Navigation and Intelligent Vehicles

Lab session 3: the Extended Kalman Filter

In this session, you will learn

a) How to deal with non-linear state and/or measurement equations
b) How to implement an Extended Kalman Filter for radar tracking purposes

As described previously, the Kalman Filter (KF) addresses the general problem of trying to estimate the state $x \in \mathbb{R}^n$ of a discrete-time controlled process that is governed by a **linear** stochastic difference equation. But what happens if the process to be estimated and/or the measurement relationship to the process are/is **non-linear**?

The solution is to linearize the function about the current mean and covariance, and a filter that works like this is called an Extended Kalman Filter (EKF). In something akin to a Taylor series, we can linearize the estimation around the current estimate using the partial derivatives of the process and measurement functions to compute estimates even in the face of non-linear relationships. To do so, we must begin by modifying some of the previously presented equations.

Our process has again a state vector $x \in \mathbb{R}^n$ and a measurement $z \in \mathbb{R}^m$, only now the process and measurement are governed by **non-linear** stochastic difference equations:

$$x(k+1) = f\big(x(k), u(k), w(k)\big) \tag{1}$$

$$z(k) = h\big(x(k), v(k)\big) \tag{2}$$

The random variables $w(k)$ and $v(k)$ again represent the process and measurement noise.

The Extended KF equations

Basically nothing much changes for the EKF, compared to the KF. It's important to note that the fundamental flaw of the EKF is that the distributions of the various random variables are no longer normal[1] after undergoing their respective non-linear transformations. The EKF is simply an ad-hoc state estimator that approximates the optimality of Bayes' rule by linearization.

Time update equations

The time update equations are given by:

$$x^-(k) = f(\hat{x}(k-1), u(k-1), 0) \tag{3}$$

$$P^-(k) = A_k P(k-1) A_k^T + G_k Q(k-1) G_k^T \tag{4}$$

As with the basic discrete KF, the time update equations project the state and covariance estimates from the previous time step $k-1$ to the current time step k. The function f is from (1), A_k and G_k are the process Jacobians at step k and $Q(k)$ is the process noise covariance equation at step k. Notice the subscript k that was added to the Jacobians A and G, which reinforces the notion that they are different at (and therefore must be recomputed at) each time step. They can be calculated as follows:

$$A_{[i,j]} = \frac{\partial f_{[i]}}{\partial x_{[j]}}(\hat{x}(k-1), u(k-1), 0) \qquad G_{[i,j]} = \frac{\partial f_{[i]}}{\partial w_{[j]}}(\hat{x}(k-1), u(k-1), 0)$$

Thus, A is the Jacobian matrix of partial derivatives of f with respect to the state x, and G is the Jacobian matrix of partial derivatives of f with respect to the process noise w.

[1] Remember, this was a condition that guaranteed the optimality of the solution of the KF.

Measurement update equations

As with the basic discrete Kalman filter, the measurement update equations correct the state and covariance estimates with the measurement z_k. The measurement update equations are given by:

$$K(k) = P^-(k)C_k^T\left[C_k P^-(k)C_k^T + H_k R(k) H_k^T\right]^{-1} \tag{5}$$

$$\hat{x}(k) = \hat{x}^-(k) + K(k)[z(k) - h(\hat{x}^-(k), 0)] \tag{6}$$

$$P(k) = [I - K(k)C_k]P^-(k) \tag{7}$$

I is the identity matrix, h is the function in (2), $R(k)$ is the measurement noise covariance at step k. The Jacobians C_k and H_k at time step k are given by:

$$C_{[i,j]} = \frac{\partial h_{[i]}}{\partial x_{[j]}}(\hat{x}^-(k), 0) \qquad H_{[i,j]} = \frac{\partial h_{[i]}}{\partial v_{[j]}}(\hat{x}^-(k), 0)$$

Thus, C is the Jacobian matrix of partial derivatives of h with respect to the state x, and H is the Jacobian matrix of partial derivatives of h with respect to the measurement noise v.

