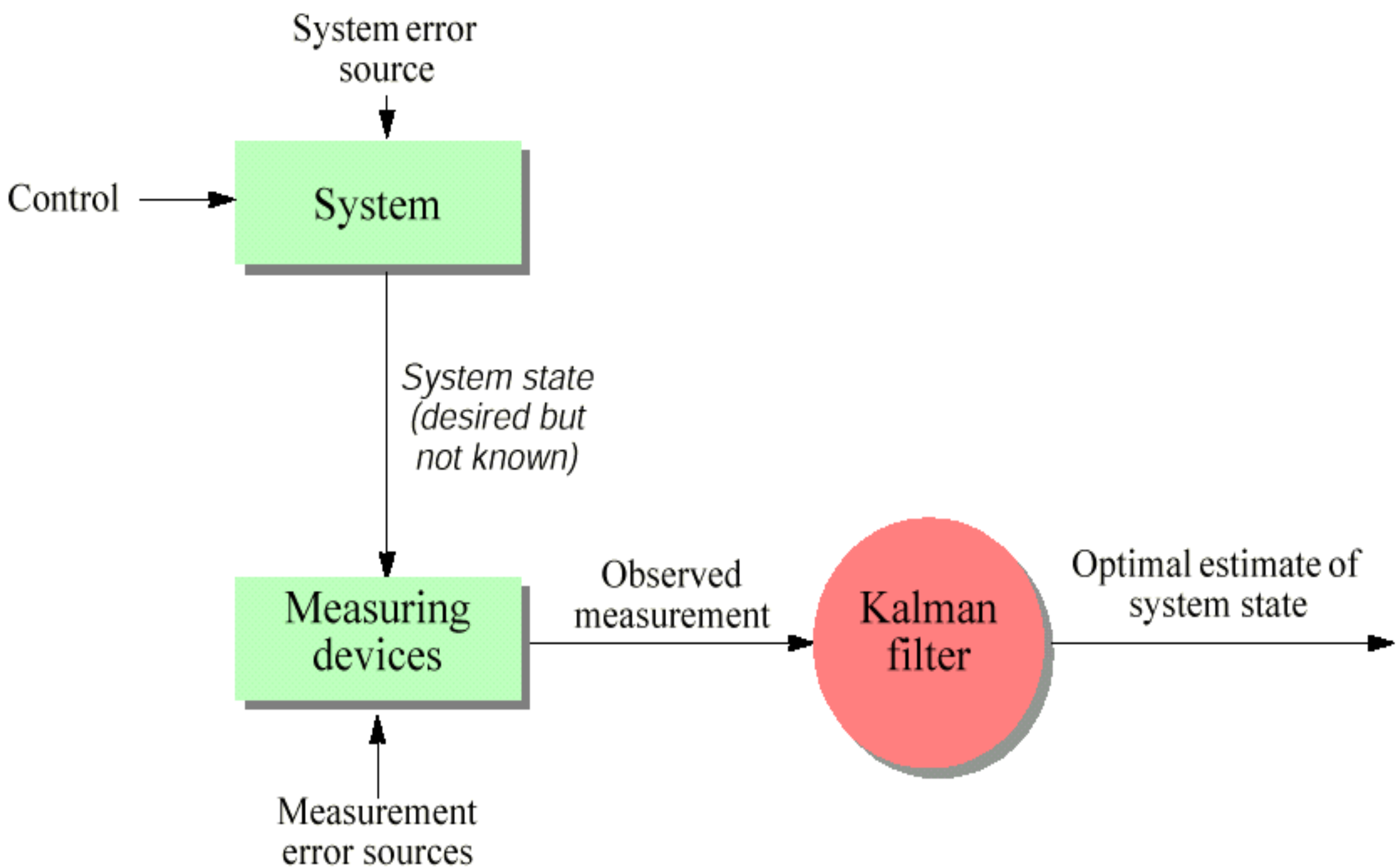
Probabilistic Robotics

Bayes Filter Implementations

Gaussian filters

Kalman Filter Localization



Bayes Filter Reminder

$$Bel(x_t) = \eta \ P(z_t \mid x_t) \int P(x_t \mid u_t, x_{t-1}) \ Bel(x_{t-1}) \ dx_{t-1}$$

Algorithm **Bayes_filter**(*Bel(x),d*):

η=0

If *d* is a perceptual data item *z* then

  For all *x* do

    $Bel'(x) = P(z \mid x) Bel(x)$

    $\eta = \eta + Bel'(x)$

  For all *x* do

    $Bel'(x) = \eta^{-1} Bel'(x)$

Else if *d* is an action data item *u* then

  For all *x* do

    $Bel'(x) = \int P(x \mid u, x') \ Bel(x') \ dx'$

Return *Bel′(x)*

Bayes Filter Reminder

- Prediction

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1})\, bel(x_{t-1})\, dx_{t-1}$$

- Correction

$$bel(x_t) = \eta\, p(z_t \mid x_t)\, \overline{bel}(x_t)$$

Kalman Filter

- Bayes filter with **Gaussians**
- Developed in the late 1950's
- Most relevant Bayes filter variant in practice
- Applications range from economics, wheather forecasting, satellite navigation to robotics and many more.

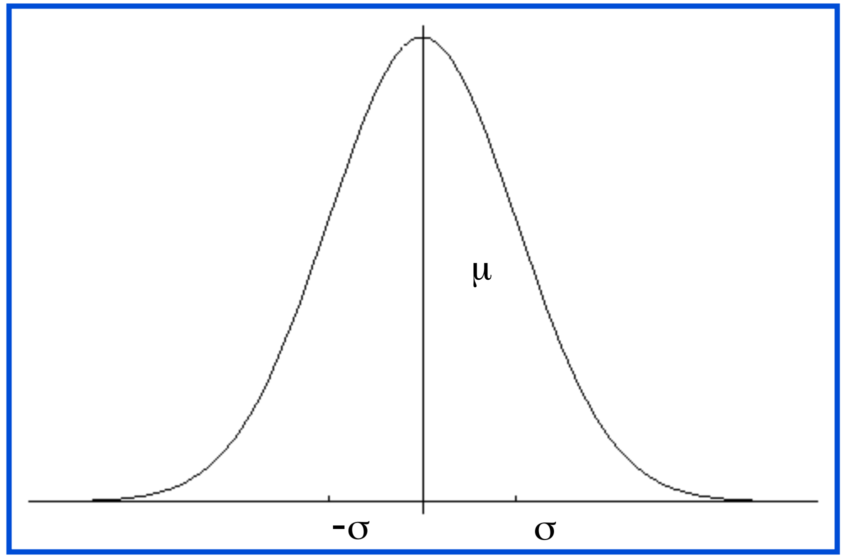
- The Kalman filter "algorithm" is a couple of **matrix multiplications**!

Gaussians

$$p(x) \sim N(\mu, \sigma^2):$$

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

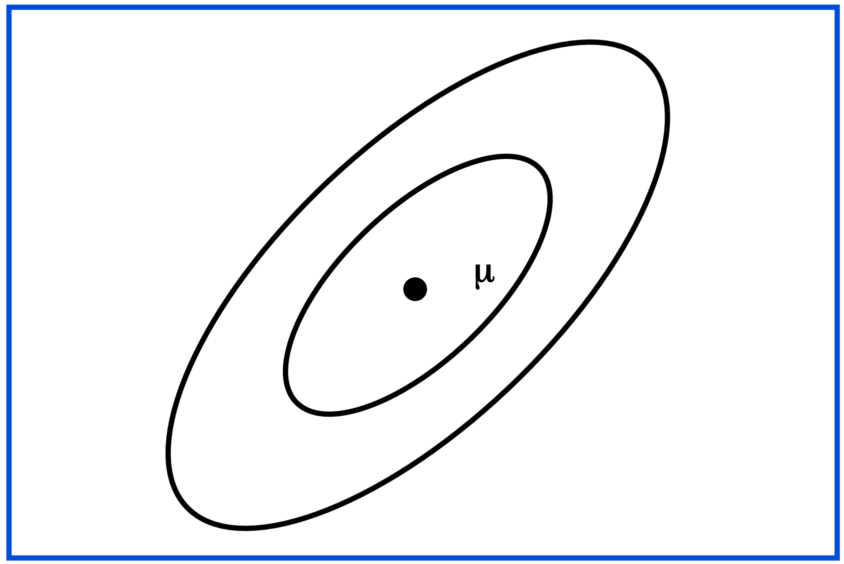
Univariate



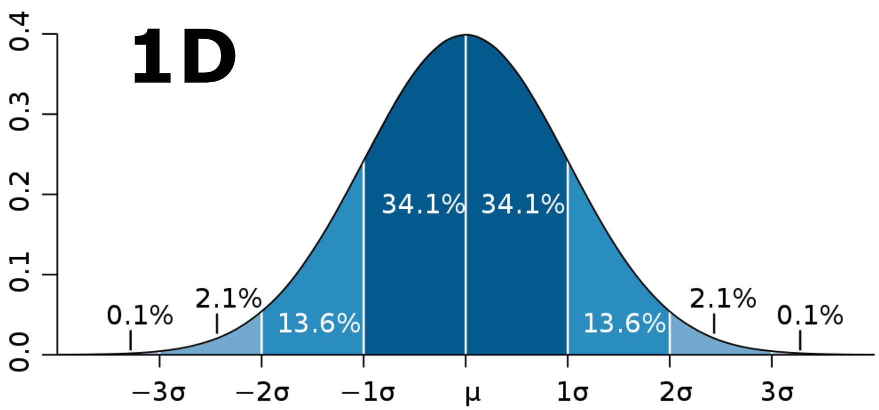
$$p(\mathbf{x}) \sim N(\boldsymbol{\mu}, \boldsymbol{\Sigma}):$$

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^t \boldsymbol{\Sigma}^{-1} (\mathbf{x}-\boldsymbol{\mu})}$$

