu_1 + \frac{\Sigma_1}{\Sigma_1 + \Sigma_2} \mu_2, \quad \frac{1}{\Sigma_1^{-1} + \Sigma_2^{-1}}\right) \]

• We stay in the “Gaussian world” as long as we start with Gaussians and perform only linear transformations
Introduction to Kalman Filter (1)

- Two measurements no dynamics
  \[ \hat{q}_1 = q_1 \text{ with variance } \sigma_1^2 \]
  \[ \hat{q}_2 = q_2 \text{ with variance } \sigma_2^2 \]

- Weighted least-square
  \[ S = \sum_{i=1}^{n} w_i (\hat{q} - q_i)^2 \]

- Finding minimum error
  \[
  \frac{\partial S}{\partial \hat{q}} = \frac{\partial}{\partial \hat{q}} \sum_{i=1}^{n} w_i (\hat{q} - q_i)^2 = 2 \sum_{i=1}^{n} w_i (\hat{q} - q_i) = 0
  \]

- After some calculation and rearrangements
  \[
  \hat{q} = q_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2} (q_2 - q_1)
  \]

- Another way to look at it – weighted mean
Discrete Kalman Filter

- Estimates the state $x$ of a discrete-time controlled process that is governed by the linear stochastic difference equation

$$x_t = A_t x_{t-1} + B_t u_t + \epsilon_t$$

- with a measurement

$$z_t = C_t x_t + \delta_t$$

$A_t$ Matrix (nxn) that describes how the state evolves from $t$ to $t-1$ without controls or noise.

$B_t$ Matrix (nxl) that describes how the control $u_t$ changes the state from $t$ to $t-1$.

$C_t$ Matrix (kxn) that describes how to map the state $x_t$ to an observation $z_t$.

$\epsilon_t$ Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance $R_t$ and $Q_t$ respectively.
Kalman Filter Updates in 1D

prediction

measurement

correction

It's a weighted mean!
Kalman Filter Updates in 1D

\[
\text{bel}(x_t) = \begin{cases} 
\mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\
\sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2
\end{cases} \quad \text{with} \quad K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}
\]

\[
\text{bel}(x_t) = \begin{cases} 
\mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\
\Sigma_t = (I - K_tC_t)\bar{\Sigma}_t
\end{cases} \quad \text{with} \quad K_t = \bar{\Sigma}_tC_t^T(C_t\bar{\Sigma}_tC_t^T + Q_t)^{-1}
\]
Kalman Filter Updates in 1D

\[ \overline{\text{bel}}(x_t) = \begin{cases} 
\overline{\mu}_t = a_t \mu_{t-1} + b_t u_t \\
\overline{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 
\end{cases} \]

\[ \overline{\text{bel}}(x_t) = \begin{cases} 
\overline{\mu}_t = A_t \mu_{t-1} + B_t u_t \\
\Sigma_t = A_t \Sigma_{t-1} A_t^T + R_t 
\end{cases} \]
Kalman Filter Updates
Linear Gaussian Systems: Initialization

- Initial belief is normally distributed:

\[ bel(x_0) = N(x_0; \mu_0, \Sigma_0) \]
Linear Gaussian Systems: Dynamics

- Dynamics are linear function of state and control plus additive noise:

\[
x_t = A_t x_{t-1} + B_t u_t + \epsilon_t
\]

\[
p(x_t \mid u_t, x_{t-1}) = N(x_t \mid A_t x_{t-1} + B_t u_t, R_t)
\]

\[
\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \quad bel(x_{t-1}) \, dx_{t-1}
\]

\[
\Downarrow
\]

\[
\sim N(x_t \mid A_t x_{t-1} + B_t u_t, R_t)
\]

\[
\Downarrow
\]

\[
\sim N(x_{t-1} \mid \mu_{t-1}, \Sigma_{t-1})
\]
Linear Gaussian Systems: Dynamics

\[ \bar{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \, \bar{bel}(x_{t-1}) \, dx_{t-1} \]

\[ \Downarrow \]

\[ \sim N(x_t; A_t x_{t-1} + B_t u_t, R_t) \sim N(x_{t-1}; \mu_{t-1}, \Sigma_{t-1}) \]

\[ \Downarrow \]

\[ \bar{bel}(x_t) = \eta \int \exp \left\{ - \frac{1}{2} (x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1} (x_t - A_t x_{t-1} - B_t u_t) \right\} dx_{t-1} \]

\[ \exp \left\{ - \frac{1}{2} (x_{t-1} - \mu_{t-1})^T \Sigma_{t-1}^{-1} (x_{t-1} - \mu_{t-1}) \right\} \]

\[ \bar{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases} \]
Linear Gaussian Systems: Observations

