1 Kalman Filter

Linear and Gaussian Models

For state vector $\mathbf{x}_k$ and observation $\mathbf{y}_k$:

$\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{q}_{k-1} \qquad \mathbf{q}_{k-1} \sim \mathcal{N}(\overline{\mathbf{q}}_{k-1}, \mathbf{Q}_{k-1})$

$\mathbf{y}_k = \mathbf{H}_k\mathbf{x}_k + \mathbf{r}_k \qquad \mathbf{r}_k \sim \mathcal{N}(\overline{\mathbf{r}}_k, \mathbf{R}_k)$

$\mathbf{x}_0 \sim \mathcal{N}(\overline{\mathbf{x}}_0, \mathbf{P}_{0|0}) \qquad \overline{\mathbf{q}}_{k-1} = 0 \qquad \overline{\mathbf{r}}_k = 0$

Kalman gain: $\mathbf{K}_k = \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}$

Innovation: $\mathbf{v}_k = \mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$

Innovation covariance: $\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k$

It holds that $p(\mathbf{y}_k|\mathbf{y}_{1:k-1}) = \mathcal{N}(\mathbf{y}_k; \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}, \mathbf{S}_k)$, which means that $\mathbf{S}_k$ is the predicted covariance of $\mathbf{y}_k$

The innovation $\mathbf{v}_k = \mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$ captures the new information in $\mathbf{y}_k$

In $\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\mathbf{v}_k$, the Kalman gain $\mathbf{K}_k$ determines how much we should trust the new information

The Kalman filter recursively computes:

Prediction: $p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1})$

Update: $p(\mathbf{x}_k|\mathbf{y}_{1:k}) = \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{x}}_{k|k}, \mathbf{P}_{k|k})$

Prediction

Compute $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$

$p(\mathbf{x}_{k-1}|\mathbf{y}_{1:k-1}) = \mathcal{N}(\mathbf{x}_{k-1}; \hat{\mathbf{x}}_{k-1|k-1}, \mathbf{P}_{k-1|k-1})$

$p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) = \mathcal{N}(\mathbf{x}_k; \mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1|k-1}, \mathbf{A}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1})$

$\hat{\mathbf{x}}_{k|k-1} = \mathbf{A}_{k-1}\hat{\mathbf{x}}_{k-1|k-1}$

$\mathbf{P}_{k|k-1} = \mathbf{A}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{A}_{k-1}^T + \mathbf{Q}_{k-1}$

Update

$p(\mathbf{x}_k|\mathbf{x}_{k-1}) = \mathcal{N}(\mathbf{x}_k; \mathbf{A}_{k-1}\mathbf{x}_{k-1}, \mathbf{Q}_{k-1})$

$p(\mathbf{y}_k|\mathbf{x}_k) = \mathcal{N}(\mathbf{y}_k; \mathbf{H}_k\mathbf{x}_k, \mathbf{R}_k)$

$$\rightarrow \begin{bmatrix}\mathbf{x}_k \\ \mathbf{y}_k\end{bmatrix} \,\middle|\, \mathbf{y}_{1:k-1} \sim \mathcal{N}\left(\begin{bmatrix}\hat{\mathbf{x}}_{k|k-1} \\ \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}\end{bmatrix}, \begin{bmatrix}\mathbf{P}_{k|k-1} & \mathbf{P}_{k|k-1}\mathbf{H}_k^T \\ \mathbf{H}_k\mathbf{P}_{k|k-1} & \mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^T + \mathbf{R}_k\end{bmatrix}\right)$$

Lemma for joint probability of x and y

$p(\mathbf{x}|\mathbf{y}) = \mathcal{N}(\mathbf{x}; \mu_x + \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}(\mathbf{y} - \mu_y), \mathbf{P}_{xx} - \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}\mathbf{P}_{yx})$

$$\begin{aligned}\hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}(\mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}) \\ &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\mathbf{v}_k\end{aligned}$$

$$\begin{aligned}P_{k|k} &= \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}(\mathbf{P}_{k|k-1}\mathbf{H}_k^T)^T \\ &= \mathbf{P}_{k|k-1} - \mathbf{P}_{k|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}\mathbf{S}_k\mathbf{S}_k^T(\mathbf{P}_{k|k-1}\mathbf{H}_k^T)^T \\ &= \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^T\end{aligned}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k\mathbf{v}_k$$
$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k\mathbf{S}_k\mathbf{K}_k^T$$

Decomposing joint expectations

$\mathbb{E}[g(x,y)] = \mathbb{E}[\mathbb{E}[g(x,y)|y]] = \mathbb{E}[h(y)]$

A well performing filter should satisfy

$\mathbb{E}[\mathbf{x}_k - \hat{\mathbf{x}}_{k|k}] = 0$

$\mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})^T|\mathbf{y}_{1:k}] = \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})^T]$

Innovation Consistency

The innovation $\mathbf{v}_k = \mathbf{y}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}$ should satisfy

$p(\mathbf{v}_k|\mathbf{y}_{1:k-1}) = \mathcal{N}(\mathbf{v}_k; \mathbf{0}, \mathbf{S}_k)$

$\mathrm{Cov}(\mathbf{v}_k, \mathbf{v}_{k-l}) = \mathrm{Cov}(\mathbf{v}_k)$ if $l = 0$ else $\mathbf{0}$

$\mathbf{S}_k^{-1/2}\mathbf{v}_k \sim \mathcal{N}(0, I) \rightarrow \mathbf{v}_k^T\mathbf{S}_k^{-1}\mathbf{v}_k \sim \mathcal{X}^2 n_y$

$\xi K = \sum_{k=1}^{K} \mathbf{v}_k^T\mathbf{S}_k^{-1}\mathbf{v}_k \sim \mathcal{N}(Kn_y, 2Kn_y)$

within 3σ-region?

