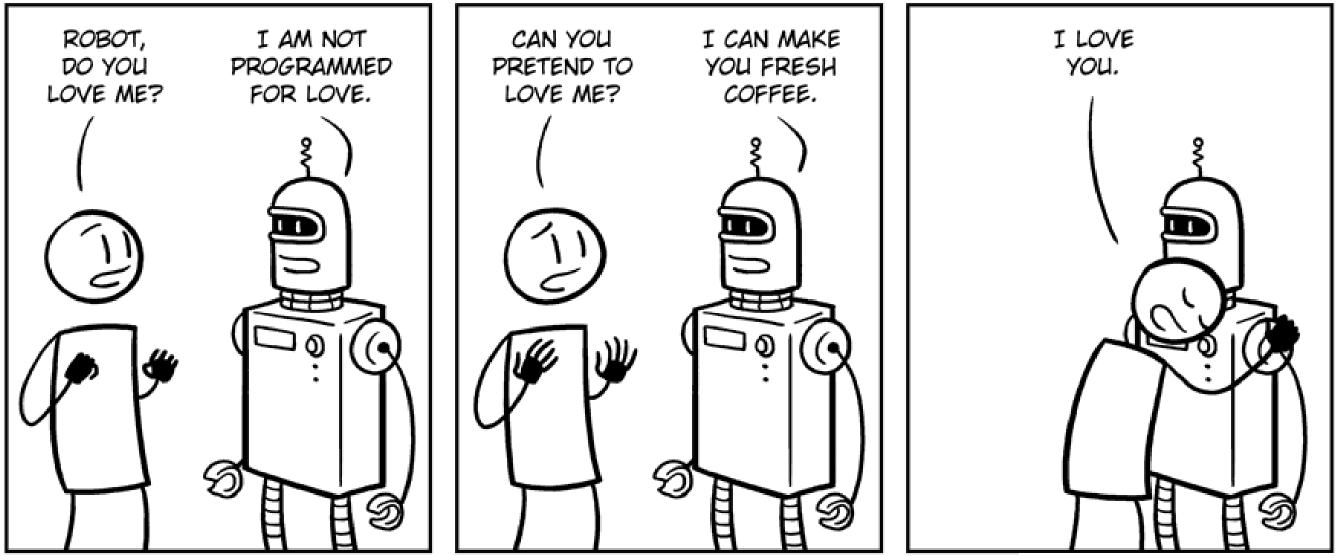
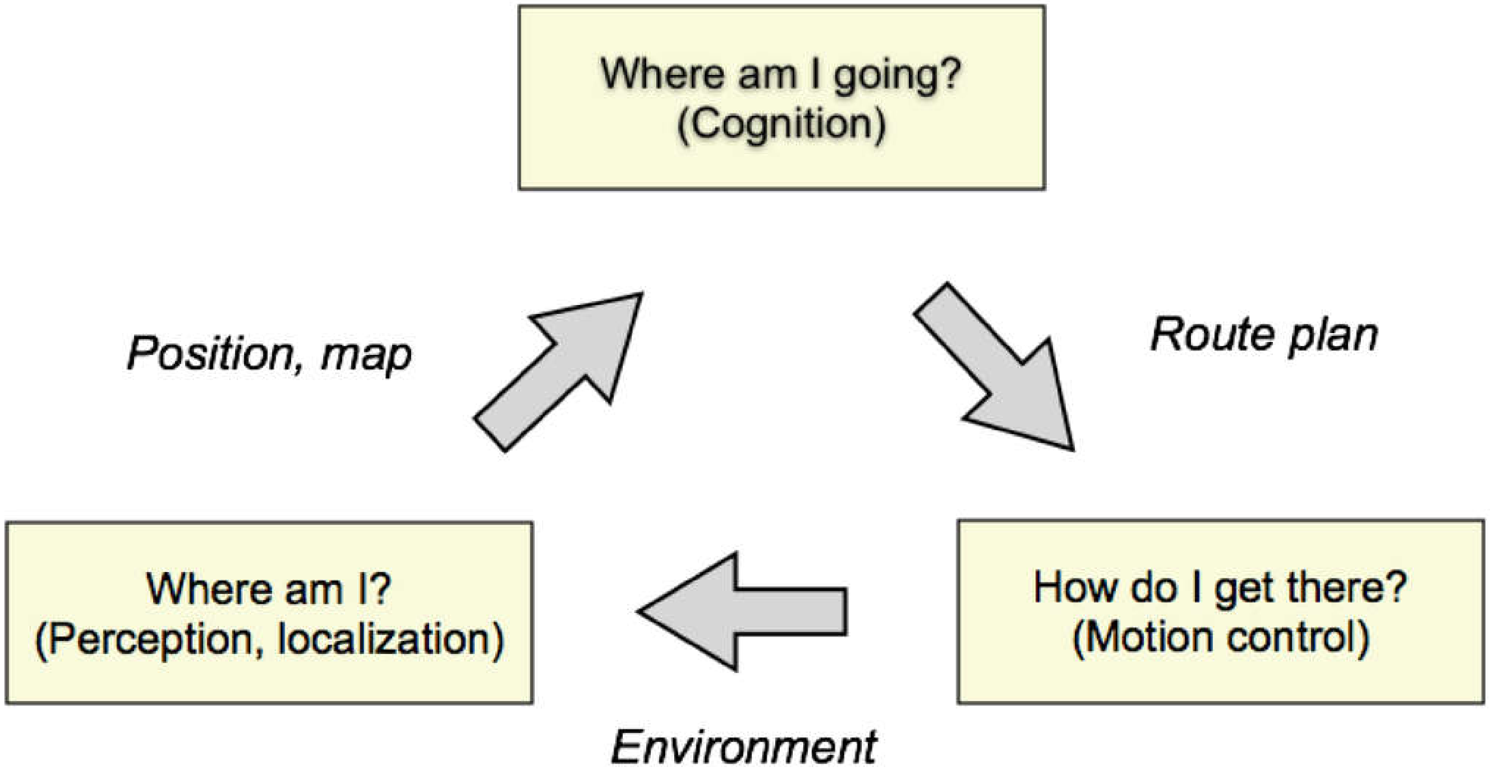
Localization



