Kalman Filter
&&
Monte Carlo Algorithm

Liu TaiChen @imorpheus.ai

2017-10-27

Outlines

- Bayes && Gaussian
- Kalman filter && Monte Carlo
- Application

IMORPHEUS.AI

Bayes && Gaussian

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \quad - State\ \ function$$

$\mathbf{x}_k - state\ vector$

$\mathbf{u}_k - control\ \ vector$

$\mathbf{w}_k - noise\ vector$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \quad - \ Measurement\ \ function$$

$\mathbf{z}_k - measurement\ vector$

$\mathbf{v}_k - noise\ vector$

$$P(\mathbf{x}_k \mid \mathbf{z}_k, \mathbf{u}_k)$$

IMORPHEUS.AI

Bayes && Gaussian

Application in GPS calibration

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \quad - State\ \ function$$

$$\mathbf{x}_k - state\ vector --- predicted\ \ GPS$$

$$\mathbf{u}_k - control\ \ vector --- displacement < -(general - IMU\ data; SLAM)$$

$$\mathbf{w}_k - noise\ vector$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \quad - Measurement\ \ function$$

$$\mathbf{z}_k - measurement\ vector --- measured\ GPS$$

$$\mathbf{v}_k - noise\ vector$$

$$\arg\max P(\mathbf{x}_k \mid \mathbf{z}_k, \mathbf{u}_k)$$

IMORPHEUS.AI

Bayes && Gaussian

$$\boxed{P(\mathbf{x} \mid \mathbf{z})} = \frac{P(\mathbf{z} \mid \mathbf{x})P(\mathbf{x})}{P(\mathbf{z})} \propto \boxed{P(\mathbf{z} \mid \mathbf{x})}\boxed{P(\mathbf{x})}$$

prior probability

posterior probability

likelihood

$$\arg\max P(\mathbf{x} \mid \mathbf{z}) \propto \arg\max P(\mathbf{z} \mid \mathbf{x})$$

IMORPHEUS.AI

Bayes && Gaussian

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k$$

$$+$$

$$\boxed{\mathbf{v}_k \sim N(0, \mathbf{Q}_k)}$$

Measurement Function

$$P(\mathbf{z}_k \mid \mathbf{x}_k) = N(h(\mathbf{x}_k), \mathbf{Q}_k)$$

$$+$$

$$P(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^N \det(\Sigma)}} \exp(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}))$$

$$-\ln(P(\mathbf{x})) = \frac{1}{2}\ln((2\pi)^N \det(\Sigma)) + \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})$$

$$\arg\max P(\mathbf{z}_k \mid \mathbf{x}_k) \Rightarrow \arg\min(-\ln(P(\mathbf{z}_k \mid \mathbf{x}_k)))$$

$$\mathbf{z}_k -> \mathbf{x}, \quad \boldsymbol{\mu} -> h(\mathbf{x}_k)$$

$$\begin{aligned} \mathbf{x}_m^* &= \arg\min((\mathbf{z}_k - h(\mathbf{x}_k))^T \mathbf{Q}_k^{-1}(\mathbf{z}_k - h(\mathbf{x}_k))) \\ &= \arg\min(\mathbf{v}_k^T \mathbf{Q}_k^{-1} \mathbf{v}_k) \end{aligned}$$

IMORPHEUS.AI

Bayes && Gaussian

The same as state function:

$$\mathbf{x}_s^{\,*} = \arg\min((\mathbf{x}_k - f(\mathbf{x}_{k-1}, \mathbf{u}_k))^T \mathbf{R}_k^{-1} (\mathbf{x}_k - f(\mathbf{x}_{k-1}, \mathbf{u}_k)) = \arg\min(\mathbf{w}_k^{\,T} \mathbf{R}_k^{-1} \mathbf{w}_k)$$

$$\mathbf{x}_m^{\,*} = \arg\min((\mathbf{z}_k - h(\mathbf{x}_k))^T \mathbf{Q}_k^{-1} (\mathbf{z}_k - h(\mathbf{x}_k)) = \arg\min(\mathbf{v}_k^{\,T} \mathbf{Q}_k^{-1} \mathbf{v}_k)$$

$$J(\mathbf{x}) = \sum_k \mathbf{w}_k^{\,T} \mathbf{R}_k^{-1} \mathbf{w}_k + \sum_k \mathbf{v}_k^{\,T} \mathbf{Q}_k^{-1} \mathbf{v}_k$$

$$\arg\min J(\mathbf{x}) = \arg\min(\sum_k \mathbf{w}_k^{\,T} \mathbf{R}_k^{-1} \mathbf{w}_k + \sum_k \mathbf{v}_k^{\,T} \mathbf{Q}_k^{-1} \mathbf{v}_k)$$

