



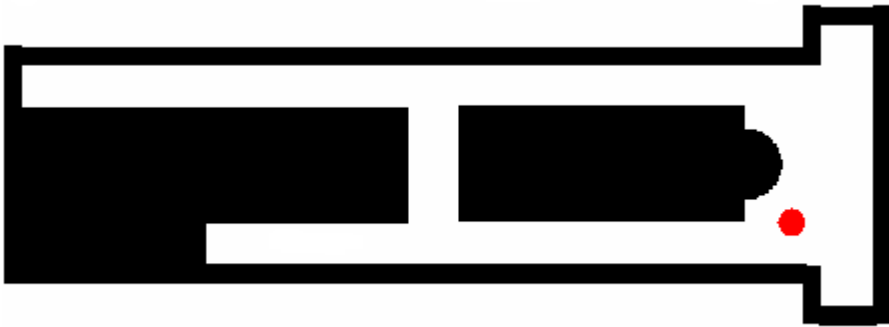
Mobile robot localization

Francesco Martinelli
University of Rome “Tor Vergata”
Roma, Italy

martinelli@disp.uniroma2.it

Objective:

Determination of the pose (= position + orientation) of a mobile robot in a known environment in order to successfully perform a given task



Summary:



- Robot localization [taxonomy](#) (4-5)
- Robot localization as an [estimation problem](#): the [Bayes filter](#) (6-17)
- [Observability](#) issue (18-23)
- Robot description (24)
 - [Motion model](#): the differential drive (25-34)
 - [Measurement model](#): rangefinder or landmarks (35-39)
-
- [Robot localization algorithms](#) (40-42):
 - Gaussian filters: [EKF](#), [UKF](#) (43-60)
 - Non parametric filters: HF, [PF](#) (61-77)
- Examples (78-104)
- Conclusions (105)
- References (106)
- Video



Robot Localization Taxonomy:

- **Position tracking**
- **Global localization**
- **Kidnapped problem**



Increasing degree
of difficulty

Other classifications include:

- type of environment (static or dynamic; completely or partially known)
- active or passive localization
- single or multi robot localization.



Robot Localization Taxonomy:

- Position tracking
- Global localization
- Kidnapped problem



Increasing degree
of difficulty

Other classifications include:

- type of environment (static or dynamic; **completely** or partially **known**)
- active or passive localization
- single or multi robot localization.



Localization as an estimation problem

The robot must infer its pose from available data

Data (noisy):

- Motion information:

→ Proprioceptive sensors (e.g. encoders, accelerometers, etc.)

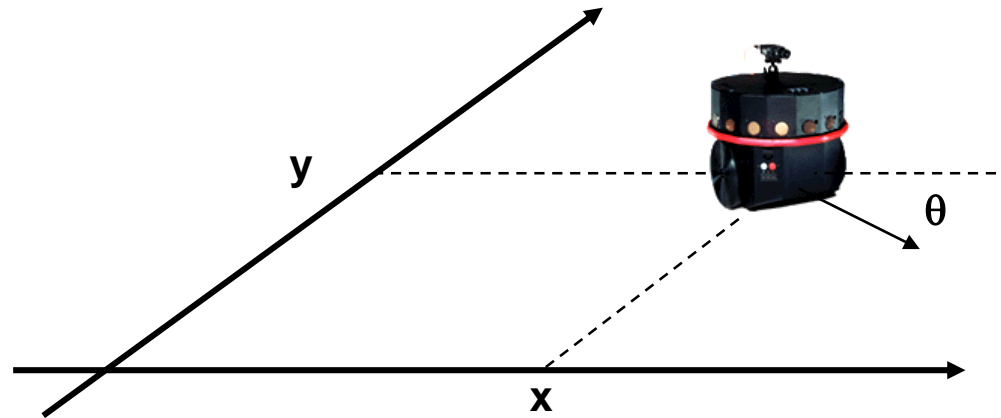
- Environment Measurements

→ Exteroceptive sensors (e.g. laser, sonar, IR, GPS, camera, RFID, etc.)

A filtering approach is required to fuse all information

Notation

$$x_r = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad \text{Robot pose}$$



$$x_{r,0:t} = \{x_{r,0}, x_{r,1}, \dots, x_{r,t}\} \quad \text{Robot poses from time 0 to time t}$$

$$z_{1:t} = \{z_1, z_2, \dots, z_t\} \quad \text{Robot exteroceptive measurements from time 1 to time t}$$

$$u_{0:t} = \{u_0, u_1, \dots, u_t\} \quad \text{Motion commands (or proprioceptive measurements) from time 0 to time t}$$

Belief of the robot at time t: probability density function (pdf) describing the information the robot has regarding its pose at time t, based on all available data (exteroceptive measurements and motion commands):

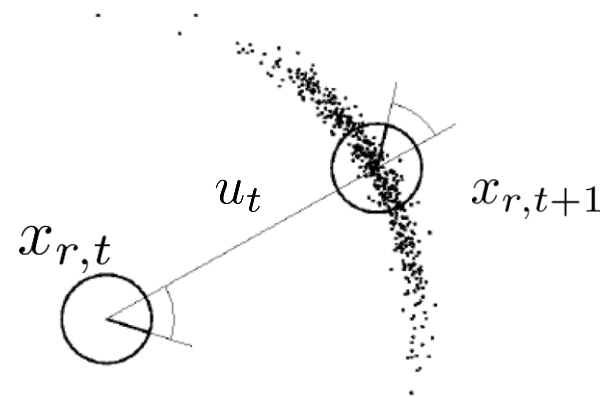
$$bel_t(x_r) = p(x_{r,t} = x_r | z_{1:t}, u_{0:t-1})$$

Prior belief of the robot at time t: pdf before acquiring the last measurement z_t :

$$\overline{bel}_t(x_r) = p(x_{r,t} = x_r | z_{1:t-1}, u_{0:t-1})$$

The robot motion model is the pdf of the robot pose at time $t+1$ given the robot pose and the motion action at time t . It takes into account the noise characterizing the proprioceptive sensors:

$$p(x_{r,t+1} | x_{r,t}, u_t)$$



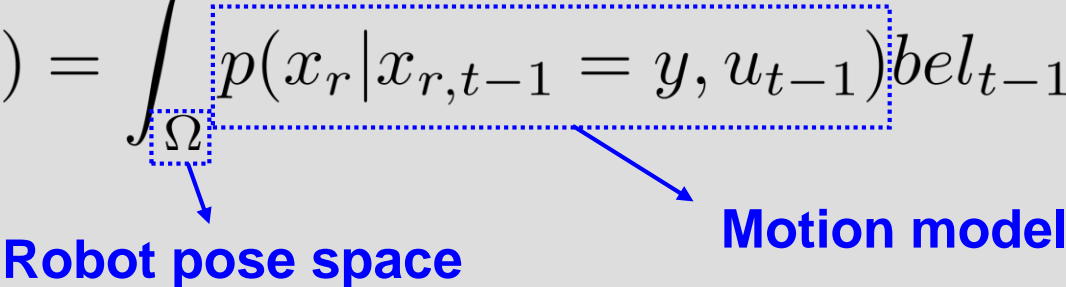
The measurement model describes the probability of observing at time t a given measurement z_t when the robot pose is $x_{r,t}$. It takes into account the noise characterizing the exteroceptive sensors:

$$p(z_t | x_{r,t})$$



The Bayes Filter

Prediction

$$\overline{bel}_t(x_r) = \int_{\Omega} p(x_r | x_{r,t-1} = y, u_{t-1}) bel_{t-1}(y) dy$$


Robot pose space

Motion model

Based on the total probability theorem:

$$prob(A) = \sum_i prob(A|B_i)prob(B_i) \quad (\text{discrete case})$$

where $B_i, i=1,2,\dots$ is a partition of Ω . In the continuous case:

$$p(x) = \int p(x|y)p(y)dy$$

Illustration of the prediction step in an ideal case

Assume a robot moving along a unidimensional space, i.e. $x_r = x$, and assume an ideal motion model:

$$x_t = x_{t-1} + u_{t-1}$$

$$\longrightarrow p(x_t = x | x_{t-1} = y, u_{t-1}) = \delta(x - y - u_{t-1})$$

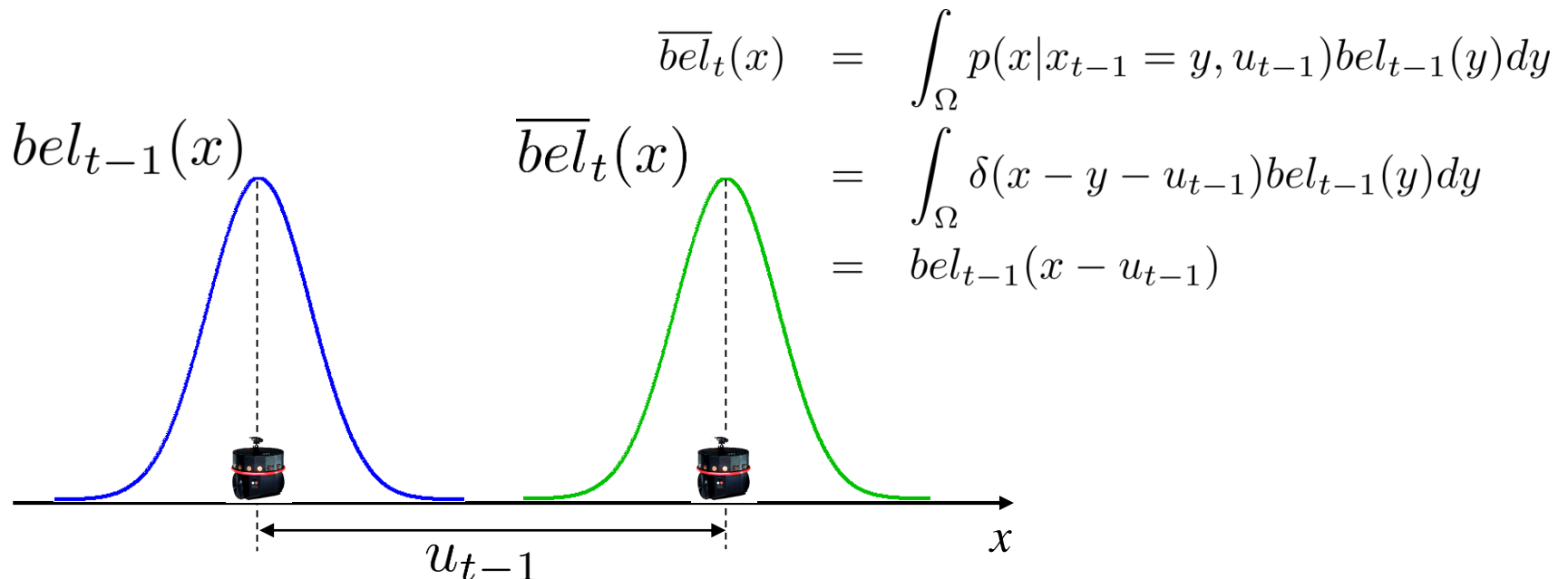


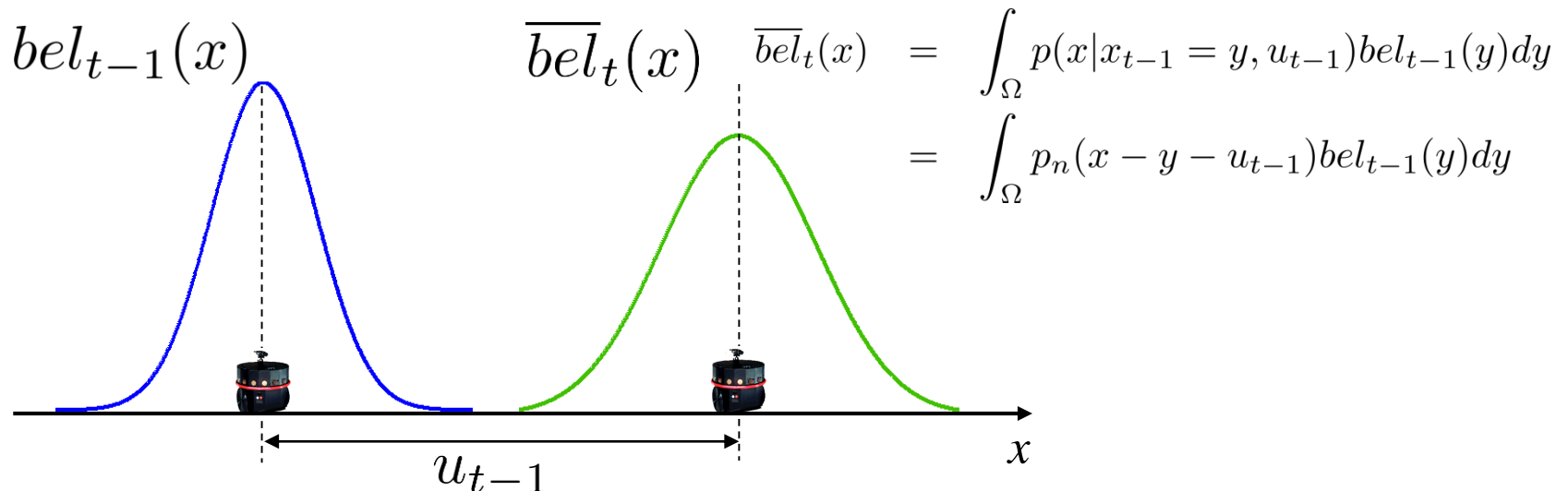
Illustration of the prediction step in a realistic case

Still a unidimensional space but consider a more realistic motion model:

$$x_t = x_{t-1} + u_{t-1} + n_{t-1}$$

where the noise $n_t \sim p_n(n_t)$

$$\longrightarrow p(x_t = x | x_{t-1} = y, u_{t-1}) = p_n(x - y - u_{t-1})$$



Correction or Measurement Update

$$\begin{aligned}
 bel_t(x_r) &= p(x_{r,t} = x_r | z_{1:t}, u_{0:t-1}) \\
 &= \eta p(z_t | x_{r,t} = x_r) \overline{bel}_t(x_r)
 \end{aligned}$$

Normalizing factor
Measurement model

Based on the Bayes Rule: $p(A|B) = p(B|A) \frac{p(A)}{p(B)}$

i.e. also: $p(A|B, C) = p(B|A, C) \frac{p(A|C)}{p(B|C)}$

Taking: $A := \{x_{r,t} = x_r\}$ $B := \{z_t\}$ $C := \{z_{1:t-1}, u_{0:t-1}\}$

We have:

$$\begin{aligned}
 bel_t(x_r) &= p(x_{r,t} = x_r | z_{1:t}, u_{0:t-1}) \\
 &= p(z_t | x_{r,t} = x_r, z_{1:t-1}, u_{0:t-1}) \frac{p(x_{r,t} = x_r | z_{1:t-1}, u_{0:t-1})}{p(z_t | z_{1:t-1}, u_{0:t-1})}
 \end{aligned}$$

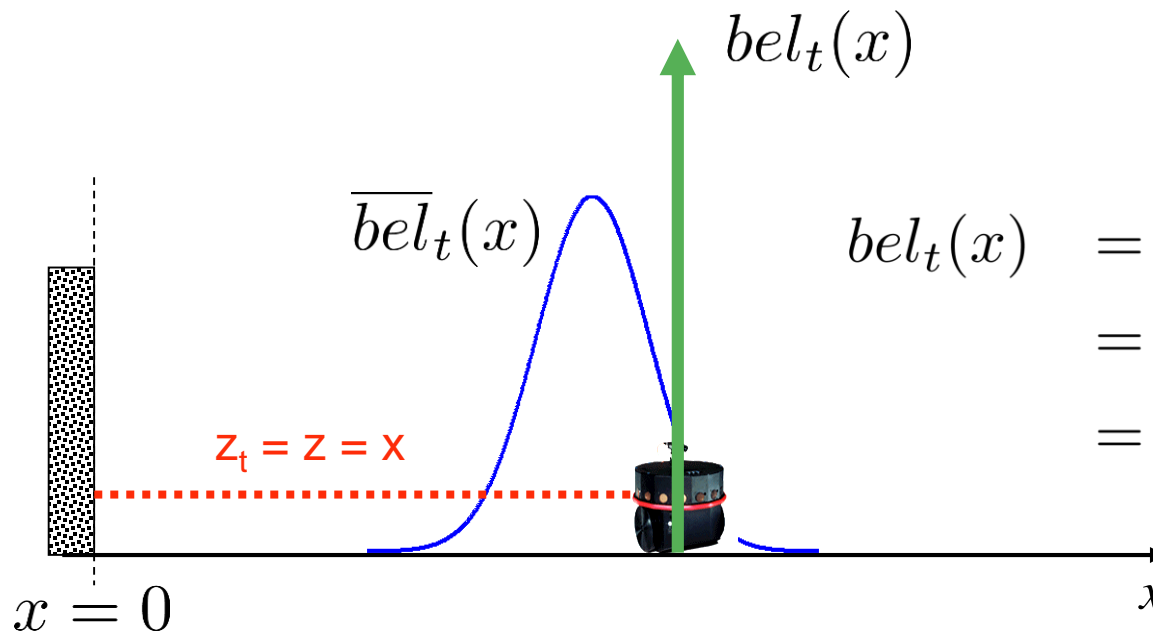
$p(z_t | x_{r,t} = x_r)$
 $1/\eta$
 $\overline{bel}_t(x_r)$

Illustration of the correction step in an ideal case

Assume a robot moving along a unidimensional space, i.e. $x_r = x$, and measuring the distance to a landmark placed in the origin. Under an ideal measurement model:

$$z_t = x_t$$

$$\longrightarrow p(z_t = z | x_t = x) = \delta(x - z)$$



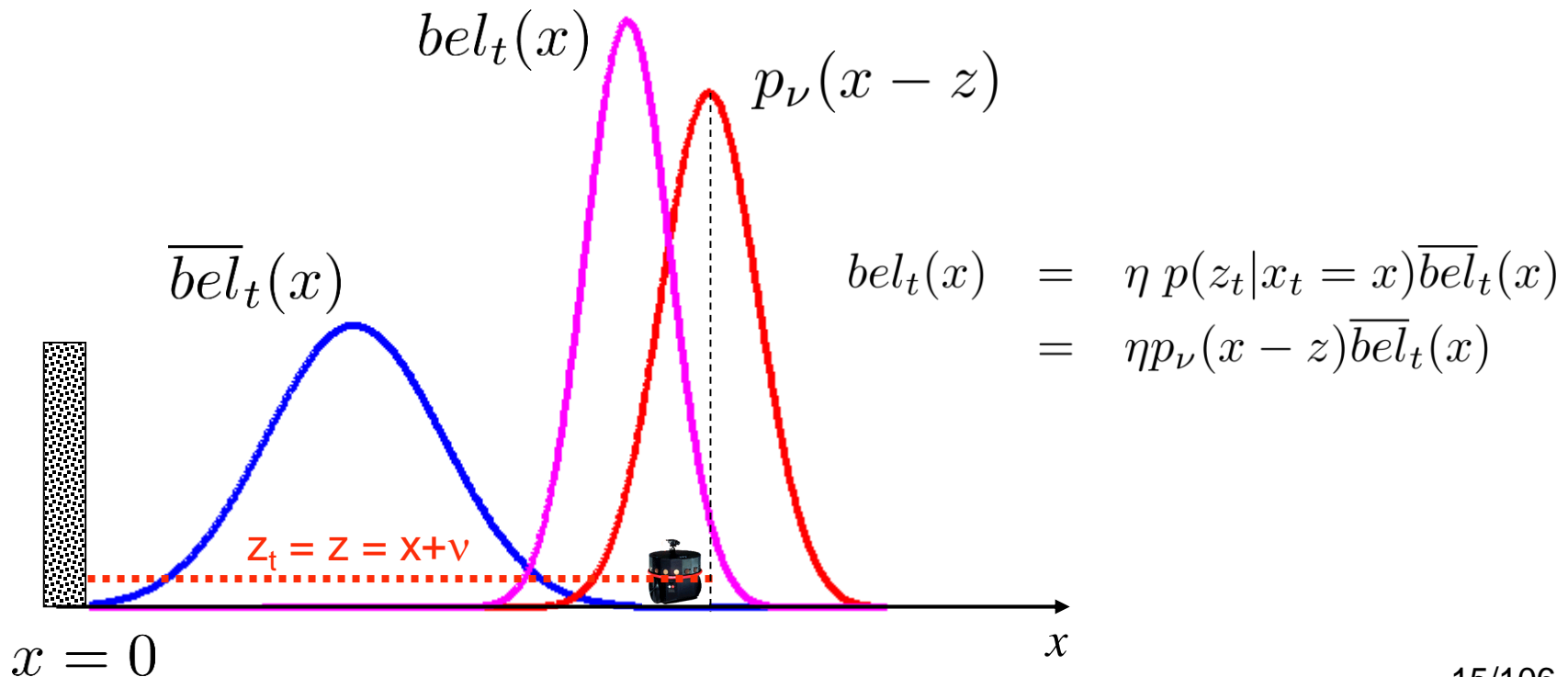
$$\begin{aligned} bel_t(x) &= \eta p(z_t | x_t = x) \overline{bel}_t(x) \\ &= \eta \delta(x - z) \overline{bel}_t(x) \\ &= \delta(x - z) \end{aligned}$$

Illustration of the correction step in a realistic case

Still a unidimensional space, still the robot is measuring the distance to a landmark placed in the origin. Under a realistic measurement model:

$$z_t = x_t + \nu_t \quad \text{where the noise } \nu_t \sim p_\nu(\nu_t)$$

$$\longrightarrow p(z_t = z | x_t = x) = p_\nu(x - z)$$



The Bayes Filter

It is a recursive algorithm. At time t , given the belief at time $t-1$ $bel_{t-1}(x_r)$, the last motion control u_{t-1} and the last measurement z_t , determine the new belief $bel_t(x_r)$ as follows:

For all $x_r \in \Omega$ do the following:

Motion model

$$\overline{bel}_t(x_r) = \int_{\Omega} p(x_r | x_{r,t-1} = y, u_{t-1}) bel_{t-1}(y) dy$$

Measurement model

$$bel_t(x_r) = \eta p(z_t | x_{r,t} = x_r) \overline{bel}_t(x_r)$$

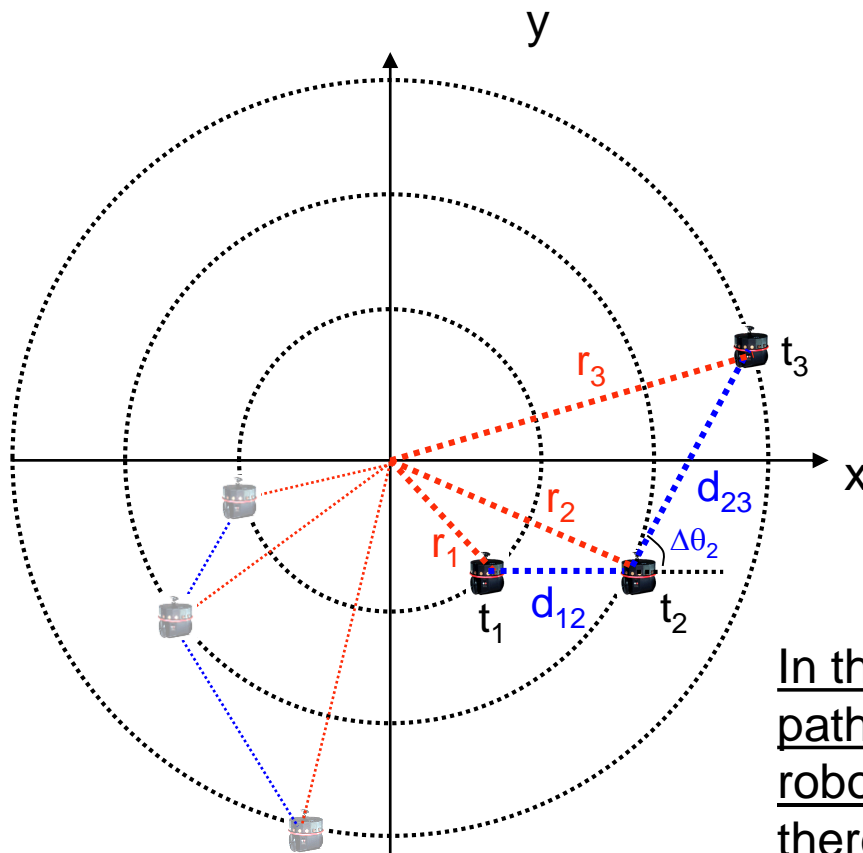
The Bayes Filter - Observations

1. The Bayes filter is not a practical algorithm
 - Gaussian Filters: Extended Kalman Filter (EKF),
Unscented Kalman Filter (UKF),
Extended Information Filter (EIF)
 - Non Parametric Filters: Histogram Filter (HF),
Particle Filter (PF)
2. Important to check if we have enough information to localize the robot (observability issue).

Sometimes the observability can be evaluated through the observability rank condition (see Hermann&Krener, TAC 1977)

Observability issue

The robot may have not enough sensors to determine its pose, even if assuming perfect (i.e. without noise) measurements.



The robot is equipped with encoder readings and can measure the distance r to a landmark located in the origin.

r_i = range measurement at time t_i

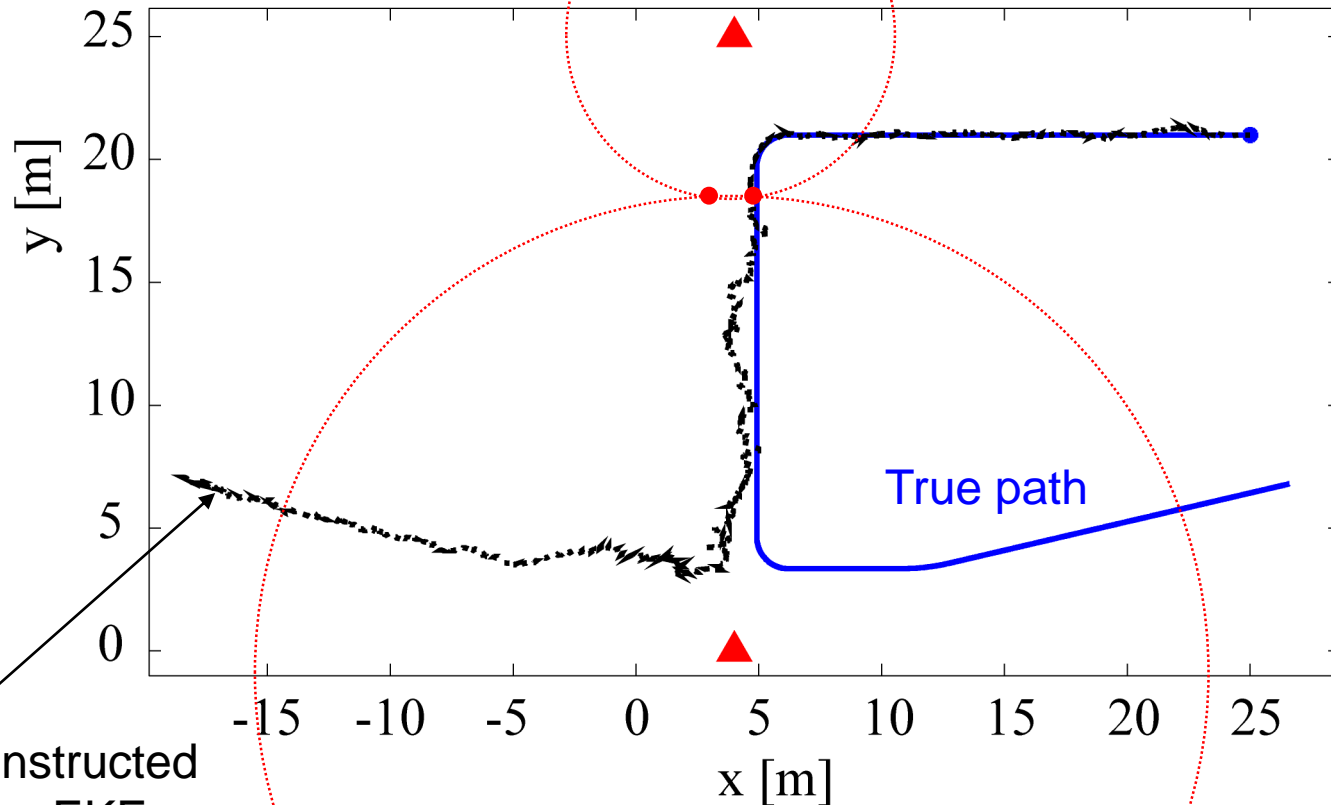
d_{ij} = distance covered (and measured) by the robot in time interval $[t_i, t_j]$

$\Delta\theta_2$ = robot turn at time t_2 (known from odometry)

In the figure we represent two different robot paths agreeing with measurements. The robot pose **is not completely observable**: there is a radial symmetry around the origin

Case the robot measures (with noise) the distance from two landmarks (in addition to the odometry). The robot pose **is not completely observable**.