Correlation

$$\rho(l) = \frac{\sum_{k=l+1}^{K} \mathbf{v}_k^T\mathbf{v}_{k-l}}{\sum_{\tau=l+1}^{K} \mathbf{v}_\tau^T\mathbf{v}_\tau}$$

check if $\rho(l) \approx 0$ for $l > 0$

A key aspect in tuning is to select the SNR (signal-to-noise ratio) $\|\mathbf{Q}\|/\|\mathbf{R}\|$
If SNR is large, a quickly adapting filter that relies more on new data than predictions.
If SNR is low, the data is noise and we rely more on the predictions, the filter thus adapts slowly to data.
The sensor noise, $\mathbf{R}$, is often described by the manufacturer and/or possible to collect data from which it can be estimated.
The motion noise, $\mathbf{Q}$, is then selected by tuning.
Unless you know the state sequence, study properties of the innovation to guide the tuning of the filter.

The Kalman filter is an LMMSE (Linear Minimum Mean Square Error) Estimator

$\hat{\mathbf{x}}_{k|k-1} = \mathbb{E}[\mathbf{x}_k|\mathbf{y}_{1:k-1}]$ $\hat{\mathbf{x}}_{k|k} = \mathbb{E}[\mathbf{x}_k|\mathbf{y}_{1:k}]$

LMMSE Objective in the Static Case

Find $\mathbf{A}$ and $\mathbf{b}$ such that $\hat{\mathbf{x}} = \mathbf{A}\mathbf{y} + \mathbf{b}$ yields the smallest possible MSE, $\mathbb{E}[(\mathbf{x} - \hat{\mathbf{x}})^T(\mathbf{x} - \hat{\mathbf{x}})]$

Finding optimum: Setting derivatives of MSE with respect to $\mathbf{b}$ and $\mathbf{A}$ to 0 yields:

$\mathbf{x} = \overline{\mathbf{x}} - \mathbf{A}\overline{\mathbf{y}}$ and $\mathbf{A} = \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}$

Orthogonality principle: Select $\mathbf{A}$ such that $\mathbb{E}[(\mathbf{x} - \hat{\mathbf{x}})^T\mathbf{y}^T] = 0$

$\hat{\mathbf{x}} = \overline{\mathbf{x}} + \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1}(\mathbf{y} - \overline{\mathbf{y}})$

LMMSE Objective in the Dynamic Case

Sequentially find $[\mathbf{L}_{k|k-1}, \mathbf{b}_{k|k-1}]$ and $[\mathbf{L}_{k|k}, \mathbf{b}_{k|k}]$ such that

$\hat{\mathbf{x}}_{k|k-1} = \mathbf{L}_{k|k-1}\mathbf{y}_{1:k-1} + \mathbf{b}_{k|k-1}$ $\hat{\mathbf{x}}_{k|k} = \mathbf{L}_{k|k}\mathbf{y}_{1:k} + \mathbf{b}_{k|k}$

Minimize the MSE, $\mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})^T(.)]$ and $\mathbf{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})^T(.)]$

$\mathbf{x}_0$, $\mathbf{q}_{k-1}$, $\mathbf{r}_k$ are independent random variables (not necessarily gaussian) with known mean and covariances

$\mathbf{P}_{k|k-1} = \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})(.)^T]$ $\mathbf{P}_{k|k} = \mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(.)^T]$

Proof

Assumption: $\mathbb{E}[(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1})\mathbf{y}_{1:k-1}^T] = 0$ $\mathbb{E}[(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1|k-1})(.)^T] = \mathbf{P}_{k-1|k-1}$

Prediction: $\mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})\mathbf{y}_{1:k-1}^T] = 0$ $\mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k-1})(.)^T] = \mathbf{P}_{k|k-1}$

Update: $\mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})\mathbf{y}_{1:k-1}^T] = 0$ $\mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})\mathbf{y}_k^T] = 0$ $\mathbb{E}[(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})(\mathbf{x}_k - \hat{\mathbf{x}}_{k|k})^T] = \mathbf{P}_{k|k}$

2 Motion and Measurement Models

Translational Kinematics
Random walks, constant velocity and constant acceleration models are important examples. Here we often view objects as point objects.

Rotational Kinematics
Useful when we wish to orient an object in 2D or in 3D. The orientation may also be connected to the translation of the object.

Difficulties
Motions are often conveniently described using differential equations. How can we discretize such models? We may have a reasonable description of the noise in the time domain. How can we describe the noise covariance in the discrete time model?

Discretization
Given a continuous-time motion model $\dot{\mathbf{x}}(t) = \tilde{a}(\mathbf{x}(t)) + \tilde{\mathbf{q}}(t)$, we would like to find a discrete-time motion model: $\mathbf{x}_k = a(\mathbf{x}_{k-1}) + \mathbf{q}_{k-1}$, where the discrete sequence is sampled from the continuous one, $\mathbf{x}_k = \mathbf{x}(kT)$.

Express distribution of $\mathbf{x}(t+T)$ given $\mathbf{x}(t)$ for different continuous time motion models. Two important tools: the Euler discretization method and an analytical solution of linear systems.