$\longleftrightarrow$ **Least square problem**

IMORPHEUS.AI

Outlines

- Bayes && Gaussian
- Kalman filter && Monte Carlo
- Application

IMORPHEUS.AI

Kalman filter

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k \ - State\ \ function$$

$$linear\ \ assumption:$$

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_k) + \mathbf{w}_k = \mathbf{A}_k \mathbf{x}_{k-1} + \mathbf{B}_k \mathbf{u}_k + \mathbf{w}_k$$

$$noise\ \ distribution:$$

$$\mathbf{w}_k \sim N(0, \ \ \mathbf{R}_k)$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k \ - Measurement\ \ function$$

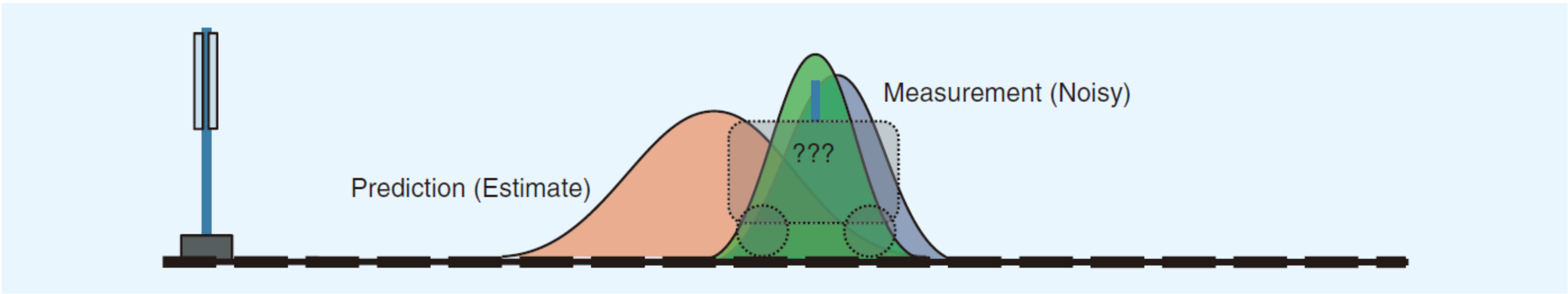
$$linear\ \ assumption:$$

$$\mathbf{z}_k = h(\mathbf{x}_k) + \mathbf{v}_k = \mathbf{C}_k \mathbf{x}_k + \mathbf{v}_k$$

$$noise\ \ distribution:$$

$$\mathbf{v}_k \sim N(0, \ \ \mathbf{Q}_k)$$

Kalman filter



Grey orange – state function's prediction

Grey blue – measurement function's output

Green – optimum estimate

$$\mathbf{x}_k = \overline{\mathbf{x}}_k + K_k(\mathbf{z}_k - \mathbf{C}_k\overline{\mathbf{x}}_k)$$

$$\overline{\mathbf{x}}_k = \mathbf{A}_k\mathbf{x}_{k-1} + \mathbf{B}_k\mathbf{u}_k \text{ ---- } \textit{state function's prediction}$$

$$\mathbf{z}_k \text{ ---- } \textit{measurement}$$

$$\mathbf{x}_k \text{ ---- } \textit{optimum estimate}$$

IMORPHEUS.AI

Monte Carlo

$$J(\mathbf{x}) = \sum_k \mathbf{w}_k^{\ T} \mathbf{R}_k^{\ -1} \mathbf{w}_k + \sum_k \mathbf{v}_k^{\ T} \mathbf{Q}_k^{\ -1} \mathbf{v}_k$$

$$\arg\min J(\mathbf{x}) = \arg\min(\sum_k \mathbf{w}_k^{\ T} \mathbf{R}_k^{\ -1} \mathbf{w}_k + \sum_k \mathbf{v}_k^{\ T} \mathbf{Q}_k^{\ -1} \mathbf{v}_k)$$

Least square problem

$$\min \frac{1}{2} \| f(x) \|_2^2$$

Generally, this problem is based on **gradient guidance-Taylor**.

$$\| f(\mathbf{x} + \Delta\mathbf{x}) \|_2^2 = \| f(\mathbf{x}) \|_2^2 + \mathbf{J}(\mathbf{x}) \Delta\mathbf{x} + \frac{1}{2} \Delta\mathbf{x}^T \mathbf{H} \Delta\mathbf{x}$$

Monte Carlo Algorithm is based on **probability** to solve this problem.

Monte Carlo

Monte Carlo Algorithm is proposed by Metropolis && Ulam.

