Kalman Filter

[Introduction to Kalman Filter](http://www.roboticsproceedings.org/rss01/p38.pdf)
http://www.roboticsproceedings.org/rss01/p38.pdf

Time Update:

$$\hat{X}_{n+1} = F_n \hat{X}_n + B_n U_n$$
$$P_{n+1} = F_n P_n F_n^\intercal + G_n Q_n G_n^\intercal$$

Measurement Update:

$$K_n = P_{n-1} H^\intercal (H P_{n-1} H^\intercal + R_n)^{-1}$$
$$X_n = X_{n-1} + K_n (z_n - H \hat{X}_{n-1})$$
$$P_n = (I - K_n H) P_{n-1} (I - K_n H)^\intercal + K_n R_n K_n^\intercal$$

Noise Update:

$$R_n = \frac{1}{N} \sum_{n=1}^{N} ((\hat{X}_n - F\hat{X}_n)(\hat{X}_n - F\hat{X}_n)^\intercal)$$
$$Q_n = \frac{1}{N+1} \sum_{n=1}^{N} ((\hat{X}_n - F\hat{X}_n)(\hat{X}_n - F\hat{X}_n)^\intercal)$$

Accel filtering (acc_node)

https://scholarworks.calstate.edu/downloads/dv13zt241
https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-696.pdf

State extrapolation equation:

$$\begin{bmatrix} x_{n+1} \\ \dot{x}_{n+1} \\ \ddot{x}_{n+1} \\ y_{n+1} \\ \dot{y}_{n+1} \\ \ddot{y}_{n+1} \\ z_{n+1} \\ \dot{z}_{n+1} \\ \ddot{z}_{n+1} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & \Delta t & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \Delta t & \frac{\Delta t^2}{2} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t & \frac{\Delta t^2}{2} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ \dot{x}_n \\ \ddot{x}_n \\ y_n \\ \dot{y}_n \\ \ddot{y}_n \\ z_n \\ \dot{z}_n \\ \ddot{z}_n \end{bmatrix}$$

Measurement equation:

$$\begin{bmatrix} \ddot{x} \\ \ddot{y} \\ \ddot{z} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_n \\ \dot{x}_n \\ \ddot{x}_n \\ y_n \\ \dot{y}_n \\ \ddot{y}_n \\ z_n \\ \dot{z}_n \\ \ddot{z}_n \end{bmatrix}$$

Gyro filtering (gyro_node)

State extrapolation equation:

$$\begin{bmatrix} \theta_{x_{n+1}} \\ \omega_{x_{n+1}} \\ \theta_{y_{n+1}} \\ \omega_{y_{n+1}} \\ \theta_{z_{n+1}} \\ \omega_{z_{n+1}} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta t & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta t \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_{x_n} \\ \omega_{x_n} \\ \theta_{y_n} \\ \omega_{y_n} \\ \theta_{z_n} \\ \omega_{z_n} \end{bmatrix}$$

Measurement equation:

$$\begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_{x_n} \\ \omega_{x_n} \\ \theta_{y_n} \\ \omega_{y_n} \\ \theta_{z_n} \\ \omega_{z_n} \end{bmatrix}$$

IMU array Kinematics (cmdvel_acc_node)

Owais Talaat Waheed, Ibrahim (Abe) M. Elfadel, “FPGA Sensor Fusion System Design for IMU Arrays”

State extrapolation equation:

$$\begin{bmatrix} \ddot{x}(n+1) \\ \ddot{y}(n+1) \\ \ddot{z}(n+1) \\ \dot{\omega}_x(n+1) \\ \dot{\omega}_y(n+1) \\ \dot{\omega}_z(n+1) \\ \omega_x^2(n+1) \\ \omega_y^2(n+1) \\ \omega_z^2(n+1) \\ \omega_x\omega_y(n+1) \\ \omega_y\omega_z(n+1) \\ \omega_z\omega_x(n+1) \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \ddot{x}(n) \\ \ddot{y}(n) \\ \ddot{z}(n) \\ \dot{\omega}_x(n) \\ \dot{\omega}_y(n) \\ \dot{\omega}_z(n) \\ \omega_x^2(n) \\ \omega_y^2(n) \\ \omega_z^2(n) \\ \omega_x\omega_y(n) \\ \omega_y\omega_z(n) \\ \omega_z\omega_x(n) \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \frac{A_{3z}-A_{4z}}{d_{3y}-d_{4y}} \\ \frac{A_{1z}-A_{2z}}{d_{2x}-d_{1x}} \\ \frac{A_{3x}-A_{4x}}{d_{4y}-d_{3y}} \\ \frac{A_{1y}-A_{2y}}{d_{1x}-d_{2x}} \\ \frac{A_{3y}-A_{4y}}{d_{4y}-d_{3y}} \\ \frac{A_{1x}-A_{2x}}{d_{2x}-d_{1x}} \end{bmatrix}$$