Observability issue



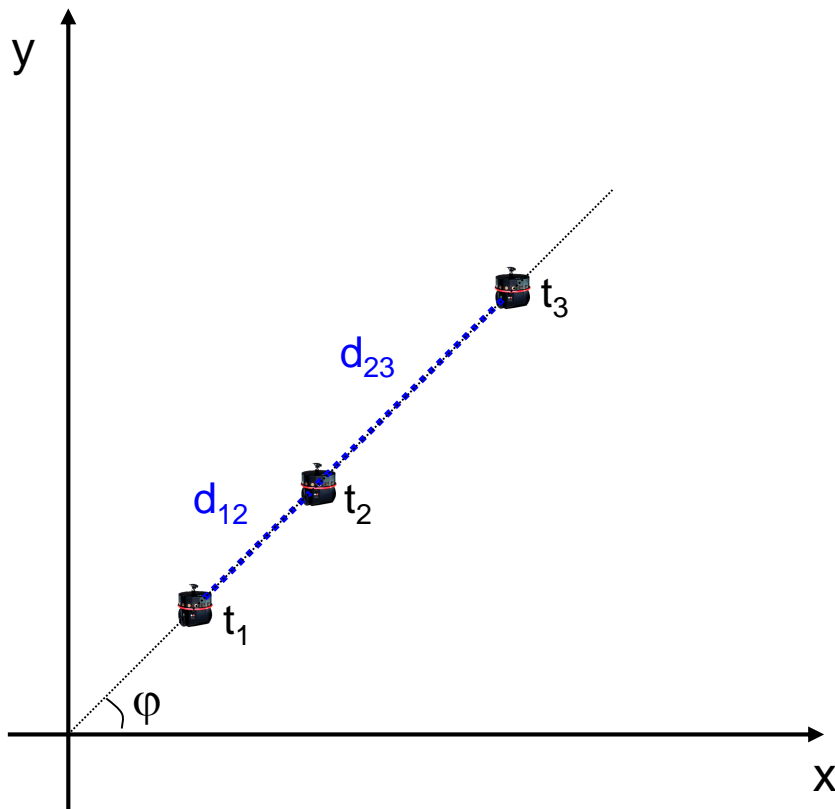
Path reconstructed
through the EKF
(position tracking)

The noise makes the estimated pose jump to the specular path.

In the nonlinear case, the observability depends on the particular control input considered.

Hence, the possibility of localizing a mobile robot may depend on the particular motion executed which in any case affects the localization performance

→ Active Localization

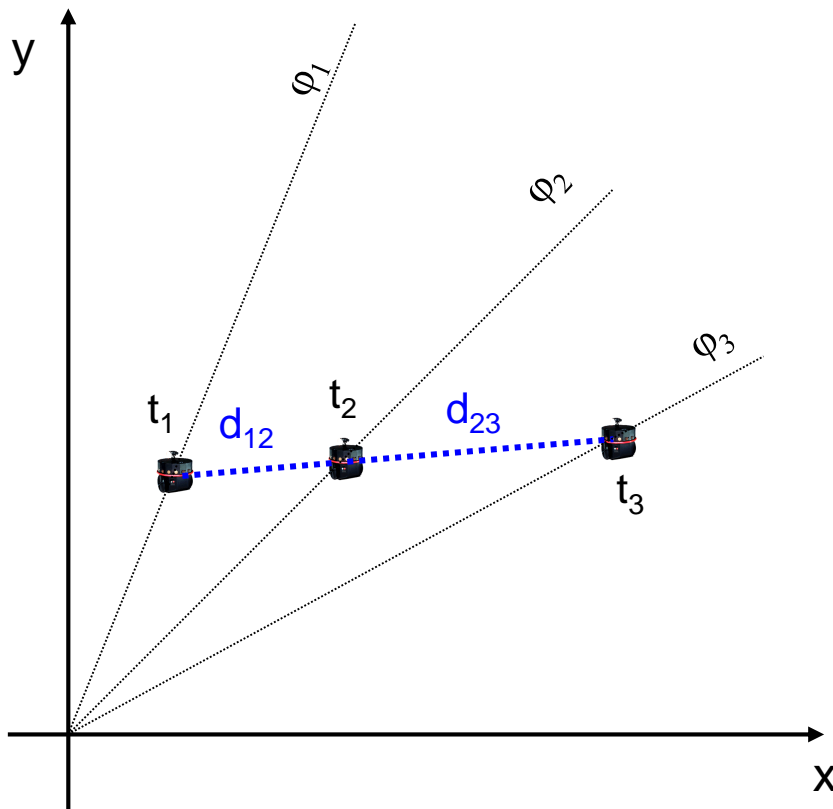


The robot is equipped with odometry and measures the angle φ (a non symmetric landmark is located in the origin).

d_{ij} = distance covered (and measured) by the robot in time interval $[t_i, t_j]$

Under this motion (radial) the distance of the robot from the origin (hence x and y) **cannot be determined** but only the orientation $\theta = \varphi$

A different kind of motion would make the pose observable (active localization):



The robot is equipped with odometry and measures the angle ϕ (a non symmetric landmark is located in the origin).

d_{ij} = distance covered (and measured) by the robot in time interval $[t_i, t_j]$

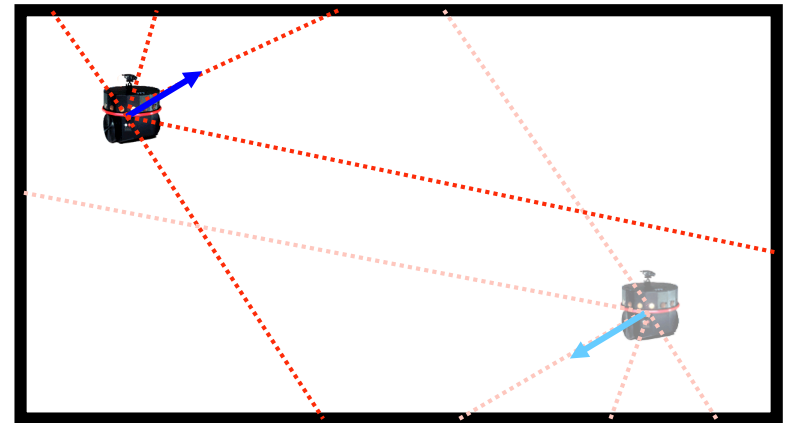
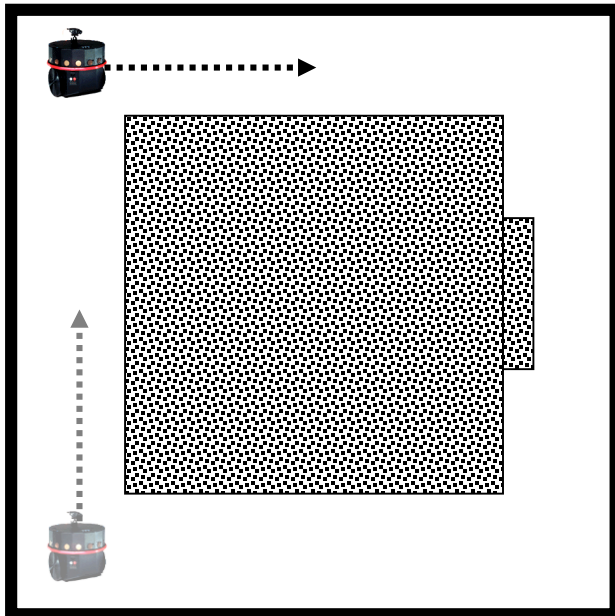
Under this motion all the pose of the robot **can be determined**

Often localization is performed under a generic motion (passive localization): the robot tries to localize itself while performing its mission (once a satisfactory pose estimation has been reached) or under a random motion away from obstacles (at the beginning, when the pose is still completely unknown).

This is because usually mobile robots are equipped with several sensors: they can often localize themselves even if not moving (e.g. if measuring the distance to at least 3 non collinear landmarks).

Nevertheless...

problems may arise with symmetrical environments (the robot has odometry + rangefinder, e.g. sonar or laser).

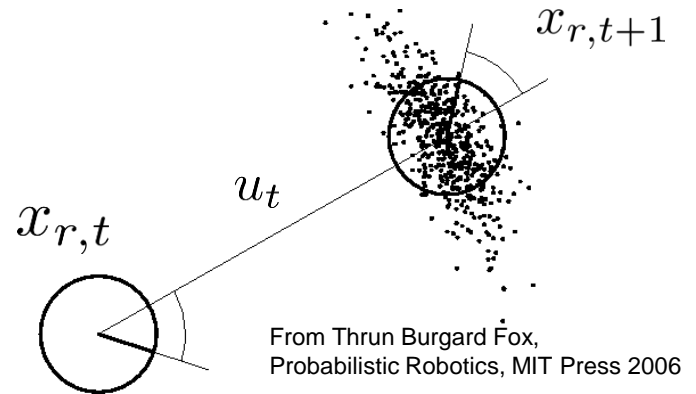


A proper robot motion may disambiguate in some cases the pose estimation.

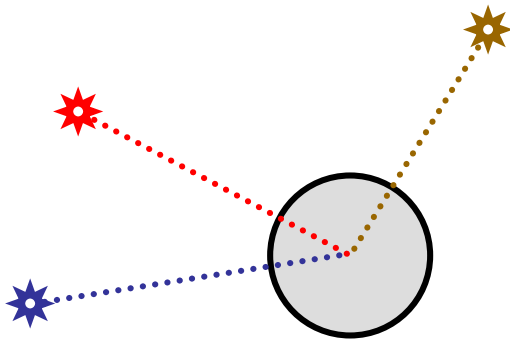
Robot description

- Motion model (proprioceptive sensors)

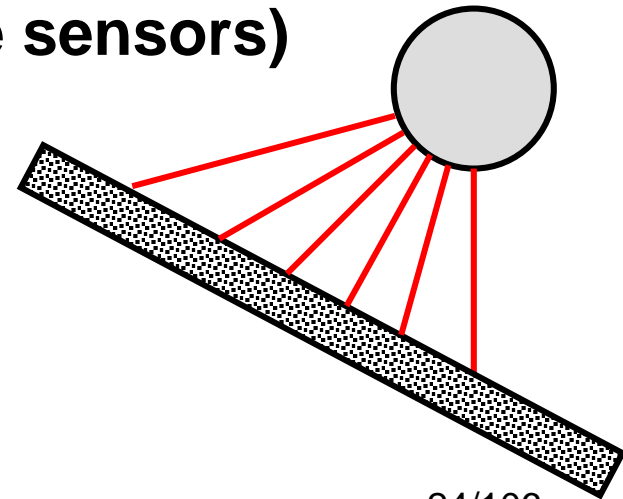
$$p(x_{r,t+1} | x_{r,t}, u_t)$$



- Measurement model (exteroceptive sensors)



$$p(z_t | x_{r,t})$$



Motion model (proprioceptive sensors)

$$p(x_{r,t+1} | x_{r,t}, u_t) = \text{prob}(x_{r,t+1} = x_r | x_{r,t}, u_t)$$

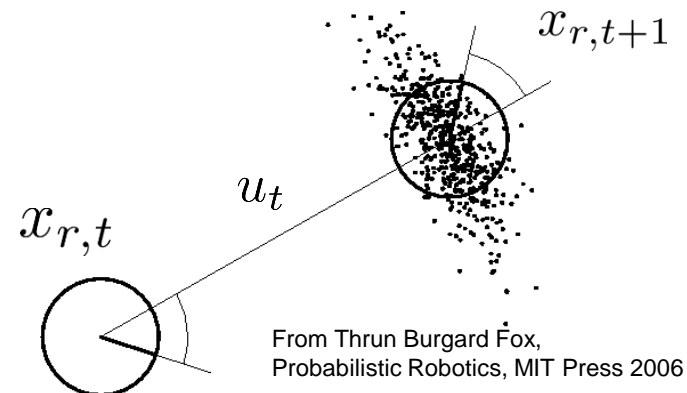
Assume:

- $x_{r,t}$ robot pose at time t
- u_t control input (i.e. motion command) at time t

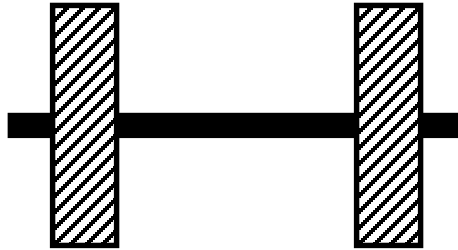
Which is the pose at time $t+1$?

Usually, u_t is obtained through proprioceptive sensors (e.g. encoders, gyroscopes, etc) → Odometry motion model

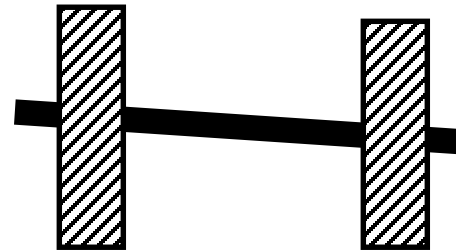
Due to sensor noise, $x_{r,t+1}$ is available only through a probability density function.



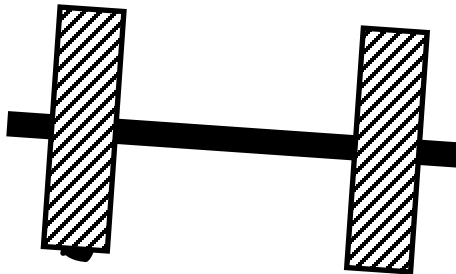
Possible sources of noise



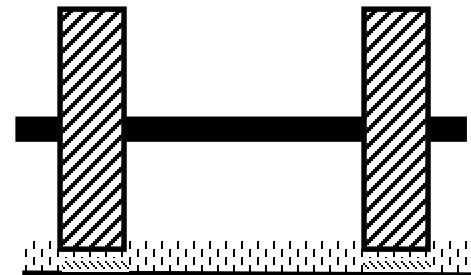
Ideal case



Different wheel
diameters (systematic source of noise)



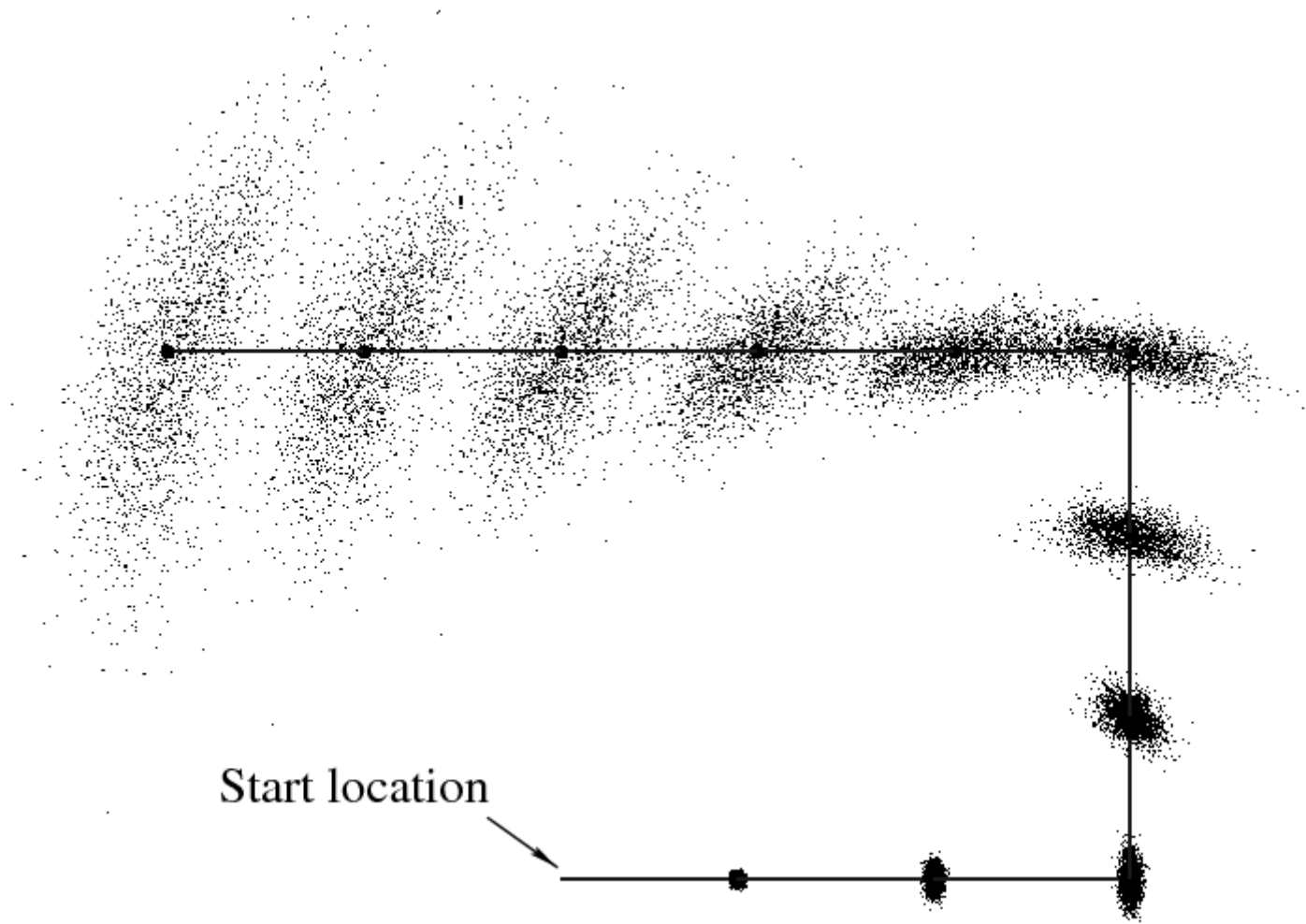
Bump (casual source of noise)



Carpet (casual source of noise)

and many more (e.g. distance between the wheels) ...

Accumulation of the pose estimation error under the robot motion
(only proprioceptive measurements)



Discretized unicycle model:

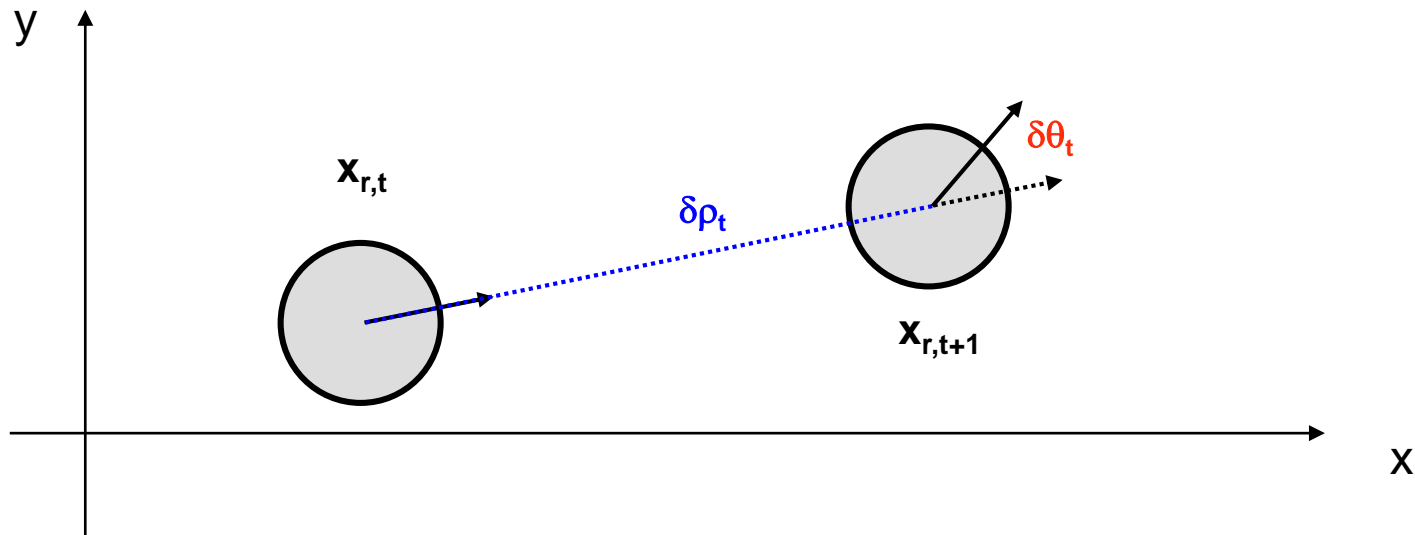
$$x_{t+1} = x_t + \boxed{\delta\rho_t} \cos(\theta_t)$$

$$y_{t+1} = y_t + \delta\rho_t \sin(\theta_t)$$

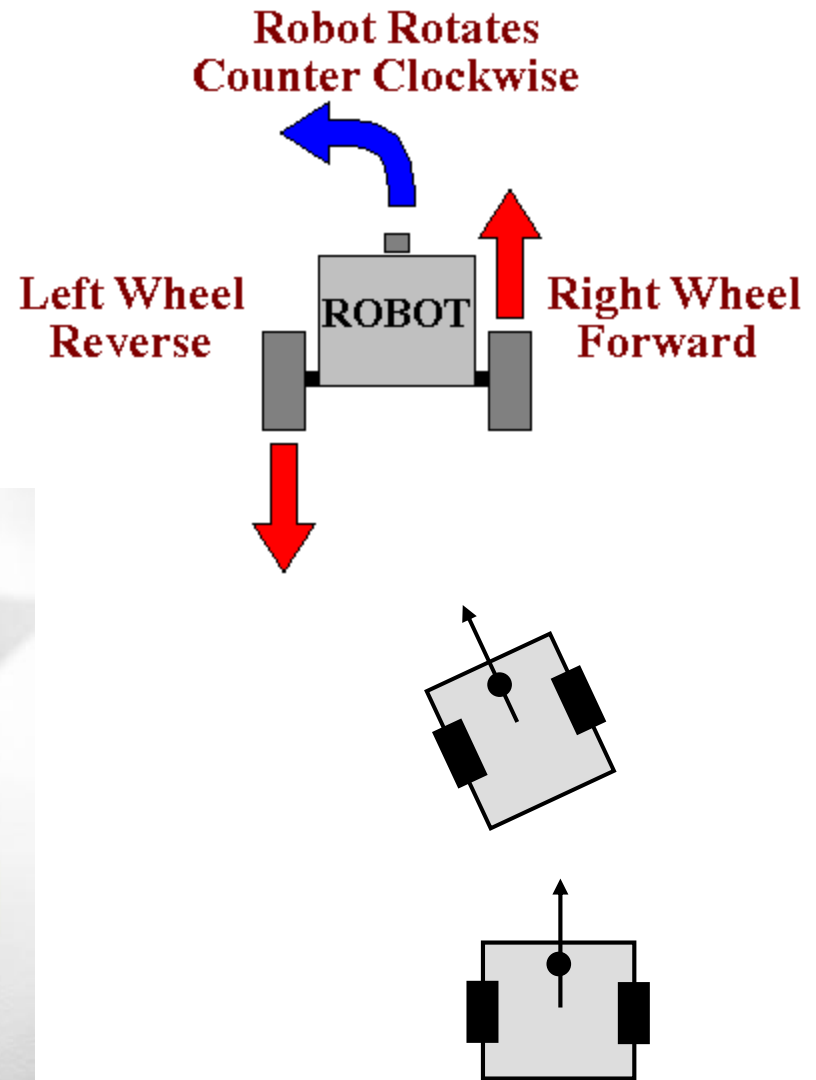
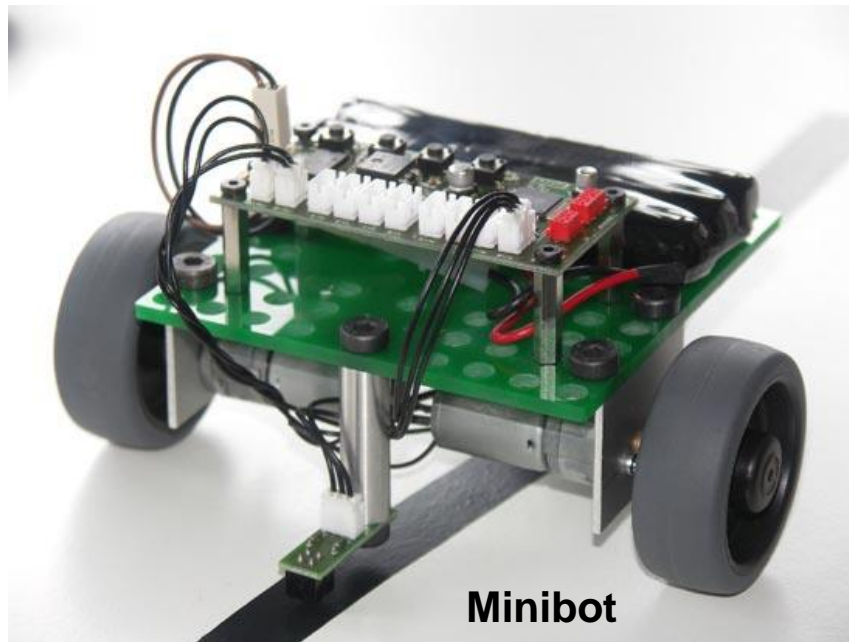
$$\theta_{t+1} = \theta_t + \boxed{\delta\theta_t}$$

Robot shift in time interval $[t, t+1]$

Robot rotation in time interval $[t, t+1]$



Discretized unicycle model: the differential drive case

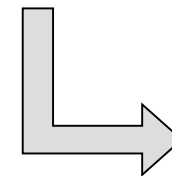




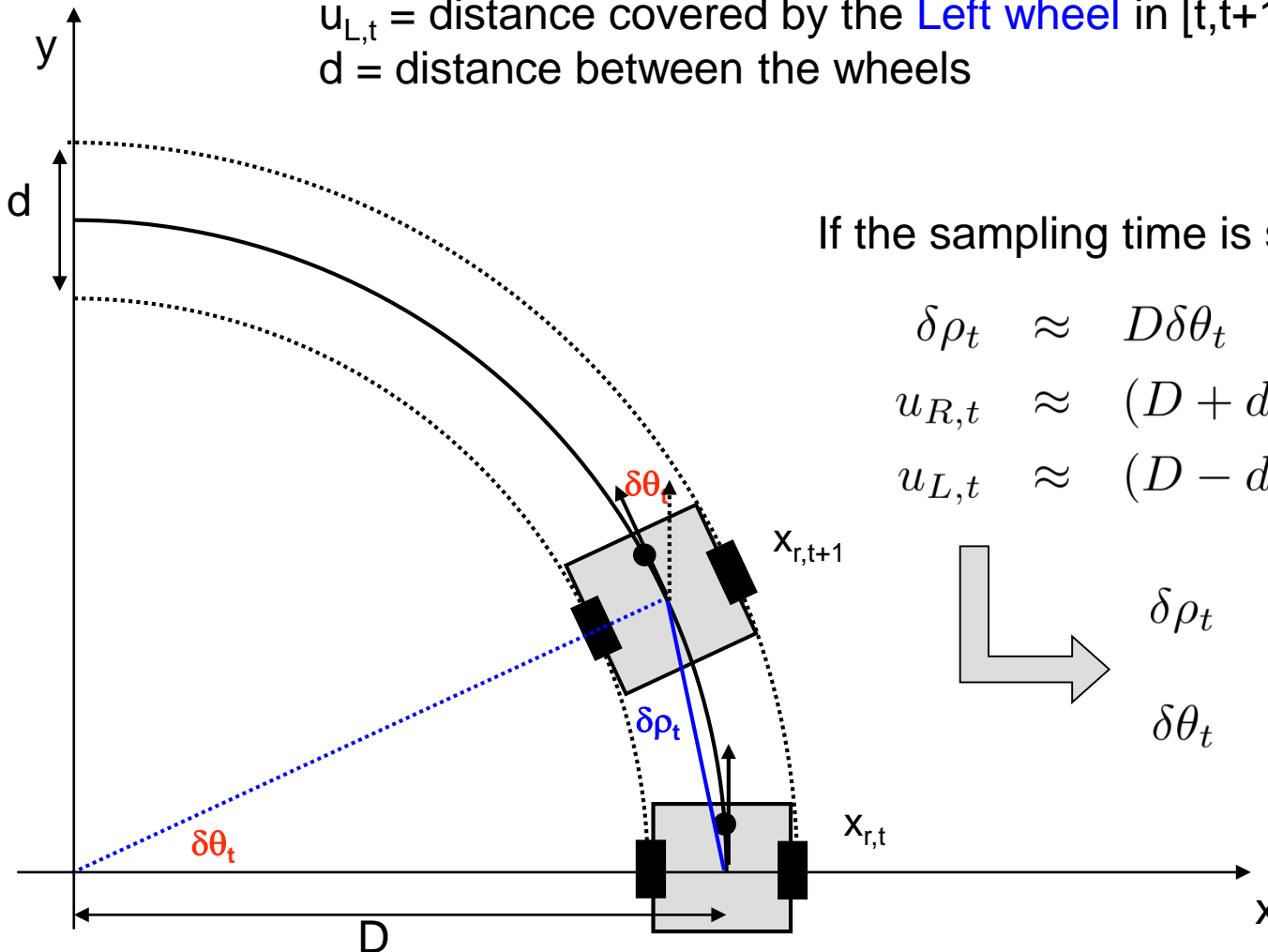
$u_{R,t}$ = distance covered by the **Right wheel** in $[t, t+1]$
 $u_{L,t}$ = distance covered by the **Left wheel** in $[t, t+1]$
 d = distance between the wheels

If the sampling time is small enough:

$$\begin{aligned}\delta\rho_t &\approx D\delta\theta_t \\ u_{R,t} &\approx (D + d/2)\delta\theta_t \\ u_{L,t} &\approx (D - d/2)\delta\theta_t\end{aligned}$$



$$\begin{aligned}\delta\rho_t &\approx \frac{u_{R,t} + u_{L,t}}{2} \\ \delta\theta_t &\approx \frac{u_{R,t} - u_{L,t}}{d}\end{aligned}$$



There is a **bijjective relation** between $(\delta\rho, \delta\theta)$ and (u_R, u_L) :

$$\begin{aligned} \delta\rho_t &= \frac{u_{R,t} + u_{L,t}}{2} \\ \delta\theta_t &= \frac{u_{R,t} - u_{L,t}}{d} \end{aligned} \quad \longleftrightarrow \quad \begin{aligned} u_{R,t} &= \delta\rho_t + \frac{d}{2} \delta\theta_t \\ u_{L,t} &= \delta\rho_t - \frac{d}{2} \delta\theta_t \end{aligned}$$

Mismatch between **real** $(u_{R,t}, u_{L,t})$ and **measured** $(u_{R,t}^e, u_{L,t}^e)$ distances covered by the wheels of the robot. The encoder readings are related to the real distances covered by the two wheels as follows:

$$u_{R,t}^e = u_{R,t} + n_{R,t}$$

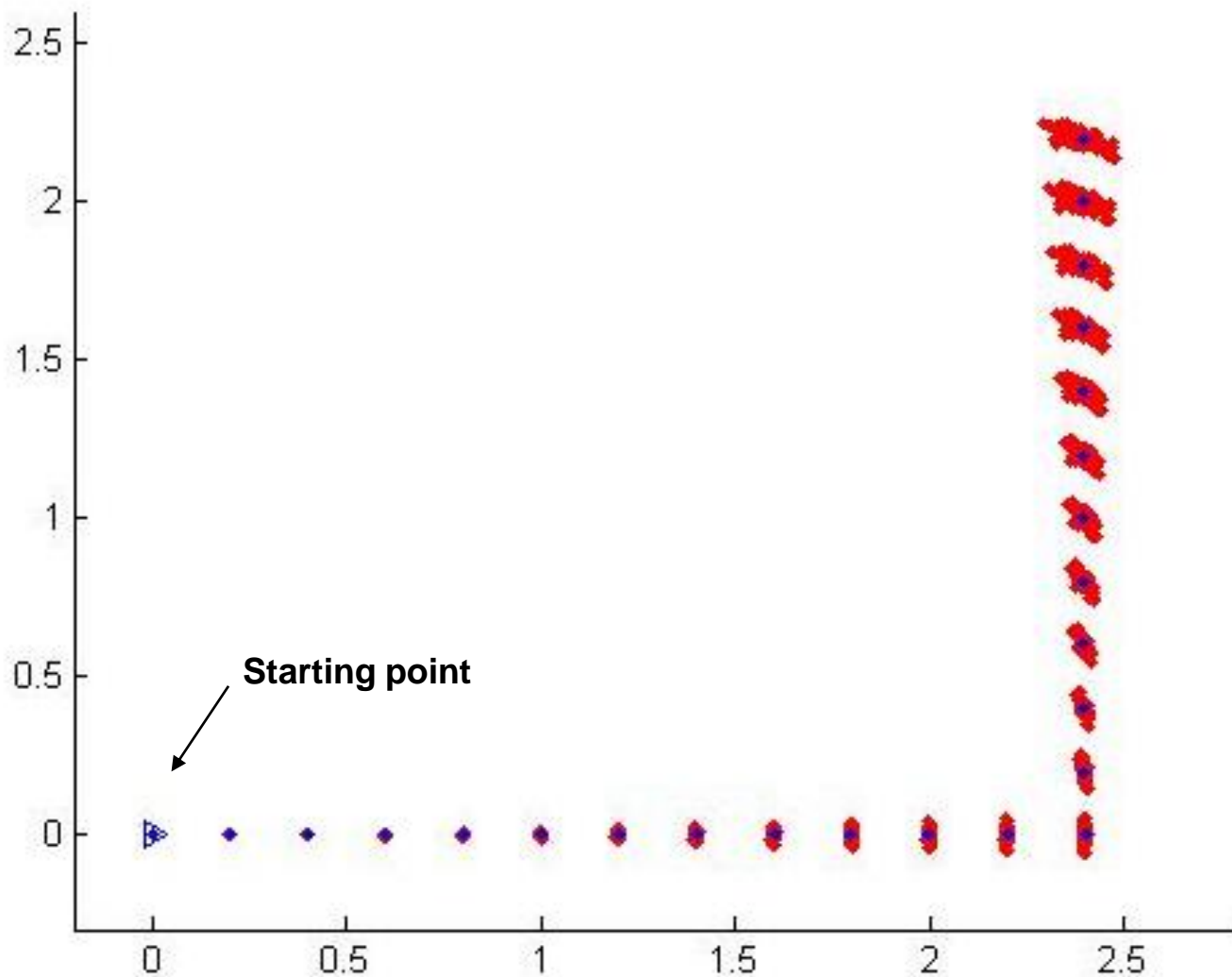
$$u_{L,t}^e = u_{L,t} + n_{L,t}$$

(**casual** source of noise)

$$n_{R,t} \sim \mathcal{N}(0, K_R |u_{R,t}|)$$

$$n_{L,t} \sim \mathcal{N}(0, K_L |u_{L,t}|)$$

Estimation (red points) of the path covered by a differential drive robot.
Blue = real position in each step.