1st version of Monte Carlo Algorithm – Enumeration Method

$$find\ \ each\ \ x_i -> \min \frac{1}{2} \| f(x) \|_2^2$$

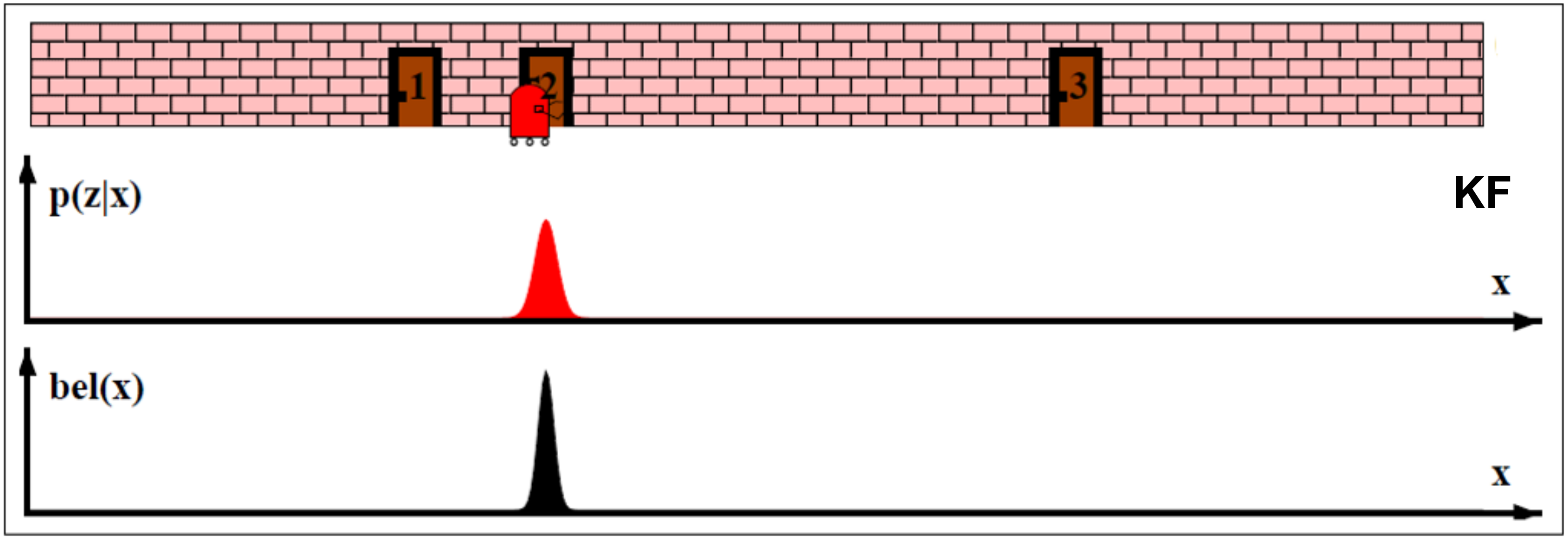
1） x_i *should be known*

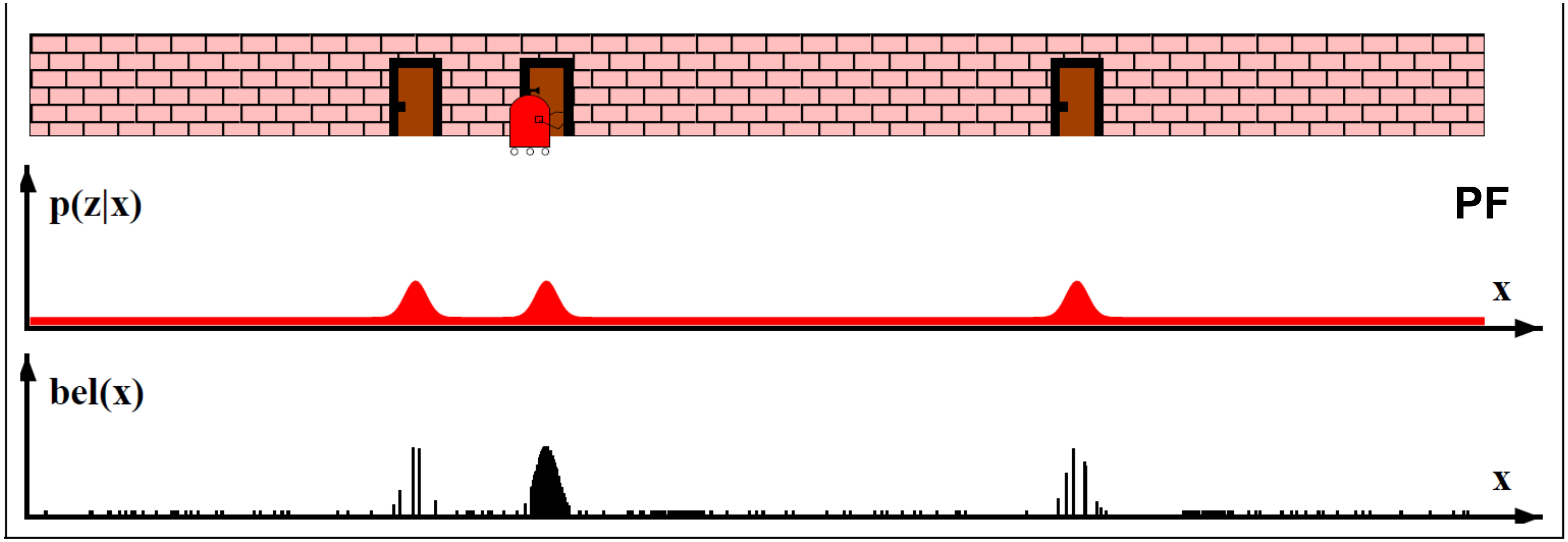
2） low efficiency

Monte Carlo Algorithm-search engine based on **probability density.**

In monte carlo location: UKF, PF are proposed.

KF VS PF





Outlines

- Bayes && Gaussian
- Kalman filter && Monte Carlo
- Application

IMORPHEUS.AI

iMorpheus

Special Guest : Mr WEN Weisong

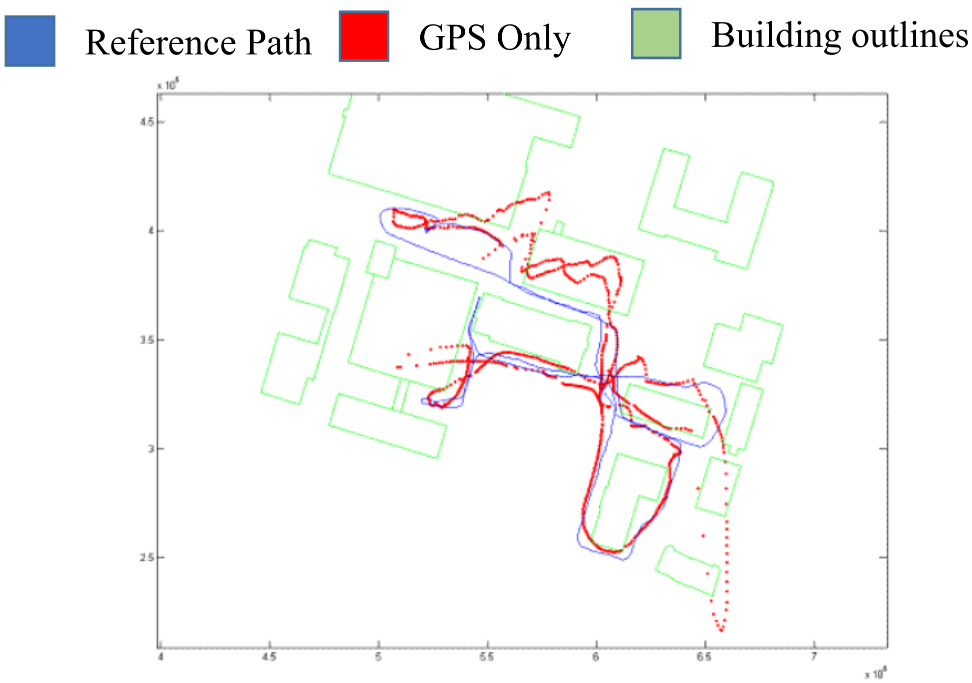
THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學

A GPS and laser-based localization for urban and non-urban outdoor environments

WEN Weisong

He is currently a PhD. student at the Department of Mechanical Engineering, The Hong Kong Polytechnic University, Hong Kong. His research interests including perception in autonomous driving including localization with multi-sensors and deep learning-based objection detection.

Abstract— This paper introduces a localization based on GPS and laser measurements for urban and non-urban outdoor environments. In this approach, the GPS pose is Kalman filtered using wheel odometry and inertial data and tightly integrated into a Monte Carlo Localization based on 3D laser range data and a line feature reference map. By applying to this kind of sensor fusion, global localization as well as precise position tracking in close distance to buildings is enabled where only poor GPS observations are available. Following the description of the localization system, real world experiments demonstrate the functionality of the presented approach.