Measurement equation:

$$\begin{bmatrix} A_{1x} \\ A_{2x} \\ A_{3x} \\ A_{4x} \\ A_{1y} \\ A_{2y} \\ A_{3y} \\ A_{4y} \\ A_{1z} \\ A_{2z} \\ A_{3z} \\ A_{4z} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & -d_{1x} & -d_{1x} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & -d_{2x} & -d_{2x} & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -d_{3y} & 0 & 0 & 0 & d_{3y} & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & -d_{4y} & 0 & 0 & 0 & d_{4y} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & d_{1x} & 0 & 0 & 0 & d_{1x} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & d_{2x} & 0 & 0 & 0 & d_{2x} & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -d_{3y} & 0 & -d_{3y} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & -d_{4y} & 0 & -d_{4y} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -d_{1x} & 0 & 0 & 0 & 0 & 0 & 0 & d_{1x} \\ 0 & 0 & 1 & 0 & -d_{2x} & 0 & 0 & 0 & 0 & 0 & 0 & d_{2x} \\ 0 & 0 & 1 & d_{3y} & 0 & 0 & 0 & 0 & 0 & 0 & d_{3y} & 0 \\ 0 & 0 & 1 & d_{4y} & 0 & 0 & 0 & 0 & 0 & 0 & d_{4y} & 0 \end{bmatrix} \begin{bmatrix} \ddot{x}(n) \\ \ddot{y}(n) \\ \ddot{z}(n) \\ \dot{\omega}_x(n) \\ \dot{\omega}_y(n) \\ \dot{\omega}_z(n) \\ \omega_x^2(n) \\ \omega_y^2(n) \\ \omega_z^2(n) \\ \omega_x\omega_y(n) \\ \omega_y\omega_z(n) \\ \omega_z\omega_x(n) \end{bmatrix}$$

Gyro, Accel, and Odom, Differential drive (indir_multi_sensor_data_fusion_node)

Ibrahim Zunaidi, Norihiko Kato, Yoshihiko Nomura and Hirokazu Matsui, “Positioning System for 4-Wheel Mobile Robot: Encoder, Gyro and Accelerometer Data Fusion with Error Model Method”
Ahmad Kamal Nasir, Hubert Roth, “Pose Estimation By Multisensor Data Fusion Of Wheel Encoders, Gyroscope, Accelerometer And Electronic Compass”

$F(t) = [\, F_1 \quad F_2 \quad F_3 \,]$
$H(t, V_l, V_r, \omega_e, A_x, A_y, \Omega) = [\, H_1 \quad H_2 \quad H_3 \,]$
$A_x(t), A_y(t)$, Accelerometer Readings in Global Frame
$A_{x_r}(t), A_{x_r}(t)$, Accelerometer Readings in Robot Frame
$\Omega(t)$, Gyroscope Readings
$\omega_g(t)$, Angular Velocity from Gyroscope
$V_l(t), V_r(t)$, Linear Velocity from Encoders
$\omega_e(t)$, Angular Velocity from Encoders
Δt, Sampling Time

Encoder Velocity Error Model

$V(t) = \frac{V_l(t)+V_r(t)}{2}$
$\omega(t) = \frac{V_l(t)-V_r(t)}{L}$
$V_x(t+1) = V(t)cos(\omega(t)\Delta t)$
$V_y(t+1) = V(t)sin(\omega(t)\Delta t)$

$\hat{V}(t) = \frac{V_l(t)+S_l(t)V_l(t)+V_r(t)+S_r(t)V_r(t)}{2}$
$\hat{\omega}(t) = \frac{(V_l(t)+S_l(t)V_l(t))-(V_r(t)+S_r(t)V_r(t))}{L+S_D(t)L}$
$\hat{V}_x(t+1) = \hat{V}(t)cos((\hat{\omega}(t) + \Delta\hat{\omega}(t))\Delta t)$
$\hat{V}_y(t+1) = \hat{V}(t)sin((\hat{\omega}(t) + \Delta\hat{\omega}(t))\Delta t)$

