Last lecture ...

HMM filtering is an instance of Bayesian filtering. To handle large discrete state spaces or continuous spaces, we need computationally efficient representations, but the underlying idea of filtering remains the same:

- Prediction update

$$p(S_t = s') = \sum_{s \in S} p(S_t = s' | S_{t-1} = s) p(S_{t-1} = s)$$

- Observation update

$$p(S_t = s | Z_t = z) = \eta p(Z_t = z | S_t = s) p(S_t = s)$$

Kalman filtering. The Kalman filter makes the following key assumptions on the system dynamic model and the observation model:

- they are **linear**;
- they have **Gaussian** noise.

As a result, the probability distribution over the state is always Gaussian. This makes the Kalman filter very efficient. At the same time, it is also the main reason for the limitations.

state

action

$$\mathrm{x}_{t+1} = A_t \mathrm{x}_t + B_t \mathrm{u}_t + \varepsilon_t \qquad \varepsilon_t \sim \mathcal{N}(0, Q_t)$$

$$\mathrm{z}_t = H_t \mathrm{x}_t + \delta_t \qquad \delta_t \sim \mathcal{N}(0, R_t)$$

observation

Gaussian noise

linear transformations

The equations above specify the probabilistic state-transition model $p(\mathrm{x}_{t+1} \mid \mathrm{x}_t, \mathrm{u}_t)$ conditioned on the action u_t and the observation model $p(\mathrm{z}_t \mid \mathrm{x}_t)$.

Following the conventions, we write these equations in terms of random variables (RVs). They serve the purpose of specifying the state-transition and observation models.

The state-transition model is conditioned on the action u_t . It is not required, but is more general. Similarly, we can have $p(s_{t+1} \mid s_t, a_t)$ for HMM filtering or particle filtering. u_t or a_t must be known at each time step t.

Example. A robot moves in an one-dimensional space. At time t, the state is (x_t, v_t), where x_t and v_t are the position and the speed of the robot, respectively. The robot can apply a force to control the acceleration a_t. The speedometer on board measures v_t, but not x_t.

$$
\begin{aligned}
x_{t+1} &= x_t + v_t \\
v_{t+1} &= v_t + a_t + \phi_t \\
z_t &= v_t + \psi_t
\end{aligned}
$$

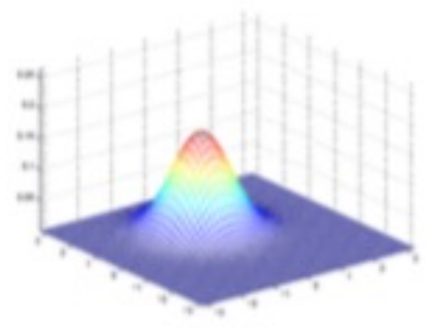
$$\Downarrow$$

$$\begin{pmatrix} x_{t+1} \\ v_{t+1} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_t \\ v_t \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} a_t + \begin{pmatrix} 0 \\ 1 \end{pmatrix} \phi_t$$

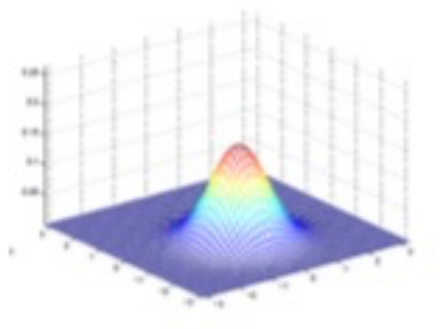
$$z_t = \begin{pmatrix} 0 & 1 \end{pmatrix} \begin{pmatrix} x_t \\ v_t \end{pmatrix} + \psi_t$$

Our objective is to calculate $p(\mathrm{x}_t \mid \mathrm{u}_{0:t-1}, \mathrm{z}_{1:t})$, which is Gaussian with mean μ_t and covariance matrix Σ_t:

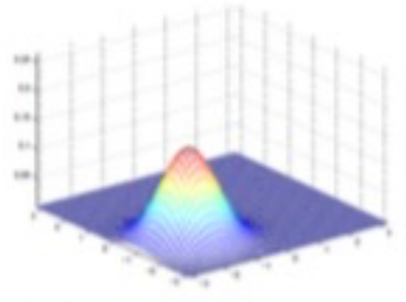
$$p(\mathrm{x}) = \frac{1}{(2\pi)^{D/2}} \frac{1}{|\Sigma|^{1/2}} \exp\left\{ -\frac{1}{2}(\mathrm{x}-\mu)^T \Sigma^{-1} (\mathrm{x}-\mu) \right\}$$