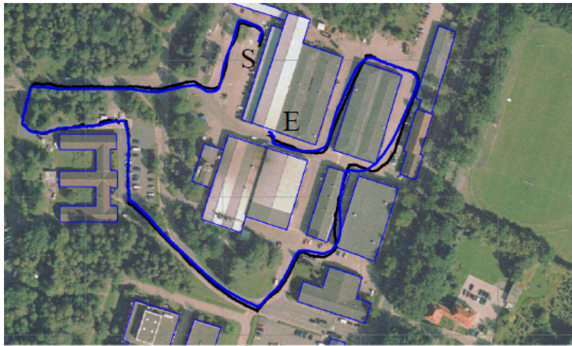


Minimal drift but much more noise

Low Noise and drift off over time

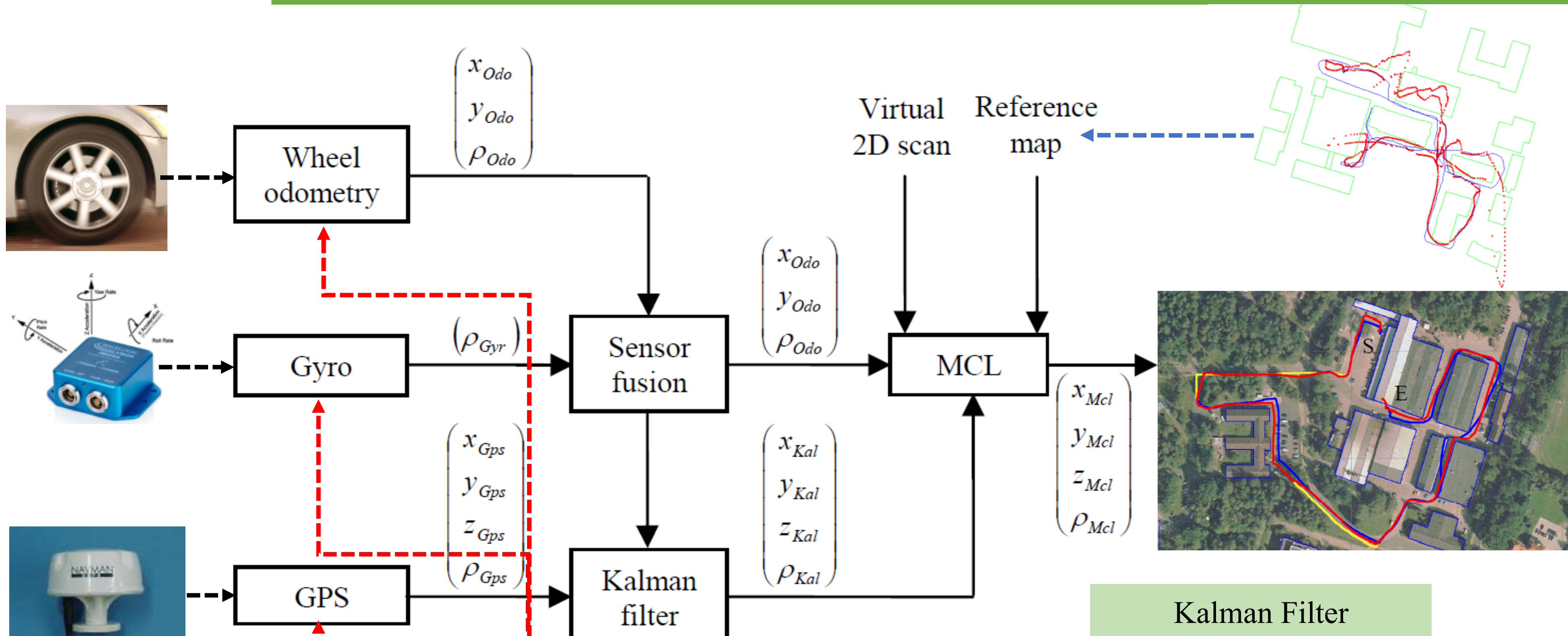
Sensor Fusion

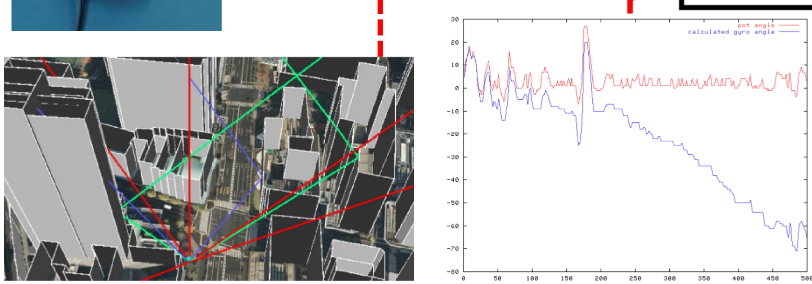
Robust Localization Result



Hentschel, M., Wulf, O., & Wagner, B. (2008). A GPS and laser-based localization for urban and non-urban outdoor environments. *Ieee/rsj International Conference on Intelligent Robots and Systems* (Vol.10, pp.149-154). IEEE.