Given:

- $x_{r,t} = [x, y, \theta]^T$ robot pose at time t ,
- encoder reading $u_t = [u_R^e, u_L^e]^T$ (i.e. result of the control input) relative to the interval $[t, t+1]$,

we want to estimate the probability that the robot pose at time $t+1$ will be a given $x_{r,t+1} = [x', y', \theta']^T$ (under the differential drive model considered), i.e.:

$$p(x_{r,t+1} | x_{r,t}, u_t) = p([x', y', \theta']^T | [x, y, \theta]^T, (u_R^e, u_L^e))$$

We proceed as follows:

Actual robot displacement if we want to arrive at $[x', y', \theta']^T$ from $[x, y, \theta]^T$:

$$\begin{aligned}\delta\rho &= \sqrt{(x - x')^2 + (y - y')^2} \\ \delta\theta &= \theta' - \theta\end{aligned}$$

Corresponding distances covered by the two wheels, while the actual encoder readings are (u_R^e, u_L^e) :

$$\begin{aligned}u_R &= \delta\rho + \frac{d}{2} \delta\theta \\ u_L &= \delta\rho - \frac{d}{2} \delta\theta\end{aligned}$$

The probability of arriving in $[x', y', \theta']^T$ from $[x, y, \theta]^T$ is the probability of reading (u_R^e, u_L^e) when the actual wheel shifts are (u_R, u_L)

$$\begin{aligned}p([x', y', \theta']^T | [x, y, \theta]^T, (u_R^e, u_L^e)) \\ = \text{prob}(n_{R,t} = u_R - u_R^e) \cdot \text{prob}(n_{L,t} = u_L - u_L^e)\end{aligned}$$

Measurement model (exteroceptive sensors)

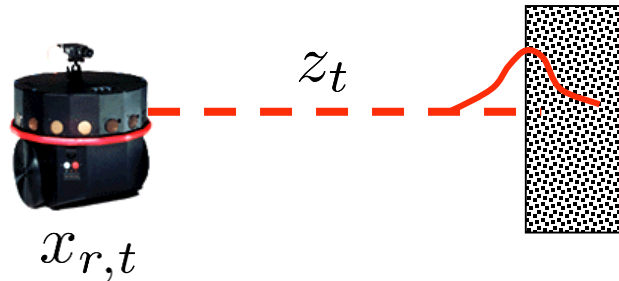
$$p(z_t | x_{r,t})$$

Assume:

- $x_{r,t}$ robot pose at time t

Which is the reading z_t of a sensor at time t ?

Due to sensor noise, z_t is available only through a probability density function.

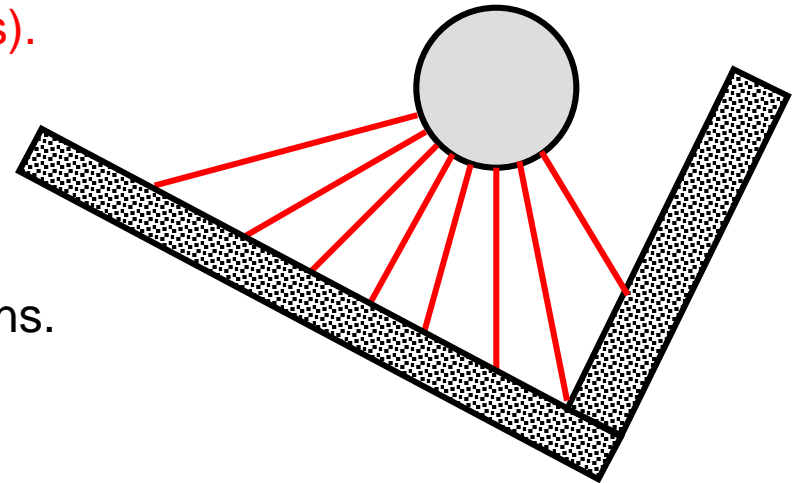


Measurement model (exteroceptive sensors)

Two typical settings are considered:

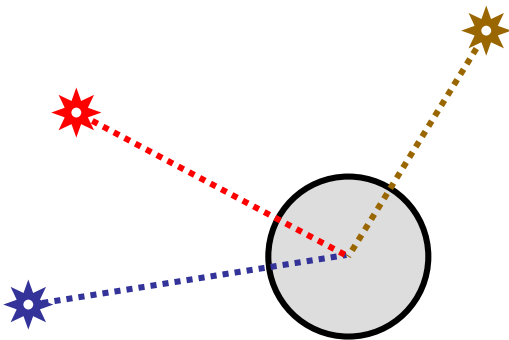
Range Finder (e.g. laser or sonar readings).

The environment is known and is represented by line segments (e.g. the walls). The robot senses the distance from them at some given directions.



Landmark measurements.

The environment is characterized by features (artificial or natural landmarks), whose position is known and whose identity can be known (e.g. RFID tags) or must be estimated (data association problem). The robot measures the distance and/or the bearing to these fixed points.



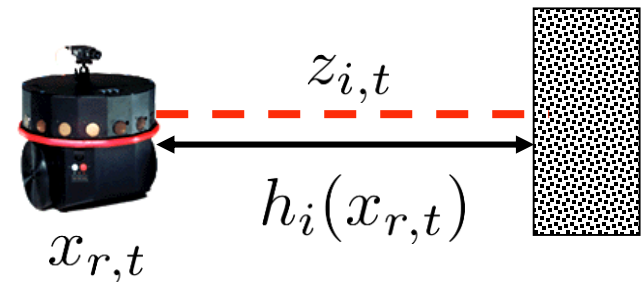
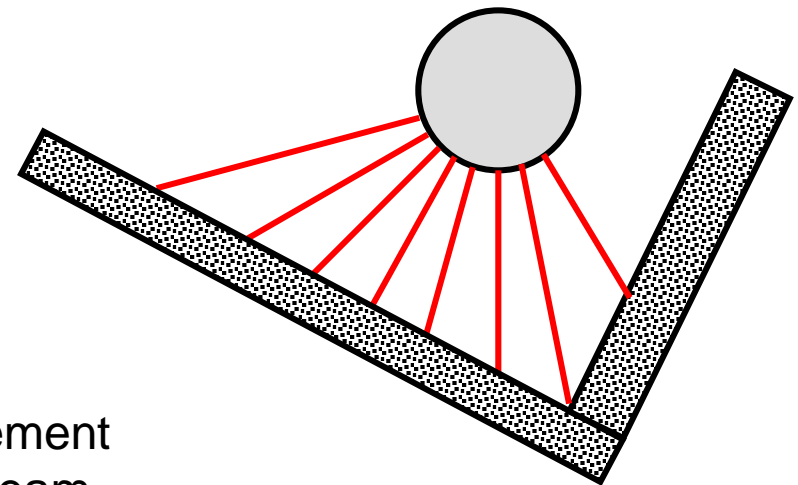
Measurement model: the Range Finder case

$$z_t = [z_{1,t}, z_{2,t}, \dots, z_{m,t}]^T$$

Each measurement $z_{i,t}$ is a range measurement typically based on the time of flight of the beam (e.g. of a sonar or a laser) generated by sensor i and reflected by an obstacle

$z_{i,t}$ Measurement given by beam i

$h_i(x_{r,t})$ Distance of sensor i from the reflecting obstacle (ideal reading of the sensor)



$$z_{i,t} = h_i(x_{r,t}) + \nu_{i,t}$$
$$\nu_{i,t} \sim \mathcal{N}(0, \zeta_i^2)$$

Given:

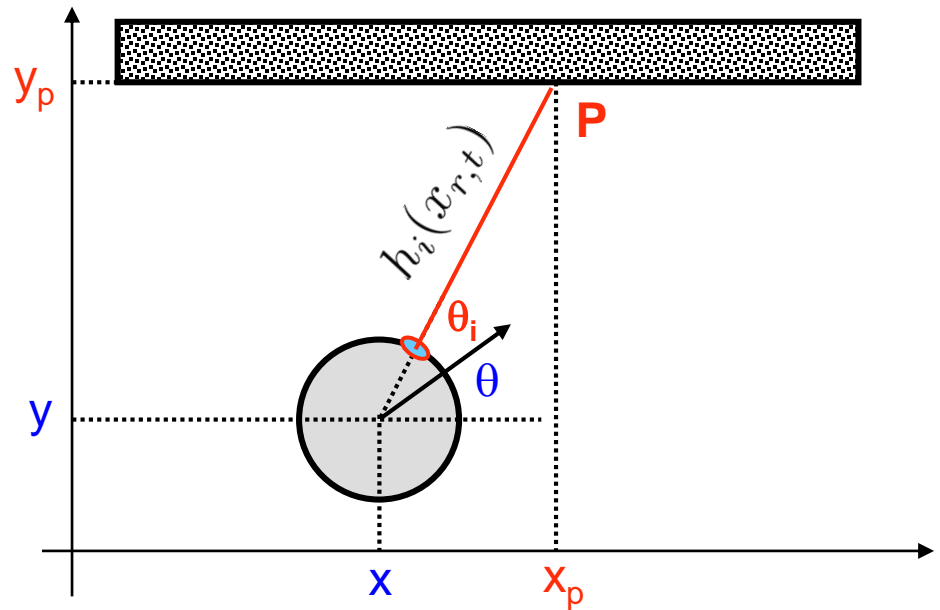
- $x_{r,t} = [x, y, \theta]^T$ robot pose at time t ,

we want to estimate the probability that the measurement of sensor i (with orientation θ_i with respect to the robot motion direction) at time t is a given z_t .

- First compute the distance $h_i(x_{r,t})$ between sensor i and the reflecting point **P** (this is possible given the robot pose since the environment is assumed known)

- Then:

$$p(z_t | x_{r,t}) = \text{prob}(\nu_t = z_t - h_i(x_{r,t}))$$



Measurement model: landmark measurements

Let $\mathbf{x}_{r,t} = [x, y, \theta]^T$ be the robot pose at time t

Assuming known the identities (and the position) of the landmarks, for each landmark $\mathbf{L}_i = (x_i, y_i)$ there is a measurement

$$z_{i,t} = \left(r_{i,t} + \nu_{i,t}^r, \phi_{i,t} + \nu_{i,t}^\phi \right)$$

where

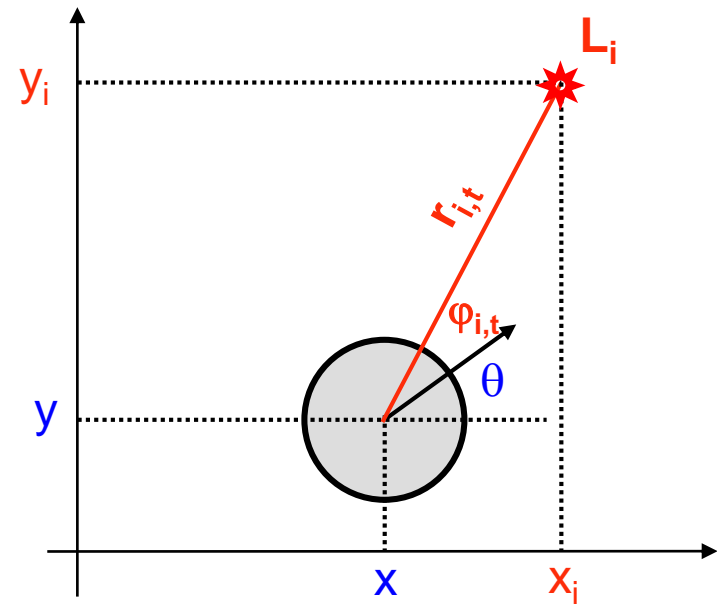
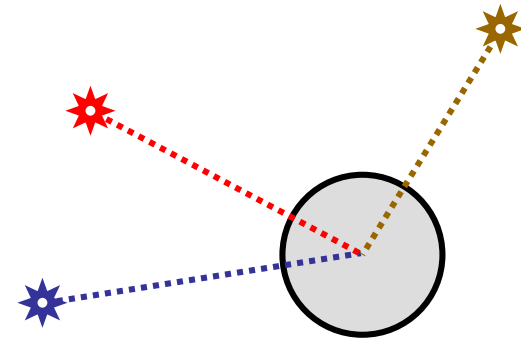
$$r_{i,t} = \sqrt{(x_i - x)^2 + (y_i - y)^2}$$

$$\phi_{i,t} = \text{atan2}((y_i - y), (x_i - x)) - \theta$$

and

$$\nu_{i,t}^r \sim \mathcal{N}(0, \zeta_{i,r}^2)$$

$$\nu_{i,t}^\phi \sim \mathcal{N}(0, \zeta_{i,\phi}^2)$$



Robot Localization Algorithms

The application of the Bayes filter is referred to as [Markov Localization](#):

For all $x_r \in \Omega$ do the following:

$$\overline{bel}_t(x_r) = \int_{\Omega} \overset{\text{Motion model}}{p(x_r | x_{r,t-1} = y, u_{t-1})} bel_{t-1}(y) dy$$

$$bel_t(x_r) = \eta \overset{\text{Measurement model}}{p(z_t | x_{r,t} = x_r)} \overline{bel}_t(x_r)$$

$$bel_0(x_r) = \frac{1}{|\Omega|} \quad \dots \text{ in the case of } \text{Global Localization}$$

Example of Markov Localization

(here $x_r = x$)

$$\overline{bel}_0(x) = \frac{1}{L}$$

z_0 : door detected

$$p(z_0|x)$$

$$bel_0(x) = \eta p(z_0|x_0) \overline{bel}_0(x) = \eta' p(z_0|x_0)$$

u_0 : the robot moves

$$\overline{bel}_1(x) = \int_0^L p(x|x_0 = y, u_0) bel_0(y) dy$$

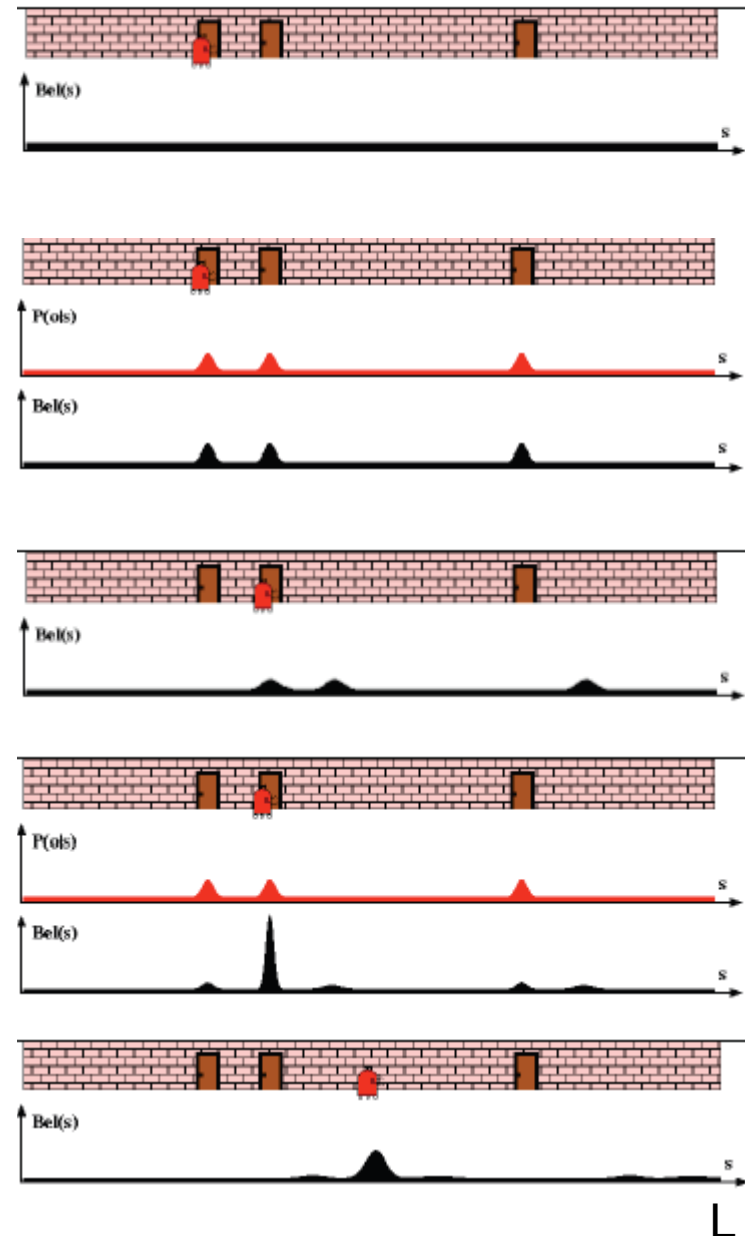
z_1 : door detected again

$$p(z_1|x)$$

$$bel_1(x) = \eta p(z_1|x_1) \overline{bel}_1(x)$$

u_1 : the robot moves again

$$\overline{bel}_2(x) = \int_0^L p(x|x_1 = y, u_1) bel_1(y) dy$$



The Bayes filter is not a practical algorithm

→ Gaussian Filters:

Extended Kalman Filter (EKF),
Unscented Kalman Filter (UKF),
Extended Information Filter (EIF)

→ Non Parametric Filters:

Histogram Filter (HF),
Particle Filter (PF)

Gaussian filters (EKF, UKF)

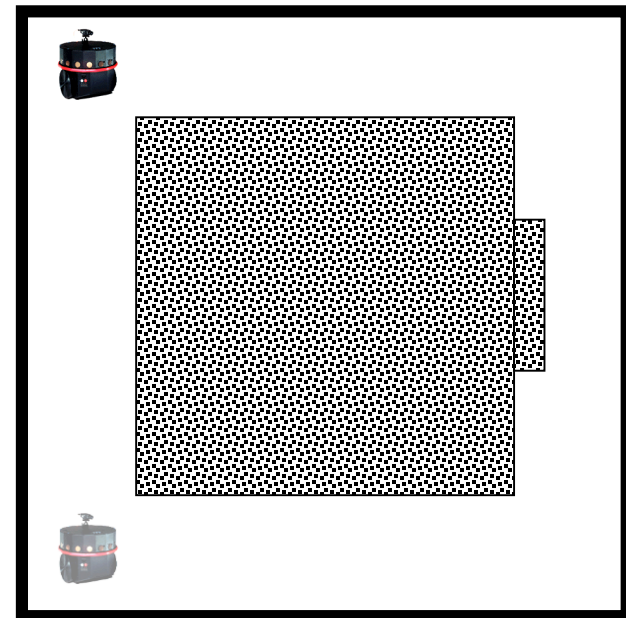
Tractable implementations of the Bayes Filter obtained by means of a [Gaussian approximation of the pdf](#)

- Unimodal approximations completely defined through mean m and covariance P
- OK for Position Tracking or in the global case if measurements allow to restrict the estimation to a small region (e.g. RFID)

The robot detects a corner and sees only free corridors → Bimodal pdf.

A bimodal pdf is not well approximated through a Gaussian density

- Multi-Hypothesis Gaussian (also in the case of unknown data association)



Extended Kalman Filter (EKF)

In the **linear-gaussian case**, all steps of the Bayes filter result in gaussian pdf
→ the Kalman Filter (KF) is optimal for these systems.

EKF: extension of KF to **non linear systems** (no more optimal)

Linear-gaussian system:

$$\begin{cases} x_{t+1} = Ax_t + Bu_t + \boxed{w_t} \\ z_t = Cx_t + \boxed{v_t} \end{cases}$$

Process noise
 $w_t \sim p_w(\cdot) = \mathcal{N}(0, Q)$

Measurement noise
 $v_t \sim p_v(\cdot) = \mathcal{N}(0, R)$

Assume:

$$bel_0(x) \sim \mathcal{N}(m_0, P_0)$$

Prediction step (Bayes filter)

$$\begin{aligned}\overline{bel}_t(x) &= \int_{R^n} p(x_t = x | x_{t-1} = y, u_{t-1}) bel_{t-1}(y) dy \\ &= \int_{R^n} p_w(x - Ay - Bu_{t-1}) bel_{t-1}(y) dy\end{aligned}$$

... is the **convolution** of two Gaussian distributions → **still Gaussian!**

The **Kalman Filter** results in the update of mean **m** and covariance **P**:

$$\overline{bel}_t(x) = \mathcal{N}(m_t^-, P_t^-)$$

where:

$$\begin{aligned}m_t^- &= A m_{t-1} + B u_{t-1} \\ P_t^- &= A P_{t-1} A^T + Q\end{aligned}$$

Prediction Step KF

Correction step (Bayes Filter)

$$\begin{aligned} bel_t(x) &= \eta p(z_t | x_t = x) \overline{bel}_t(x) \\ &= \eta p_v(z_t - Cx_t) \overline{bel}_t(x) \end{aligned}$$

... is the **product** of two Gaussian distributions → **still Gaussian!**

The **Kalman Filter** results in the update of mean **m** and covariance **P**:

$$bel_t(x) = \mathcal{N}(m_t, P_t)$$

where:

Kalman
Gain

$$K_t = P_t^- C^T (C P_t^- C^T + R)^{-1}$$

$$m_t = m_t^- + K_t (z_t - C m_t^-)$$

$$P_t = (I - K_t C) P_t^-$$

innovation

**Correction
Step KF**

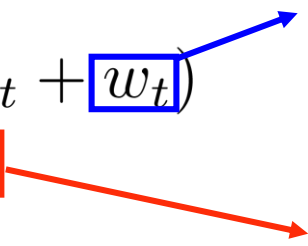
Extension to non linear systems (EKF)

Non linear - Gaussian system:

$$\begin{cases} x_{t+1} = f(x_t, u_t + \boxed{w_t}) \\ z = h(x_t) + \boxed{v_t} \end{cases}$$

Process noise
 $w_t \sim p_w(\cdot) = \mathcal{N}(0, Q)$

Measurement noise
 $v_t \sim p_v(\cdot) = \mathcal{N}(0, R)$



Assume:

$$bel_0(x) \sim \mathcal{N}(m_0, P_0)$$

$$\begin{aligned} m_t^- &= A m_{t-1} + B u_{t-1} \\ P_t^- &= A P_{t-1} A^T + Q \end{aligned} \quad \begin{array}{l} \text{Prediction} \\ \text{Step KF} \end{array}$$



$$\begin{aligned} m_t^- &= f(m_{t-1}, u_{t-1}) \\ P_t^- &= F_{x,t} P_{t-1} F_{x,t}^T + F_{u,t} Q F_{u,t}^T \end{aligned} \quad \text{Prediction Step EKF}$$

$$\begin{aligned} K_t &= P_t^- C^T (C P_t^- C^T + R)^{-1} \\ m_t &= m_t^- + K_t (z_t - C m_t^-) \\ P_t &= (I - K_t C) P_t^- \end{aligned} \quad \begin{array}{l} \text{Correction} \\ \text{Step KF} \end{array}$$



$$\begin{aligned} K_t &= P_t^- H_{x,t}^T (H_{x,t} P_t^- H_{x,t}^T + R)^{-1} \\ m_t &= m_t^- + K_t (z_t - h(m_t^-)) \\ P_t &= (I - K_t H_{x,t}) P_t^- \end{aligned} \quad \text{Correction Step EKF}$$

Application to the localization problem:

$$x \rightarrow x_r = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \quad u \rightarrow \begin{bmatrix} u_R^e \\ u_L^e \end{bmatrix} \quad w \rightarrow \begin{bmatrix} n_R \\ n_L \end{bmatrix}$$

$$f(x, u + n) \rightarrow f(x_r, [u_R^e, u_L^e]^T + [n_R, n_L]^T) := \begin{cases} x + \frac{u_R^e + n_R + u_L^e + n_L}{2} \cos(\theta) \\ y + \frac{u_R^e + n_R + u_L^e + n_L}{2} \sin(\theta) \\ \theta + \frac{u_R^e + n_R - u_L^e - n_L}{d} \end{cases}$$

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} = \begin{bmatrix} h_1(x) + \nu_1 \\ h_2(x) + \nu_2 \\ \vdots \\ h_m(x) + \nu_m \end{bmatrix} = h(x) + \nu$$

Unscented Kalman Filter (UKF)

Specific for non linear systems: [avoids jacobians](#) and derives the pdf of a random variable transformed through a non linear function [through the transformation of a set of Sigma Points](#). Same numerical complexity of EKF. Generally more performant, but still not optimal!

