CENG4480

Lecture 08: Kalman Filter

Bei Yu

byu@cse.cuhk.edu.hk

(Latest update: November 18, 2019)

Fall 2019

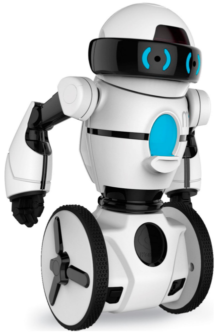
香港中文大學
The Chinese University of Hong Kong

Overview

Overview

Self Balance Vehicle / Robot

- http://www.segway.com/
- http://wowwee.com/mip/

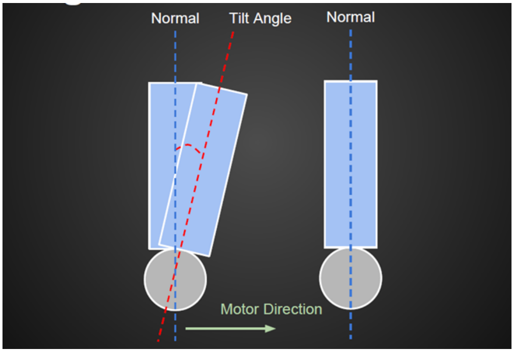


The WowWee MiP Robot paired with WowWee's free app creates a consumer robot with lots of potential.
Relatively small in size, the WowWee MiP Robo is only 7 inches tall and has no feet. It is black and white in design with a round head and emoticon eyes. It might remind you of Disney's Wall-E.

The WowWee MiP Robot can balance and move quite well, similar to a Segway.

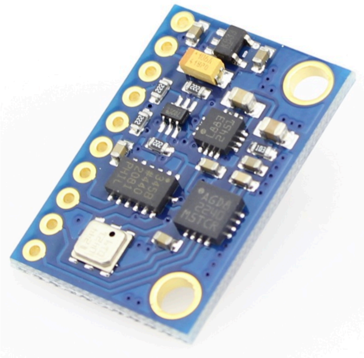
The connection is extremely easy. To connect MiP to your mobile device you simply need to download the app. Once opened the MiP's ID will appear on the screen and you choose how you want to control the robot.

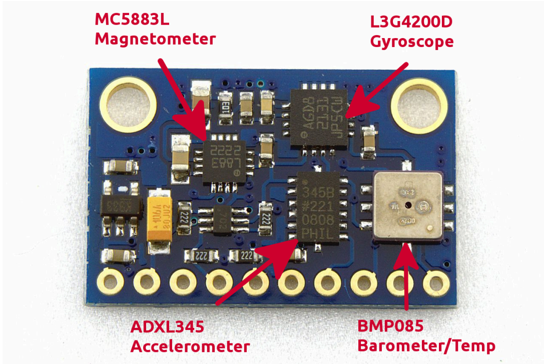
Basic Idea



Motion against the tilt angle, so it can stand upright.

IMU Board



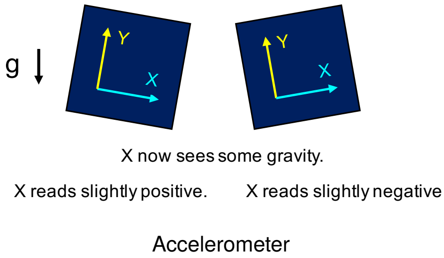


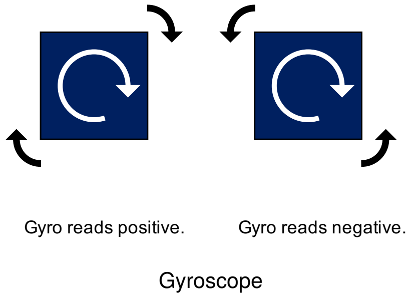
http://www.hotmcu.com/imu-10dof-l3g4200dadxl345hmc5883lbmp180-p-190.html

- L3G4200D: gyroscope, measure angular rate (relative value)
- ADXL345: accelerometer, measure acceleration

Overview

Complementary Filter



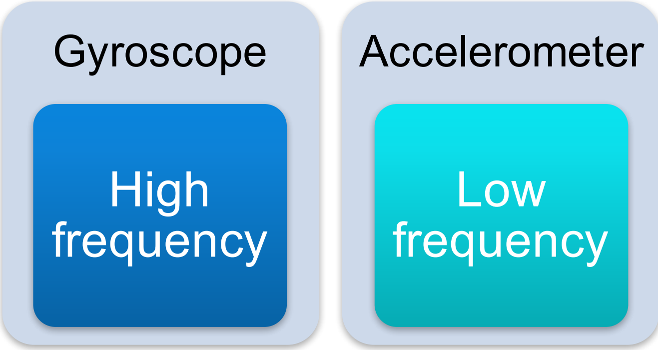


- Give accurate reading of tilt angle
- Slower to respond than Gyro's
- prone to vibration/noise