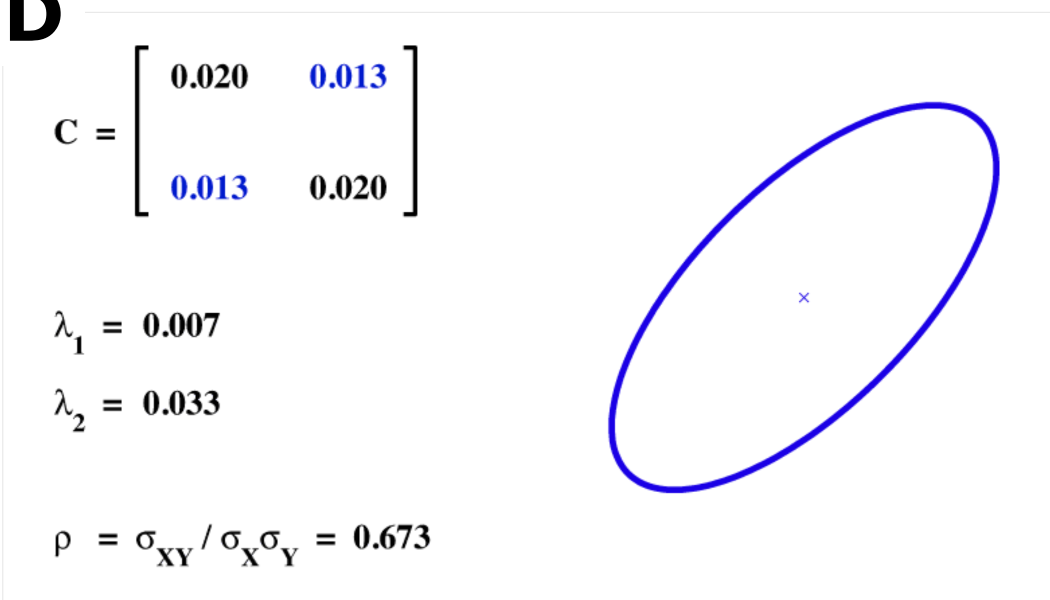
Multivariate



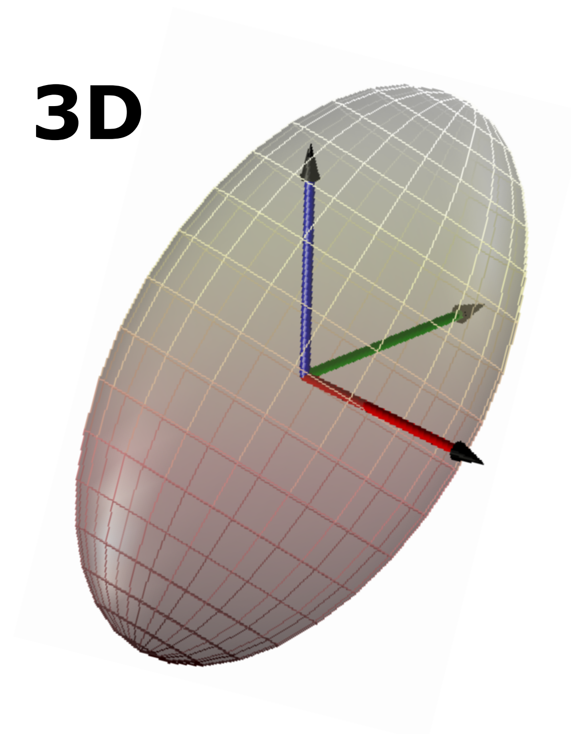
Gaussians



2D



3D



Video

Properties of Gaussians

- Univariate

$$\left.\begin{array}{l} X \sim N(\mu,\sigma^2) \\ Y = aX + b \end{array}\right\} \Rightarrow \quad Y \sim N(a\mu + b, a^2\sigma^2)$$

$$\left.\begin{array}{l} X_1 \sim N(\mu_1,\sigma_1^2) \\ X_2 \sim N(\mu_2,\sigma_2^2) \end{array}\right\} \Rightarrow p(X_1)\cdot p(X_2) \sim N\left(\frac{\sigma_2^2}{\sigma_1^2+\sigma_2^2}\mu_1 + \frac{\sigma_1^2}{\sigma_1^2+\sigma_2^2}\mu_2, \quad \frac{1}{\sigma_1^{-2}+\sigma_2^{-2}}\right)$$

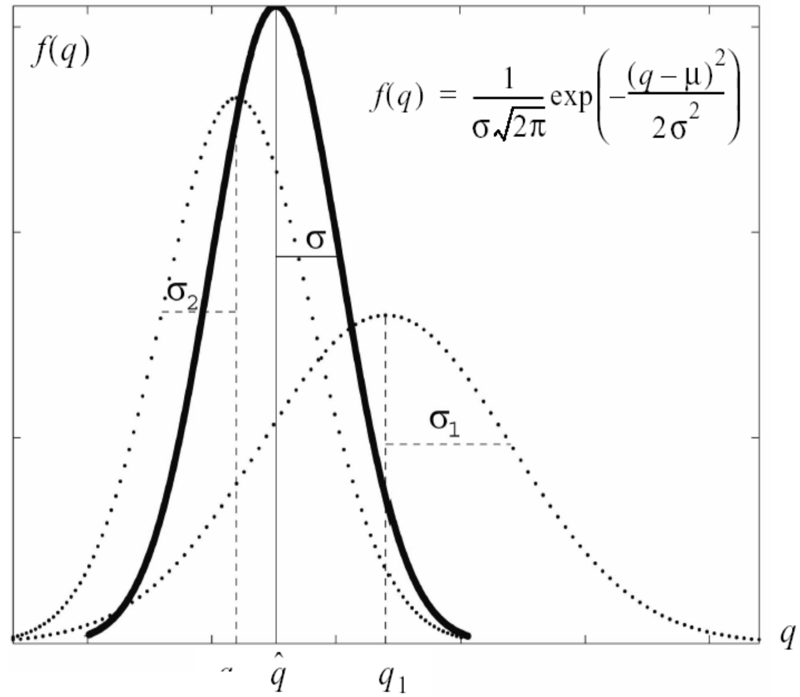
- Multivariate

$$\left.\begin{array}{l} X \sim N(\mu,\Sigma) \\ Y = AX + B \end{array}\right\} \Rightarrow \quad Y \sim N(A\mu + B, A\Sigma A^T)$$

$$\left.\begin{array}{l} X_1 \sim N(\mu_1,\Sigma_1) \\ X_2 \sim N(\mu_2,\Sigma_2) \end{array}\right\} \Rightarrow p(X_1)\cdot p(X_2) \sim N\left(\frac{\Sigma_2}{\Sigma_1+\Sigma_2}\mu_1 + \frac{\Sigma_1}{\Sigma_1+\Sigma_2}\mu_2, \quad \frac{1}{\Sigma_1^{-1}+\Sigma_2^{-1}}\right)$$

- We stay in the "Gaussian world" as long as we start with Gaussians and perform only linear transformations

Introduction to Kalman Filter (1)



- Two measurements no dynamics

$$\hat{q}_1 = q_1 \text{ with variance } \sigma_1^2$$
$$\hat{q}_2 = q_2 \text{ with variance } \sigma_2^2$$

- Weighted least-square

$$S = \sum_{i=1}^{n} w_i (\hat{q} - q_i)^2$$

- Finding minimum error

$$\frac{\partial S}{\partial \hat{q}} = \frac{\partial}{\partial \hat{q}} \sum_{i=1}^{n} w_i (\hat{q} - q_i)^2 = 2 \sum_{i=1}^{n} w_i (\hat{q} - q_i) = 0$$

- After some calculation and rearrangements

$$\hat{q} = q_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}(q_2 - q_1)$$

- Another way to look at it – weigthed mean

Discrete Kalman Filter

- Estimates the state x of a discrete-time controlled process that is governed by the linear stochastic difference equation

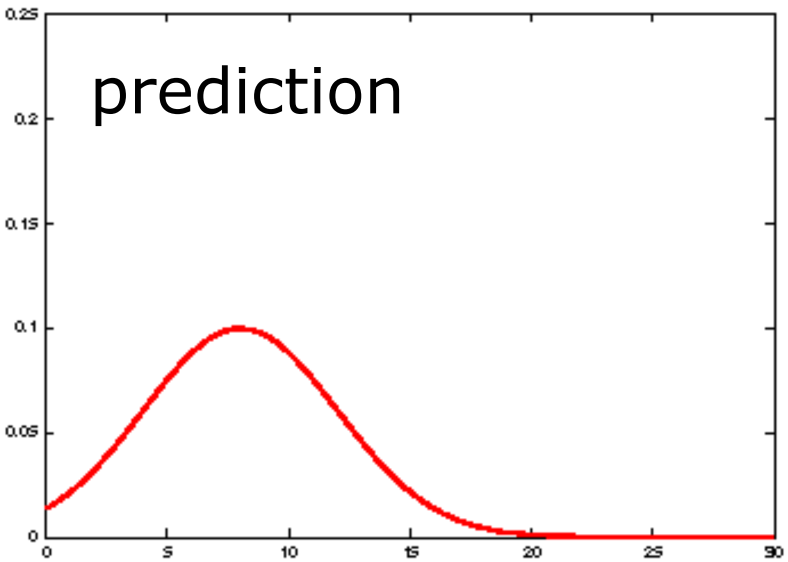
$$\boxed{x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t}$$

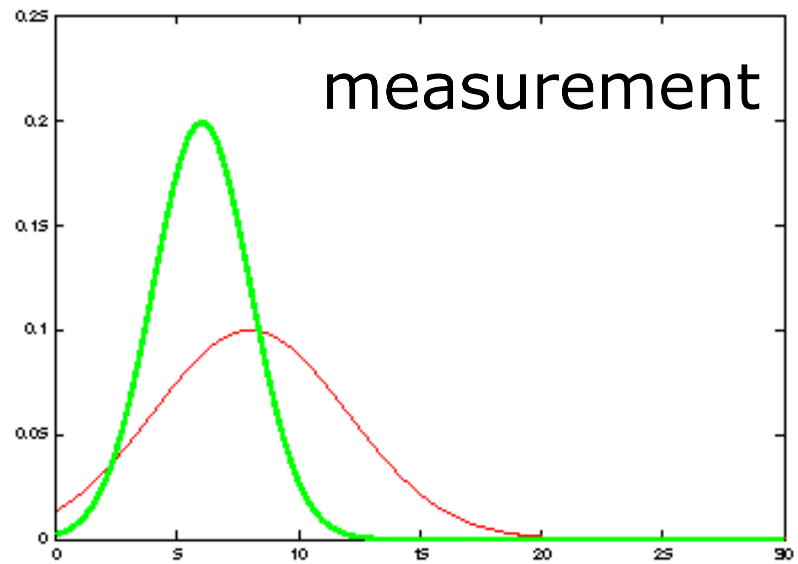
- with a measurement

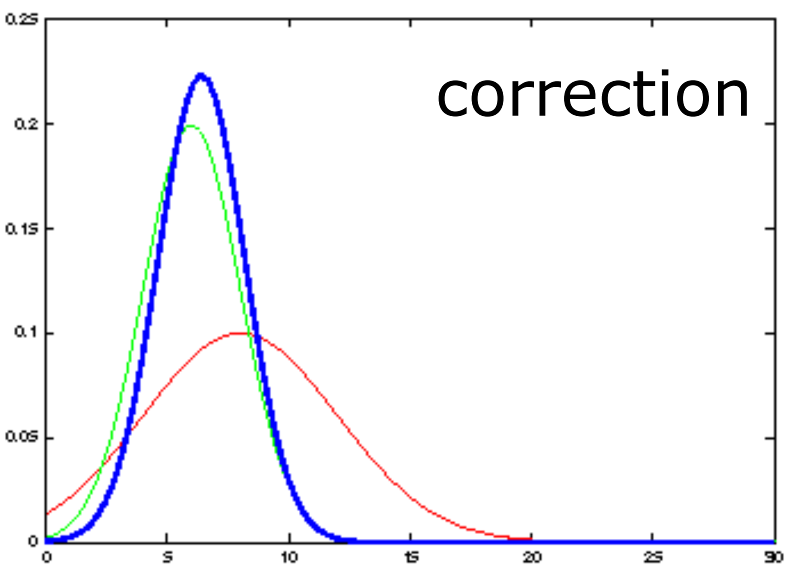
$$\boxed{z_t = C_t x_t + \delta_t}$$

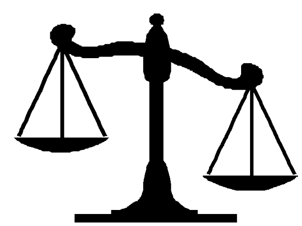
$\boxed{A_t}$	Matrix (nxn) that describes how the state evolves from t to $t\text{-}1$ without controls or noise.
$\boxed{B_t}$	Matrix (nxl) that describes how the control u_t changes the state from t to $t\text{-}1$.
$\boxed{C_t}$	Matrix (kxn) that describes how to map the state x_t to an observation z_t.
$\boxed{\varepsilon_t}$ $\boxed{\delta_t}$	Random variables representing the process and measurement noise that are assumed to be independent and normally distributed with covariance R_t and Q_t respectively.