Let $x \sim \mathcal{N}(m_x, \sigma_x^2)$

a [scalar gaussian random variable](#). Let

$$y = g(x)$$

be a [non linear function](#). Clearly [y is not gaussian](#). If using Gaussian filters, [y is approximated through a Gaussian random variable](#), i.e.:

$$y \sim \mathcal{N}(m_y, \sigma_y^2)$$

The EKF:

$$m_y = g(m_x)$$
$$\sigma_y^2 = \left(\left. \frac{dg}{dx} \right|_{m_x} \right) \sigma_x^2 \left(\left. \frac{dg}{dx} \right|_{m_x} \right)^T$$

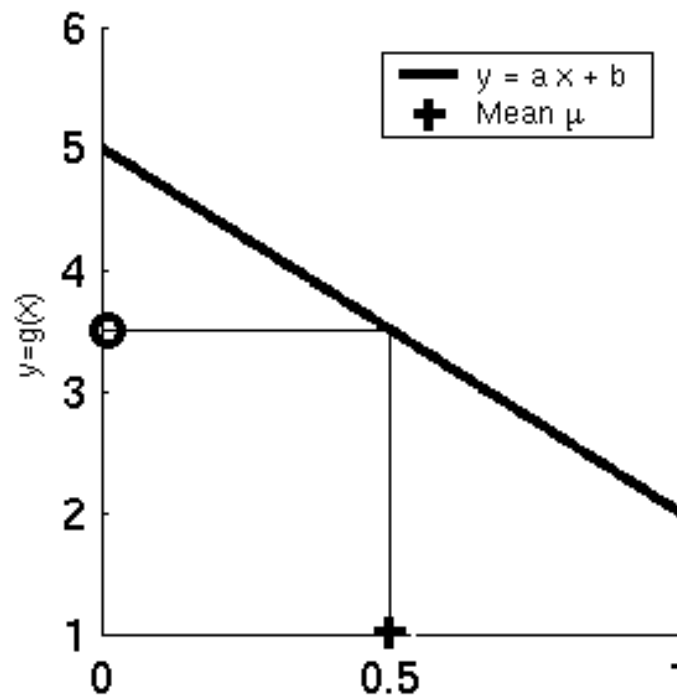
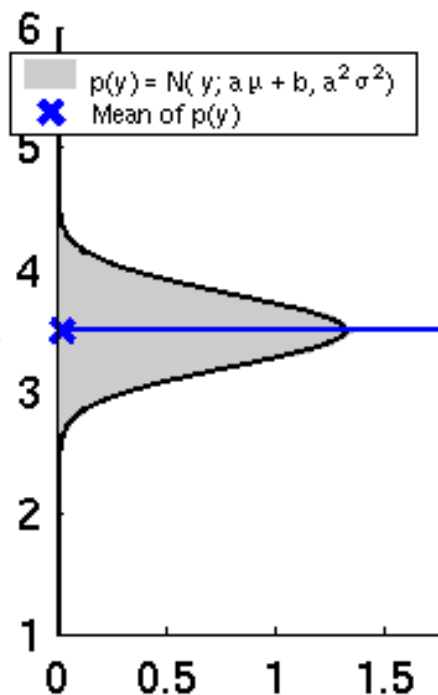
The UKF:

1. Generate a set of Sigma Points ξ_i and corresponding weights w_i .
2. Compute the transformation of each Sigma Point:

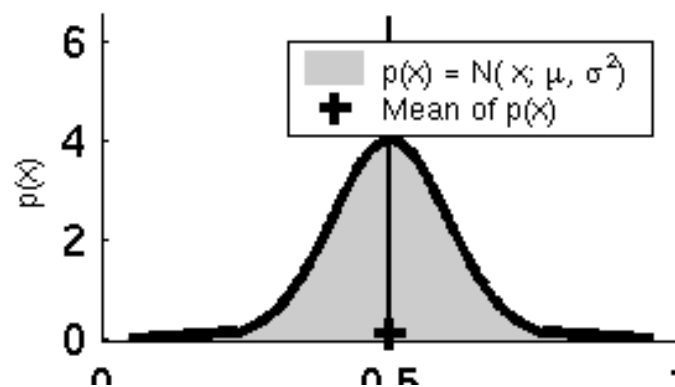
$$\psi_i = g(\xi_i)$$

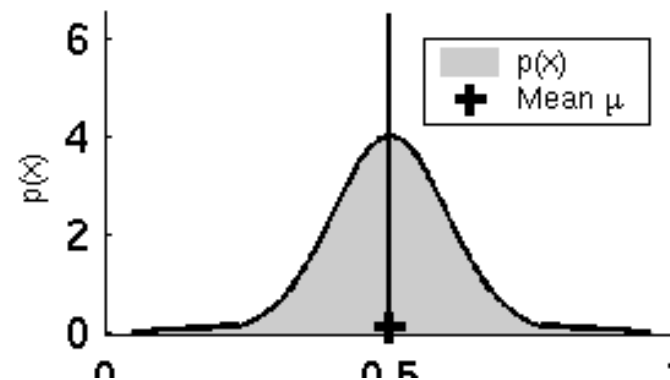
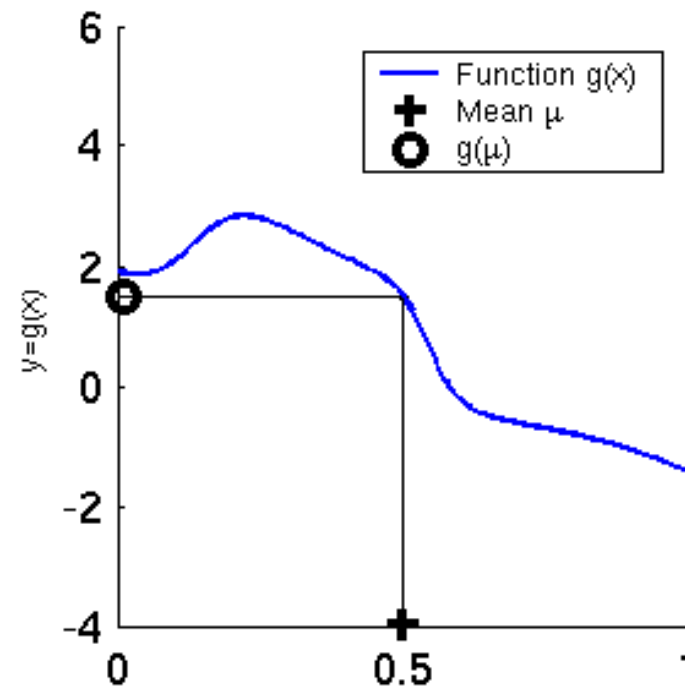
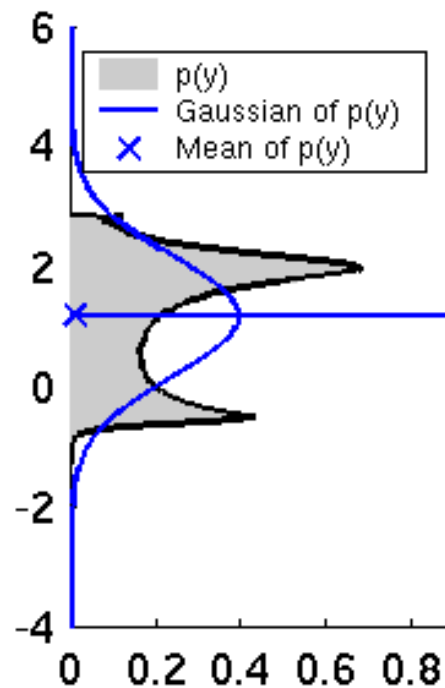
3. Determine mean and covariance of the transformed variable y by:

$$m_y = \sum_i w_i \cdot \psi_i$$
$$\sigma_y^2 = \sum_i w_i \cdot (\psi_i - m_y)(\psi_i - m_y)^T$$

Linear transformation of a Gaussian variable

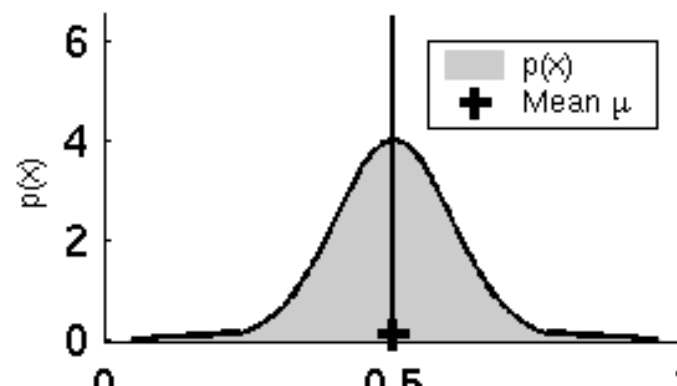
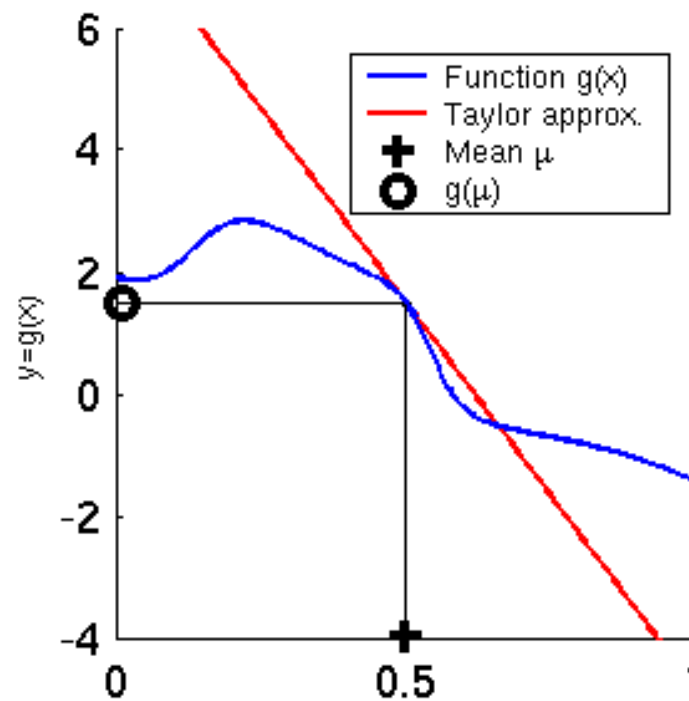
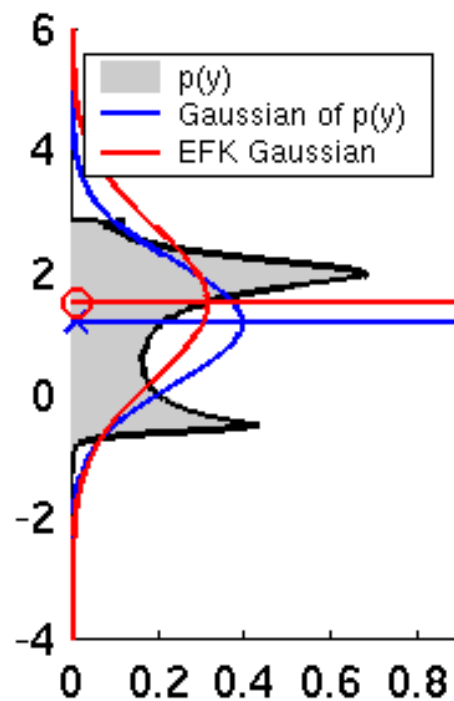
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006



Non Linear transformation of a Gaussian variableGaussian

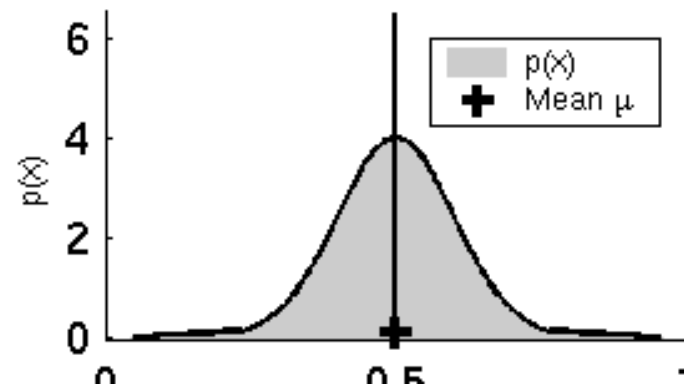
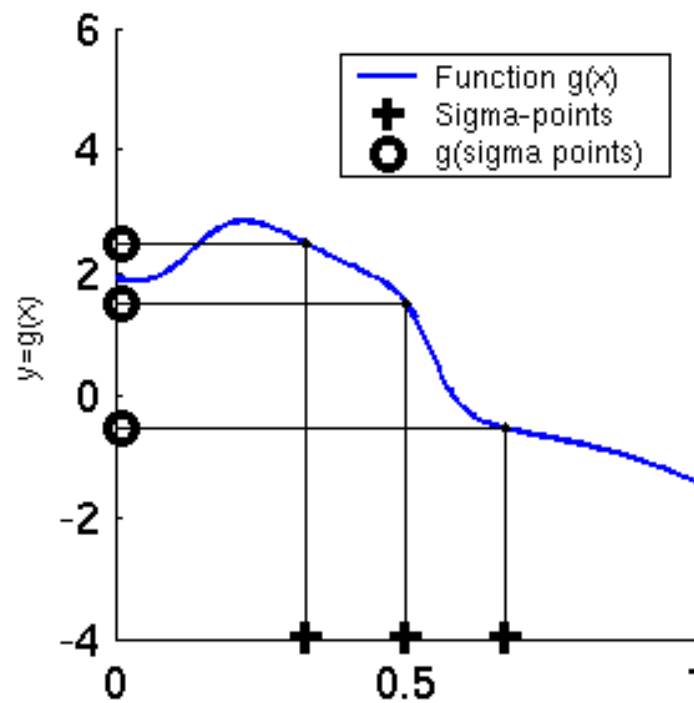
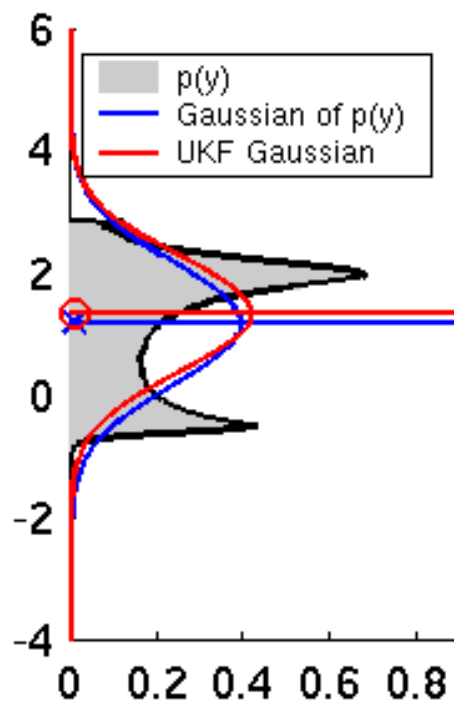
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

EKF approximation



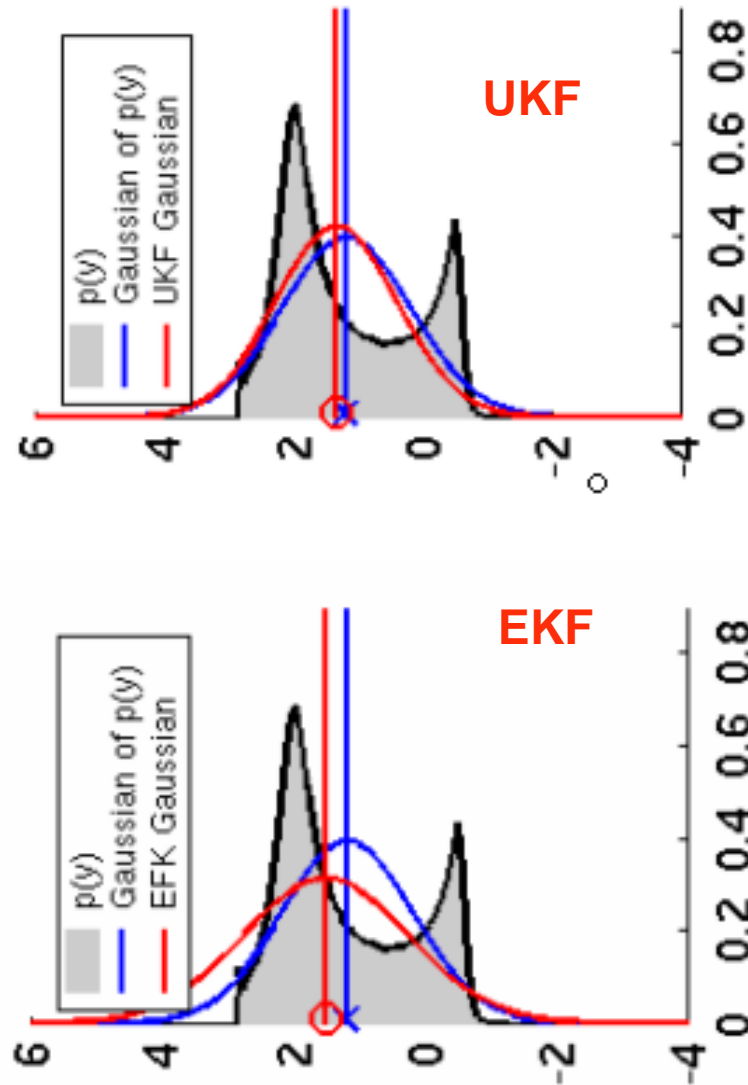
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

UKF approximation



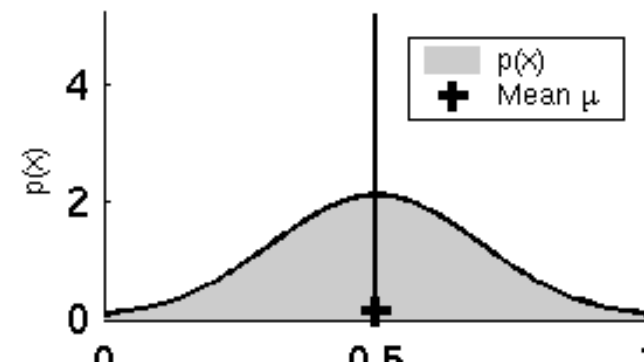
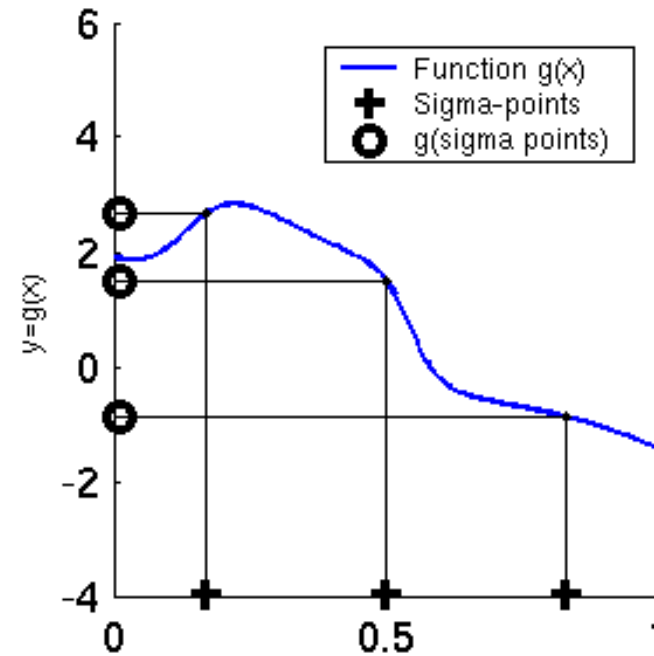
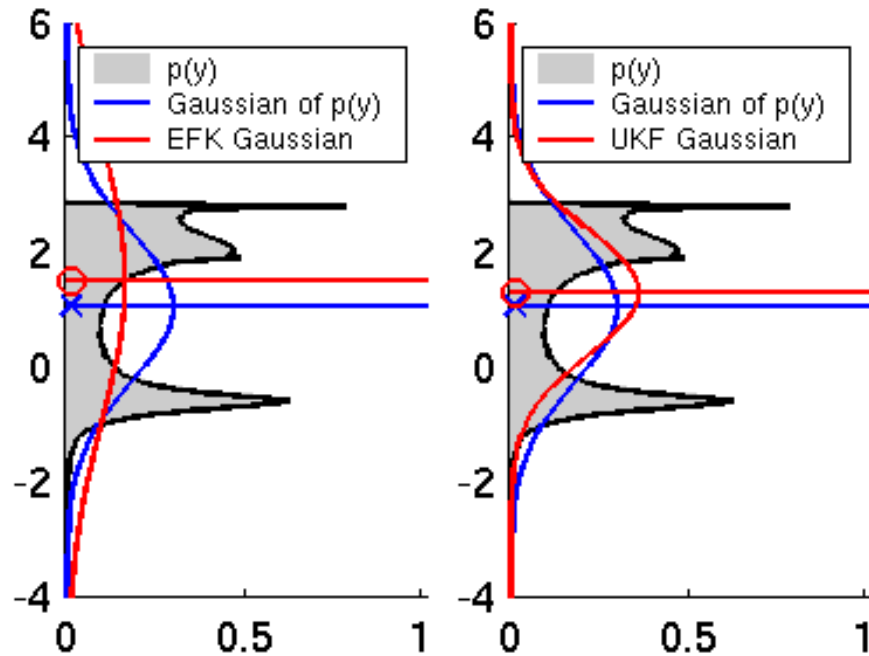
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

EKF vs UKF approximation



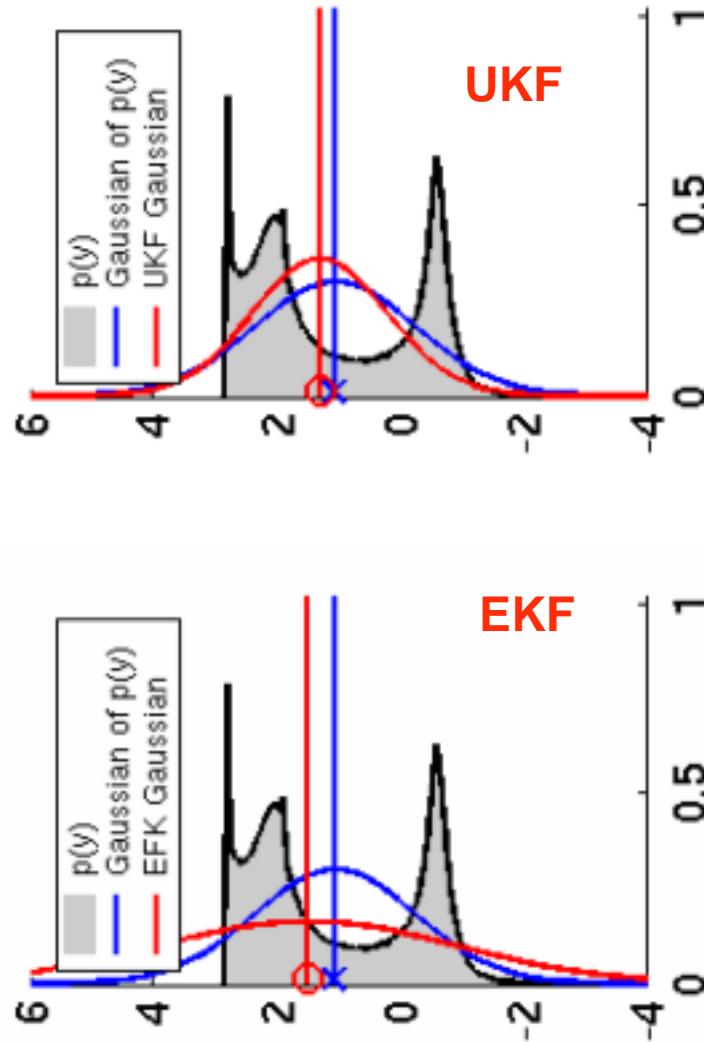
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

EKF vs UKF approximation (2)



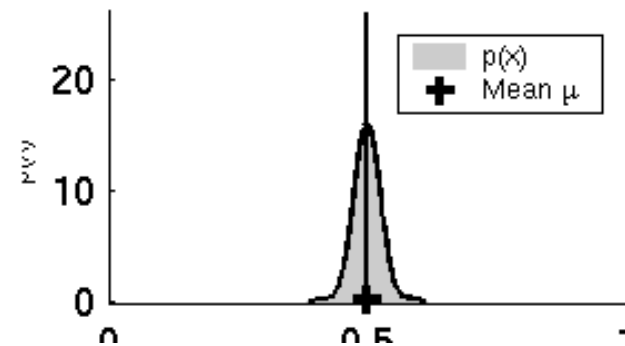
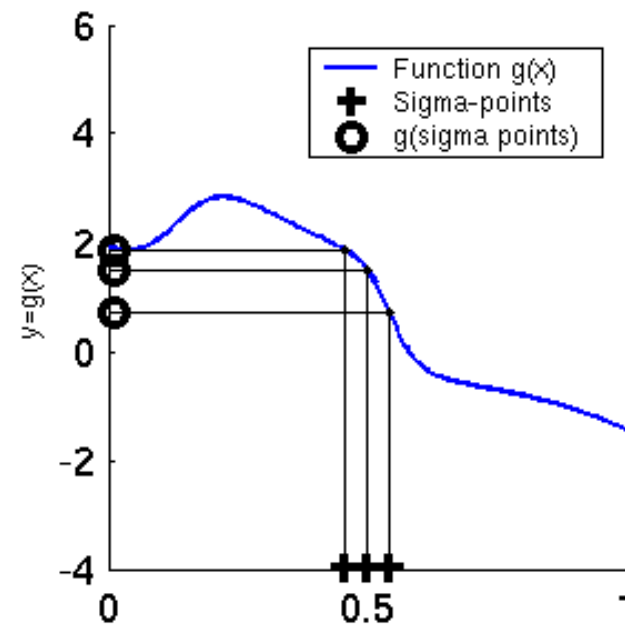
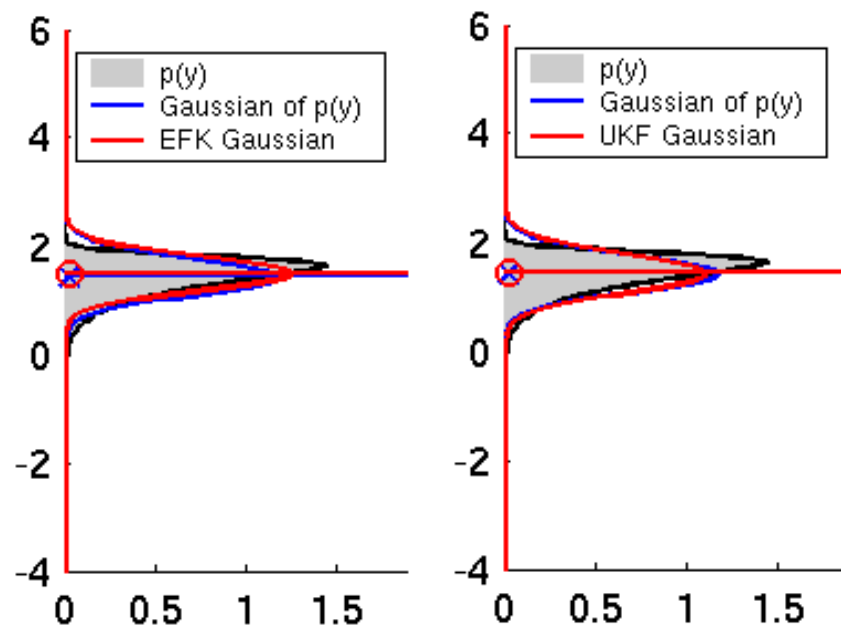
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

EKF vs UKF approximation (2)



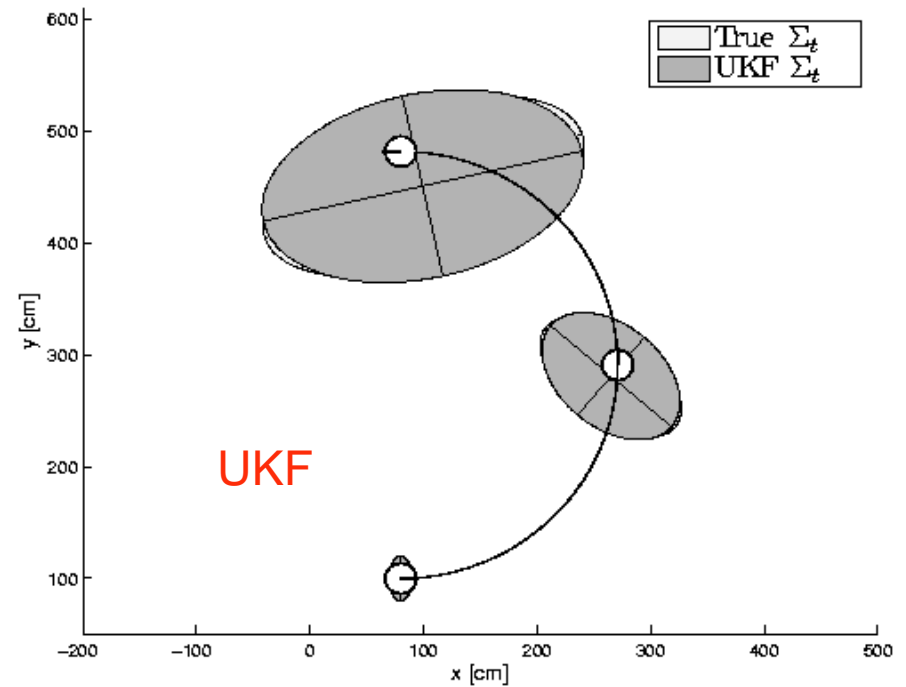
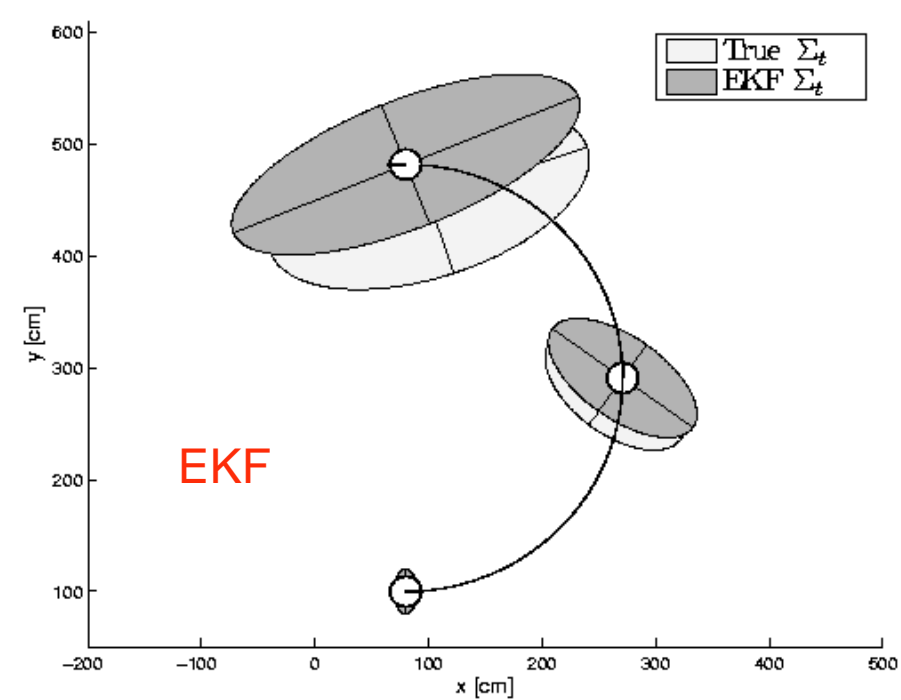
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

EKF vs UKF approximation (3)