- Observations are linear function of state plus additive noise:

\[ z_t = C_t x_t + \delta_t \]

\[ p(z_t \mid x_t) = N(z_t; C_t x_t, Q_t) \]

\[ bel(x_t) = \eta \quad p(z_t \mid x_t) \]

\[ \downarrow \]

\[ \sim N(z_t; C_t x_t, Q_t) \]

\[ \sim N(x_t; \mu_t, \Sigma_t) \]
Linear Gaussian Systems: Observations

\[ bel(x_i) = \eta \ p(z_t \mid x_t) \]

\[ \downarrow \]

\[ \sim N(z_t; C_t x_t, Q_t) \]

\[ \sim N(x_t; \mu_t, \Sigma_t) \]

\[ \downarrow \]

\[ bel(x_i) = \eta \ \exp \left\{ -\frac{1}{2} (z_t - C_t x_t)^T Q_t^{-1} (z_t - C_t x_t) \right\} \exp \left\{ -\frac{1}{2} (x_t - \mu_t)^T \Sigma_t^{-1} (x_t - \mu_t) \right\} \]

\[ bel(x_i) = \begin{cases} 
\mu_t &= \mu_t + K_t (z_t - C_t \mu_t) \\
\Sigma_t &= (I - K_t C_t) \Sigma_t 
\end{cases} \quad \text{with} \quad K_t = \Sigma_t C_t^T (C_t \Sigma_t C_t^T + Q_t)^{-1} \]
Kalman Filter Algorithm

1. **Algorithm** \texttt{Kalman\_filter}( \mu_{t-1}, \Sigma_{t-1}, u_t, z_t ):

2. **Prediction:**
   \[ \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \]
   \[ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \]

3. **Correction:**
   \[ K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1} \]
   \[ \mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t) \]
   \[ \Sigma_t = (I - K_t C_t) \bar{\Sigma}_t \]

4. **Return** \( \mu_t, \Sigma_t \)
Kalman Filter Algorithm
Kalman Filter Algorithm

- **Prediction**
  \[
  \hat{x}(k+1|k) = f(\hat{x}(k|k), u(k+1))
  \]
  \[
  P(k+1|k) = \nabla f_x P(k|k) \nabla f_x^T + \nabla f_u U(k+1) \nabla f_u^T
  \]

- **Observation**
  \[
  z(k+1), R(k+1)
  \]

- **Matching**
  \[
  v_{m_j}(k+1) = z_{m_j}(k+1) - h(\hat{x}(k+1|k), m_j)
  \]

- **Correction**
  \[
  \hat{x}(k+1|k+1) = \hat{x}(k+1|k) + W(k+1) v(k+1)
  \]
  \[
  P(k+1|k+1) = P(k+1|k) - W(k+1) S(k+1) W^T(k+1)
  \]
The Prediction-Correction-Cycle

\[
\overline{bel}(x_t) = \begin{cases} 
\mu_t = a_t \mu_{t-1} + b_t u_t \\
\sigma_t^2 = a_t^2 \sigma_{t-1}^2 + \sigma_{act,t}^2 
\end{cases}
\]

\[
\overline{bel}(x_t) = \begin{cases} 
\mu_t = A_t \mu_{t-1} + B_t u_t \\
\Sigma_t = A_t \Sigma_{t-1} A_t^T + R_t 
\end{cases}
\]
The Prediction-Correction-Cycle

\[ \text{bel}(x_t) = \left\{ \begin{array}{l} \mu_t = \overline{\mu}_t + K_t(z_t - \overline{\mu}_t) \\
\sigma_t^2 = (1 - K_t)\overline{\sigma}_t^2 \end{array} \right., \quad K_t = \frac{\overline{\sigma}_t^2}{\sigma_t^2 + \overline{\sigma}_\text{obs,t}^2} \]

\[ \text{bel}(x_t) = \left\{ \begin{array}{l} \mu_t = \overline{\mu}_t + K_t(z_t - C_t\overline{\mu}_t) \\
\Sigma_t = (I - K_tC_t)\overline{\Sigma}_t \end{array} \right., \quad K_t = \overline{\Sigma}_tC_t^T(C_t\overline{\Sigma}_tC_t^T + Q_t)^{-1} \]
The Prediction-Correction-Cycle

\[
\begin{align*}
\text{bel}(x_t) &= \left\{ \begin{array}{l}
\mu_t = \bar{\mu}_t + K_t (z_t - \bar{\mu}_t) \\
\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t
\end{array} \right. \\
& \quad K_t = \frac{\sigma^2_t}{\sigma^2_t + \sigma^2_{\text{obs},t}} \\
\text{Correction}
\end{align*}
\]

\[
\begin{align*}
\overline{\text{bel}}(x_t) &= \left\{ \begin{array}{l}
\bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\
\bar{\Sigma}_t = a_t^2 \sigma_t^2 + \sigma^2_{\text{act},t}
\end{array} \right.
\end{align*}
\]

\[
\begin{align*}
\text{Prediction}
\end{align*}
\]

\[
\begin{align*}
\text{bel}(x_t) &= \left\{ \begin{array}{l}
\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t) \\
\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t
\end{array} \right. \\
& \quad K_t = \frac{\bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}}{\Sigma_t}
\end{align*}
\]

\[
\begin{align*}
\overline{\text{bel}}(x_t) &= \left\{ \begin{array}{l}
\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\
\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t
\end{array} \right.
\end{align*}
\]
Kalman Filter Summary

- **Highly efficient**: Polynomial in measurement dimensionality $k$ and state dimensionality $n$:
  \[ O(k^{2.376} + n^2) \]

- Optimal for linear Gaussian systems!

- Most robotics systems are **nonlinear**!
Nonlinear Dynamic Systems

• Most realistic robotic problems involve nonlinear functions

\[ x_t = g(u_t, x_{t-1}) \]

\[ z_t = h(x_t) \]

• To be continued