Kalman Filter Updates in 1D







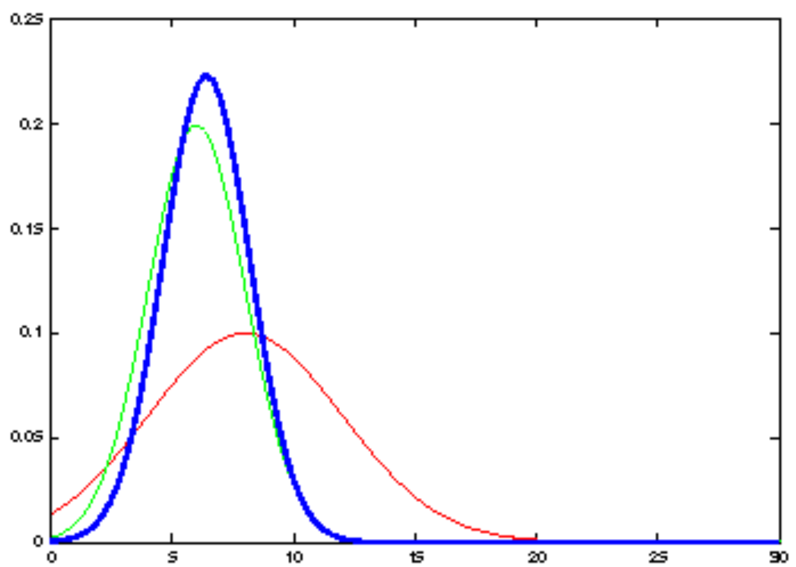


It's a weighted mean!

Kalman Filter Updates in 1D

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases} \quad \text{with} \quad K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

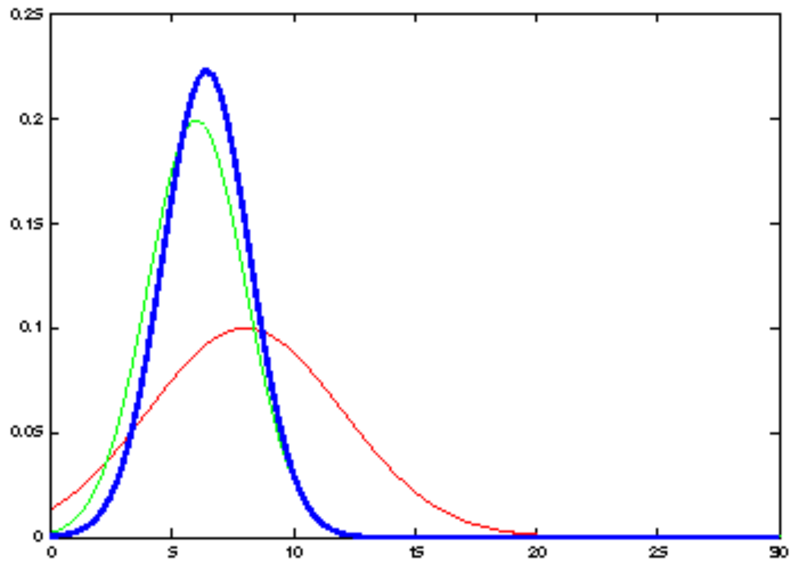
$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \end{cases} \quad \text{with} \quad K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

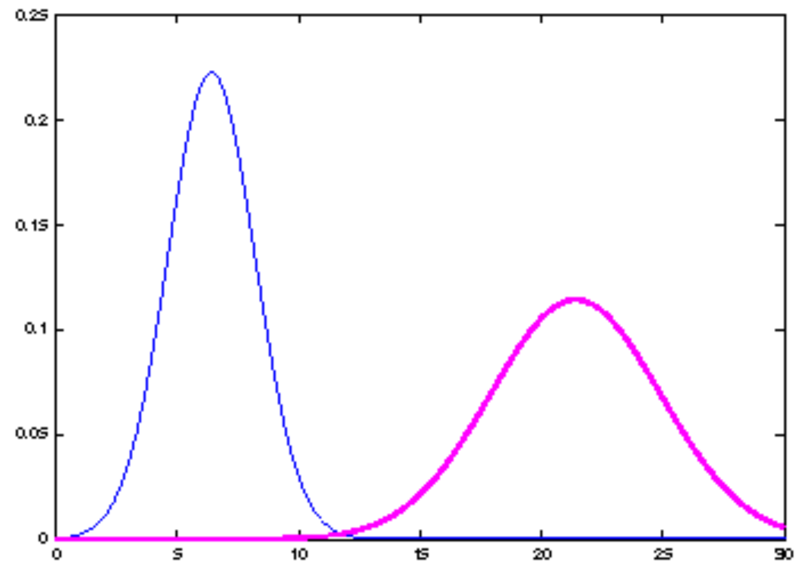


Kalman Filter Updates in 1D

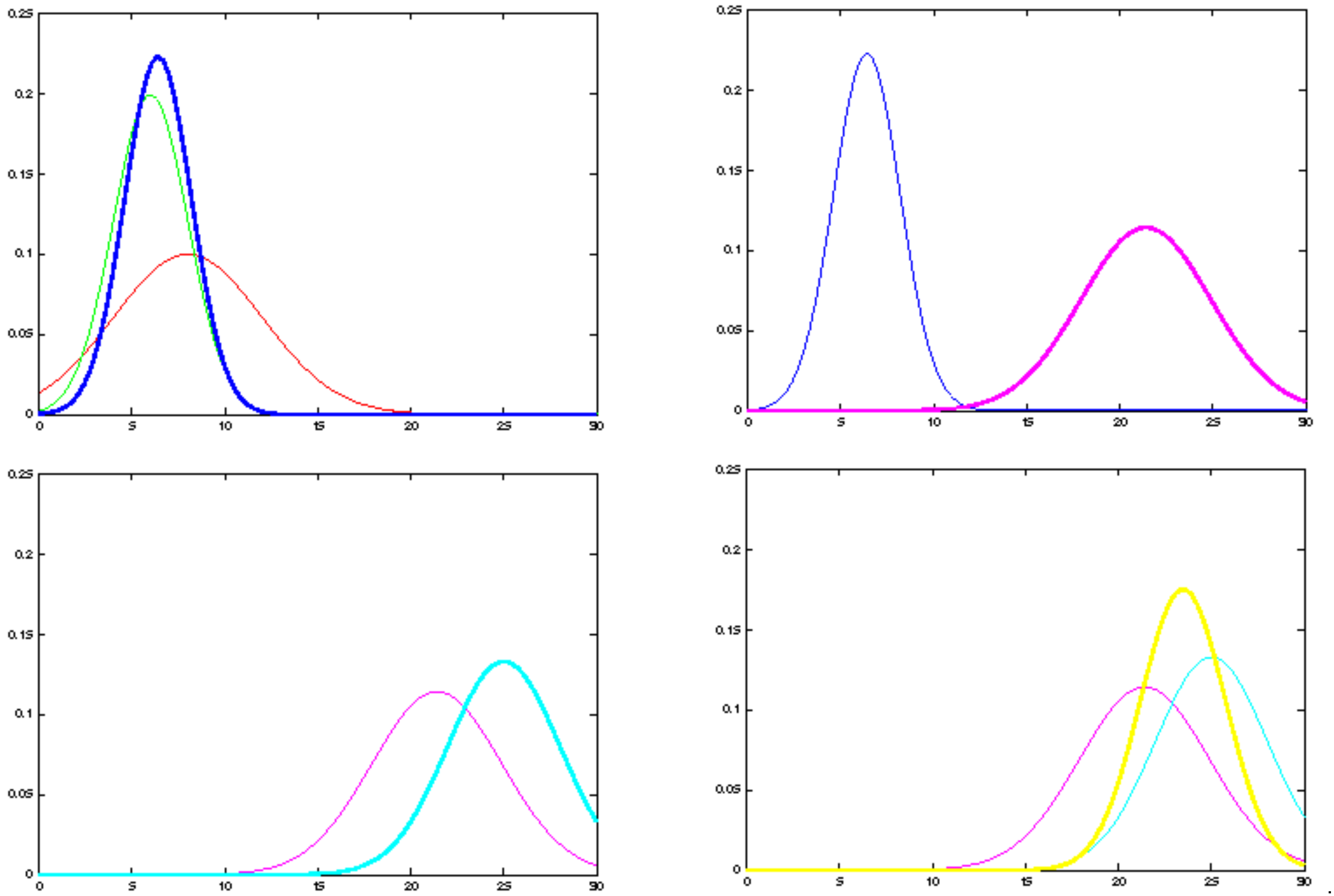
$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 \end{cases}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$





Kalman Filter Updates



Linear Gaussian Systems: Initialization

- Initial belief is normally distributed:

$$\boxed{bel(x_0) = N\left(x_0; \mu_0, \Sigma_0\right)}$$

Linear Gaussian Systems: Dynamics

- Dynamics are linear function of state and control plus additive noise:

$$\boxed{x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t}$$

$$\boxed{p(x_t \mid u_t, x_{t-1}) = N\left(x_t; A_t x_{t-1} + B_t u_t, R_t\right)}$$

$$\boxed{\begin{array}{cc} \overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) & bel(x_{t-1})\, dx_{t-1} \\ \Downarrow & \Downarrow \\ \sim N\left(x_t; A_t x_{t-1} + B_t u_t, R_t\right) & \sim N\left(x_{t-1}; \mu_{t-1}, \Sigma_{t-1}\right) \end{array}}$$

Linear Gaussian Systems: Dynamics

$$\overline{bel}(x_t) = \int p(x_t \mid u_t, x_{t-1}) \qquad\qquad bel(x_{t-1})\, dx_{t-1}$$

$$\Downarrow \qquad\qquad\qquad\qquad \Downarrow$$

$$\sim N\left(x_t; A_t x_{t-1} + B_t u_t, R_t\right) \quad \sim N\left(x_{t-1}; \mu_{t-1}, \Sigma_{t-1}\right)$$

$$\Downarrow$$

$$\overline{bel}(x_t) = \eta \int \exp\left\{-\frac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1} (x_t - A_t x_{t-1} - B_t u_t)\right\}$$

$$\exp\left\{-\frac{1}{2}(x_{t-1} - \mu_{t-1})^T \Sigma_{t-1}^{-1} (x_{t-1} - \mu_{t-1})\right\} dx_{t-1}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$

Linear Gaussian Systems: Observations

- Observations are linear function of state plus additive noise:

$$\boxed{z_t = C_t x_t + \delta_t}$$

$$\boxed{p(z_t \mid x_t) = N\left(z_t; C_t x_t, Q_t\right)}$$

$$\begin{array}{llll} bel(x_t) = & \eta & p(z_t \mid x_t) & \overline{bel}(x_t) \\ & & \Downarrow & \Downarrow \\ & & \sim N\left(z_t; C_t x_t, Q_t\right) & \sim N\left(x_t; \bar{\mu}_t, \bar{\Sigma}_t\right) \end{array}$$

Linear Gaussian Systems: Observations

$$bel(x_t) = \eta \quad p(z_t \mid x_t) \qquad \overline{bel}(x_t)$$

$$\Downarrow \qquad\qquad \Downarrow$$

$$\sim N\left(z_t; C_t x_t, Q_t\right) \qquad \sim N\left(x_t; \bar{\mu}_t, \bar{\Sigma}_t\right)$$

$$\Downarrow$$

$$bel(x_t) = \eta \exp\left\{-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C_t x_t)\right\} \exp\left\{-\frac{1}{2}(x_t - \bar{\mu}_t)^T \bar{\Sigma}_t^{-1}(x_t - \bar{\mu}_t)\right\}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \end{cases} \qquad \text{with} \quad K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