Euler Method

$$\dot{\mathbf{x}}(t) = \tilde{a}(\mathbf{x}(t)) + \tilde{\mathbf{q}}(t)$$

$$\dot{\mathbf{x}}(t) \approx \frac{\mathbf{x}(t+T) - \mathbf{x}(t)}{T}$$

$$\mathbf{x}(t+T) = \mathbf{x}(t) T\dot{\mathbf{x}}(t)$$

$$\mathbf{x}(t+T) = \mathbf{x}(t) + T\tilde{a}(\mathbf{x}(t)) + T\tilde{\mathbf{q}}(t)$$

Analytical Solution of Linear Systems

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{b}$$

$$\frac{d}{dt} e^{-\mathbf{A}t}\mathbf{x}(t) = e^{-\mathbf{A}t}\dot{\mathbf{x}}(t) - e^{-\mathbf{A}t}\mathbf{A}\mathbf{x}(t)$$

$$\begin{aligned}
\dot{\mathbf{x}}(t) - \mathbf{A}\mathbf{x}(t) &= \mathbf{b} \\
e^{-\mathbf{A}t}\dot{\mathbf{x}}(t) - e^{-\mathbf{A}t}\mathbf{A}\mathbf{x}(t) &= e^{-\mathbf{A}t}\mathbf{b} \\
\frac{d}{dt} e^{-\mathbf{A}t}\mathbf{x}(t) &= e^{-\mathbf{A}t}\mathbf{b} \\
e^{-\mathbf{A}(t+T)}\mathbf{x}(t+T) - e^{-\mathbf{A}t}\mathbf{x}(t) &= \int_t^{t+T} e^{-\mathbf{A}\tau}\mathbf{b}\, d\tau \\
\mathbf{x}(t+T) &= e^{\mathbf{A}T}\mathbf{x}(t) + \int_0^T e^{\mathbf{A}\tau}\, d\tau \mathbf{b} \\
\mathbf{x}(t+T) &= e^{\mathbf{A}T}\mathbf{x}(t) + (\mathbf{I}T + \frac{\mathbf{A}T^2}{2} + \frac{\mathbf{A}^2T^3}{3!} + ...)\mathbf{b}
\end{aligned}$$

Finding the Transition Matrix
Suppose $\dot{\mathbf{x}}(t) = \tilde{\mathbf{A}}\mathbf{x}(t) + \tilde{\mathbf{q}}(t)$. How should we select $\mathbf{A}_{k-1}$ in $\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{q}_{k-1}$?

Euler Method
$\mathbf{x}(t+T) \approx \mathbf{x}(t) + T(\tilde{\mathbf{A}}\mathbf{x}(t) + \tilde{\mathbf{q}}(t))$
$\mathbf{A}_{k-1} = \mathbf{I} + T\tilde{\mathbf{A}}$

Exact Solution for Linear Systems
$\mathbf{x}(t+T) = e^{\tilde{\mathbf{A}}T}\mathbf{x}(t) + \int_t^{t+T} e^{\mathbf{A}(t+T-\tau)}\tilde{\mathbf{q}}(\tau)\, d\tau$
$\mathbf{A}_{k-1} = e^{T\tilde{\mathbf{A}}} = \mathbf{I} + \tilde{\mathbf{A}}T + \tilde{\mathbf{A}}^2T^2/2 + ...$

Example: Continuous-time Constant Velocity Motion Model
$\mathbf{x}(t) = \begin{bmatrix} p(t) & v(t) \end{bmatrix}^T$
$\begin{bmatrix} \dot{p}(t) \\ \dot{v}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} p(t) \\ v(t) \end{bmatrix} + \tilde{\mathbf{q}}(t)$
Euler Method: $\mathbf{A}_{k-1} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$
Exact Solution: $\mathbf{A}_{k-1} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$
$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix} \qquad \mathbf{A}_{k-1} = \begin{bmatrix} \mathbf{I}_n & T\mathbf{I}_n \\ \mathbf{0}_n & \mathbf{I}_n \end{bmatrix}$

Example: Continuous-time Constant Acceleration Motion Model
$\mathbf{x}(t) = \begin{bmatrix} p(t) & v(t) & a(t) \end{bmatrix}^T$
$\begin{bmatrix} \dot{p}(t) \\ \dot{v}(t) \\ \dot{a}(t) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p(t) \\ v(t) \\ a(t) \end{bmatrix} + \tilde{\mathbf{q}}(t)$
Euler Method: $\mathbf{A}_{k-1} = \begin{bmatrix} 1 & T & 0 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}$
Exact Solution: $\mathbf{A}_{k-1} = \begin{bmatrix} 1 & T & T^2/2 \\ 0 & 1 & T \\ 0 & 0 & 1 \end{bmatrix}$
$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{a} \end{bmatrix} \qquad \mathbf{A}_{k-1} = \begin{bmatrix} \mathbf{I}_n & T\mathbf{I}_n & T^2/2\mathbf{I}_n \\ \mathbf{0}_n & \mathbf{I}_n & T\mathbf{I}_n \\ \mathbf{0}_n & \mathbf{0}_n & \mathbf{I}_n \end{bmatrix}$

Continuous-time Motion Noise
$\dot{\mathbf{x}}(t) = \tilde{\mathbf{A}}\mathbf{x}(t) + \tilde{\mathbf{q}}(t)$
We usually assume that $\tilde{\mathbf{q}}(t)$ is a white Gaussian noise process:
$\mathbb{E}[\tilde{\mathbf{q}}(t)] = 0 \qquad \text{Cov}(\tilde{\mathbf{q}}(\tau_1), \tilde{\mathbf{q}}(\tau_2)) = \delta(\tau_1 - \tau_2)\tilde{\mathbf{Q}}$