Where am I ?


Where am I going?
(Cognition)
Position, map
Route plan
Where am I?
(Perception, localization)
How do I get there?
(Motion control)
Environment

Dead Reckoning

- intrinsic sensors

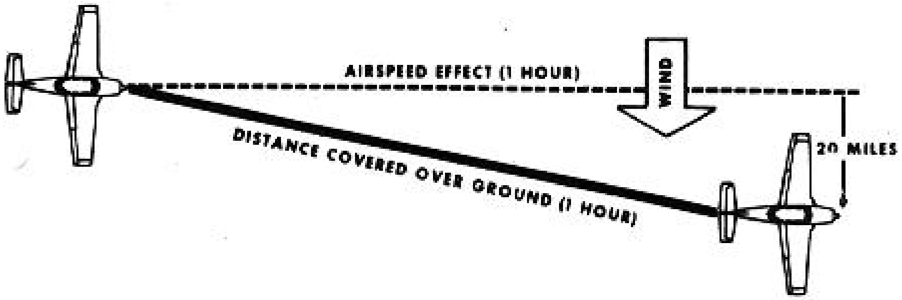
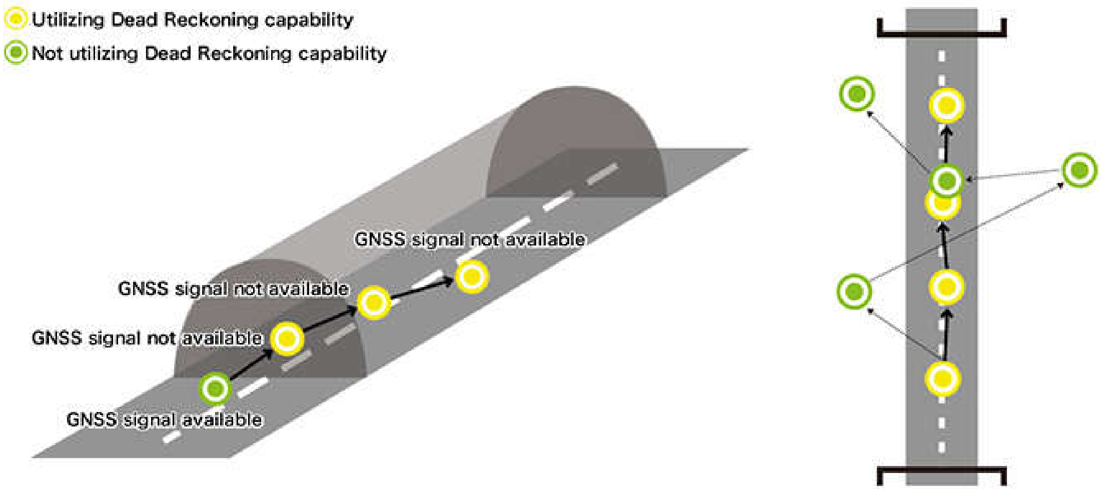