From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

Prediction quality of EKF vs UKF



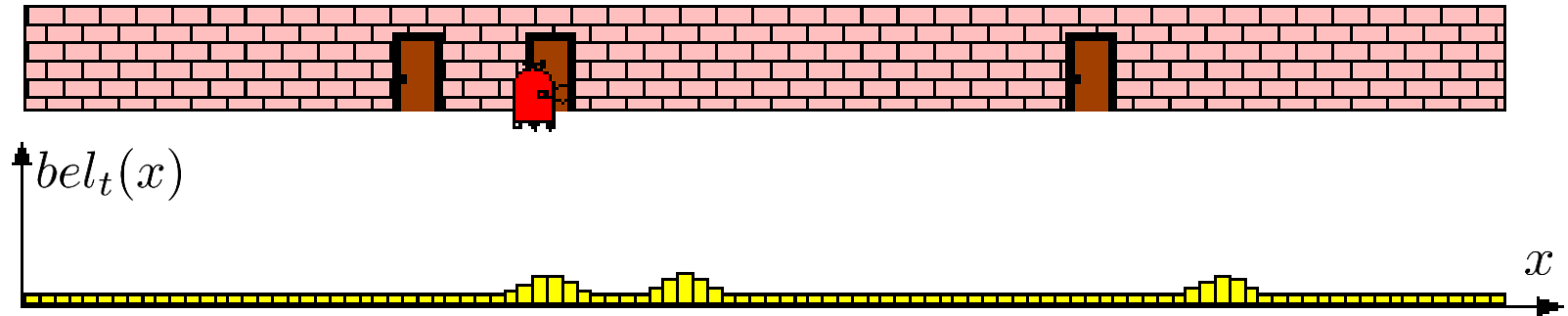
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

Non parametric filters (HF, PF)

- [Numerical implementation](#) of the Bayes Filter
- Suitable also for [global localization](#) problems
- Precision dependent on the available [computational time](#)

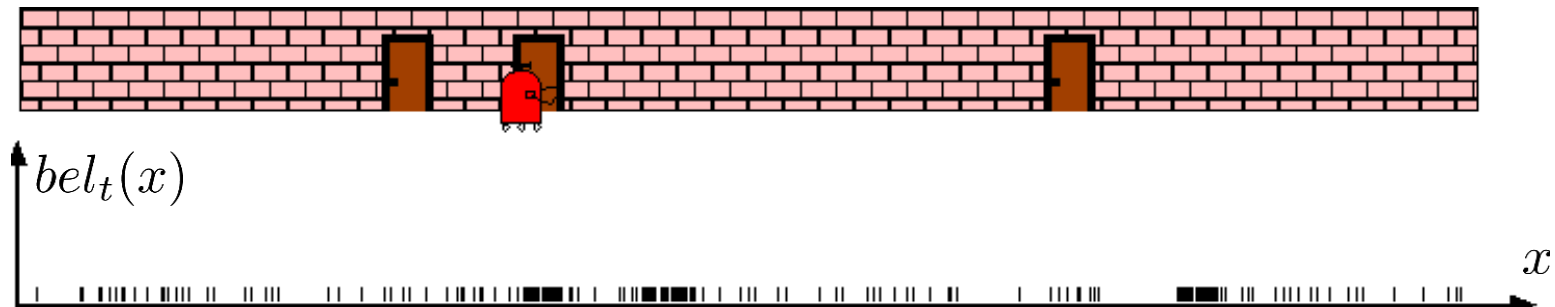
Histogram filter: grid representation of the pdf

→ The environment is partitioned into a set of cells and the pdf is numerically represented by the probability of each cell



Particle filter: sample (particles) representation of the pdf

→ The pdf is represented by a set of particles: the larger the pdf in a given region, more particles in that region

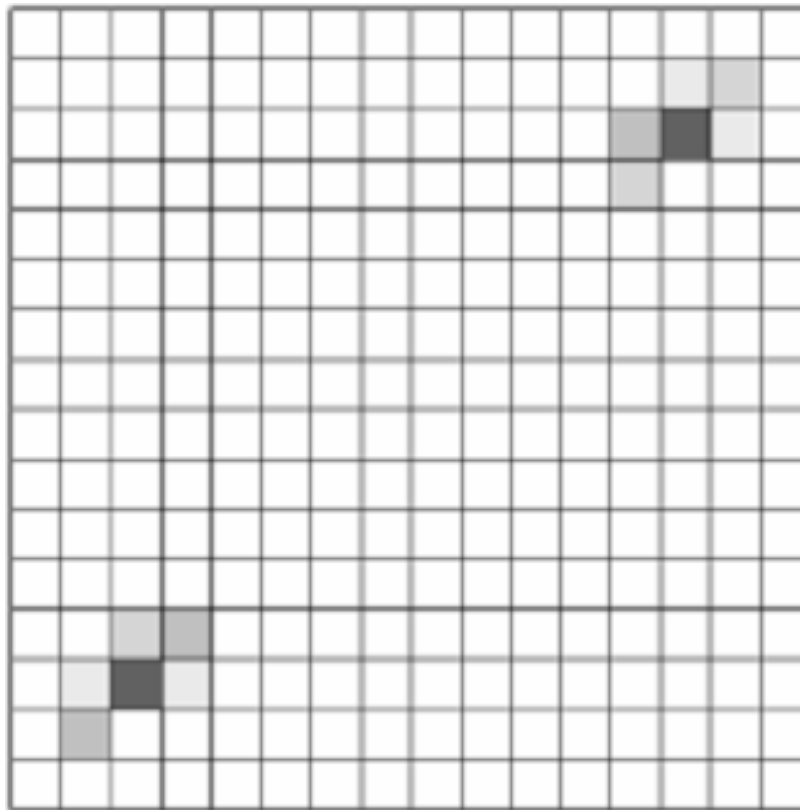


Histogram filter

Grid representation of the belief:

$$\overline{bel}_t(x_r) := \{\bar{p}_t(c_k)\}_{c_k \in \Omega_D}$$

$$bel_t(x_r) := \{p_t(c_k)\}_{c_k \in \Omega_D}$$



From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

Prediction step (Bayes filter)

For all $x_r \in \Omega$ do the following:

$$\overline{bel}_t(x_r) = \int_{\Omega} p(x_r | x_{r,t-1} = y, u_{t-1}) bel_{t-1}(y) dy$$

For all discretized $c_k \in \Omega_D$ do:

Prediction Step HF

$$\begin{aligned} \bar{p}_t(c_k) &= \\ &= \sum_{c_i \in \Omega_D} p(x_{r,t} \in c_k | u_{t-1}, x_{r,t-1} \in c_i) p_{t-1}(c_i) \end{aligned}$$

Correction step (Bayes filter)

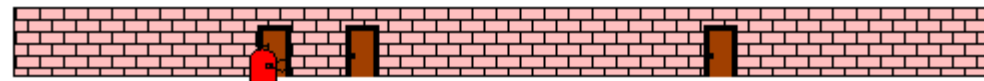
For all $x_r \in \Omega$ do the following:

$$bel_t(x_r) = \eta p(z_t | x_{r,t} = x_r) \overline{bel}_t(x_r)$$

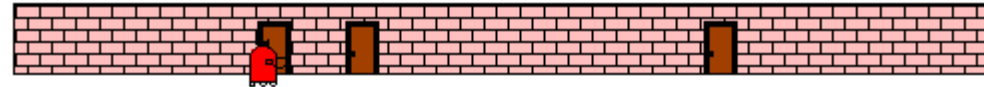
For all discretized $c_k \in \Omega_D$ do:

Correction Step HF

$$p_t(c_k) = \eta p(z_t | x_{r,t} \in c_k) \bar{p}_t(c_k)$$



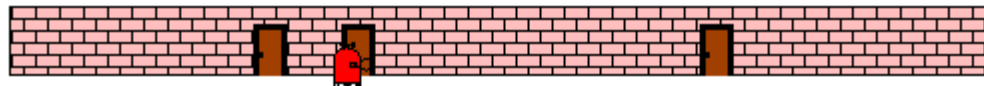
$$\bar{p}_0(c_k) = \frac{1}{N}$$



$$p(z_0|x)$$

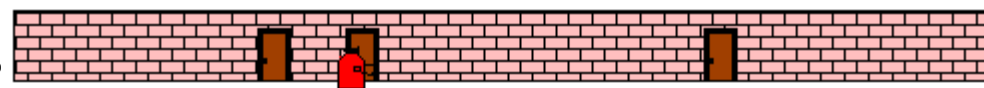


$$p_0(c_k) = \eta p(z_0|x_0 \in c_k) \bar{p}_0(c_k)$$



$$\bar{p}_1(c_k) =$$

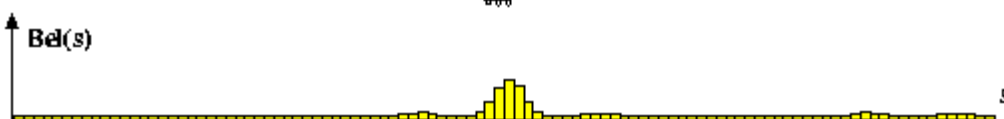
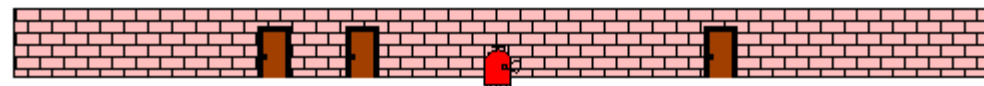
$$\sum_{c_i \in \Omega_D} p(x_1 \in c_k | u_0, x_0 \in c_i) p_0(c_i)$$



$$p(z_1|x)$$



$$p_1(c_k) = \eta p(z_1|x_1 \in c_k) \bar{p}_1(c_k)$$

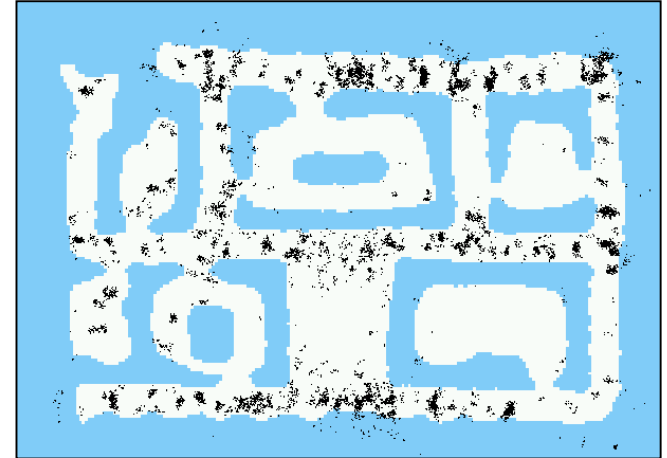
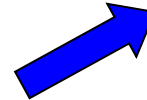
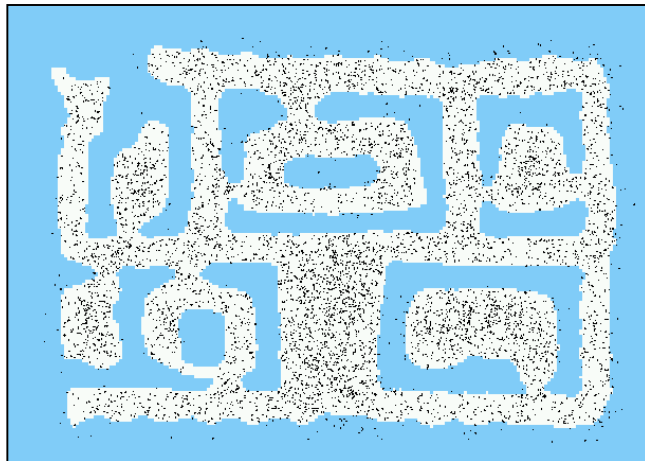


$$\bar{p}_2(c_k)$$

Particle filter

Particle representation of the belief:

$$bel_t(x_r) = \{x_{r,t}^m, w_t^m\}_{m=1,\dots,M}$$



Prediction step (Bayes filter)

For all $x_r \in \Omega$ do the following:

$$\overline{bel}_t(x_r) = \int_{\Omega} p(x_r | x_{r,t-1} = y, u_{t-1}) bel_{t-1}(y) dy$$

Prediction Step PF

For all $m = 1, 2, \dots, M$ sample the new position of particle m according to the motion model:

$$x_{r,t}^m = f(x_{r,t-1}^m, u_{t-1} + \boxed{n^m})$$

 Motion noise for particle m

(weights w_t^m unchanged)

Motion model

$$f(x_r, u + n) := \begin{cases} x + \frac{u_R^e + n_R + u_L^e + n_L}{2} \cos(\theta) \\ y + \frac{u_R^e + n_R + u_L^e + n_L}{2} \sin(\theta) \\ \theta + \frac{u_R^e + n_R - u_L^e - n_L}{d} \end{cases}$$

Correction step (Bayes filter)

For all $x_r \in \Omega$ do the following:

$$bel_t(x_r) = \eta p(z_t | x_{r,t} = x_r) \overline{bel}_t(x_r)$$

Correction Step PF

For all $m = 1, 2, \dots, M$ update particle weights according to the last measurement (particle poses remain unchanged in this step):

$$w_t^m = p(z_t | x_{r,t}^m)$$

Resampling (performed periodically, possibly at all steps, to regenerate the set of particles):

For M times:

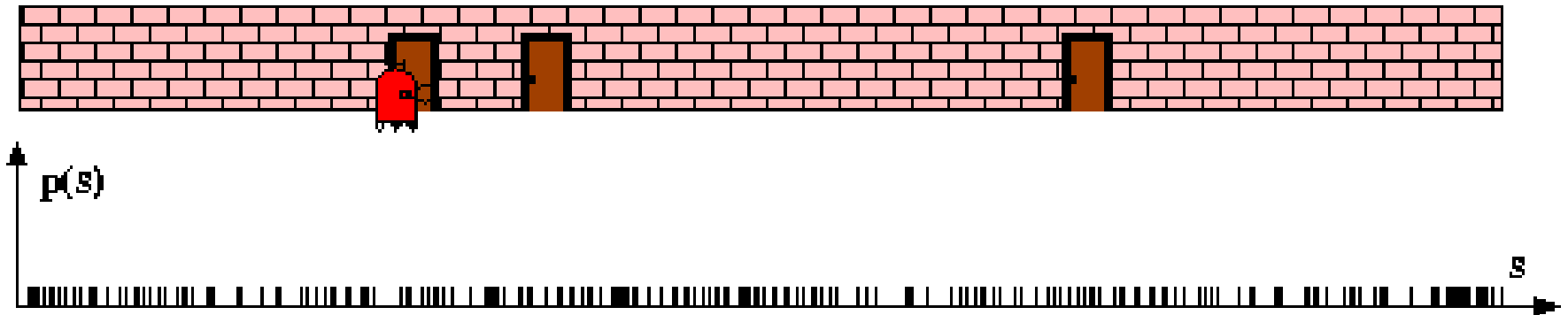
1. Draw particle i with probability $\frac{w_t^i}{\sum_m w_t^m}$
2. Add the particle to the set of particles of time t+1
3. The weights are re-initialized: $w_{t+1}^m = \frac{1}{M}$

Belief at time 0:

$$\overline{bel}_0(x_r) = \{x_{r,0}^m, w_0^m\}_{m=1,\dots,M}$$

Global localization problem:

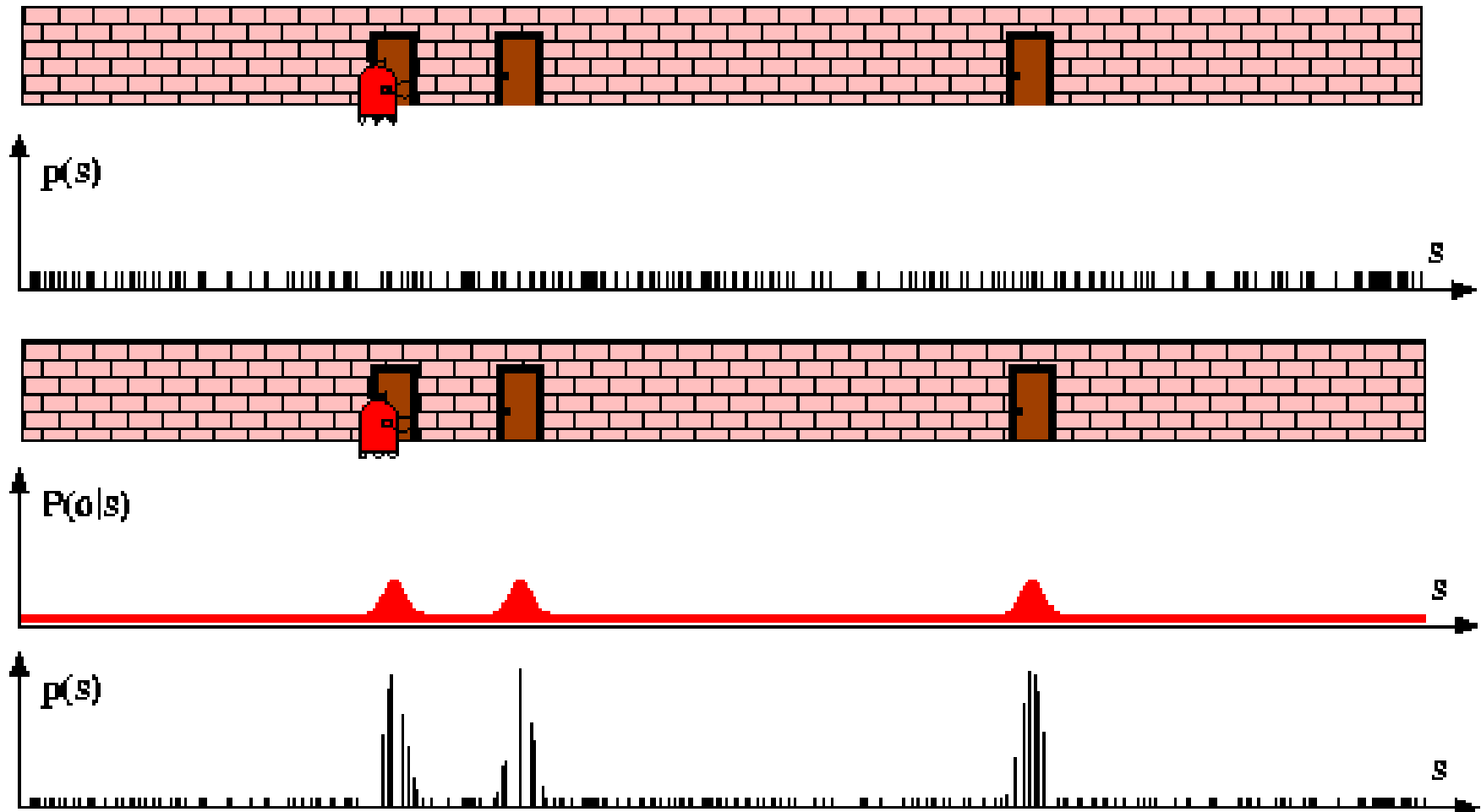
- Particles spread uniformly in the environment and
- $w_0^m = \frac{1}{M}$



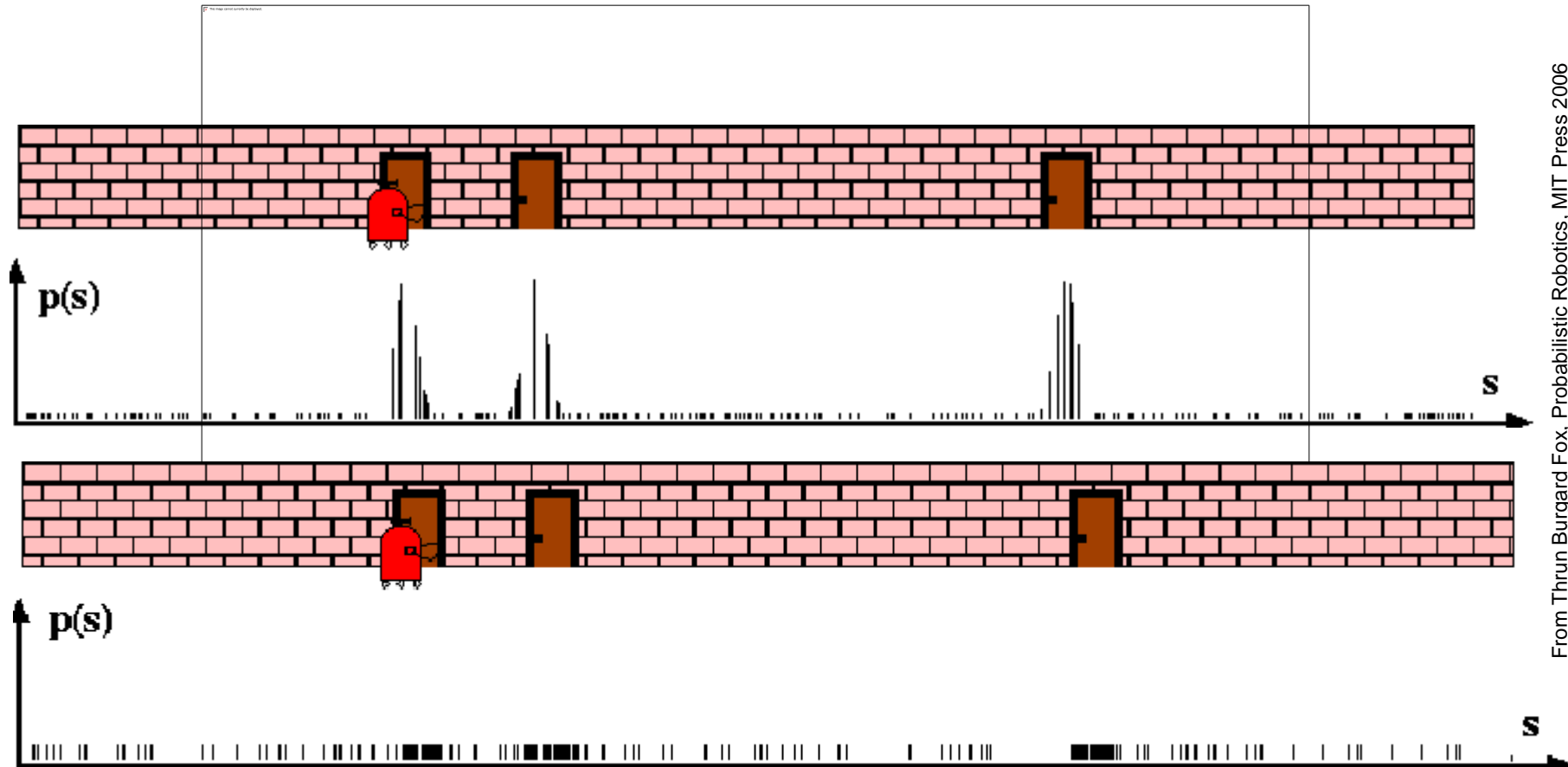
From Thrun Burgard Fox, Probabilistic Robotics, MIT Press 2006