Wiener Process
$\mathbf{w}(t) = \int_0^t \tilde{\mathbf{q}}(\tau)\, d\tau$
$\mathbb{E}[\mathbf{w}(t)\mathbf{w}(t)^T] = \mathbb{E}[\int_0^t \tilde{\mathbf{q}}(\tau_1)\, d\tau_1 \int_0^t \tilde{\mathbf{q}}(\tau_2)^T\, d\tau_2] = \int_0^t \int_0^t \mathbb{E}[\tilde{\mathbf{q}}(\tau_1)\tilde{\mathbf{q}}(\tau_2)^T]\, d\tau_2\, d\tau_1 = \int_0^t 1\, d\tau_1 \tilde{\mathbf{Q}} = t\tilde{\mathbf{Q}}$

We seek a discrete time motion model: $\mathbf{x}_k = \mathbf{A}_{k-1}\mathbf{x}_{k-1} + \mathbf{q}_{k-1}$ where $\mathbf{q}_{k-1} \sim \mathcal{N}(0, \mathbf{Q}_{k-1})$ How can we select $\mathbf{Q}_{k-1}$?
$\mathbf{Q}_{k-1} = \text{Cov}(\mathbf{q}_{k-1}) = \text{Cov}(\mathbf{x}_k|\mathbf{x}_{k-1}) = \text{Cov}(\mathbf{x}(t+T)|\mathbf{x}(t))$

Exact Solution
$\mathbf{x}(t+T) = e^{\tilde{\mathbf{A}}T}\mathbf{x}(t) + \int_0^T e^{\tilde{\mathbf{A}}\tau}\tilde{\mathbf{q}}(\tau)\, d\tau$

$\mathbf{Q}_{k-1} = \text{Cov}(\mathbf{x}(t+T)|\mathbf{x}(t)) = \text{Cov}(\int_0^T e^{\tilde{\mathbf{A}}\tau}\tilde{\mathbf{q}}(\tau)\,d\tau) = \int_0^T e^{\tilde{\mathbf{A}}\tau}\tilde{\mathbf{Q}}e^{\tilde{\mathbf{A}}^T\tau}\,d\tau$

Modified Euler Method
$\dot{\mathbf{x}}(\tau) \approx \tilde{\mathbf{a}}(\mathbf{x}(t)) + \tilde{\mathbf{q}}(\tau)$
$\mathbf{x}(t+T) = \mathbf{x}(t) + T(\tilde{\mathbf{a}}(\mathbf{x}(t)) + \tilde{\mathbf{q}}(t))$
$\mathbf{x}(t+T) = \mathbf{x}(t) + \int_t^{t+T}\dot{\mathbf{x}}(\tau)\,d\tau = \mathbf{x}(t) + \int_t^{t+T}\tilde{\mathbf{a}}(\mathbf{x}(t))\,d\tau + \int_t^{t+T}\tilde{\mathbf{q}}(\tau)\,d\tau$
$\text{Cov}(\mathbf{x}(t+T)|\mathbf{x}(t)) \approx T\tilde{\mathbf{Q}}$

$\tilde{\mathbf{q}}(t) = \gamma\mathbf{q}_c(t)$
$\tilde{\mathbf{Q}} = \gamma\mathbf{Q}_c\gamma^T$
Constant velocity model: $\gamma = \begin{bmatrix}0\\1\end{bmatrix} \rightarrow \tilde{\mathbf{Q}} = \gamma\mathbf{Q}_c\gamma^T = \begin{bmatrix}0 & 0\\0 & Q_c\end{bmatrix}$

Continuous-time Coordinated Turn Model
$$\begin{bmatrix}\dot{x}(t)\\\dot{y}(t)\\\dot{v}(t)\\\dot{\phi}(t)\\\dot{\omega}(t)\end{bmatrix} = \begin{bmatrix}v(t)\cos(\phi(t))\\v(t)\sin(\phi(t))\\0\\\omega(t)\\0\end{bmatrix} + \begin{bmatrix}0&0\\0&0\\1&0\\0&0\\0&1\end{bmatrix}\begin{bmatrix}q_c^v(t)\\q_c^\omega(t)\end{bmatrix}$$

Modified Euler Method
$\mathbf{x}(t+T) \approx \mathbf{x}(t) + T\tilde{\mathbf{a}}(\mathbf{x}(t)) + \int_t^{t+T}\tilde{\mathbf{q}}(\tau)\,d\tau$
$$\begin{bmatrix}x_k\\y_k\\v_k\\\phi_k\\\omega_k\end{bmatrix} = \begin{bmatrix}x_{k-1} + Tv_{k-1}\cos(\phi_{k-1})\\y_{k-1} + Tv_{k-1}\sin(\phi_{k-1})\\v_{k-1}\\\phi_{k-1} + T\omega_{k-1}\\\omega_{k-1}\end{bmatrix} + \mathbf{q}_{k-1}$$

$$\mathbf{Q}_{k-1} = T\tilde{\mathbf{Q}} = T\gamma\begin{bmatrix}\sigma_v^2 & 0\\0 & \sigma_w^2\end{bmatrix}\gamma^T = \text{diag}(0,0,T\sigma_v^2,0,T\sigma_w^2)$$

Exact Solution
$$\begin{bmatrix}x_k\\y_k\\v_k\\\phi_k\\\omega_k\end{bmatrix} = \begin{bmatrix}x_{k-1} + \frac{2v_{k-1}}{\omega_{k-1}}\sin(\frac{\omega_{k-1}T}{2})\cos(\phi_{k-1} + \frac{\omega_{k-1}T}{2})\\y_{k-1} + \frac{2v_{k-1}}{\omega_{k-1}}\sin(\frac{\omega_{k-1}T}{2})\sin(\phi_{k-1} + \frac{\omega_{k-1}T}{2})\\v_{k-1}\\\phi_{k-1} + T\omega_{k-1}\\\omega_{k-1}\end{bmatrix} + \mathbf{q}_{k-1}$$