- response faster
- but has drift over time

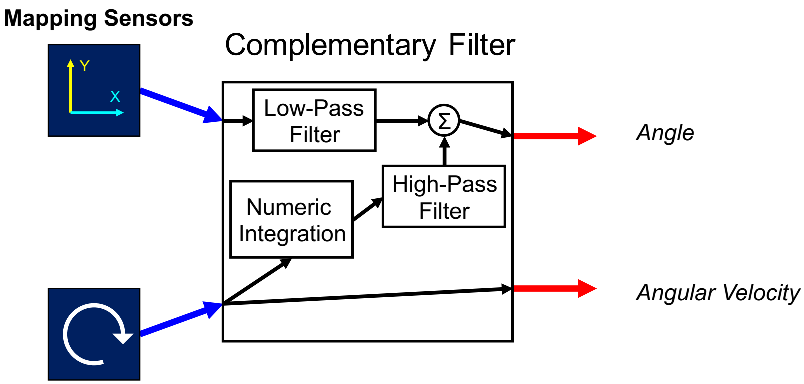
Complementary Filter (cont.)

- Since



- Combine two sensors to find output

Complementary Filter (cont.)




```
Read_acc();
Read_gyro();
Ayz=atan2(RwAcc[1],RwAcc[2])*180/PI;                    //angle by accelerometer
Ayz-=offset;                                            //adjust to correct
Angy = 0.98*(Angy+GyroIN[0]*interval/1000)+0.02*Ayz;  //complement filter
```

Overview

Rudolf Kalman (1930 – 2016)

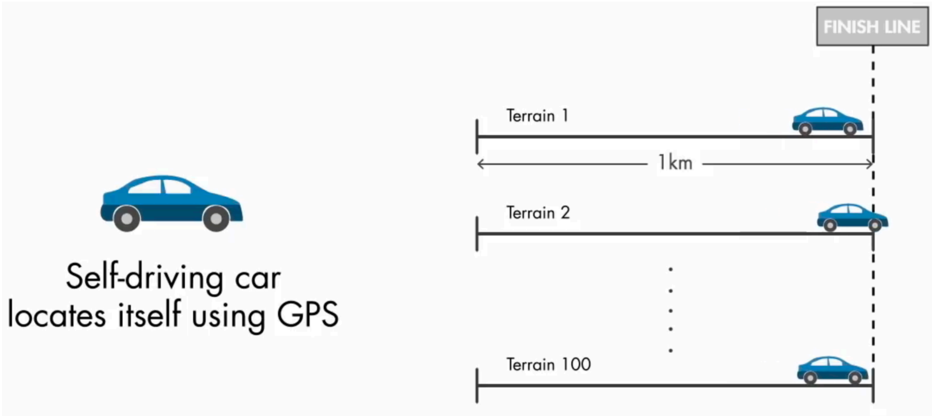
- Born in Budapest, Hungary
- BS in 1953 and MS in 1954 from MIT electrical engineering
- PhD in 1957 from Columbia University.

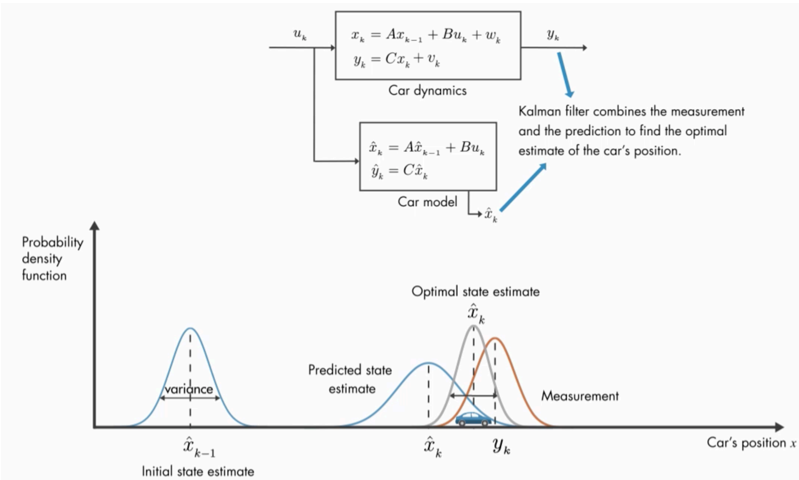
- Famous for his co-invention of the Kalman filter – widely used in control systems to extract a signal from a series of incomplete and noisy measurements.
- Convince NASA Ames Research Center 1960
- Kalman filter was used during Apollo program

He was a professor at Stanford University from 1964 until 1971, and then at the University of Florida from 1971 until 1992