- μ = [1; 0]
- Σ = [1 0; 0 1]

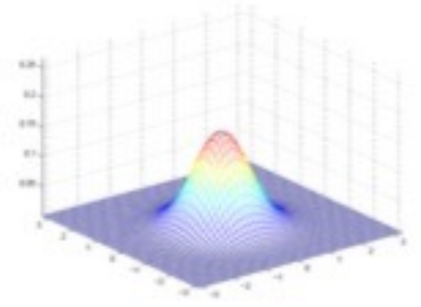


- μ = [-.5; 0]
- Σ = [1 0; 0 1]

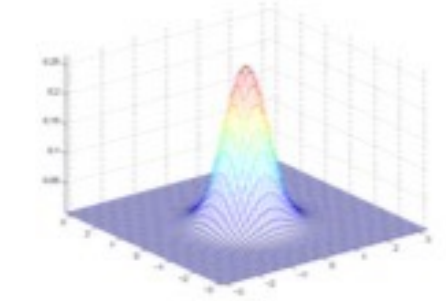


- μ = [-1; -1.5]
- Σ = [1 0; 0 1]

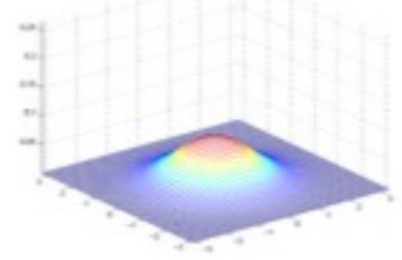
Translate



- μ = [0; 0]
- Σ = [1 0 ; 0 1]

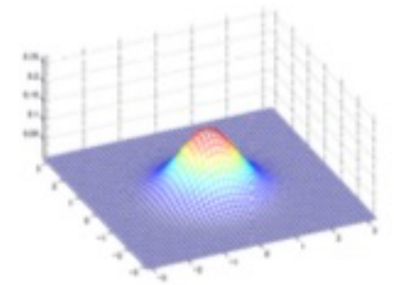


- μ = [0; 0]
- Σ = [.6 0 ; 0 .6]

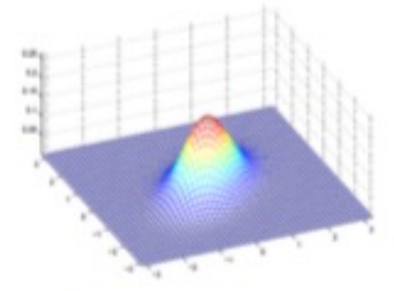


- μ = [0; 0]
- Σ = [2 0 ; 0 2]

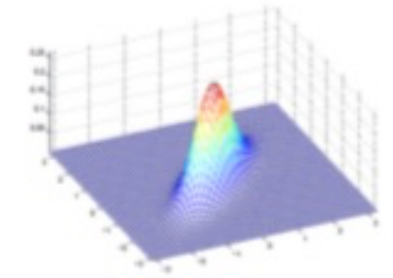
Change the spread.



- μ = [0; 0]
- Σ = [1 0; 0 1]

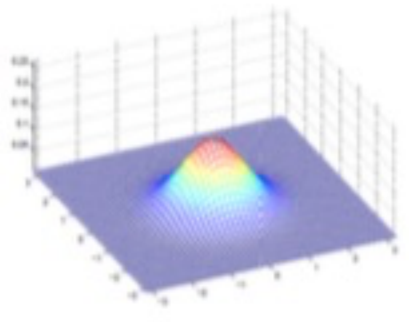


- μ = [0; 0]
- Σ = [1 0.5; 0.5 1]

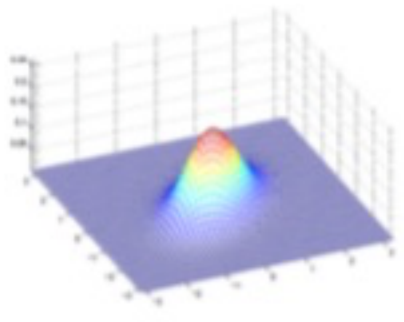


- μ = [0; 0]
- Σ = [1 0.8; 0.8 1]

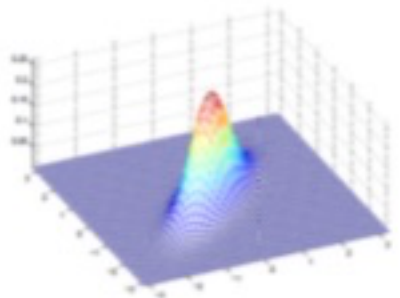
Change the spread and rotate.