Sensor Information: Importance Sampling

$$w_0^m = p(z_0 | x_{r,0}^m)$$

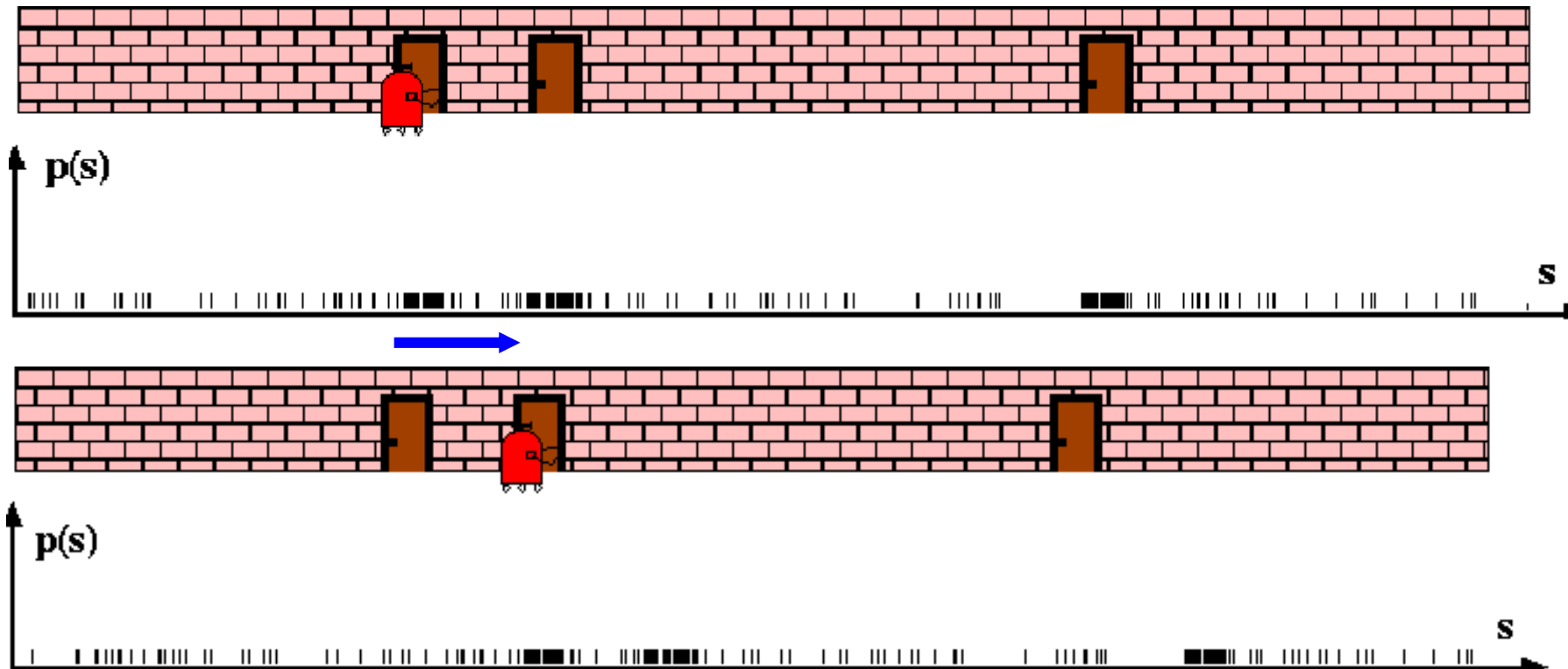


Resampling



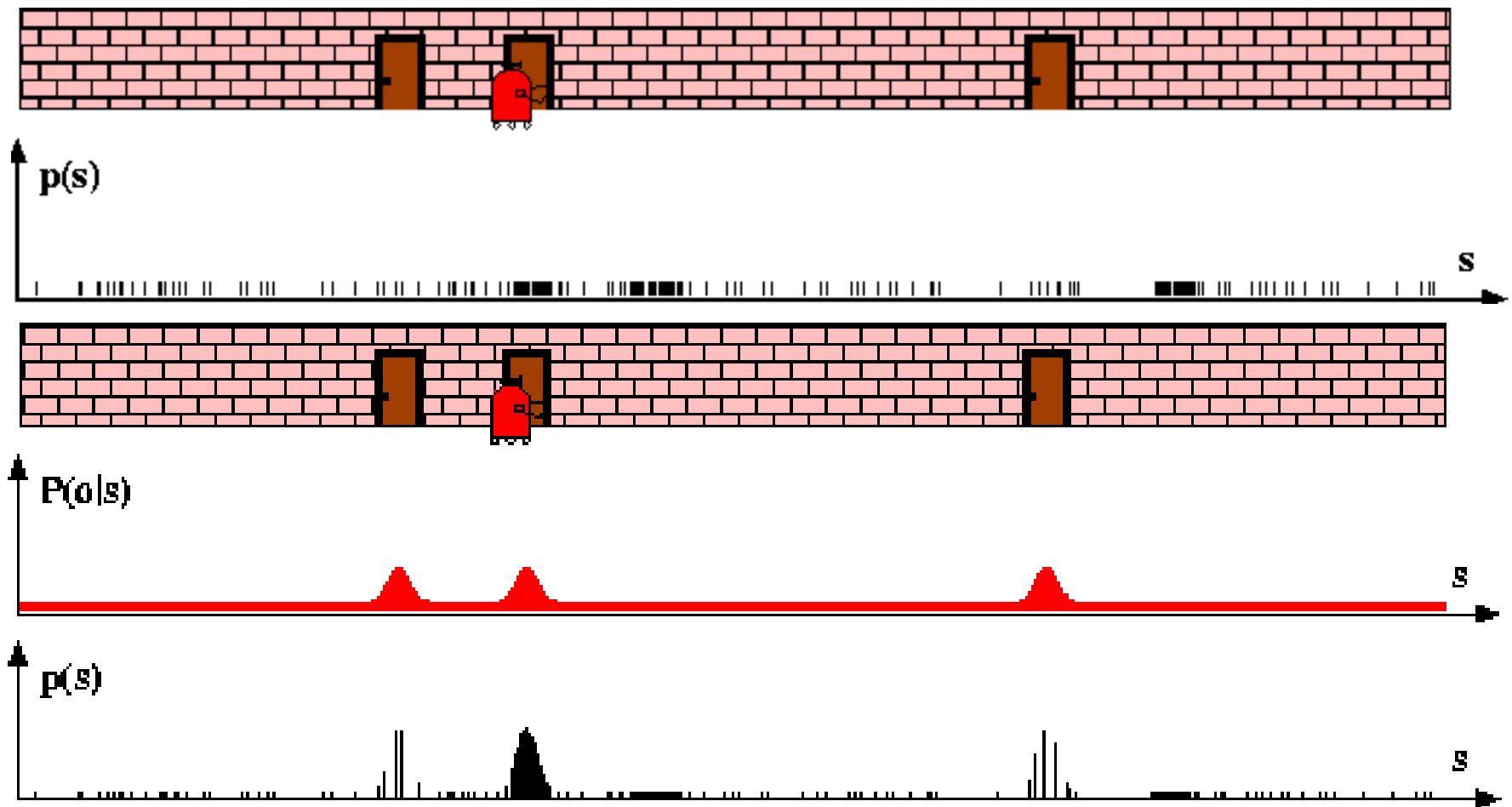
Robot motion

$$x_{r,1}^m = f(x_{r,0}^m, u_0 + n^m)$$



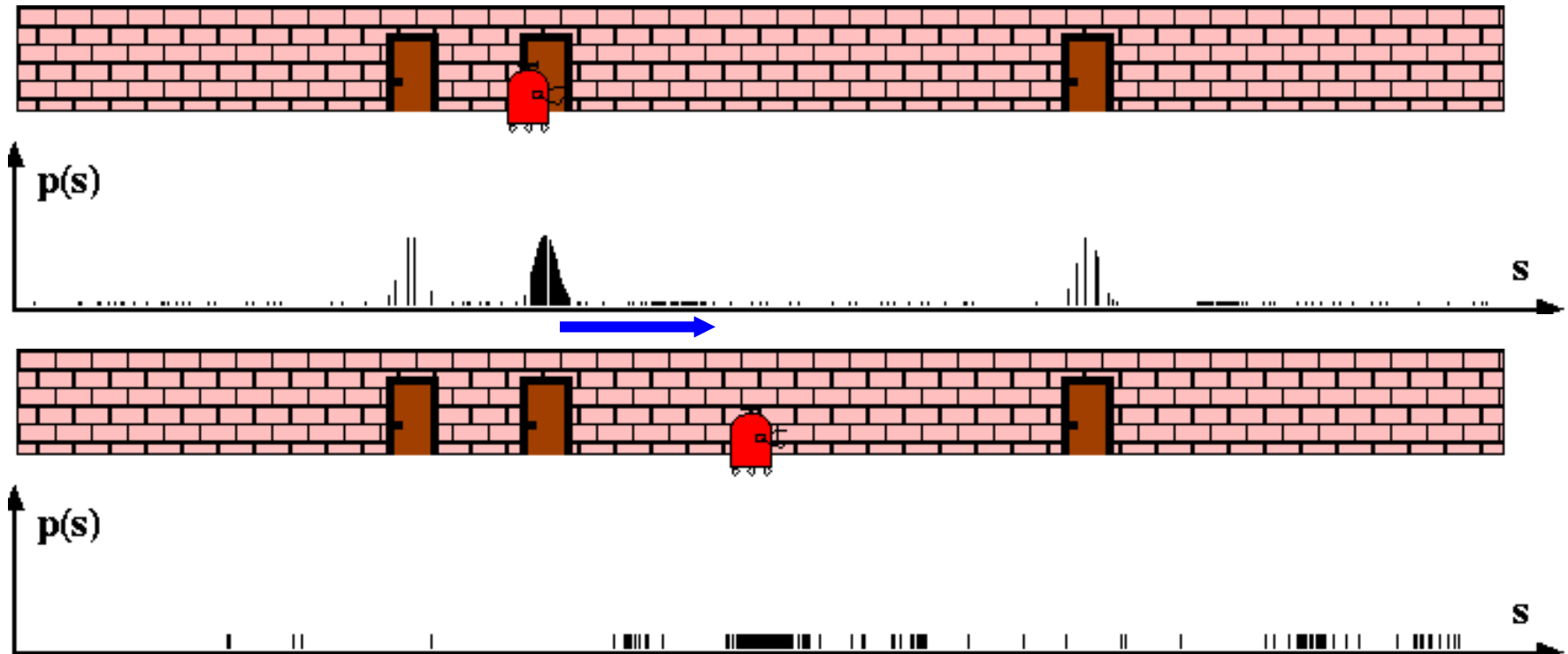
Sensor Information: Importance Sampling

$$w_1^m = p(z_1 | x_{r,1}^m)$$



Resampling + Robot Motion

$$x_{r,2}^m = f(x_{r,1}^m, u_1 + n^m)$$



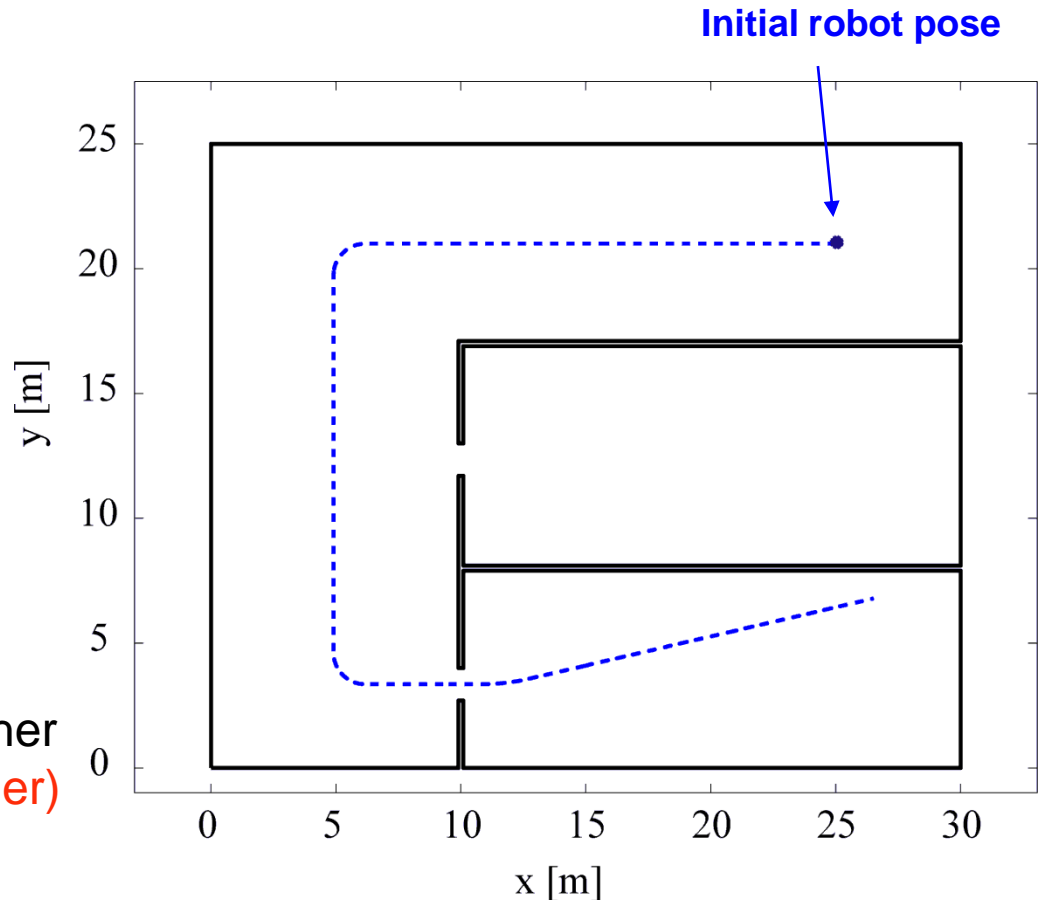
Examples

The presentation ends with [a simulative comparison](#) of the approaches described so far under different scenarios.

The environment considered is shown in the figure, together with the robot path (blue-dashed).

It will be reported a comparison between the estimation obtained through Gaussian approaches ([EKF](#), [UKF](#)) and the one obtained through non-parametric approaches ([PF](#)), in the case of:

- [global localization problems](#)
 - [or position tracking problems](#);
- considering a robot endowed with odometry and using, as exteroceptive measurements, either
- [sonar/laser readings \(range finder\)](#)
 - [range to a set of landmarks.](#)



Simulation data

Motion model (in the simulation we consider a synchro drive system):

$$\begin{aligned}\delta\rho_t &= \delta\rho_t^e + n_{\rho,t} & n_{\rho,t} &\sim \mathcal{N}(0, K_\rho|\delta\rho_t|) \\ \delta\theta_t &= \delta\theta_t^e + n_{\theta,t} & n_{\theta,t} &\sim \mathcal{N}(0, K_\theta|\delta\rho_t|)\end{aligned}$$

$$\begin{aligned}\text{with: } K_\rho &= 0.01\text{m} \\ K_\theta &= 0.02\text{rad}^2/\text{m}\end{aligned}$$

Measurement model (range measurements):

$$z_{i,t} = h_i(x_{r,t}) + \nu_{i,t} \quad \nu_{i,t} \sim \mathcal{N}(0, \zeta_{i,2})$$

$$\text{with: } \zeta_{i,2} = 0.1\text{m}^2$$

Path:

$$T = 5000 \text{ steps}$$

On straight lines:

$$\delta\rho_t = 0.012 \text{ m}$$

$$\delta\theta_t = 0 \text{ rad}$$

On the curves:

$$\delta\rho_t = 0.01 \text{ m}, \delta\theta_t = 7.85 \cdot 10^{-3} \text{ rad}$$

We assume that the **exteroceptive measurements** (laser or landmark readings) are available **every N_s steps**, with $N_s \geq 1$, while odometry is available at all steps

Standard deviation for the estimate of the initial pose in the **position tracking** case: **0.01 m for x and y and 0.02 rad for θ**

To evaluate the performance we consider an **average position estimation error**:

$$J = \frac{1}{T} \sum_{t=1}^T \sqrt{[x(t) - \hat{x}(t)]^2 + [y(t) - \hat{y}(t)]^2}$$

$[x(t), y(t)]$ **real** position of the robot at time t

$[\hat{x}(t), \hat{y}(t)]$ **estimated** position of the robot at time t

Odometry estimation = integration of the encoder readings:

$$\begin{cases} \hat{x}_{t+1} = \hat{x}_t + \delta\rho_t^e \cos(\hat{\theta}_t) \\ \hat{y}_{t+1} = \hat{y}_t + \delta\rho_t^e \sin(\hat{\theta}_t) \\ \hat{\theta}_{t+1} = \hat{\theta}_t + \delta\theta_t^e \end{cases}$$

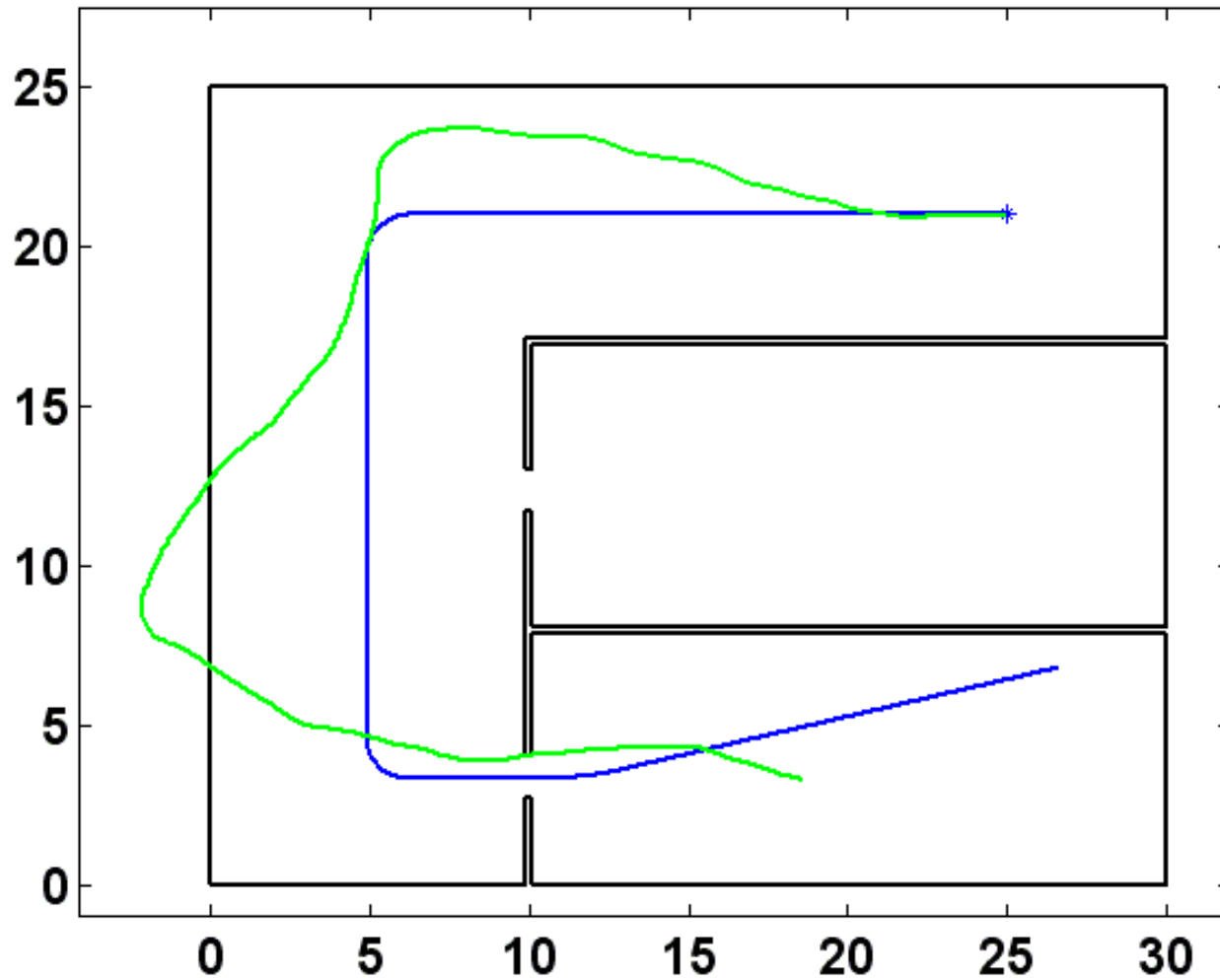
while the **real robot pose** is given by:

$$\begin{cases} x_{t+1} = x_t + \delta\rho_t \cos(\theta_t) \\ y_{t+1} = y_t + \delta\rho_t \sin(\theta_t) \\ \theta_{t+1} = \theta_t + \delta\theta_t \end{cases}$$

$$\delta\rho_t = \delta\rho_t^e + n_{\rho,t}$$

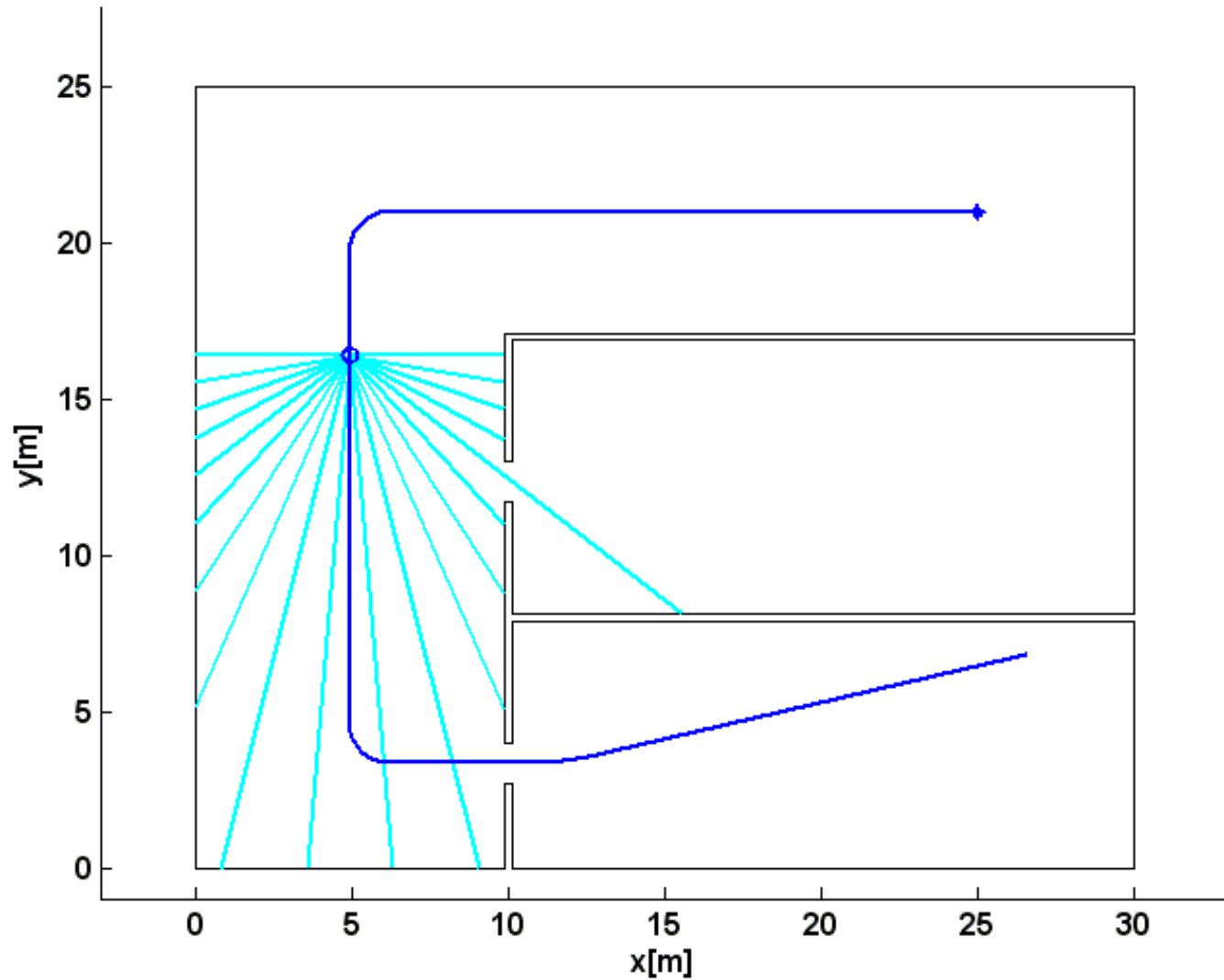
$$\delta\theta_t = \delta\theta_t^e + n_{\theta,t}$$

Odometry estimation:

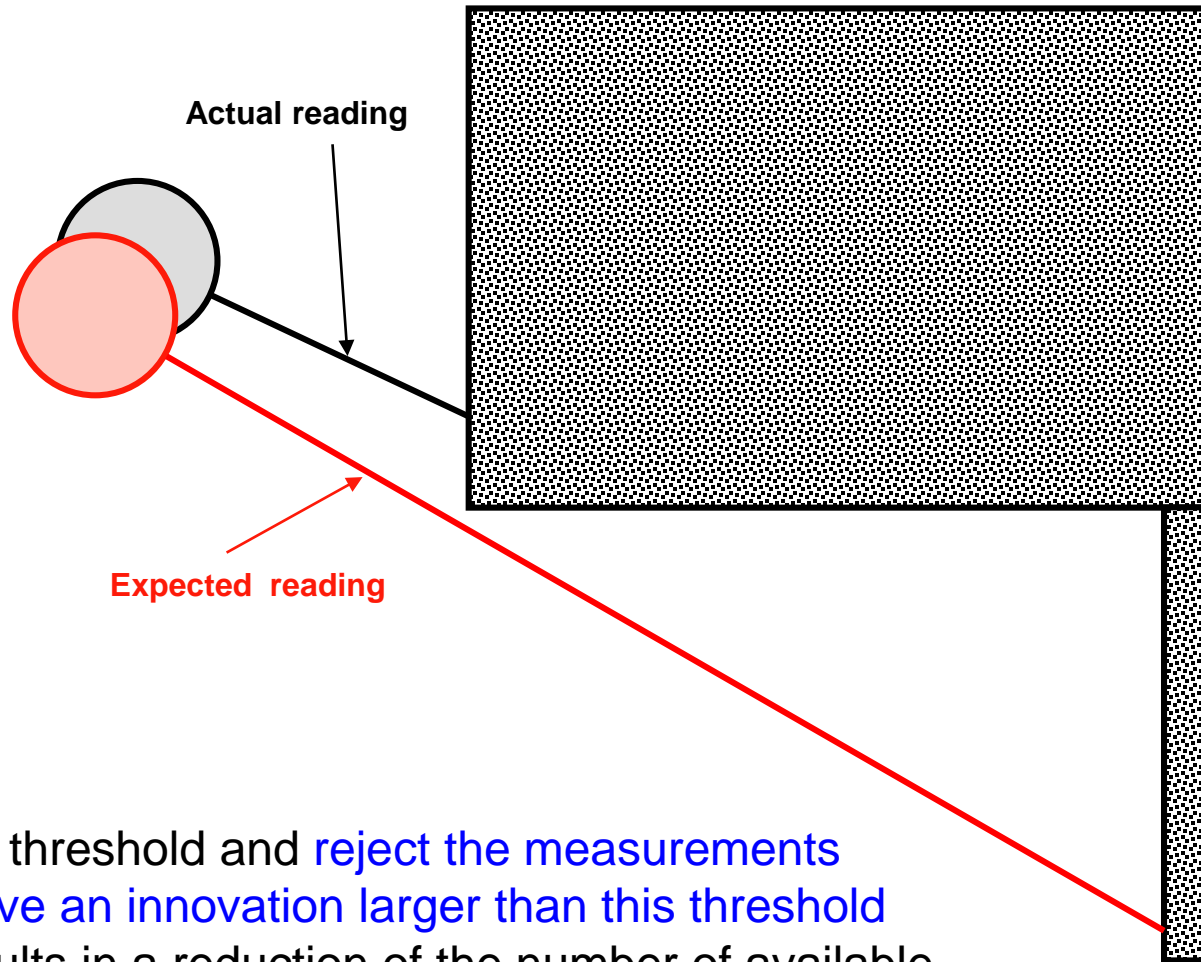


$J = 4.9 \text{ m}$

Range finder case: 20 beams equally spaced in the robot motion direction

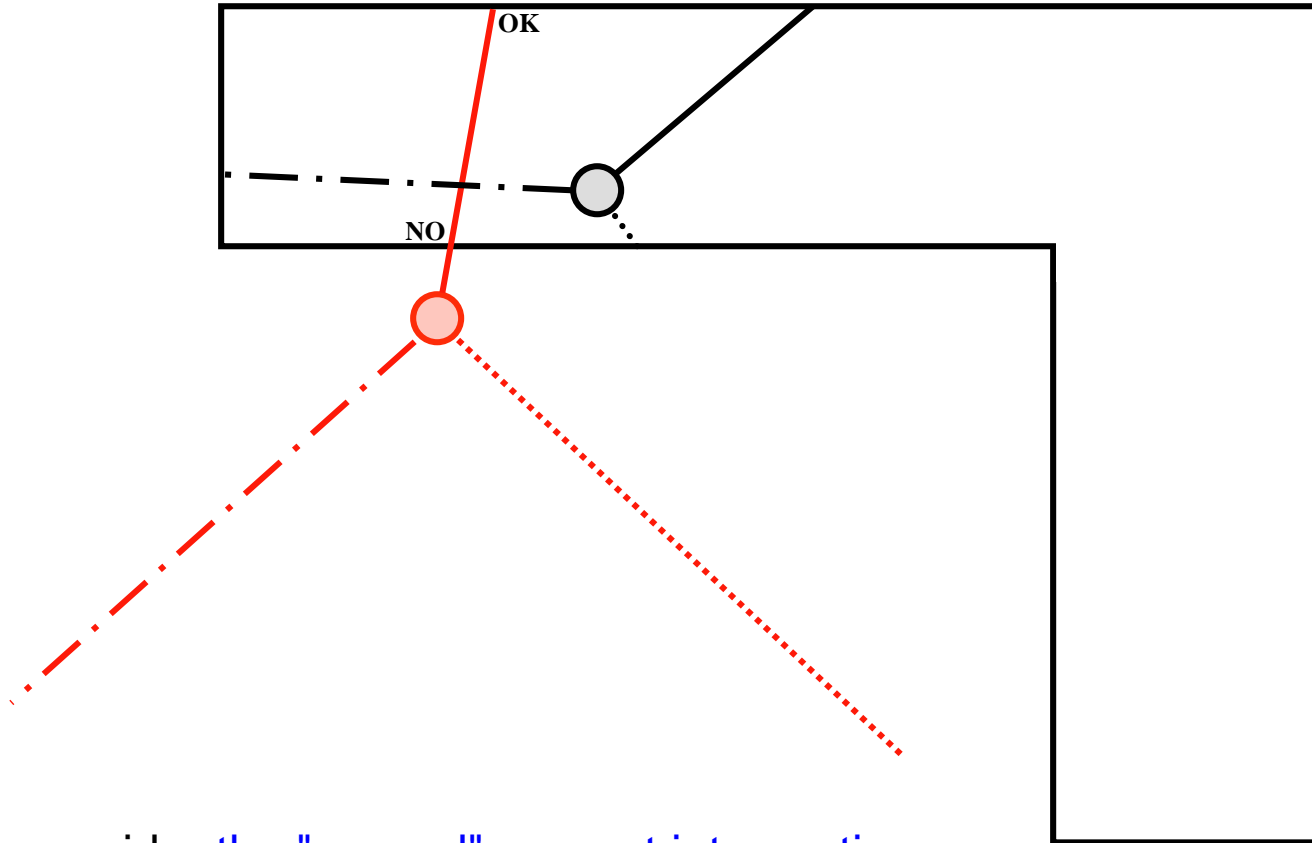


What to do if a measurement is **very different** from the expected one?
It is possible that this does not depend on a bad pose estimation....



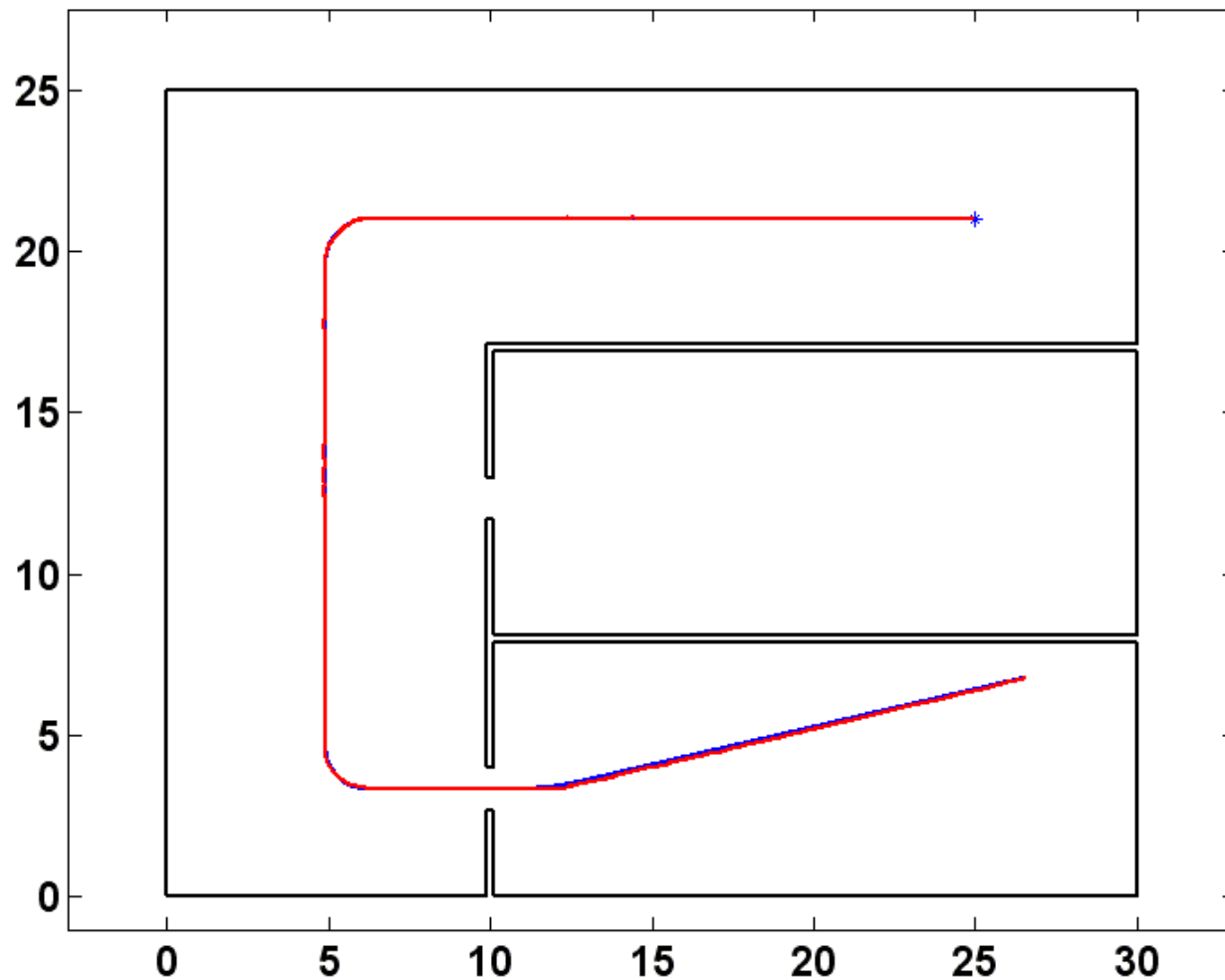
- ➡ Select a threshold and **reject the measurements which give an innovation larger than this threshold** (this results in a reduction of the number of available measurements but makes the filter more **robust**).

What to do if the estimate (in the EKF case) or a Sigma Point (in the UKF case) **falls outside** the map?



... consider the "second" nearest intersection...

