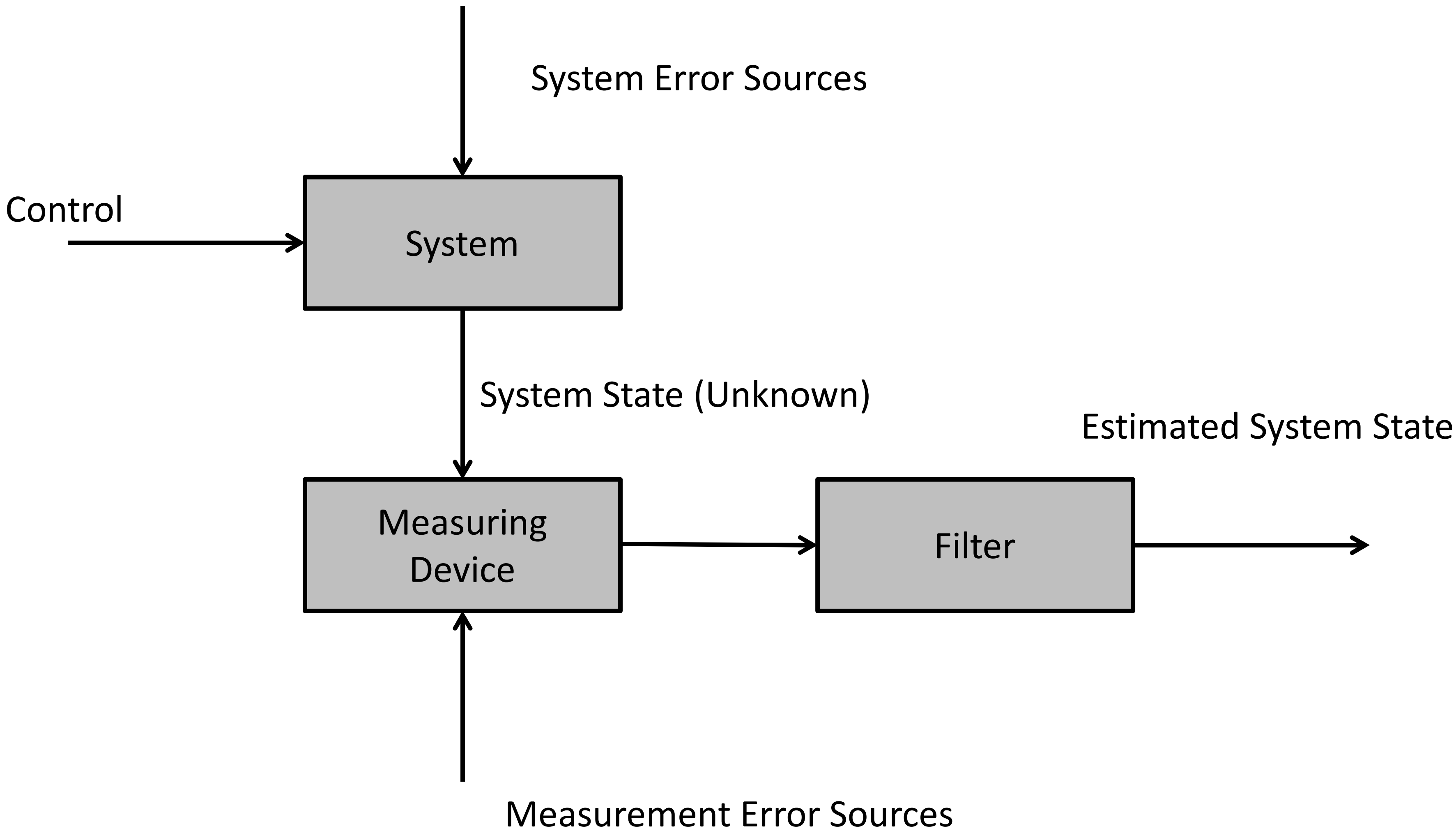
CS 545 Lecture 17 – State Estimation

- Gaussian Distribution
- Bayes Rule
- Kalman Filter
- Extended Kalman Filter
- Bayes Filter (Discrete/Continuous)
- Particle Filter