Problem Example 1

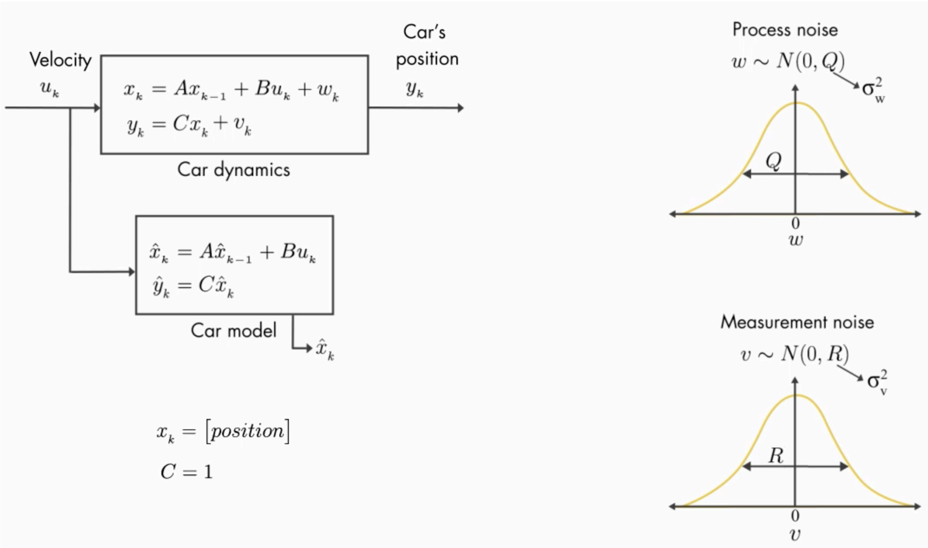
Self-Driving Car Location Problem

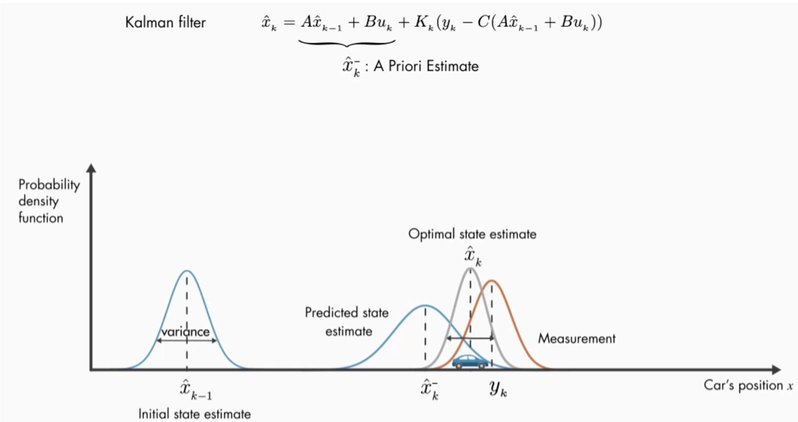




Problem Example 1

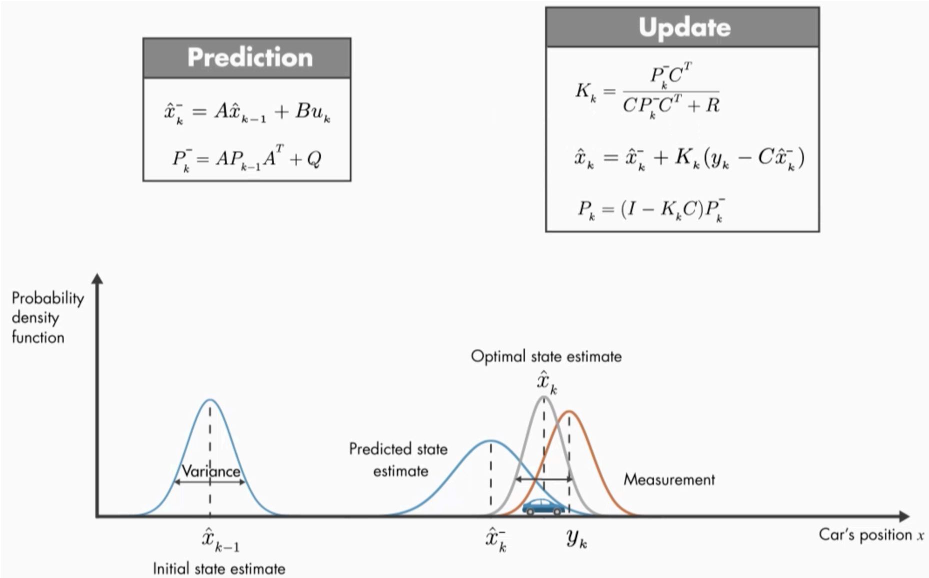
Self-Driving Car Location Problem

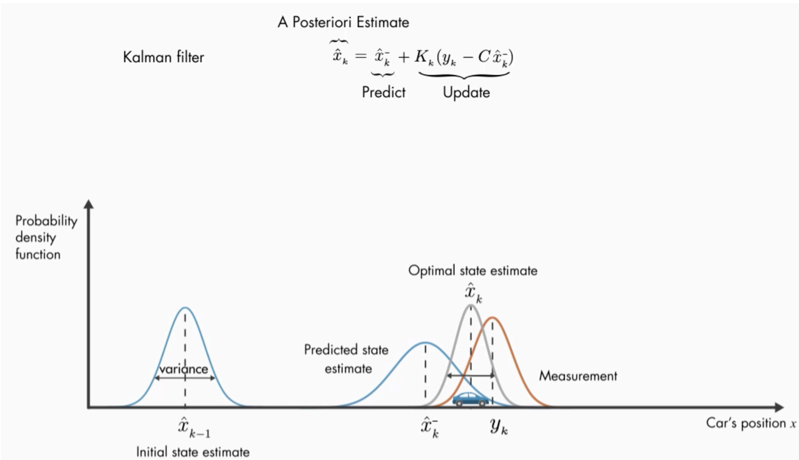




Problem Example 1

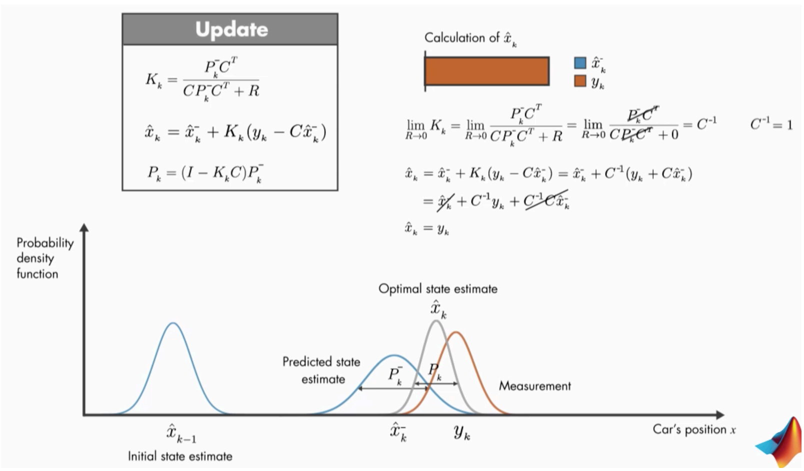
Self-Driving Car Location Problem





Exercise: Analyse Kalman Gain

What is Kalman Gain $\boldsymbol{K}_k$, if measurement noise $\boldsymbol{R}$ is very small? What if $\boldsymbol{R}$ is very big?



Problem Example 2

Angle Measurement System

$$\boldsymbol{x}_t = \boldsymbol{A}_t \boldsymbol{x}_{t-1} + \boldsymbol{B}_t \boldsymbol{u}_t + \boldsymbol{w}_t$$

- $\boldsymbol{x}_t$: state in time t
- $\boldsymbol{A}_t$: state transition matrix from time $t-1$ to time t
- $\boldsymbol{u}_t$: input parameter vector at time t
- $\boldsymbol{B}_t$: control input matrix – apply the effort of $\boldsymbol{u}_t$
- $\boldsymbol{w}_t$: process noise, $\boldsymbol{w}_t \sim N(0, \boldsymbol{Q}_t)*$