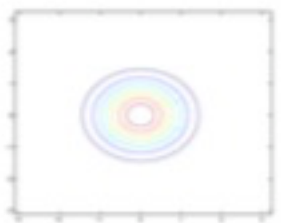
- $\mu = [0; 0]$
- $\Sigma = [1\ \ 0; 0\ \ 1]$

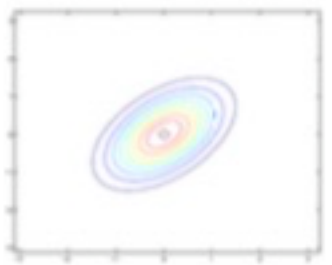


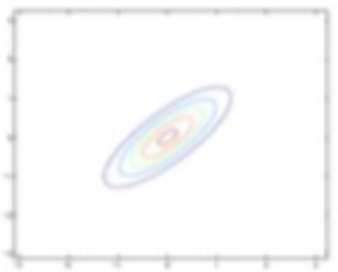
- $\mu = [0; 0]$
- $\Sigma = [1\ \ 0.5; 0.5\ \ 1]$



- $\mu = [0; 0]$
- $\Sigma = [1\ \ 0.8; 0.8\ \ 1]$







Kalman filtering update μ_t and Σ_t recursively:

- At time 0, we have μ_0, Σ_0.
- For $t = 1,2,\ldots,$

  prediction update

  $$\mu_{t+1}^{-} = A_t\mu_t + B_t\mathrm{u}_t$$
  $$\Sigma_{t+1}^{-} = A_t\Sigma_t A_t^T + Q_t$$

  The control action does not affect the covariance, i.e., the uncertainty. This is a unique property of linear systems.