State Estimation



Models

- Goal: Maintain a *belief* over the states

$$p(s)$$

- State Transition Model

$$p(x_t | x_{t-1}, \mu)$$

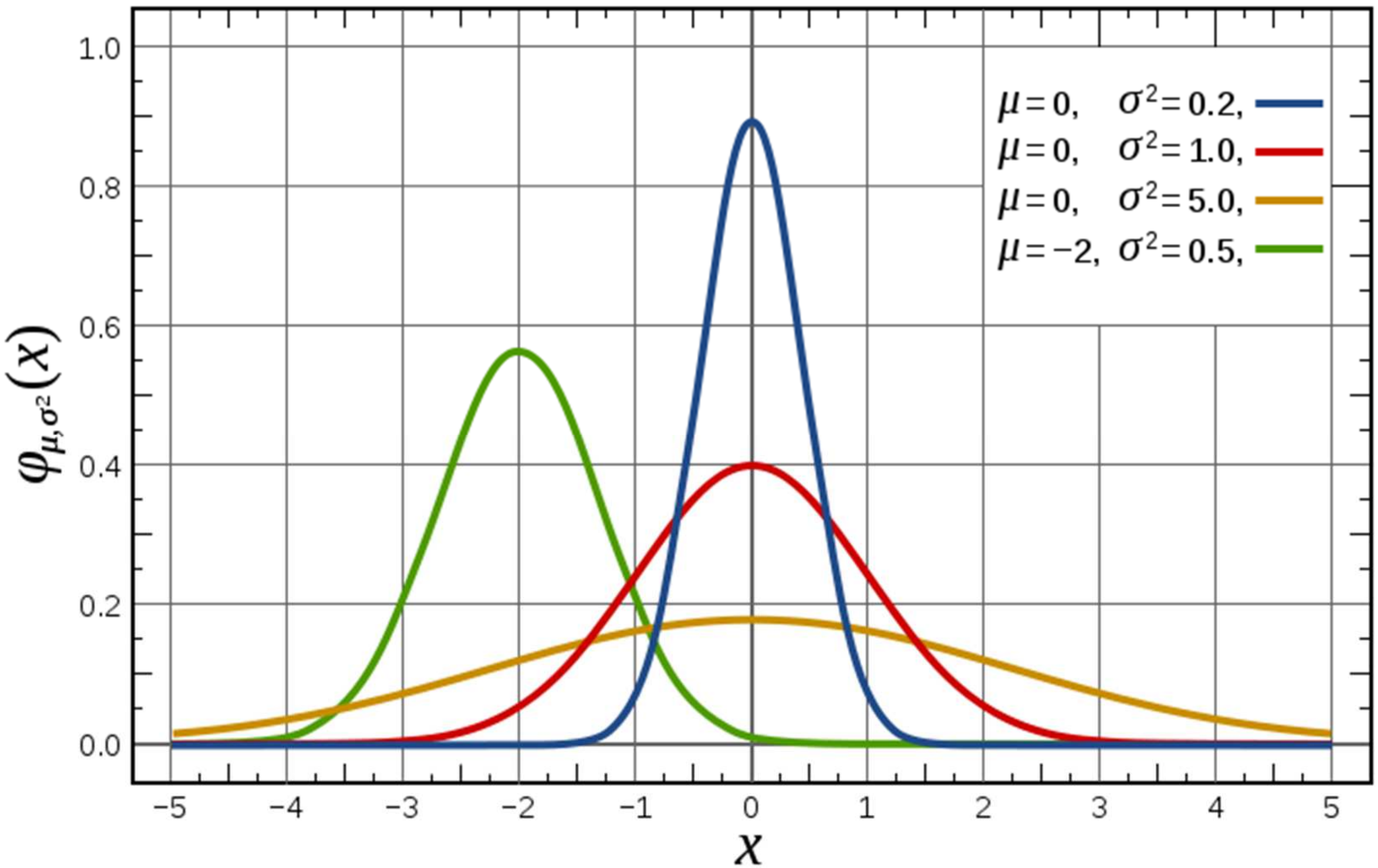
- Observation Model

$$p(z_t | x_t)$$

Gaussian (Normal) Distribution

- $X \sim N(\mu, \sigma^2)$ or $X \sim N(\mu, \Sigma)$
- Mean: $\mu = E[\mathrm{X}] = \frac{1}{n}\sum_{i=1}^{n} x_i$
- Variance:
  $\sigma^2 = E[(X-\mu)^2] = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \mu_i)^2$
- Covariance Matrix:
- $\Sigma_{\mathrm{i,j}} = E\left[(x_i - \mu_i)\left(x_j - \mu_j\right)\right]$
- $P(x) = \frac{1}{\sqrt{|2\pi\Sigma|}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$