Lab assignment: radar tracking

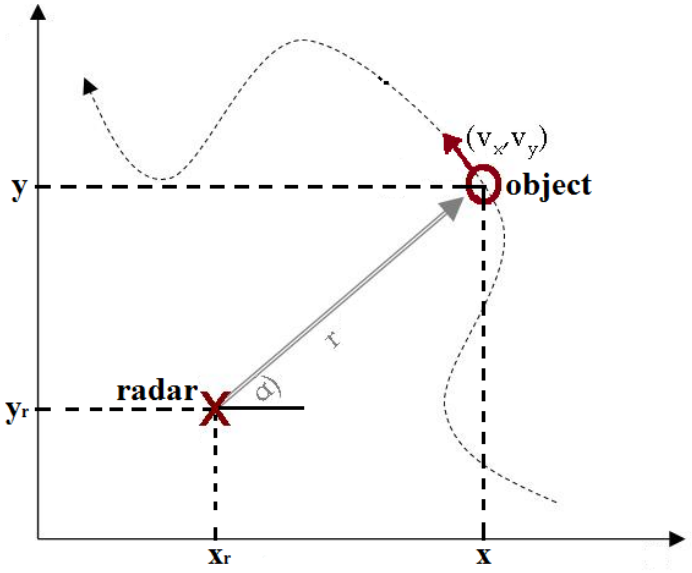

Figure 1: The radar scans the environment around its antenna in a circular way.

Let's assume a rotation speed of $\omega_r = 4\pi$ rad/s. For slow moving objects that are being tracked, it's thus safe to assume a measurement update every $T_r = 0.5$ s. The measurement vector z consists of two components, the distance to the object and the angle under which it is observed:

$$z(k) = [r(k) \quad \alpha(k)]^T \tag{8}$$

For the tracked object, we suggest to use a four dimensional state vector, containing the position and velocity for the two dimensional space in which the object moves:

$$x(k) = [p_x(k) \quad v_x(k) \quad p_y(k) \quad v_y(k)]^T \tag{9}$$

Thus we need to extend the model we used before to 2D motion. Also, our measurements are a nonlinear function of the states with some white noise, where h gives the coordinate transformation from Cartesian to polar: $z(k) = h\big(x(k), v(k)\big)$.

Discrete-time 2D model

Assuming a correlation between the X and Y coordinate implies that the tracked object would constantly be involved in the action of turning. From there it comes that this 2D filter consists of two independent 1D filters. For computational reasons, it's better to keep them separated: smaller matrices to compute.

$$x(k+1) = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} x(k) + w(k) \qquad w(k) \sim N([0 \quad 0 \quad 0 \quad 0]^T, Q) \tag{10}$$

With a state vector $x(k) = [p_x(k) \quad v_x(k) \quad p_y(k) \quad v_y(k)]^T$ and noise correlation for the white noise acceleration model in 2D:

$$Q = \sigma_a^2 \begin{bmatrix} T^3/3 & T^2/2 & 0 & 0 \\ T^2/2 & T & 0 & 0 \\ 0 & 0 & T^3/3 & T^2/2 \\ 0 & 0 & T^2/2 & T \end{bmatrix}$$

From the above, you should remember that it doesn't matter whether you are tracking a 1D movement or a 2D movement from the process model point of view. Differences can still exist in the measurement equations.

Analysis and report tasks

Here in Lab 3 the you can use the provided file '`generate_data_2D.m`'. You can visualise the generated data by setting the variable `plot_grafs = true` in the 13th line.

Derive a piecewise constant velocity motion model in 2D for tracking as shown in the example for white noise acceleration model. In this case, the process equation stays linear, with A and G similar to the matrices in Lab 2, but extended to 2D.

In contrast to the time update equations (3)-(4), the measurement update equations (5)-(7) of the EKF are nonlinear, because h is the nonlinear relationship between the polar and Cartesian coordinates. Note that we assume the noise to be additive, entering the equations in a linear way.

Make a short (maximum six pages) report in which you **discuss** and illustrate how the EKF implementation for radar tracking works. Focus on a clear and correct implementation of EKF. Describe how the data, the noise and the initialization impact on the filter performance. Elaborate on the indicators NIS and NEES and how they support your interpretation of the results. Check noise parameters, matrices form and dimensions, and parameters initialization.

References

- Yaakov Bar-Shalom, Xiao-Rong Li, (1993). Estimation and Tracking, Principles, Techniques, and Software, Artech House Publishing, ISBN 0-89006-643-4
- Kailath, T., Sayed, A. H., and Hassibi, B. (2000). Linear estimation. Prentice-Hall.
- Welch G., and Bishop, G. (2001). An Introduction to the Kalman Filter. SIGGRAPH 2001.
- Schön, T., and Lindsten, F. (2017). Learning of dynamical systems - particle filters and Markov chain methods. Lecture notes, Appendix B. [Online]. Available: http://www.it.uu.se/research/systems_and_control/education/2017/smc/SMC2017.pdf