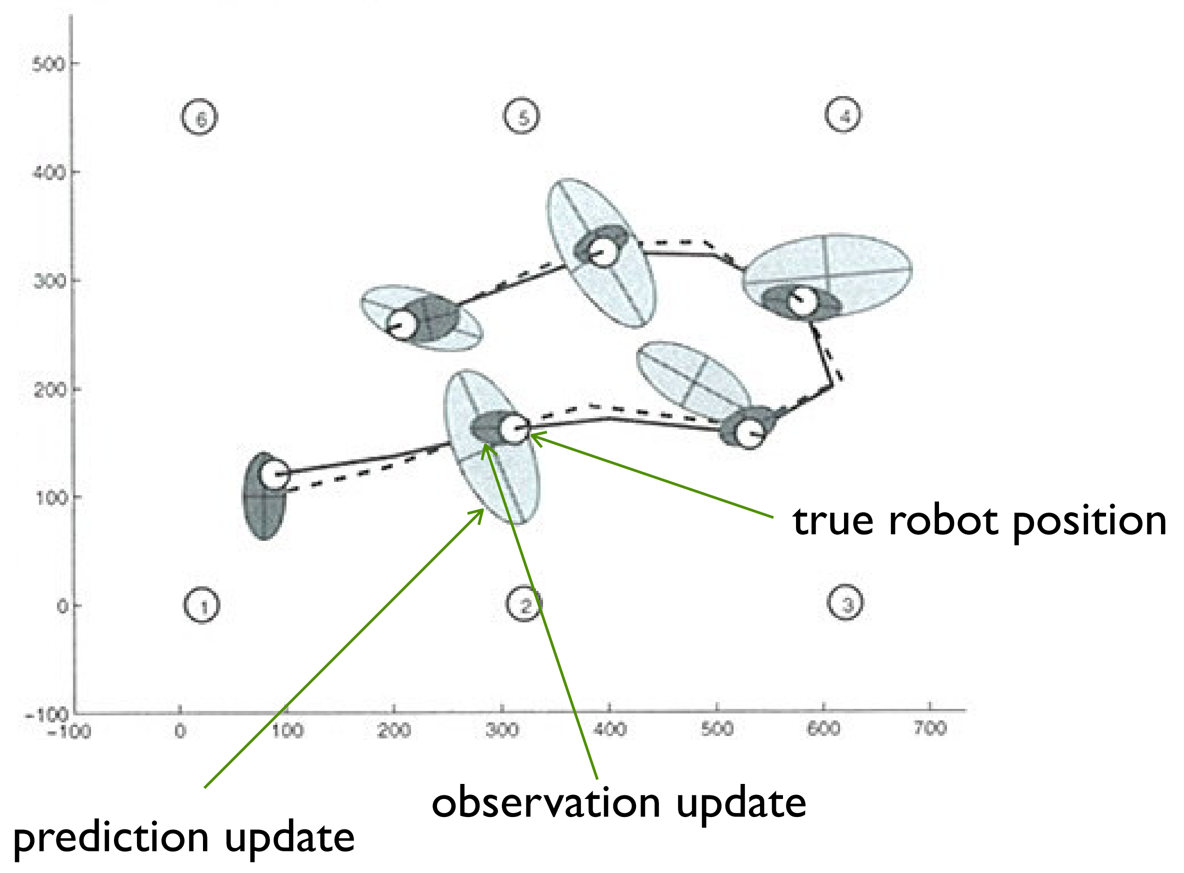
  observation update

  $$\mu_{t+1} = \mu_{t+1}^{-} + K_{t+1}(z_{t+1} - H_{t+1}\mu_{t+1}^{-})$$
  $$\Sigma_{t+1} = (I - K_{t+1}H_{t+1})\Sigma_{t+1}^{-}$$

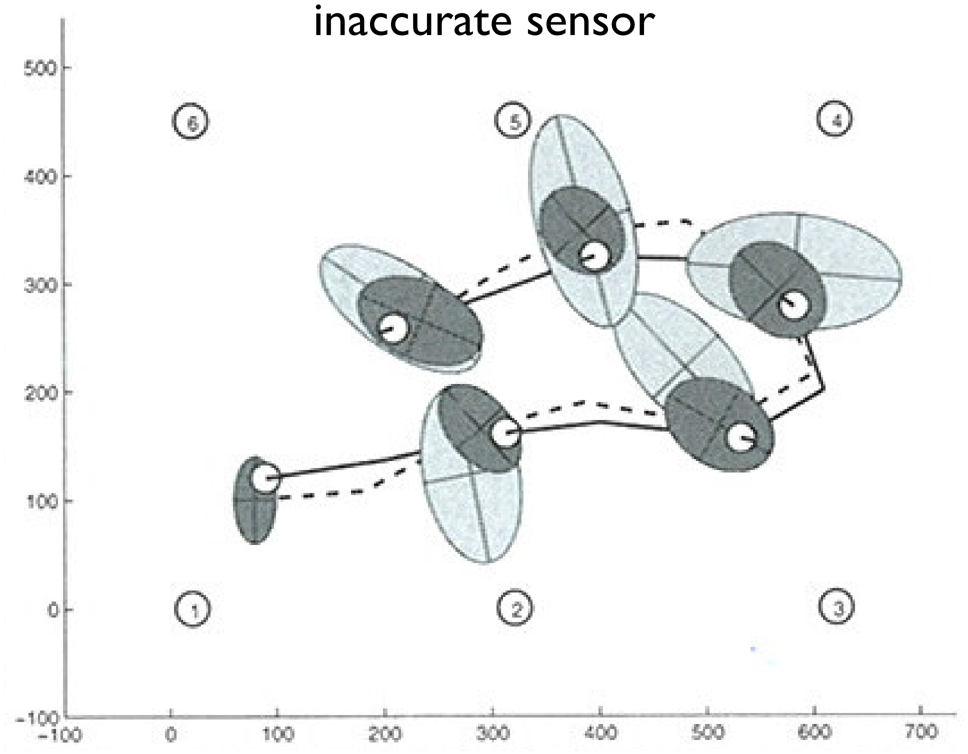
  where $K_{t+1} = \Sigma_{t+1}^{-}H_{t+1}^T(H_{t+1}\Sigma_{t+1}^{-}H_{t+1}^T + R_{t+1})^{-1}$ is the Kalman gain.

$$\boxed{\begin{aligned} \mathrm{x}_{t+1} &= A_t\mathrm{x}_t + B_t\mathrm{u}_t + \varepsilon_t \\ \mathrm{z}_t &= H_t\mathrm{x}_t + \delta_t \\ \varepsilon_t &\sim \mathcal{N}(0,Q_t) \\ \delta_t &\sim \mathcal{N}(0,R_t) \end{aligned}}$$

accurate sensor



inaccurate sensor



With an accurate sensor, the state distribution is sharper after the observation update.