Block diagram of the GPS and laser-based localization

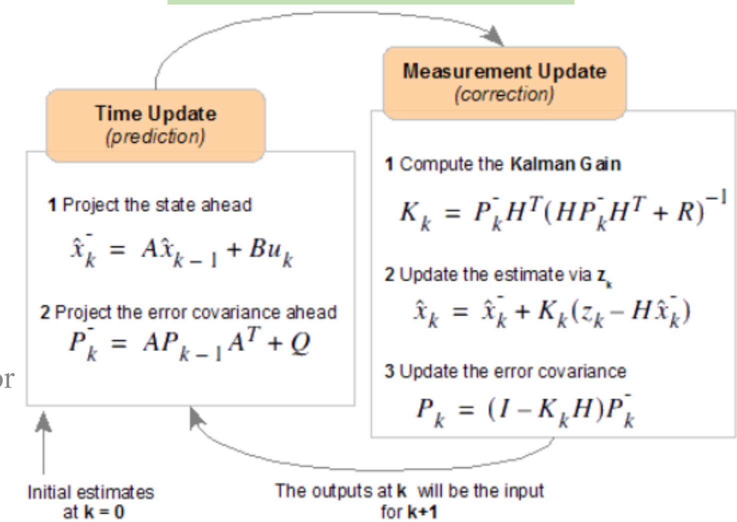




Multipath effects

Drift by time

Kalman Filter



Hentschel, M., Wulf, O., & Wagner, B. (2008). A GPS and laser-based localization for urban and non-urban outdoor environments. *Ieee/rsj International Conference on Intelligent Robots and Systems* (Vol.10, pp.149-154). IEEE.

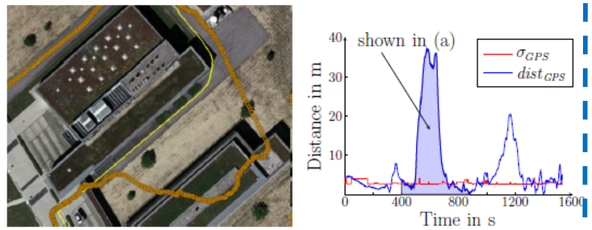
GPS Dead Reckoning (Kalman Filter)

GPS measurement

$$\underline{z}_{k,Gps} = \left(x_{Gps}, y_{Gps}, z_{Gps}, \rho_{Gps}\right)^T$$

Gauss-Krueger coordinate system

GPS measurement uncertainty



(a) (b)

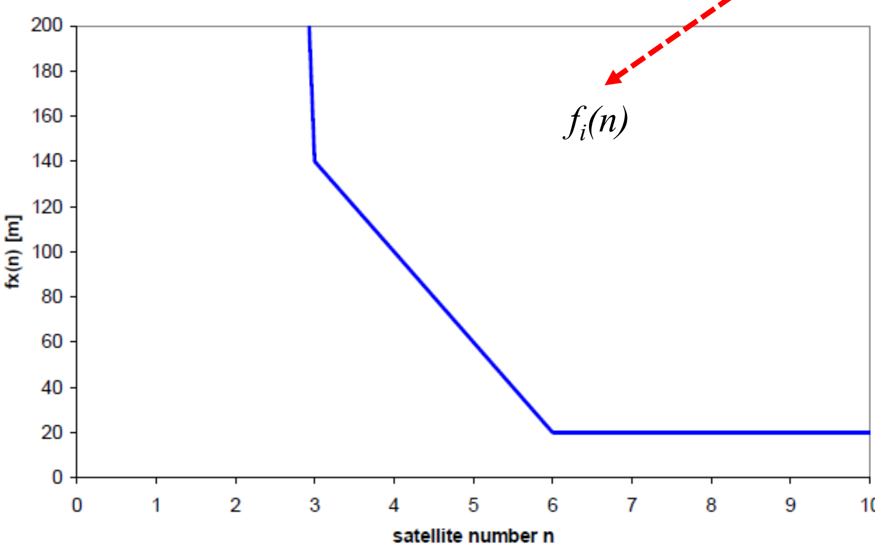
$$\underline{\sigma}_{k,Gps} = \begin{pmatrix} \sigma_x \\ \sigma_y \\ \sigma_z \\ \sigma_\rho \end{pmatrix} = \begin{pmatrix} f_x(n) \cdot g_x(v_{Odo}) \\ f_y(n) \cdot g_y(v_{Odo}) \\ f_z(n) \cdot g_z(v_{Odo}) \\ f_\rho(n) \cdot g_\rho(v_{Odo}) \end{pmatrix}$$

Prediction with motion difference from Odometry

$$\underline{x}_{k+1}^- = A_k \underline{x}_k = \begin{pmatrix} x^- \\ y^- \\ z^- \\ \rho^- \end{pmatrix} = \begin{pmatrix} x + \Delta x_{Odo} \\ y + \Delta y_{Odo} \\ z + \Delta z_{Odo} \\ \rho + \Delta \rho_{Odo} \end{pmatrix}$$