$\mathbf{Q}_{k-1} = \text{diag}(0,0,0,T\sigma_v^2,0,T\sigma_w^2)$

Direct Discretization of the Noise
In order to simplify the derivation of the noise covariance, it is common to assume a simpler distribution for $\tilde{\mathbf{q}}(t)$
Idea: assume that the noise $\tilde{\mathbf{q}}(t)$ is a piecewise constant between samples, i.e., that is it constant in every interval, $[0,T],[T,2T]$, etc.
Assume that $\tilde{\mathbf{q}}(t) \sim \mathcal{N}(0, \tilde{\mathbf{Q}}/T)$
$\int_0^T \tilde{\mathbf{q}}(t)\,dt = T\tilde{\mathbf{q}}(t) \qquad \mathbb{E}[T\tilde{\mathbf{q}}(t)\tilde{\mathbf{q}}(t)^T T] = T\tilde{\mathbf{Q}}$

Discretized Linearization
$\dot{\mathbf{x}}(\tau) \approx \tilde{\mathbf{a}}(\hat{\mathbf{x}}(t)) + \tilde{\mathbf{a}}'(\hat{\mathbf{x}}(t))(\mathbf{x}(\tau) - \hat{\mathbf{x}}(t)) + \tilde{\mathbf{q}}(\tau) \qquad \tau \in [t, t+T]$
Suppose that $\tilde{\mathbf{q}}(\tau)$ is also constant for $\tau \in [t, t+T]$ and $\dot{\mathbf{x}}(\tau) = \tilde{\mathbf{A}}\mathbf{x}(\tau) + \mathbf{b}$ for $\tau \in [t, t+T]$:
$$\mathbf{x}(t+T) = \exp(\tilde{\mathbf{A}}T)\mathbf{x}(t) + \left(\mathbf{I}T + \frac{\tilde{\mathbf{A}}T^2}{2} + \frac{\tilde{\mathbf{A}}^2T^3}{3!} + ...\right)\mathbf{b}$$

Measurement Models
$\mathbf{y}_k = h_k(\mathbf{x}_k) + \mathbf{r}_k \qquad \mathbf{r}_k \sim \mathcal{N}(0, \mathbf{R}_k)$

Global Navigation and Satellite System (GNSS)
State: $\mathbf{x}_k = \begin{bmatrix} p_k^1 & p_k^2 & v_k^1 & v_k^2 \end{bmatrix}^T$
Observation: noisy position in 2D
$$\mathbf{y}_k = \begin{bmatrix} p_k^1 \\ p_k^2 \end{bmatrix} + \begin{bmatrix} r_k^1 \\ r_k^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \mathbf{x}_k + \mathbf{r}_k$$

Gyroscope (yaw-rate sensor)
State: $\mathbf{x}_k = \begin{bmatrix} p_k^1 & p_k^2 & v_k & \phi_k & \omega_k \end{bmatrix}^T$
Observation: noisy observation of yaw rate
$\mathbf{y}_k = \omega_k + r_k = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}_k + r_k$

Radar Sensor
State: $\mathbf{x}_k = \begin{bmatrix} p_k^1 & p_k^2 & v_k^1 & v_k^2 \end{bmatrix}^T$
Observation: noisy observation of distance and angle
$$\mathbf{y}_k = \begin{bmatrix} \sqrt{(p_k^1)^2 + (p_k^2)^2} \\ \arctan(\frac{p_k^2}{p_k^1}) \end{bmatrix} + \begin{bmatrix} r_k^1 \\ r_k^2 \end{bmatrix}$$

Wheel Speed Encoders
State: $\mathbf{x}_k = \begin{bmatrix} p_k^1 & p_k^2 & v_k^1 & v_k^2 \end{bmatrix}^T$
Observation: noisy observation of speed
$y_k = \sqrt{(v_k^1)^2 + (v_k^2)^2} + r_k$

Sensor Calibration and Bias Filtering
Suppose our sensor has an offset, or a bias, $\mathbf{s}$ such that we observe:
$\mathbf{y}_k = h_k(\mathbf{x}_k) + \mathbf{s} + \mathbf{r}_k \qquad \mathbf{r}_k \sim \mathcal{N}(0, \mathbf{R}_k)$
If $\mathbf{s}$ is constant, it can usually be estimated from a set of training data. For low quality sensors, it is common that the bias drifts significantly over time.
Common solution: include $\mathbf{s}_k$ in the state vector and describe its motion as a random walk: $\mathbf{s}_k = \mathbf{s}_{k-1} + \mathbf{q}_{k-1}^s$

3 Nonlinear Filtering

State Space Models
$x_k = f_{k-1}(x_{k-1}) + q_{k-1} \qquad q_{k-1} \sim \mathcal{N}(0, Q_{k-1})$
$y_k = h_k(x_k) + r_k \qquad r_k \sim \mathcal{N}(0, R_k)$

General Filtering Solution
Prediction: $p(x_k|y_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|y_{1:k-1})\, dx_{k-1}$
Update: $p(x_k|y_{1:k}) = \dfrac{p(y_k|x_k)p(x_k|y_{1:k-1})}{\int p(y_k|x_k)p(x_k|y_{1:k-1})\, dx_k}$

3.1 Extended Kalman Filter (EKF)

Prediction: Linearize $f_{k-1}(x_{k-1})$ around $\hat{x}_{k-1|k-1}$ and use the Kalman filter prediction
Update: Linearize $h_k(x_k)$ around $\hat{x}_{k|k-1}$ and use the Kalman filter update