$\Delta V_{x_e}(t+1) = \hat{V}_x(t+1) - V_x(t+1)$
$\Delta V_{y_e}(t+1) = \hat{V}_y(t+1) - V_y(t+1)$
$\Delta V_{x_e}(t+1) = \frac{S_l(t)V_l(t)+S_r(t)V_r(t)}{2}cos(\omega_e(t)\Delta t) + \frac{V_l(t)+V_r(t)}{2}\Delta\omega_e\Delta t sin(\Delta\omega_e(t)\Delta t)$
$\Delta V_{x_e}(t+1) = \frac{S_l(t)V_l(t)+S_r(t)V_r(t)}{2}sin(\omega_e(t)\Delta t) - \frac{V_l(t)+V_r(t)}{2}\Delta\omega_e\Delta t cos(\Delta\omega_e(t)\Delta t)$
$\Delta\omega_e(t+1) = \frac{(S_l(t)V_l(t))-S_r(t)V_r(t))}{L} - \frac{S_D(V_l(t)-V_r(t))}{L}$

$S_l(t+1) \approx S_l(t)$
$S_r(t+1) \approx S_r(t)$
$S_D(t+1) \approx S_D(t)$

Accelerometer Velocity Error Model

$A_x(t) = A_{x_r}(t)cos(\omega(t)\Delta t) + A_{y_r}(t)sin(\omega(t)\Delta t)$
$A_y(t) = A_{x_r}(t)sin(\omega(t)\Delta t) + A_{y_r}(t)cos(\omega(t)\Delta t)$

$V_x(t+1) = V_x(t) + S_{ax}(t)A_x(t)\Delta t + B_{ax}$
$V_y(t+1) = V_y(t) + S_{ay}(t)A_y(t)\Delta t + B_{ay}$

$\hat{V}_x(t+1) = \hat{V}_x(t) + (S_{ax}(t) + \Delta S_{ax}(t))A_x(t)\Delta t + (B_{ax} + \Delta B_{ax}(t))$
$\hat{V}_y(t+1) = \hat{V}_y(t) + (S_{ay}(t) + \Delta S_{ay}(t))A_y(t)\Delta t + (B_{ay} + \Delta B_{ay}(t))$

$\Delta V_{x_a}(t+1) = \hat{V}_x(t+1) - V_x(t+1)$
$\Delta V_{y_a}(t+1) = \hat{V}_y(t+1) - V_y(t+1)$
$\Delta V_{x_a}(t+1) = \Delta V_{x_a} + \Delta S_{ax}(t)A_x(t)\Delta t + \Delta B_{ax}(t)$
$\Delta V_{y_a}(t+1) = \Delta V_{y_a} + \Delta S_{ay}(t)A_y(t)\Delta t + \Delta B_{ay}(t)$

$S_{ax}(t+1) \approx S_{ax}(t)$
$B_{ax}(t+1) \approx B_{ax}(t)$
$S_{ay}(t+1) \approx S_{ay}(t)$
$B_{ay}(t+1) \approx B_{ay}(t)$

Gyroscope Error Model

$\omega_g(t+1) = S_{gz}(t)\Omega(t) + B_{gz}(t)$
$\hat{\omega}_g(t+1) = (S_{gz}(t) + \Delta S_{gz}(t))\Omega(t) + (B_{gz}(t) + \Delta B_{gz}(t))$

$\Delta\omega_g(t) = \hat{\omega}_g(t+1) - \omega_g(t+1)$
$\Delta\omega_g(t) = \Delta S_{gz}(t)\Omega(t) + \Delta B_{gz}(t)$

$S_{gz}(t+1) \approx S_{gz}(t)$
$B_{gz}(t+1) \approx B_{gz}(t)$

$$F_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \Delta X = \begin{bmatrix} \Delta V_{x_e}(t) \\ \Delta V_{y_e}(t) \\ \Delta V_{x_a}(t) \\ \Delta V_{y_a}(t) \\ \\ \Delta \omega_e(t) \\ \Delta \omega_g(t) \\ \Delta \theta_c(t) \\ \Delta S_l(t) \\ \Delta S_r(t) \\ \\ \Delta S_D(t) \\ \Delta S_{Ax}(t) \\ \Delta B_{Ax}(t) \\ \Delta S_{Ay}(t) \\ \Delta B_{Ay}(t) \\ \Delta S_{Gz}(t) \\ \Delta B_{Gz}(t) \\ \Delta B_C(t) \end{bmatrix} \Delta z = \begin{bmatrix} \Delta V_{x_e}(t) - \Delta V_{x_a}(t) \\ \Delta V_{y_e}(t) - \Delta V_{y_a}(t) \\ \Delta \omega_e(t) - \Delta \omega_g(t) \end{bmatrix}$$