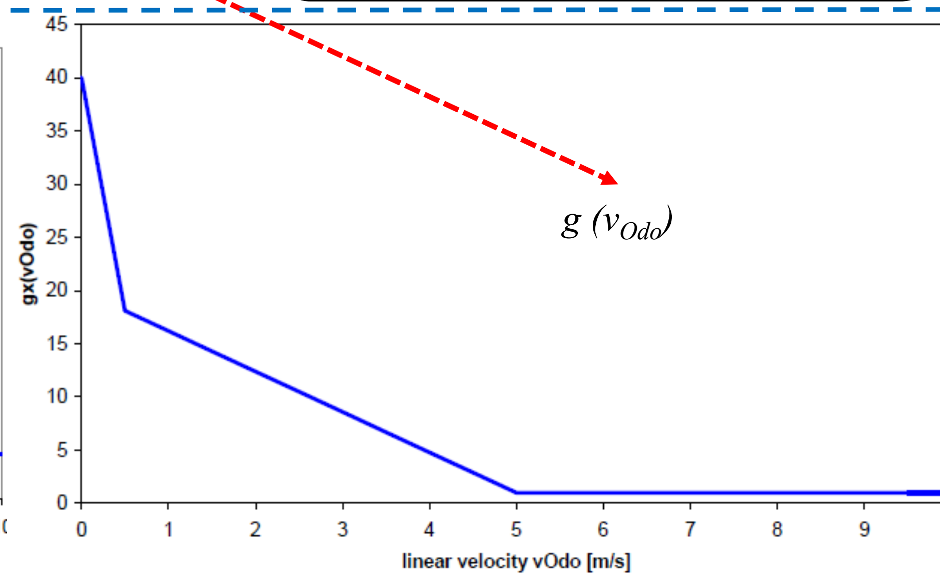
$$\underline{z}_{k,Kal} = \left(x_{Kal}, y_{Kal}, z_{Kal}, \rho_{Kal}\right)^T = \underline{x}_k$$

$$\underline{\sigma}_{k,Kal} = \left(p_x, p_y, p_z, p_\rho\right)^T$$



$f_i(n)$

$g\,(v_{Odo})$



Monte Carlo Localization

1. Resampling: draw a random sample s_{k-1} from the current belief, with a likelihood given by the importance factors of the belief $Bel(s_{k-1})$.

B. Draw a subset of m samples from a Gaussian distribution centered at the filtered GPS position $z_{k,Kal}$ .and the standard deviation $\sigma_{k,Kal}$ of the filtered GPS position.

C. Add the subset of m GPS samples to the sample set representing $Bel(s_k)$ by replacing m samples with the lowest weight.

Sampling

2. Predict: for this sample s_{k-1}, predict a successor pose s_k, according to the motion model $p(s_k \mid s_{k-1}, a_{k-1}, m)$.

Prediction by Odometry

3. Update: assign a preliminary (non-normalized) importance factor $p(o_k \mid s_k, m)$ to this sample and add it to the new sample set representing $Bel(s_k)$.

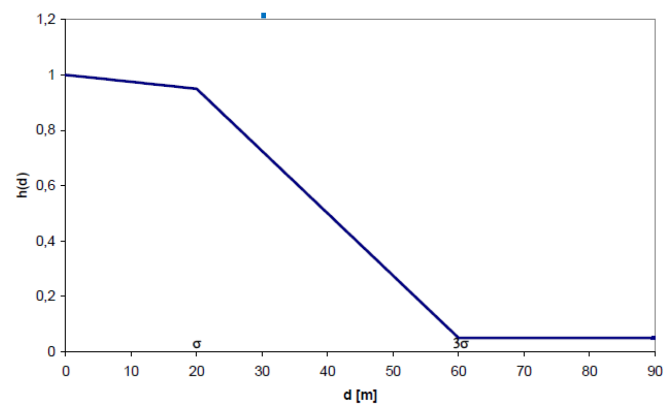
A. Adjust the importance factors of all n samples in dependence of the Euclidian distance between samples s_k and the filtered GPS observation $z_{k,Kal}$.

Update with constraints

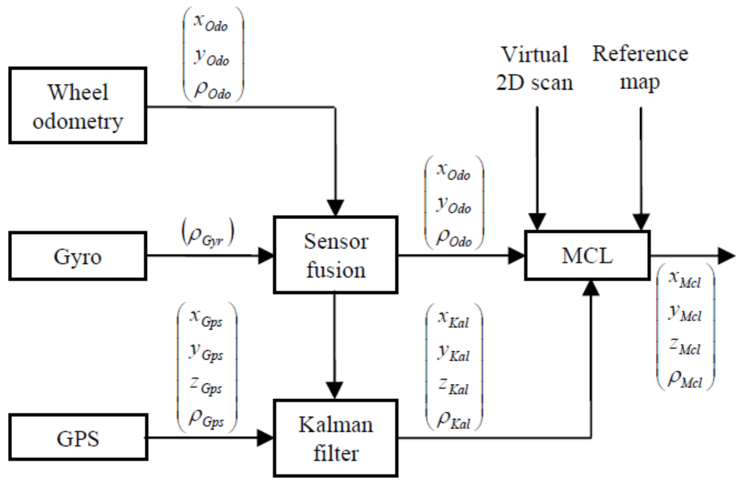
4. Repeat step 1 through 3 n times. Finally, normalize the importance factors in the new sample set $Bel(s_k)$ that they sum up to 1.

$$Bel(s) \approx \left\{s^{(i)}, w^{(i)}\right\}_{i=1,\dots,n}$$

$$w^{(i)} = w^{(i)} \cdot h\left(d^{(i)}\right)$$



Hentschel, M., Wulf, O., & Wagner, B. (2008). A GPS and laser-based localization for urban and non-urban outdoor environments. *Ieee/rsj International Conference on Intelligent Robots and Systems* (Vol.10, pp.149-154). IEEE.