Figure 12–4 Effect of Wind in One Hour



Ground tracking in tunnels where the GPS/GNSS signals are shielded and unavailable

Landmark-based navigation

- Extrinsic sensors

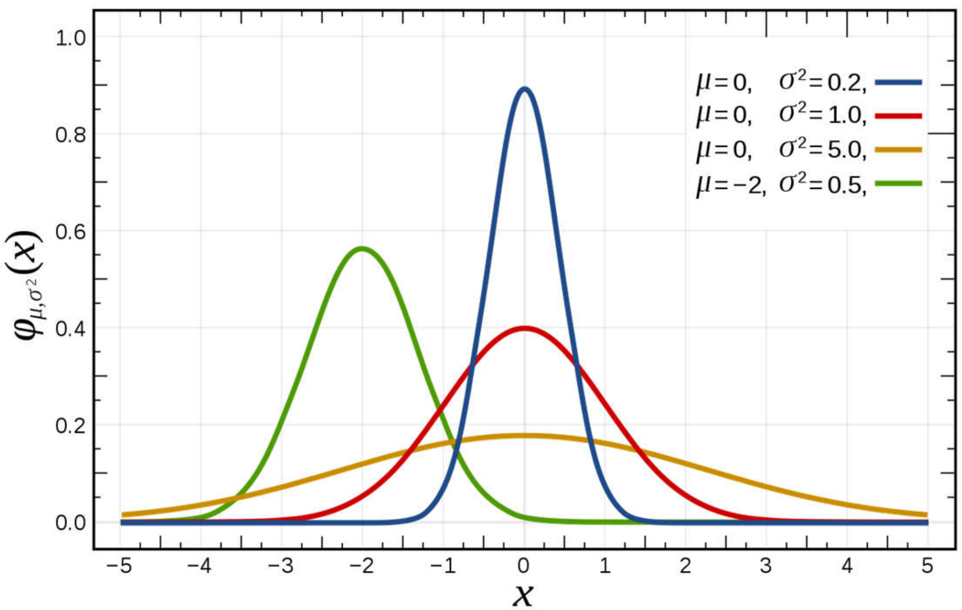
Motion model

$$x(n+1) = x(n) + \delta_x + v_x$$

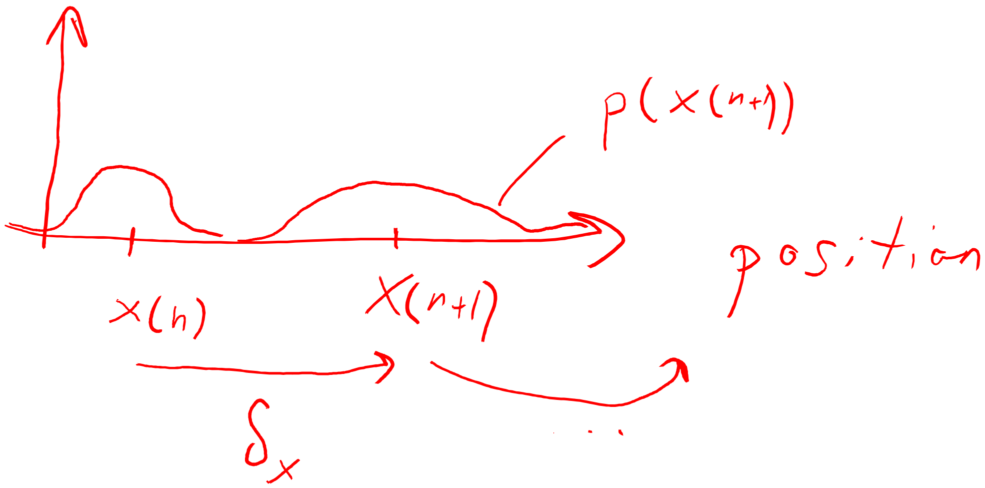
stø

$x(n)$ - position at time n
δ_x - "delta x", change in position during time step
v_x - noise term (e.g. $v_x \sim \mathrm{N}(0, \sigma^2)$) varians

