Kalman Filter Equations for a (2,0) mobile robot localization with a Lidar and odometry

Xavier DAUPTAIN
REDEV Project

General Extended Kalman Filter Equations

We consider a non-linear discrete system in state-space form. The equations for a generalized extended Kalman filter are recalled below for conveniance. X is the state vector, f is the state function and U is the system Y will be the measurement vector or system output. Q_v will be the covariance matrix of a variable v.

Evolution Equation

$$X_{k+1} = f(X_k, {U_k}^*) + \alpha_k \tag{1}$$

With ${U_k}^* = U_k + \beta_k$ (with noisy input)

Observation Equation

$$Y_k = g(X_k) + \gamma_k \tag{2}$$

Where α and γ are additives noises on the evolution model and the measurement.

Prediction Phase

$$\begin{cases} \widehat{X}_{k+1} = f(\widehat{X}_{k/k}, {U_k}^*) \\ P_{k+1/k} = A_k \cdot P_{k/k} \cdot {A_k}^T + B_k \cdot Q_\beta \cdot {B_k}^T + Q_\alpha \end{cases} \tag{3}$$

With $A_k = \frac{\partial f}{\partial X}(\widehat{X}_{k/k}, {U_k}^*)$, $B_k = \frac{\partial f}{\partial U}(\widehat{X}_{k/k}, {U_k}^*)$

Estimation Phase

$$\begin{cases} \widehat{X}_{k+1/k+1} = \widehat{X}_{k+1/k} + K_k[Y_k - g(\widehat{X}_{k+1/k})] \\ P_{k+1/k+1} = (I - K_k \cdot C_k) \cdot P_{k+1/k} \end{cases} \tag{4}$$

With

$$\begin{cases} C_k = \frac{\partial g}{\partial X}(\widehat{X}_{k+1/k}) \\ K_k = P_{k+1/k} \cdot {C_k}^T (C_k \cdot P_{k+1/k} \cdot {C_k}^T + Q_\gamma)^{-1} \end{cases} \tag{5}$$

Filter Implementation: Prediction Phase

An "odometry()" routine performs the prediction phase. It is called inside an infinite loop in the embedded code. The input U is : $U = \begin{pmatrix} \delta_{dist} \\ \delta_\theta \end{pmatrix}$,
where δ_{dist} and δ_θ are the distance and angle increment of the robot since the last call of the function. This variable is computed only with the data from the proprioceptive sensors (wheels encoders) of the robot and the geometry (wheel radius,etc...) Thus, we have for the state prediction:

$$\begin{cases} x = x + \delta_{dist} * cos(\theta) \\ y = y + \delta_{dist} * sin(\theta) \\ \theta = \theta + \delta_\theta \\ X = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} \end{cases} \tag{6}$$

And for the error propagation:

$$\begin{cases} A = \begin{pmatrix} 1 & 0 & -\delta_{dist} * sin(\theta) \\ 0 & 1 & \delta_{dist} * cos(\theta) \\ 0 & 0 & 1 \end{pmatrix} \\ B = \begin{pmatrix} cos(\theta) & 0 \\ sin(\theta) & 0 \\ 0 & 1 \end{pmatrix} \\ P = A \cdot P \cdot A^T + B \cdot Q_\beta \cdot B^T \end{cases} \tag{7}$$

X and P are initialized inside a setup() function : we assume we know the initial position of the robot with a degre of certainty.

$$P_{init} = \begin{pmatrix} \sigma_x & 0 & 0 \\ 0 & \sigma_y & 0 \\ 0 & 0 & \sigma_\theta \end{pmatrix}$$

Q_β is computed at the beginning given the geometry and the quantization of the wheels rotation measurements, inherent in the use of incremental encoders: If Rcd is the right wheel radius , Rcg the left wheel radius and e the distance between the center of each wheel :

$$\zeta = \begin{pmatrix} Rcd/2 & Rcg/2 \\ -Rcd/e & Rcg/e \end{pmatrix}$$
$$Q_{wheels} = \begin{pmatrix} {\sigma_{wheels}}^2 & 0 \\ 0 & {\sigma_{wheels}}^2 \end{pmatrix}$$
$$Q_\beta = \zeta \cdot Q_{wheels} \cdot \zeta^T$$

Filter Implementation: Estimation Phase

The estimation phase falls into two part : First the estimated position of the beacons in the lidar frame is computed on given the observation equation, with the Jacobian matrix C and th Kalman gain K. This process is triggered on request of the embedded computer when raw data from the lidar is available. The estimated position of the beacons is sent with the covariance matrix of $\sigma_{maha} = Y - \widehat{Y}$ (the mahalanobis covariance matrix). Then the embedded computer extract three clusters from the point cloud , related to the three beacons, and send back to the microcontroller the position of the beacons. We assume the time needed for this process is negligible, so that measurements are available as soon as the estimated pose is computed. In the second part, the pose X and its covariance are refreshed. The observation equation is quite straightforward : if $\rho_i = \begin{pmatrix} u_i & v_i & 1 \end{pmatrix}^T$ are the homogeneous coordinates of the beacon i in the map frame then we can compute its coordinate into the LIDAR frame given the current pose of the robot.

$$\widehat{Y} = sT0 \cdot \rho_i \tag{8}$$

With $sT0 = \begin{pmatrix} cos(\theta) & sin(\theta) & -decx_{lidar} - xcos(\theta) - ysin(\theta) \\ -sin(\theta) & cos(\theta) & xsin(\theta) - ycos(\theta) - decy_{lidar} \\ 0 & 0 & 1 \end{pmatrix}$

Because $0Ts = \begin{pmatrix} cos(\theta) & -sin(\theta) & x + decx_{lidar} \cdot cos(\theta) - decy_{lidar} \cdot sin(\theta) \\ sin(\theta) & cos(\theta) & y + decy_{lidar} \cdot cos(\theta) + decx_{lidar} \cdot sin(\theta) \\ 0 & 0 & 1 \end{pmatrix}$

Then, the jacobian matrix is :

$$C = \begin{pmatrix} -cos(\theta) & -sin(\theta) & (v_i - y)cos(\theta) + (x - u_i)sin(\theta) \\ sin(\theta) & -cos(\theta) & (x - u_i)cos(\theta) + (y - v_i)sin(\theta) \end{pmatrix} \tag{9}$$

And

$$\sigma_{maha} = \left(C \cdot P \cdot C^T + Q_\gamma\right)^{-1} \tag{10}$$

$$K = P \cdot C^T \cdot \sigma_{maha} \tag{11}$$

When the measurement has been sent we finally refresh the pose and its covariance matrix as follow :

$$\begin{cases} X = X + K \cdot (Y - \widehat{Y}) \\ P = (I_3 - K \cdot C)P \end{cases} \tag{12}$$

Q_{gamma} can be initialized in the setup() function as a diagonal 2x2 matrix. The coefficients must be estimated given the lidar precision and the motion of the robot during a lidar scan.