Gaussian Distribution Review



Bayes Rule

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

- *P(A|B)* – Posterior (after observed data)
- *P(B|A)* – likelihood
- *P(A)* – prior

Kalman Filter

- Type of Gaussian Filter
- Most popular
- Maintain *belief* as Gaussian distribution
- Thus: estimate μ and Σ at time t

Model

- Plant model

$$x_t = f(x_{t-1}, u_t) + \varepsilon_t$$

- Linear plant model

$$x_t = Ax_{t-1} + Bu_t + \varepsilon_t$$

  - *A* – How system evolves with no input
  - *B* -- How system evolves due to input
  - ε – Gaussian noise, zero mean, Covariance *R*
  - (For simplicity, only use *A* or *B*)
- Observation model

$$z_t = g(x_{t-1}, u_t) + \delta_t$$

- Linear Observation model

$$z_t = Cx_t + \delta_t$$

  - *C* – Measurements as a linear function of state
  - δ_t - Gaussian noise, zero mean, Covariance *Q*

Posterior

- IF

1. State Transition is linear
2. Observation Model is linear
3. Initial belief is Gaussian

- Then posterior will be Gaussian

Example Linear

- Omnidirectional robot
- $A, B = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ $u_t = \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix}$
- $\boldsymbol{x}_t = \begin{bmatrix} x_t + \Delta x_t \\ y_t + \Delta y_t \end{bmatrix}$
- Perfect sensing
- $C = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Example Non-Linear

- Differential drive
- $u_t = \begin{bmatrix} D_t \\ \Delta\theta_t \end{bmatrix}$
- $\boldsymbol{x}_t = \begin{bmatrix} x_t + D_t \cos(\theta_t) \\ y_t + D_t \sin(\theta_t) \\ \theta_t + \Delta\theta_t \end{bmatrix}$

Posteriors

$$P(x_t|u_t, x_{t-1}) = \frac{1}{\sqrt{|2\pi R_t|}} e^{-\frac{1}{2}(x_t - Ax_{t-1} - Bu_t)^T R_t^{-1}(x_t - Ax_{t-1} - Bu_t)}$$

$$P(z_t|x_t) = \frac{1}{\sqrt{|2\pi C_t|}} e^{-\frac{1}{2}(z_t - C_t x_t)^T Q_t^{-1}(z_t - C x_t)}$$

$$p(x_0) = \frac{1}{\sqrt{|2\pi \Sigma_0|}} e^{-\frac{1}{2}(x_o - \mu_0)^T \Sigma_0^{-1}(x_0 - \mu_0)}$$

Kalman Filter

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

Kalman Filter

1. Update μ and Σ based on motion model
2. Calculate Kalman Gain
3. Update μ and Σ based on observation model, weighted by Kalman Gain

- Kalman Gain – Degree to which the measurement is incorporated into the new state
- Innovation – Difference between the actual measurement and the expected measurement

Extended Kalman Filter

- Kalman filter only works for linear models
- Extended Kalman Filter: linearize the model!
- Take Jacobian of model
- Use Jacobian evaluated at current estimated point as motion model
- (need to re-evaluate periodically)