Example. An illustrative example for localization with Kalman filtering.

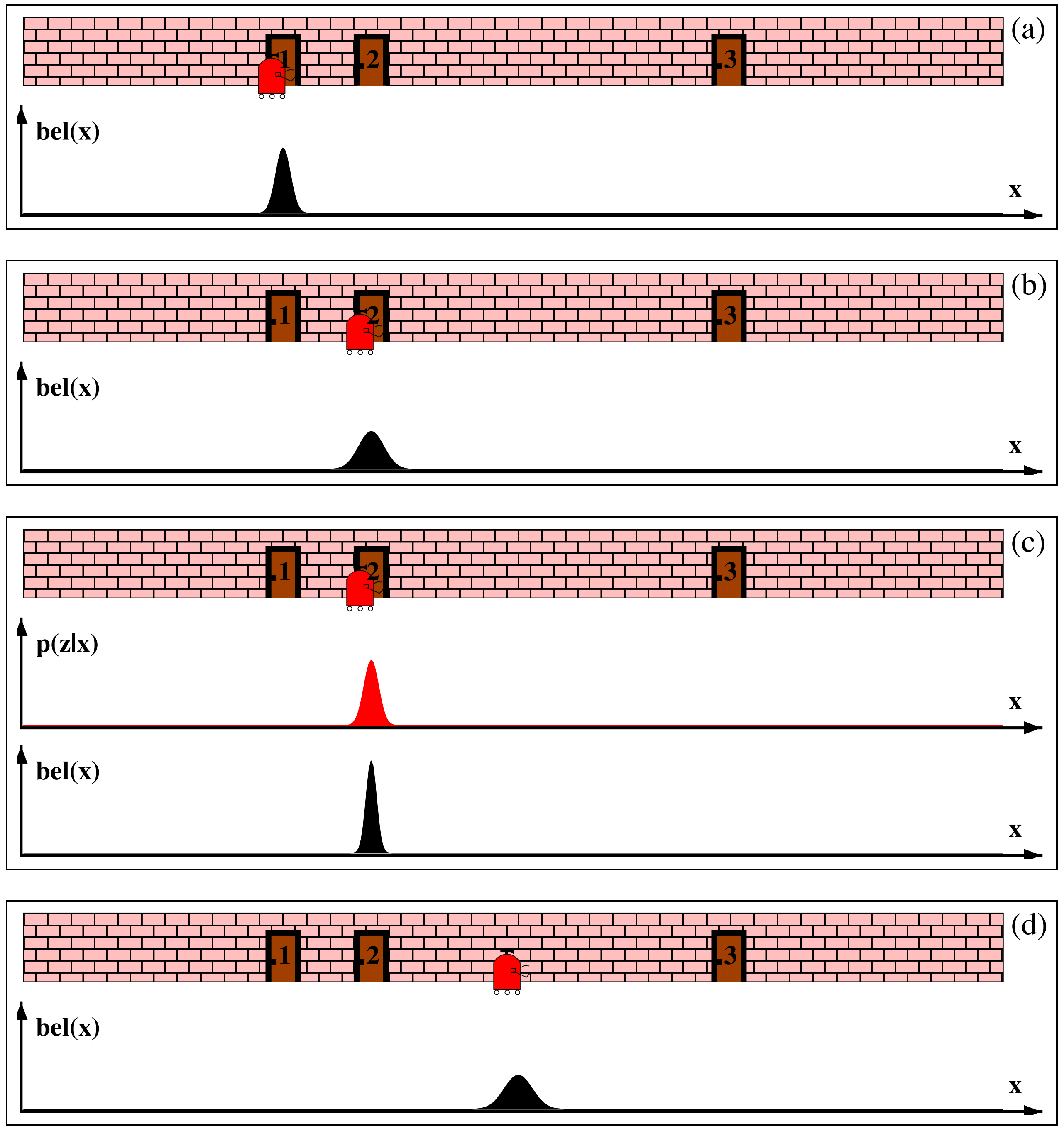
initial belief

prediction update

observation update

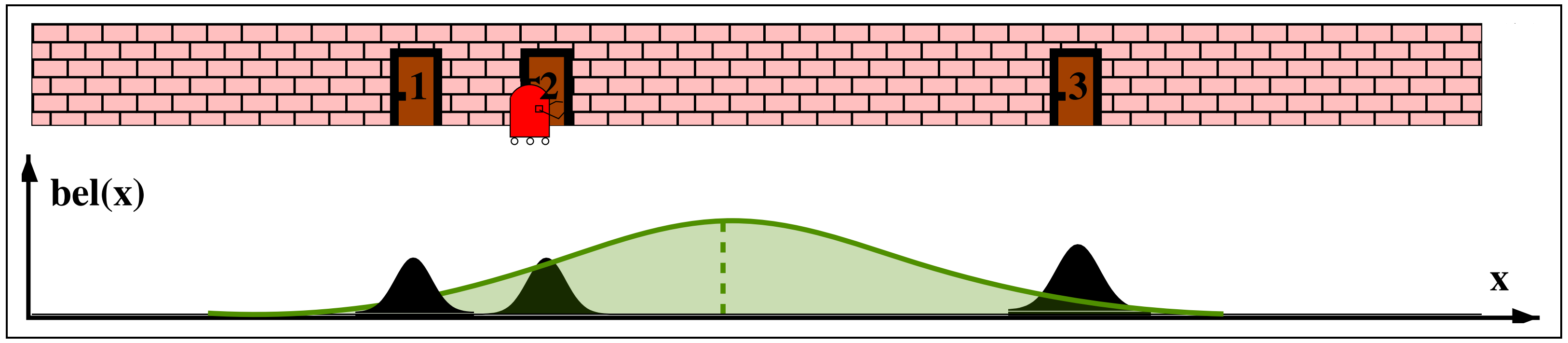
prediction update

...

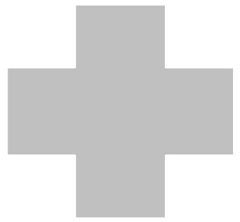


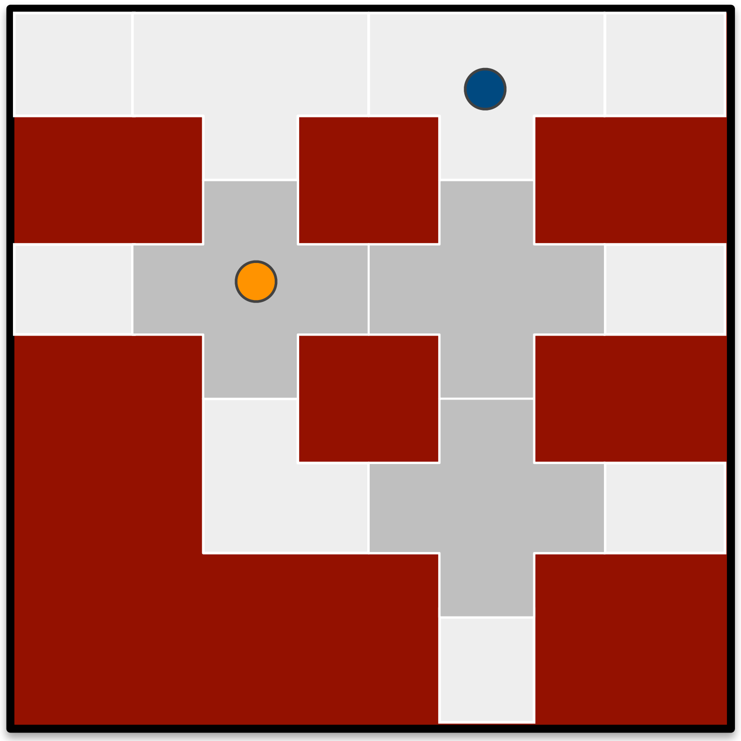
Kalman filtering is $O(D^2)$, where D is the dimension of the state space S. In contrast, HMM filtering is $O(|S|^2)$. PF is $O(N)$, where N is the number of particles. In general, $D \ll N \ll |S|$. So Kalman filtering is the most efficient one computationally.

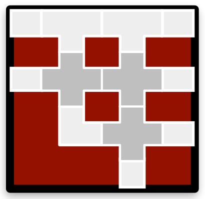
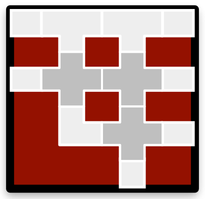
At the same time, the Gaussian assumption is restrictive. Kalman filtering does not perform well for complex, multi-modal distributions. Consider the example below. The ground-truth state distribution has three modes, each located at a door. The approximating Gaussian has its mean located in front the wall. Very likely, the robot will make the wrong decision as a result.



