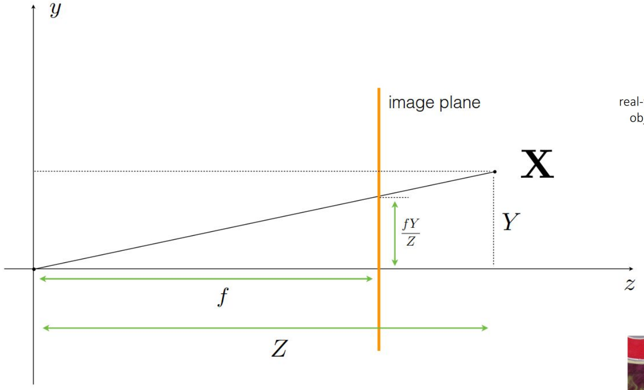
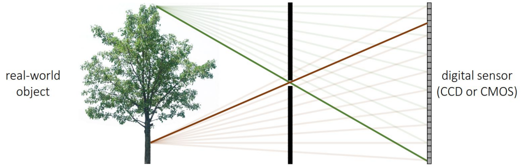
SES598-2026-Lec7

Jnaneshwar Das
2/3/2026

Image formation, camera models, feature descriptors



$$[X \quad Y \quad Z]^\top \mapsto [fX/Z \quad fY/Z]^\top$$



$$\boldsymbol{x} = \mathbf{PX}$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

homogeneous image 3 x 1 | Camera matrix 3 x 4 | homogeneous world point 4 x 1

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

2D Image Coordinates | Intrinsic properties (Optical Centre, scaling) | Extrinsic properties (Camera Rotation and translation) | 3D World Coordinates

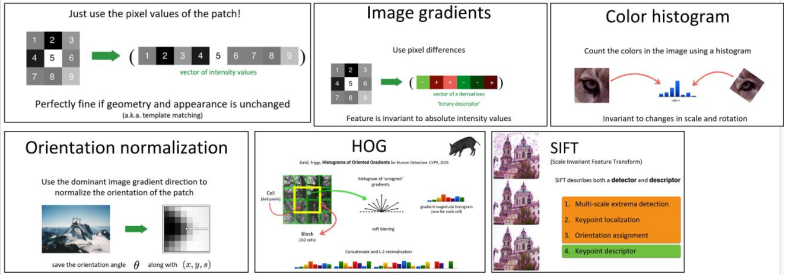
CCD camera

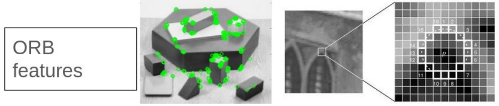
$$\mathbf{P} = \begin{bmatrix} \alpha_x & 0 & p_x \\ 0 & \alpha_y & p_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

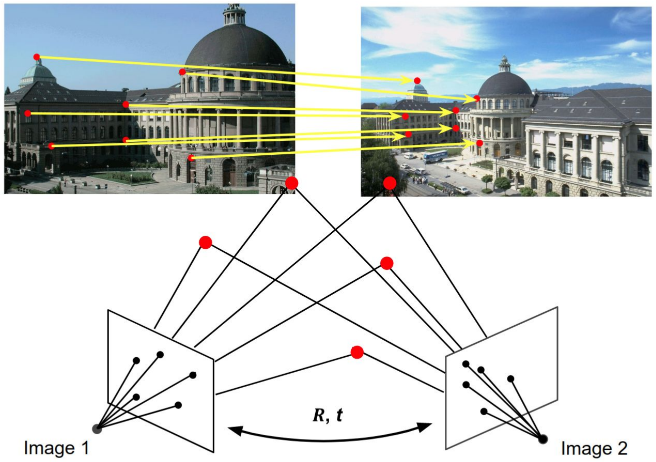
(assuming that axes are aligned)

How many degrees of freedom?

10 DOF

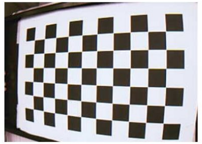






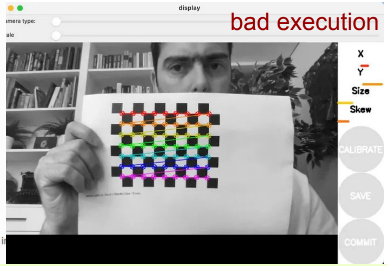
Calibrating cameras

https://docs.ros.org/en/jazzy/p/camera_calibration/



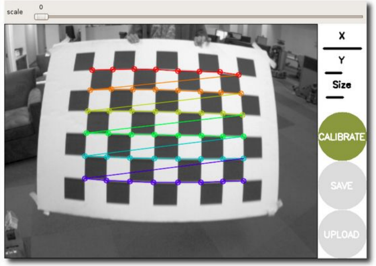
Zhang, A Flexible New Technique for Camera Calibration, MSR-TR-98-71

Single Camera



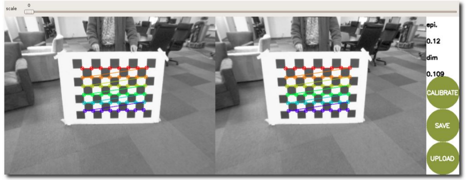
As you move the checkerboard around you will see three bars on the calibration sidebar increase i length. When the **CALIBRATE** button lights, you have enough data for calibration and can click **CALIBRATE** to see the results.

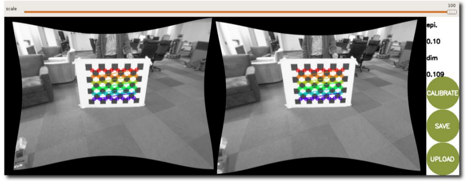
Calibration can take about a minute. The windows might be greyed out but just wait, it is worki




```
D =  [-0.33758562758914146, 0.11161239414304096, -0.00021819272592442094, -3.029195446330518e-(
K =  [430.21554970319971, 0.0, 306.6913434743704, 0.0, 430.53169252696676, 227.22480030078816,
R =  [1.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 1.0]
P =  [1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0, 0.0, 0.0, 0.0, 1.0, 0.0]
# oST version 5.0 parameters

[image]

width
640

height
480

[narrow_stereo/left]

camera matrix
430.215550 0.000000 306.691343
0.000000 430.531693 227.224800
0.000000 0.000000 1.000000

distortion
-0.337586 0.111612 -0.000218 -0.000030 0.0000

rectification
1.000000 0.000000 0.000000
0.000000 1.000000 0.000000
0.000000 0.000000 1.000000

projection
1.000000 0.000000 0.000000 0.000000
0.000000 1.000000 0.000000 0.000000
0.000000 0.000000 1.000000 0.000000
```

Stereo Camera






```
Left:
D =  [-0.31428196940330423, 0.091934927292188081, 0.0012952332628749881, -0.0016196264582460102
K =  [425.19601412158482, 0.0, 308.29689772295882, 0.0, 426.02782697756809, 223.53789948448097,
R =  [0.99783514664821127, 0.013543024537547797, 0.064355315242164368, -0.013644479706482148, 0
P =  [308.02438413775218, 0.0, 270.8484992980957, 0.0, 0.0, 308.02438413775218, 226.91282844543

Right:
D =  [-0.30558530964404188, 0.082701072309439266, 0.0019667520195065093, -0.0016409796174513081]
K =  [425.18741896821433, 0.0, 323.8787136811938, 0.0, 426.46867413824884, 226.19589833365407,
R =  [0.99769116642055589, 0.011418153515469445, 0.066947458627553177, -0.011312576360776634, 0
P =  [308.02438413775218, 0.0, 270.8484992980957, -28.144333365644293, 0.0, 308.02438413775218,

# oST version 5.0 parameters

[image]

width
640

height
480

[narrow_stereo/left]

camera matrix
425.196014 0.000000 308.296898
0.000000 426.027827 223.537899
0.000000 0.000000 1.000000

distortion
-0.314282 0.091935 0.001295 -0.001620 0.0000

rectification
0.997835 0.013543 0.064355
-0.013644 0.999906 0.001137
-0.064334 -0.002013 0.997926

projection
308.024384 0.000000 270.848499 0.000000
0.000000 308.024384 226.912828 0.000000
0.000000 0.000000 1.000000 0.000000

# oST version 5.0 parameters

[image]

width
640

height
480

[narrow_stereo/right]

camera matrix
425.187419 0.000000 323.878714
0.000000 426.468674 226.195898
0.000000 0.000000 1.000000

distortion
-0.305585 0.082701 0.001967 -0.001641 0.0000

rectification
0.997691 0.011418 0.066947
-0.011313 0.999934 -0.001956
-0.066965 0.001194 0.997755

projection
308.024384 0.000000 270.848499 -28.144333
0.000000 308.024384 226.912828 0.000000
0.000000 0.000000 1.000000 0.000000
```

Controllability and Observability summary

$$Q_c = [B, AB, A^2B, \ldots, A^{n-1}B]$$

controllability matrix

If controllability matrix is full rank, then, it is possible to steer dynamical system from an arbitrary initial state to an arbitrary final state using the set of admissible controls.

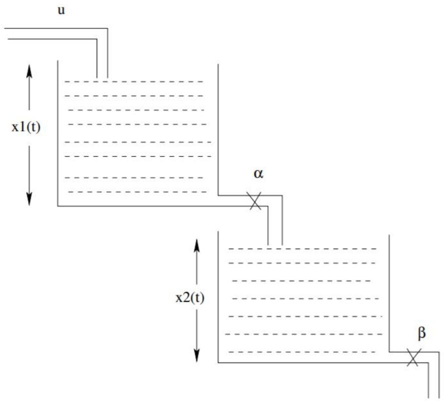
$$Q_o = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix}$$

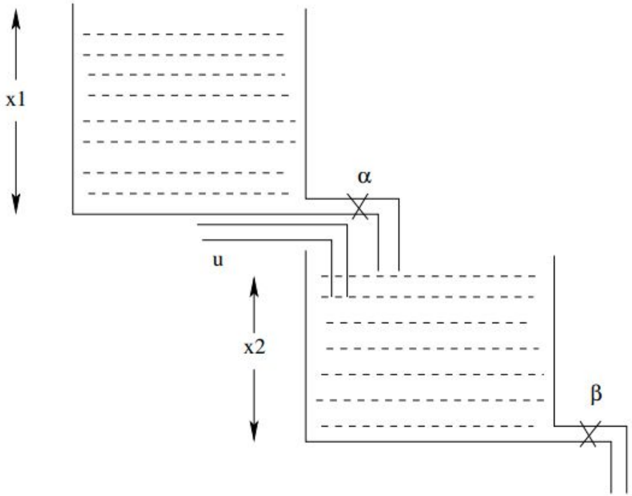
observability matrix

If observability matrix is full rank, system is observable, current values of all of its state variables can be determined through output sensors.

Can both tanks be controlled for each model?
Are tank water levels fully controllable for both models?