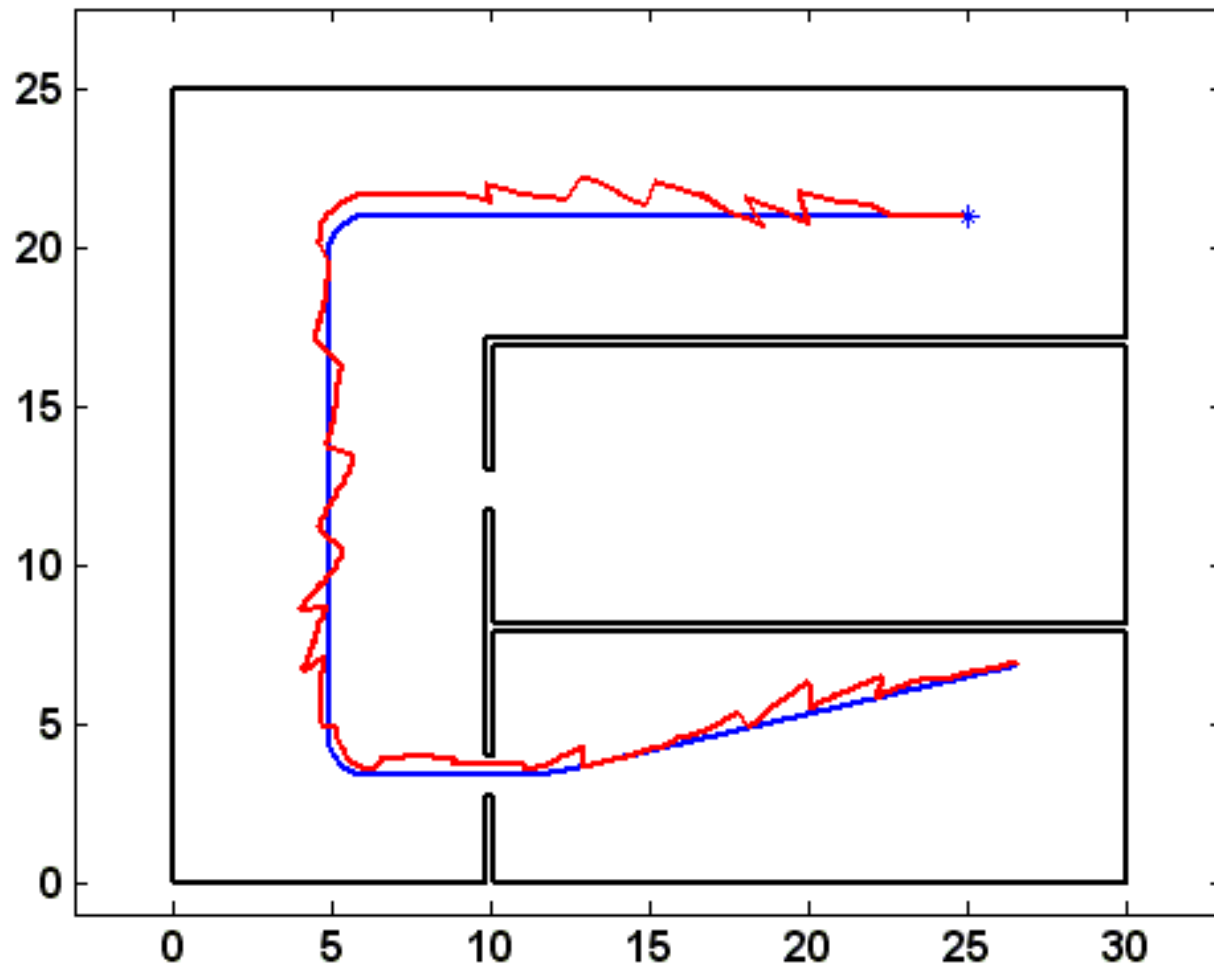
EKF estimation, $N_s = 10$:



$J = 0.15 \text{ m}$

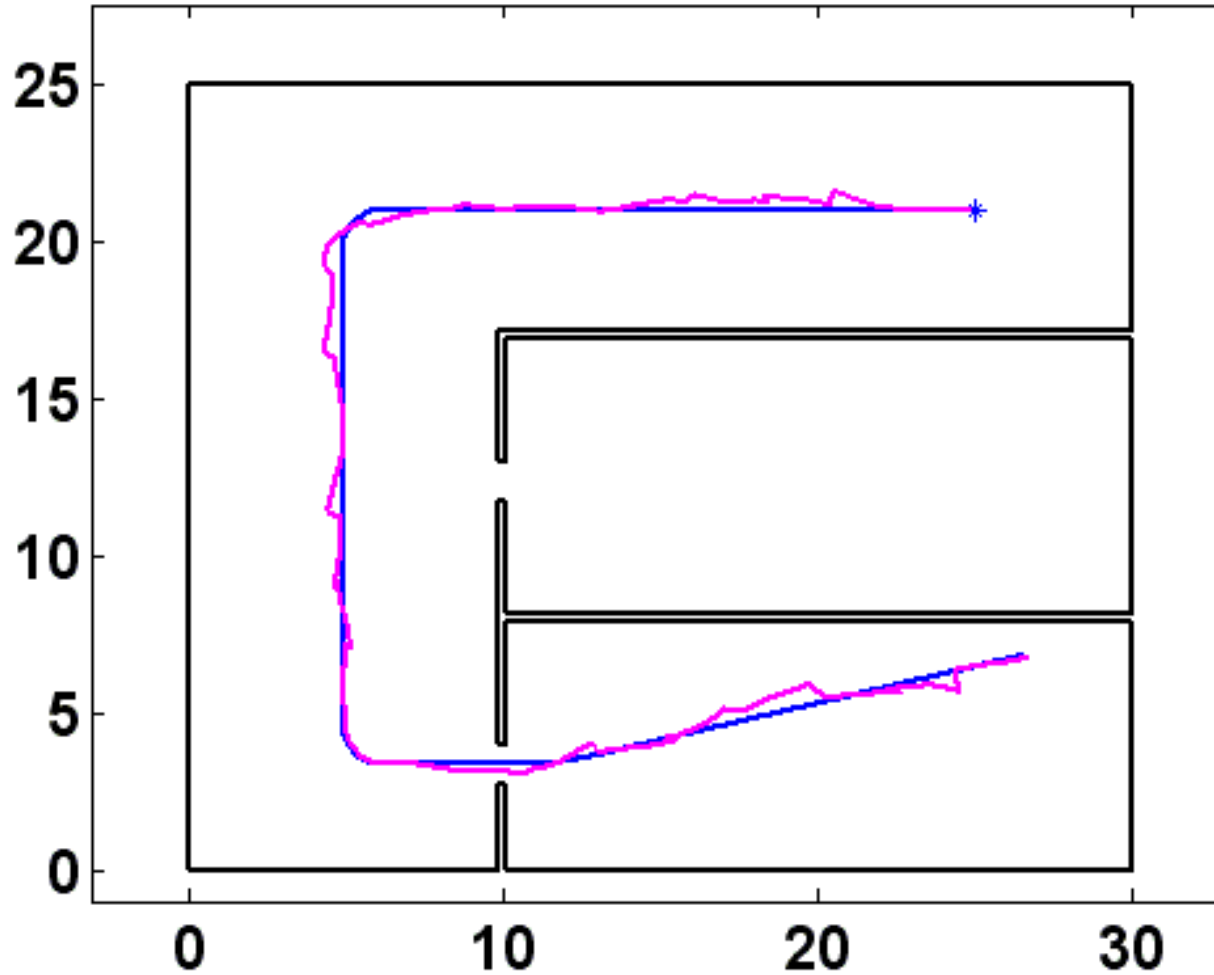
N_s = number of steps between two consecutive exteroceptive measurements, while odometry is assumed available at all 5000 steps

EKF estimation, $N_s = 200$:



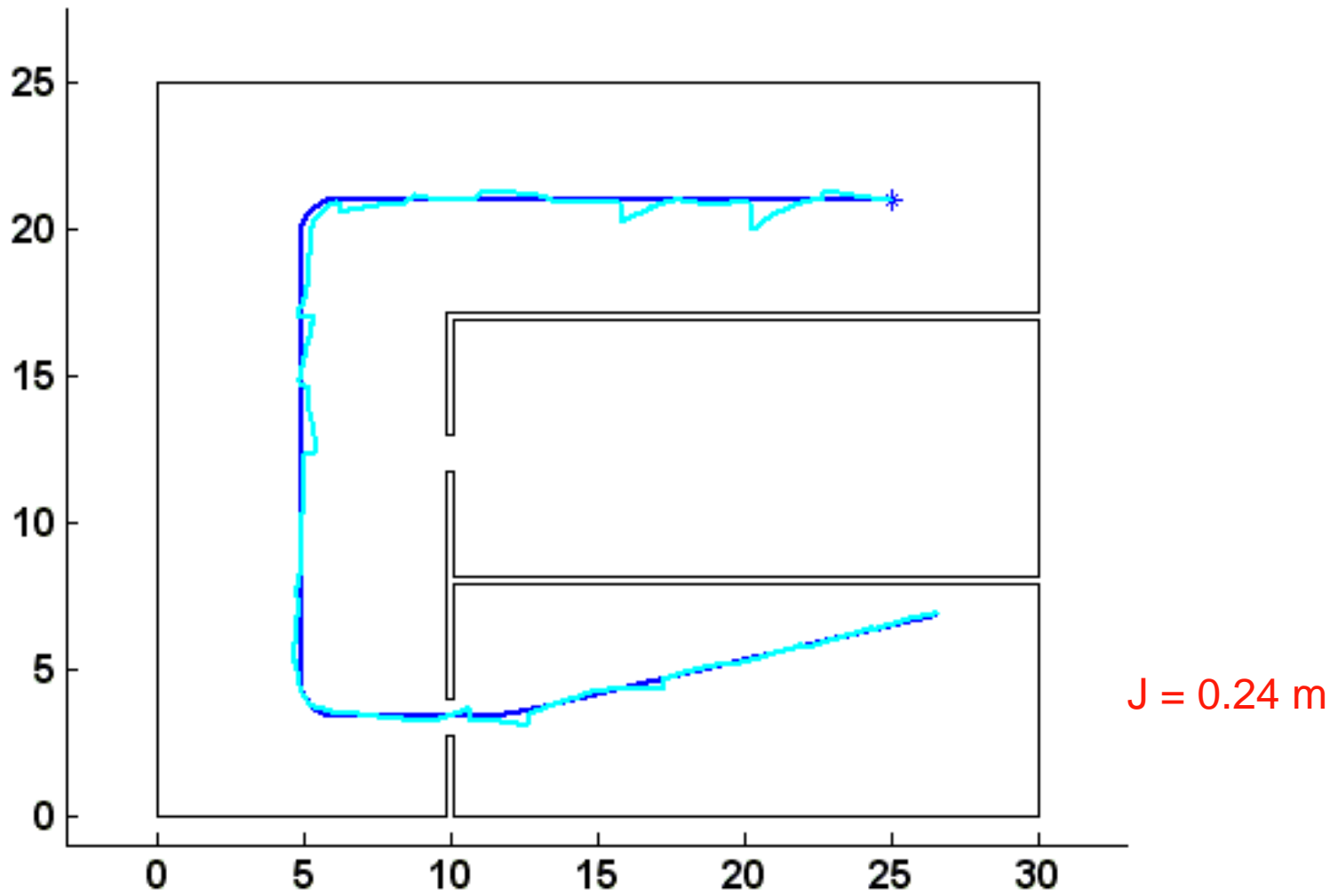
$J = 0.87 \text{ m}$

UKF estimation, $N_s = 200$:

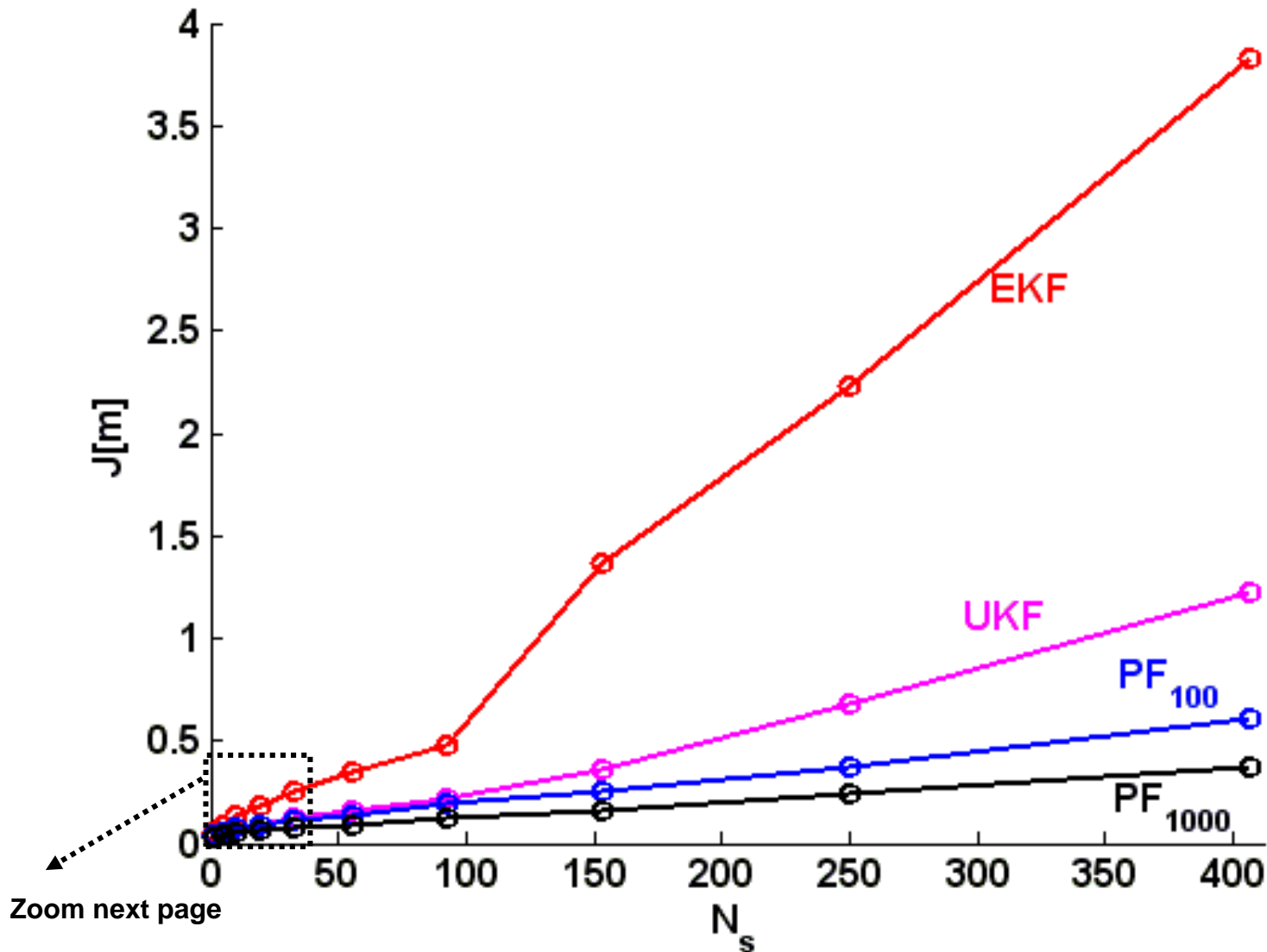


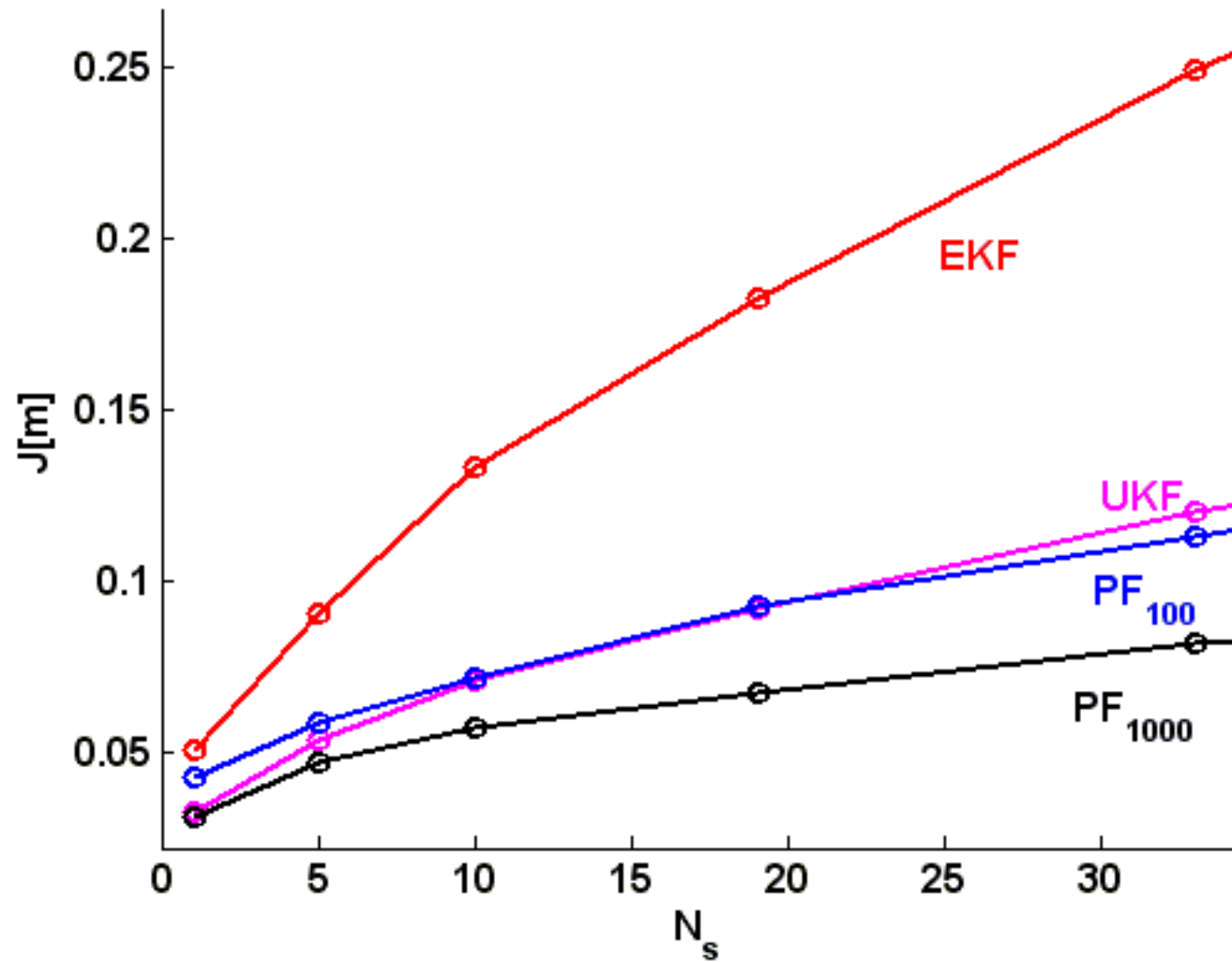
$J = 0.45 \text{ m}$

PF estimation, $M_p = 100$ particles, $N_s = 200$:



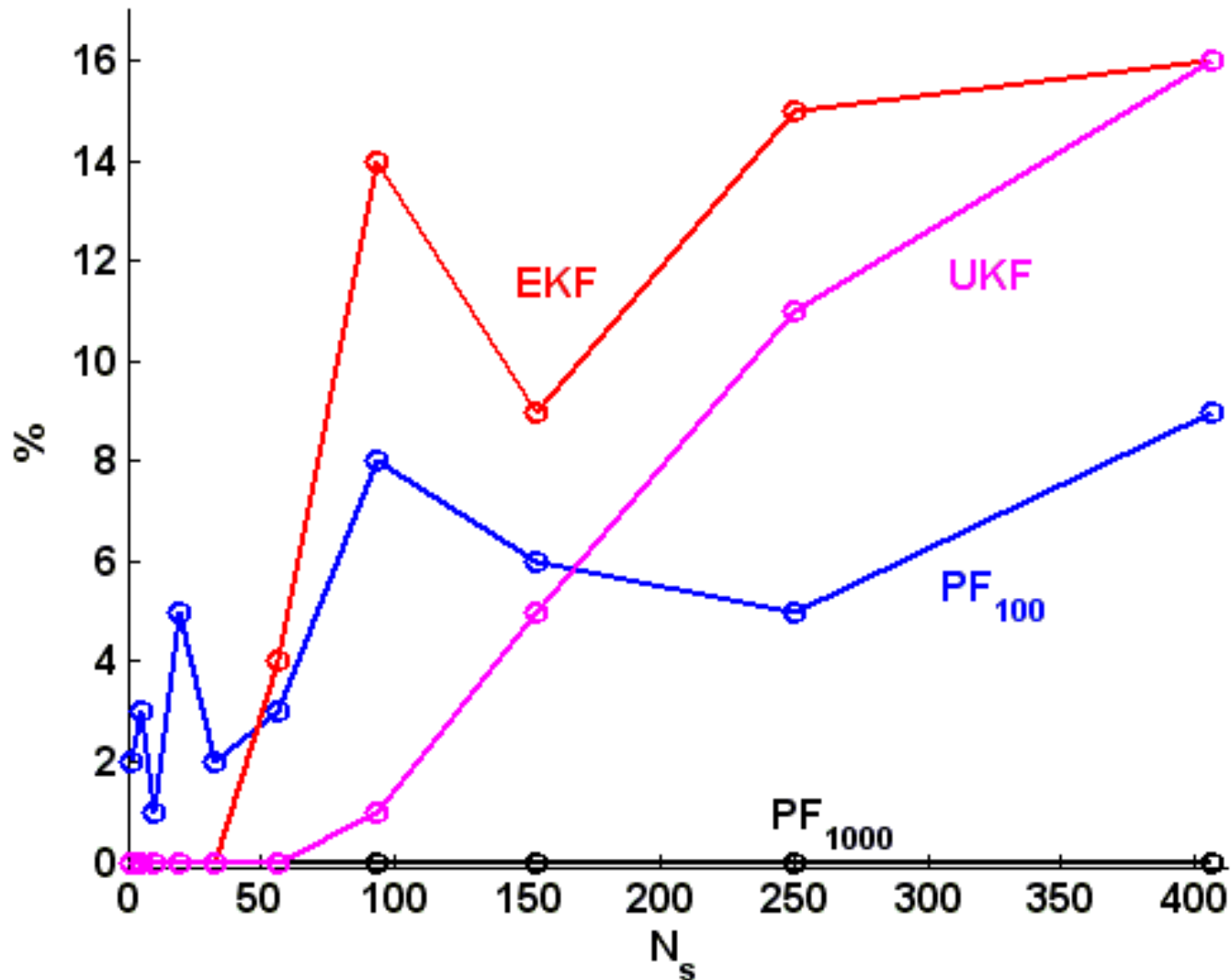
A comparison between EKF, UKF and PF is reported below for the range finder case (position tracking), after removing some outliers ($J_{odo}=7.8m$):





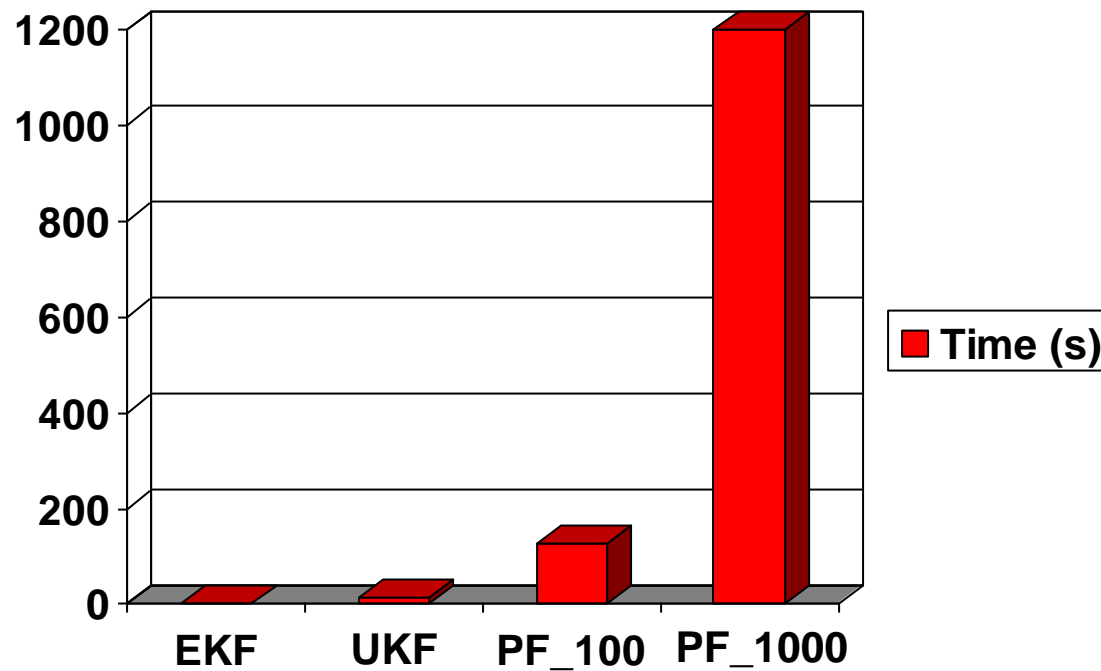
The **percentage of outliers** (over 100 independent simulation runs, 10 for PF_{1000}).
 J_k is considered an outlier if:

$$\frac{|J_k - \bar{J}|}{\bar{J}} > 1$$

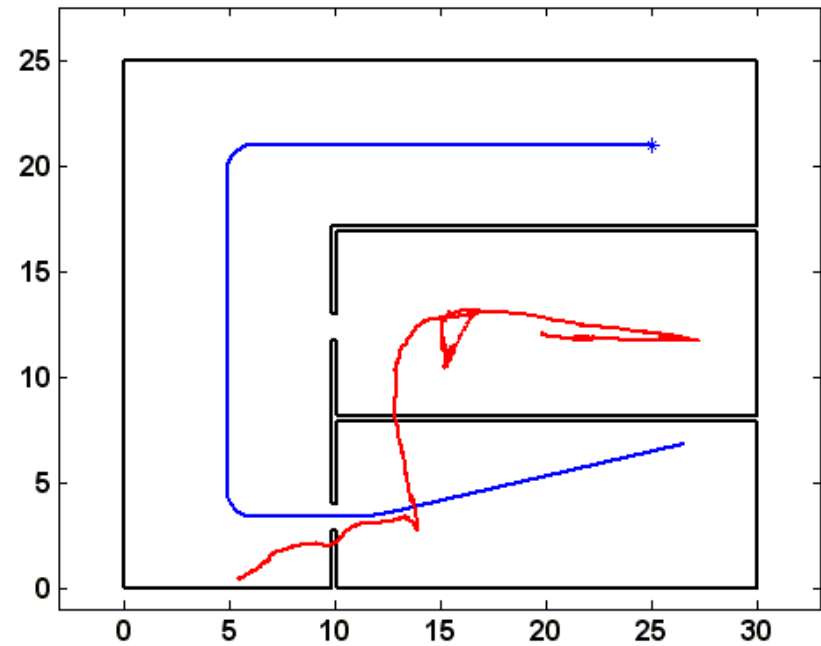
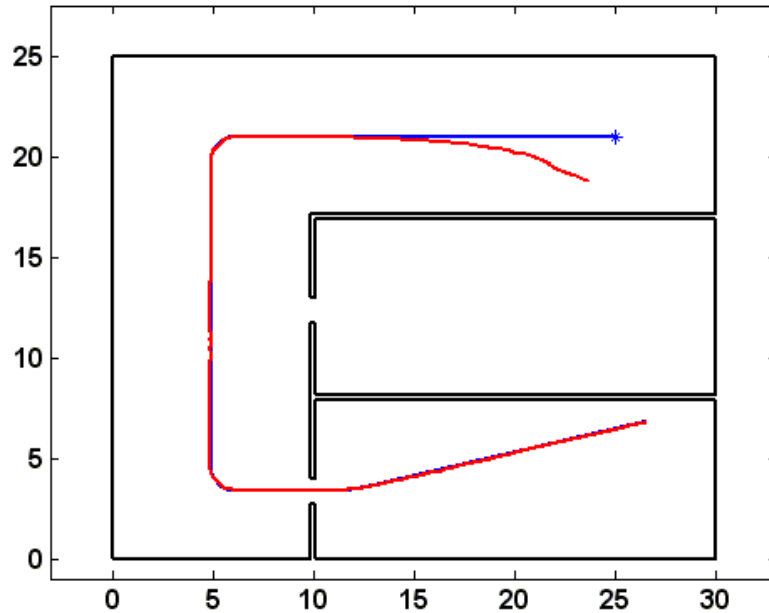


Computational times (for $N_s=1$):

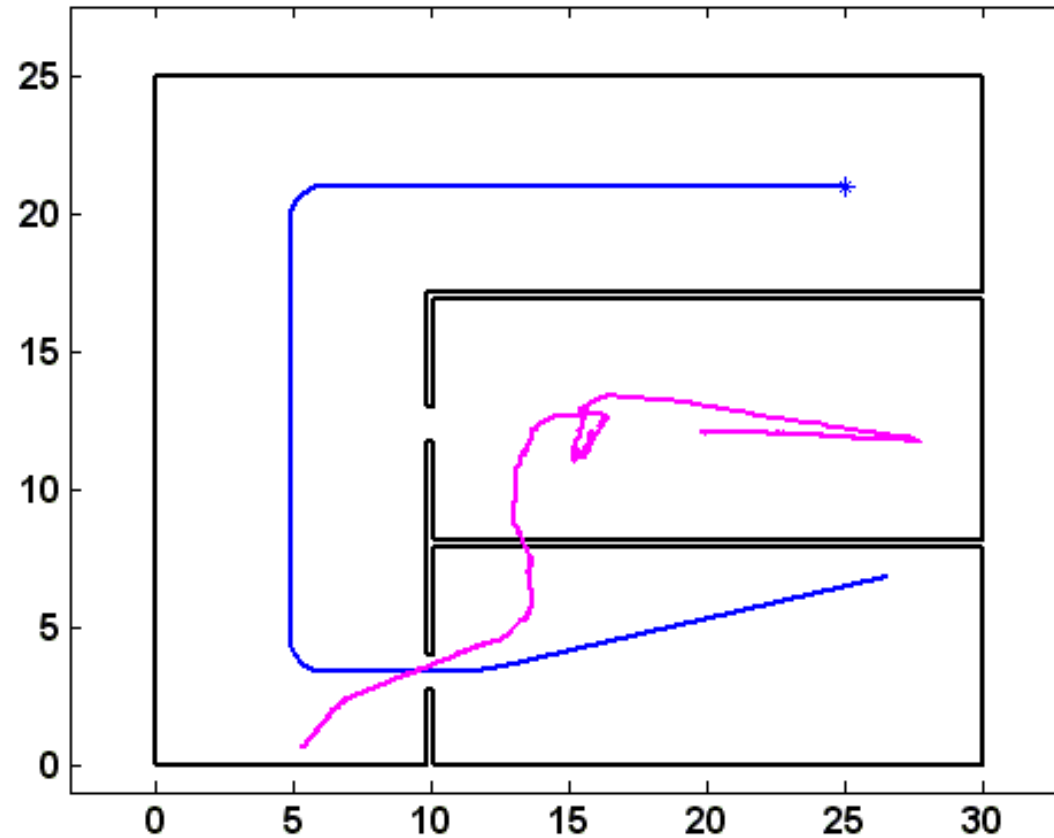
- EKF: **3 s**
- UKF: **15 s**
- PF, 100 particles: **127 s**
- PF, 1000 particles: **1198 s**



