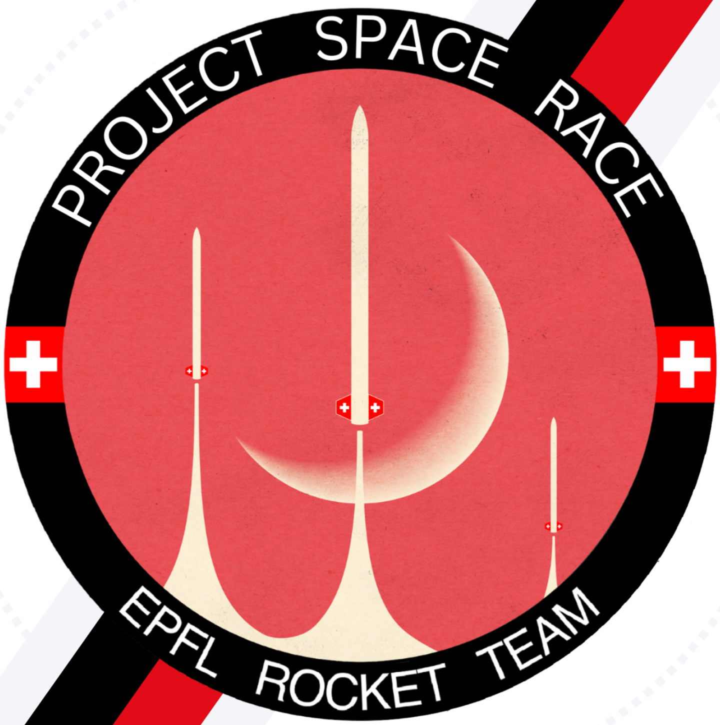
GENERIC ERT OVERLEAF TEMPLATE

NAME Surname - 33000
NAME Surname - 33001



Date: October 21, 2024

Doc ID: 202X_P_SS_GENERIC-ERT-OVERLEAF-TEMPLATE

ME-300 Course Name

Contents

Navigation Algorithms

Kalman Filtering

Abstract

Kalman filtering is a powerful and widely used technique for estimating the state of a dynamic system in the presence of noise. It provides a recursive solution to the problem of combining noisy measurements with a mathematical model of the system to obtain an optimal estimate of the true state. The filter operates by iteratively updating its estimate based on new measurements, taking into account the uncertainty associated with both the measurements and the system model.

Algorithm

The Kalman Filter (KF) work by using and combining input from different sensors in order to compute the **state** of the system. The term "state" refers to **all variables we want to know about the system at any given time**.

The KF is an iterative algorithm composed of **two steps** :

1. **Predict** : the KF **extrapolates** a predicted state based on the **current state** as well as **sensors measurements** of the system according to a given **dynamical system**.
2. **Update** : the KF gathers **observations** from various sensors to **correct** the predicted state. By **comparing the predicted state with the actual measurements**, the algorithm can adjust the predicted state and make it more precise.

Mathematical Description

Prediction

Symbol	Name	Description	Size
$\boldsymbol{x}$	state vector		$n_x * 1$
$\boldsymbol{P}$	covariance matrix	**uncertainty** associated with the state	$n_x * n_x$
$\boldsymbol{u}$	control vector	prediction data	$n_u * 1$
$\boldsymbol{F}$	state transition matrix	**influence of the state** on the dynamical system	$n_x * n_x$
$\boldsymbol{G}$	control matrix	**influence of the controls** on the dynamical system	$n_x * n_u$

Symbol	Name	Description	Size
$\boldsymbol{Q}$	covariace noise matrix	**noise** of the predict step	$n_x * n_x$

State extrapolation is performed as follows :

$$\boldsymbol{x}_{n+1,n} = \boldsymbol{F}\boldsymbol{x}_n + \boldsymbol{G}\boldsymbol{u}_n$$

Covariance extrapolation is performed as follows :

$$\boldsymbol{P}_{n+1,n} = \boldsymbol{F}\boldsymbol{P}_n\boldsymbol{F}^T + \boldsymbol{Q}$$

$\boldsymbol{x}_{n+1,n}$ denotes the prediction based on the current state at time .

A good example would be the usual equation of motion for a linear movement : $p(tp = p_0 + v_0 t + \frac{1}{2}at^2$ (p denotes the position). The only sensor we'll be using is an accelerometer and we want to know both position and velocity at any given time. For the sake of simplicity, we choose to ignore noise and uncertainty.