Example: Tank Problem :





Let $x_1(t)$ be the water level in Tank 1 and $x_2(t)$ be the water level in Tank 2. Let α be the rate of outflow from Tank 1 and β be rate of outflow from Tank 2. Let u be the supply of water to the system. The system can be modelled into the following differential equations:

$$\frac{dx_1}{dt} = -\alpha x_1 + u$$

$$\frac{dx_2}{dt} = \alpha x_1 - \beta x_2$$

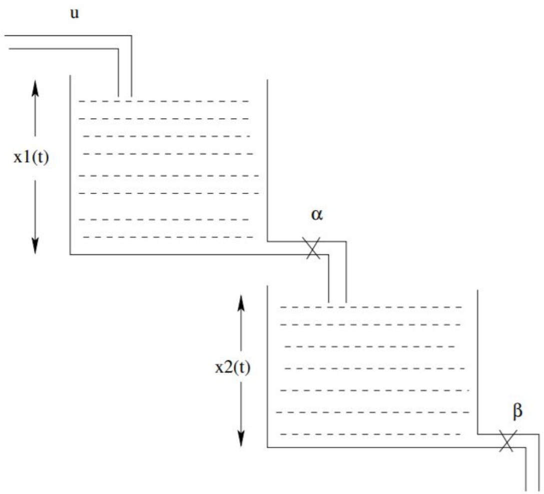
$$\frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -\alpha & 0 \\ \alpha & -\beta \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u$$

$$\frac{dx_1}{dt} = -\alpha x_1$$

$$\frac{dx_2}{dt} = \alpha x_1 - \beta x_2 + u$$

$$\frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -\alpha & 0 \\ \alpha & -\beta \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} u$$

Example: Tank Problem :



$$\frac{dx_1}{dt} = -\alpha x_1 + u$$

$$\frac{dx_2}{dt} = \alpha x_1 - \beta x_2$$

$$\frac{d}{dt}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} -\alpha & 0 \\ \alpha & -\beta \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} u$$

Let $x_1(t)$ be the water level in Tank 1 and $x_2(t)$ be the water level in Tank 2. Let α be the rate of outflow from Tank 1 and β be rate of outflow from Tank 2. Let u be the supply of water to the system. The system can be modelled into the following differential equations:

Why are controllability and observability important?

- LQG systems, state space should be reachable, full-state feedback needed for control
- …

$$Q_o = \begin{pmatrix} C \\ CA \\ \vdots \\ CA^{n-1} \end{pmatrix}$$

observability matrix

If observability matrix is full rank, system is observable, current values of all of its state variables can be determined through output sensors.

Probabilistic Robotics – foundation of modern autonomous vehicles

probabilistic algorithms represent information by probability distributions over a whole space of possible hypotheses.

by basing control decisions on probabilistic information, these algorithms degrade nicely in the face of the various sources of uncertainty.

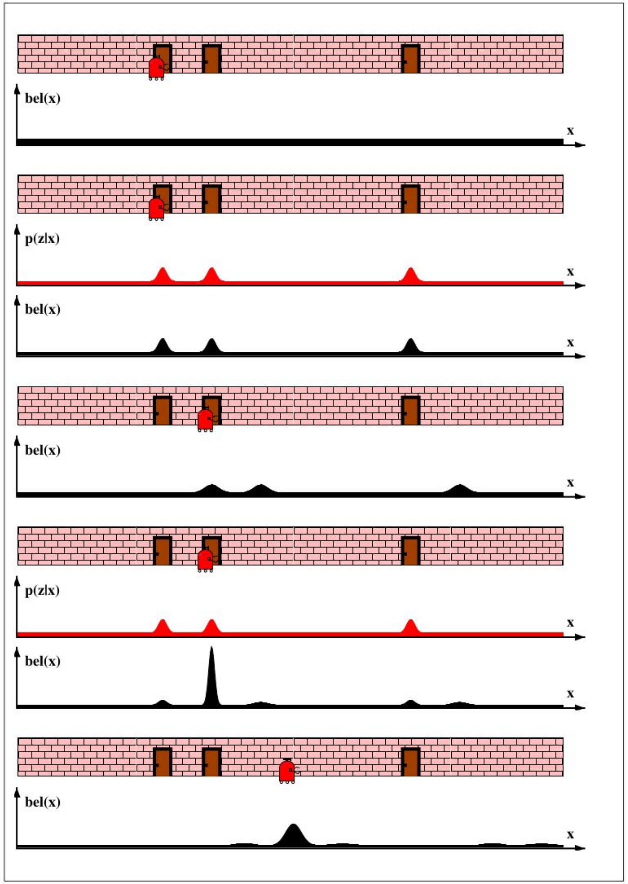


Figure 1.1 The basic idea of Markov localization: A mobile robot during global localization.

Mobile robot localization: Example of Bayes filter

Localization is the problem of estimating a robot's coordinates in an external reference frame from sensor data, using a map of the environment.

global localization, where a robot is placed somewhere in the environment and has to localize itself from scratch.

belief is represented by a probability density function over the space of all locations.

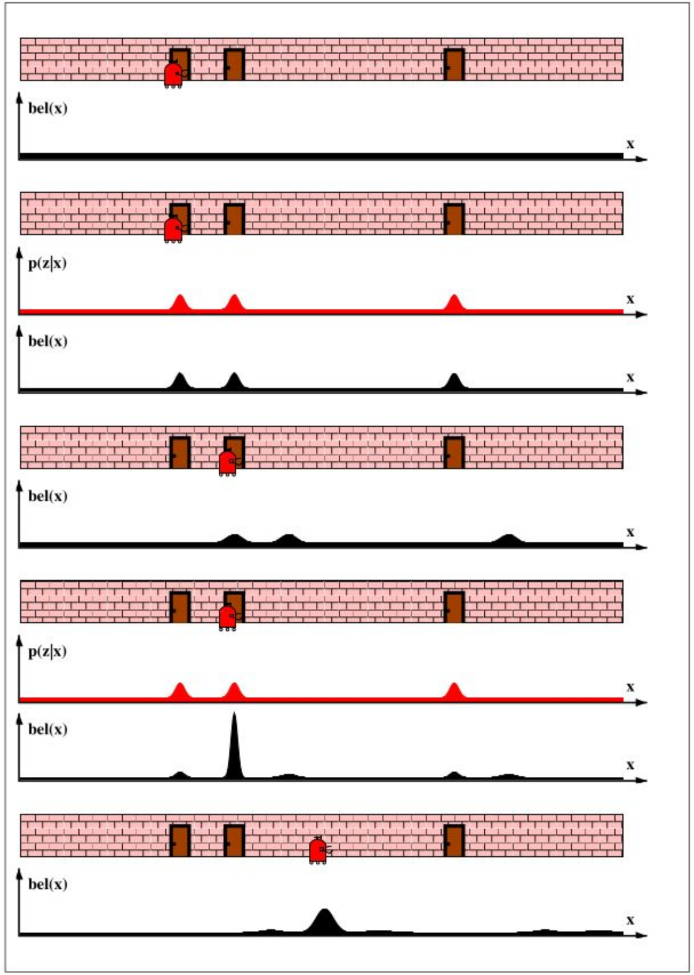
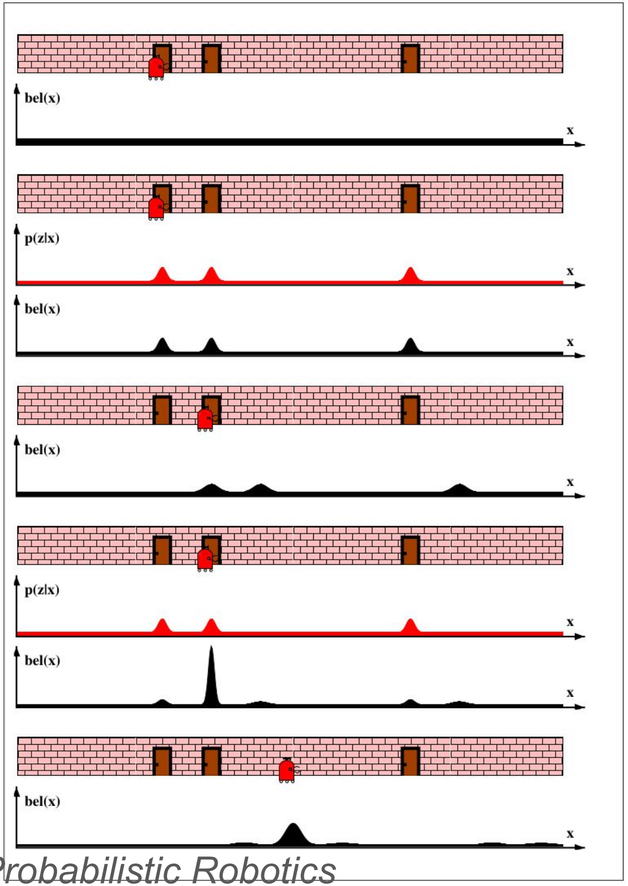


Figure 1.1 The basic idea of Markov localization: A mobile robot during global localization.

Localization examples



Probabilistic Robotics
by Sebastian Thrun

Figure 1.1 The basic idea of Markov localization: A mobile robot during global localization.

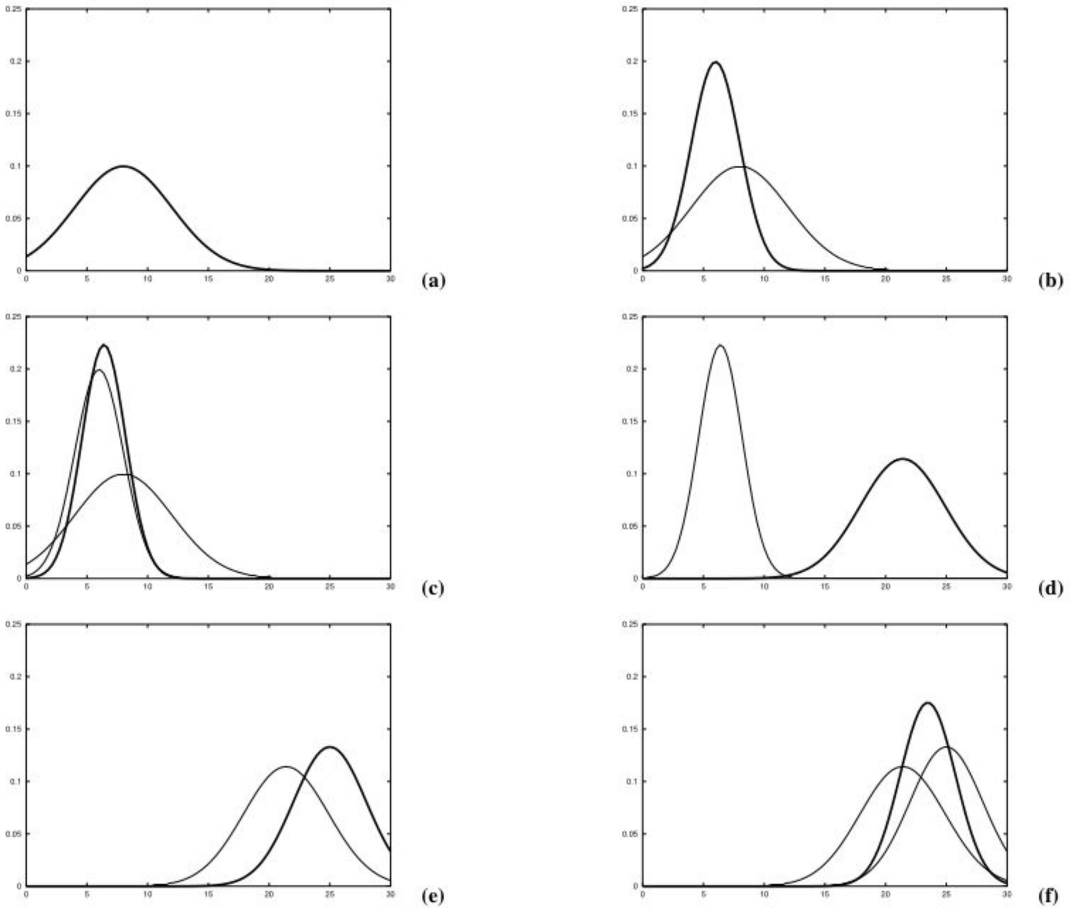


Figure 3.2 Illustration of Kalman filters: (a) initial belief, (b) a measurement (in bold) with the associated uncertainty, (c) belief after integrating the measurement into the belief using the Kalman filter algorithm, (d) belief after motion to the right (which introduces uncertainty), (e) a new measurement with associated uncertainty, and (f) the resulting belief.