$*\boldsymbol{w}_t$ assumes zero mean multivariate normal distribution, covariance matrix $\boldsymbol{Q}_t$

Problem Example 2 (Update on Oct. 29, 2018)

Angle Measurement System

$$\boldsymbol{x}_t = \boldsymbol{A}_t \boldsymbol{x}_{t-1} + \boldsymbol{B}_t \boldsymbol{u}_t + \boldsymbol{w}_t$$

- $\boldsymbol{x}_t = [x_t, \dot{x}_t]^\top$: x_t is current angle, while $\dot{x}_t$ is current rate
- $\boldsymbol{A}_t = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}$
- $\boldsymbol{B}_t = [\frac{(\Delta t)^2}{2}, \Delta t]^\top$
- $\boldsymbol{u}_t = \Delta \dot{x}_t$

Problem Example 2

System Measurement

$$z_t = \boldsymbol{C}\boldsymbol{x}_t + \boldsymbol{v}_t$$

- z_t: measurement vector
- $\boldsymbol{C}$: transformation matrix mapping state vector to measurement
- $\boldsymbol{v}_t$: measurement noise, $\boldsymbol{v}_t \sim N(0, \boldsymbol{R}_t)$†

†$\boldsymbol{w}_t$ assumes zero mean multivariate normal distribution, covariance matrix $\boldsymbol{R}_t$