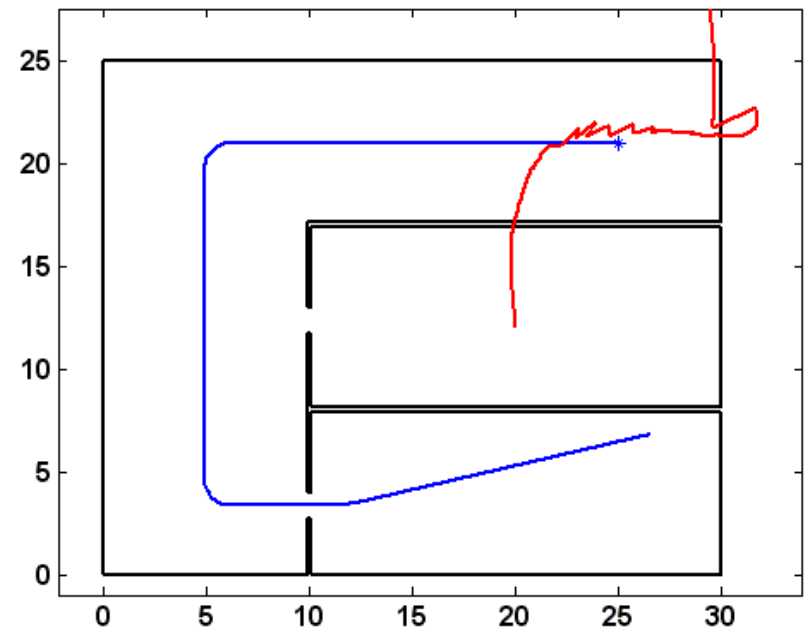
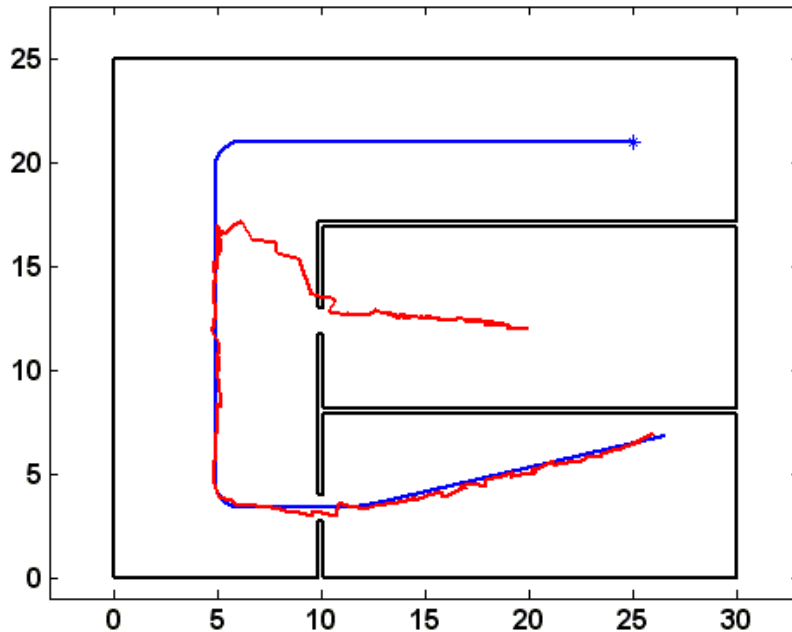
The EKF can not be applied to **global** localization problems ($N_s = 10$):



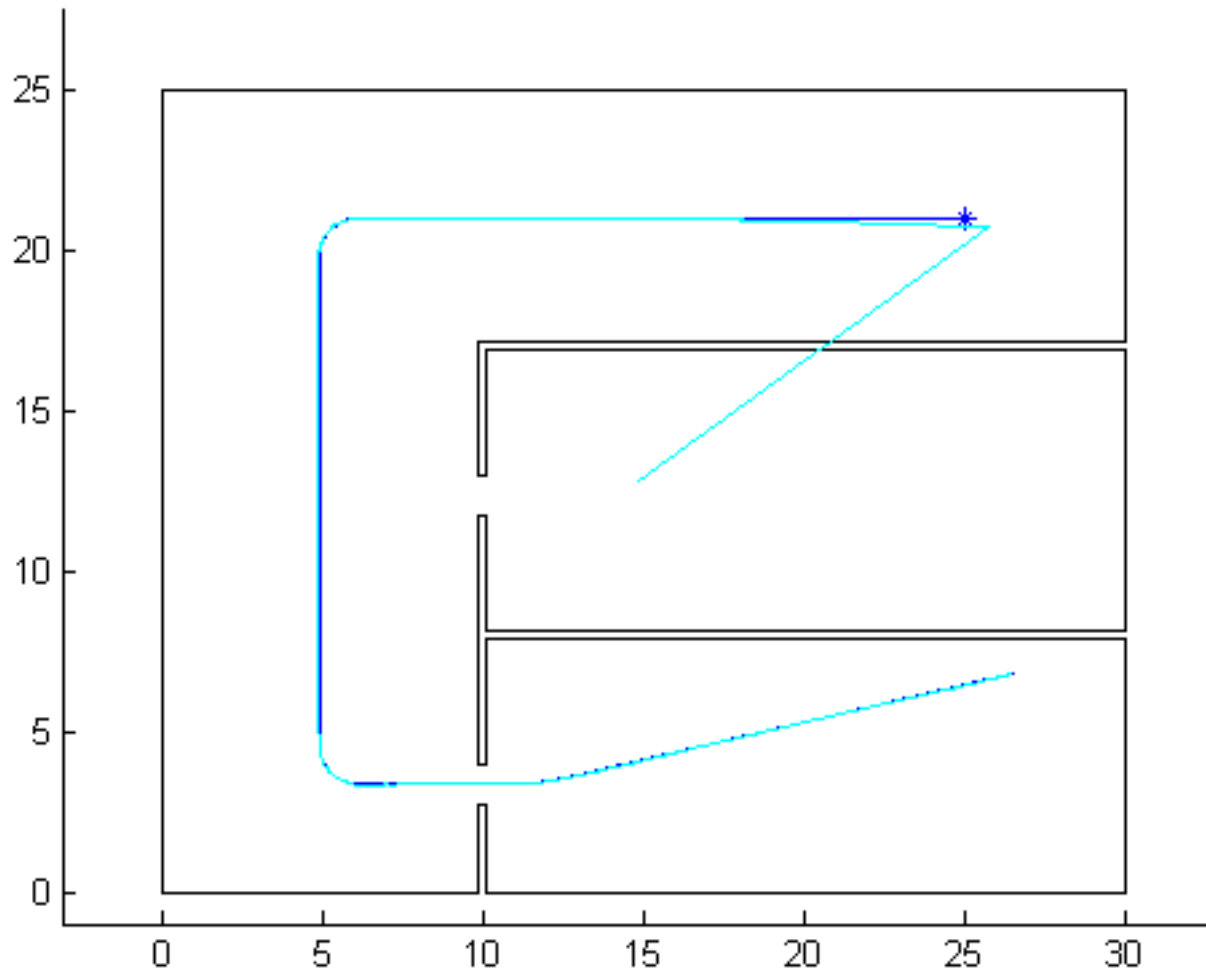
Similarly the UKF can not be applied to [global](#) localization problems ($N_s = 10$):



The EKF may work in the [global](#) case for suitable initial conditions of the estimation (left: the estimation of the initial orientation allows to recover the correct trajectory). Here $N_s = 100$.

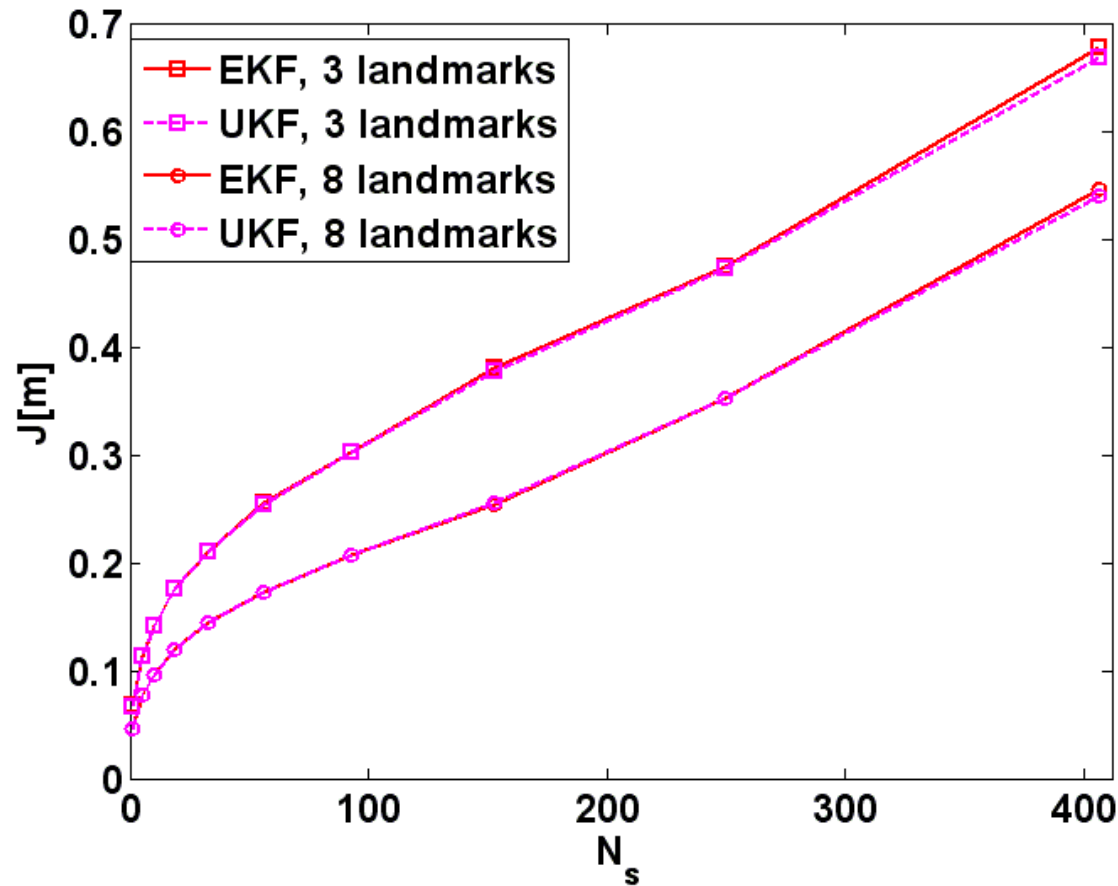


The PF can be applied to **global** localization problems if a sufficient number of particles is adopted (here $M_p = 1000$ and $N_s = 10$):



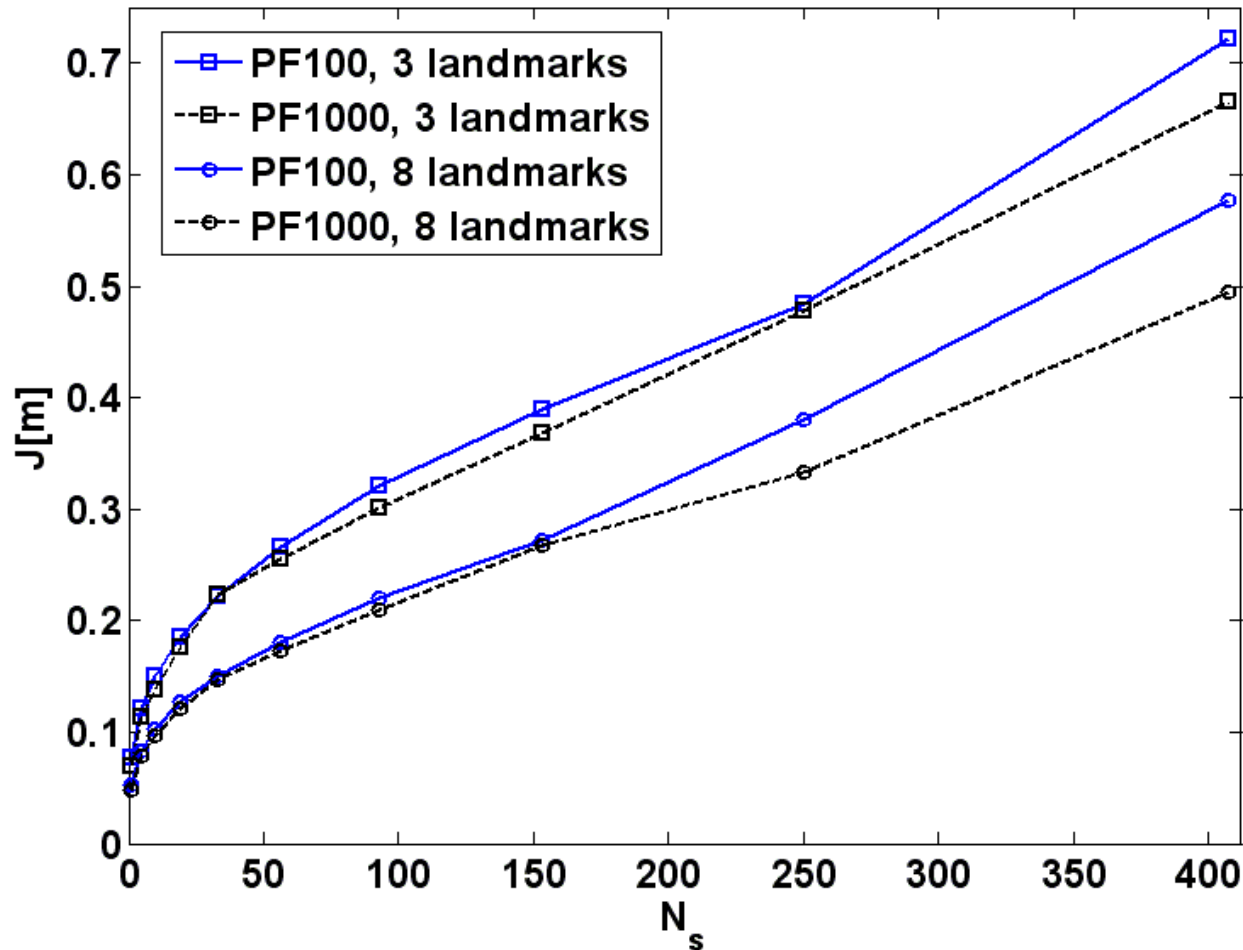
Landmark (range) case:

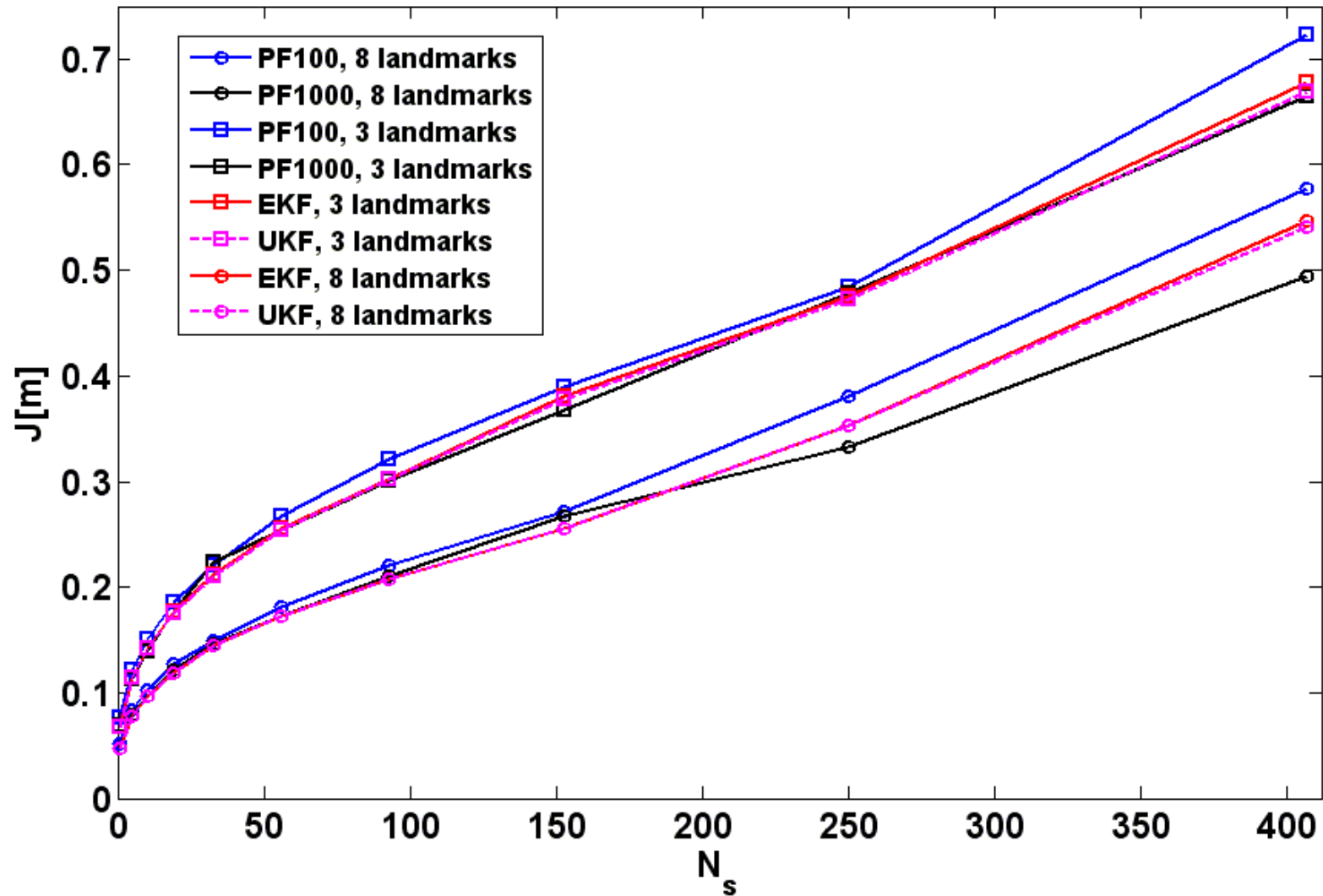
A comparison between **EKF** and **UKF** is reported below for the **landmark** case (position tracking):



In the landmark case the EKF and the UKF provide identical performances

A comparison between PF100 and PF1000 is reported below for the landmark case (position tracking):

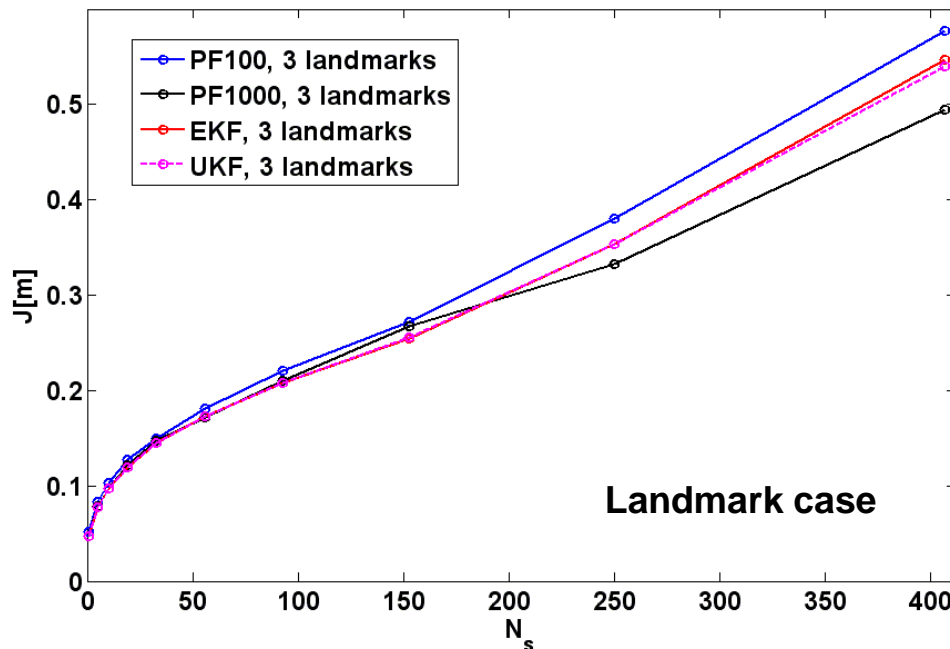
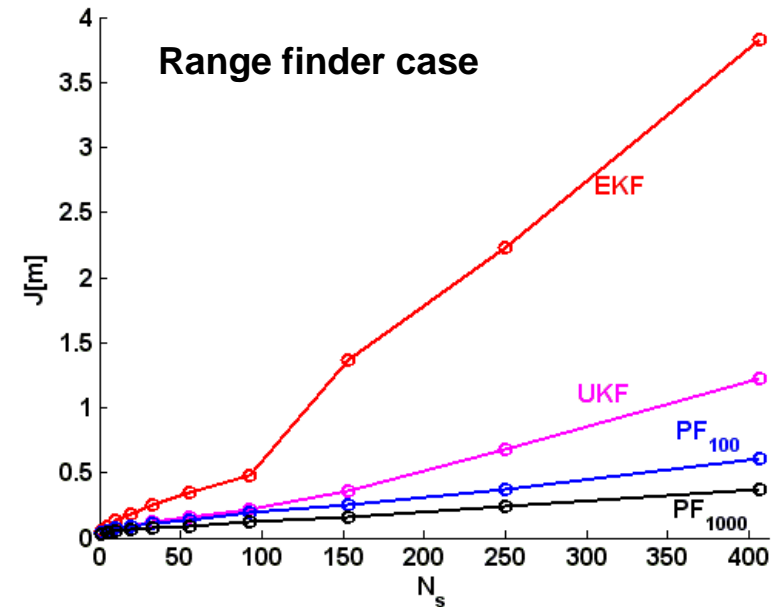




All the filters show **similar** performances...

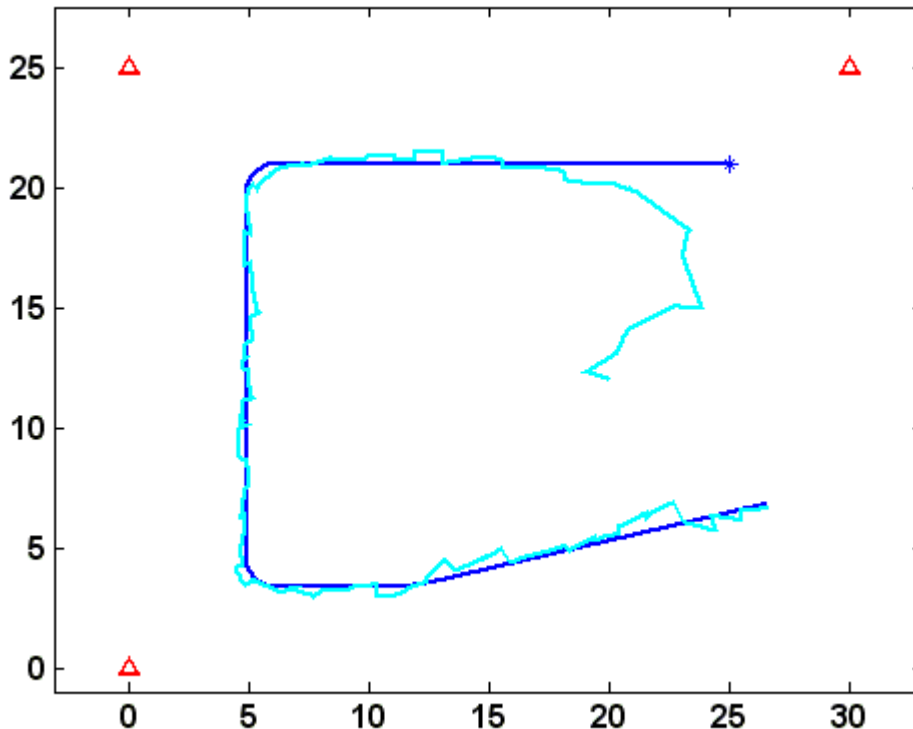
A comparison between the range finder and the landmark case (position tracking):

- in the landmark case all filters provide similar performance;
- in the landmark case the performance is higher



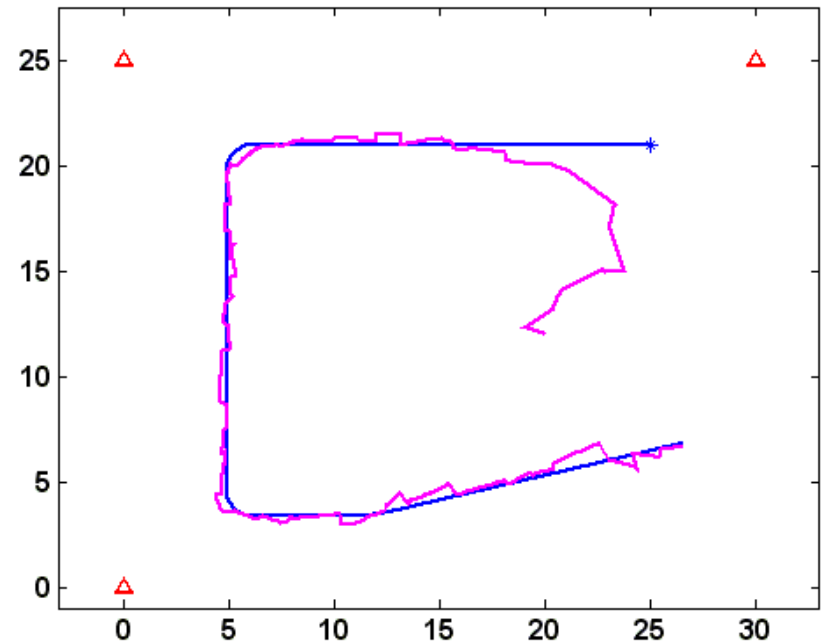
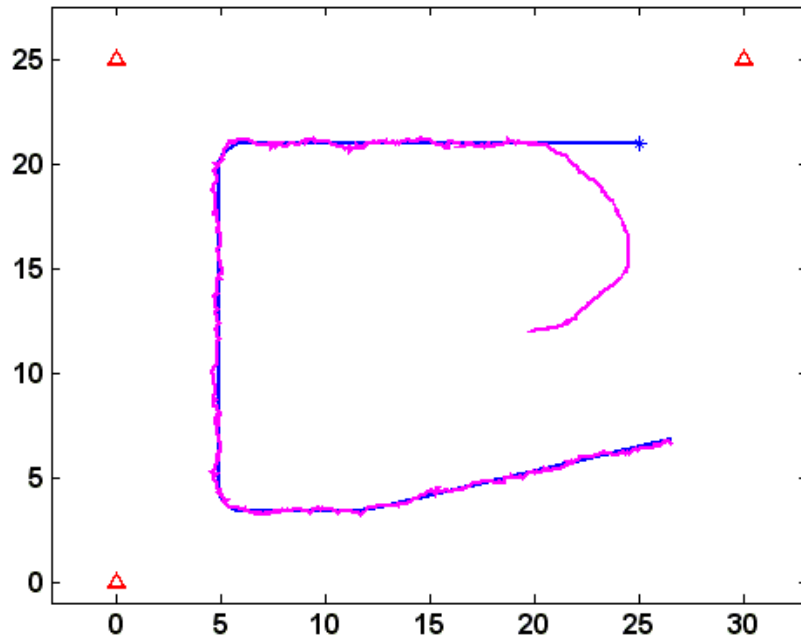
The [percentage of outliers](#) in the landmark case (over 100 independent simulation runs, 10 for PF_{1000}) is 0 for EKF, UKF and PF_{1000} . There is only 1% of outliers for PF_{100} .

In the [landmark](#) case the EKF and the UKF can also be applied to solve a [global](#) localization problem, unlike in the laser reading case (see also videos).



EKF estimation, $N_s = 100$

UKF estimation (global), $N_s = 10$ (left) and 100 (right)



Conclusions

- The localization of a mobile robot in a known environment can be cast as a **filtering problem**.
- The robot must be equipped with **enough sensors to guarantee observability**. Symmetric environment can also be problematic.
- The **theoretical solution** of the problem is the **Bayes filter** which usually cannot be implemented.
- **Gaussian or non parametric implementations** should be considered according to the considered problem (global vs position tracking, computational time).
- **Active localization** allows to improve localization capabilities of the robot.
- The **basic** problem can be **extended to: dynamic environments, multi-robot, partially known environments, etc.**

References

1. [S. Thrun, W. Burgard and D. Fox. "Probabilistic Robotics"](#). MIT Press, 2006
(and references therein)

Robot motion models (Ch. 5), measurement models (Ch. 6), Localization (Ch. 7-8), Filtering (Ch. 2-4), Active localization (Ch. 17).

2. [Observability issues](#):

- R. Herman and A. Krener. "Nonlinear controllability and observability". IEEE Transactions on Automatic Control, n. 22(5), p. 728-740, 1977
- A. Martinelli and R. Siegwart. "Observability analysis for mobile robot localization". Intelligent Robots and Systems, 2005. (IROS 2005), p. 1471 - 1476. Edmonton, Alberta, Canada. August 2005

3. [A. Censi. "On achievable accuracy for range-finder localization"](#). 2007 IEEE International Conference on Robotics and Automation, (ICRA 2007), p. 4170 - 4175. Roma, Italy. April 2007