Mobile robot localization: Example of Bayes filter

Algorithm Markov_localization($bel(x_{t-1}), u_t, z_t, m$)**:**

for all x_t *do*

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

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

endfor

return $bel(x_t)$

$$bel(x_0) \quad = \quad \frac{1}{|X|} \tag{7.3}$$

$$p(z_t \mid x_t, m)$$

$$p(x_t \mid u_t, x_{t-1})$$

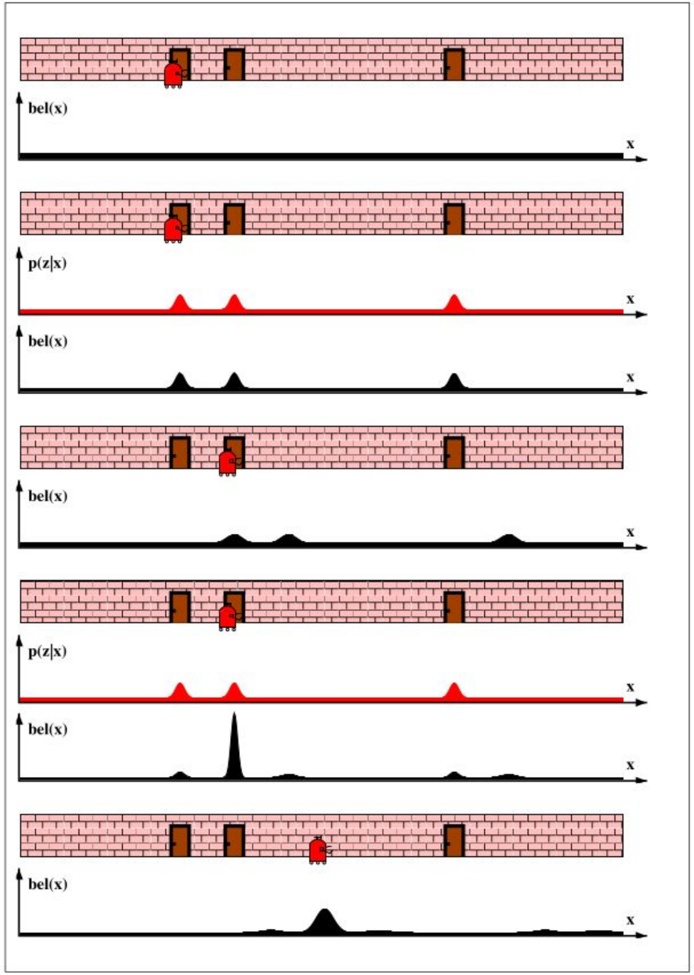


Figure 1.1 The basic idea of Markov localization: A mobile robot during global localization.

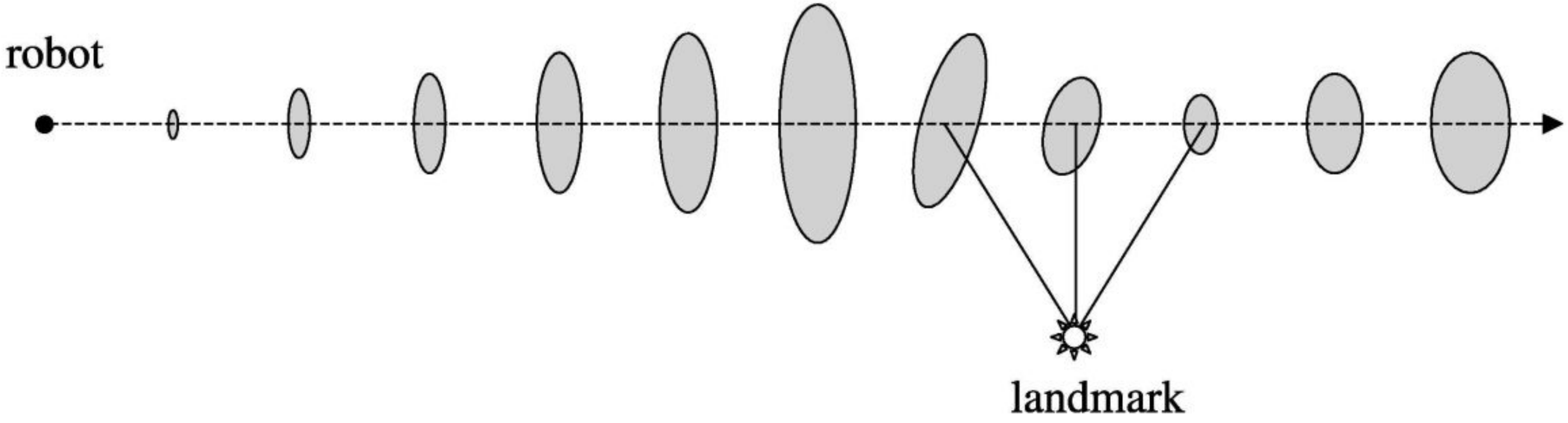


Figure 7.6 Example of localization using the extended Kalman filter. The robot moves on a straight line. As it progresses, its uncertainty increases gradually, as illustrated by the error ellipses. When it observes a landmark with known position, the uncertainty is reduced.

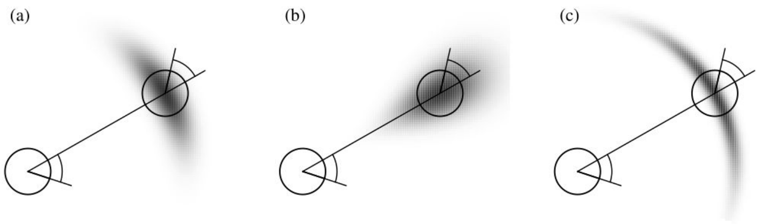


Figure 5.8 The odometry motion model, for different noise parameter settings.

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta+\omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta+\omega_t\Delta t) \\ \omega_t\Delta t + \gamma_t\Delta t \end{pmatrix} \tag{7.4}$$

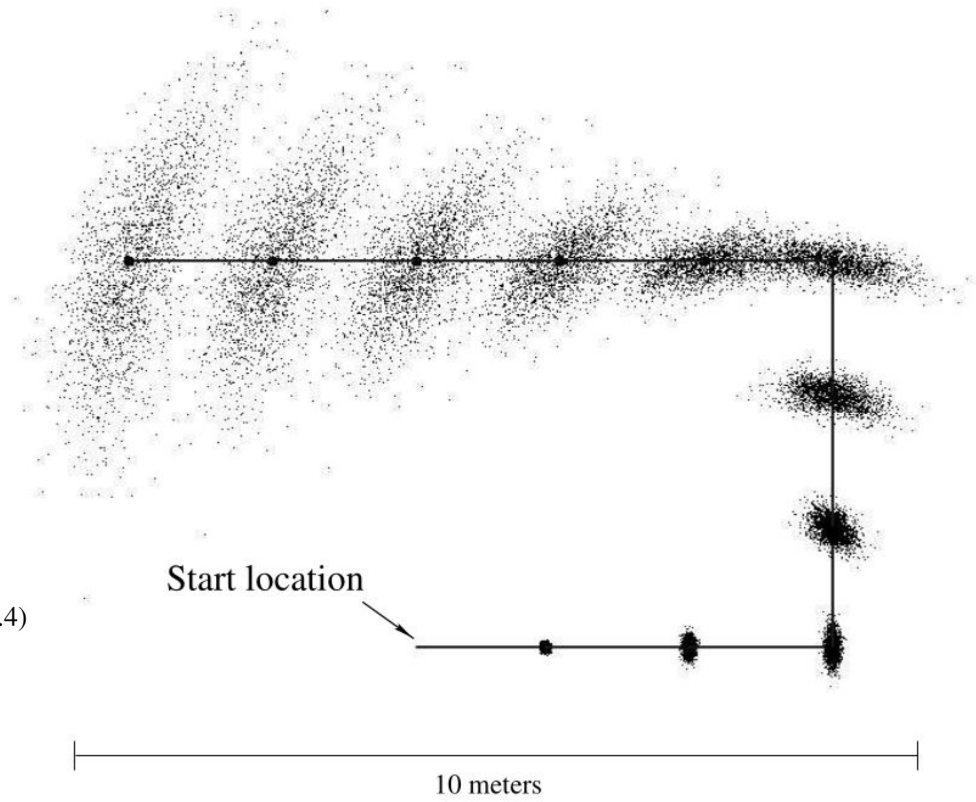


Figure 5.10 Sampling approximation of the position belief for a non-sensing robot. The solid line displays the actions, and the samples represent the robot's belief at different points in time.

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\ \sin(\theta+\omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\ \cos(\theta+\omega_t\Delta t) \\ \omega_t\Delta t + \gamma_t\Delta t \end{pmatrix} \tag{7.4}$$

$$u_t = \begin{pmatrix} v_t \\ \omega_t \end{pmatrix}, \tag{7.5}$$

$$G_t = g'(u_t, \mu_{t-1}) = \begin{pmatrix} 1 & 0 & \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 1 & \frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 0 & 1 \end{pmatrix} \tag{7.8}$$

$$
\begin{aligned}
z_t^i &= \begin{pmatrix} r_t^i \\ \phi_t^i \\ s_t^i \end{pmatrix} \\
&= \begin{pmatrix} \sqrt{(m_{j,x}-x)^2+(m_{j,y}-y)^2} \\ \operatorname{atan2}(m_{j,y}-y, m_{j,x}-x)-\theta \\ m_{j,s} \end{pmatrix} + \begin{pmatrix} \mathcal{N}(0,\sigma_r) \\ \mathcal{N}(0,\sigma_\phi) \\ \mathcal{N}(0,\sigma_s) \end{pmatrix} \qquad (7.9)
\end{aligned}
$$

$$
H_t^i = \begin{pmatrix} \frac{m_{j,x}-\bar{\mu}_{t,x}}{\sqrt{q}_t} & \frac{y_t-\bar{\mu}_{t,y}}{\sqrt{q}_t} & 0 \\ \frac{\bar{\mu}_{t,y}-y_t}{q_t} & \frac{m_{j,x}-\bar{\mu}_{t,x}}{q_t} & -1 \\ 0 & 0 & 0 \end{pmatrix} \qquad (7.15)
$$

$$
p(z_t \mid x_t, m) = \prod_i p(z_t^i \mid x_t, m) \qquad (7.16)
$$

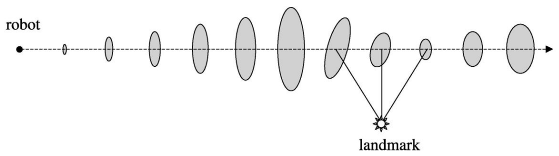


Figure 7.6 Example of localization using the extended Kalman filter. The robot moves on a straight line. As it progresses, its uncertainty increases gradually, as illustrated by the error ellipses. When it observes a landmark with known position, the uncertainty is reduced.