Exercise

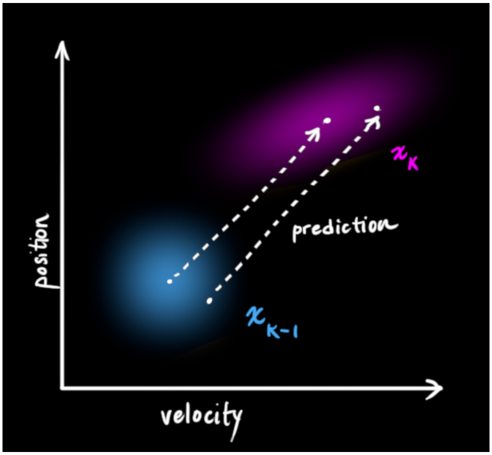
In angle measurement lab, what is the transformation matrix $\boldsymbol{C}$?

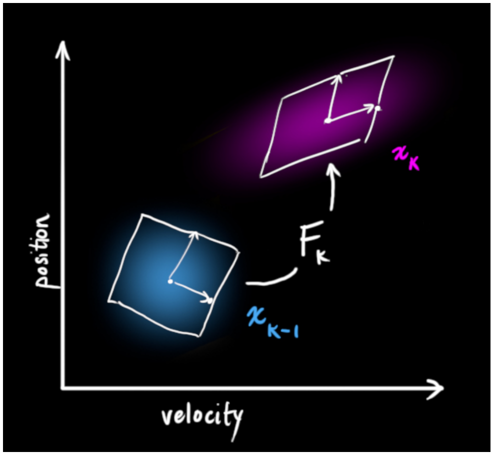
$$z_t = \boldsymbol{C}\boldsymbol{x}_t + \boldsymbol{v}_t$$

[1, 0]

Model with Uncertainty

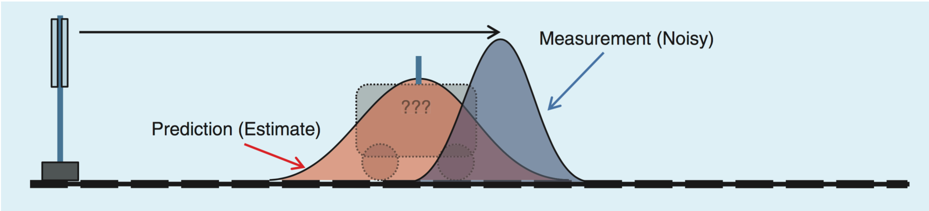
- Model the measurement w. uncertainty (due to noise $\boldsymbol{w}_t$)
- $\boldsymbol{P}_k$: covariance matrix of estimation $\boldsymbol{x}_t$
- On how much we trust our estimated value – the smaller the more we trust



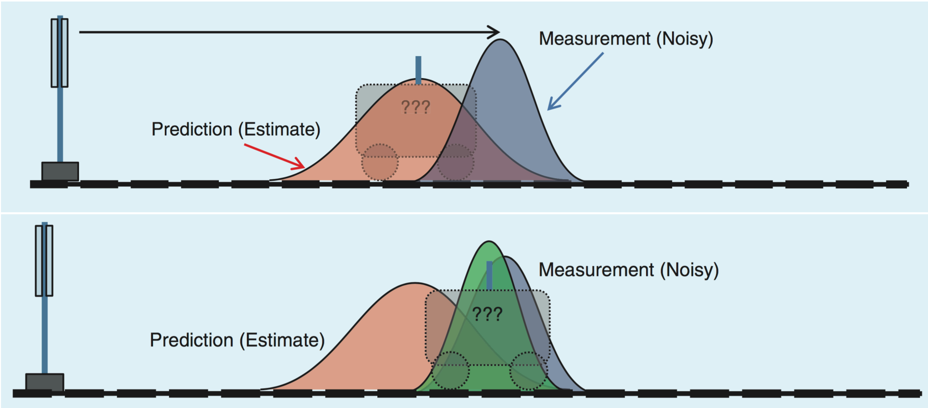


note: here $\boldsymbol{F}_k = \boldsymbol{A}_k$

Fuse Gaussian Distributions



Fuse Gaussian Distributions



Exercise

Given two Gaussian functions $y_1(r;\mu_1,\sigma_1)$ and $y_2(r;\mu_2,\sigma_2)$, prove the product of these two Gaussian functions are still Gaussian.

$$y_1(r;\mu_1,\sigma_1) = \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{-\frac{(r-\mu_1)^2}{2\sigma_1^2}} \qquad y_2(r;\mu_2,\sigma_2) = \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(r-\mu_2)^2}{2\sigma_2^2}}$$

Their product is

$$f(x)g(x) = \frac{1}{2\pi\sigma_f\sigma_g} e^{-\left(\frac{(x-\mu_f)^2}{2\sigma_f^2}+\frac{(x-\mu_g)^2}{2\sigma_g^2}\right)}$$