$$F_2 = \begin{bmatrix} \frac{(V_l(t)+V_r(t))\Delta t sin(\omega_e(t)\Delta t)}{2} & 0 & 0 & \frac{V_l(t)cos(\omega_e(t)\Delta t)}{2} & \frac{V_r(t)cos(\omega_e(t)\Delta t)}{2} \\ \frac{-(V_l(t)+V_r(t))\Delta t cos(\omega_e(t)\Delta t)}{2} & 0 & 0 & \frac{V_l(t)sin(\omega_e(t)\Delta t)}{2} & \frac{V_r(t)sin(\omega_e(t)\Delta t)}{2} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{V_l(t)}{L} & \frac{-V_r(t)}{L} \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$F_3 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & A_x(t)\Delta t & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & A_y(t)\Delta t & 1 & 0 & 0 & 0 \\ \frac{V_r(t)-V_l(t)}{L} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \Omega(t) & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$H_1 = \begin{bmatrix} 0 & 0 & -1 & 0 & \frac{-(V_l(t)+V_r(t))\Delta t sin(\omega_e(t)\Delta t)}{2} \\ 0 & 0 & 0 & -1 & \frac{(V_l(t)+V_r(t))\Delta t cos(\omega_e(t)\Delta t)}{2} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$
$$H_2 = \begin{bmatrix} 0 & 0 & \frac{V_l(t)cos(\omega_e(t)\Delta t)}{2} & \frac{V_r(t)cos(\omega_e(t)\Delta t)}{2} \\ 0 & 0 & \frac{V_l(t)sin(\omega_e(t)\Delta t)}{2} & \frac{V_r(t)sin(\omega_e(t)\Delta t)}{2} \\ 0 & 0 & \frac{V_l(t)}{L} & \frac{-V_r(t)}{L} \end{bmatrix}$$
$$H_3 = \begin{bmatrix} 0 & -A_x(t)\Delta t & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -A_y(t)\Delta t & -1 & 0 & 0 & 0 \\ \frac{V_r(t)-V_l(t)}{L} & 0 & 0 & 0 & 0 & -\Omega(t) & -1 & 0 \end{bmatrix}$$

Federated Kalman Filter (array_sensor_data_fusion_node)

Hongwei Zhang, Barry Lennox, Peter R Goulding, Yufei Wang, “ADAPTIVE INFORMATION SHARING FACTORS IN FEDERATED KALMAN FILTERING”

Master filter

Time update

$$\hat{X}_m(n+1,n) = F_m(n+1,n)\hat{X}_m(n)$$
$$P_m(n+1) = F_m(n+1,n)P(n)F_m^{\intercal}(n+1,n) + G_m(n)Q_m(n)G_m^{\intercal}(n)$$

Measurement Update:

$$P_m(n+1) = P_m(n+1,n)$$

Local filter

Time update

$$\hat{X}_i(n+1,n) = F_i(n+1,n)\hat{X}_i(n)$$
$$P_i(n+1) = F_i(n+1,n)P(n)F_i^{\intercal}(n+1,n) + G_i(n)Q_i(n)G_i^{\intercal}(n)$$

Measurement Update:

$$K_i(n) = P_i(n-1)H_i^{\intercal}(n)\Big(H_i(n)P_i(n-1)H_i^{\intercal}(n) + R_i(n)\Big)^{-1}$$
$$X_i(n) = X_i(n-1,n) + K_i(n)\Big(z_i(n) - H_i(n)\hat{X}_i(n-1,n)\Big)$$
$$P_i(n) = \Big(I - K_i(n)H_i(n)\Big)P_i(n-1)\Big(I - K_i(n)H_i(n)\Big)^{\intercal} + K_i(n)R_i(n)K_i^{\intercal}(n)$$

Fusion algorithm

$$P_f^{-1}(n+1) = P_1^{-1}(n+1) + \ldots + P_n^{-1}(n+1) + P_m^{-1}(n+1)$$
$$\hat{X}_f(n+1) = P_f(n+1)\Big(P_m^{-1}(n+1)\hat{X}_m(n+1) + \sum_{i=1}^{k} P_i^{-1}(n+1)\hat{X}_i(n+1)\Big)$$