Taylor Series Expansion
$y \approx g(\hat{x}) + g'(\hat{x})(x - \hat{x})$
$[g'(x)]_{ij} = \dfrac{\partial g_i(x)}{\partial x_j}$
$\mathbb{E}[y] \approx g(\hat{x})$
$\mathrm{Cov}(y) \approx g'(\hat{x})Pg'(\hat{x})^T$

Prediction Step
$x_k \approx f(\hat{x}_{k-1|k-1}) + f'(\hat{x}_{k-1|k-1})(x_{k-1} - \hat{x}_{k-1|k-1}) + q_{k-1}$
$\hat{x}_{k|k-1} = f(\hat{x}_{k-1|k-1})$
$P_{k|k-1} = f'(\hat{x}_{k-1|k-1})P_{k-1|k-1}f'(\hat{x}_{k-1|k-1})^T + Q_{k-1}$

Update Step
$y_k \approx h(\hat{x}_{k|k-1}) + h'(\hat{x}_{k|k-1})(x_k - \hat{x}_{k|k-1}) + r_{k-1}$
$\hat{x}_{k|k} = \hat{x}_{k|k-1} + K_k(y_k - h(\hat{x}_{k|k-1}))$
$P_{k|k} = P_{k|k-1} - K_k S_k K^T$
$S_k = h'(\hat{x}_{k|k-1})P_{k|k-1}h'(\hat{x}_{k|k-1})^T + R_k$
$K_k = P_{k|k-1}h'(\hat{x}_{k|k-1})^T S_k^{-1}$

The EKF is also approximately LMMSE. As long as the systems are not too nonlinear, the EKF tends to perform well. Compared to other nonlinear filters, the EKF has the lowest computational complexity.

Iterated EKF (IEKF)
Once we have computed $\hat{x}_{k|k}$, we have a better guess about x_k. We can linearize $h(x_k)$ around that point and update $p(x_k|y_{1:k-1})$ using the new linearization and y_k

We obtain an iterative algorithm that we can run until convergence. Each update starts with the predicted density $\mathcal{N}(x_k; \hat{x}_{k|k-1}, P_{k|k-1})$ to avoid using y_k multiple times.

The IEKF solves $\hat{x}_{k|k} = \hat{x}_{k|k}^{MAP} = \arg\max_{x_k} p(x_k|y_{1:k})$ iteratively using a Gauss-Newton search. Pros: Usually performs very well when the measurement noise is small (the MAP estiamte is accurate then), Often

converges in very few iterations. Cons: Posterior pdfs are often rather skewed (the MAP estimate is far from the posterior mean), May diverge (more robust alternatives exist)

Gaussian Filtering
The KF, EKF, and IEKF are all examples of Gaussian filters. Both $p(x_{k-1}|y_{1:k=1})$ (prior) and $p(x_k|y_{1:k})$ (posterior) are Gaussian.

Assumed Density Filters
A wide range of Bayesian filters can be written on the following form: 1. Select a density parameterization $p(x;\theta)$, 2. Start from $p(x_{k-1}|y_{1:k-1}) \approx p(x_{k-1};\theta_{k-1|k-1})$ and find $\theta_{k|k}$ such that $p(x_k|y_{1:k}) \approx p(x_k;\theta_{k|k})$

Assumed Density	Filters
$p(x;\theta) = \mathcal{N}(x;\mu,P)$	KF, EKF, UKF, CKF, NN, GNN, PDA, JPDA
$p(x;\theta) = \sum_{i=1}^{N} w_i \mathcal{N}(x;\mu_i,P_i)$	IMM, MHT
$p(x;\theta) = \sum_{i=1}^{N} w^{(i)} \delta(x-x^{(i)})$	Particle filters
$p(x_l,x_n;\theta) = \sum_{i=1}^{N} w^{(i)} \delta(x_n - x_n^{(i)}) \mathcal{N}(x_l;\mu^{(i)},P^{(i)})$	Rao-Blackwellized particle filters

Gaussian Approximations by Moment Matching
Suppose that we are given a non-Gaussian density $p(x)$. The task is to find $\hat{x}$ and P such that $p(x) \approx \mathcal{N}(x;\hat{x},P)$.

One strategy is to match the moments of $p(x)$:
$\hat{x} = \mathbb{E}_{p(x)}[x] = \int x p(x)\, dx$
$P = \mathrm{Cov}_{p(x)}(x) = \int (x-\hat{x})(x-\hat{x})^T p(x)\, dx$
One can show that moment matching minimizes the Kullback-Leibler divergence: $KL(p(x)|\mathcal{N}(x;\hat{x},P)) = \int p(x) \log \frac{p(x)}{\mathcal{N}(x;\hat{x},P)}\, dx$

Prediction
Given Gaussian prior (updated posterior density): $\mathcal{N}(\hat{x}_{k-1|k-1}, P_{k-1|k-1})$
Prediction by moment matching:
$\hat{x}_{k|k-1} = \mathbb{E}[x_k|y_{1:k-1}] = \int f(x_{k-1}) \mathcal{N}(x_{k-1};\hat{x}_{k-1|k-1},P_{k-1|k-1})\, dx_{k-1}$
$P_{k|k-1} = \mathrm{Cov}(x_k|y_{1:k-1}) = \int (f(x_{k-1}) - \hat{x}_{k|k-1})(.)^T \mathcal{N}(x_{k-1};\hat{x}_{k-1|k-1},P_{k-1|k-1})\, dx_{k-1} + Q_{k-1}$