Examine the term in the exponent

$$\beta = \frac{(x-\mu_f)^2}{2\sigma_f^2} + \frac{(x-\mu_g)^2}{2\sigma_g^2}$$

Expanding the two quadratics and collecting terms in powers of x gives

$$\beta = \frac{(\sigma_f^2+\sigma_g^2)x^2 - 2(\mu_f\sigma_g^2+\mu_g\sigma_f^2)x + \mu_f^2\sigma_g^2 + \mu_g^2\sigma_f^2}{2\sigma_f^2\sigma_g^2}$$

Dividing through by the coefficient of x^2 gives

$$\beta = \frac{x^2 - 2\frac{\mu_f\sigma_g^2+\mu_g\sigma_f^2}{\sigma_f^2+\sigma_g^2}x + \frac{\mu_f^2\sigma_g^2+\mu_g^2\sigma_f^2}{\sigma_f^2+\sigma_g^2}}{2\frac{\sigma_f^2\sigma_g^2}{\sigma_f^2+\sigma_g^2}}$$

This is again a quadratic in x, and so Eq. 2 is a Gaussian function. Compare the terms in Eq. 5 to a the usual Gaussian form

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x^2-2\mu x+\mu^2)}{2\sigma^2}}$$

Since a term ϵ that is independent of x can be added to complete the square in β, this is sufficent to complete the proof in cases where the normalisation can be ignored. The product of two Gaussian PDFs is proportional to a Gaussian PDF with a mean that is half the coefficient of x in Eq. 5 and a standard deviation that is the square root of half of the denominator i.e.

$$\sigma_{fg} = \sqrt{\frac{\sigma_f^2\sigma_g^2}{\sigma_f^2+\sigma_g^2}} \quad \text{and} \quad \mu_{fg} = \frac{\mu_f\sigma_g^2+\mu_g\sigma_f^2}{\sigma_f^2+\sigma_g^2}$$

Step 1: Prediction

$$\boldsymbol{x}_t^- = \boldsymbol{A}_t \boldsymbol{x}_{t-1} + \boldsymbol{B}_t \boldsymbol{u}_t \quad (1)$$

$$\boldsymbol{P}_t^- = \boldsymbol{A}_t \boldsymbol{P}_{t-1} \boldsymbol{A}_t^\top + \boldsymbol{Q}_t \quad (2)$$

Step 1: Prediction

$$x_t^- = A_t x_{t-1} + B_t u_t \tag{1}$$

$$P_t^- = A_t P_{t-1} A_t^\top + Q_t \tag{2}$$

Step 2: Measurement Update

$$x_t = x_t^- + K_t(z_t - C x_t^-) \tag{3}$$

$$P_t = P_t^- - K_t C P_t^- \tag{4}$$

$$K_t = P_t^- C^\top (C P_t^- C^\top + R_t)^{-1} \tag{5}$$

$$y_1(s;\mu_1,\sigma_1,c) \triangleq \frac{1}{\sqrt{2\pi\left(\frac{\sigma_1}{c}\right)^2}} e^{-\frac{\left(s-\frac{\mu_1}{c}\right)^2}{2\left(\frac{\sigma_1}{c}\right)^2}} \tag{14}$$

and

$$y_2(s;\mu_2,\sigma_2) \triangleq \frac{1}{\sqrt{2\pi\sigma_2^2}} e^{-\frac{(s-\mu_2)^2}{2\sigma_2^2}}, \tag{15}$$

where both distributions are now defined in the measurement domain, radio signals propagate along the time "s" axis, and the measurement unit is the second.

Following the derivation as before we now find

$$\frac{\mu_{\text{fused}}}{c} = \frac{\mu_1}{c} + \frac{\left(\frac{\sigma_1}{c}\right)^2\left(\mu_2 - \frac{\mu_1}{c}\right)}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}$$

$$\Rightarrow \mu_{\text{fused}} = \mu_1 + \left(\frac{\frac{\sigma_1^2}{c}}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}\right) \cdot \left(\mu_2 - \frac{\mu_1}{c}\right). \tag{16}$$

Substituting $H = 1/c$ and $K = (H\sigma_1^2)/(H^2\sigma_1^2 + \sigma_2^2)$ results in

$$\mu_{\text{fused}} = \mu_1 + K \cdot (\mu_2 - H\mu_1). \tag{17}$$

Similarly the fused variance estimate becomes

$$\frac{\sigma_{\text{fused}}^2}{c^2} = \left(\frac{\sigma_1}{c}\right)^2 - \frac{\left(\frac{\sigma_1}{c}\right)^4}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}$$

$$\Rightarrow \sigma_{\text{fused}}^2 = \sigma_1^2 - \left(\frac{\frac{\sigma_1^2}{c}}{\left(\frac{\sigma_1}{c}\right)^2 + \sigma_2^2}\right) \frac{\sigma_1^2}{c}.$$

$$= \sigma_1^2 - KH\sigma_1^2 \tag{18}$$

- $K = \frac{H\sigma_1^2}{H^2\sigma_1^2 + \sigma_2^2} \rightarrow \mathbf{K}_t$
  $= \mathbf{P}_{t|t-1}\mathbf{H}_t^{\mathbf{T}}(\mathbf{H}_t\mathbf{P}_{t|t-1}\mathbf{H}_t^{\mathbf{T}} + \mathbf{R}_t)^{-1}$: the Kalman gain.