Kalman Filter Algorithm

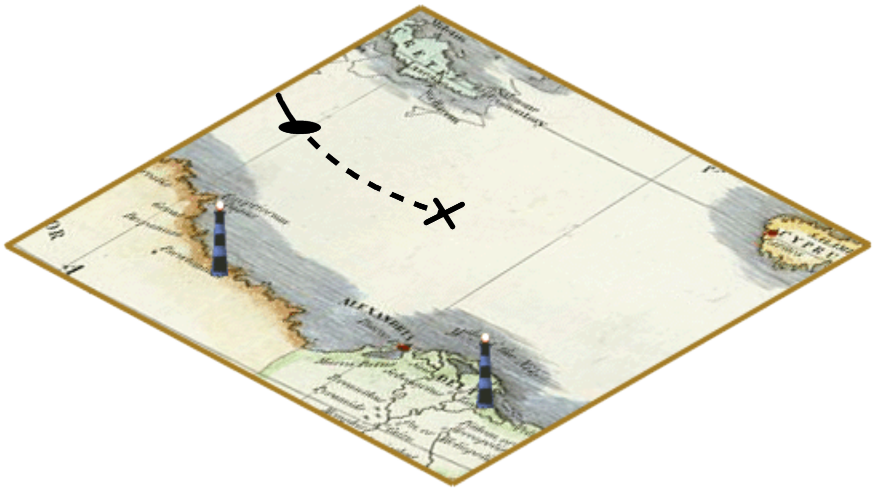
1. Algorithm **Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$):

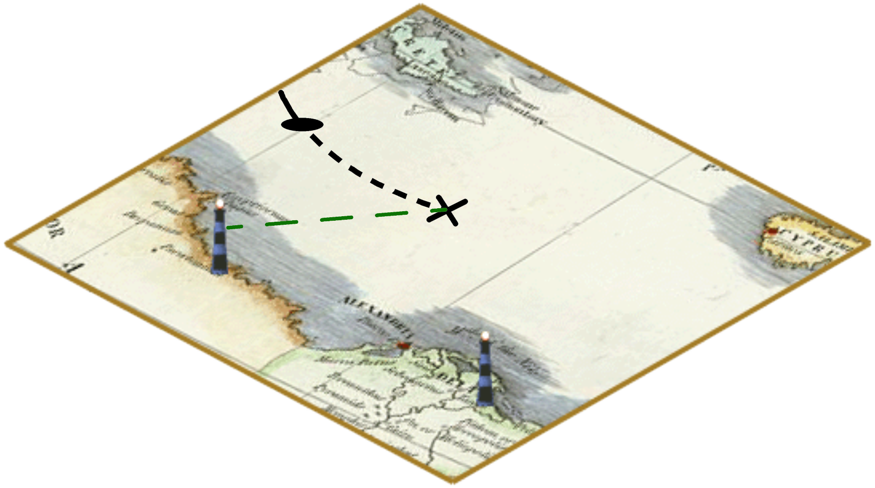
2. Prediction:
3. $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$
4. $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

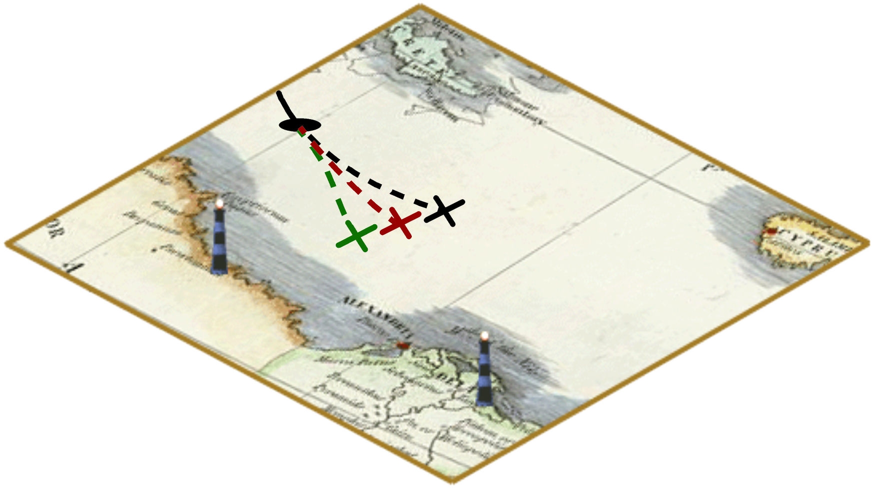
5. Correction:
6. $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$
7. $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$
8. $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9. Return μ_t, Σ_t

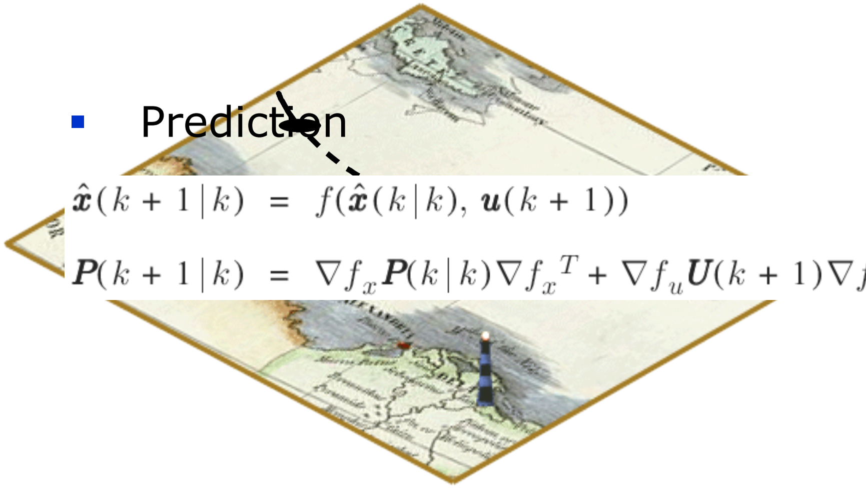
Kalman Filter Algorithm







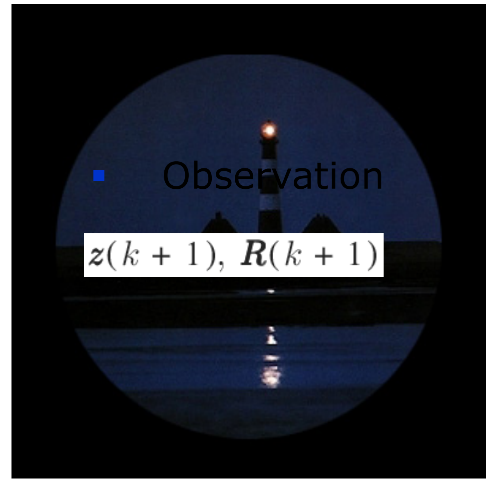
Kalman Filter Algorithm



- Prediction

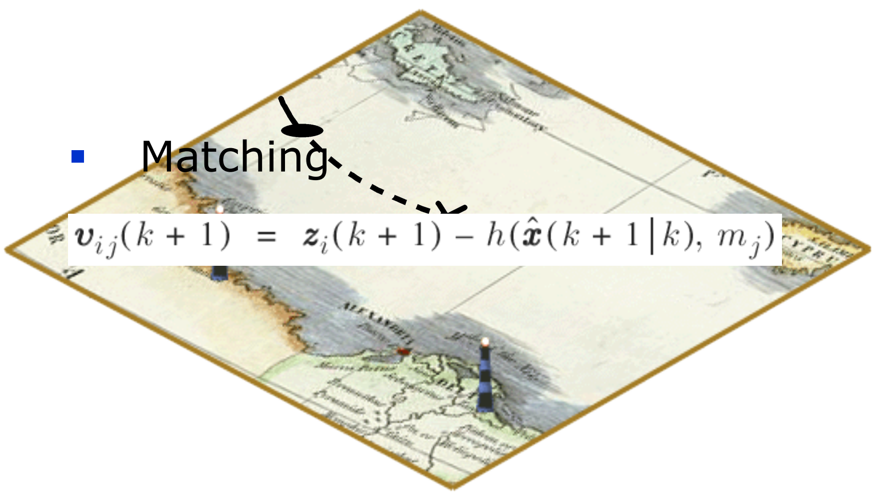
$$\hat{\boldsymbol{x}}(k+1|k) = f(\hat{\boldsymbol{x}}(k|k), \boldsymbol{u}(k+1))$$

$$\boldsymbol{P}(k+1|k) = \nabla f_x \boldsymbol{P}(k|k) \nabla f_x{}^T + \nabla f_u \boldsymbol{U}(k+1) \nabla f_u{}^T$$



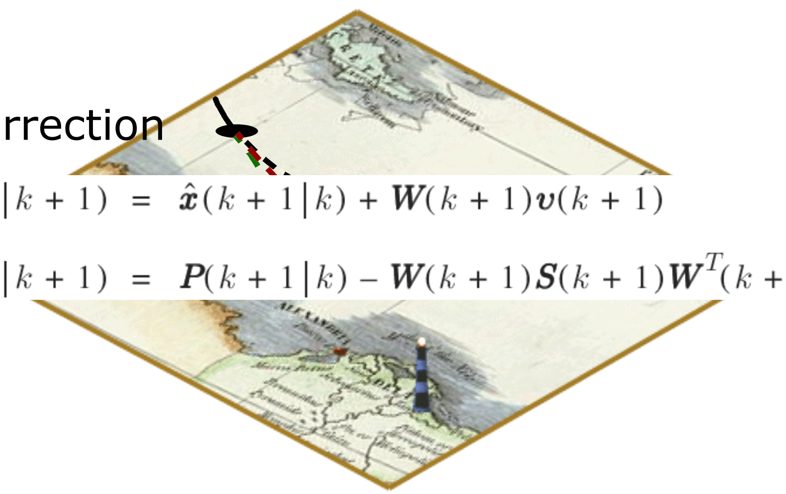
- Observation

$$\boldsymbol{z}(k+1),\ \boldsymbol{R}(k+1)$$



- Matching

$$\boldsymbol{v}_{ij}(k+1) = \boldsymbol{z}_i(k+1) - h(\hat{\boldsymbol{x}}(k+1|k), m_j)$$



- Correction

$$\hat{\boldsymbol{x}}(k+1|k+1) = \hat{\boldsymbol{x}}(k+1|k) + \boldsymbol{W}(k+1)\boldsymbol{v}(k+1)$$

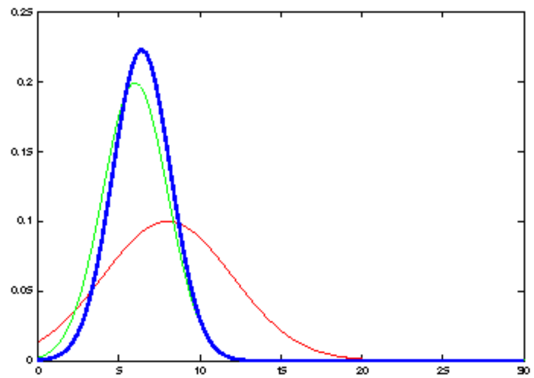
$$\boldsymbol{P}(k+1|k+1) = \boldsymbol{P}(k+1|k) - \boldsymbol{W}(k+1)\boldsymbol{S}(k+1)\boldsymbol{W}^T(k+1)$$