Extended Kalman Filter

- Goal: Maintain μ_t and Σ_t

$$\mu'_t = \nabla A_t \mu_{t\_1} + \nabla B_t u_t$$

$$\Sigma'_t = \nabla A_t \Sigma_{t\_1} \nabla A_t^T + R_t$$

$$K_t = \Sigma'_t C_t^T \left(C_t \Sigma'_t C_t^T + Q_t \right)^{-1}$$

$$\mu_t = \mu'_t + K_t \left(z_t - C_t \mu'_t \right)$$

$$\Sigma_t = \left(I - K_t C_t \right) \Sigma'_t$$

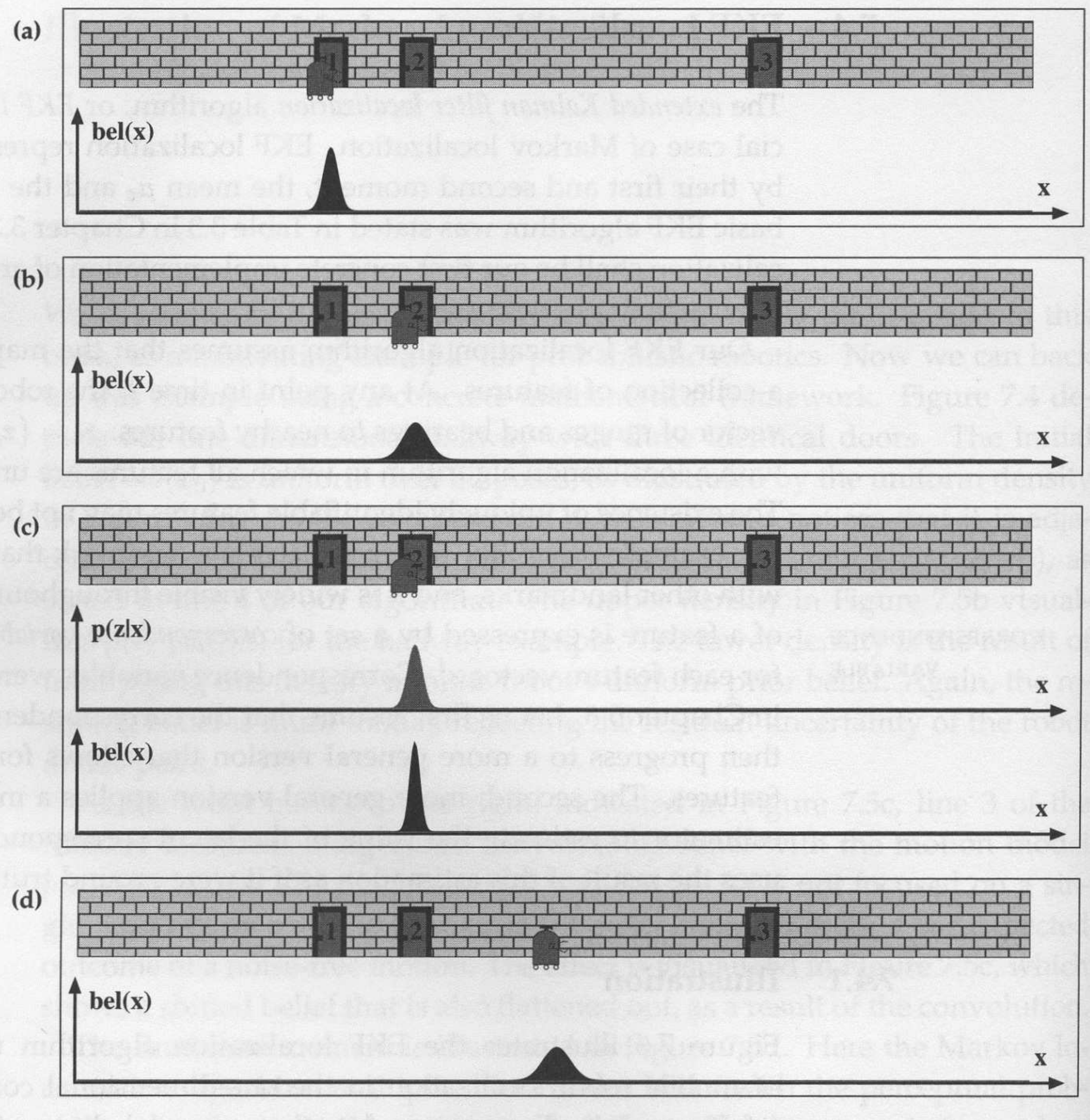


Figure 7.6 Application of the Kalman filter algorithm to mobile robot localization. All densities are represented by unimodal Gaussians.