$$\begin{pmatrix} x' \\ y' \\ \theta' \end{pmatrix} = \begin{pmatrix} x \\ y \\ \theta \end{pmatrix} + \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\theta + \frac{v_t}{\omega_t}\sin(\theta + \omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\theta - \frac{v_t}{\omega_t}\cos(\theta + \omega_t\Delta t) \\ \omega_t\Delta t + \gamma_t\Delta t \end{pmatrix} \tag{7.4}$$

$$\begin{aligned} z_t^i &= \begin{pmatrix} r_t^i \\ \phi_t^i \\ s_t^i \end{pmatrix} \\ \imath \quad &= \begin{pmatrix} \sqrt{(m_{j,x}-x)^2 + (m_{j,y}-y)^2} \\ \operatorname{atan2}(m_{j,y}-y, m_{j,x}-x) - \theta \\ m_{j,s} \end{pmatrix} + \begin{pmatrix} \mathcal{N}(0,\sigma_r) \\ \mathcal{N}(0,\sigma_\phi) \\ \mathcal{N}(0,\sigma_s) \end{pmatrix} \end{aligned} \tag{7.9}$$

.5)

$$G_t = g'(u_t, \mu_{t-1}) = \begin{pmatrix} 1 & 0 & \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 1 & \frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 0 & 1 \end{pmatrix} \tag{7.8}$$

$$H_t^i = \begin{pmatrix} \frac{m_{j,x} - \bar\mu_{t,x}}{\sqrt{q_t}} & \frac{y_t - \bar\mu_{t,y}}{\sqrt{q_t}} & 0 \\ \frac{\bar\mu_{t,y} - y_t}{q_t} & \frac{m_{j,x} - \bar\mu_{t,x}}{q_t} & -1 \\ 0 & 0 & 0 \end{pmatrix} \tag{7.15}$$

$$p(z_t \mid x_t, m) = \prod_i p(z_t^i \mid x_t, m) \tag{7.16}$$

Algorithm EKF_localization($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t, m$):

$$\bar\mu_t = \mu_{t-1} + \begin{pmatrix} -\frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} + \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ \omega_t\Delta t \end{pmatrix}$$

$$G_t = \begin{pmatrix} 1 & 0 & \frac{v_t}{\omega_t}\cos\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\cos(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 1 & \frac{v_t}{\omega_t}\sin\mu_{t-1,\theta} - \frac{v_t}{\omega_t}\sin(\mu_{t-1,\theta} + \omega_t\Delta t) \\ 0 & 0 & 1 \end{pmatrix}$$

$$\bar\Sigma_t = G_t\,\Sigma_{t-1}\,G_t^T + R_t$$

$$Q_t = \begin{pmatrix} \sigma_r & 0 & 0 \\ 0 & \sigma_\phi & 0 \\ 0 & 0 & \sigma_s \end{pmatrix}$$

for all landmarks k in the map m do

$$\delta_k = \begin{pmatrix} \delta_{k,x} \\ \delta_{k,y} \end{pmatrix} = \begin{pmatrix} m_{k,x} - \bar\mu_{t,x} \\ m_{k,y} - \bar\mu_{t,y} \end{pmatrix}$$

$$q_k = \delta_k^T\delta_k$$

$$\hat z_t^k = \begin{pmatrix} \sqrt{q_k} \\ \operatorname{atan2}(\delta_{k,y}, \delta_{k,x}) - \bar\mu_{t,\theta} \\ m_{k,s} \end{pmatrix}$$

$$H_t^k = \frac{1}{q_k}\begin{pmatrix} \sqrt{q}_k\delta_{k,x} & -\sqrt{q}_k\delta_{k,y} & 0 \\ \delta_{k,y} & \delta_{k,x} & -1 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\Psi_k = H_t^k\,\bar\Sigma_t\,[H_t^k]^T + Q_t$$

endfor

for all observed features $z_t^i = (r_t^i\ \phi_t^i\ s_t^i)^T$ do

$$j(i) = \operatorname*{argmin}_k (z_t^i - \hat z_t^k)^T\,\Psi_k^{-1}\,(z_t^i - \hat z_t^k)$$

$$K_t^i = \bar\Sigma_t\,[H_t^{j(i)}]^T\Psi_{j(i)}^{-1}$$

endfor

$$\mu_t = \bar\mu_t + \textstyle\sum_i K_t^i\,(z_t^i - \hat z_t^{j(i)})$$

$$\Sigma_t = (I - \textstyle\sum_i K_t^i\,H_t^{j(i)})\,\bar\Sigma_t$$

return μ_t, Σ_t

Discussion

- Why localization?
- Why Markovian?
- Why EKF?

Kalman filter

$$x_t = \begin{pmatrix} x_{1,t} \\ x_{2,t} \\ \vdots \\ x_{n,t} \end{pmatrix} \quad \text{and} \quad u_t = \begin{pmatrix} u_{1,t} \\ u_{2,t} \\ \vdots \\ u_{m,t} \end{pmatrix}$$

Bayes filter

Filtering and prediction for linear systems

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

State at time t, mean μ_t and the covariance Σ_t

State transition probability

Markov assumption

Sensor updates

$$p(x_t \mid u_t, x_{t-1})$$

What is the posterior?

$$\begin{aligned} & p(x_t \mid u_t, x_{t-1}) \\ = \; & \det(2\pi R_t)^{-\frac{1}{2}} \exp\left\{-\tfrac{1}{2}(x_t - A_t x_{t-1} - B_t u_t)^T R_t^{-1}(x_t - A_t x_{t-1} - B_t u_t)\right\} \end{aligned} \tag{3.4}$$

Sebastian Thrun Chapter 3.2

Kalman filter

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

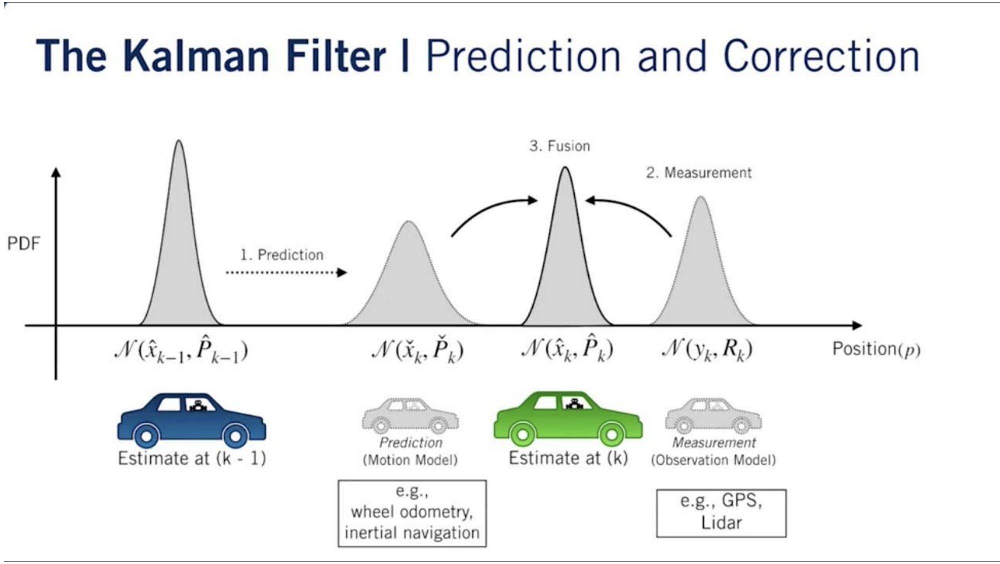
Measurements (observation model) $$z_t = C_t x_t + \delta_t$$

Initial belief $$bel(x_0) = p(x_0) = \det(2\pi\Sigma_0)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(x_0 - \mu_0)^T \Sigma_0^{-1} (x_0 - \mu_0)\right\}$$

posterior $$p(z_t \mid x_t) = \det(2\pi Q_t)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(z_t - C_t\, x_t)^T\, Q_t^{-1}\, (z_t - C_t\, x_t)\right\} \tag{3.6}$$

Predict-correct

Kalman gain applied on discrepancy between predicted and observed states



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

2: $\bar{\mu}_t = A_t\, \mu_{t-1} + B_t\, u_t$

3: $\bar{\Sigma}_t = A_t\, \Sigma_{t-1}\, A_t^T + R_t$

4: $K_t = \bar{\Sigma}_t\, C_t^T (C_t\, \bar{\Sigma}_t\, C_t^T + Q_t)^{-1}$

5: $\mu_t = \bar{\mu}_t + K_t (z_t - C_t\, \bar{\mu}_t)$

6: $\Sigma_t = (I - K_t\, C_t)\, \bar{\Sigma}_t$

7: return μ_t, Σ_t

Kalman Filter - Part 1, Prof. Jonathan Kelly, Univ. of Toronto

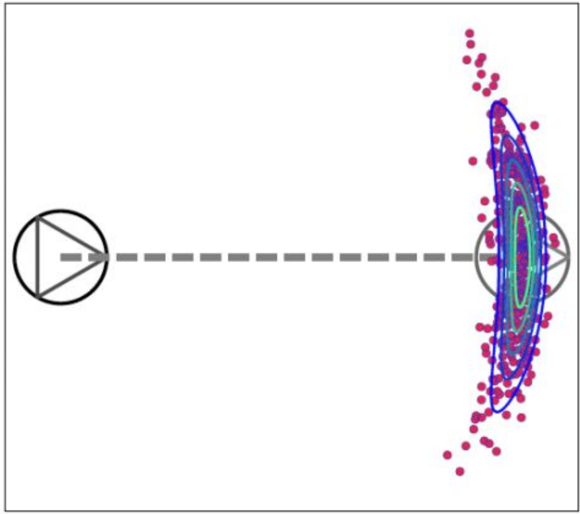
https://www.youtube.com/watch?v=LioOvUZ1MiM

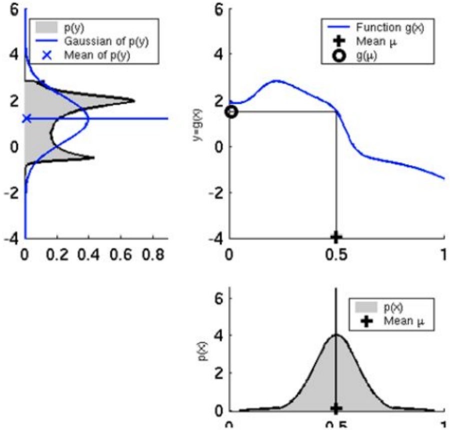
Kalman filtering options

Extended Kalman filter (EKF)

Unscented Kalman filter (UKF)

Non-linear, non-Gaussian, multi-modal world

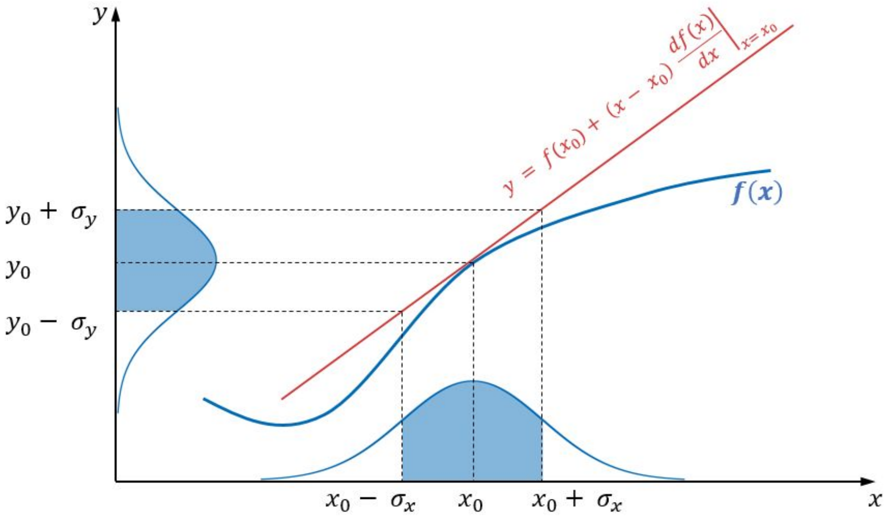




Effect of nonlinear transformation of Gaussian random variable.