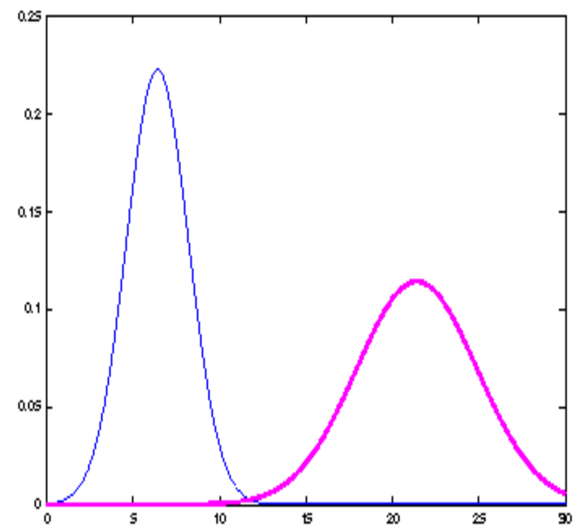
The Prediction-Correction-Cycle

Prediction

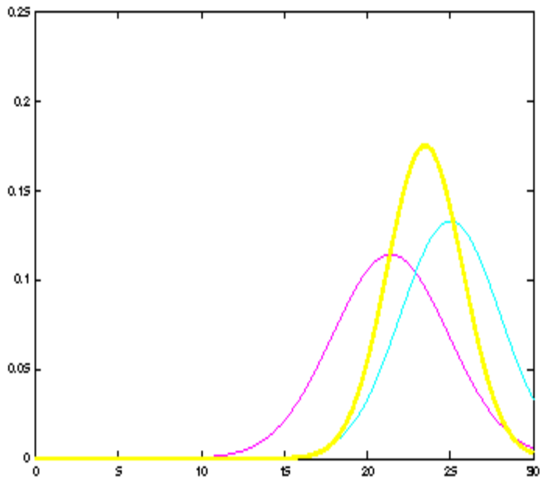
$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 \end{cases}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$



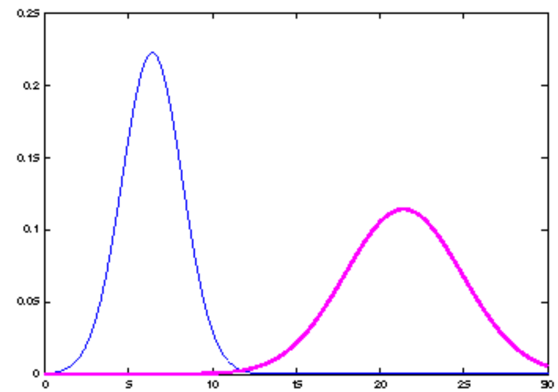


The Prediction-Correction-Cycle



$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t\bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$



The Prediction-Correction-Cycle

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = a_t \mu_{t-1} + b_t u_t \\ \bar{\sigma}_t^2 = a_t^2 \sigma_t^2 + \sigma_{act,t}^2 \end{cases}$$

$$\overline{bel}(x_t) = \begin{cases} \bar{\mu}_t = A_t \mu_{t-1} + B_t u_t \\ \bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t \end{cases}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - \bar{\mu}_t) \\ \sigma_t^2 = (1 - K_t)\bar{\sigma}_t^2 \end{cases}, K_t = \frac{\bar{\sigma}_t^2}{\bar{\sigma}_t^2 + \bar{\sigma}_{obs,t}^2}$$

$$bel(x_t) = \begin{cases} \mu_t = \bar{\mu}_t + K_t(z_t - C_t \bar{\mu}_t) \\ \Sigma_t = (I - K_t C_t)\bar{\Sigma}_t \end{cases}, K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$$

Kalman Filter Summary

- Highly efficient: Polynomial in measurement dimensionality k and state dimensionality n:

  $$O(k^{2.376} + n^2)$$

- Optimal for linear Gaussian systems!

- Most robotics systems are nonlinear!

Nonlinear Dynamic Systems

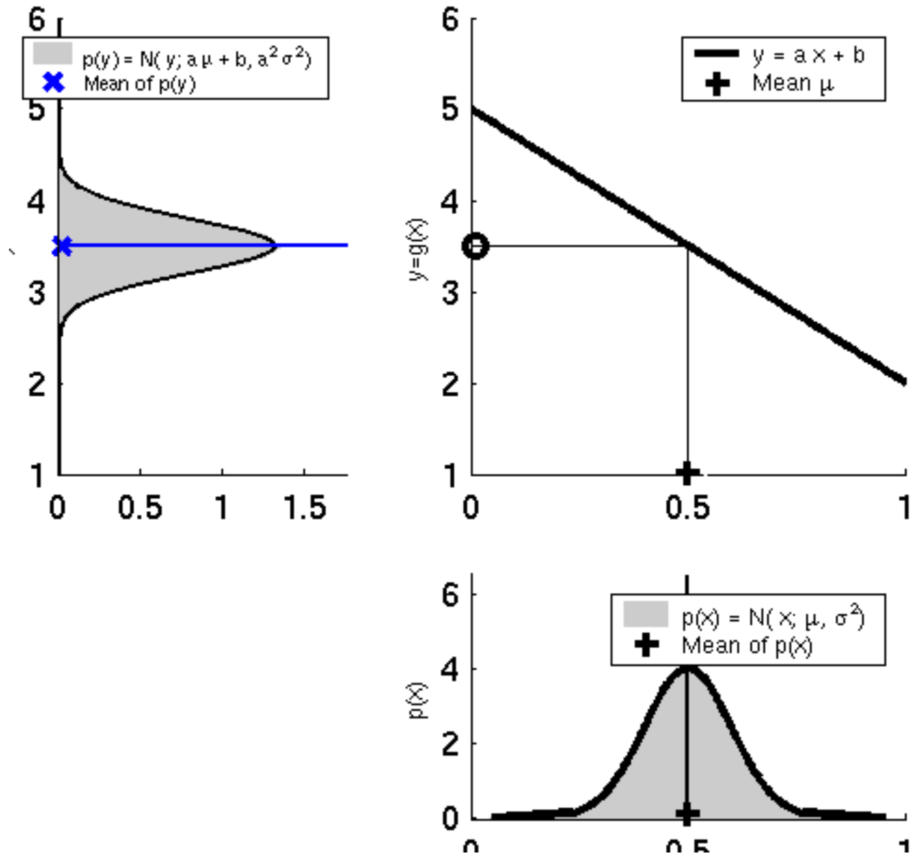
- Most realistic robotic problems involve nonlinear functions