middel



Motion model



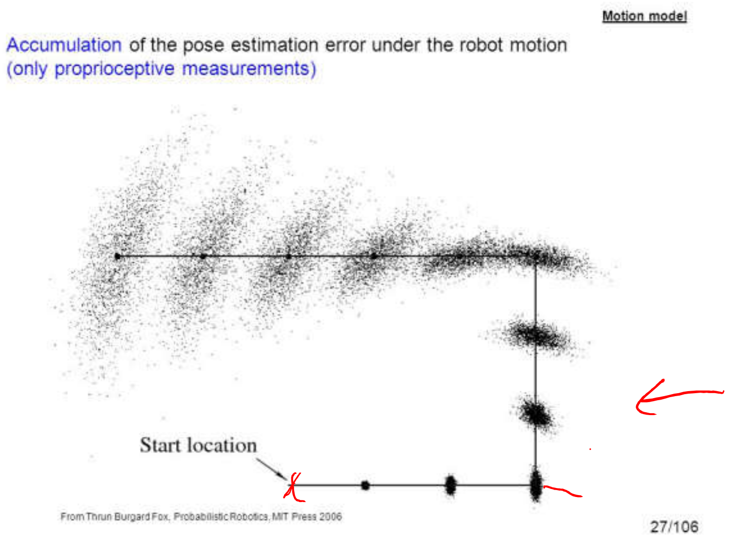
Motion models

Linear model :

$$x(n+1) = x(n) + \delta_x + v_x$$

General model :