Extended Kalman Filter (EKF)

- Linearize at estimated state (Jacobian matrix)
- State propagation and observation model used directly
- Jacobians can be computed offline
- Covariance propagation and update based on linearized model
- Optimality of state estimation not guaranteed, understate estimates the covariances (too optimistic)



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

$\bar{\mu}_t = g(u_t, \mu_{t-1})$

$\bar{\Sigma}_t = G_t\, \Sigma_{t-1}\, G_t^T + R_t$

$K_t = \bar{\Sigma}_t\, H_t^T (H_t\, \bar{\Sigma}_t\, H_t^T + Q_t)^{-1}$

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

$\Sigma_t = (I - K_t\, H_t)\, \bar{\Sigma}_t$

return μ_t, Σ_t

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \cdots$$
$$\cdots + \frac{h^{(n-1)}}{(n-1)!}f^{(n-1)}(x) + \frac{h^n}{n!}f^n(x+\lambda h)$$

$$g'(u_t, x_{t-1}) \quad := \quad \frac{\partial g(u_t, x_{t-1})}{\partial x_{t-1}}$$

Taylor series expansion

https://www.cc.gatech.edu/~bboots3/STR-Spring2018/Lectures/Lecture16/Lecture16_notes.pdf

Extended Kalman Filter (EKF)

$$g'(u_t, x_{t-1}) \quad := \quad \frac{\partial g(u_t, x_{t-1})}{\partial x_{t-1}}$$

$$\begin{aligned} g(u_t, x_{t-1}) \quad &\approx \quad g(u_t, \mu_{t-1}) \; + \; \underbrace{g'(u_t, \mu_{t-1})}_{=:\, G_t} (x_{t-1} - \mu_{t-1}) \\ &= \quad g(u_t, \mu_{t-1}) \; + \; G_t \; (x_{t-1} - \mu_{t-1}) \end{aligned} \tag{3.50}$$

$$\begin{aligned} & p(x_t \mid u_t, x_{t-1}) \\ & \quad \approx \quad \det(2\pi R_t)^{-\frac{1}{2}} \; \exp\left\{-\tfrac{1}{2} \; [x_t - g(u_t, \mu_{t-1}) - G_t \; (x_{t-1} - \mu_{t-1})]^T \right. \\ & \qquad\quad \left. R_t^{-1} \; [x_t - g(u_t, \mu_{t-1}) - G_t \; (x_{t-1} - \mu_{t-1})]\right\} \end{aligned} \tag{3.51}$$

Extended Kalman Filter (EKF)

$$\begin{aligned} h(x_t) &\approx h(\bar{\mu}_t) + \underbrace{h'(\bar{\mu}_t)}_{=:\,H_t} (x_t - \bar{\mu}_t) \\ &= h(\bar{\mu}_t) + H_t \, (x_t - \bar{\mu}_t) \end{aligned} \tag{3.52}$$

with $h'(x_t) = \frac{\partial h(x_t)}{\partial x_t}$. Written as a Gaussian, we have

$$\begin{aligned} p(z_t \mid x_t) = \; & \det\left(2\pi Q_t\right)^{-\frac{1}{2}} \exp\left\{-\tfrac{1}{2} \left[z_t - h(\bar{\mu}_t) - H_t \, (x_t - \bar{\mu_t})\right]^T \right. \\ & \left. Q_t^{-1} \left[z_t - h(\bar{\mu}_t) - H_t \, (x_t - \bar{\mu_t})\right]\right\} \end{aligned} \tag{3.53}$$

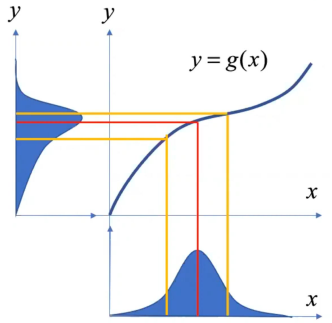
Kalman Filter Algorithms

1: **Algorithm Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$)**:**
2: $\bar{\mu}_t = A_t\ \mu_{t-1} + B_t\ u_t$
3: $\bar{\Sigma}_t = A_t\ \Sigma_{t-1}\ A_t^T + R_t$
4: $K_t = \bar{\Sigma}_t\ C_t^T (C_t\ \bar{\Sigma}_t\ C_t^T + Q_t)^{-1}$
5: $\mu_t = \bar{\mu}_t + K_t (z_t - C_t\ \bar{\mu}_t)$
6: $\Sigma_t = (I - K_t\ C_t)\ \bar{\Sigma}_t$
7: return μ_t, Σ_t

1: **Algorithm Extended_Kalman_filter**($\mu_{t-1}, \Sigma_{t-1}, u_t, z_t$)**:**
2: $\bar{\mu}_t = g(u_t, \mu_{t-1})$
3: $\bar{\Sigma}_t = G_t\ \Sigma_{t-1}\ G_t^T + R_t$
4: $K_t = \bar{\Sigma}_t\ H_t^T (H_t\ \bar{\Sigma}_t\ H_t^T + Q_t)^{-1}$
5: $\mu_t = \bar{\mu}_t + K_t (z_t - h(\bar{\mu}_t))$
6: $\Sigma_t = (I - K_t\ H_t)\ \bar{\Sigma}_t$
7: return μ_t, Σ_t

	Kalman filter	EKF
state prediction (Line 2)	$A_t\ \mu_{t-1} + B_t\ u_t$	$g(u_t, \mu_{t-1})$
measurement prediction (Line 5)	$C_t\ \bar{\mu}_t$	$h(\bar{\mu}_t)$

Unscented Kalman Filter - (Julier and Uhlmann [1997])



- Different method to compute covariance matrices
- Does not use the Riccati equations or cov propagation update laws
- Unscented transform
  - find a set of deterministic vectors called sigma points whose ensemble mean and covariance are equal to Z and P
  - apply known nonlinear function y = h(x) to each deterministic vector to obtain transformed vectors.

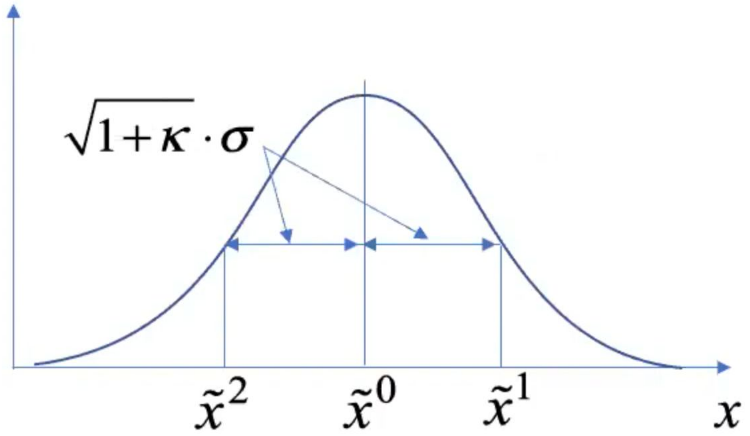
$$p(x) = \frac{1}{\sqrt{2\pi}\cdot\sigma}\exp\left(-\frac{(x-\bar{x})^2}{\sigma^2}\right)$$

$$\tilde{x}^0 = \bar{x} \qquad W_0 = \frac{\kappa}{1+\kappa}$$

$$\tilde{x}^1 = \bar{x} + \sqrt{1+\kappa}\cdot\sigma$$

$$\tilde{x}^2 = \bar{x} - \sqrt{1+\kappa}\cdot\sigma \qquad W_1 = W_2 = \frac{1}{2(1+\kappa)}$$

where κ is a parameter of sigma points to be tuned, and W_i is the weight of the i^{th} sigma point used for computing mean and variance.



Unscented Kalman Filter - (Julier and Uhlmann [1997])

- Weighted mean of sigma points agree with true mean of the Gaussian distribution

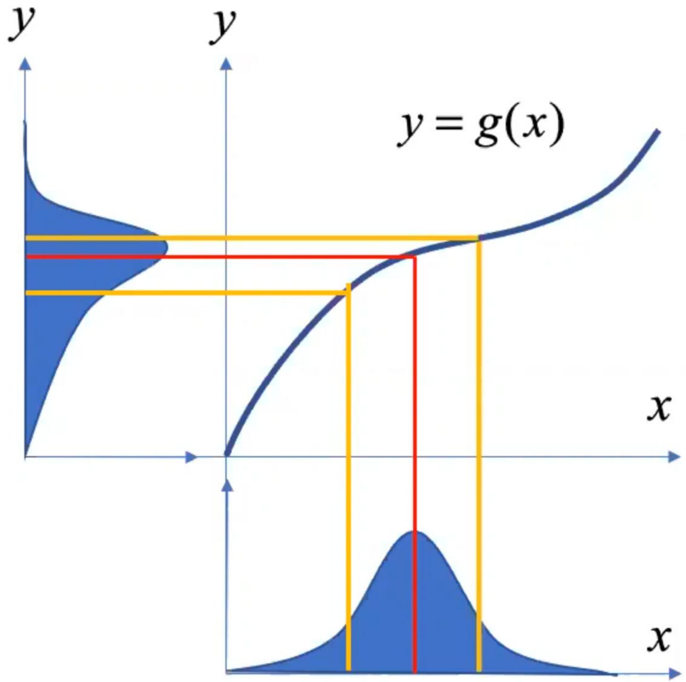
$$\sum_{i=0}^{2} W_i \tilde{x}^i = \frac{\kappa}{1+\kappa}\bar{x} + \frac{1}{2(1+\kappa)}\left\{(\bar{x}+\sqrt{1+\kappa}\cdot\sigma)+(\bar{x}-\sqrt{1+\kappa}\cdot\sigma)\right\}$$

$$= \frac{\kappa}{1+\kappa}\bar{x} + \frac{2}{2(1+\kappa)}\bar{x} = \bar{x}$$

- Weighted variance of sigma points agree with true variance of the Gaussian distribution

$$\sum_{i=0}^{2} W_i (\tilde{x}^i - \bar{x})^2 = \frac{\kappa}{1+\kappa}(\bar{x}-\bar{x}) + \frac{1}{2(1+\kappa)}\left\{(\bar{x}+\sqrt{1+\kappa}\cdot\sigma-\bar{x})^2+(\bar{x}-\sqrt{1+\kappa}\cdot\sigma-\bar{x})^2\right\}$$

$$= \frac{2}{2(1+\kappa)}(\sqrt{1+\kappa}\cdot\sigma)^2 = \sigma^2$$



Unscented Kalman Filter - (Julier and Uhlmann [1997])

- Weighted mean of sigma points agree with true mean of the Gaussian distribution