Update
Given $p(x_k|y_{1:k-1}) \approx \mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})$:
Approximate $p(x_k,y_k|y_{1:k-1})$ as a Gaussian using moment matching.
$$\begin{bmatrix} x_k \\ y_k \end{bmatrix} \Big| y_{1:k-1} \sim \mathcal{N}\Big(\begin{bmatrix} \hat{x}_{k|k-1} \\ \hat{y}_{k|k-1} \end{bmatrix}, \begin{bmatrix} P_{k|k-1} & P_{xy} \\ P_{yx} & S_k \end{bmatrix} \Big)$$
$\hat{y}_{k|k-1} = \mathbb{E}[y_k|y_{1:k-1}] = \mathbb{E}[h(x_k)|y_{1:k-1}] = \int h(x_k) \mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})\, dx_k$
$P_{xy} = \mathbb{E}[(x_k - \hat{x}_{k|k-1})(y_k - \hat{y}_{k|k-1})^T|y_{1:k-1}] = \int (x_k - \hat{x}_{k|k-1})(h(x_k) - \hat{y}_{k|k-1})^T \mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})\, dx_k$
$S_k = \mathrm{Cov}(y_k|y_{1:k-1}) = \mathrm{Cov}(h(x_k),y_{1:k-1}) + R_k = \int (h(x_k) - \hat{y}_{k|k-1})(.)^T \mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})\, dx_k + R_k$

$\hat{x}_{k|k} = \hat{x}_{k|k-1} + P_{xy} S_k^{-1} (y_k - \hat{y}_{k|k-1})$
$P_{k|k} = P_{k|k-1} - P_{xy} S_k^{-1} P_{xy}^T$
Need to approximate $\hat{y}_{k|k-1}$, S_k, and P_{xy}

3.2 Integrals in Gaussian Filtering

Finding integrals of the form: $\int g(x)\mathcal{N}(x;\hat{x},P)\,dx$

Prediction
$\hat{x}_{k|k-1} = \mathbb{E}[x_k|y_{1:k-1}] = \int f(x_{k-1})\mathcal{N}(x_{k-1};\hat{x}_{k-1|k-1},P_{k-1|k-1})\,dx_{k-1}$
$P_{k|k-1} = \mathrm{Cov}(x_k|y_{1:k-1}) = \int (f(x_{k-1})-\hat{x}_{k|k-1})(.)^T\mathcal{N}(x_{k-1};\hat{x}_{k-1|k-1},P_{k-1|k-1})\,dx_{k-1} + Q_{k-1}$
$\rightarrow p(x_k|y_{1:k-1}) \approx \mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})$

Update
$\hat{y}_{k|k-1} = \mathbb{E}[y_k|y_{1:k-1}] = \mathbb{E}[h(x_k)|y_{1:k-1}] = \int h(x_k)\mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})\,dx_k$
$P_{xy} = \mathbb{E}[(x_k-\hat{x}_{k|k-1})(y_k-\hat{y}_{k|k-1})^T|y_{1:k-1}] = \int (x_k-\hat{x}_{k|k-1})(h(x_k)-\hat{y}_{k|k-1})^T\mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})\,dx_k$
$S_k = \mathrm{Cov}(y_k|y_{1:k-1}) = \mathrm{Cov}(h(x_k),y_{1:k-1}) + R_k = \int (h(x_k)-\hat{y}_{k|k-1})(.)^T\mathcal{N}(x_k;\hat{x}_{k|k-1},P_{k|k-1})\,dx_k + R_k$
$\hat{x}_{k|k} = \hat{x}_{k|k-1} + P_{xy}S_k^{-1}(y_k-\hat{y}_{k|k-1})$
$P_{k|k} = P_{k|k-1} - P_{xy}S_k^{-1}P_{xy}^T$

Monte Carlo Method
Generate i.i.d. samples $x^1, x^2, ..., x^N$ from $p(x)$. Approximate $\int g(x)p(x)\,dx = \frac{1}{N}\sum_{i=1}^N g(x^i)$. The Monte Carlo is simple to use to perform Gaussian filtering, asymptotically exact, but seldomly used since it needs many samples.

Stochastic Decoupling
Suppose $P^{1/2}$ is a matrix such that $P^{1/2}(P^{1/2})^T = P$. We often use the Cholesky decomposition to find $P^{1/2}$ (MATLAB: $chol(P,'lower')$)
If $\xi \sim \mathcal{N}(0,I)$, then $x = \hat{x} + P^{1/2}\xi \sim \mathcal{N}(\hat{x},P)$
By changing the variable of integration from x to ξ: $\int g(x)\mathcal{N}(x;\hat{x},P)\,dx = \int g(\hat{x}+P^{1/2}\xi)\mathcal{N}(\xi;0,I)\,d\xi$. It is sufficient to be able to compute $\int g(\xi)\mathcal{N}(\xi;0,I)\,d\xi$

Sigma-point Methods
Approximate $\int g(x)\mathcal{N}(x;\hat{x},P)\,dx \approx \sum_{i=1}^N W^{(i)}g(\mathcal{X}^{(i)})$ where $\mathcal{X}^{(i)}$ are called σ-points and $W^{(i)}$ are weights. Compared to MC, points are selected deterministically
Many σ-point methods: unscented transform, cubature rule, Gauss-Hermite quadrature, Gaussian process quadrature, marginalized transform, etc.
Each used in Gaussian filtering and the filters are known as UKF, CKF, GHKF, GPKF, MKF, etc.

3.3 Sigma-point Methods

Suppose $x \sim \mathcal{N}(\hat{x},P)$, we can then approximate: $\mathbb{E}[g(x)] = \int g(\hat{x}+P^{1/2}\xi)\mathcal{N}(\xi;0,I)\,d\xi \approx \sum_{i=1}^N W_i g(\hat{x}+P^{1/2}\xi^{(i)}$ where $\xi^{(i)}$ are called σ-points.