Characteristics of Kalman Filters

- Working with Gaussians is very efficient
- Very popular, despite limitations
- Unimodal
  - Good when there is a single hypothesis, with some error
- Only works on continuous values

Non-Parametric Approaches

- Kalman filter assumes posterior is Gaussian
- Non-parametric forms do not make assumptions about posterior. No fixed form
- Represent posterior with varying number of parameters
  - Can adjust parameters to change complexity
  - Sometimes can be done dynamically
- Ideal for representing multimodal beliefs
- Bayes Filter – Partition state space.
- Particle Filter – Approximate state space with samples

Discrete Bayes Filter

- Represent space as finite set of states
- Want to maintain discrete probability distribution over each state *x*
- Recursive

- (Forward pass of Hidden Markov Models)
- Can partition continuous state space

Bayes Filter

Algorithm Discrete_Bayes_filter($\{p_{k,t-1}\}, u_t, z_t$**):**

for all k *do*

$$\bar{p}_{k,t} = \sum_i p(X_t = x_k \mid u_t, X_{t-1} = x_i)\; p_{i,t-1}$$

$$p_{k,t} = \eta\; p(z_t \mid X_t = x_k)\; \bar{p}_{k,t}$$

endfor

return $\{p_{k,t}\}$

Particle Filters

- Estimate posterior as a random set of particles drawn from the posterior
- Maintain set of particles
  - $X_t = X_t^1, X_t^2, \ldots, X_t^M$
  - M is usually large (M=1000) but can be altered dynamically based on resources
- At each step, sample set X_t from X_{t-1}

Particle Filters

1. For each particle, sample a new particle based on motion model (move the particles forward)
2. Weigh each particle based on the observation model
3. Resample from the temporary set based on the weights
   – Resample WITH replacement. Can have duplicates
   – This is the "trick" to particle filters