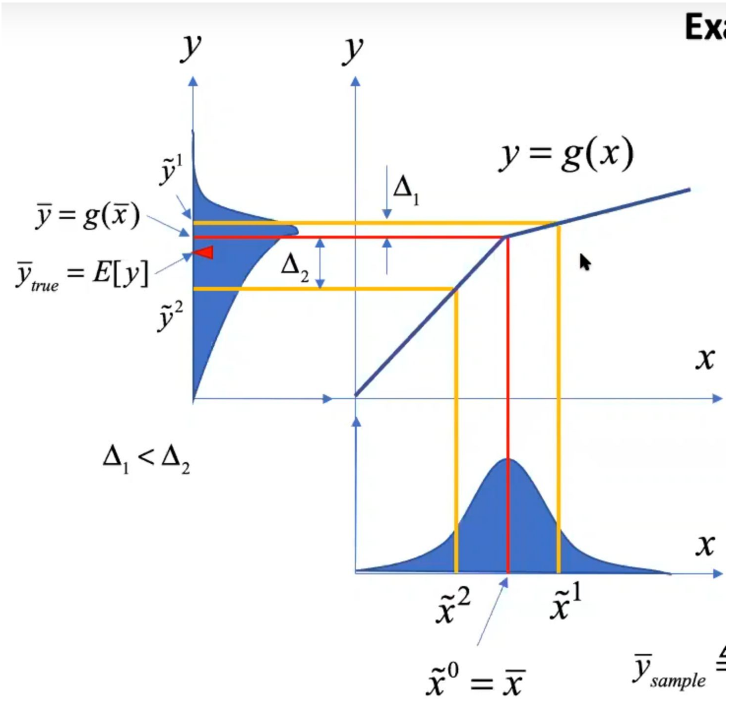
$$\sum_{i=0}^{2} W_i \tilde{x}^i = \frac{\kappa}{1+\kappa}\bar{x} + \frac{1}{2(1+\kappa)}\left\{(\bar{x}+\sqrt{1+\kappa}\cdot\sigma)+(\bar{x}-\sqrt{1+\kappa}\cdot\sigma)\right\}$$

$$= \frac{\kappa}{1+\kappa}\bar{x} + \frac{2}{2(1+\kappa)}\bar{x} = \bar{x}$$

- Weighted variance of sigma points agree with true variance of the Gaussian distribution

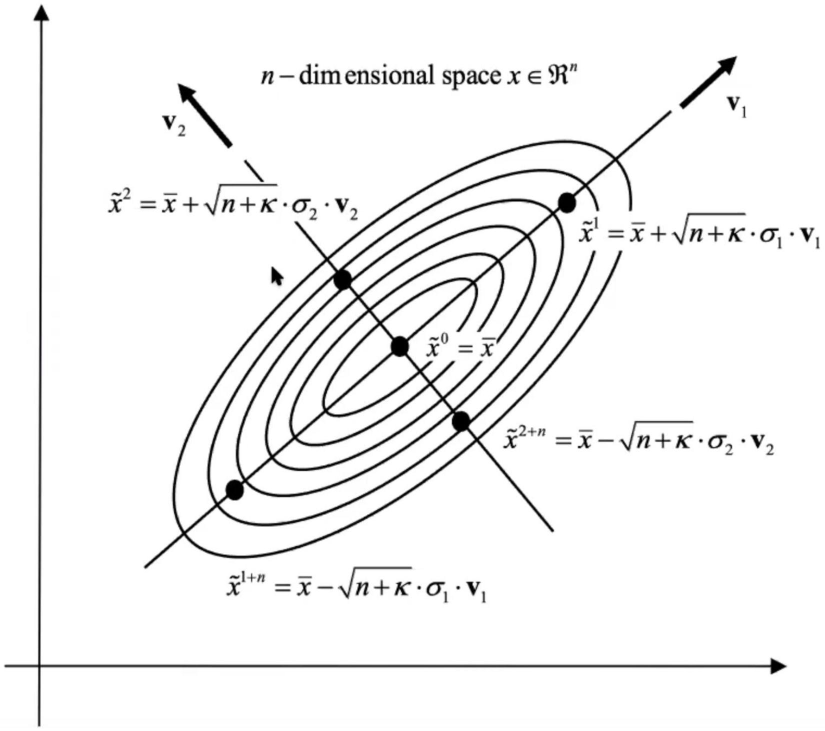
$$\sum_{i=0}^{2} W_i (\tilde{x}^i - \bar{x})^2 = \frac{\kappa}{1+\kappa}(\bar{x}-\bar{x}) + \frac{1}{2(1+\kappa)}\left\{(\bar{x}+\sqrt{1+\kappa}\cdot\sigma-\bar{x})^2+(\bar{x}-\sqrt{1+\kappa}\cdot\sigma-\bar{x})^2\right\}$$

$$= \frac{2}{2(1+\kappa)}(\sqrt{1+\kappa}\cdot\sigma)^2 = \sigma^2$$



Unscented Kalman Filter - (Julier and Uhlmann [1997])

For n dimensional Gaussian, 2n+1 sigma points can be used



Recursive algorithm for unscented Kalman filter

- Given $\hat{x}_{t-1}$ and $P_{t\text{-}1}$, sample sigma points by computing eigenvalues and eigen vectors of $P_{t\text{-}1}$;
- Propagate the sigma points through the nonlinear model to obtain $\tilde{x}^{i*}_{t|t-1} = f(\tilde{x}^i_{t-1}, t-1)$;
- From the (2n+1) sigma points compute the mean and variance:

$$\hat{x}_{t|t-1,sample} = \sum_{i=0}^{2n} W_i \hat{x}^{i*}_{t|t-1} \qquad P_{t|t-1,sample} = \sum_{i=0}^{i=2n} W_i (\tilde{x}^i_{t|t-1} - \hat{x}_{t|t-1,sample})(\tilde{x}^i_{t|t-1} - \hat{x}_{t|t-1,sample})^T + Q_{t-1}$$

- Sample again (2n+1) sigma points for $P_{t|t\text{-}1,sample}$;
- Transform the propagated sigma points to output estimate $\tilde{y}^i_t$ based on the nonlinear measurement equation, and compute the estimated output

$$\hat{y}_{t,sample} = \sum_{i=0}^{2n} W_i \tilde{y}^i_t \qquad \tilde{y}^i_t = h(\tilde{x}^i_{t|t-1}, t)$$

- Evaluate the innovation covariance and the cross covariance by using (2n+1) points of propagated output estimates to find the Kalman gain

$$K_t = P_{xy} P_y^{-1} \qquad P_y = \sum_{i=0}^{2n} W_i (\tilde{y}^i_t - \hat{y}_{t,sample})(\tilde{y}^i_t - \hat{y}_{t,sample})^T + R_t \qquad P_{xy} = \sum_{i=0}^{2n} W_i (\tilde{x}^i_{t|t-1} - \hat{x}_{t|t-1,sample})(\tilde{y}^i_t - \hat{y}_{t,sample})^T$$

- Update the state estimate with the Kalman gain;

$$\hat{x}_t = \hat{x}_{t|t-1,sample} + K_t[y_t - \hat{y}_{t,sample}]$$

- Update the a posteriori covariance;

$$P_t \cong P_{t|t-1,sample} - K_t P_y K_t^T$$

- Set $t = t + 1$, and repeat the above process.

Unscented Kalman Filter - (Julier and Uhlmann [1997])

- Given $\hat{x}_{t-1}$ and $P_{t\text{-}1}$, sample sigma points by computing eigenvalues and eigen vectors of $P_{t\text{-}1}$;
- Propagate the sigma points through the nonlinear model to obtain $\tilde{x}^{i*}_{t|t-1} = f(\tilde{x}^{i}_{t-1}, t-1)$;
- From the (2n+1) sigma points compute the mean and variance:

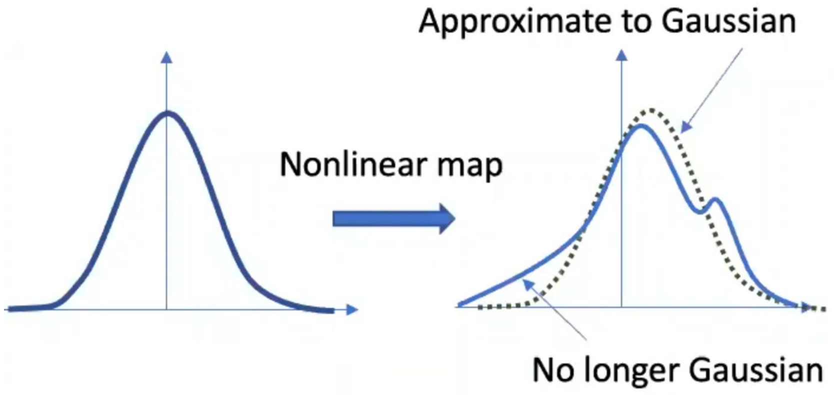
$$\hat{x}_{t|t-1,sample} = \sum_{i=0}^{2n} W_i \hat{x}^{i*}_{t|t-1} \qquad P_{t|t-1,sample} = \sum_{i=0}^{i=2n} W_i (\tilde{x}^{i}_{t|t-1} - \hat{x}_{t|t-1,sample})(\tilde{x}^{i}_{t|t-1} - \hat{x}_{t|t-1,sample})^T + Q_{t-1}$$

- Sample again (2n+1) sigma points for $P_{t|t\text{-}1,sample}$;
- Transform the propagated sigma points to output estimate $\tilde{y}^i_t$ based on the nonlinear measurement equation, and compute the estimated output

$$\hat{y}_{t,sample} = \sum_{i=0}^{2n} W_i \tilde{y}^i_t \qquad \tilde{y}^i_t = h(\tilde{x}^i_{t|t-1}, t)$$

- Evaluate the innovation covariance and the cross covariance by using (2n+1) points of propagated output estimates to find the Kalman gain

$$K_t = P_{xy} P_y^{-1} \qquad P_y = \sum_{i=0}^{2n} W_i (\tilde{y}^i_t - \hat{y}_{t,sample})(\tilde{y}^i_t - \hat{y}_{t,sample})^T + R_t \qquad P_{xy} = \sum_{i=0}^{2n} W_i (\tilde{x}^i_{t|t-1} - \hat{x}_{t|t-1,sample})(\tilde{y}^i_t - \hat{y}_{t,sample})^T$$



- Update the state estimate with the Kalman gain;

$$\hat{x}_t = \hat{x}_{t|t-1,sample} + K_t [y_t - \hat{y}_{t,sample}]$$

- Update the a posteriori covariance;

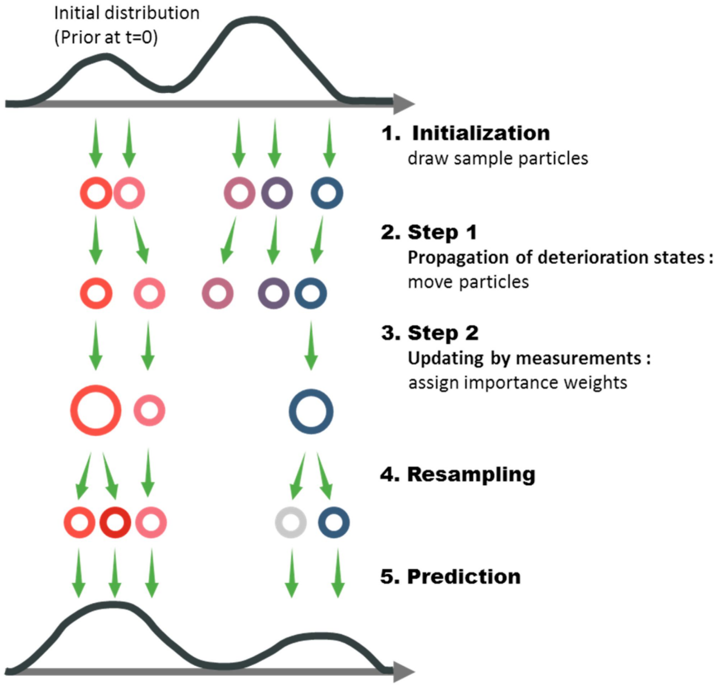
$$P_t \cong P_{t|t-1,sample} - K_t P_y K_t^T$$

- Set $t = t + 1$, and repeat the above process.