Idea 1: it is easier to approximate a probability distribution than it is to approximate an arbitrary nonlinear function or transformation

Idea 2: if $y = g(x)$ and $x \sim \mathcal{N}(\hat{x},P)$, we can approximate:
$\mathbb{E}[y] \approx \mu_y = \sum_{i=0}^N W_i g(\mathcal{X}^{(i)})$
$\mathrm{Cov}(y) \approx \sum_{i=0}^N W_i(g(\mathcal{X}^{(i)})-\mu_y)(.)^T$

Unscented Transform (UT)
Form a set of $2n+1$ σ-points as follows:
$\mathcal{X}^{(0)} = \hat{x}$
$\mathcal{X}^{(i)} = \hat{x} + \sqrt{\frac{n}{1-W_0}} P_i^{1/2}$
$\mathcal{X}^{(i+n)} = \hat{x} - \sqrt{\frac{n}{1-W_0}} P_i^{1/2}$
$W_i = \dfrac{1-W_0}{2n}$
$P_i^{1/2}$ is the i^{th} column
Other versions with more design parameters.
If x is Gaussian, set $W_0 = 1 - n/3$

Cubature Rule
Form a set of $2n$ σ-points as follows:
$\mathcal{X}^{(i)} = \hat{x} + \sqrt{n} P_i^{1/2}$
$\mathcal{X}^{(i+n)} = \hat{x} - \sqrt{n} P_i^{1/2}$
$W_i = \dfrac{1}{2n}$
Special case of UT: $W_0 = 0$
No tuning parameters are no negative weights

Prediction in UKF
Form $2n+1$ σ-points:
$\mathcal{X}_{k-1}^{(0)} = \hat{x}_{k-1|k-1}$
$\mathcal{X}_{k-1}^{(i)} = \hat{x}_{k-1|k-1} + \sqrt{\frac{n}{1-W_0}} P_{k-1|k-1}^{1/2}$
$\mathcal{X}_{k-1}^{(i+n)} = \hat{x}_{k-1|k-1} - \sqrt{\frac{n}{1-W_0}} P_{k-1|k-1}^{1/2}$
$W_i = \frac{1-W_0}{2n}$
Compute the predicted moments: $\hat{x}_{k|k-1} \approx \sum_{i=0}^{2n} f(\mathcal{X}_{k-1}^{(i)}) W_i$
$P_{k|k-1} \approx Q_{k-1} + \sum_{i=0}^{2n} (f(\mathcal{X}_{k-1}^{(i)}) - \hat{x}_{k|k-1})(.)^T W_i$

Prediction in CKF
Form $2n$ σ-points:
$\mathcal{X}_{k-1}^{(i)} = \hat{x}_{k-1|k-1} + \sqrt{n} P_{k-1|k-1}^{1/2}$
$\mathcal{X}_{k-1}^{(i+n)} = \hat{x}_{k-1|k-1} - \sqrt{n} P_{k-1|k-1}^{1/2}$
$W_i = \frac{1}{2n}$
Compute the predicted moments: $\hat{x}_{k|k-1} \approx \sum_{i=1}^{2n} f(\mathcal{X}_{k-1}^{(i)}) W_i$
$P_{k|k-1} \approx Q_{k-1} + \sum_{i=1}^{2n} (f(\mathcal{X}_{k-1}^{(i)}) - \hat{x}_{k|k-1})(.)^T W_i$

Update in UKF
Form $2n+1$ σ-points:
$\mathcal{X}_{k}^{(0)} = \hat{x}_{k|k-1}$
$\mathcal{X}_{k}^{(i)} = \hat{x}_{k|k-1} + \sqrt{\frac{n}{1-W_0}} P_{k|k-1}^{1/2}$
$\mathcal{X}_{k}^{(i+n)} = \hat{x}_{k|k-1} - \sqrt{\frac{n}{1-W_0}} P_{k|k-1}^{1/2}$
$W_i = \frac{1-W_0}{2n}$
Compute the predicted moments:
$\hat{y}_{k|k-1} \approx \sum_{i=0}^{2n} h(\mathcal{X}_{k}^{(i)}) W_i$

$P_{xy} \approx \sum_{i=0}^{2n} (\mathcal{X}_k^{(i)} - \hat{x}_{k|k-1})(h(\mathcal{X}_k^{(i)}) - \hat{y}_{k|k-1})^T W_i$
$S_k \approx R_k + \sum_{i=0}^{2n} (h(\mathcal{X}_k^{(i)}) - \hat{y}_{k|k-1})(.)^T W_i$

Update in CKF
Form $2n$ σ-points:
$\mathcal{X}_k^{(i)} = \hat{x}_{k|k-1} + \sqrt{n} P_{k|k-1}^{1/2}$
$\mathcal{X}_k^{(i+n)} = \hat{x}_{k|k-1} - \sqrt{n} P_{k|k-1}^{1/2}$
$W_i = \frac{1}{2n}$
Compute the predicted moments:
$\hat{y}_{k|k-1} \approx \sum_{i=1}^{2n} h(\mathcal{X}_k^{(i)}) W_i$
$P_{xy} \approx \sum_{i=1}^{2n} (\mathcal{X}_k^{(i)} - \hat{x}_{k|k-1})(h(\mathcal{X}_k^{(i)}) - \hat{y}_{k|k-1})^T W_i$
$S_k \approx R_k + \sum_{i=1}^{2n} (h(\mathcal{X}_k^{(i)}) - \hat{y}_{k|k-1})(.)^T W_i$