We first define the state vector and control vector :

$$\boldsymbol{x} = [p, v]^T \boldsymbol{u} = a$$

We also need the system to be written such a way that the state at time only depends on the state at time . This may be achieved by writing the following :

$$p_{n+1,n} = p_n + v_n * \Delta t v_{n+1,n} = v_n + a_n * \Delta t$$

We thus get :

$$\begin{bmatrix} p_{n+1,n} \\ v_{n+1,n} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_n \\ v_n \end{bmatrix} \tag{1}$$

-

$$\begin{bmatrix} 0 \\ \Delta t \end{bmatrix}$$

a_n$$ ### Update | Symbol | Name | Description | Size | | — | — | — | — | | $\boldsymbol{z}$ | observation vector | | $n_z * 1$ | | $\boldsymbol{H}$| observation matrix | state influence on measurements | $n_z * n_x$| | $\boldsymbol{R}$| measurement covariance matrix | **noise** of the update step |$n_x * n_x$| | $\boldsymbol{K}$| **Kalman Gain**| determines the **relative contributions of each observation** |$n_x * n_z$|

The **Kalman Gain** is computed as follows:

$$\boldsymbol{K}_n = \boldsymbol{P}_{n+1,n}\boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{P}_{n+1,n}\boldsymbol{H}^T + \boldsymbol{R}_n)^{-1}$$

State update is then performed :

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_{n+1,n} + \boldsymbol{K}_n(\boldsymbol{z}_n - \boldsymbol{H}\boldsymbol{x}_{n+1,n})$$

Finally, the **Covariance update** is:

$$\boldsymbol{P}_{n+1} = (\boldsymbol{I} - \boldsymbol{K}_n\boldsymbol{H})\boldsymbol{P}_{n+1,n}$$

Further resources may be found here. Note that while the equations may slightly differ from what is shown here, both are equivalent.

Extended KF (EKF)

Motivation

As the most astute readers may already have noticed, the **“classical” KF only works with linear transformations**, and unfortunately, most models cannot necessarily be written as linear transformations.

There is, however, a solution : the most exalted **Taylor Series** ! Indeed, let $f : \mathbb{R}^n \to \mathbb{R}^m$ s.t $f \in \mathcal{C}^1$, then for any $\boldsymbol{a} \in \mathbb{R}^n$:

$$f(\boldsymbol{x}) = f(\boldsymbol{a}) + \boldsymbol{J}_f(\boldsymbol{a})(\boldsymbol{x} - \boldsymbol{a}) + \mathcal{O}(\boldsymbol{x})$$

The **Extended Kalman Filter** provides an approach based on this results.

Mathematical Description

Prediction

We first define the following relationship :

$$\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w})$$

Where $f, \boldsymbol{w}$ respectiveley denote the **state transition function** and the **process noise** for this step (similar to $\boldsymbol{Q}$ in the regular KF). From this alone, we can derive everything we need for the predict step.

$$\begin{aligned} \boldsymbol{F} &= \partial_{\boldsymbol{x}} f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w}) \\ \boldsymbol{G} &= \partial_{\boldsymbol{u}} f(\boldsymbol{x}, \boldsymbol{u}, \boldsymbol{w}) \end{aligned}$$

We’ll also define $\boldsymbol{Q} = \boldsymbol{G}\, diag(\boldsymbol{w})\, \boldsymbol{G}^T + \boldsymbol{Q}_s$, where Q_s is constant. Giving us an expression for $\boldsymbol{P}$

$$\dot{\boldsymbol{P}} = \boldsymbol{F}\boldsymbol{P} + \boldsymbol{P}\boldsymbol{F}^T + \boldsymbol{Q}$$

The **State extrapolation** thus becomes:

$$\begin{bmatrix} \boldsymbol{x}_{n+1,n} \\ \boldsymbol{P}_{n+1,n} \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}_n \\ \boldsymbol{P}_n \end{bmatrix} + \int_{\Delta t} \begin{bmatrix} \dot{\boldsymbol{x}}_n \\ \dot{\boldsymbol{P}}_n \end{bmatrix} dt$$

Update

The trick here is very much the same

$$\begin{aligned} \hat{\boldsymbol{z}} &= h(\boldsymbol{x}) \\ \boldsymbol{H} &= \partial_{\boldsymbol{x}} h(\boldsymbol{x}) \end{aligned}$$

The equations for **state update are then the exact same as the “classical” KF’s**

Other filters

Other powerful filtering techniques based on the Kalman filter exist. Some expanding on the inner workings of the EKF and some relying on completely different approaches.

To illustrate this, let us dive into the Unscented Kalman filter :

Unscented KF (UKF)

The approach is fundamentaly differnet from the EKF. Instead of linearizing the state transition and observation functions. It approximates the non-linear functions using a set of carefully chosen sigma points. These sigma points are chosen to capture the mean and covariance of the state distribution. The number of sigma points is typically $2n_x + 1$. Applying such non-linear functions to the sigma points then finding mean and covariance is called an **Unscented transform**.

Predict

We only consider a single pass through the predict step, but note that it works for every iteration. We refer to the i-th sigma point as X_i.

$$\bar{X} = \sum_{i=0}^{2n_x} W_i^m f(X_i) \bar{P} = \sum_{i=0}^{2n_x} W_i^c [f(X_i) - \bar{X}][f(X_i) - \bar{X}]^T + Q$$

Where W_i^m and W_i^c are a set of carefuly tuned weights.