Particle Filter

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

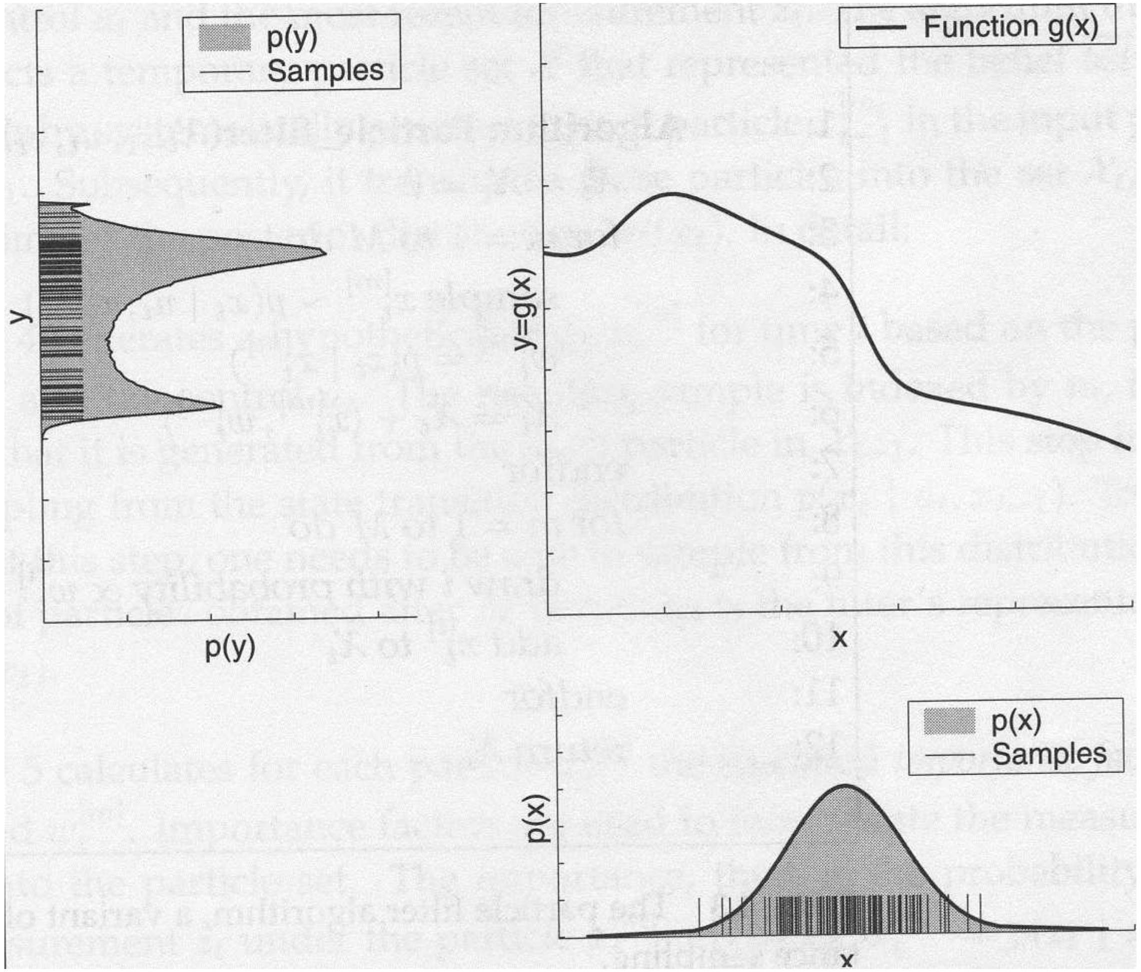
for $m = 1$ to M do

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

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

endfor

return $\mathcal{X}_t$



p(y)
Samples
y
p(y)
Function g(x)
y=g(x)
x
p(x)
Samples
p(x)
x

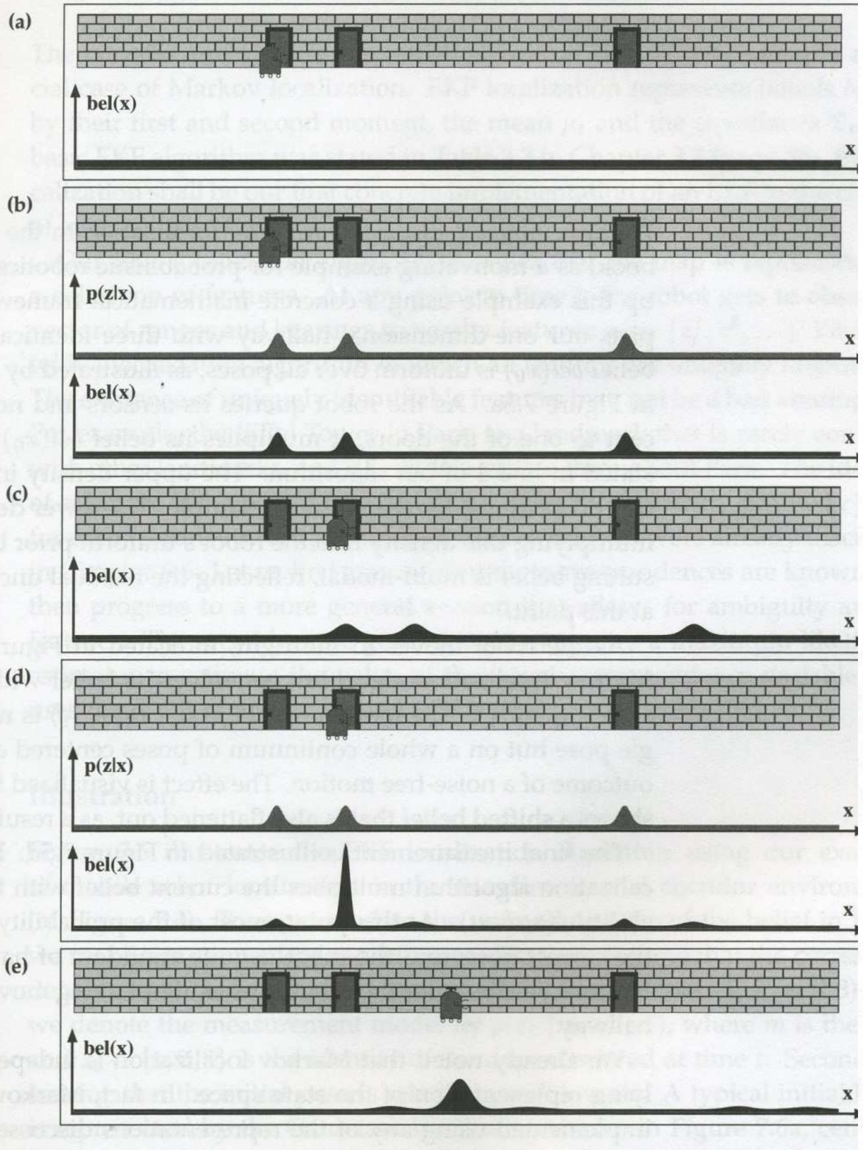


Figure 7.5 Illustration of the Markov localization algorithm. Each picture de
the position of the robot in the hallway and its current belief $bel(x)$. (b) and (c
ditionally depict the observation model $p(z_t \mid x_t)$, which describes the probabil
observing a door at the different locations in the hallway.