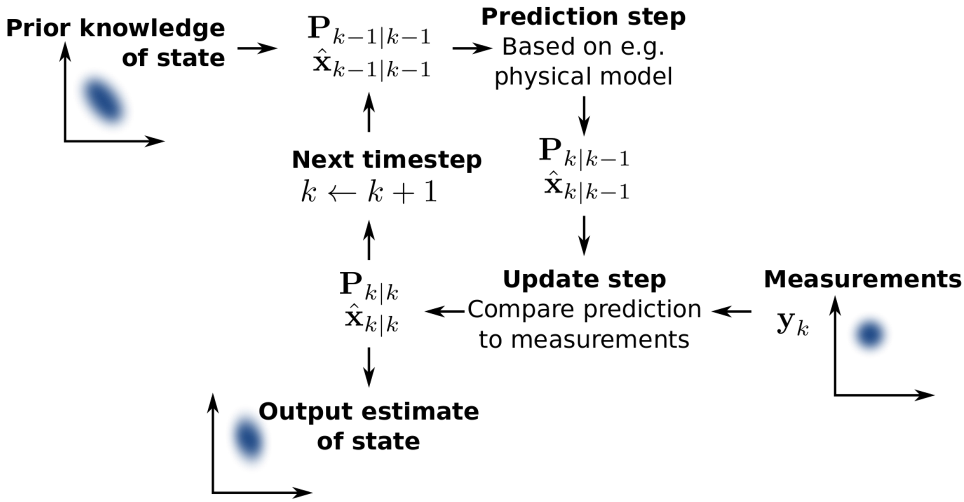
It is now easy to see how the standard Kalman filter equations relate to (17) and (18) derived above:

$$\mu_{\text{fused}} = \mu_1 + \left(\frac{H\sigma_1^2}{H^2\sigma_1^2 + \sigma_2^2}\right) \cdot (\mu_2 - H\mu_1)$$

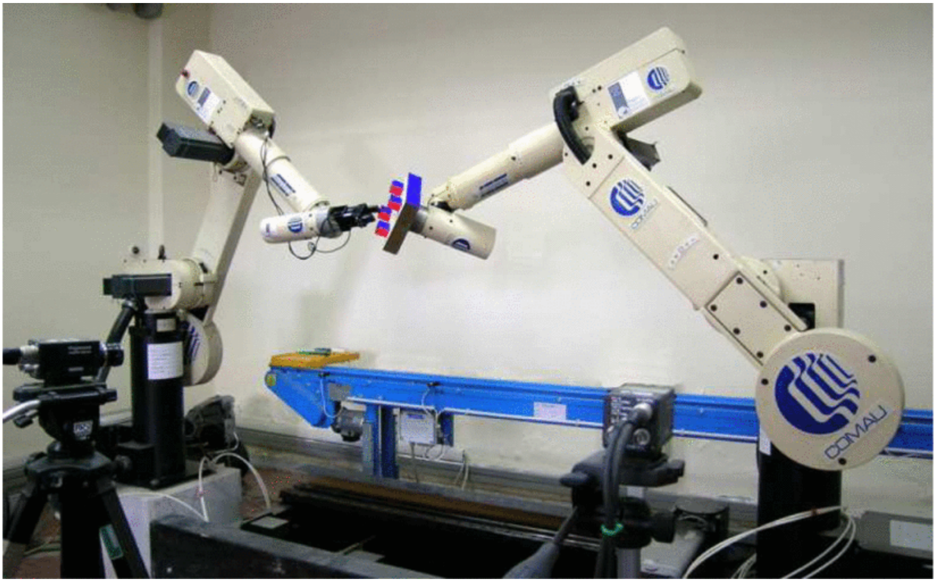
$$\rightarrow \hat{\mathbf{x}}_{t|t} = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{z}_t = \mathbf{H}_t\hat{\mathbf{x}}_{t|t-1})$$

$$\sigma_{\text{fused}}^2 = \sigma_1^2 - \left(\frac{H\sigma_1^2}{H^2\sigma_1^2 + \sigma_2^2}\right) H\sigma_1^2$$