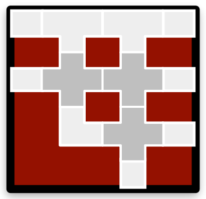
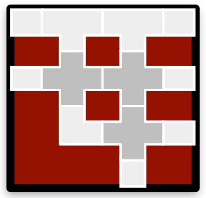
Localization with a map. What information does a map provide? How do we use a map for robot localization? The map contains spatial information about the environment. We use the map M to build the observation model $p(z\,|\,s, M)$:





$p(z\,|\,s)$: $p($ $|$ ● ,  $) = 0,$ $p($ $|$ ● ,  $) = 1,$

Localization:

$p(s\,|\,z)$: $p($ ● $|$,  $) = 0,$ $p($ ● $|$,  $) = 1/3$

Localization variants. There are various localization tasks that differ in their assumptions:

- tracking
- global localization
- “kidnapped robot”
- active localization.

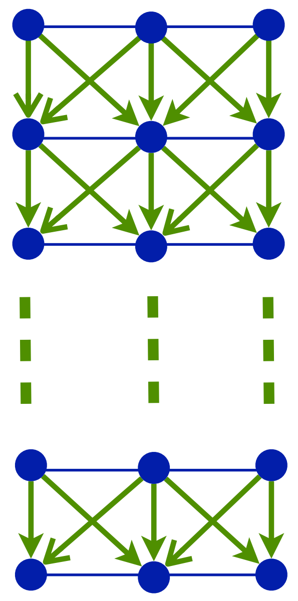
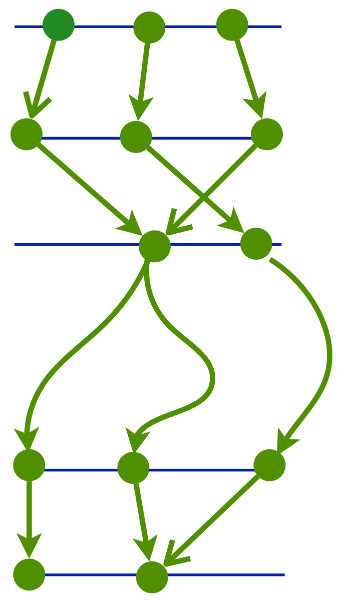
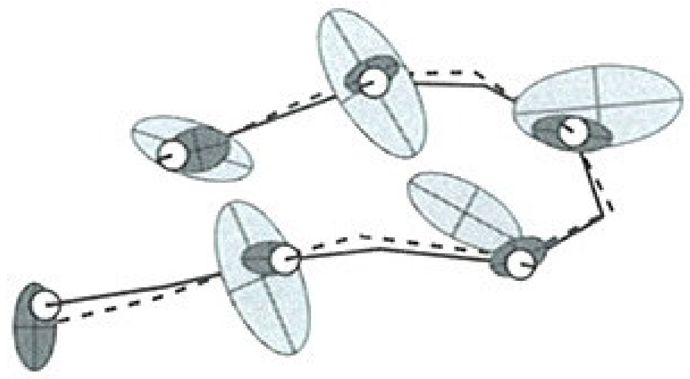
Tracking assumes good knowledge of the robot initial state. Further, the uncertainty remains small over time. If the underlying state distribution is well approximated as Gaussian, then Kalman filtering is a good solution.

In global localization, the robot initial state may not be known accurately. The uncertainty on the robot state may become very large. Under these conditions, PF is the most common solution.