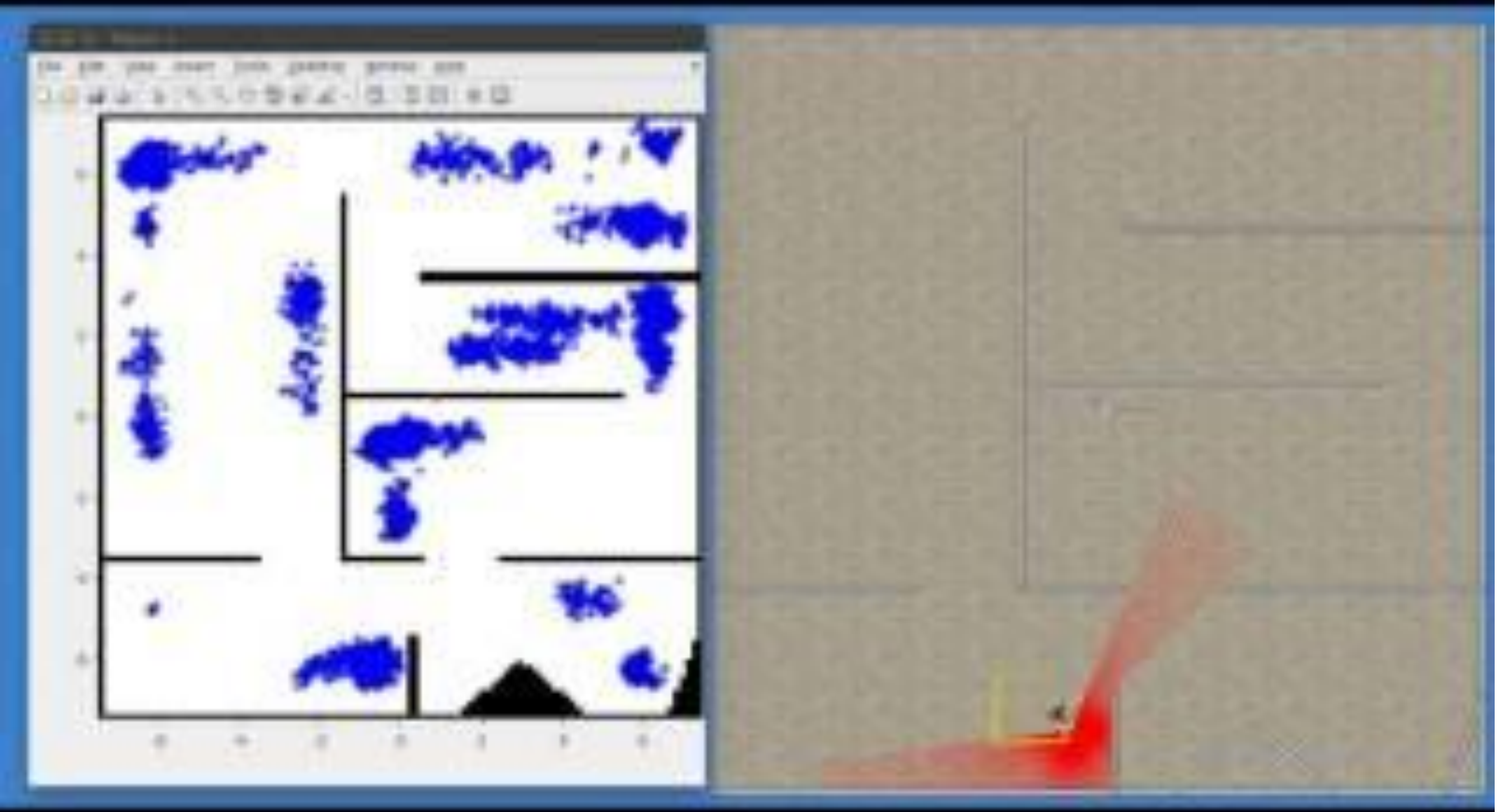
- No Jacobian, no partial derivatives are needed.
- The estimated covariance using sigma points is more accurate than the Jacobian-based one.
- ***Caveat!*** The distribution of random variables after transformed through nonlinear equations, e.g. $f(x,t)$, $h(x,t)$, is no longer Gaussian, although the original distribution was Gaussian. Unscented Kalman Filter approximates this distribution to a Gaussian and characterizes with mean and covariance. Although this approximation is accurate to the 2nd order, the discrepancy from a complete Gaussian may grow, as the process is repeated.

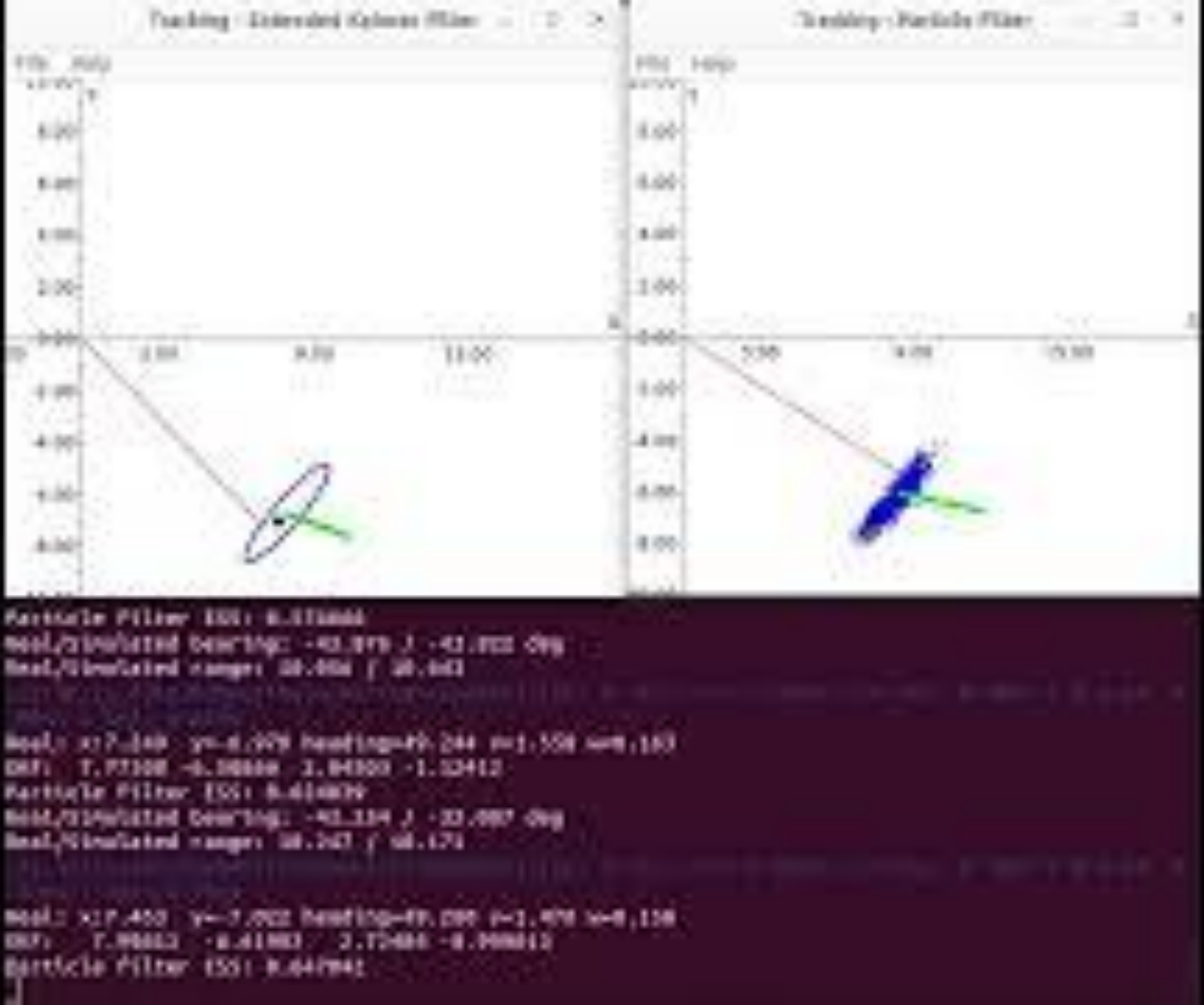
Non-parametric filters

- No fixed functional form
- Approximate posterior by finite number of values
- Can represent a broader space of distributions



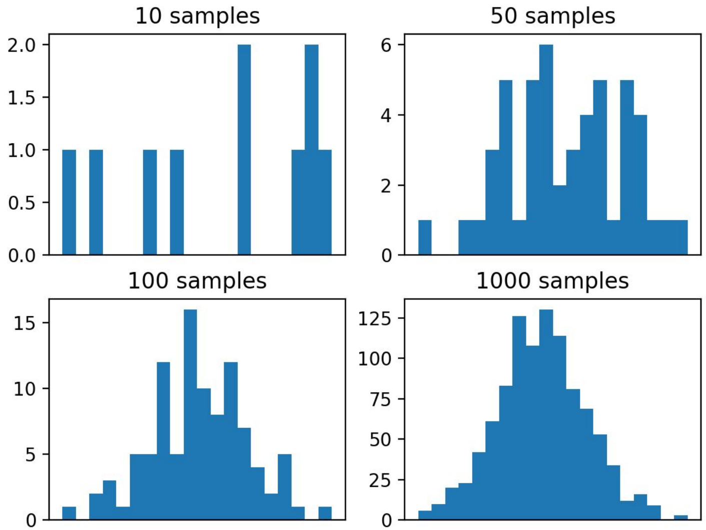
Sebastian Thrun
Chapter 4





Tracking - Extended Kalman Filter
File Help
Tracking - Particle Filter
File Help
Particle Filter ESS:
Real/Simulated bearing:
Real/Simulated range:
Real:
EKF:
Particle Filter ESS:
Real/Simulated bearing:
Real/Simulated range:
Real:
EKF:
Particle Filter ESS:

Non-parametric representation of probability distributions

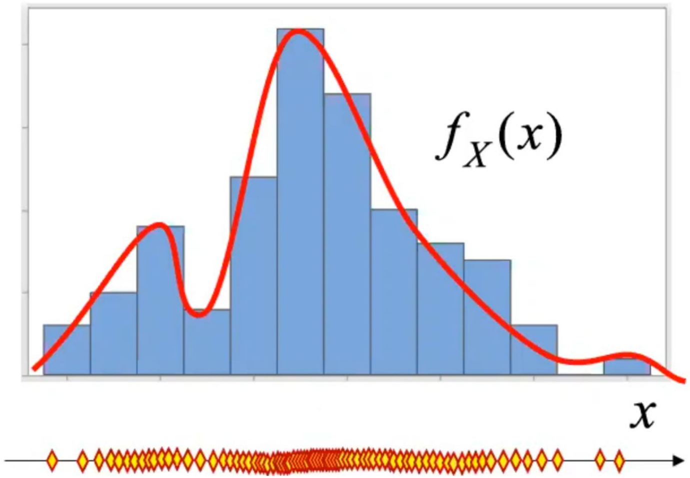


Histograms

Monte Carlo approximation helps in the estimation of mean

$$\mu = \mathbb{E}(f(\boldsymbol{X}))$$

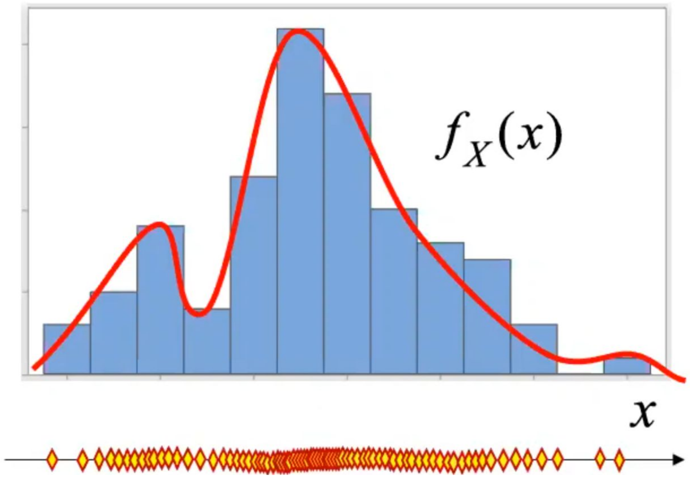
Non-parametric representation of probability distributions