$$\boxed{x_t = g(u_t, x_{t-1})}$$

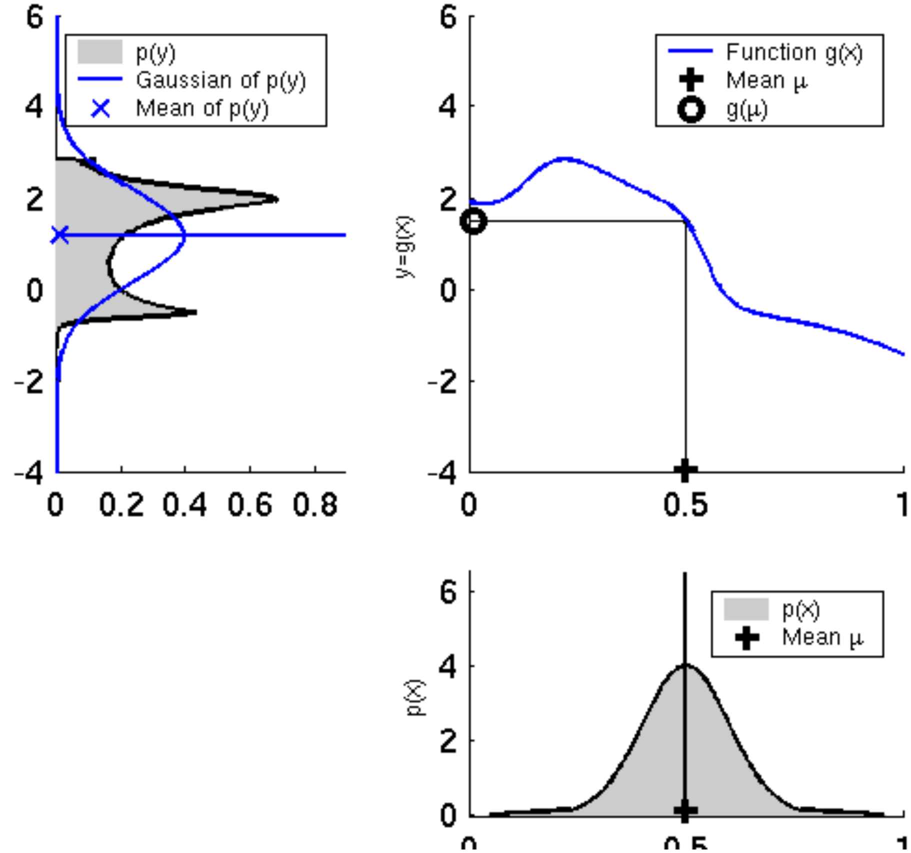
$$\boxed{z_t = h(x_t)}$$

- To be continued

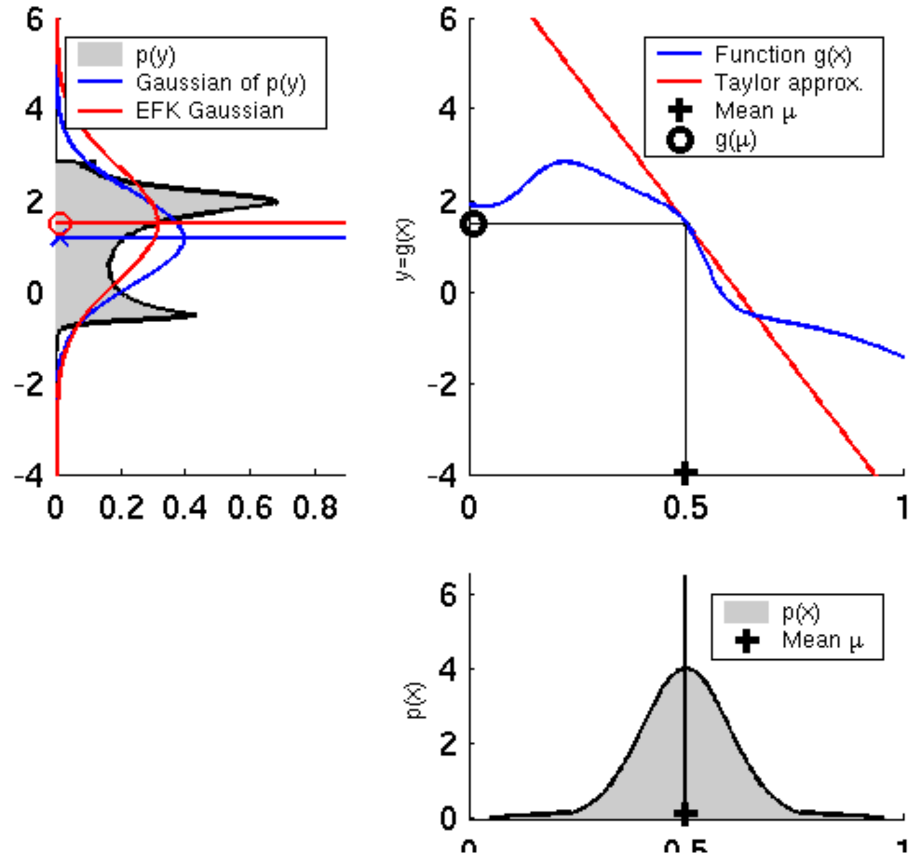
Linearity Assumption Revisited



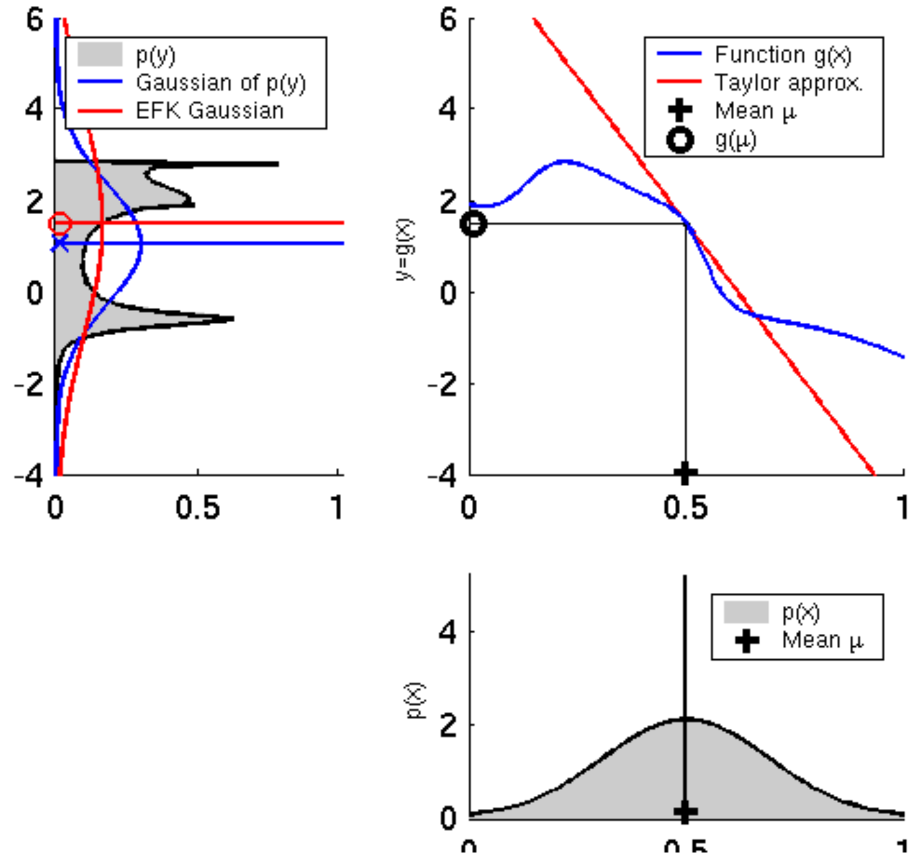
Non-linear Function



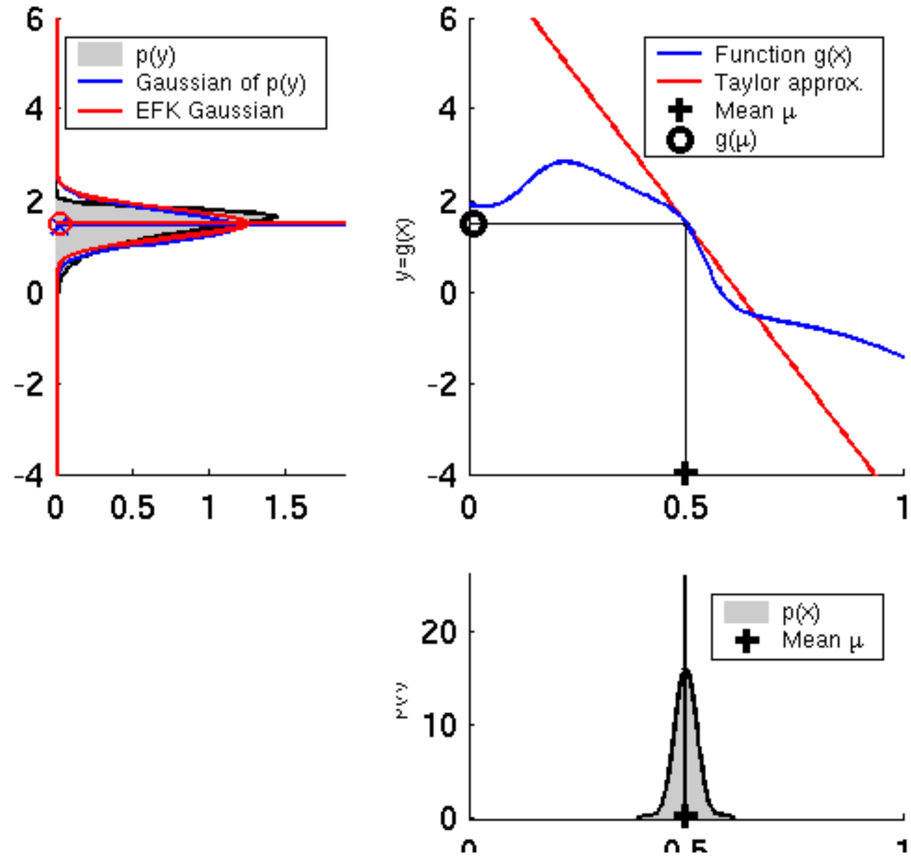
EKF Linearization (1)



EKF Linearization (2)



EKF Linearization (3)



Depends on uncertainty

EKF Linearization: First Order Taylor Series Expansion

- Prediction:

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + \frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}} (x_{t-1} - \mu_{t-1})$$

$$g(u_t, x_{t-1}) \approx g(u_t, \mu_{t-1}) + G_t \ (x_{t-1} - \mu_{t-1})$$

- Correction:

$$h(x_t) \approx h(\bar{\mu}_t) + \frac{\partial h(\bar{\mu}_t)}{\partial x_t} (x_t - \bar{\mu}_t)$$

$$h(x_t) \approx h(\bar{\mu}_t) + H_t \ (x_t - \bar{\mu}_t)$$

EKF Algorithm

1. **Extended_Kalman_filter**(μ_{t-1}, Σ_{t-1}, u_t, z_t):

2. Prediction:

3. $\bar{\mu}_t = g(u_t, \mu_{t-1})$ $\longleftarrow$ $\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t$

4. $\bar{\Sigma}_t = G_t \Sigma_{t-1} G_t^T + R_t$ $\longleftarrow$ $\bar{\Sigma}_t = A_t \Sigma_{t-1} A_t^T + R_t$

5. Correction:

6. $K_t = \bar{\Sigma}_t H_t^T (H_t \bar{\Sigma}_t H_t^T + Q_t)^{-1}$ $\longleftarrow$ $K_t = \bar{\Sigma}_t C_t^T (C_t \bar{\Sigma}_t C_t^T + Q_t)^{-1}$

7. $\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$ $\longleftarrow$ $\mu_t = \bar{\mu}_t + K_t (z_t - C_t \bar{\mu}_t)$

8. $\Sigma_t = (I - K_t H_t) \bar{\Sigma}_t$ $\longleftarrow$ $\Sigma_t = (I - K_t C_t) \bar{\Sigma}_t$

9. Return μ_t, Σ_t

$$H_t = \frac{\partial h(\bar{\mu}_t)}{\partial x_t} \qquad G_t = \frac{\partial g(u_t, \mu_{t-1})}{\partial x_{t-1}}$$

Kalman Filter for Mobile Robot Localization

Robot Position Prediction

- In a first step, the robots position at time step k+1 is predicted based on its old location (time step k) and its movement due to the control input u(k):

$$\hat{p}(k+1|k) = f(\hat{p}(k|k), u(k))$$

f: Odometry function

$$\Sigma_p(k+1|k) = \nabla_p f \cdot \Sigma_p(k|k) \cdot \nabla_p f^T + \nabla_u f \cdot \Sigma_u(k) \cdot \nabla_u f^T$$

Kalman Filter for Mobile Robot Localization

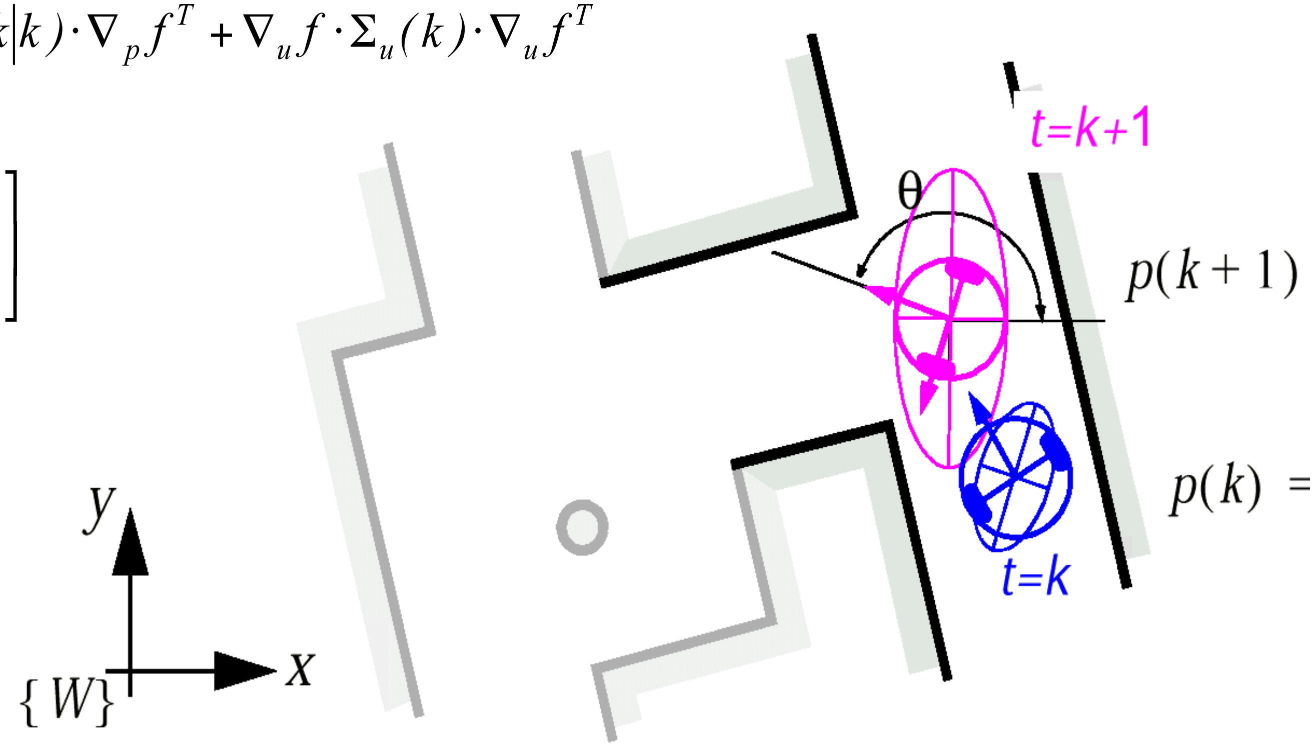
Robot Position Prediction: *Example*

$$\hat{p}(k+1|k) = \hat{p}(k|k) + u(k) = \begin{bmatrix} \hat{x}(k) \\ \hat{y}(k) \\ \hat{\theta}(k) \end{bmatrix} + \begin{bmatrix} \frac{\Delta s_r + \Delta s_l}{2} \cos(\theta + \frac{\Delta s_r - \Delta s_l}{2b}) \\ \frac{\Delta s_r + \Delta s_l}{2} \sin(\theta + \frac{\Delta s_r - \Delta s_l}{2b}) \\ \frac{\Delta s_r - \Delta s_l}{b} \end{bmatrix}$$

Odometry

$$\Sigma_p(k+1|k) = \nabla_p f \cdot \Sigma_p(k|k) \cdot \nabla_p f^T + \nabla_u f \cdot \Sigma_u(k) \cdot \nabla_u f^T$$

$$\Sigma_u(k) = \begin{bmatrix} k_r|\Delta s_r| & 0 \\ 0 & k_l|\Delta s_l| \end{bmatrix}$$