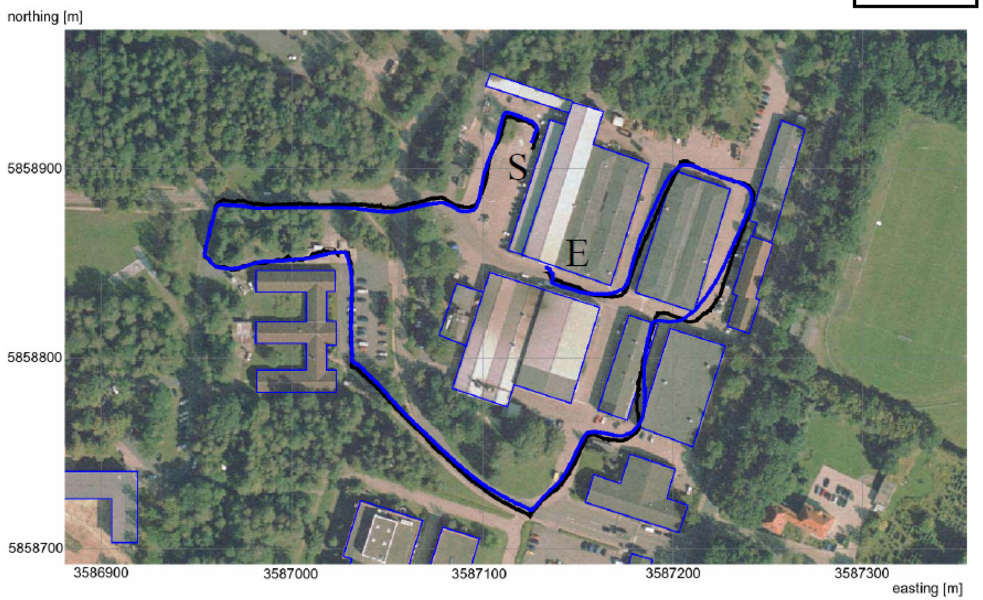


Fig. 6. GPS position (black line) and Kalman filtered GPS positon (blue line) of the experimental test run. The test run start at *S* and ends at the point labelled with *E*.

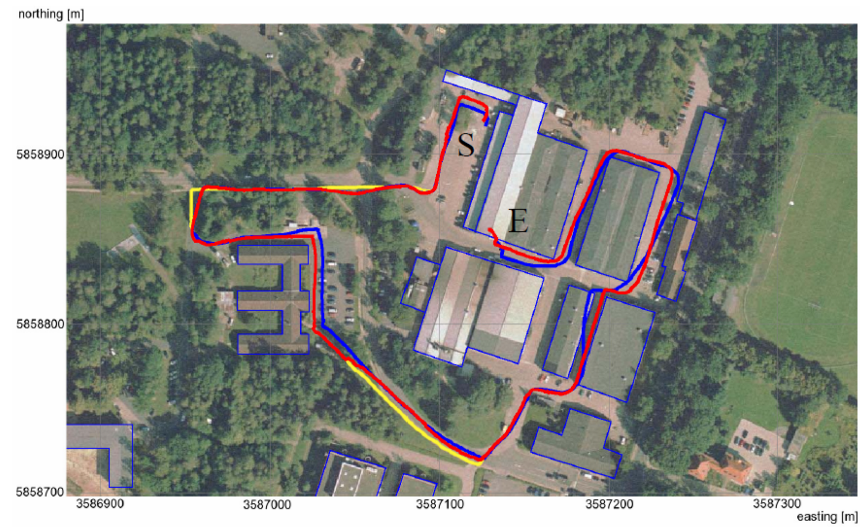


Fig. 7. Kalman filtered GPS position (blue line) and position from Monte Carlo Localization (red line). The actual robot path is indicated in yellow if different from the Monte Carlo Localization. Thin blue lines mark line features of the reference map used for MCL. The test run starts at *S* and ends at the point labelled with *E*.

Hentschel, M., Wulf, O., & Wagner, B. (2008). A GPS and laser-based localization for urban and non-urban outdoor environments. *Ieee/rsj International Conference on Intelligent Robots and Systems* (Vol.10, pp.149-154). IEEE.