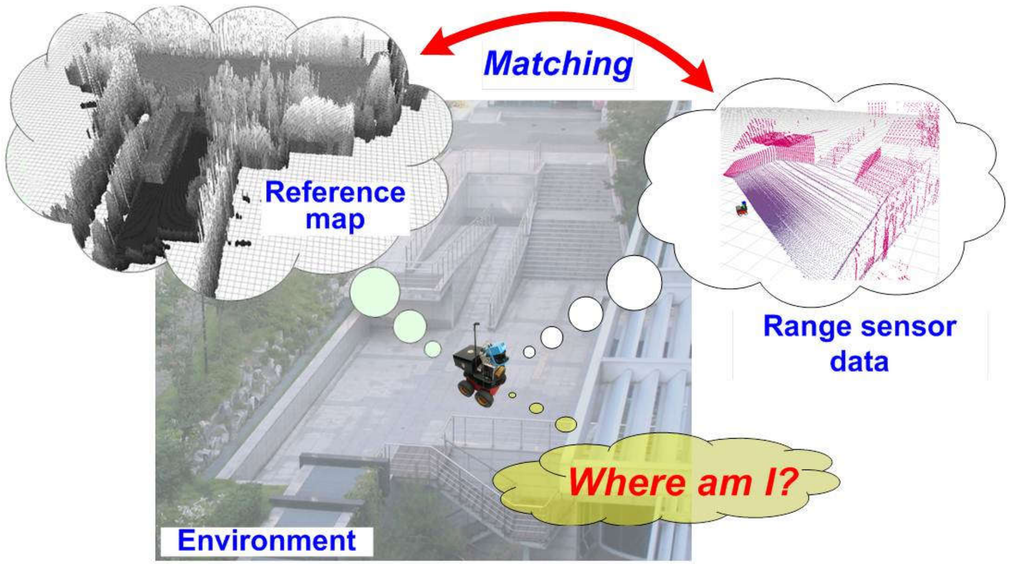
$$x(n+1) = f(x(n), u(n), v(n))$$



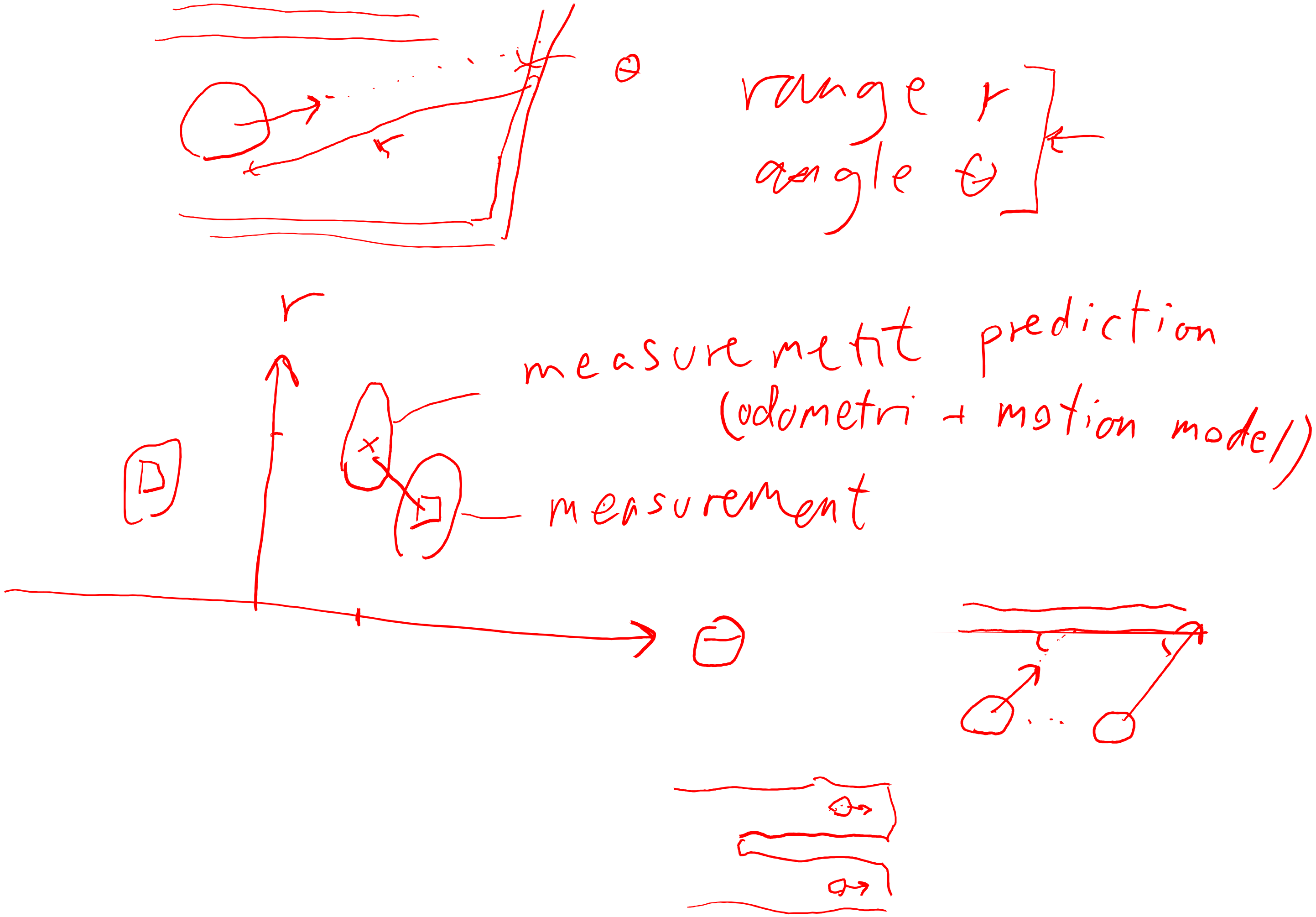
Measurement model and matching/data association

$$\mathrm{z}(n) = h(x(n), w(n), map/features)$$

z(n) – measured position
x(n) – true position
w(n) – noise on measurement



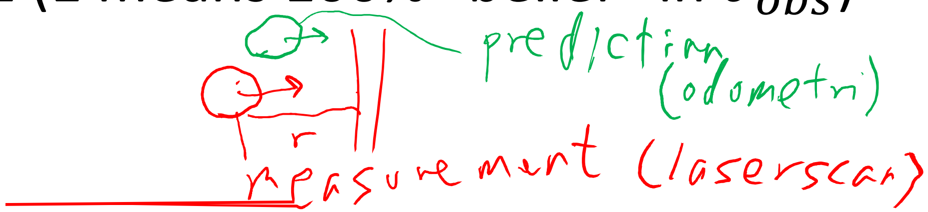
Eksempel – range/angle



1. Simple measurement update

a) Prediction : Update pose using motion model
b) Measurement : measure (range, angle)-data to wall with depth sensor
c) Measurement prediction : find (range, angle)-data from the map and estimated pose
d) Matching : data association between measurements and measurement predictions
e) Pose estimation : Use e.g. simple filter to update angle - $\hat{\theta}(n+1) = \hat{\theta}(n) + k_1\left(\theta_{obs} - \hat{\theta}(n)\right)$