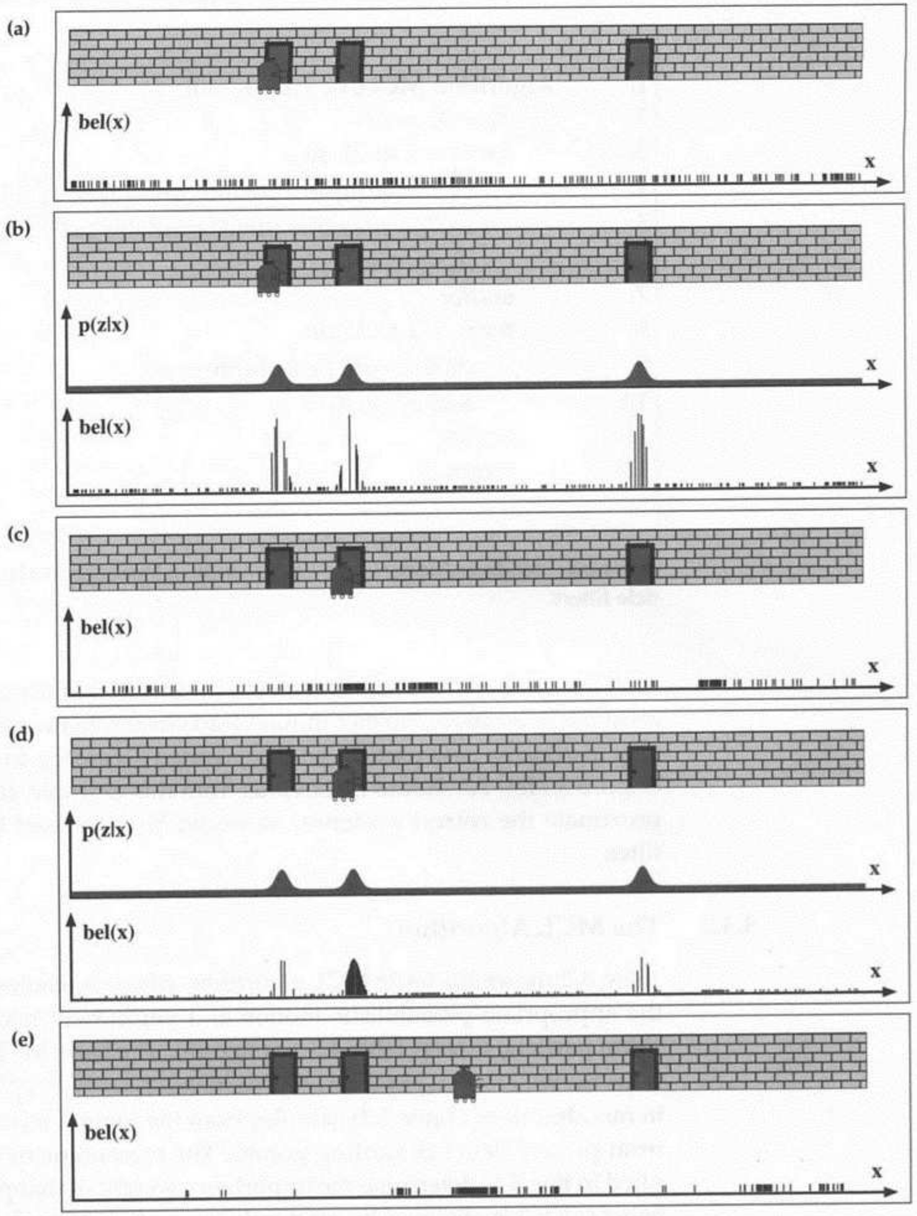


Figure 8.11 Monte Carlo Localization, a particle filter applied to mobile robot localization.