Filtering in C++

Amro Al Baali

March 10, 2021

1 Why this document?

This document is provided to explain and clarify the code uploaded with it. The repository includes examples of implementing filters, usually Kalman fitlers, in C++. The filters will be mainly implemented on

1. a linear system,
2. a nonlinear Euclidean system, and
3. a non-Euclidean nonlinear system (usually defined by a Lie group).

2 The Kalman filter

2.1 The system

Consider the linear ordinary differential equation (ODE) describing a mass-spring-damper system

$$m\ddot{x}(t) + b\dot{x}(t) + kx(t) = u(t), \tag{1}$$

where m is the mass, b is the damping, k is the spring constant, and $u(t)$ is the forcing function. The system (1) can be written in state space form

$$\begin{aligned} \dot{\mathbf{x}} &= \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \mathbf{x} + \begin{bmatrix} 0 \\ \frac{1}{m} \end{bmatrix} u \qquad (2) \\ &= \boldsymbol{A}\mathbf{x} + \boldsymbol{B}u_t, \qquad (3) \end{aligned}$$

where

$$\mathbf{x} = \begin{bmatrix} x & \dot{x} \end{bmatrix}^{\mathsf{T}}, \tag{4}$$

and the time arguments (t) are dropped for brevity.

The disrete-time kinematic model is given by

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}u_{k-1}, \tag{5}$$

where the discrete-time system matrices $\mathbf{A}$ and $\mathbf{B}$ are computed using some discretization scheme. For the linear example above, the $\mathbf{A}$ matrix is given by

$$\mathbf{A} = \exp\left(\boldsymbol{A}T_k\right), \tag{6}$$

$$\mathbf{B} = \int_0^{T_k} \exp(\boldsymbol{A}\alpha)\mathrm{d}\alpha\boldsymbol{B}, \tag{7}$$

where T_k is the sampling period.

The matrix $\mathbf{B}$ can be approximated using forward Euler to get

$$\mathbf{B} \approx T_k\boldsymbol{B}. \tag{8}$$

2.2 Process model

The discrete-time process model[1] is used in the *prediction* step of the Kalman filter. It is given by

[1] Also referred to as the kinematic model, motion model, progression model, *e.t.c.*.

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} + \mathbf{L}\mathbf{w}_{k-1}, \tag{9}$$

where $\mathbf{x}_k \in \mathbb{R}^{n_x}$ is the state, $\mathbf{u}_k \in \mathbb{R}^{n_u}$ is the control input, and $\mathbf{w}_k \in \mathbb{R}^{n_w}$ where $\mathbf{w}_k \sim \mathcal{N}\left(\mathbf{0}, \mathbf{Q}_k\right)$ is the process noise and $\mathbf{Q}_k$ is the process noise covariance.

2.3 Measurement functions

The correction step requires a measurement model which is given by

$$\mathbf{y}_k = \mathbf{C}\mathbf{x}_k + \mathbf{M}\mathbf{n}_k, \tag{10}$$

where $\mathbf{y}_k \in \mathbb{R}^{n_y}$ and $\mathbf{n}_k \in \mathbb{R}^{n_n}$ where $\mathbf{n}_k \sim \mathcal{N}\left(\mathbf{0}, \mathbf{R}_k\right)$ is the measurement noise and $\mathbf{R}_k$ is the measurement noise covariance.

For the example presented, the measurement is a position measurememnt, so the measurement function is given by

$$y_k = \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x}_k + n_k \tag{11}$$

$$= \mathbf{C}\mathbf{x}_k + n_k. \tag{12}$$