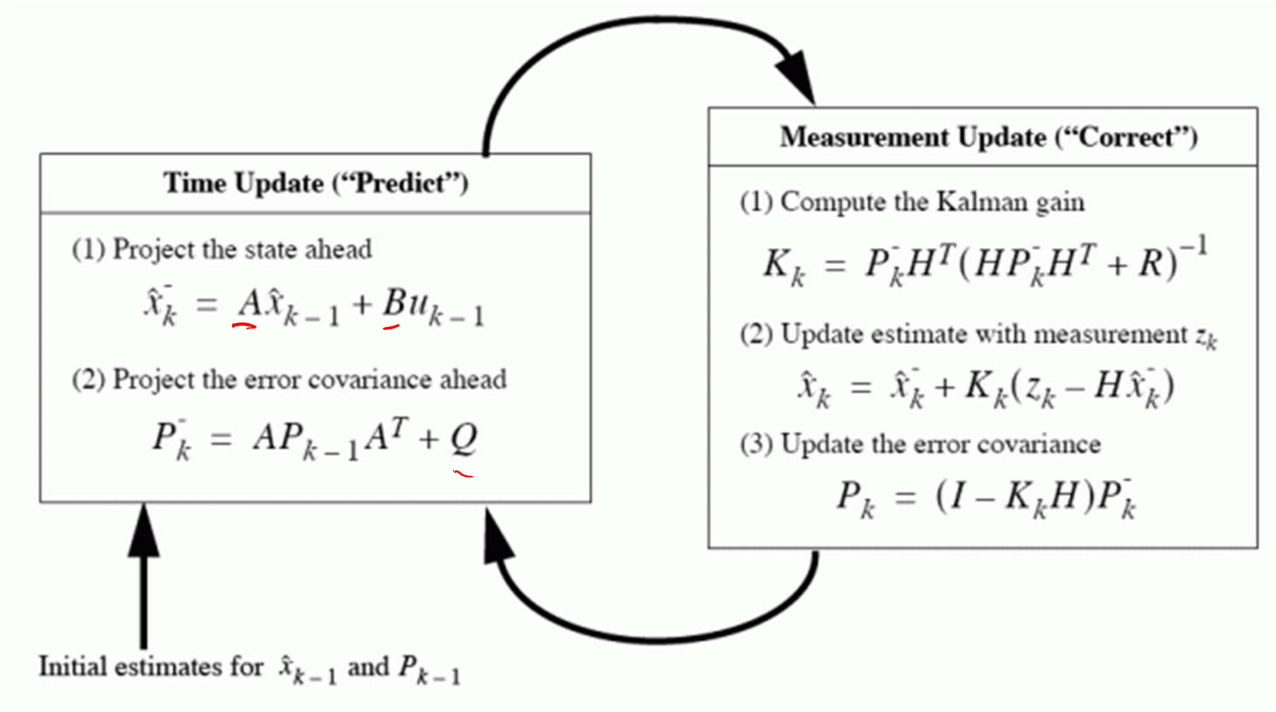
$\hat{\theta}(n)$=current angle estimate, θ_{obs}=observed angle, k_1 = fixed parameter between 0 and 1 (1 means 100% ”belief” in θ_{obs})



2. Kalman filter

$$x(n+1) = Fx(n) + v(n)$$ motion model

$$z(n+1) = Hx(n+1) + w(n)$$ måling



3. Extended Kalman Filter

$$x(n+1) = f(x(n), v(n))$$

$$\mathrm{z}(n+1) = h(x(n+1), \mathrm{w(n)})$$

Linearization

$$x(n+1) = f\big(\hat{x}(n)\big) + F_x(x(n) - \hat{x}(n)) + F_v v(n)$$

$$\mathrm{z}(n+1) = h\big(\hat{x}(n+1)\big) + H_x(x(n+1) - \hat{x}(n+1)) + H_w w(n)$$