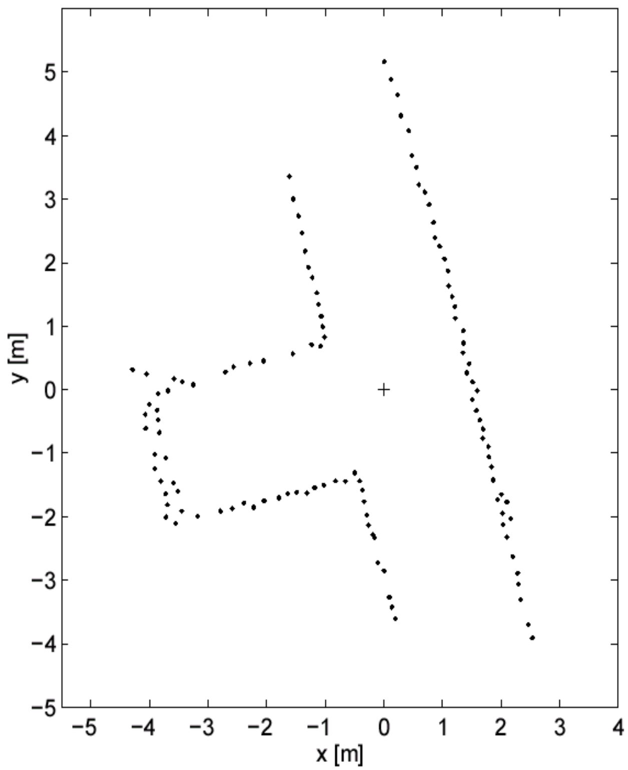
Observation

- The second step it to obtain the observation $Z(k+1)$ (measurements) from the robot's sensors at the new location at time $k+1$
- The observation usually consists of a set n_0 of single observations $z_j(k+1)$ extracted from the different sensors signals. It can represent *raw data scans* as well as *features* like *lines*, *doors* or *any kind of landmarks*.
- The parameters of the targets are usually observed in the sensor frame {S}.
  - Therefore the observations have to be transformed to the world frame {W} or
  - the measurement prediction have to be transformed to the sensor frame {S}.
  - This transformation is specified in the function h_i (seen later) – measurement prediction

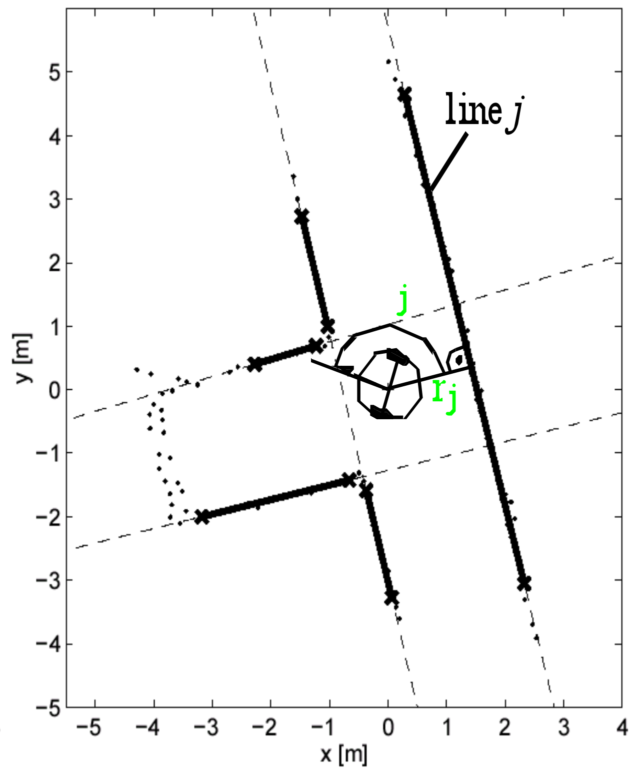
Kalman Filter for Mobile Robot Localization

Observation: *Example*

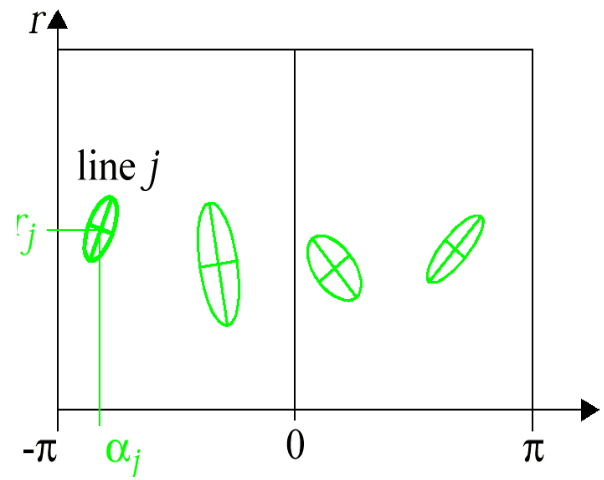
Raw Date of Laser Scanner



Extracted Lines



Extracted Lines in Model Space



$$z_j(k+1) = {}^{R}\begin{bmatrix} \alpha_j \\ r_j \end{bmatrix}$$

Sensor (robot) frame

$$\Sigma_{R,j}(k+1) = \begin{bmatrix} \sigma_{\alpha\alpha} & \sigma_{\alpha r} \\ \sigma_{r\alpha} & \sigma_{rr} \end{bmatrix}_j$$

Set of discrete line features

Measurement Prediction

- In the next step we use the predicted robot position $\hat{p} = (k+1|k)$ and the map $M(k)$ to generate multiple predicted observations z_t – what you should see
- They have to be transformed into the sensor frame

$$\hat{z}_i(k+1) = h_i\left(z_t, \hat{p}(k+1|k)\right)$$

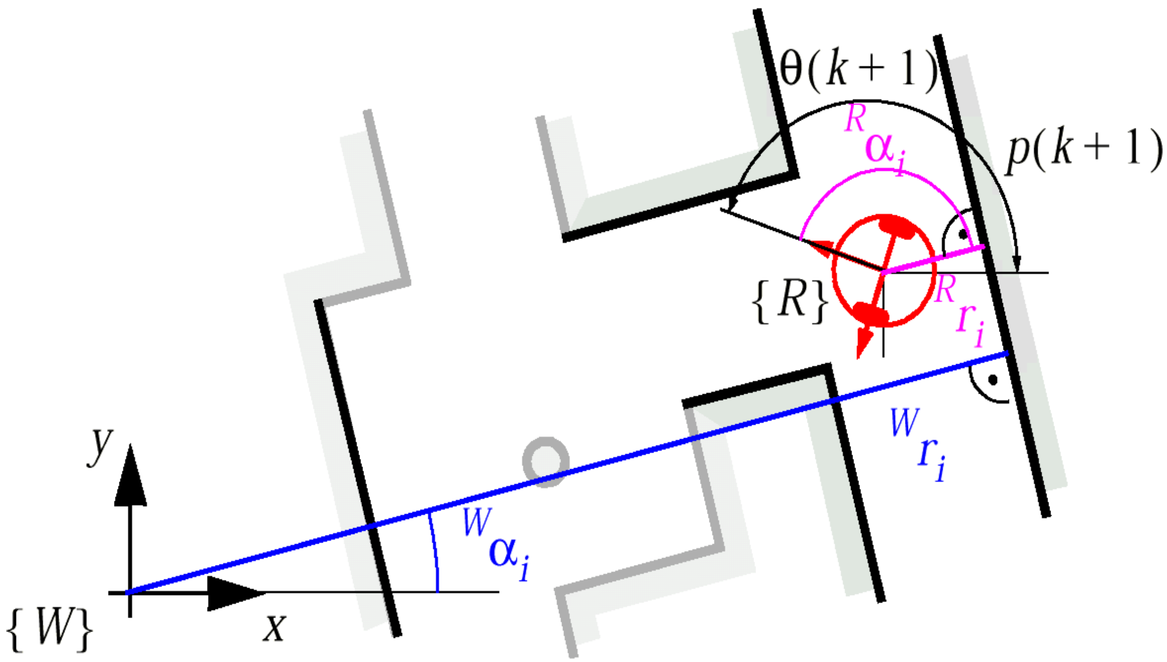
- We can now define the measurement prediction as the set containing all n_i predicted observations

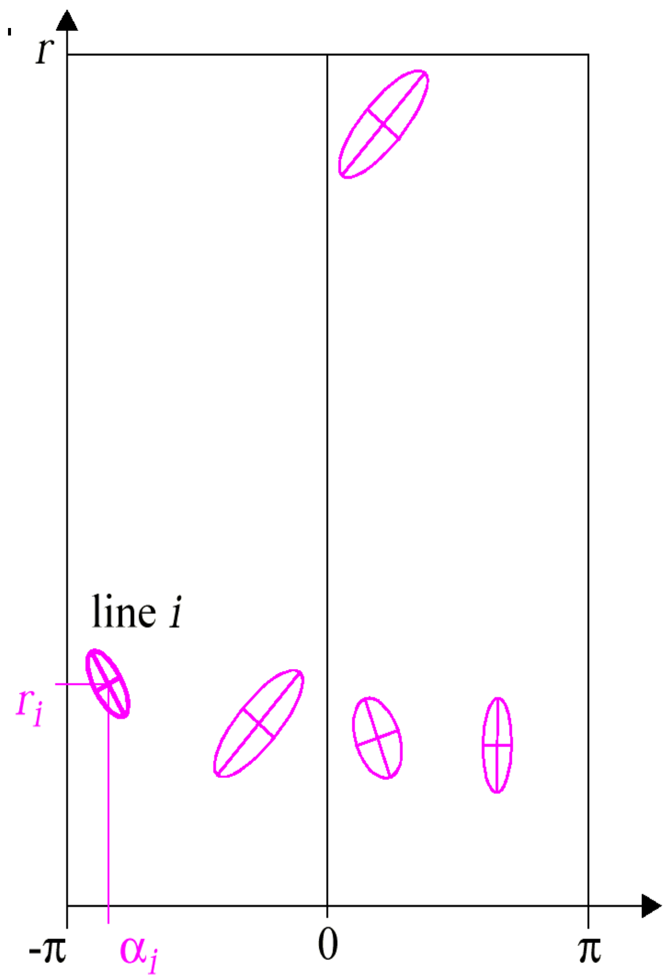
$$\hat{Z}(k+1) = \left\{\hat{z}_i(k+1) \middle| (1 \le i \le n_i)\right\}$$

- The function h_i is mainly the coordinate transformation between the world frame and the sensor frame
- Need to compute the measurement Jacobian to also predict uncertainties

Measurement Prediction: *Example*

- For prediction, only the walls that are in the field of view of the robot are selected.
- This is done by linking the individual lines to the nodes of the path





Measurement Prediction: *Example*

- The generated measurement predictions have to be transformed to the robot frame $\{R\}$

$$ {}^{W}z_{t,i} = {}^{W}\begin{bmatrix} \alpha_{t,i} \\ r_{t,i} \end{bmatrix} \rightarrow {}^{R}z_{t,i} = {}^{R}\begin{bmatrix} \alpha_{t,i} \\ r_{t,i} \end{bmatrix} $$

- According to the figure in previous slide the transformation is given by

$$ \hat{z}_i(k+1) = {}^{R}\begin{bmatrix} \alpha_{t,i} \\ r_{t,i} \end{bmatrix} = h_i(z_{t,i}, \hat{p}(k+1|k)) = \begin{bmatrix} {}^{W}\alpha_{t,i} - {}^{W}\hat{\theta}(k+1|k) \\ {}^{W}r_{t,i} - ({}^{W}\hat{x}(k+1|k)\cos({}^{W}\alpha_{t,i}) + {}^{W}\hat{y}(k+1|k)\sin({}^{W}\alpha_{t,i})) \end{bmatrix} $$

and its Jacobian by

$$ \nabla h_i = \begin{bmatrix} \dfrac{\partial \alpha_{t,i}}{\partial \hat{x}} & \dfrac{\partial \alpha_{t,i}}{\partial \hat{y}} & \dfrac{\partial \alpha_{t,i}}{\partial \hat{\theta}} \\[2ex] \dfrac{\partial r_{t,i}}{\partial \hat{x}} & \dfrac{\partial r_{t,i}}{\partial \hat{y}} & \dfrac{\partial r_{t,i}}{\partial \hat{\theta}} \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 \\ -\cos{}^{W}\alpha_{t,i} & -\sin{}^{W}\alpha_{t,i} & 0 \end{bmatrix} $$

