Probabilistic Robotics

Bayes Filter Implementations

Gaussian filters

Markov $\Leftrightarrow$ 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

Bayes Filter Reminder

\[
Bel(x_i) = \eta \cdot P(z_i \mid x_i) \int P(x_i \mid u_i, x_{i-1}) \ Bel(x_{i-1}) \ dx_{i-1}
\]

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. \( Bel'(x) = P(z \mid x)Bel(x) \)
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. \( Bel'(x) = \int P(x \mid u, x') \ Bel(x') \ dx' \)
12. **Return** Bel'(x)
Bayes Filter Reminder

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

Univariate

\[ p(x) \sim N(\mu, \sigma^2) : \]
\[ p(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{1}{2} \left( \frac{x-\mu}{\sigma} \right)^2} \]

Multivariate

\[ p(x) \sim N(\mu, \Sigma) : \]
\[ p(x) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2} (x-\mu)^T \Sigma^{-1} (x-\mu)} \]

1D

\[ C = \begin{bmatrix} 0.020 & 0.013 \\ 0.013 & 0.020 \end{bmatrix} \]
\[ \lambda_1 = 0.007 \]
\[ \lambda_2 = 0.033 \]
\[ \rho = \frac{\sigma_{x'} \sigma_{y'}}{\sigma_{x} \sigma_{y}} = 0.673 \]
Introduction to Kalman Filter (1)

- Two measurements no dynamics
  \( \hat{q}_1 = q_1 \) with variance \( \sigma_1^2 \)
  \( \hat{q}_2 = q_2 \) 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 – weighed mean

Properties of Gaussians

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

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

- We stay in the “Gaussian world” as long as we start with Gaussians and perform only linear transformations
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

It's a weighted mean!
Kalman Filter Updates in 1D/2D

\[
\begin{align*}
\text{bel}(x_i) &= \begin{cases} 
\mu_i = \bar{\mu}_i + K_i (z_i - \bar{\mu}_i) \\
\sigma_i^2 = (1 - K_i) \bar{\sigma}_i^2
\end{cases} \quad \text{with} \quad K_i = \frac{\bar{\sigma}_i^2}{\bar{\sigma}_i^2 + \bar{\sigma}_{\text{obs}}^2}
\end{align*}
\]

\[
\text{bel}(x_i) = \begin{cases} 
\mu_i = \bar{\mu}_i + K_i (z_i - \bar{\mu}_i) \\
\Sigma_i = (I - K_i C_i) \bar{\Sigma}_i
\end{cases} \quad \text{with} \quad K_i = \bar{\Sigma}_i C_i^T (C_i \bar{\Sigma}_i C_i^T + Q_i)^{-1}
\]
Kalman Filter Updates

Linear Gaussian Systems: Initialization

- Initial belief is normally distributed:

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

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

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

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

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

\[ \overline{bel}(x_t) \Downarrow \overline{bel}(x_{t-1}) \Downarrow \]

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

---

Linear Gaussian Systems: Dynamics

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

\[ \downarrow \]

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

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

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

\[ \overline{bel}(x_t) = \left\{ \overline{\mu}_t = A_t \mu_{t-1} + B_t u_t, \overline{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \right\} \]
Observations are linear function of state plus additive noise:

\[ z_t = C_t x_t + \delta_t \]

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

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

\[ \downarrow \quad \downarrow \]

\[ \sim N(z_t; C_t x_t, Q_t) \quad \sim N(x_t; \bar{\mu}_t, \bar{\Sigma}_t) \]

\[ bel(x_t) = \eta \exp \left( -\frac{1}{2} (z_t - C_t x_t)^T Q_t^{-1} (z_t - C_t x_t) \right) \exp \left( -\frac{1}{2} (x_t - \bar{\mu}_t)^T \bar{\Sigma}_t^{-1} (x_t - \bar{\mu}_t) \right) \]

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

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

2. Prediction:
3. \mu_t = A \mu_{t-1} + B u_t
4. \Sigma_t = A \Sigma_{t-1} A^T + R

5. Correction:
6. \hat{K}_t = \Sigma_t C^T (C \Sigma_t C^T + Q)^{-1}
7. \mu_t = \mu_t + \hat{K}_t (z_t - C \mu_t)
8. \Sigma_t = (I - \hat{K}_t C) \Sigma_t
9. Return \mu_t, \Sigma_t
Kalman Filter Algorithm

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

- Observation
  \[ z(k+1) = H_x \hat{x}(k+1|k) \]

- Matching
  \[ H = H_x \hat{x}(k+1|k) \]

- Correction
  \[ \hat{x}(k+1|k+1) = \hat{x}(k+1|k) + K_z(z(k+1) - H_x \hat{x}(k+1|k)) \]
  \[ P(k+1|k+1) = P(k+1|k) - K_z H_x P(k+1|k) \]

The Prediction-Correction-Cycle

\[ P(z) = a_1 \mu + b_1 \sigma^2 \]
\[ \sigma^2 = a_1^2 \sigma^2 + \sigma_{\text{act}}^2 \]
\[ P(z) = A \mu + B \sigma \]
\[ \Sigma = A \Sigma A^T + R \]
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