3. Extended Kalman Filter

k

Prediction step

State prediction and measure estimation

$$\tilde{X}_{i,k} = f(\hat{X}_{i,k-1}, U_{i,k}, 0)\big|_{odometry}$$

$$Z_{i,k} = g(\hat{X}_{i-1,k-1}, 0)\big|_{laser}$$

Prediction of error covariance matrix

$$\tilde{P}_{i,k} = J_{f,X_i} P_{i,k-1} J^T_{f,X_i} + J_{f,W} \Sigma_w J^T_{f,W}$$

Correction step

Kalman gain calculation

$$K_{i,k} = \tilde{P}_{i,k} J^T_{g,\bar{X}_i} \left(J_{g,\bar{X}_i} \tilde{P}_{i,k} J^T_{g,\bar{X}_i} + J_{g,V} \Sigma_V J^T_{g,V} \right)^{-1}$$

State correction

$$\hat{X}_{i,k} = \tilde{X}_{i,k} + \Theta_k K_{i,k} \left[Z_{i,k} - \tilde{X}_{i,k} \right]$$

Covariance matrix update

$$P_{i,k} = \left(I - K_{i,k} J_{g,\bar{X}_i} \right) \tilde{P}_{i,k}$$

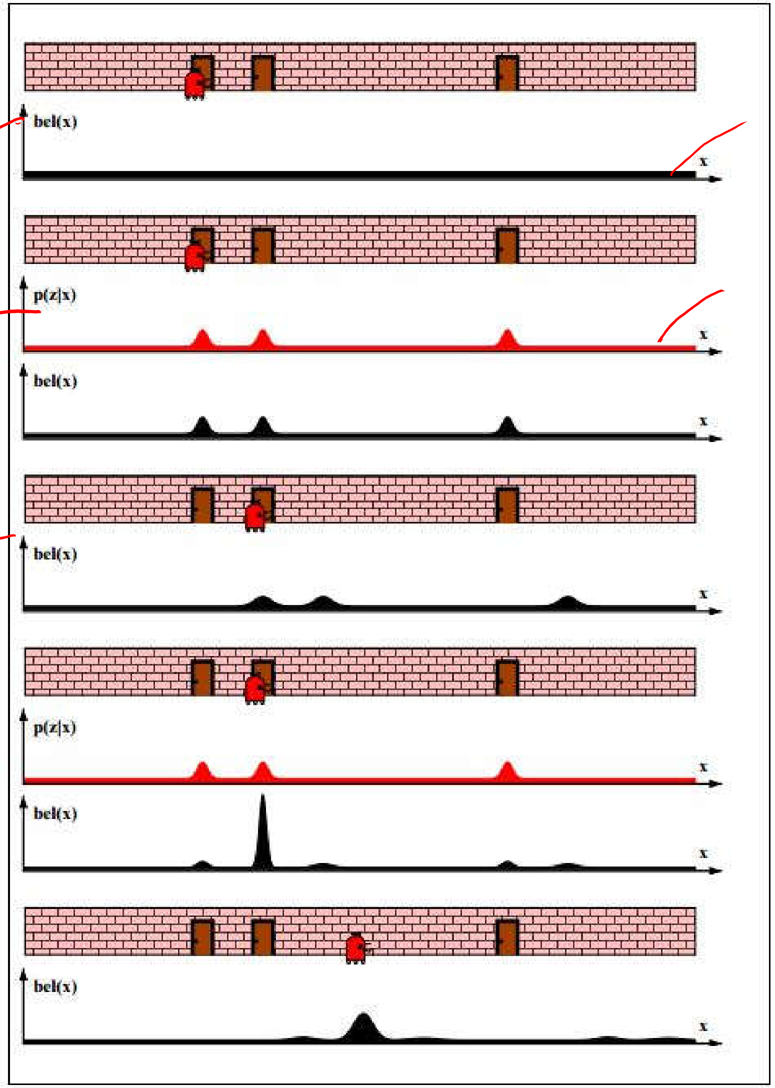
$k+1$

Probabilistic Robot Localization

belief

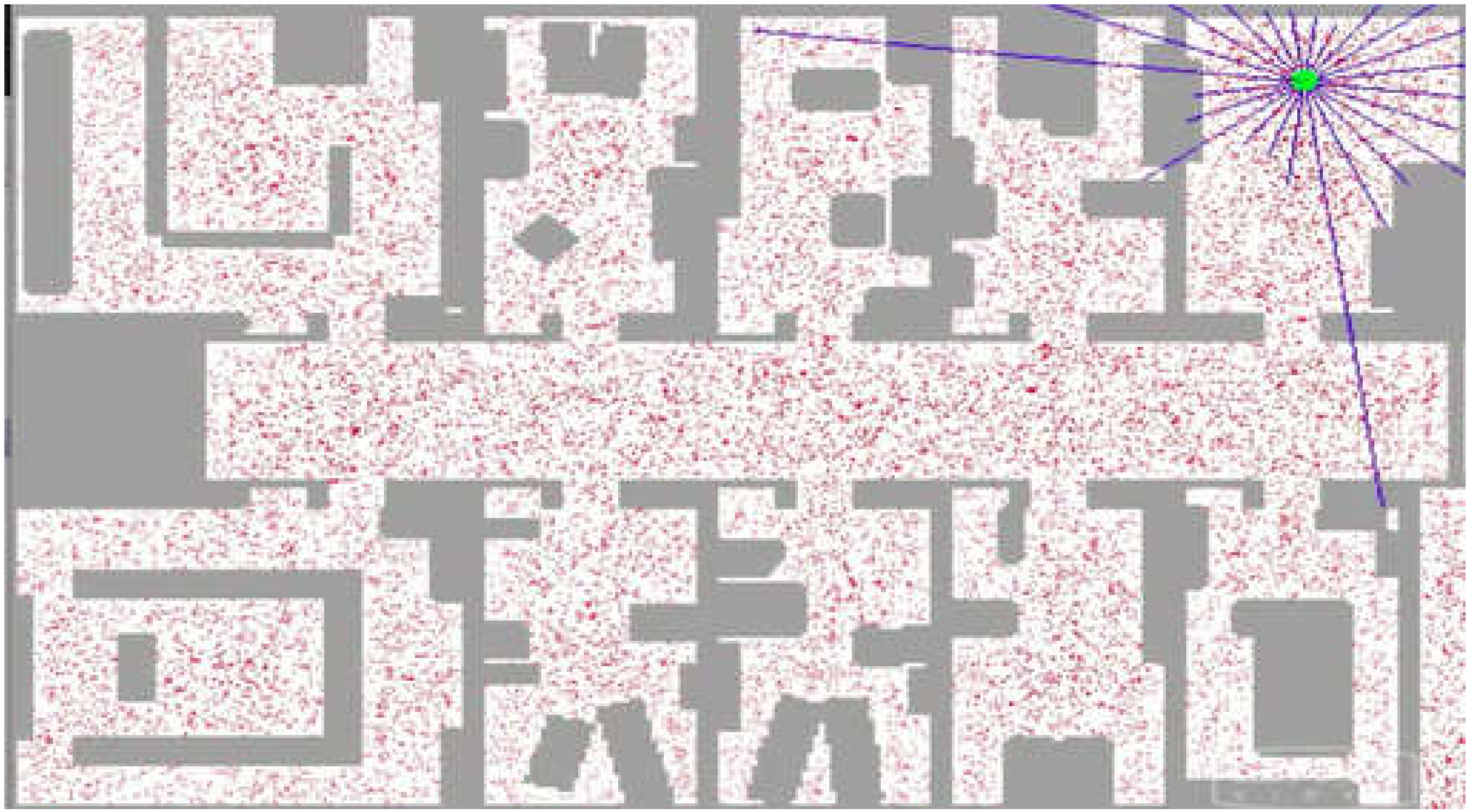
måling

motion model



4. Particle Filter (=Sequential Monte Carlo)

$P(x(n+1)|x(n), u(n))$ $P(z(n+1)|x(n+1), map)$



4. Particle Filter (Monte carlo simulation)

a) Initialize : N particles randomly distributed
b) Motion model for each particle (stochastic)
c) Step 1b) to 1d) -> how ”close” are measurements to map
d) Update weights for each particle (based on ”closeness” from c)
e) Resampling : Remove/add particles based on weights (stochastic)
f) Iterate b) to e)

Simultaneous Localisation and Mapping (SLAM)