Matching

- Assignment from observations $z_j(k+1)$ (gained by the sensors) to the targets z_t (stored in the map)
- For each measurement prediction for which an corresponding observation is found we calculate the innovation:

$$v_{ij}(k+1) = [z_j(k+1) - h_i(z_t, \hat{p}(k+1|k))]$$

$$= \begin{bmatrix} \alpha_j \\ r_j \end{bmatrix} - \begin{bmatrix} {}^W\alpha_{t,i} - {}^W\hat{\theta}(k+1|k) \\ {}^Wr_{t,i} - ({}^W\hat{x}(k+1|k)\cos({}^W\alpha_{t,i}) + {}^W\hat{y}(k+1|k)\sin({}^W\alpha_{t,i})) \end{bmatrix}$$

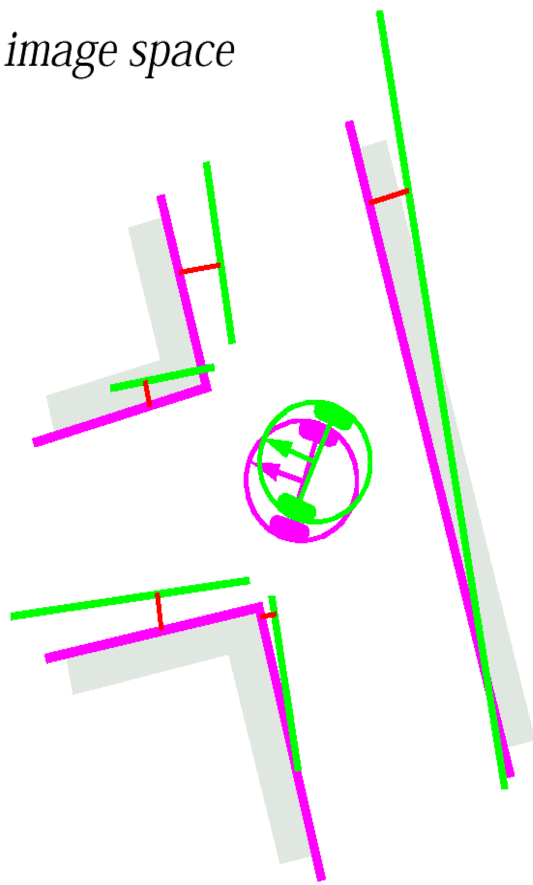
and its innovation covariance found by applying the error propagation law:

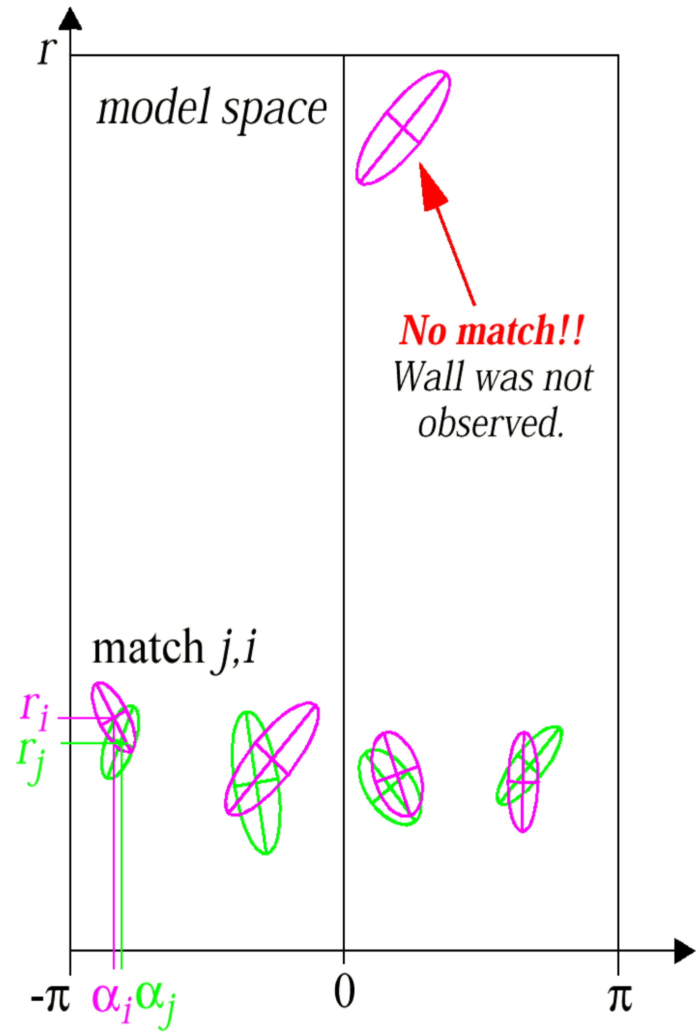
$$\Sigma_{IN,ij}(k+1) = \nabla h_i \cdot \Sigma_p(k+1|k) \cdot \nabla h_i^{\,T} + \Sigma_{R,i}(k+1)$$

- The validity of the correspondence between measurement and prediction can e.g. be evaluated through the Mahalanobis distance:

$$v_{ij}^T(k+1) \cdot \Sigma_{IN,ij}^{-1}(k+1) \cdot v_{ij}(k+1) \le g^2$$

Matching: *Example*





Matching: *Example*

- To find correspondence (pairs) of predicted and observed features we use the Mahalanobis distance

$$v_{ij}(k+1) \cdot \Sigma_{IN,\,ij}^{-1}(k+1) \cdot v_{ij}^{T}(k+1) \le g^2$$

$$v_{ij}(k+1) = [z_j(k+1) - h_i(z_t, \hat{p}(k+1|k))]$$

$$= \begin{bmatrix} \alpha_j \\ r_j \end{bmatrix} - \begin{bmatrix} {}^{W}\alpha_{t,\,i} - {}^{W}\hat{\theta}(k+1|k) \\ {}^{W}r_{t,\,i} - ({}^{W}\hat{x}(k+1|k)\cos({}^{W}\alpha_{t,\,i}) + {}^{W}\hat{y}(k+1|k)\sin({}^{W}\alpha_{t,\,i})) \end{bmatrix}$$

$$\Sigma_{IN,\,ij}(k+1) = \nabla h_i \cdot \Sigma_p(k+1|k) \cdot \nabla h_i^{\,T} + \Sigma_{R,\,i}(k+1)$$

Estimation: Applying the Kalman Filter

- Kalman filter gain:

$$K(k+1) = \Sigma_p(k+1|k) \cdot \nabla h^T \cdot \Sigma_{IN}^{-1}(k+1)$$

- Update of robot's position estimate:

$$\hat{p}(k+1|k+1) = \hat{p}(k+1|k) + K(k+1) \cdot v(k+1)$$

- The associate variance

$$\Sigma_p(k+1|k+1) = \Sigma_p(k+1|k) - K(k+1) \cdot \Sigma_{IN}(k+1) \cdot K^T(k+1)$$

Estimation: 1D Case

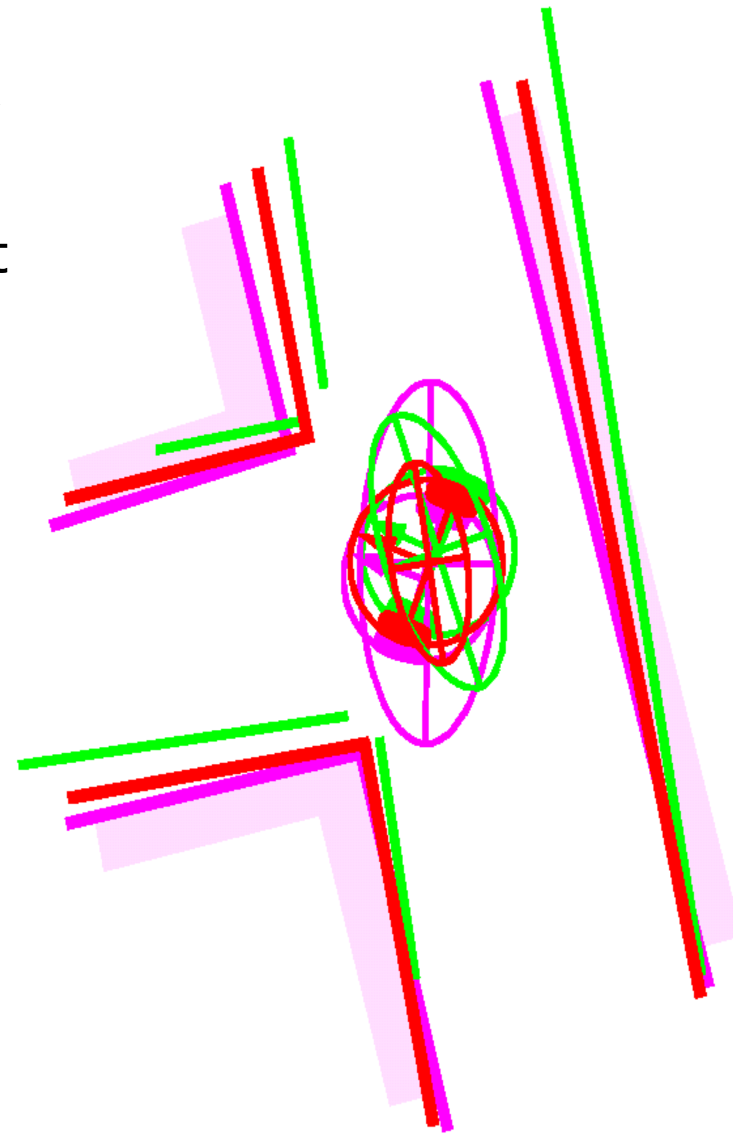
- For the one-dimensional case with $h_i(z_t, \hat{p}(k+1|k)) = z_t$ we can show that the estimation corresponds to the Kalman filter for one-dimension presented earlier.

$$K(k+1) = \frac{\sigma_p^2(k+1|k)}{\sigma_{IN}^2(k+1)} = \frac{\sigma_p^2(k+1|k)}{\sigma_p^2(k+1|k) + \sigma_R^2(k+1)}$$

$$\begin{aligned}\hat{p}(k+1|k+1) &= \hat{p}(k+1|k) + K(k+1) \cdot v(k+1) \\ &= \hat{p}(k+1|k) + K(k+1) \cdot [z_j(k+1) - h_i(z_t, \hat{p}(k+1|k))] \\ &= \hat{p}(k+1|k) + K(k+1) \cdot [z_j(k+1) - z_t]\end{aligned}$$

Estimation: *Example*

- Kalman filter estimation of the new robot position $\hat{p}(k|k)$:
  - By fusing the prediction of robot position (magenta) with the innovation gained by the measurements (green) we get the updated estimate of the robot position (red)



Localization

> "Using sensory information to locate the robot in its environment is the most fundamental problem to providing a mobile robot with autonomous capabilities." [Cox '91]

- **Given**
  - Map of the environment.
  - Sequence of sensor measurements.
- **Wanted**
  - Estimate of the robot's position.
- **Problem classes**
  - Position tracking
  - Global localization
  - Kidnapped robot problem (recovery)

Landmark-based Localization