Update

In a similar way. We now use $f(X_i)$ as the i-th sigma point.

$$\bar{Z} = \sum_{i=0}^{2n} W_i^m f(X_i) \bar{S} = \sum_{i=0}^{2n} W_i^c (f(X_i) - \bar{Z})(f(X_i) - \bar{Z})^T + RC_{xz} = \sum_{i=0}^{2n} W_i^c (X_i - \bar{X})(f(X_i) - \bar{Z})^T$$

Once again, W_i^m and W_i^c are a set of carefuly tuned weights.

The Kalman gain then becomes $K = C_{xz}\bar{S}^{-1}$ and the update can finally be written :

$$X' = \bar{X} + K(Z - \bar{Z})P' = \bar{P} - K\bar{S}K^T$$

Perks & Drawbacks

Compared to the EKF, the UKF boasts a more “cautious” nature. Indeed, comparatively speaking, the EKF tends to be overconfident in its approximations. Furthermore, the choice of W_i^m and W_i^c enables the agressiveness of the filter to be precisely controlled.

On the downside, the UKF tens to be computationnaly much less efficient, due in part to the way the different weights are computed. Furthermore, EKF’s are usually much easier to implement.

Calibration

Calibration is crucial for the Extended Kalman Filter (EKF) for several key reasons: - Enhancing Accuracy: Calibration ensures the precision of sensor inputs and model parameters, leading to more reliable state estimation in dynamic systems. - Correcting Sensor Flaws: It identifies and compensates for systematic sensor errors and biases, enhancing the EKF's accuracy. - Noise Management: Calibration accurately characterizes process and measurement noise, vital for the EKF's performance. - Model Refinement: It validates and improves the mathematical model of the observed system, aligning EKF predictions with real-world behavior. - Maintaining Adaptability: Regular calibration adapts to changing sensor characteristics in dynamic environments, keeping the EKF robust. - Preventing Divergence: Proper calibration avoids incorrect state estimates and the potential for filter divergence, ensuring EKF stability.

For our system, the calibration has been done on the accelerometer, gyroscope, barometer, and magnetometer data.

Accelerometer and gyroscope calibration

To ensure accurate calibration of accelerometer and gyroscope, several rotations are executed in **six distinct orientations**.

This process was designed to separate the sensor outputs into two segments: pre-rotation and post-rotation. This separation is crucial for eliminating the transient phase that could skew the data.

Following this, a rotation transformation is applied on the post-rotation dataset to align the data within a consistent reference system. Let $\boldsymbol{R}_{nb}$ be the rotation from the body-frame to the NED intertial frame

$$\boldsymbol{x}_{post} = R_{nb}\boldsymbol{x}_{post}$$

The final step involved calculating the bias and variance for both accelerometer and gyroscope readings. This is achieved by averaging the mean and variances, both before (a priori) and after (a posteriori) the calibration procedure, as described below:

$$\boldsymbol{b} = \frac{\bar{x}_{pre} + \bar{x}_{post}}{2}$$

$$\boldsymbol{\sigma} = \frac{\sigma^2_{x,pre} + \sigma^2_{x,post}}{4}$$

Magnetometer calibration

Calibrating a magnetometer **through ellipsoid fitting** is a sophisticated and effective method. It **corrects for hard iron and soft iron distortions**, which are common issues affecting magnetometer accuracy. The process involves capturing magnetometer readings in various orientations, fitting these readings to an ellipsoid, and then applying corrections based on this fit. Here's a more detailed algorithm for ellipsoid fitting in magnetometer calibration:

Step 1: Data Collection

Rotate the magnetometer through a **wide range of orientations** to gather a comprehensive set of measurements. Ensure a **uniform distribution across all directions** to adequately sample the 3D space (i.e. be smooth yet swift).

Step 2: Ellipsoid Fitting

Use the collected data to **fit an ellipsoid**. This can be done using a least squares fitting algorithm or other numerical methods.

The fitted ellipsoid can be represented by the following equation:

$$\frac{(xx_c)^2}{a^2} + \frac{(yy_c)^2}{b^2} + \frac{(zz_c)^2}{c^2} = 1$$

with (x_c, y_c, z_c) are the ellipsoid's center coordinates, and (a, b, c) are the radii along the X, Y, and Z axes.

Step 3: Calculate Correction Parameters

The bias is defined as as the ellipsoid center: $b = [X_c, Y_c, Z_c]^T$.

Each axis is *scaled* by a certain amount (hence why an ellispoid and not a sphere):

$$K = \begin{bmatrix} \frac{1}{a} & 0 & 0 \\ 0 & \frac{1}{b} & 0 \\ 0 & 0 & \frac{1}{c} \end{bmatrix}$$

If available, also keep the rotation information yielded by the fitting for later use.

Step 4: Apply Corrections

For each axis, the calibrated measurement is therefore:

$$\hat{\boldsymbol{m}} = K(\boldsymbol{m_{uncal}} - \boldsymbol{b})$$

If rotation corrections are necessary, apply the appropriate rotation matrix to the corrected data.

Step 5: Verification

After applying the corrections, the magnetometer should be tested again in various orientations. The corrected readings should form a sphere, indicating that distortions have been successfully removed. Verifying the accuracy against known reference values or in a controlled environment forms the best practice.

Resources