$$\mu = \mathbb{E}(f(\boldsymbol{X}))$$

Draw **particles** that capture probability distribution

Particle filtering



Particles

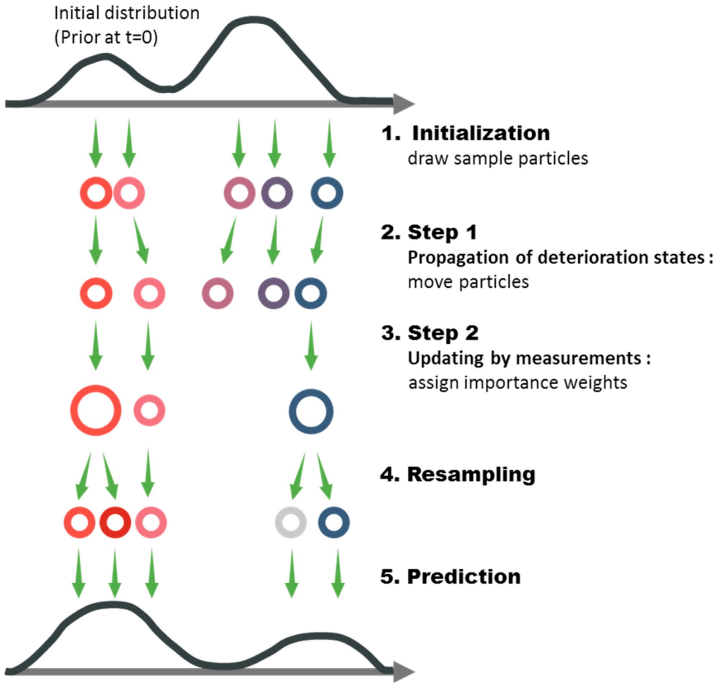
- Bayes filter, non-parametric
- Draw particles that capture probability distribution
- Compute how each point behaves under transformations

Monte Carlo approximation helps in the estimation of mean

$$\mu = \mathbb{E}(f(\boldsymbol{X}))$$

Importance sampling

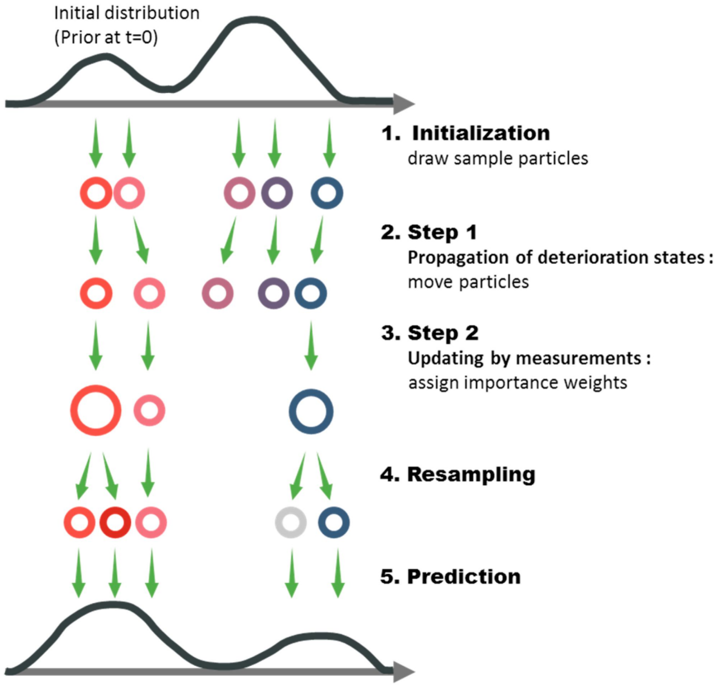
- Get samples from the interesting or important region
- Sample from a distribution that overweights important region
- Adjust estimate to account for having sampled from this other distribution
- Study one distribution while sampling from another



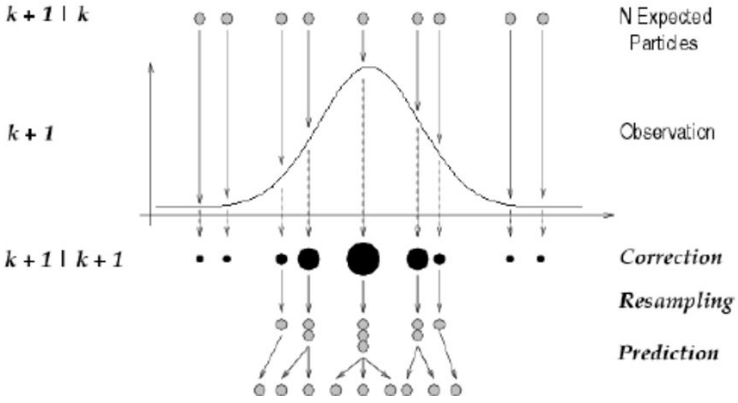
Importance sampling

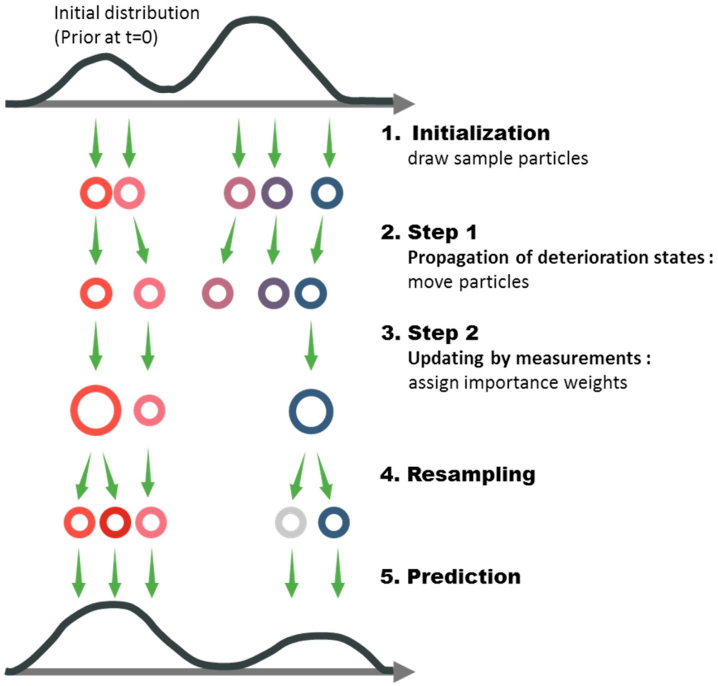
- Get samples from the interesting or important region
- Sample from a distribution that overweights important region
- Adjust estimate to account for having sampled from this other distribution
- Study one distribution while sampling from another

$$\mu = \int_{\mathcal{D}} f(\boldsymbol{x})p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \int_{\mathcal{D}} \frac{f(\boldsymbol{x})p(\boldsymbol{x})}{q(\boldsymbol{x})}\,q(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = \mathbb{E}_q\left(\frac{f(\boldsymbol{X})p(\boldsymbol{X})}{q(\boldsymbol{X})}\right)$$



Particle filtering





Particle filtering

Belief

$$\mathcal{X}_t \quad := \quad x_t^{[1]}, x_t^{[2]}, \ldots, x_t^{[M]}$$

$$x_t^{[m]} \quad \sim \quad p(x_t \mid z_{1:t}, u_{1:t})$$

Measurement update

$$w_t^{[m]} = p(z_t \mid x_t^{[m]})$$

Sebastian Thrun
Chapter 4.2

Particle filtering

belief $bel(x_t)$

Belief

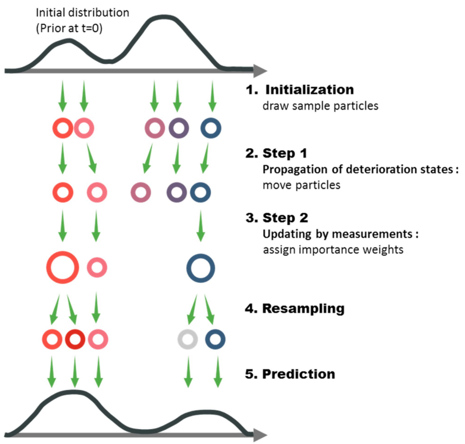
$$\mathcal{X}_t \quad := \quad x_t^{[1]}, x_t^{[2]}, \ldots, x_t^{[M]}$$

$$x_t^{[m]} \quad \sim \quad p(x_t \mid z_{1:t}, u_{1:t})$$

Measurement update

$$w_t^{[m]} = p(z_t \mid x_t^{[m]})$$

Algorithm Particle_filter($\mathcal{X}_{t-1}, u_t, z_t$):
$\quad \bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$
$\quad$ for $m = 1$ to M do
$\quad\quad$ sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$
$\quad\quad$ $w_t^{[m]} = p(z_t \mid x_t^{[m]})$
$\quad\quad$ $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$
$\quad$ endfor
$\quad$ for $m = 1$ to M do
$\quad\quad$ draw i with probability $\propto w_t^{[i]}$
$\quad\quad$ add $x_t^{[i]}$ to $\mathcal{X}_t$
$\quad$ endfor
$\quad$ return $\mathcal{X}_t$



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

$\bar{\mu}_t = A_t \; \mu_{t-1} + B_t \; u_t$

$\bar{\Sigma}_t = A_t \; \Sigma_{t-1} \; A_t^T + R_t$

$K_t = \bar{\Sigma}_t \; C_t^T (C_t \; \bar{\Sigma}_t \; C_t^T + Q_t)^{-1}$

$\mu_t = \bar{\mu}_t + K_t (z_t - C_t \; \bar{\mu}_t)$

$\Sigma_t = (I - K_t \; C_t) \; \bar{\Sigma}_t$

return μ_t, Σ_t

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

$\bar{\mu}_t = g(u_t, \mu_{t-1})$

$\bar{\Sigma}_t = G_t \; \Sigma_{t-1} \; G_t^T + R_t$

$K_t = \bar{\Sigma}_t \; H_t^T (H_t \; \bar{\Sigma}_t \; H_t^T + Q_t)^{-1}$

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

$\Sigma_t = (I - K_t \; H_t) \; \bar{\Sigma}_t$

return μ_t, Σ_t

Algorithm Particle_filter($\mathcal{X}_{t-1}, u_t, z_t$):

$\bar{\mathcal{X}}_t = \mathcal{X}_t = \emptyset$

for $m = 1$ to M do

  sample $x_t^{[m]} \sim p(x_t \mid u_t, x_{t-1}^{[m]})$

  $w_t^{[m]} = p(z_t \mid x_t^{[m]})$

  $\bar{\mathcal{X}}_t = \bar{\mathcal{X}}_t + \langle x_t^{[m]}, w_t^{[m]} \rangle$

endfor

for $m = 1$ to M do

  draw i with probability $\propto w_t^{[i]}$

  add $x_t^{[i]}$ to $\mathcal{X}_t$

endfor

return $\mathcal{X}_t$