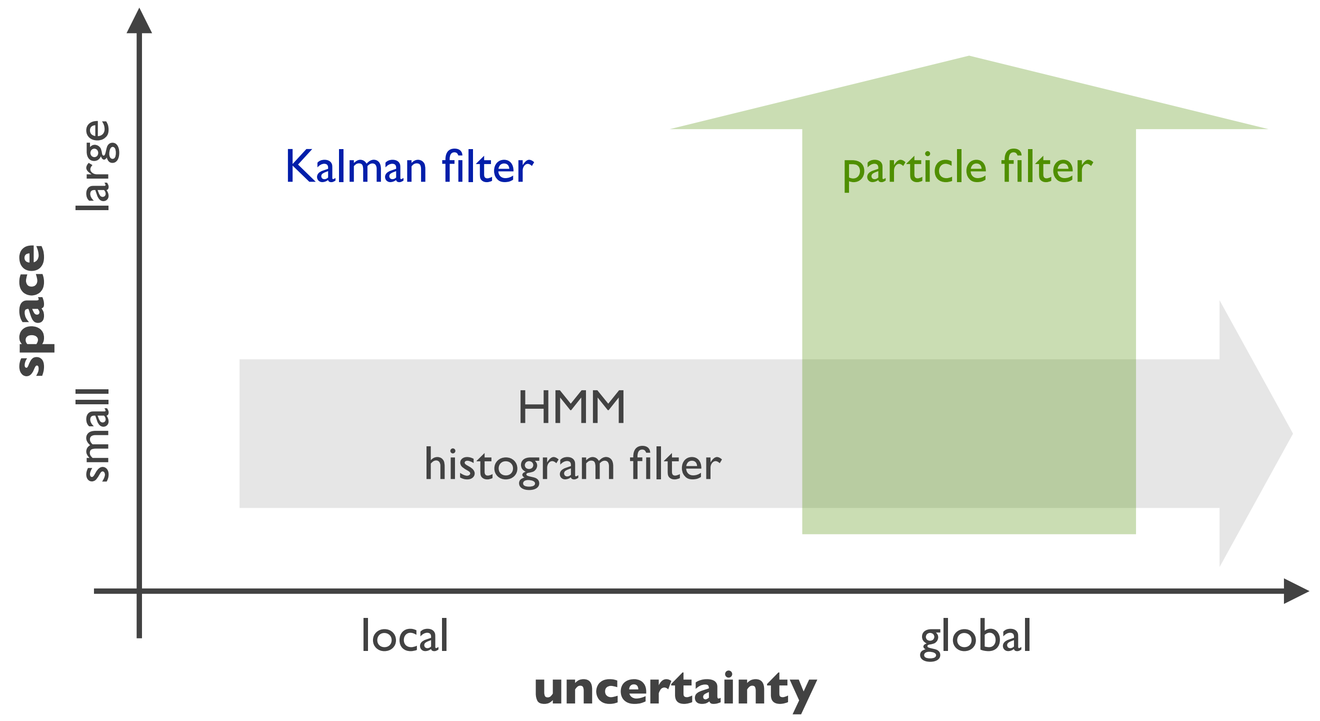
In the “kidnaped robot” setting, the robot is teleported to another location. It is related to global localization, but differs in that the robot does not realize when the kidnapping occurs. This problem tries to captures **unmodeled** localization failures.

All filtering algorithms **passively** absorb information. The robot may actively explore in order to localize, e.g., by moving to a location with uniquely identifiable landmarks.

Summary.

	HMM, histogram filter	particle filter	Kalman filter
			
	● state	● sampled state	
uncertainty model	arbitrary state distribution	arbitrary state distribution	Gaussian
algorithm	dynamic programming	Monte Carlo sampling	parametric approximation closed form recursion
task	global localization	global localization	tracking
computational efficiency	✗	✓	✓✓
robustness	✓	✓✓	✗

Summary.



- A good choice for the filtering algorithm depends on the size of the state space, the action space, and the observation space. It also depends on the assumption on the scale of uncertainty.
- Histogram filter
  - To apply the histogram filter, we need to represent the state-transition model $T_{s,a,s'} = p(s' \mid s, a)$ and $M_{s,a,z} = p(z \mid a, s)$ explicitly as tables. It thus suffers from the "curse of dimensionality". It is practical only if the state space, the action space, and the observation space are all small.
  - For global localization in a small state space, the histogram filter is more efficient than the particle filter, through the use of dynamic programming.

- Particle filter
  - To overcome the curse of dimensionality, the particle filter uses probabilistic sampling. It approximates an arbitrary state distribution with a sufficiently large number of particles. It is also reasonably efficient.
  - The particle filter is relatively simple to implement and widely used in practice.
  - However, sampling results in statistical variance, manifested as particle depletion. This is a major difficulty and must be addressed carefully for the particle filter to work well in practice.
- The Kalman filter approximates the underlying state distribution as a Gaussian distribution. It is very efficient because of the parametric approximation, but works well only if the Gaussian assumption is verified. It is often used for tracking when the initial robot state is known accurately and the uncertainty remains small.

For robot localization in 2-D environments, the state space size is moderate. Nevertheless, the particle filter is sometimes preferred over the histogram filter, as it does not require explicit discretization of continuous systems and is easier to use.

Required readings.

Supplementary readings.

- [Siegwart, Nourbakhsh & Scaramuzza] Sect 5.6.8

Key concepts.

- Kalman filter