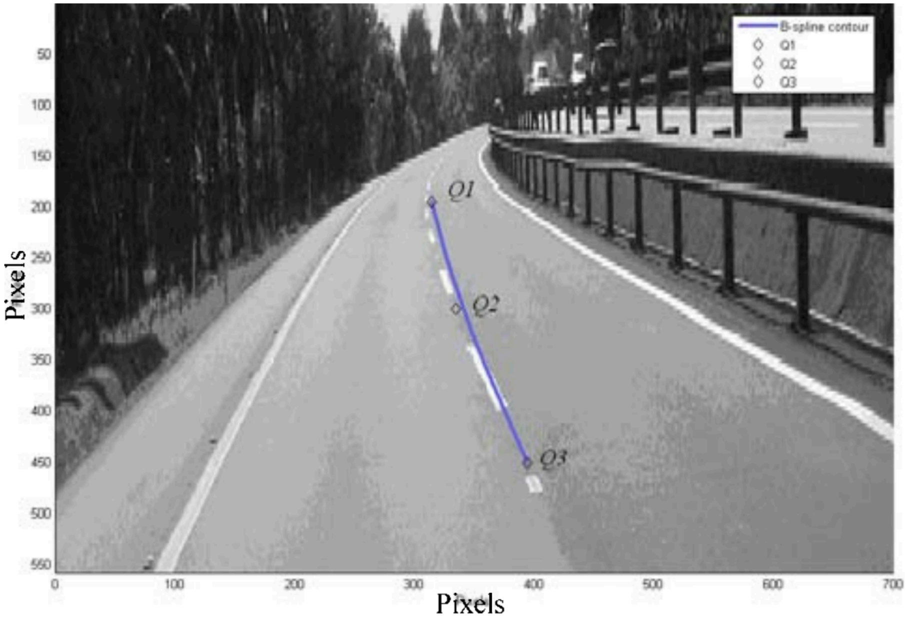
Basic Concepts



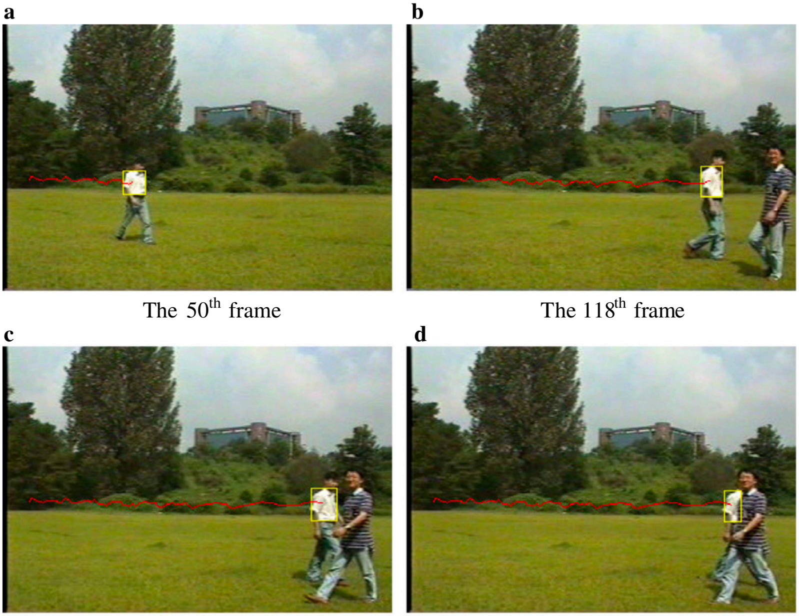
More Applications: Robot Localization



More Applications: Path Tracking



More Applications: Object Tracking



The 50^{th} frame

The 118^{th} frame

The 124^{th} frame

The 127^{th} frame

Overview

C Implementation

```
// Kalman filter module
float Q_angle  =  0.001;
float Q_gyro   =  0.003;
float R_angle  =  0.03;

float x_angle = 0;
float x_bias = 0;
float P_00 = 0, P_01 = 0, P_10 = 0, P_11 = 0;
float dt, y, S;
float K_0, K_1;
```

- Q:
- R:
- P:

C Implementation (cont.)

```
float kalmanCalculate(float newAngle, float newRate,int looptime)
{
    dt = float(looptime)/1000;
    x_angle += dt * (newRate - x_bias);
    P_00    += dt * (P_10 + P_01) + Q_angle * dt;
    P_01    += dt * P_11;
    P_10    += dt * P_11;
    P_11    += Q_gyro * dt;

    y   = newAngle - x_angle;
    S   = P_00 + R_angle;
    K_0 = P_00 / S;
    K_1 = P_10 / S;

    x_angle +=  K_0 * y;
    x_bias  +=  K_1 * y;
    P_00 -= K_0 * P_00;
    P_01 -= K_0 * P_01;
    P_10 -= K_1 * P_00;
    P_11 -= K_1 * P_01;

    return x_angle;
}
```

Summary

- Complementary Filter
- Kalman Filter