Localization: Part 6
The Kalman Filter

Computer Science 4766/6912

Department of Computer Science
Memorial University of Newfoundland

July 11, 2018
The Kalman filter is perhaps the most widely-known implementation of Bayes filter. Numerous applications outside of robot localization. Based upon a Gaussian belief representation:
- Belief represented as the mean $\mu_t$ and covariance matrix $\Sigma_t$ of the Gaussian.

Assumes that the system update equations are linear:
- Non-linearity handled by linearization: The Extended Kalman Filter.
Assumptions

- “The system” is some process evolving over time, through control inputs $u_t$ and measurements $z_t$
- It has an unknown state $x_t$ and unknown past state $x_{t-1}$
- $x_t$, $u_t$, and $z_t$ are all column vectors with $n$, $m$, and $k$ elements, respectively
- We assume that $x_t$ evolves through the following linear update equation,

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

- $A_t$ is a matrix of size $n \times n$
- $B_t$ is a matrix of size $n \times m$
- $\epsilon_t$ is a Gaussian random vector of size $n$
  - Zero mean and covariance matrix $R_t$
The measurements $z_t$ are assumed to be another linear function of the system state,

$$z_t = C_t x_t + \delta_t$$

- $C_t$ is a matrix of size $k \times n$
- $\delta_t$ is a Gaussian random vector of size $k$
  - Zero mean and covariance matrix $Q_t$
- It is necessary for the noise vectors $(\epsilon_0, \epsilon_1, \ldots, \epsilon_t, \delta_0, \delta_1, \ldots, \delta_t)$ to be mutually independent.

$bel(x_t)$ is represented by mean $\mu_t$ and covariance matrix $\Sigma_t$

- The initial belief $bel(x_0)$ must have a Gaussian p.d.f with mean $\mu_0$ and covariance matrix $\Sigma_0$
- If the above assumptions hold, the belief $bel(x_t)$ will always be Gaussian
The Kalman Filter Algorithm

1. \texttt{Kalman\_Filter}(\mu_{t-1}, \Sigma_{t-1}, u_t, z_t)
2. \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t
3. \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t
4. \texttt{K}_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}
5. \mu_t = \bar{\mu}_t + \texttt{K}_t (z_t - C_t \bar{\mu}_t)
6. \Sigma_t = (I - \texttt{K}_t C_t) \bar{\Sigma}_t
7. return \mu_t, \Sigma_t

Lines 2 and 3 implement the prediction step, which determines \textit{bel}(x_t).

Lines 4 - 6 implement the measurement update, which incorporates \( z_t \) to determine \textit{bel}(x_t). On line 4 the \textit{Kalman gain}, \texttt{K}_t, is computed. \texttt{K}_t specifies the degree of trust placed in \( z_t \) over \textit{bel}(x_t).

On line 5 the actual measurement \( z_t \) is compared with the \textbf{measurement prediction} \( C_t \bar{\mu}_t \). This difference is known as the \textbf{innovation}. It describes how different \( z_t \) is from what was expected.
Example: Simple 1-D Robot

We apply the Kalman filter to a simple robot living in a 1-D world. The “state vector” is just the scalar position $x$. The matrices $A_t$, $B_t$, and $C_t$ are assumed constant and 1-D—they reduce to scalars $a$, $b$, and $c$. The covariance matrices $R_t$ and $Q_t$ are also reduced to scalars $r$ and $q$. (Here, $r = 0.25$ and $q = 1$)

We assume that unless it is commanded to move, the robot will maintain its current position: $a = 1$

The control signal $u_t$ is the distance the robot is commanded to move: $b = 1$ (the robot obeys and moves the commanded distance)

The robot has a position sensor which gives 1-D position directly: $c = 1$

The robot begins with $\mu_0 = 0, \sigma_0 = 0...$
Notice when $x_t = 2$ we get an erroneous sensor value of $z = 4$. The new belief is closer to 2 than 4 because $\bar{\sigma} < q$
Consider a robot constrained to move along a 1-D track. We wish to track the robot’s position and speed over time. The state vector is,

\[ x_t = \begin{bmatrix} p_t \\ v_t \end{bmatrix} \]

where \( p_t \) is position and \( v_t \) is velocity. The control input, \( u_t \), is a force applied on the robot, which has a mass of \( m \). From Newton’s second law we know that force = mass \( \times \) acceleration. Thus,

\[ u = m \frac{dv}{dt} \]

which means that the acceleration \( \frac{dv}{dt} = \frac{u}{m} \). We will assume that the time period between discrete updates, \( \Delta t \), is sufficiently small such that,

\[ \frac{dv}{dt} \approx \frac{v_t - v_{t-1}}{\Delta t} \]
Given these assumptions we can give update equations for our two main variables,

\[ p_t = p_{t-1} + \Delta t v_{t-1} + \text{Noise} \]
\[ v_t = v_{t-1} + \frac{\Delta t}{m} u_t + \text{Noise} \]

In order to apply the Kalman filter, we must write these equations together in matrix form,

\[
\begin{bmatrix}
    p_t \\
    v_t
\end{bmatrix} = \begin{bmatrix}
    1 & \Delta t \\
    0 & 1
\end{bmatrix} \begin{bmatrix}
    p_{t-1} \\
    v_{t-1}
\end{bmatrix} + \begin{bmatrix}
    0 \\
    \frac{\Delta t}{m}
\end{bmatrix} u_t + \epsilon_t
\]

where \( \epsilon_t \) represents additive Gaussian noise with covariance matrix \( R_t \). This equation is now in the standard form required by the Kalman filter:

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

Assume that this robot is also equipped with a position sensor (subject to noise of course)…
The standard form for generating measurements is as follows,

$$z_t = C_t x_t + \delta_t$$

In our case the matrix $C_t$ is quite simple,

$$z_t = [1 \ 0] \begin{bmatrix} p_t \\ v_t \end{bmatrix} + \delta_t$$

where $\delta_t$ represents Gaussian noise with covariance matrix $Q_t$. In this case, $Q_t$ is just a variance.

The Kalman filter can now be applied. The parameters we will use are as follows:

- Covariance matrix of motion equation: $R = \begin{bmatrix} 0.2 & 0.05 \\ 0.05 & 0.1 \end{bmatrix}$
- Variance of measurement equation: $Q = 0.5$
- Mass of robot: $m = 1$
- Time step: $\Delta t = 1$
Properties of Kalman filters:

- Kalman filters are highly efficient
  - Cost of matrix inversion on line 4: $O(k^{2.376})$
  - Cost of multiplying $n \times n$ matrices: $O(n^2)$
- Optimal for linear Gaussian systems...
- Unfortunately, most robotic systems are non-linear!
- For non-linear systems you can linearize and apply the Extended Kalman Filter